

The Missing Link

Before we continue our story, one concept in particular requires explanation: my use of Goldratt's TOC as a basis for SKI's TOC.

In its simplest form, Goldratt's TOC asks three questions:

1. What to change?
2. What to change to?
3. How to cause the change?

As you will recall (Please, tell me that you have these items memorized!), my seven tips are the keys to making something happen. Often, things begin happening from nothing more than an idea:

1. Write down your Goal.
2. Make a List.
3. Visualize the End Result.
4. Develop a Project Plan for Success.
5. Involve Others.
6. Find a Working Model and Adapt It!
7. Create a Feedback Loop: POOGI.

Okay, I "stole" Goldratt's POOGI (Process of On-Going Improvement) concept and molded it into my own.

It provides the best environment for synergy that I have ever experienced. Remember that one plus one can equal twenty-seven? Or as I suggested, with TOC as a catalyst, it could even equal more than 27,227.

There are at least two applications in real business life that lend themselves well to the melding of these two common sense approaches (Goldratt's Theory of Constraints and SKI's Throughput on Command).

I still don't "get" how you arrive at 27,227 and there may also be others who don't.

—Thayer

Good point. I have been using the term "synergy" since 1979 to describe how results can be increased through cooperative action. In Marine Corps boot camp, we had to make our bunks up every morning. If everyone did their bunk separately, it would take, as I recall, about three minutes. Instead, we were taught to "partner" with the recruit next to us and share in the making of both bunks. It took only slightly longer than one minute to make two bunks as a team. My point here is that by adding in TOC, it might be possible to make 100 bunks in less than ten seconds. I'd have to do an experiment to be sure, but what I'm trying to illustrate is that TOC can increase results dramatically.

—SKI

One application is the business startup.

My favorite. Your goal is to create a new venture.

The first two of Goldratt's questions relate to determining what you should do as the first of my seven tips. Make sense?

Not sure?

When I set out to create eDivision.net LLC, I asked myself, "What to change?"

I wanted to get back to being self-employed. But, the internet was changing everything. So, what I needed was a way to profit from the internet's explosion. What was my expertise? What contacts did I have in that marketplace?

As Zig Ziglar says, "I am about to say something profound":

Find the weakest link, and fix **ONLY** the weakest link.

I believe Dr. Deming died before he had the chance to make this point extremely clear to everyone jumping on the TQM bandwagon. When I worked for the State of Ohio, we were busy rolling out TQM. Management mounted a huge effort, training almost every employee in the Department of Transportation in TQM. Then, they set us all loose.

99% of the effort was wasted (hence, there was a jump to the “reengineering” trend as soon as it became the new “best thing”). Why was most of it wasted? No one identified the weakest link. Please tell me that you recall that ALL effort spent fixing any link that is not the weakest link, is wasted?

What to change?

For eDivision, my challenge was how to add qualified staff (I could not afford to hire an army of techies), and the problem needed to be solved before eDivision could ever make any serious money. Sure, it started out as a business with me as the lone hired gun. But that business model was limiting. Using it would mean eDivision would never become lucrative enough to create financial freedom for me.

However, with the advent of the “Open Source Movement,” the best of the best techies wanted the freedom to work for themselves. eDivision could exploit this new trend and attract necessary talent on an as-needed basis without paying ongoing salaries to permanent employees.

What to change to?

I decided that eDivision was to have no employees. I was a partner in the LLC. All techies who ever put finger to keyboard for eDivision were to be subcontractors. The problem was solved and eDivision began with a solid plan for growth without risk.

How to cause the change?

By using my seven tips, I could cause the change. Also, it was the Open Source Movement and the success of Linux, Apache and other similar initiatives that created a working model for me to adapt (steal). I merely expanded the opportunities for techies who had the skillset I needed for any given project.

You adapted an Open Source model as well as Goldratt's concept?

—Thayer

Yes. Remember that I use Goldratt's TOC within my own SKI's Throughput on Command. So, when I applied my seven tips to eDivision, the model that I used in tip six (find a working model to steal) was the Open Source Movement.

—SKI

Another excellent application for a combination of Goldratt's TOC and SKI's Throughput on Command is to use it to correct something that is flawed, less than perfect, or broken.

Another application is an ongoing situation that needs to be fixed.

Lets take, as an example, the Silicon Valley company engaged in “workforce procurement” that I mentioned previously. They had already launched. They had a business plan, some key players and a number of contacts in their field of endeavor. So how did I apply SKI's TOC?

First, what needed to be changed?

They wanted a web database. It would provide a platform they could use to expand their business plan with real facts and figures about the technology required to make their dream come true. They wanted me to first create a “sample” of the database I would later develop and they gave me a budget to do so.

However, once I examined the scope of what was needed, I determined that they could have a real working solution for the same amount of money that they expected to pay for a small scale prototype—so I went ahead and

created it. To say that they were pleased with the results and the savings is a major understatement. In the last year or so, as other software developers have discovered the same robust toolset I used, prototyping in this area has become almost obsolete.

Second, what to change to?

Again, the Open Source Movement made this all possible. If you haven't already done it, please reconsider buying (and reading!) *Under the Radar*. By using the proper tools, tools that allowed me to create a scalable web database using industry standard products, I changed a company's hope for a prototype into a final solution.

Third, how to cause the change?

I caused change by using my seven simple tips. The dream of a real working solution was easy to write down. I then made a list of the project requirements and jotted down some ideas of how to address them. I even had a list to track all the lists of things that needed to be accomplished. Visualizing the end product was easy for me in this case because I had previously built similar projects on a much smaller scale. I had also built parts of a much larger system.

Developing the plan for success was also easy. The startup company had a brilliant designer. His user interface was simply the most functional web interface that I have ever seen. From his vision of the "front end" user interaction, I was able to make the "back end" server perform beyond all expectations.

The list of others that we had to involve was quite long and varied. The JavaScript guru was one of the most unique. A friend of a friend knew this guy who knew a guy... but we found the answer we needed. Actually, he gave me a prototype that did "XYZ" and it gave me enough insight to make the interface do "ZYX" instead.

Looking ahead, while you await your copy of *Critical Chain*, here is a deeper look at Goldratt's TOC in the form of his five focusing steps:

1. Identify your system's constraint.
2. Decide how to exploit that constraint.
3. Subordinate all throughput to that decision.
4. Elevate the constraint (revisit it until it is no longer the weakest link).
5. POOGI (Process of On-Going Improvement) by going back to step #1 and starting all over on the the new weakest link.

Here's the bottom line: Use Goldratt's method to define your dream (the first step in SKI's TOC) in terms of "what to change" and "what to change to."

Then, use my other six tips to "cause the change"!