

Hands-On Logic for Inventive Problem Solving – 0. Prologue

Mi Jeong Song Ph.D.; V. Leniachine, V.; Sung Cheol Kim; S. Antonov, Doctor

Six Sigma Innovation Team, Samsung Advanced Institute of Technology

San 14-1, Nongseo-ri, Giheung-eup, Yongin-si, Gyeonggi-do, 449-712, Korea

Abstract

TRIZ is known as one of most fruitful methodologies to excavate creative concepts for technical system improvement. But the bottleneck of ‘classical’ TRIZ is its ambiguity of the process to formulate the problem model from the initial situation. To eliminate this obstacle, after benchmarking valuable analysis tools, the SAIT TRIZ team has developed a simple and strong logic to define Technical Contradiction, so called Induced Technical Contradiction, even in very unclear stage of project initiation. This logic has been proved simple and strong enough for R&D researchers *themselves* to analyze valuable problem model sets and create innovative concepts. This logic is designed to be compatible with conventional DFSS roadmap and could be conducted as the sub-module of a comprehensive “Design for Six Sigma” (DFSS) project or the subject of an independent activity.

We’ve planned to publish a series of articles about the TRIZ logic. The beginning approach of simple logic of Technical Contradiction formulation is presented in this article.

Why we developed this logic?

Since Altshuller and his colleagues had developed TRIZ, this theory has been recognized at present as the most successful methodology to create innovative concepts in technical field (TRIZ implementation in non-technical field is out of the scope of this article). That’s why Samsung Advanced Institute of Technology (SAIT) has introduced TRIZ as one of the main tools for the project stage gate process in 2000. Indeed, the idea, “TRIZ can resolve contradiction”, is so attractive for almost all engineers and executive boards that many people have wanted to utilize TRIZ for their own problems.

Since 2000, TRIZ consultants (Dr. N. Shpackovsky, Mr. V. Leniachine, and his Korean colleague, Mr. H.J. Kim) have shown excellent performance to resolve many bottleneck problems. Their activity has driven engineers and executive board enthusiasm to learn

TRIZ.

In 2003, SAIT ex-president Mr. Sohn took the decision to apply TRIZ for planning “every R&D project”. Figure 1 illustrates the time when researchers used TRIZ in R&D management process flow of SAIT from the ‘R&D project initiation stage’ till the ‘critical decision stage’. This situation has forced us to develop a simple but strong logic to extract problem models from very ambiguous initial situation.

Very often, at the beginning stage of project initiation, everything is unclear: engineers don’t know who are their customers, what they (customers) want to have, why they want it and so on. Many kinds of marketing tools and scenario based roadmap activity are helpful for us to clarify this situation. These activities enable us to define who will be our customers and how high level of properties should be satisfied. It’s better situation comparing with absolutely initial situation. But it is still unclear. We don’t know what system should be analyzed; moreover, we don’t know what kind of contradiction should be overcome.

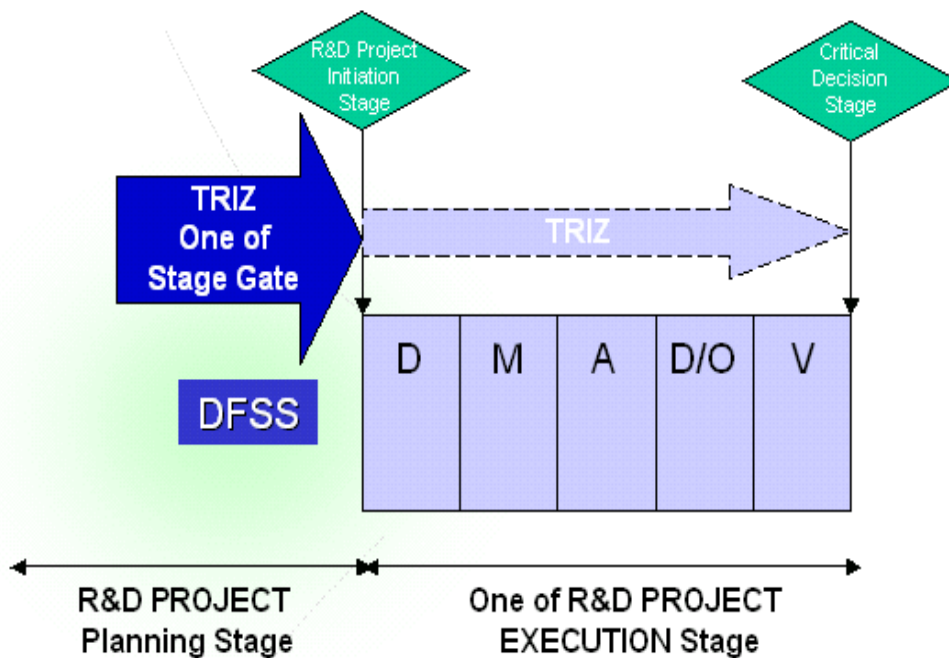


Figure 1. TRIZ in whole R&D Management Process in SAIT

Almost every TRIZ user knows the method to resolve “existing Contradiction”. But when we initiate project at the very first time, we don’t know what exact problem we have or we will have. Who knows how we can apply TRIZ for initiation project? Some TRIZ experts might say that evolution trend would be helpful enough to generate probable concepts for next technology. But random application of evolution trend is just

one of trial-error-methods modification but nothing else. But we need a relatively simple logic to show us what will be most valuable problems to resolve which will guide us to most promising concepts.

Some groups (3,4) have tried to clarify the formulating process from initial situation to Technical Contradiction (one of mini-problem) but their primary viewpoint is focused on the just “existing problems“. It differs from our situation. It is necessary for us to find out most valuable problem sets that have worth to resolve for very initial stage of scientific and/or engineering projects.

Recently B.Winkless, and Dr. J.Cooney (11) proposed a remarkable method to combine TRIZ with concurrent design from project planning stage. Their approach provides us meaningful insights but they didn't provide us for “any definite and workable logic chain”.

V.Fey et al. (9) also suggested a great logic for effective innovation driven by TRIZ but they haven't provided us a compatible logic with DFSS yet.

To meet the needs of SAIT, since 2003 TRIZ consultants in SAIT have evaluated many kinds of valuable problem analysis logics in TRIZ and other fields (Six Sigma, TOC, etc). In the end of 2003, we've defined the first prototype of simple logic for Technical Contradiction formulation and during 7 months we've confirmed it for more than 30 initiation projects including IT/SW projects.

Benchmarking result of existing problem analysis methods

Several well-known approaches (1~8) of problem analysis methods (according to our opinion, they are most appropriate for problems set up) have been investigated with respect to their advantages (good feature) and disadvantages (bad features) for implementation in the SAIT R&D structure based on DFSS.

The results of our analysis are represented in Table 1.

Table 1. Benchmarking finding for TC formulation logic

	Good feature	Bad feature
Classical TRIZ	<ul style="list-style-type: none"> ▪useful to develop new concepts for well-defined technical contradictions, 	<ul style="list-style-type: none"> ▪ineffective to formulate an appropriate TC from initial situation; ▪ambiguous and philosophical for practical purpose
TechOptimizer™	<ul style="list-style-type: none"> ▪Useful to generate system model ▪3 step algorithm is good to formulate TC ▪Excellent database for concept generation 	<ul style="list-style-type: none"> ▪3 step algorithm has no direct link with function/process analysis result ▪no appropriate method to formulate these multiple TC sets in systematical way.
Root Cause Analysis	<ul style="list-style-type: none"> ▪effective to describe logical links of problems ▪easy to create root cause analysis diagram ▪Based on multi-screen vision 	<ul style="list-style-type: none"> ▪too ambiguous for formal description of each elements, function, and so on ▪if there is no 'existing' technical contradiction, we can't formulate technical contradiction
IWB	<ul style="list-style-type: none"> ▪easy to create root cause analysis diagram ▪useful to extract 'existing' technical contradiction, ▪holistic description logic based on multi-screen scheme 	<ul style="list-style-type: none"> ▪too ambiguous for formal description of each elements, function ▪Ambiguous for project initiation stage
FMEA	<ul style="list-style-type: none"> ▪Very effective to formulate 'probable' bad effect of some elements, functions ▪Ultimate iteration work frame to minimize probable bad effects, ▪very friendly for SAIT researchers 	<ul style="list-style-type: none"> ▪tedious and boring to fill down every blanks in the 'table'
OTSM-TRIZ	<ul style="list-style-type: none"> ▪provides good philosophical background to understand initial situation 	<ul style="list-style-type: none"> ▪too ambiguous for practical purpose

The preparatory part for problem solving

There are a lot of methods to transform “real situation” to “situation description”, i.e. just saying, and/or writing a novel, and/or drawing down picture, and/or organizing diagrams, and/or filling down tablets – each of them is good for relatively narrow purposes, but does not resolve common problems.

Everybody will agree the importance of situation description. Really, it is the first step to clarify problem and if we took mistakes on this step, in almost all case we would fail to find good problems and moreover good solutions. But unfortunately, if we try to apply above approaches just method by method, we will fail to formulate appropriate Technical Contradictions from the initial situation.

Even though we thought we knew the problem exactly, it is necessary to describe the situation in formal symbolic method and confirm it one more time. The fact that problem still exists implies that there must be a strong mental inertia in our knowledge including system description and problem description. It is necessary to formulate what kind of

mental inertia we have for element and problem.

Every research project has its own customers. The goal of research project is to satisfy these customers. We know the height of level we should reach, but we don't know the way to achieve there over a lot of obstacles. It is so called "Administrative Contradiction" (1).

Our "required logic" should have two most important features to clarify this messed situation:

- At first, it should be very *effective* to "pick out the right Technical Contradiction" from very ambiguous initial situation
- At second, it should be *easy* enough for our researcher (***by themselves!***) to use it after 40 hours education of TRIZ pragmatic course developed in SAIT, and this logic should have DFSS compatible platform also.

To build such a simple but effective logic we tried to combine positive features of all mentioned logic described in Table 1, and introduce the "all-in-one logic chain" as a set of steps. The whole steps of Technical Contradiction formulation that is the base for mini-problem formulation from initial situation we describe as the following:

Step 1. Needs analysis

Step 2. Benchmarking Prototype systems

Step 3. Formulating prototype system model

Step 4. Determining cause/effect of undesirable function(s)

Step 5. Choosing Transition Action to fix up 'single undesirable function'

Step 6. Formulating effects of transition actions to system/supersystem

Step 7. Formulating induced Technical Contradictions

Step 8. Evaluating importance of induced Technical Contradiction

In this article we consider only two initial steps of this logical chain – all another will be included in future publications.

Every step has been incorporated to fight with specific mental inertias for customer, supersystem, system, elements, function, cause and effect, problem, solution and so on. This scheme has been applied in SAIT since 2003 and has been proved successful to describe the situation for initial project planning and patent overcoming projects.

The main destination of "classic TRIZ" is to resolve the problems by means of creative concepts generation. But a weak point of classic TRIZ is: how to pick up the problem from an ambiguous initial situation. On contrary, DFSS has relatively strong tools for

picking up the problem/set of problems from initial situation, but relatively weak tools for creative concepts generation. Moreover, there are no common points between these two methodologies that make additional difficulties for their common use.

Our new developed TRIZ logic includes TRIZ implementation in the framework of DFSS roadmap (13). Figure 2 summarizes the whole scheme of TRIZ activity process in framework of DFSS.

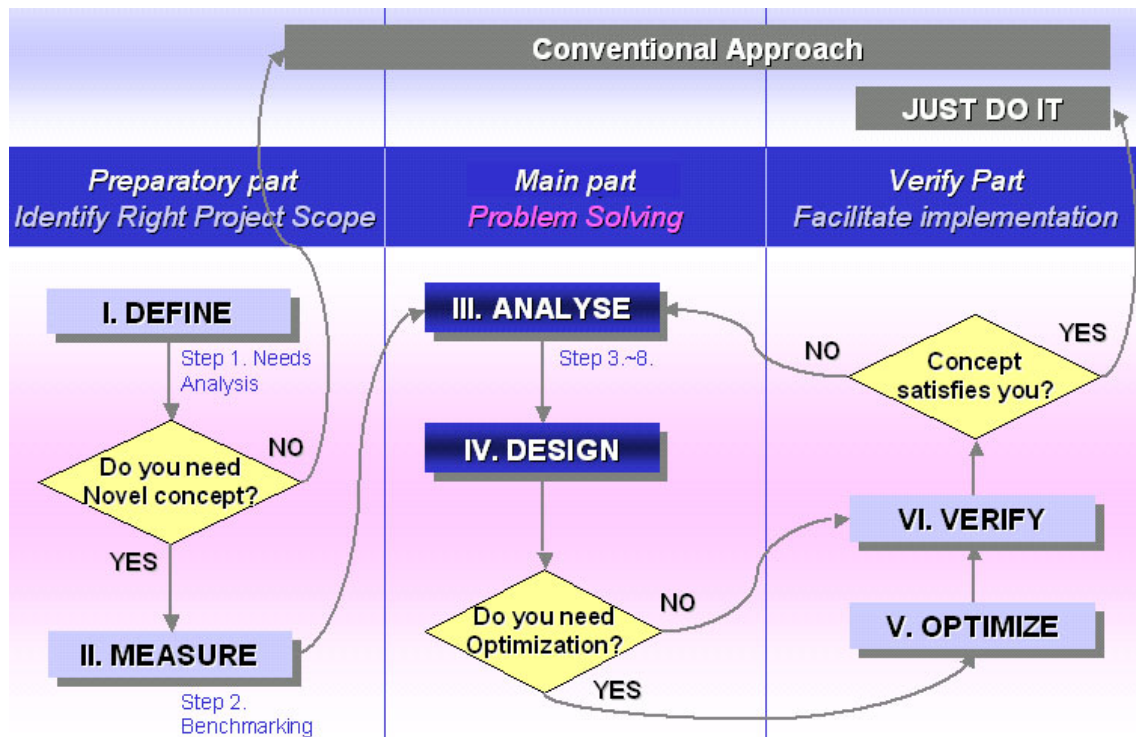


Figure 2. Typical execution process of TRIZ activity following DFSS roadmap in SAIT

“DEFINE” part of DFSS devotes to disclose of the reasons of problem solving – why we should solve this problems, and “MEASURE” phase dedicates to describe prototype system measurements, namely, what kind of existing prototype systems we have and how to measure the goodness of developed prototype, which comprises initial situation description. These parts could be considered as one of the way to choose appropriate scope of prototype system that will be analyzed more deeply.

Note, that in early versions of ARIZ (ARIZ-71, 77)(1), such define part was preserved but that part disappears in more modern version of ARIZ, for example, most developed version ARIZ-85-C(14).

In DFSS after “ANALYZE” and “DESIGN” steps, which are main parts of problem solving, we move to “Optimize” and/or “Verify” phase. If there are minimal problems in selected

prototype, we can consider this prototype as a concept of solution and in these parts we will try to develop the concept in real project execution. Above roadmap is mini-roadmap when concept generation activity is executed as independent module for comprehensive DFSS project. If there is no need to develop novel concepts, we can follow the conventional ways of DFSS that is well known way for SAIT researchers.

Let's consider more detail the main contents of beginning steps.

Step 1. Needs Analysis

Before start of resolving any real problem, we try to identify *future* customers and their wishes, so called "Critical To Quality" (CTQ). We consider TRIZ as a concerned with "Design for Six Sigma (DFSS)" part of TRIZ-activity in SAIT. We follow the same logic of usual DFSS. Needs analysis (they called it as "DEFINE" phase) is the first step of usual DFSS logic scheme (12). Our contribution to hybridize DFSS and TRIZ has been introduced before (10) and will be discussed in other form.

The reason of needs analysis is clear. Most engineers have strong mental inertia about who their customer is and what their customers want. If an engineer tries his/her best to design perfect floppy disc even in 2004, we would think this person is mad. Even though our customers don't want a system, some engineer insists to produce the system. This is why needs analysis is the indispensable step. Needs analysis should be done to avoid similar stupid situation in real project planning before detailed problem solving steps.

We should keep it in mind that the strongest mental inertia is mental inertia about customers and their needs. To overcome mental inertia, the different tools of TRIZ (for example, "Multi-screen vision", "Size-Time-Cost" operator and so on) must be applied together with regular DFSS tools.

Rule of thumb: Before resolving problem, ask why should I resolve that problem and who will use the results of our problem solving and what properties he or she wants or doesn't want. (11)

Step 2. Benchmarking Prototype Systems

The next activity of our researchers is benchmarking existing prototype systems that have been developed to satisfy specific needs of our customers. If we investigate every

element in prototype system and its functions, we can pick up the weakest part/element of current prototype. This is the reason why our customers are not satisfied. Benchmarking proto systems provides us two different benefits. One is more detailed information about what properties make our customers satisfied – i.e. extension of needs analysis. The other is a reasonable criteria and goal what/how much level we should meet. Table 2 shows typical results of benchmarking prototype systems. From this table we can evaluate each prototype and select which of prototype systems seem to be more promising than others.

Table 2. Typical example of benchmarking for proto systems

Benchmarking criteria	Target	Priority	Technology 1	Technology 2	Technology 3
CTQ-1: Theoretical efficiency	↑	5	+	S	+
CTQ-2: Manufacturing cost	↓	9	-	S	S
CTQ-3: Electricity production cost	↓	9	-	S	+
Novelty	↑	5	S	S	S
Number of '+'			1	0	2
Number of '-'			2	0	0
Weighted score of '+'			5	0	14
Weighted score of '-'			18	0	0
Total sum			-13	0	+14

“+” – means good (score is “+1”),

“-“ – means bad (score is” -1”),

“S” – means neutral (score is “0”)

Suppose that we put our goal as to design and produce excellent fuel cell (it is just an example, no relation with specific projects in SAIT). We know the main function of fuel cell – change chemical energy to electrical energy. There are some systems having the same main function. We call these systems as existing “prototype systems”.

We will collect data about each system, and evaluate which system seems better than other systems, following the conventional benchmarking guideline (13). After this preliminary evaluation described in above table, we can select the most promising

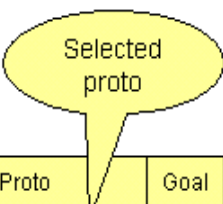
system to meet our goal. We will call this system as “*selected prototype*”. This selected prototype is the base for further deep analysis, which will be proceeded in step 3~8.

Fortunately in almost all cases we can pick up the prototypes from the past (in accordance to history of technical system evolution), because only God can create something from nothing.

After benchmarking current prototype, we can select the main one that we will try to overcome, which is usually a prototype that has high potential to develop. Different TRIZ parts (Laws of Technical System Evolution, S-curve, Criteria of Ideality and so on) could be added to regular DFSS procedure on this stage. Above activity will be executed under the name of ‘MEASURE’ phase of DFSS activity; it means we will “measure” our and our competitor’s technology potential.

Moreover, the scorecard to evaluate prototypes should be developed in this stage. Table 3 shows a typical scorecard for Prototypes evaluation developed in SAIT by following Pugh matrix (12). Note that sometimes a prototype can be considered as a concept of solution, but with some disadvantages about CTQ. The right vacant part will be used as evaluation format of developed concepts at the end of DESIGN phase and/or VERIFY phase of big roadmap denoted in Figure 2.

Table 3. Concept evaluation scorecard with including of benchmarking information



			Proto			Goal	Concept			
Concept evaluation criteria	target	priority	proto 1	Proto 2	Proto 3		Concept 1	Concept 2	Concept 3	Concept 4
CTQ 1 – resolution	Max	9	-	-	+					
CTQ 2 – speed	Max	7	s	s	-					
Manufacturing easiness	Max	5	+	+	+					
novelty	Exist	5	s	s	s					
Cost	Min	7	s	s	s					
Number of +			1	1	2					
Number of -			1	1	1					
Weighted sum of +			5	5	14					
Weighted sum of -			9	9	7					
Weighted sum of total			-4	-4	7	10				

Even though Step 1 and Step 2 have been described very briefly, these steps are extraordinarily important steps for problem statement and further resolving. If we have chosen wrong customers, wrong CTQ, it is very hard to expect we will succeed. Needs analysis clarifies why we should solve specific problems and what properties should be enhanced.

Step 2 is also an important part of problem statement. Of course, we cannot always find out the appropriate “existing” prototype. In this case it is inevitable to suppose “an imaginary prototype”, which doesn’t exist in reality (but theoretically possible). But in this case it will be necessary to answer the additional questions: why such a system does not exist and what is the obstacles for it realization. Anyhow, before we settle down the interactive pair of objects, that we will deeply analyze, we should choose the most promising “existing/imaginary” prototype. If there are no appropriate comparable prototypes, it will take a long time to analyze detail of existing system only. Promising prototype can reduce the time of analysis about good/bad features for our customers. In real practice, to choose most promising prototype helps us to understand customers, the kind of their needs, and the level of their needs.

Step 1 and Step 2 can lead us to identify right project scope and provides us compatible links between conventional DFSS and TRIZ.

Rule of thumb: *It is impossible to go ahead without deep knowledge about past. Before resolving problem in real situation, it is necessary to know “specific systems”, its evolution and to select the “best prototype”. “Best prototype” is not resolving of problem, but it allows us to estimate exact goal of our project and create the evaluation criteria for future concepts development.*

Instead of conclusion.

Usually, there was more than one existing prototype on the project initiation stage. After selecting most promising prototype, we could excavate sets of valuable problem models and try to resolve them. Of course, we clear understand that prototype’s estimation criteria, suggested in the article, looks relatively primitive. But we consider that it is better then nothing

Our logic is very effective to generate problems models iteratively – really, we always keep in the mind about set of prototypes, and if we failed in resolving problem with most promising prototype, we can come back to the closest one. This way allows us to develop not only high level concept design, but also introduce the detailed concept design for system, manufacturing process and probable failure of system. i.e. problem flow technology (2).

Now our logic is fully integrated into the workflow of SAIT project planning stage and also integrated into workflow of DFSS successfully.

We expect that our logic chain will be helpful for everyone who is suffering to clarify the initial situation in his/her projects.

Detailed description of the remained Step 3~8 and the verification result will be continued in the next articles.

Acknowledgement

Special thanks to Dr. N. Shpackovsky and all colleagues in six sigma innovation team in SAIT.

References

1. <Creativity as an Exact Science>, G.S.Altshuller, Translated by A. Williams, 4th printing, Gordon and Breach Publishers Inc., 1998
2. <Seminar of Introduction of OTSM-TRIZ>, N.Khomenko, SAIT, Sep., 2002
3. <TRIZ technology of conceptual design>, Presentation material of 5-day TRIZ seminar, Z. Royzen, in Samsung Electronics in Korea, 1998.

4. IWB™ v.2.8.0, Ideation International Inc.
5. TechOptimizer™, v.3.0, 3.5, 4.0, Invention Machine, Inc.
6. “Ideation TRIZ methodology One-day orientation”, A.Zusman, B. Zlotin, TRIZCon2004, Seattle, April, 2004
7. <Potential failure mode and effects analysis – reference manual, 2nd edition>, Chrysler corp., Ford motor company, General Motors Corp., February, 1995
8. “TRIZ in the Process Industry”, Gert Poppe and Bart Gras, Presented at ETRIA World Conference TRIZ Future 2001, Held at Univ. of Bath, Bath, UK on November 7-9, 2001.
9. <Effective Innovation>, pp27-103, D.Clausing and V.Fey, The American Society of Mechanical Engineers, New York, , 2004.
10. “TRIZ in Samsung R&D Process”, M.J.Song, V.Leniachine, S.H.Cheong, TRIZCon2004, Seattle, April, 2004
11. “Mapping the Innovation Space One: Novel Tools for Problem Definition in Product Innovation”, By: Barry Winkless, Dr. John Cooney, TRIZ-Journal, July, 2004
12. <DFSS guideline> (in Korean), company private document in SAIT, 2004
13. <http://www.bsi-global.com/Small+Business/Library+Resources/strategy/ST4BENCH.pdf>
14. <Engineering of Creativity>, pp314-330, S.D. Savransky, CRC Press, 2000.