Abstract— A major component of the software development process is requirement engineering. This takes into account discovering, obtaining and validating the different requirements needed to be met by the system. Most industries today face problems in developing and improving their software's, and the concurring expenses involved. This paper primarily deals with the first step of requirement engineering, i.e., requirement elicitation and requirements prioritization. Requirement elicitation is of the most important and critical intensive activities of the development of any software. Requirements prioritization basically aims at identifying the most important requirements from the clients and as well from the system. There are many requirement elicitation and requirement prioritization techniques available in the market; still there is lack of proof of which technique to be preferred. The main reasons could be the differences in measurements, context of variables and usage of different data sets. In this paper, the area of requirements elicitation and prioritization has been systematically reviewed in order to assess what evidence regarding different techniques exist. It seeks to develop a new and concise flow model illustrating an efficient way of gathering raw data from users, stakeholders and customers, as well as technical experts by monitoring usage and social networking techniques.

Index Terms— Elicitation, Prioritization, Software Engineering, Brainstorming

I. INTRODUCTION

Searching is considered a very important Software plays an integral part in the functioning of the modern world. Infrastructures and electrical products include a computer and controlling software. Financial systems, transportation systems and industrial manufacturing are all software intensive. Entertainment, including music, games, videos and movies are all computerized. “Software engineering does not yet have a widely recognized and accepted set of research paradigms in the way that other parts of computer science do”[1]. Software systems are abstract and nonrepresentational. There are no physical laws or manufacturing processes that govern software engineering. New software or enhanced and superior versions of previously existing software incessantly keep making their way into the market. In such a situation, collection of data from users, stakeholders and clients that meets their requirements is imperative. Requirements engineering refers to the process of accumulating, formulating, and maintain software requirements. Changes to software requirements are inevitable during most stages of the software development lifecycle [3]. Ever-changing
software requirements is one of the major problem areas affecting software productivity. There are three
generic methods of elicitation, ie, viewpoints, brainstorming and interviews [4].

A. Viewpoints

One methodology adopted for collection of requirements for software development is Viewpoints. Viewpoint
helps to structure the requirement elicitation and analysis process. Formally defined [2] “A viewpoint is an
encapsulation of partial information about a system’s requirements from different sources. Information from
different viewpoints must be integrated to form the final system specification”. The different sources from
which viewpoints are collected maybe stakeholders, consumers and select technical experts. All of the above
mentioned parties may have different expectations and stipulations from the software [5]. Hence, it is crucial
to take their requirements into consideration before drafting the formal requirements document so as to cover
all facets of the software being developed. That being said, there is some discrepancy associated with
viewpoints. Despite there being some guidelines there is no structured process associated with viewpoint
collection and resolution. Software engineers find it difficult to resolves overlapping requirements and this
may lead to ambiguities [6].

B. Interviews

Interviewing the stakeholders is another way to collect requirements from them. The software engineer puts
forth questions to the stakeholders regarding usage of the software, their expectations and constraints. Based
on the answers to these questions the requirements are drafted. There are two types of interviews-structured
and open ended interview. If the interview has a pre-defined scheme and a fixed set of questions it is called a
structured interview [7]. Open ended interviews have no fixed design and a variety of areas are covered in the
questions to develop better insight into the system requirements. Interviews are not an efficient way of
collecting requirements. It is not always possible to cover all the aspects of the software [8], some areas are
bound to get left out and will remain unexplored. Also, the answers put forth by the stakeholders also depend
to an extent on the manner in which the interview is conducted and the interviewing style. These factors
might lead to inconsistency in the final draft.

C. Brainstorming

Stakeholders or their representatives congregate to discuss and develop a large range of ideas and elicit
knowledge. Brainstorming encourages out of the box thinking. It is a group discussion with the requirements
ingineer playing the role of a mediator [10]. Each discussion has a specific focus and the participants discuss
that topic only. Only a few people actively participate in these meetings and some participants provide
feedback on the feasibility of the ideas discussed [11]. The result of this approach is formulation of
innovative ideas .Brainstorming however has its own weakness, a major one being the inability to resolve
issues and reach a consensus when there is a difference of opinion. An argumentation process may not
always result in an outcome. In order to overcome the aforementioned problems, we have come up with an
elicitation model which combines ethnography techniques as well as social networking in order to produce
better quality products [12].

II. LITERATURE SURVEY

A. Requirements Elicitation techniques

Table 1 below shows the various Requirements Elicitation techniques along with their merits and demerits
[13]. There are a many techniques available that can be used for elicit requirements from the system, Clients
or Domain experts. The approach taken by a software engineer is not restricted to one defined technique.
Type of application, available resources in the organization, and individual preference order will all play a
crucial role in determining a proper defined approach [14].

B. Requirements Prioritization techniques

After requirements are defined and identified, they need to be prioritized based on the system requirement.
Requirement prioritization process is used to identify which customer requirement of a software development
project should be embedded in a certain release of the software, for this purpose different techniques of
prioritization are used [1].
Table 2 below shows the various Requirements Prioritization techniques along with their merits and demerits
[15].

637
III. PROPOSED SYSTEM

This paper is divided into three major sections. The first section will introduce and describe a social networking tool called the “Idea Station” which may be a cellphone app, or a website, made available to stakeholders, select users and target customers [16]. This tool’s primary functionality is to generate ideas for the development of new software’s [9]. The second section will describe an ethnographic model which will help regulate and improve an existing software product. Finally, in the third section we take a case study to illustrate the use of these tools shown in fig 1.

Algorithm:

**Step 1:** Open the Idea Station to target customers, stakeholders and users.

**Step 2:** Select the popular and beneficial ideas which can be implemented

**Step 3:** Develop the software and release it to technical users

**Step 4:** Monitor its usage using tool X. If any improvements can be made, move to step 5. Otherwise, move to step 6.

**Step 5:** Update the software in a regular fashion. Move to step 4.

**Step 6:** With no further improvements, the software can be released.

![Fig 1: Proposed System](image)

A. Idea Station

The Idea Station will be a social networking platform open only to users, stakeholders and select customers [18]. This tool will enable the following features:

1) Share ideas: Members will be able to publish their ideas on the main homepage.

2) Like and comment on published ideas: Every member can like and comment on all ideas visible to them.

3) Create groups: Two or more members can form groups for discussion, and the conversations held in a group are private to the group.

4) Tags: Every idea published should have one or more tag/label which identifies which category the idea belongs to.

5) Search: A member can search for ideas based on tags/labels.

The administration monitoring the tool may then select popular ideas to develop novel software’s based on the input from the members of the “idea station” [19].

B. Tool BackApp

The main focus of this tool is to monitor the usage of the software in the background [20]. The following flow model depicts the functioning of BackApp.

**Monitor Usage frequency:** The usage of each feature of the software is plotted on a graph.

**Error Reports:** If a feature malfunctions, the user will be given the option to send an error report to the developers.

“Fly on the wall”: In case of large errors encountered, the user has the flexibility to turn on the ‘fly on the wall’ option shown fig 2. This option will record the user’s computer screen, and also enables the user to physically voice the problems they are encountering. This audiovisual recording will be stored in a database maintained exclusively for this technique [21].
**Backup algorithm:**

**Step 1:** Release software for technical users.

**Step 2:** Collect usage frequency and error reports on the various features of the software periodically.

**Step 3:** If usage frequency is high and error reports are low then Fine tune the feature Charge money for the feature

**Else if** usage frequency is high and error reports are high then administer the “fly on the wall” technique

**Else if** usage frequency is low and error reports are low then Update the feature or consider its deletion

Else if usage frequency is low and error reports are high then delete or replace the feature.

**Step 4:** Update the software. If changes have been made, go to step 2.

**Step 5:** Software can be released in the market.

---

*Fig 2: System Architecture of a proposed System*
<table>
<thead>
<tr>
<th>SL No</th>
<th>Method</th>
<th>Description</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Prototyping</td>
<td>Prototype is a version of a product launched into market to provide the service to the customers. Prototyping is used to provide a version of the software and which is not final so the customer can gain the experience and also may be able to provide other requirements that need to be implemented in the next prototyping. The response of the user is in the form of a feedback which is recorded as like requirements of the system [22].</td>
<td>Prototyping provides the detail information by investing each and every prototype by the customer [17]. Prototypes are mostly used in conjunction with other elicitation techniques such as interviews. Prototypes useful when developing human computer GUI interfaces. Prototypes provide a good chance to the stakeholders and to be involved in the requirements engineering. The technique is extremely helpful developing new systems for entirely new applications [23].</td>
<td>In many cases prototypes are expensive to produce in terms of time and cost. A great problem for prototyping is that the user often resists making changes if once they are made.</td>
</tr>
<tr>
<td>2</td>
<td>Workshop, focus groups</td>
<td>Stakeholder representatives get together for a short but intensely focused period to create or review high-level features of the desired products.</td>
<td>This technique is very much effective to resolve the conflicts among customers in order to bring them to one table. Each and every aspect of requirements is discussed and proper suggestions are given using group work [24]. The stakeholders provide the direct remarks about the software requirements. Stakeholders work in the environment. Group work provides the remarkable feedback.</td>
<td>This technique needs a lot of effort as compared the other requirements engineering techniques. Sometimes all the stakeholders can join at the same time as it may be possible that they may be busy in other tasks. Group work is less effective in the highly political tense situation.</td>
</tr>
<tr>
<td>3</td>
<td>Interviews</td>
<td>Analyst discusses the desired product with different groups of people and builds up an understanding of their requirements. If the interview is conducted with pre-defined agenda and questions, it is called structured interview; otherwise, it is an open-ended interview.</td>
<td>Collecting the rich and detailed data. Collecting information to design a survey or other usability activity. Getting a holistic view of the whole system.</td>
<td>Collecting data from large samples or people. When it need to collect the data very rapidly.</td>
</tr>
<tr>
<td>4</td>
<td>Brainstorming</td>
<td>Stakeholder representatives gather together and rapidly develop a large and broad list of ideas. It encourages “out-of-the-box” thinking without normal constraints, and involves both idea generation and idea reduction.</td>
<td>Brainstorming is mostly used for the innovative sort of projects where each participant provides his or her own ideas after their personal research about the project to be started. This technique is often used make the key decisions about the requirements of the project. It promotes free thinking and expression of ideas. Brainstorming provides the innovative ideas about the project to be developed.</td>
<td>Brainstorming is seriously affected by exploring the critique ideas. Brainstorming is not used to resolve the major issues.</td>
</tr>
<tr>
<td>5</td>
<td>Scenarios, passive Storyboards</td>
<td>It is an interaction session to describe a sequence of actions and events for a specific case of some generic task which the system is intended to accomplish. Clarified system requirements related to procedures and data flows of a task. In a highly uncertain situation, an effective and relatively inexpensive way to develop an initial set of requirements [25].</td>
<td>Because storyboards exist independently of the software system they describe, they have many advantages over regular prototypes. They cannot crash, are very easy to share with large groups, and do not give the false impression that the system is already developed. Additionally, feedback is easier to accommodate.</td>
<td>One of the biggest problems with storyboards is that they can become outdated very quickly. User interfaces originally defined often change over time, and that creates a Maintenance burden.</td>
</tr>
<tr>
<td>SL No</td>
<td>Method</td>
<td>Description</td>
<td>Advantages</td>
<td>Disadvantages</td>
</tr>
<tr>
<td>-------</td>
<td>-------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1</td>
<td>Binary search tree (BST)</td>
<td>In BST, a requirement from the set of requirements is selected as the root node. Then, a binary tree is constructed by inserting less important requirements to the left and more important ones to the right of the tree. A prioritized list of requirements is generated by traversing the BST in order. The output is a prioritized list of requirements with the most important requirements at the start of the list, and the least important ones at the end.</td>
<td>This method is simple to implement</td>
<td>Provides only a simple ranking of requirements as no priority values are assigned to the requirements.</td>
</tr>
<tr>
<td>2</td>
<td>Win-win approach</td>
<td>In the win-win approach proposed by Boehm, stakeholders negotiate to resolve disagreements about candidate requirements. Using this approach, each stakeholder ranks the requirements privately before negotiations start. They also consider the requirements they are willing to give up on. Stakeholders then work collaboratively to forge an agreement through identifying conflicts and negotiating a solution.</td>
<td>Win-win negotiations encourage stakeholders to focus on their interest rather than positions, negotiate towards achieving mutual gain, and use objective criteria to priorities requirements.</td>
<td>The approach is labor intensive, particularly in large projects.</td>
</tr>
<tr>
<td>3</td>
<td>Hierarchical cumulative voting (HCV)</td>
<td>Enables prioritizations to be performed at different levels of a hierarchy. Stakeholders perform prioritization using 100-point test within each prioritization block. The intermediate priorities for the requirements are calculated based on the characteristics of the requirements hierarchy. Final priorities are calculated for all requirements at the level of interest through normalization. If several stakeholders have prioritized the requirements, their individual results are then weighted and combined.</td>
<td>The hierarchical prioritization in HCV makes it easier for the stakeholders to keep an overview of all the requirements</td>
<td>The prioritizations need to be interpreted in a rational way as stakeholders can easily play around with the numbers.</td>
</tr>
<tr>
<td>4</td>
<td>100-point test</td>
<td>Each stakeholder is given 100 points that they can distribute as they desire among the requirements [24]. Requirements that are more important to a stakeholder are given more points. Requirements are then prioritized based on the total points allocated to them [25].</td>
<td>100-point test incorporates the concept of constraint in the stakeholder’s prioritization by giving each of them a limited number of points.</td>
<td>It can be easily manipulated by stakeholders seeking to accomplish their own objectives [23]. For example, stakeholders may distribute their points based on how they think others will do it. In addition, it is difficult for stakeholders to keep an overview of a large number of requirements.</td>
</tr>
<tr>
<td>5</td>
<td>Value-oriented prioritization method</td>
<td>Prioritizes requirements based on their contribution to the core business values and their perceived risks. The first step in setting up a value-oriented prioritization process is to establish the framework and this framework is used to identify the value of the business and the relative relationship of those values. Business values are established at the level of organization. After identifying the core values, the organization must provide some indication of importance of those values to the organization. This is accomplished by assigning weights that use a simple ordinal scale ranging from 0 (not important) to 10 (critical).</td>
<td>As prioritizations involve a small subset of stakeholders; the results are biased towards the perspective of those involved in the process.</td>
<td></td>
</tr>
</tbody>
</table>

IV. CONCLUSION

Requirements engineering able to interact with many technical and non-technical requirements to improve the requirement elicitation, and requirements Prioritization. Efficiency of the requirement elicitation process...
of software engineering can be improved by application of Idea Station as well as BackApp. This paper provides a general outline for both the software’s.

REFERENCES


