Framework used to Enrich TROPOS Methodology for Requirements of Engineering

Imtiyaz Ahmad[1] Vipin Jaiswal[2]
[1,2]M.Tech Student
[1,2]Department of Computer Science and Engineering
[1,2]Dr. A.P.J. Abdul Kalam Technical University Lucknow, Lucknow Institute of Technology, India

Abstract— This review paper shows the research being done by many researchers and their achievement in the field of TROPOS Methodology. This paper focus on the creation of a methodological roadmap for guiding the application of different goal-oriented strategies. The roadmap provides the infrastructure for modularizing and consequently integrating parts of alternative goal-oriented methods, called method fragments. This allows the adaptation and extension of existing methods so that they can fit to the characteristics of real projects and their contexts. Navigation of a roadmap needs to be facilitated by associated guidelines. These guidelines describe knowledge regarding the situations under which a method fragment is applicable as well as the process that should be followed to apply a particular method fragment. Progress of our work in this area has been reported.

Keywords: Framework, TROPOS Methodology

I. INTRODUCTION

Requirements are the primary purpose behind an item adventure. Each phase in programming improvement like examination, plan, and testing, etc, clearly and in an indirect route depends up upon the necessities. That is the explanation nowadays most of the item improvement affiliations is offering importance to necessities, to the item progression measure and the idea of the item thing. Various item things have failed in view of defenseless requirements planning works out. The reason behind this issue is nonattendance of getting ready in programming progression concerning architects and adventure chiefs, enormous quantities of whom have student programming science degrees, yet are awful at essentials. Thus, to keep up a vital good way from this issue, every item improvement affiliation ought to set up a couple of draftsmen (planners) on essential structure.

Programming Requirements Specification (SRS) infers developing an understanding between the customer and the originator or essential boss. The SRS gives full nuances on the customer wants and we consent to pass on a thing that fulfills them.

Particularly documented/created SRS will reduce tries of the designer, since they will helpfully fathom the essentials and form right and capable code for the endeavor. Richly created SRS will moreover decrease the cost of the endeavor.

Utilizing suitable RE measure models and procedures in an undertaking is the initial move towards expanding the general nature of a product item. Prerequisites are traits that characterize the ability, attributes, execution, and quality attributes of a framework.

To guarantee the nature of programming prerequisites determination, there should be a solid accentuation on actualizing building disciplines into the RE cycle by utilizing different great practices, strategies, and philosophies. This proposal focuses on the most proficient method to inspire, indicate, and approve the necessity of the product framework to be built. These exercises are completed within the restrained called prerequisite designing. In this postulation we learn about the prerequisite designing structure that dependent on the strategy called Tropos.

Tropos is a novel specialist arranged programming building philosophy that it permits to catch what or the how, yet in addition the why a bit of programming is created. It manages all the periods of framework necessity investigation and all the periods of framework structure and execution in a uniform and homogeneous manner.

II. LITERATURE REVIEW

An announcement distinguishing a capacity, physical trademark, or quality factor that limits an item or cycle requirement for which an answer will be sought after. The general target of is to fabricate sound prerequisites determination for the framework to be created. Consequently, as one of the means towards overcoming any issues between RE hypothesis and practice, this examination investigates and distinguishes information about RE measure models and methods.

There are many different definitions for "Requirements".

Acc. to Kotonya and Summerville “Necessity designing is an orderly cycle which incorporates a lot of exercises, for example, prerequisites elicitation, prerequisites investigation and prerequisites arrangement and validation." This cycle is embraced to determine, approve and keep up a framework necessities archive.

Macaulay L.A. characterizes it as "The way toward characterizing what should be structured instead of how it is to be planned".

Somerville I. and Sawyer indicate, "Prerequisites are the depiction of how the framework ought to act, or of a framework property or quality and examined in the RE space alludes to "what a framework ought to do as opposed to how it ought to do it".

Dorfman characterize as "An archive that obviously and accurately portrays every one of the basic necessities (capacities, execution, structure imperatives, and quality traits) of the product and outer interfaces." Each prerequisite is characterized so that its accomplishment can be dispassionately checked by a recommended technique.

Sawyer P. characterizes it as "A property that the product must display with the goal for it to enough plays out its capacity".

The IEEE Standard 729 defines it as:

1) "A condition or capacity required by a client to take care of an issue or accomplish a target".
2) "A condition or capacity that must be met or controlled by a framework – to fulfill an agreement, standard, detail, or other officially forced archive." [6]

In basic words, Requirements are "something that the partners need the product to do or to adjust to". Regularly prerequisites begin dynamically. Consequently, distinguishing this something that the partners need is the most testing part. Recognizing the necessities right the first run through decides the accomplishment of an undertaking.

The total depiction of what the product will do must be recorded. This record is known as the Software Requirements Specifications (SRS) report and ought to be non-questionable, complete, steady, and discernible.
Wieringa R.J. characterizes SRS as "The portrayal of the destinations that an item should fulfill, and the noticeable conduct that the item should have so as to satisfy these goals".

SRS is regularly part of the agreement between the client and seller. SRS helps in assessing cost, arranging group exercises, performing undertakings, and following the group's advancement all through the improvement action.

Frequently the executives neglect to apportion the fundamental assets required for the necessities designing cycle. Designation of adequate time and assets on necessities stage may characterize the achievement or disappointment of a product venture. The more the time spent on the prerequisites building measure, the less time is required for the entire advancement measure since less time is spent on revise. The best possible administration of the prerequisites building cycle can improve and quicken the entire venture advancement.

Prerequisites building is a cycle of finding the requirements of partners and reporting them for examination, correspondence, and execution. Numerous blunders can be distinguished in the prerequisites stage. Davis asserts that fixing of blunders distinguished in later phases of programming advancement is more costly than the underlying stages. On the off chance that blunders are not distinguished in the necessities stage. it prompts wrong item advancement. Wrong necessities can likewise prompt wastage of important assets. Collecting prerequisites is not a simple errand. Prerequisites designing has basic issues which can be because of absence of partner's inclusion in the necessities cycle. Absence of prerequisites the executive's abilities additionally prompts awful necessities designing. Muddled obligations and correspondence among partners can likewise prompt terrible prerequisites building.

Sawyer P. orders prerequisites based on levels at which necessities are planned:

1) User Requirements and
2) System necessities.

Client prerequisites characterize the outcomes and characteristics the client needs and are assembled from the clients of the framework or item. The deliverable toward the finish of the client prerequisites stage is the User Requirements Document (URD) that shapes the premise of programming venture acknowledgment. Framework prerequisites comprise of client necessities, prerequisites of different partners, (for example, administrative specialists) and necessities that do not have a recognizable human source (for example a few administrations of the framework or act to oblige the framework). [5] These are the 'Product Requirements'. Programming necessities give a theoretical model that shows that it takes care of all aspects of the issue. Programming prerequisites characterize what the product must do to accomplish framework necessities, and should all follow back to the framework prerequisites, which made them essential.

Bahill and Dean further classify system requirements into two:

1) Mandatory requirements and
2) Preference requirements.

They characterize obligatory prerequisites as the vital and adequate conditions that a negligible framework must be satisfactory (normally communicated with "will" and "should"). As indicated by them obligatory necessities have the scoring of just pass or fizzle and are not defenseless to compromises between prerequisites. They characterize inclination prerequisites as conditions that would make the client more joyful (frequently communicated with "should" and "need"). As per them the inclination prerequisites use scoring capacities to create figures of legitimacy and are assessed with a multi rules choice method, as none of the doable options would probably improve all the standards (helpless to compromises).

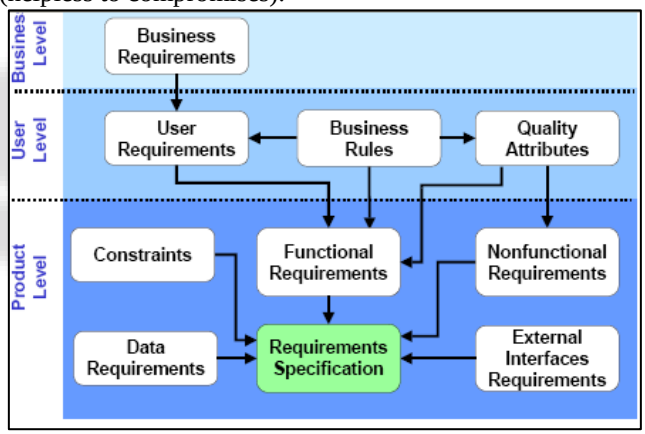

Fig.: 2.1: levels and types of requirements

Summerville and Sawyer identify three categories of requirements:

1) Product requirements – related to performance, reliability, usability, and portability of the system
2) Process requirements – related to standards, programming languages etc. for the system (3) External requirements – related to interoperability, cost etc. of the system. Necessities are best confirmed when communicated quantitatively. Consequently, the arrangement of necessities relies upon the degree of deliberation at which the association sees its task. Figure-2.1 distinguishes one such deliberation of necessities.

Grinter R.E. characterizes necessities the board as "A cycle of surveying the effect of a change before it is made, distinguishing and dealing with the different adaptations of things which a change creates, modifying determined components after source components are changed, and monitoring all the progressions that are made to a framework". [20] This definition restricts the extent of

prerequisites the board to change sway examination, setup the executives and necessities discernibility.

Eberlein A. characterizes necessities the board as "The general cycle of prerequisites advancement including data stockpiling, association, detectability, investigation, representation, and documentation". [21] This definition covers everything identified with prerequisites, aside from setup the executives, as a piece of necessities the board cycle.

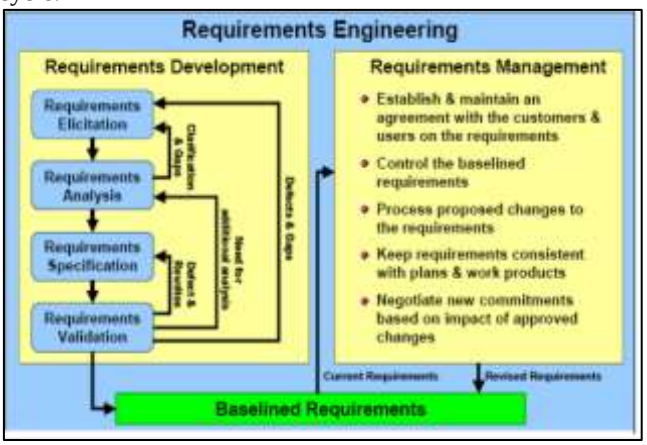

Fig. 2.4: Requirements Engineering Process and Management

Davis and Leffingwell characterize it as "The way toward inspiring, archiving, sorting out, and following evolving necessities, and conveying this data over the task group to set up a typical comprehension (for example an understanding) between the client and the product venture of the client's necessities". [6] Requirements the executives guarantee that iterative and unexpected changes are kept up all through the task lifecycle.

Satta J. characterizes prerequisites the board, as "The way toward catching and conveying the clients' needs to all the colleagues toward the beginning of a task and all through its lifecycle, in an unmistakable and brief way". [22] Requirements the board is a cycle of legitimate correspondence inside the group and it has the capability of significantly expanding the achievement of a task.

Kotonya and Sommerville characterize it as "The way toward overseeing a lot of data and guaranteeing that it is conveyed to the perfect individuals at the perfect time". [15] This is an exceptionally wide definition where the extent of "overseeing a lot of data" is indistinct. "The motivation behind Requirements Management is to build up a typical comprehension between the partners on client's prerequisites that will be tended to by the product venture. It includes setting up and keeping up a concurrence with the client on specialized and non-specialized prerequisites [8]. It helps in recognizing the necessities mistakes and evading superfluous expenses related with actualizing incorrectly prerequisites". CMM characterizes necessities the executives as a cycle of conveying, arranging, performing and following prerequisites. As CMM doesn't characterize how the cycle is accomplished, it doesn't consider change the board, setup the executives, and effect examination as a component of prerequisites the board.

According to AnalystPro for each association endeavoring to reach CMM level 2, or basically attempting to improve the manner in which they work together, necessities the board is one of the fundamental venture the executives controls.[23] However, the extent of prerequisites the executives should be characterized by the association.

A summed-up definition would then be: Prerequisites Management is a cycle of evoking, reporting, dissecting the effect of progress, following, putting away, arranging, envisioning the necessities, distinguishing and dealing with numerous forms of things, and imparting the changed necessities to all the colleagues. Necessities the board is the way toward distinguishing, evoking, reporting, examining, following, organizing and concurring on prerequisites and afterward controlling change and imparting to applicable partners. It is a nonstop cycle all through an undertaking. A prerequisite is an ability to which an undertaking result (item or administration) ought to adjust.

The motivation behind prerequisites the executives is to guarantee the association records, confirms and addresses the issues and desires for its clients and inward or outer partners. Prerequisites the executives starts with the examination and elicitation of the destinations and imperatives of the association. Prerequisites the board further incorporates supporting getting ready for necessities, coordinating necessities and the association for working with them (qualities for prerequisites), just as associations with other data conveying against necessities, and changes for these. Prerequisites the executives includes correspondence between the task colleagues and partners, and acclimation to necessities changes over the span of the undertaking Lamsweerde characterizes an objective as a target that the framework ought to accomplish through participation of specialists in the product to-be and in nature.

Anton states that objectives are elevated level goals of the business, association, or framework; they catch the reasons why a framework is required and direct choices at different levels inside the endeavor. A significant part of prerequisites designing is the examination of non-useful (quality) necessities (NFRs). NFRs are generally spoken to in necessities building models by soft goals. There is no obvious fulfillment condition for a soft goal. Soft goals are identified with the idea of fulfilling [34]. In contrast to customary objectives, soft goals would seldom be able to be supposed to be cultivated or fulfilled. For soft goals one needs to discover arrangements that are "adequate", where soft goals are fulfilled to an adequate degree. Elevated level non-practical necessities are bountiful in associations and every now and again the achievement of frameworks relies upon the fantastic of their non-useful prerequisites.

III. FURTHER WORK

Further work is necessary to fully realize the benefits of the proposed approach. We would like to develop a computerized tool to help the analyst in completing the steps of the methodology. Once a user-friendly tool has been developed, broader use could be achieved. We are also working on verification methods and tools for the ConGolog framework and on integrating them in our methodology. The methodology should also be tested on realistic projects.

Moreover, we would also like to extend the methodology to the design phase of system development. Finally, we would like to refine the methodology to better model agent's mental states — what agents know and want. For this, we will use an extended version of the ConGolog framework that explicitly represents agents' knowledge and goals (using modal operators) and their dynamics, i.e., how they are affected by communication actions (e.g. inform, request, cancel-request, etc.) and perception actions.

The Tropos methodology is not intended for any type of software. For system software (such as a compiler) or embedded software, the operating environment of the system-to-be is engineering artifact, with no identifiable stakeholders. In such cases, traditional software development techniques may be most appropriate. However, a large and growing percentage of software does operate within open, dynamic organizational environments. For such software, the Tropos methodology and others in the same family apply and promise to deliver more robust, reliable and usable software systems. The Tropos methodology in its current form is also not suitable for sophisticated software agents requiring advanced reasoning mechanisms for plans, goals and negotiations. Further extensions will be required, mostly at in detailed design phase, to address this class of software applications.

Much remains to be done to further refine the proposal and validate its usefulness with very large case studies. We are currently working on the development of additional formal analysis techniques for Tropos including goal analysis and social structures engineering. We are also developing tools that support different phases of the methodology.

REFERENCES

[1] A. Fuxman, L. Liu, M. Pistore, M. Roveri, and J. Mylopoulos. Specifying and analyzing early requirements in Tropos: Some experimental results. In Pro- ceedings of the 11th IEEE International Requirements Engineering Conference, Monterey Bay, California USA, September 2003. ACM-Press.
[2] A. Fuxman, M. Pistore, J. Mylopoulos, and P. Traverso. Model checking early requirements specifications in Tropos. In IEEE Int. Symposium on Re- quirements Engineering, pages 174–181, Toronto (CA), August 2001. IEEE Computer Society.
[3] E. Gamma, R. Helm, J. Johnson, and J. Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995.
[4] P. Giorgini, E. Nicchiarelli, J. Mylopoulos, and R. Sebastiani. Reasoning with Goal Models. In Proc. Int. Conference of Conceptual Modeling – ER'02, LNCS, Tampere, Finland, October 2002. Springer.
[5] S. Hayden, C. Carrick, and Q. Yang. Architectural design patterns for mul- tiagent coordination. In Proc. of the 3rd Int. Conf. on Autonomous Agents, Agents'99, Seattle, USA, May 1999.
[6] N. R. Jennings. On agent-based software engineering. Artificial Intelligence, 117(2), 2000.
[7] M. Kolp, P. Giorgini, and J. Mylopoulos. A goal-based organizational per- spective on multi-agents architectures. In Proc. of the 8th Int. Workshop on Intelligent Agents: Agent Theories, Architectures, and Languages, ATAL'01, Seattle, USA, August 2001.
[8] Y. Labrou, T. Finin, and Y. Peng. The current landscape of agent communi- cation languages. Intelligent Systems, 14(2):45–52, 1999.
[9] A. V. Lamsweerde. Goal-oriented requirements engineering: A guided tour. In IEEE Int. Symposium on Requirements Engineering, pages 249–261, Toronto (CA), August 2001. IEEE Computer Society.
[10] H. Mintzberg. Structure in fives: designing effective organizations. Prentice- Hall, 1992.
[11] J. Morabito, I. Sack, and A. Bhate.Organization modeling: innovative architectures for the 21st century. Prentice Hall, 1999.
[12] N. Nilsson. Problem Solving Methods in Artificial Intelligence. McGraw Hill, 1971.
[13] W.R. Scott. Organizations: rational, natural, and open systems. Prentice Hall, 1998.
[14] M. Shaw and D. Garlan. Software Architecture: Perspectives on an Emerging Discipline. Prentice Hall, 1996.
[15] E. Yu. Modelling Strategic Relationships for Process Reengineering. PhD thesis, University of Toronto, Department of Computer Science, 1995

