A Reliable Framework to Power Tropos Methodology in Engineering Requirements

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 research 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: Requirement Engineering Framework (REF)

I. INTRODUCTION

Prerequisites are the main thrust behind a product venture. Each stage in programming improvement like investigation, plan, and testing, and so on, straightforwardly and in a roundabout way relies up upon the necessities. That is the reason these days the greater part of the product improvement associations is offering significance to necessities, to the product advancement measure and the nature of the product item. Numerous product items have fizzled because of helpless prerequisites designing exercises. The purpose behind this issue is absence of preparing in programming advancement with respect to engineers and venture directors, huge numbers of whom have undergrad software engineering degrees, yet are bad at prerequisites. So, to maintain a strategic distance from this issue, each product improvement association should prepare a few architects (designers) on prerequisite building.

Programming Requirements Specification (SRS) implies building up an agreement between the client and the designer or prerequisite supervisor. The SRS gives full subtleties on the client desires and we agree to convey an item that satisfies them. Very much archived/composed SRS will lessen endeavors of the developer, since they will handily comprehend the prerequisites and compose right and proficient code for the undertaking. Elegantly composed SRS will likewise diminish the expense of the venture.

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. OBJECTIVE OF RESEARCH

- To analyze existing requirement engineering process models.
- To develop a framework for requirement engineering.
- To improve quality of requirement engineering.

III. METHODOLOGY

In the ongoing years, the prominence of objective arranged prerequisites building approaches has expanded drastically. The principle purpose behind this is the deficiency of the conventional framework's examination approaches [24] when managing increasingly more unpredictable programming frameworks. At the prerequisites level, these methodologies treat necessities as comprising just of cycles and information and don't catch the basis for the product frameworks, consequently making it hard to comprehend prerequisites as for some elevated level worries in the difficult space. Most strategies center around demonstrating and detail of the product alone. Hence, they need uphold for thinking about the composite framework involved the framework to-be and its condition. Be that as it may, erroneous suspicions about the earth of a product framework is known be liable for some blunders in necessities details [25]. Non-useful prerequisites are additionally by and large left outside of necessities details.

Also, customary demonstrating and examination procedures don't permit elective framework setups where pretty much usefulness is mechanized or various tasks of duty are investigated, and so on to be spoken to and looked at. Objective Situated Requirements Engineering (GORE) endeavors to tackle these and other significant issues. Note that objective situated prerequisites elaboration measure closes where most customary determination methods would begin [25]. GORE centers around the exercises that go before the plan of programming framework necessities.

The accompanying fundamental exercises are regularly present in GORE draws near: objective elicitation, objective refinement and different sorts of objective

investigation, and the task of duty regarding objectives to operators.

IV. PROPOSED WORK

Necessities building is the part of programming designing worried about this present reality objectives for, elements of, and requirements on programming frameworks. It is likewise worried about the relationship of these variables to exact determinations of programming conduct, and to their development after some time and across programming families. Prerequisites must be resolved and consented to by the clients, clients, and providers of a product item before the product can be fabricated. Prerequisites Engineering (RE) measure demonstrating has become a significant region of enthusiasm for the product designing field of study which is a fundamental piece of the improvement of programming arrangements. It is a restrained way to deal with overseeing undertakings and duties in an improvement association.

RE process modeling is important because it gives an understanding of how the organization works and enables the design of the key practices of the identified organizational tasks and presents them in a form that is suitable or a wide range of projects and organizations. RE involves requirements negotiation as an activity during software development. In this process, that is requirements negotiation, there are a lot of software requirements that are vague or not clear that stakeholders put forward. Most stakeholders know what they want but do not know what they need. For example, some stakeholders find it difficult to lay out requirements to the developers therefore this leads to misinterpretation and eventually errors which end up delaying the project.

Prerequisites building (RE) has advanced into a key movement in the field of programming designing. RE is both a hierarchical action and a venture action. It is an authoritative movement regarding choosing what kind of prerequisites will go into items, and the last necessities that will be delivered. It is likewise a task movement with regards to actualizing them. The way toward building up the administrations that the client requires from a framework and the limitations under which it works and is created. The necessities themselves are the depictions of the framework administrations and imperatives that are produced during the prerequisites designing cycle

The essential proportion of accomplishment of a product framework is how much it meets the reason for which it was expected. Prerequisite Engineering (RE) is the way toward finding that reason. Different prerequisites building systems are objective situated, perspective arranged, situation based, and specialist situated RE structures. The RE structures dependent on the idea of office(for example Specialist, objective and purposeful reliance) have been perceived as having the capacity to lead towards a more homogeneous and normal programming designing cycle, running from elevated level association needs to framework organization. Objectives are significant Requirements Engineering measures since it features the 'why' just as the 'what' a framework needs to do.

Basically, goal is an objective the system under consideration should achieve. Goal formulations thus refer to intended properties to be ensured; they are statements as opposed to indicative ones, and bounded by the subject matter. Goals may be formulated at different levels of abstraction, ranging from high-level, strategic concerns (such as “serve more passengers” for a train transportation system or “provide ubiquitous cash service” for an ATM network system) to low-level, technical concerns (such as “acceleration command delivered on time” for a train transportation system or “card kept after 3 wrong password entries” for an ATM system).

Objectives additionally spread various kinds of concerns: utilitarian concerns related with the administrations to be given, and nonfunctional concerns related with nature of administration –, for example, wellbeing, security, precision, execution, etc. The framework which an objective alludes to might be the current one or the framework to-be; them two are associated with the RE cycle. Elevated level objectives frequently allude to the two frameworks. The framework to-be is generally composite; it contains both the product and its condition, and is made of dynamic parts, for example, people, gadgets and programming. One of the significant results of the RE cycle is the choice on what parts of the framework will be robotized and what parts won't. An objective under obligation of a solitary operator in the product to-be turns into a prerequisite though an objective under duty of a solitary specialist in nature of the product to-be turns into a suspicion.

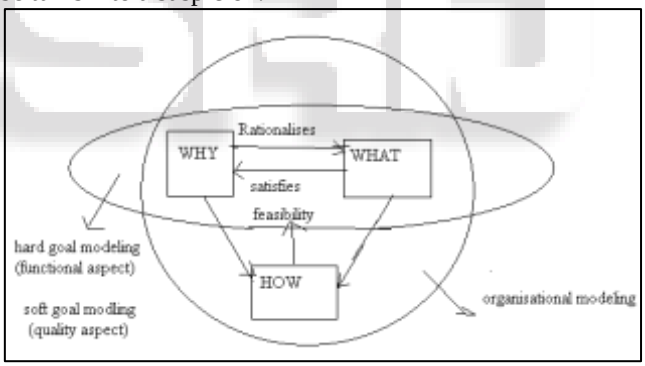

Fig. 4.1: Proposed Requirement Engineering Framework (REF)

Tropos [Trop] is a specialist situated programming advancement system which is established on the ideas of objective based prerequisites received from the i* [I Star] and GRL (Goal-arranged Requirements Language) [GRL] [41]. Tropos-explicit demonstrating bargains essential with displaying of necessities and deliberate parts of the operator framework, from the early prerequisites investigation to the late plan. Prerequisites building began with the investigation of what the framework ought to do, for example late-stage necessities investigation, which centers around the determination of prerequisites, their fulfillment, consistency, mechanized confirmation, and so on. Beginning stage prerequisites investigation centers around why the framework must be grown, how the ideal framework will meet its objectives.

As per the idea of an objective, a qualification is made between hard-objectives and delicate objectives.

Delicate objectives are utilized to indicate, at a subjective level, not pointedly cut destinations, the exact meaning of which require to create, while hard-objectives unmistakably characterize a state/focus on, an entertainer wants to reach. [50]. For instance, .having a visa conveyed is plainly a hard-objective, while .having it conveyed rapidly is a delicate objective, being the thought of .rapidly, exceptionally emotional.

Delicate objectives assume a focal job, giving an orderly and composed method of dealing with non-utilitarian prerequisites (or quality properties, characteristics, or all the more conversationally "-capacities") à limitations in operational terms, or fit rules for appraisal purposes.

As indicated by this proposed structure the prerequisites designing characterize the product item.

A. Improve the quality

Requirements engineering (RE) is concerned with the identification of the goals to be achieved by the envisioned system, the operationalization of such goals into services and constraints, and the assignment of responsibilities for the resulting requirements to agents such as humans, devices, and software.

The processes involved in RE include domain analysis, elicitation, specification, assessment, negotiation, documentation, and evolution.

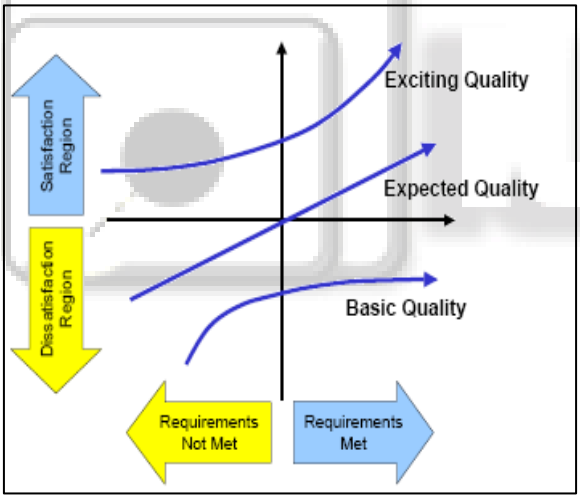

Fig. 5.1: Relationship between customer satisfaction and quality requirements

The customer will be dissatisfied (and go buy a car somewhere else) if their explicit requirements are not met. The customer's satisfaction increases as more of their explicit requirements are met. When enough of their explicit requirements are met, the customer shifts from being dissatisfied with the product to being a satisfied customer. There is a basic level of quality requirements that a customer expects the product to have. These are requirements that are assumed by the customer and are typically not explicitly stated. This level of requirements does not satisfy the customer. Note that the entire basic line is in the dissatisfaction region. Absence of this level of quality requirements will increase a customer's dissatisfaction. Exciting quality is the innovative requirements level and represents unexpected items. These are items that customers do not even know they want, but they love them when they see them. However, it is easy to miss both the basic and exciting quality requirements if they do not do a thorough job of detailed requirements elicitation and analysis. In addition, if practitioners miss a stakeholder group or if they do not get the users involved in the requirements process, they can end up with gaps even in their expected requirements.

In software-intensive systems, the achievement of qualities—such as performance, availability, security, and modifiability—is dependent on the software architecture. Quality should be measured from the early stages of software building or else the development can end up with software that fulfills the requirements but fails to satisfy the customer.

High Quality Requirement documents are a must for successful software projects. Requirements are the initial step in every project, and therefore it is necessary to collect good requirement as early as possible in the project life cycle. All process phases directly and indirectly depend on the requirement document. As Tropos methodology is that it allows to capture not only what or the how, but also the why a piece of software is developed. The 'why' question helps to discover the objectives and rationale behind the goals which in fact identify the higher goals. It also helps to improve the quality.

V. CONCLUSION

In this research paper, we presented the overview of requirement engineering and introduced an agent-oriented Requirements Engineering Framework (REF), explicitly designed to support the analysts in transforming high-level organizational needs into system requirements, while redesigning the organizational structure itself. The former investigates the use of goal analysis in terms of different RE process (requirements elicitation, requirement analysis, requirement validation. On the other hand, compares different goal-oriented methodologies on the basis of their goal modelling and specification approaches. In addition it provides an overview of different goal modelling strategies (goal refinement, goal decomposition, analogical reuse, goal operationalization, goal conflict management, and selection between alternatives). Rather than providing a comprehensive framework for analyzing the contribution of alternative approaches, the objective of these works has been mainly to stress the significance of goal concepts in RE and to draw the attention of the research community to goal-driven RE. The advantage of the analysis framework presented in this chapter is that it provides an overall picture of goal-oriented research, enabling researchers to identify possible extensions (e.g. the need for methodological support). It also assists requirements engineers to understand and accordingly select the best fit for usage goal modelling method.

Stakeholder goals and their role in defining and solving design problems are topics of longstanding interest in the field of requirements engineering. Goal analysis approaches emphasizes the use of the goal notion in order to understand or describe aspects of the real world, in an attempt to rationally find better ways of coping with the complexity of human affairs. Much of the work has been

confined to the research domain and it is indeed a challenge to researchers, method engineers and practitioners to facilitate the transition of these techniques from the research laboratory to practical use. We have presented a comparison of three prominent methodologies. Overall, all three methodologies provide a reasonable support for basic agent-oriented concepts such as autonomy, mental attitudes, pro-activeness, reactivates etc. They all are also regarded by their developers and the students as clearly agent oriented. In addition, the notation of the three methodologies is generally good. Regarding the process, all the methodologies provide examples and heuristics to assist developers from requirements gathering to detailed design. Implementation was supported to some degree by all methodologies whereas testing/debugging and maintenance are not clearly well-supported by any methodology. Additionally, some important software engineering issues such as quality assurance, estimating guidelines and supporting management decisions are not supported by any of the methodologies.

There are other notable proposals for agent-oriented software development, such proposals are mostly extensions to known object-oriented engineering methodologies. Moreover, all these proposals focus on design –as opposed to requirements analysis -- and are therefore considerably narrower in scope than Tropos. Indeed, Tropos proposes to adopt the same concepts, inspired by requirements modeling research, for describing requirements and system design models in order to narrow the semantic gap between them. The architecture and software design models produced within our framework are intentional in the sense that system actors have associated goals that are supposed to be fulfilled. They are also social in the sense that each component (actor) has obligations/expectations towards/from other components. Obviously, such models are best suited for cooperative, dynamic and distributed applications.

Our current research 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 particular process that should be followed in order to apply a particular method fragment. Progress of our work in this area has been reported.

REFERENCES

[1] Y. Aridor and D.B. Lange. Agent design patterns: Elements of agent application design. In Proc. of the 2nd Int. Conf. on Autonomous Agents, Agents'98, pages 108–115, St. Paul, USA, May 1998.
[2] J. Castro, M. Kolp, and J. Mylopoulos. Towards Requirements-Driven In- formation Systems Engineering: The Tropos Project. Information Systems. Elsevier, Amsterdam, the Netherlands, (to appear).
[3] L.K. Chung, B. Nixon, E. Yu, and J. Mylopoulos. Non-Functional Requirements in Software Engineering. Kluwer Publishing, 2000.
[4] Cimatti, E.M. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella. NuSMV.
[5] E. Clarke, O.Grumberg, and D.Peled. Model Checking. MIT Press, 1999.
[6] Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. Science of Computer Programming, 20(1–2):3–50, 1993.
[7] T.T. Do, M. Kolp, and A. Pirotte. Social patterns for designing multi-agent sys- tems. In Proc. of the 15th Int. Conf. on Software Engineering and Knowledge Engineering, SEKE'03, 2003.
[8] Fuxman. Formal analysis of early requirements specifications. Master's thesis, University of Toronto, 2001

