CHAPTER 2

Review of Literature

2.1 Techniques of Requirements Prioritization

The prioritization techniques can be categorized as Nominal scale, Ordinal scale, and Ratio scale

2.1.1 Nominal Scale

In nominal scale method, requirements are assigned to different priority groups, in which all requirements that are in one priority group represents equal priority.

2.1.1.1 Numerical Assignment

Numerical Assignment prioritization method is mentioned in a large number of studies such as (Karlsson et al. 1998). Numerical Assignment is a fundamental technique for requirement prioritization in which several requirement prioritization groups are created and requirements are assigned to these groups on the basis of their priority There are several groups but the most commonly requirements are grouped as critical, standard, and optional. The results of a Numerical assignment are from the range of nominal scale i.e. from 1 to 5 to indicate the importance (Karlsson et al. 1998), where the range represents:

1. Need not matter.
2. May not be important.
3. Important.
4. Very important.
5. Mandatory.

The probable problem in this technique is dividing the requirements into the three groups because a stakeholder thinks every requirement to be crucial. Thus all the requirements are considered high in priority and are implemented during the early releases. Due to which the cost benefit and real benefit of the prioritization is lost. One
technique is to apply restriction on the number of requirements allowed in each group which in turn decreases the usefulness of the priority because the stakeholders are forced to divide requirements into the respective groups. All requirements from one priority group represent equal priority. There is no such parameter that shows which requirement is of higher priority than another requirement within one priority group. Another problem with respect this technique is that all requirements under a group are categorized under same priority but using techniques like AHP or 100 test requirements in a single group can be prioritized.

2.1.1.2 Moscow (Museum of Soviet Calculators on the Web)

MoSCoW is a type of numerical assignment technique which is described in. There are four priority groups that are MUST have, SHOULD have, COULD have and WONT have, which are given by Moscow. Requirement prioritization is done by placing each requirement in one group based on their priority as elucidated in the Fig 2.1.

"MUST have” means that requirements in this group must be accommodated in the project. Failure to achieve these requirements means the entire project would be a failure.

"SHOULD have” means that the project would be working efficiently if it consists of the requirements in this group.

"COULD have” also means that the requirements are of less importance than that of in ”SHOULD have” but are necessary to have these requirements.
"WONT have" means that the requirements in this group are not implemented during the current stage. Due to their low priority they might be implemented in the next release.

All requirements from one priority group represent equal priority. There is no such parameter that shows which requirement is of higher priority than another requirement within one priority group.

2.1.2 Ordinal Scale prioritization technique

For ordinal scale methods, the result is an ordered list of requirements. This technique only produces result of which technique is of higher priority than the other, but not the extent.

2.1.2.1 Simple ranking

Simple ranking requirements prioritization technique is suggested by Berander and Andrews Hatton (Hatton 2007). Requirements are ranged from 1 to n where n is any integer value. Ranking for a higher priority requirement is given by 1 and for lower priority requirement it is given by n.

Bubble sort, binary search tree (Karlsson et al. 1997) and quick sort techniques are used to obtain ranks of the requirements. The simple ranking technique is optimal when a single stakeholder needs to prioritize requirements. Bespoke development uses this kind of prioritizing technique. While ranking, multiple aspects could be combined by taking mean priority of each requirement into consideration but that results in ties for a requirement which needs to be avoided.

The exclusive and relative ranking is not possible in simple ranking technique as that of Numerical assignment and AHP respectively.

2.1.2.2 Binary Search Tree

Binary search tree method was described by Hopcroft, Aho, and Ullman which is another technique used for sorting requirements which has been elucidated in the Fig 2.2. In binary search tree, all nodes have maximum of two children. The first binary search tree for requirement prioritization was presented by Karlsson (Karlssona and Wohlinb 1997).

Each node in a binary search tree represents a requirement. The low priority requirements are those requirements which are placed on the left side of the node and high
Priority requirements are those requirements which are placed on the right side of the node in binary search tree.

During requirement prioritization, initially, we take a requirement and accommodate that requirement as a base node. After that, we take any random requirement and compare that with the base node. If that requirement is of less importance than the base node then we compare the requirement to the left child node of the root node and if the requirement is of larger importance than the base node then compare it with the right child node of the base node. If the base node does not have any child node then place that requirement as a new child to that root node. If the requirement is having larger priority than the root node, then place that as a right child of the root node and if it is of less priority then place that as a left child of the root node. Repeat this process till each and every requirement is placed in the binary search tree. The average complexity for a binary search tree is $O(n \log n)$.

2.1.2.3 Bubble Sort

Bubble sort is a technique which is used for sorting of requirements. Hopcroft, Aho, and Ullman, Karlsson (Karlsson et al. 1997) prioritize software requirements first by using Bubble sort technique. Requirement prioritizing using bubble sort is carried out by taking two requirements at a time and comparing them with each other if the requirements are not in sequence then swap the requirements and take another requirement and start comparing with it as elucidated in the Fig 2.3. The comparisons continue until no more swapping is needed. The average and worst-case complexity and comparisons for bubble sort are $(n^2)$ and $n^*(n-1)/2$ respectively.

Bubble sort prioritization technique requires the following steps to be performed
Fig. 2.3 Bubble Sort

1. List out all the requirements column wise.

2. Compare and contrast the first two requirements from the top to determine which is having higher value and swap the requirements in order to place higher requirement above the lower one.

3. Repeat this comparison for second with third requirement and then third with fourth requirement and so on till we reach the end of the column and swap the requirements simultaneously.

4. If there is change in the position of requirement while comparing with the previous pass position then the process is repeated until there are no changes in the positions of the requirements. At the end of the process all requirements are arranged in priority order.

5. The produced result is of the form of a column where requirements are arranged such that the requirement with higher priority is placed at the top and the requirement with lower priority is placed at the bottom.

The simple ranking, bubble sort, and binary search tree all the three methods are used for ranking the requirements. The simple ranking is straight forward for use, but
bubble sort and binary search tree methods are complex methods. There may be a discussion on why bubble sort and binary search tree needed while we have the simple ranking method. The answer is that if requirements are less in number then we can apply the simple ranking method and it is easy to use. But if requirements increase in numbers then there arises a complication in remembering all the requirements thus it becomes complex. Miller stated that it is difficult during prioritization to remember more than seven requirements (plus or minus two). Hatton described that it inappropriate for people to use the simple ranking method for ranking more than 15 requirements. Therefore, if a large number of requirements need to be ranked then it is better to use bubble sort or binary search tree rather than using simple ranking in order to get a high degree of accuracy.

2.1.2.4 Priority Grouping

The priority groups technique was stated by Karlsson (Karlsson et al. 1998). The ultimate aim of priority groups is that it is similar to numerical assignment technique. In this method, requirements are not compared with each other with respect to an aspect or criteria, essentially requirements are grouped into three major groups low, medium and high priority groups or essential, conditional and optional groups or amongst the four categories i.e., most needed, good to have, ok to have and not to have based on the importance of requirements which has been elucidated in the Fig 2.4.

The following are the essential steps that are required to be performed in order to achieve requirement prioritization using priority group technique (Karlsson et al. 1998).

1. Initially, all the candidate requirements are gathered.

2. Categorizes each requirement into any one of the three groups: high, medium and low priority.
Table 2.1 The two commonly used priority scale for grouping

<table>
<thead>
<tr>
<th>Scale</th>
<th>Group Levels</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scale-1</td>
<td>High</td>
<td>Mission Critical</td>
</tr>
<tr>
<td></td>
<td>Medium</td>
<td>Required but can wait</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>Can live without</td>
</tr>
<tr>
<td>Scale-2</td>
<td>Essential</td>
<td>A requirement is must</td>
</tr>
<tr>
<td></td>
<td>Conditional</td>
<td>Not unacceptable without</td>
</tr>
<tr>
<td></td>
<td>Optional</td>
<td>May or may not be there</td>
</tr>
</tbody>
</table>

3. If a group contains more than one group then create three new subgroups i.e. (high, medium, low) and then categorize the requirements in the parent group into these subgroups.

4. Unless there is one requirement in each subgroup repeat step 3 which has been elucidated in the Table 2.1.

In Kano analysis in order to achieve customer satisfaction the analysis tries to discover and differentiate the incremental requirements from essential requirements as elucidated in the Table 2.2.

<table>
<thead>
<tr>
<th>Groups</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>Surprise and Delight</td>
<td>Features of differentiation from competition</td>
</tr>
<tr>
<td>More is Better</td>
<td>Features offering increasing utility</td>
</tr>
<tr>
<td>Must be</td>
<td>Essential features needed by customer</td>
</tr>
<tr>
<td>Better not be</td>
<td>Features of Dissatisfaction to customer</td>
</tr>
</tbody>
</table>

2.1.2.5 Wiegers Method

In context to Weiger’s, initial prioritization is done by customers with respect to the aspect i.e value. Later when technical risk and cost come into account then the priorities changes once again. This method directly relates a customer to value of each requirement. Priority in this method is calculated by the value of requirement by the sum of the cost and technical risk associated with the implementation.

In Weiger’s method, the value of requirement depends on both, value which is being provided to the customer by the client and the penalty if any value of the requirement is missing. Developers evaluate the requirements cost and the associated implementation risks along with a penalty if there are any requirements missing.

Within same abstraction level requirements are listed sequentially and for each requirement, the following four parameters are measured on a scale of 1 to 9: Cost, value, risk, penalty. Where 1 indicates low value and 9 indicates a high value on the scale.

Relative weight is measured twice in order to calculate the total value, where total
value includes benefits of implementation and penalty if the desired feature is not implemented. Where cost indicates the cost for development and risk signifies the risk associated with the feature. An example for wiegers requirement prioritization is elucidated in the Table 2.3.

The Priority is given by:

\[
\text{Priority} = \frac{\text{value}\%}{\text{cost}\% \times \text{cost weight} + \text{risk}\% \times \text{risk weight}}
\]

Weiger stated that this method can only estimate 4 aspects for an individual requirement and it should be used to make decisions. Weiger’s method includes a penalty or negative value for not implementing an aspect. Cost includes the existing modules benefits and impacts are enclosed within risks.

2.1.2.6 Triage Method

The Development managers perform set of activities such as estimating efforts, the time required to implement a specific feature which is required and should retain features that satisfy the budget and fits in the schedule and removing all other requirements is called as Requirement triage. Requirement triage impacts the profit and revenue; triage needs to gather a set of features which can be implemented with the help of available resources and risk level which can then be sold at an affordable price in the market with acceptable levels of revenue and profit.

In Davis point of view, the variables that are disposing of the team performance triage are:

- Adding or deleting a feature and change.
- Postponed or advanced delivery date.
- Increased or decreased resources.
- Increased or decreased price.

Triage alters the assumption made by these variables after which product plan is generated with all the features that need to be involved, markets that need to be addressed, and needs of resources and expectation of revenue generation.

Triage is defined by Davis as “The art of selecting the right features to include in the next release, balancing the requirements with development cost, risk; schedule; market,
sales, revenues, pricing, profits, ROI”.
Triage process is performed as elucidated in the Fig 2.5.

Fig. 2.5 Triage Methods

2.1.3 Ratio Scale
The relative difference between a set of requirement is provided by the results of ratio scale methods.

2.1.3.1 Hundred Dollar Method
The Cumulative voting method which is also referred as Hundred Dollar prioritization method mentioned in (J.karlsson et al., 2007). In the hundred dollars method, 100 points i.e. (in terms of hours, importance, penalty, money etc) are given to the stakeholders to partition the 100 points amongst the requirements. Then every stakeholder partitions the points among the requirements on the basis of its essentiality. For example if there are 100 points given to the every stakeholder and there are 5 requirements which are needed to be prioritized, every stakeholder has an option to partition 20 points to every requirement, or may provide all of the 100 points to a single requirement if the stakeholder assumes it to be a necessary requirement which has been elucidated in the Fig 2.6. This prioritization technique is complex in terms of sophistication and fine in terms of granularity (Berander 2004).
The problem with this technique is that if the number of requirements goes beyond hundred or if during calculating the allocated points to the requirements if it exceeds or falls below hundred (Berander 2004). In order to overcome these problems, there is the usage of automated tools during the partitioning which keeps track of the count.
Another problem is that if the stakeholder partitions the points to a single requirement or to the requirements which other stakeholders do not prioritized as high requirement thus it biases the prioritization process. Thus this can be avoided by limiting the distribution of points to a single requirement. So, that it forces the stakeholders not to prioritize only to their requirements. Prioritization should be done only once to a set
of requirements because during the second time there is a chance of biasing if desired requirements are not elected.

![Fig. 2.6 Cumulative Voting](image)

2.1.3.2 Analytic Hierarchy Process (AHP)

Analytic hierarchy process has been stated by Saaty (1990) and it is designed for complex decision making. In AHP, comparison of all possible pairs of hierarchical requirements is done to determine the priority. The user first discovers the attributes and alternatives for each requirement and uses them to create the hierarchy. Then the user specifies his or her choice to each pair of the requirement by providing a preference scale which is generally ranging between 1 till 9. Where 1 represents equal value and 9 indicates extreme value. After that AHP transforms the users evaluation to numerical values and numerical priority is derived for each element. During requirement prioritizing using AHP there might be redundancy, therefore a ratio should be calculated to judge the valid prioritization method. If prioritization is needed then, \( n^2(n-1)/2 \) pairwise comparisons are done using AHP method. Therefore the complexity of AHP is \( O(n^2) \). The Table 2.4 elucidated the overall representation of the levels.

There are five levels of judgment - 1,3,5,7,9 which correspond to the equal value, slightly more value, strong value, very strong value and extreme value respectively which are defined on the scale. 2,4,6,8 provide intermediate judgment values when there is a need for compromise. The reciprocal of accredit number in one requirement
Table 2.4 Fundamental scale used for AHP

<table>
<thead>
<tr>
<th>Importance level</th>
<th>Definition of Judgement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Equal Importance</td>
</tr>
<tr>
<td>3</td>
<td>Moderate Importance of one over the other</td>
</tr>
<tr>
<td>5</td>
<td>Essential or strong importance</td>
</tr>
<tr>
<td>7</td>
<td>Very strong importance</td>
</tr>
<tr>
<td>9</td>
<td>Extreme importance</td>
</tr>
<tr>
<td>2,4,6,8</td>
<td>Intermediate judgements</td>
</tr>
<tr>
<td>Reciprocals</td>
<td>$S(j,i)=1/S(i,j)$</td>
</tr>
<tr>
<td>Rationals</td>
<td>Ratios arising from the scale</td>
</tr>
</tbody>
</table>

develops into the priority for the pairs other requirement. If there are ratios arising out of comparisons, rationales support the scale.

In order to prioritize Requirements using AHP four steps are needed to be followed:

1. Requirements are arranged in an $n\times n$ matrix in a sequential arrangement.

2. Consider two unique pair requirement $A$ and $B$, relative value of importance is place in the position where the row $A$ and column $B$ meets each other. Similarly place the reciprocal value in transpose of the position of $AB$.

3. In order to find each requirements relative priority, the resulting Eigen value of the comparison is calculated.

2.1.3.3 Hierarchy AHP

Davis (1993) stated that requirements are structured in a hierarchy in larger projects, with well-established requirements placed at the top of the hierarchy and the more specific requirements placed at the lower levels of the hierarchy. In which prioritization of requirements is done by AHP only at the same level of a hierarchy. This method decreases the number of decisions when compared to AHP method because not all requirements are compared pair-wise. This can decrease the number of redundant comparisons, but the problem is that the ability to discover inconsistent judgments is also reduced.

2.1.3.4 Minimal Spanning Tree

Minimal spanning tree is another prioritization method which is stated by Karlsson Karlsson et al. (1998) for referencing software requirements. Minimal spamming tree eliminates the redundant comparisons that were done in AHP. (eg if during comparison of three requirements if we know that $A$ is of higher priority than $B$, and $B$ is of higher
Table 2.5 Pairwise comparison Assigning Relative priorities

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Relative Benefit</th>
<th>Relative Penalty</th>
<th>Total Value</th>
<th>Value %</th>
<th>Relative Cost</th>
<th>Cost %</th>
<th>Relative Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>6</td>
<td>4</td>
<td>16</td>
<td>39</td>
<td>3</td>
<td>38</td>
<td>1</td>
</tr>
<tr>
<td>R2</td>
<td>9</td>
<td>7</td>
<td>25</td>
<td>61</td>
<td>5</td>
<td>63</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 2.6 Eigen Values Representing Relative Value

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Normalized column sum</th>
<th>Average normalized column</th>
<th>Eigen values</th>
<th>Relative values</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>1.05</td>
<td>1.05/4</td>
<td>.26</td>
<td>26%</td>
</tr>
<tr>
<td>R2</td>
<td>1.98</td>
<td>1.98/4</td>
<td>.50</td>
<td>50%</td>
</tr>
<tr>
<td>R3</td>
<td>.34</td>
<td>.34/4</td>
<td>.09</td>
<td>9%</td>
</tr>
<tr>
<td>R4</td>
<td>.62</td>
<td>.62/4</td>
<td>.16</td>
<td>16%</td>
</tr>
</tbody>
</table>

priority than C then there is no need to compare A and C as the above comparisons indicate A is of higher priority than C). Thus this redundancy allows discovering judgment errors but results in scalability issues. Due to redundancy the number of comparisons are reduced to n-1 with respect to n*(n-1)/2 in the case of AHP. Minimal Spanning tree constructs unique pairs of requirements, and it is a directed graph which is minimally connected.

2.1.3.5 Cost-Value Approach

Cost-Value approach is given by Karlsson and Ryan (Karlsson et al. 1997). The idea behind the Cost-Value approach is that each requirement is categorized on two factors, the value to the users and the cost of implementing the requirement. AHP technique is used to compare requirements pair-wise according to the relative values and costs. The cost-value approach for requirement prioritization is time-consuming as per the studies made by Karlsson.

Two facts are taken into consideration. If the two factors are true then those factors can, in turn, encompass all the other facts internally, re-evaluation of prioritized requirements is caused due to interdependencies of requirements for feasibility of implementation. The Table 2.5 and Table 2.6 elucidates the process of assigning relative priorities and eigen values that represents the respective relative priorities.

2.1.4 Combined method

Some Requirement prioritization methods which do not fall under the category of Nominal scale, Ordinal scale and ratio scale fall under this category
2.1.4.1 B-Tree

B-Tree approach is a structured way in which the comparison made can be maintained low. Md.Rizwan Beg stated B-Tree prioritization technique Beg et al. (2008). B-Tree merges certain aspect such as dynamic incoming requirements i.e. the requirements which are never frozen and it also deals with the situation if certain requirements are dropped during runtime. B-Tree Prioritization utilizes the similar technique as that of a Binary tree but it is more balanced, structured and has other advantages over the Binary tree prioritization. It supports the run-time capability i.e. if there are n number of requirements waiting till they are finalized and prioritization can be started with as many requirements available at the moment. If the total number of comparisons are known in advance then the number of comparisons are fixed and controlled for prioritizing a requirement. Even without finalizing the result, B-Tree Prioritization technique presents its results better than other techniques. B-tree prioritization has been elucidated in the Fig 2.7.

2.1.4.2 Planning Game

Beck in 1999 stated a prioritization technique, known as Planning Game, Planning game deals with a combination of prioritization techniques. Agile software development projects use planning game method.

Planning game categories requirements into three categories:

1. Those requirements without which the system will not function.
2. Those that are not much essential but provide significant business value.
3. Those requirements that are nice to be included.

Once the requirements are assigned into the three groups, requirements are ranked with simple ranking technique in each group (Beg et al. 2008).

Release Planning is done in planning game technique because of the use of agile software projects. This process focuses on those requirements which are needed to be included in the immediate release. The ultimate aim of planning game is to guide in the delivery of a software product. It also aims at steering the project to delivery with the help of straightforward approach rather than just mere prediction of dates of delivering the deliverables.

2.1.4.3 Win-Win Method

Theory-W or Win-Win method was developed by Boehm and Park at Southern California University in 1989. Win-Win method supports negotiation in order to solve the disagreements regarding requirements and it remarks each stakeholder as a Winner as elucidated in the Fig2.8. Two Principles that are followed in Win-Win method are:

![Win-Win Method Diagram]

**Fig. 2.8 Win-Win Method**

1. Flight planning and flying the plan.

2. Managing the identified risks.

The initial step is Building a plan which is well structured and that meets standards which are defined already in order to achieve easy development, query and classification. The next step involves assessment and handling the risk. Re-ranking is done with respect to constraint and targets to arrive at requirement prioritization which is conflict free.

Win-Win or theory -W Follows these four steps:
1. Problem and Stakeholders are separated.

2. Focus is on interests and not with their positions.

3. Stakeholders invest options in order to get mutual gains.

4. Objective criteria are insisted to use.

2.1.4.4 Quality functional deployment QFD

QFD considers customers needs which are applied through a systematic methodology (Crow 1994). A matrix for a product is created a quality house, which entitles both how(needs of customers) and what ( needs of a designer). The most important criteria that QFD delivers is that it helps us to view the product features from different angles, generally the customer and the company as elucidated in the Fig 2.9. Manufacturer

![Fig. 2.9 Quality functional Deployment methods](image)

prioritizes the requirements that are provided by the customers. During this process, the manufacturer completely understands the priorities of the requirements and categorizes them into engineering and business process requirements. When this process ends, the manufacturer checks for the design criteria to ensure that the requirements are meet with as that of the requirements stated by the customer.

The customer’s needs are linked with design, development and other organizational functions by QFDs standard deployment:

1. QFD describes the out spoken and unspoken customer needs.
2. QFD is used to uncover the "true" needs of the customer and "positive" quality of the customer.

3. The customer needs are translated into design characteristics and other deliverable actions.

4. QFD delivers the built quality product and service by taking into consideration numerous functions in order to achieve a common goal i.e. Customer satisfaction

2.1.5 Which Prioritization technique is apt for usage

**Table 2.7** Comparison between prioritization techniques

<table>
<thead>
<tr>
<th>Technique</th>
<th>Scale</th>
<th>Granularity</th>
<th>Sophistication</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numerical Assignment</td>
<td>Nominal</td>
<td>Coarse</td>
<td>Very Easy</td>
</tr>
<tr>
<td>Moscow</td>
<td>Nominal</td>
<td>Fine</td>
<td>Easy</td>
</tr>
<tr>
<td>Simple Ranking</td>
<td>Ordinal</td>
<td>Medium</td>
<td>Very Easy</td>
</tr>
<tr>
<td>Binary Search Tree</td>
<td>Ordinal</td>
<td>Fine</td>
<td>Complex</td>
</tr>
<tr>
<td>Bubble Sort</td>
<td>Ordinal</td>
<td>Medium</td>
<td>Complex</td>
</tr>
<tr>
<td>Priority Grouping</td>
<td>Ordinal</td>
<td>Medium</td>
<td>Easy</td>
</tr>
<tr>
<td>Weigers Method</td>
<td>Ordinal</td>
<td>Coarse</td>
<td>Very Complex</td>
</tr>
<tr>
<td>Triage</td>
<td>Ordinal</td>
<td>Coarse</td>
<td>Complex</td>
</tr>
<tr>
<td>Hundred Dollar</td>
<td>Ratio</td>
<td>Fine</td>
<td>Very Easy</td>
</tr>
<tr>
<td>AHP</td>
<td>Ratio</td>
<td>Fine</td>
<td>Very Complex</td>
</tr>
<tr>
<td>Hierarchy AHP</td>
<td>Ratio</td>
<td>Fine</td>
<td>Complex</td>
</tr>
<tr>
<td>Minimal Spanning Tree</td>
<td>Ratio</td>
<td>Medium</td>
<td>Complex</td>
</tr>
<tr>
<td>Cost Value</td>
<td>Ratio</td>
<td>Coarse</td>
<td>Easy</td>
</tr>
<tr>
<td>B-Tree</td>
<td>Combined</td>
<td>Fine</td>
<td>Complex</td>
</tr>
<tr>
<td>Planning Game</td>
<td>Combined</td>
<td>Fine</td>
<td>Easy</td>
</tr>
<tr>
<td>Win Win</td>
<td>Combined</td>
<td>Medium</td>
<td>Easy</td>
</tr>
<tr>
<td>Quality Functional Deployment</td>
<td>Combined</td>
<td>Medium</td>
<td>Complex</td>
</tr>
</tbody>
</table>
The following Table 2.7 elucidates an overall picture of the prioritization techniques that have been discussed on the basis of measurement scale, granularity, and sophistication level associated with the techniques. The best method is to use the simplest prioritization technique when the requirements are basic and to go with the more sophisticated one when there is a need for more sensitive analysis in order to resolve the disagreements. (Maiden and Ncube 1998) A more sophisticated technique consumes more time whereas the simple prioritization technique ensures cost effective decisions. The aim is to decide which technique is appropriate without affecting the quality of decisions.