TOPE: Tech of Problem Evaluation

Shortcut to Requirement Mapping

Jay Mohan

Rifluxyss

Overview

When it comes to requirement gathering for development of software system, a lot of software writers and analysts follow the age-old processes wherein the needs of the customers are given priority. This follows the ideology that customers require software for complementing their needs and making their work hassle free.

The key factor, which always stands out in the above methodology is that 'our needs are vague'. The psychological factors named “greed” and “desire” forces the human mind to ask for more and more. The intangible becomes complex, with a myriad of addendums that evolves from the human factors involved.

The application of a change control process becomes a priority. In reviewing some major software application development efforts by our partners it was revealed that we loose 30-60% of time on changes and modifications because priorities are left out at various stages of the software development cycle. The customer in the end will have a jazzy application wherein certain key areas might have been ignored in the initial requirement-gathering phase itself.

TOPE

TOPE relies on the strong foundation that

"NEEDS ARISE BECAUSE OF PROBLEMS".

The foundation for the initial phase of the Software Development Life Cycle (SDLC) should be from the grass root level since SDLC is of a pyramid form.

Needs are abstractions of problems. Solving problems are the inherent and invisible requirement of any customer. Rifluxyss, with its deep knowledge in the software development process studied this concept and came out with “Technique Of Problem Evaluation” (TOPE). TOPE is a step by step process to plot out problems, then their cause, and in turn detail out the expectation of a customer, which we commonly call “Requirements”.

Another major factor that has led us to structure this new technique is the fact that in any software application requirement, the customer relies on some 20% of operations for 80% of their work. The other 80% are only extra operations that are used a lesser number of times. If we relate this to the problem solving technique outlined in this document, these 20% of operations can be identified easily by identifying the key problems of the customer.

TOPE follows a structured data gathering operation for storing these problems and identifying their priorities and impact on the entire system. This process lets us identify the key operations that might have the major impact on the entire application development process.

The total SDLC involves a two-layer approach based on TOPE. The key problems are solved in a PILOT phase that is developed and tested prior to the development of the entire system in the PRODUCTION phase. The steps involved are detailed in subsequent pages.

This process in fact serves all the major kinds of software project. It covers the situation where the system to be developed is a totally new application, to it is a re-engineering requirement of an existing application. This technique will also address the software integration requirements, where the usual requirement gathering method fails.

TOPE Process

The TOPE structure follows definite processes at various levels to extract deeper level data for unambiguous software system specification. The individual processes involved are listed below.

1. Listing Visible Problems
2. Seriousness Mapping
3. Problem Division to Sub-Problems
4. Complete Problem Tree
5. Solution Identification
6. Problem-Solution Tree
7. Solution Abstraction
8. Pilot phase Identification

Each process has to be done in a structured way for the success of the entire model. The output of one phase acts as the input of the next and we do have to maintain the process order since data collection needs clarity of flow.

Listing Visible Problems

Customers are always happy to tell you their problems rather than give requirements. They feel that giving requirement is a tough job which requires lots of efforts from their side. The approach in all processes of TOPE is to make them divulge their problems in a clear and specific manner.

Interviews and Group Discussions are the major mode of data extraction in this initial phase. The Analyst requests the customer to detail their problems by totally forgetting what kinds of solutions are available, or possible, to resolve their problems. This in turn forces the customer to think about the difficulties he/she is facing on a daily basis. They will come up with all kinds of problems they are facing daily.

The Analyst uses the "Stick-It" method. The commonly seen Yellow "Stick It" papers are used to display the mounting problems that the customer is facing. Each and every specific problem is written on a piece of "Stick-It" paper and pasted on the wall or table in front of the customer. An example of this is shown in the figure.

Accounting is complex with 14 layers of billing information
(High)
Inventory verification is slow.
(High)

The “Stick-It” method has a great advantage as it demarcates the problem boundary in the initial extraction phase itself. Each “Stick-It” paper pasted depicts a single problem.

Moreover there is a temptation by the analyst as well as the customer to fill as many yellow papers as possible. It will stimulate the brains involved so that the work faster and more efficiently.

The pasted “Stick-It” sheets are segregated into different groups after filtering for redundancy. Each group becomes the software modules in the development phase.

Seriousness Mapping

Each and every problem has its own seriousness and it is the customer who will be the right person to associate seriousness with the problem. In this process the major factor that affects the workflow is a “Seriousness Matrix” that is formulated on a project-to-project basis. This matrix depends on the worse case scenarios expected out of each problem. The customer is the best person to analyze and fix the general seriousness.

Generally there are four level of seriousness of problems as detailed in the following table:

Major	This belongs to the most critical problems of the entire application. If this problem occurs the entire application does not work.
High	This is also critical but the seriousness level is slightly reduced because on occurrence of this problem the total application is not useless. But the problems will affect the major operations of the system.
Average	This falls in the less serious nature because the effect on occurrence of this on the other parts is not major.

Low	This is the least serious level of problem that can happen. These are minor operations where in the occurrence of this does not have any effect on the other parts of the software system.

The above table is depicted here to give a general idea as to how a matrix is formulated. Once the matrix is established with the help of the customer, it is then the responsibility of the analyst to award a seriousness level to each "Stick-It" paper that has been pasted in order to formulate the total problem.

The final efforts in this mapping process is to port all the data available on "Stick-It" papers to a well documented format, explaining the details of each problem and why the corresponding seriousness has been awarded for each.

Division to Sub-Problems

The previous step generates a basic problem list that details the visible problems that are apparently clear because of the experience of the customer. There is a high probability that each of these listed visible problems might contain invisible sub-problems. The key to this process is identifying these inherent invisible sub-problems.

Using methods like "Creative Brain Storming" and "Problem Post-Mortem" by Functional Experts these Sub-Problems can be identified and listed out.

Let us look at a brief example that makes the above clear. The customer has identified a problem that:

> "He is not able to file the Pension Funds paper on time because of the manual process involved for processing and printing the required papers for all staff involved in his firm".

Let us look at the "Worse Case" scenario for this problem. This problem is critical but it does not affect the other operations of the company in any major manner. Hence we can term the seriousness level of this problem as High by the above-mentioned matrix.

This problem on analysis by an expert will raise further problems like

> "Any increase in the basic salary does not automatically increase the Pension Fund contributions unless, and until, the necessary forms are submitted and accepted by the PF Department".

Sub-Problem listing is more difficult than the visible problem listing. Even Seriousness can be associated to these sub-problems but it is not mandatory because TOPE is concerned more on the Seriousness measure of the Parent problem rather than the Sub-Problem.

The pasted Stick-It sheets are segregated as different groups after filtering for redundancy and each of this group becomes the software modules in the development phase.

The major advantage with TOPE is that it follows a two-phase approach for the entire problem solving:

a) Pilot Phase

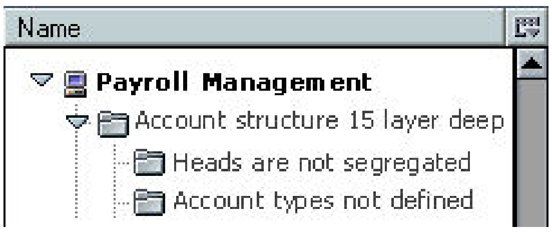
b) Production Phase.

Initially the Sub-Problem generation and other subsequent processes are applied only to the visible problems having High & Major Seriousness levels.

This in effect will result in a Pilot system that can merge all the major operations that are required by the customer. The customer can then use this prototype for getting a feel for the problem solving software application. This in effect can help the customer realize the intangible system to an initial level of working system.

Complete Problem Tree

Once the detail list of problems and Sub-Problems have been identified, the analyst can plot this entire information in a tree like format, with seriousness plotted against each branch. A sample of a "Problem Tree" is given in the following figure.



Name

Payroll Management

Account structure 15 layer deep

Heads are not segregated

Account types not defined

The "Problem Tree" facilitates a quick overview of the overall problems faced by the customer. However the most important reason as to the need of the "Problem Tree" is the psychological advantage of the human brain to work and develop the necessary solution list once the overall problems are listed out in total.

The analyst can use Microsoft Word or some other document writing software to prepare the "Problem Tree". There are certain specifications and notations to be followed when preparing this tree. These are proprietary information of Rifluxyss and hence not being discussed here.

Another key property that can be advantageous to the analyst is the option of marking the cause of the problems in this tree itself. This makes the development of solutions a bit easier at a later stage.

The main goal of preparing this "Problem Tree" is to associate all the problems in an interlinked manner, wherein all the relationships among the problems will be shown in an appropriate forms. This in fact targets those kinds of problems that might have some kind of association with another problem that is already listed in the tree. These inter-relationships are to be noted for the successful development of the overall system.

Solution Identification

Even though all problems may or may not have one or more solutions, we follow this process with a similar assumption. Our assumption can prove to be invalid in certain stages and this results in special procedures to be invoked for the system development.

With the help of effective brainstorming and expert advice, solutions are plotted for each branch of the "Problem Tree". The solutions, which evolve out of these discussions, might not be totally technical ones. At later stages however the technical experts process these non-technical solutions to technical solutions. These solutions can be fed into the other normal SDLC process.

The solutions that are identified are commonly called “requirements” in the existing requirement analysis phases. The major difference from the usual method is that all the relevant requirements will be projected out with their priorities since the process involved prevents information loss.

Problem-Solution Tree

The identified solutions are plotted in the previously constructed “Problem Tree” following the specifications followed by the company. This complete “Problem-Solution Tree” will be a totalitarian view of the entire system that needs to be developed.

It is quite important to note that all the problems might not have solutions. The problems that don't have solutions will be removed from customer discussions. These problems are removed from the “Problem-Solution Tree”. In fact “Problem Solution Tree” should have a “one – one” or “one – many” relationship between problem and solution.

In this tree there is no marking for Seriousness of the problem. Rather we will be marking the priority of the solution. The priority calculation is based on certain rules that have been formulated by Rifluxyss. They are not described in this document for reasons of confidentiality.

Solution Abstraction

Once we have a complete “Problem Solution Tree”, the system requirements can be extracted by structured analysis of this through the “Solution Abstraction” process. This involves detailing out or even splitting various solutions till we get a finer level of solutions, which we can term as “Requirements” for the entire system.

The grinding of the solutions to finer level, even nested level of solutions requires expertise and functional knowledge of the customer activity domain. Once the basic solutions are listed out in the abstraction format (again this is proprietary for Rifluxyss and hence not displayed here), we have what is called the “Requirement Specification”.

Another key property that can be advantageous to the analyst is the option of marking the cause of the problems in this tree itself. This makes the development of solutions a bit easier at a later stage.

Production Phase Commencing

The main factor to be noted in here is that the entire cycle is repeated twice. The first time, this cycle applies to the Pilot phase problems. Once the requirement specification for this has been generated, the SDLC results in a prototype that solves the major problems of the client.

This prototype is deployed in the client location and the client is trained as to how to use the same. After sufficient exposure the client can come out with more problems, suggestions and opinions that can be absorbed documented and processed further in the second phase. This phase is the production phase wherein the entire problems are solved. This phase also follows the same cycle as described above and results in a productive development of the system wherein change and modifications are reduced by 30-50%.

Conclusion

Various stages are still under research and development. Those parts are not included in this document. Once these activities are established as standard in Rifluxyss this document will cover those areas. This document however gives a through overview of the TOPE process that can in effect increase your ROI by reducing wastage and time involved in the crucial phases of Requirement Analysis.

The Author

Mr. Jay Mohan is a Computer Engineer (B.Tech) graduated from the Cochin University of Science and Technology. As a certified Microsoft solution developer he undertook responsibility of leading the technical team of the software development firm Rifluxyss in 2004.

He restructured the entire team, standards and processes and the company has been having a flying growth of over 45% on annual basis since then. His project team grew from 3 to over 150 in a span of 3 short years. His in depth knowledge of software development methodology and the need for refinement in the legacy models made him research on new paradigms that is the need of the current time for successful software development.

He has been recongnized by various institutions like Goethe Institute and MILT for his contribution towards software development methodology and team building.

About Project Perfect

Project Perfect is a project management software consulting and training organisation based in Sydney Australia. Their focus is to provide organisations with the project infrastructure they need to successfully manage projects.

Project Perfect sell "Project Administrator" software, which is a tool to assist organisations better manage project risks, issues, budgets, scope, documentation planning and scheduling. They also created a technique for gathering requirements called "Method H"™, and sell software to support the technique. For more information on Project tools or Project Management visit www.projectperfect.com.au