2015 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
SYSTEMS ENGINEERING (SE) TECHNICAL SESSION
4AUGUST 12-14, 2014 – NOVI, MICHIGAN

Creating Innovation within a Structured Systems Engineering Framework

Kevin J. Dutcher

Maneuver Collaboration Center (mc2)
www.gdls.com/mc2
General Dynamics Land Systems
Sterling Heights, MI
dutcherk@gdls.com (586) 718-4348

Abstract. A necessary, but not sufficient, condition for innovation is that it be different. Given this, a technique is proposed to develop innovative solutions at each step of a systems engineering based product development effort. This technique, while not guaranteeing results, allows ventures into innovation which can be planned, scheduled, and measured.

GENESIS OF THIS PAPER

Don't constrain me. I got tired of hearing people complain that a structured development process kills innovation. At it's heart, structured processes take care of the repeatable part, so more time and energy can be spent on the new, unique, harder aspects of the job. I decided to see if I could take the very notion of a structured development process, and bake into it, a simple way to generate innovation.

DESIRE FOR INNOVATION

The better mousetrap. The global economy that we are all operating in has turned many products into almost pure commodities, where price is the only differentiator. This type of market erodes brand loyalty, profit, and employee enthusiasm while rewarding only the low cost producer. The way out of this type of market is to build a better mousetrap, so, as the saying goes, the world will beat a path to your door. Deliver to the customer a new function, a new splashy colour, a different user interface, i.e. deliver ***innovation*** to the customer, and watch brand loyalty, profit, and employee enthusiasm soar.

Go forth and innovate. An Internet search on the word innovation returned over 120 million hits. On Amazon.com, over 64,000 books relate to innovation. Clearly, innovation is in the air, but talking about it isn't doing it. Proof by repeated assertion is no proof at all. Telling the troops to go forth and innovate without baking it into your process, without permeating it through your corporate culture, without dedicating resources to it, and without measuring it, will get you no innovation at all.

DEFINITION OF INNOVATION

Wow, that's cool! Innovation, as pointed out on the INCOSE (International Council on Systems Engineering) email discuss list, is solely in the eye of the beholder. The beholder may be the end customer seeing an iphone for the first time, an employee on the shop floor learning a new process, or the Hyundai car salesman selling a car priced far below the competition. People interacting with innovation want to look at it, interact with it, talk about it, and show it off. But what is it? What is innovation?

By definition, an innovative product is one that stands apart from the crowd. It could be due to differences in performance, functionality, cost, mass, aesthetics, or any number of reasons. A necessary, but not sufficient, reason it appears innovative is because it's different. It distinguishes itself in one or more categories. In the rest of this paper, a method is proposed that gets the innovation engine running by first generating ideas that are different from the norm. The technique works by merely spending a bit more time on each of the product development tasks within a structured systems engineering framework.

SYSTEMS ENGINEERING PROCESS

The Basic SE Framework. The specific systems engineering process used for product development at your company or within your industry may vary from the generic process that will be used for this paper, but the methods will be transferable. The basic systems engineering framework for this paper consists of the following tasks:

1. Identify external interfaces
2. Define functionality and performance parameters at the external interfaces
3. Create a functional architecture
4. Select technologies to use
5. Create a physical architecture
6. Allocate functions to physical architecture elements
7. Allocate attributes (i.e. cost, mass, and volume) to physical architecture elements

The tasks do not necessarily happen in the order depicted above, and are most likely, iterated numerous times. These tasks then get repeated for each lower level element of the detail of the item under analysis / design.

THINK DIFFERENT

The Secret. TRY SOMETHING DIFFERENT. That's it. That's the entire thrust of this approach. Look at the various design views of you system, and TRY SOMETHING DIFFERENT.

We will look at the seven steps in our basic systems engineering framework and show how this secret process can be applied, to separate your product from the competition.

1. Identify external interfaces. This step is frequently referred to as "setting the boundaries" or "framing the problem". In many instances of contracted development, typical of government contracts, these external interfaces will be already identified for you. There may be little you can do, other than bringing ideas up to the contracting officer. Regardless, here's the, TRY SOMETHING DIFFERENT process applied at this step.

The identification of external interfaces typically is graphically shown as a context diagram, Figure 1. It's a nice, recognizable method of documentation that puts the item under development at the center of the universe, surrounded by its physical interfaces.

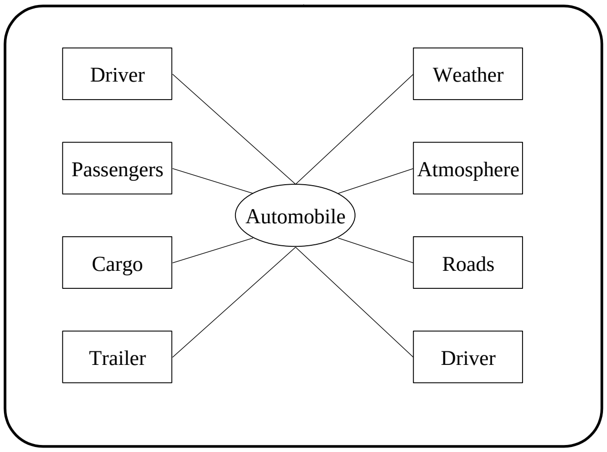

Figure 1. External interfaces are typically identified on a context diagram.

What is not readily seen in this format is the relationship of the identified external interfaces to the rest of the world. While this step identifies what is included in the interface set, in also defines what is excluded, namely, the rest of the world, as depicted in Figure 2. And, it's a good bet that somewhere out there is a new external interface that will make your product innovative.

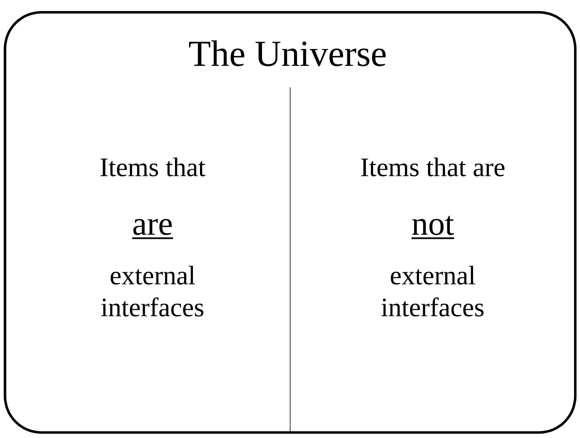

Figure 2. Identifying external interfaces also excludes possible interfaces.

TRY SOMETHING DIFFERENT has two different steps to try and generate innovation:

1. Review the "normal" external interfaces and try to eliminate each one separately or in combinations
2. Try to add new external interfaces

Without having to create anything other than the standard context diagram, you can now generate potential innovative ideas. For eliminating interfaces, you can either use the brute force method of creating a matrix showing all possible combinations of valid external interfaces and evaluate each one or explore the idea of deleting external interfaces in a less structured brainstorming session. In 1930 you could have imagined a telephone that doesn't interface with the cord. It immediately would have triggered thoughts of tying a telephone to a 2-way radio, and we could have had cell phones a long, long time ago.

For adding new interfaces, you are limited only by your own imagination. Observe your products in use, and look around the environment at every object, evaluating it for a possible interface. Walk through a department store and ask yourself how you could interface with every product in the store. Or maybe interface with the store itself. Surf the net, looking at web sites related to your industry, and study the items that are offered for sale. Set up a brainstorming session, with all kinds of novelty items on the table to free their minds, and generate a list of items to interface with. Tap into people familiar with your product. Visit people who highly modify your product. Ask the marketing department. Talk to people who repair your product. There is literally no end to the list of items you could evaluate. This step of identifying the external interfaces sets the foundation for the rest of the product development process, and is very often neglected in high carry-over programs. Who ever thought that a refrigerator should interface with the internet? A simple app can now notify you at work if the fridge door was left open by your kids.

Clearly, TRY SOMETHING DIFFERENT applies very simply at the External Interface step.

Merely stopping after the context diagram is made, and going through the TRY SOMETHING DIFFERENT exercise could launch a new market segment.

2. Define functionality and performance parameters at the external interfaces. A product's observable behavior occurs at its external interfaces. This behavior, the transformation of inputs to outputs, is termed its functionality. What the product does. A technical specification is typically used to document the functional requirements (what must the product do) and the associated performance parameters (how well it must be done). More and more frequently, a CAE (Computer Aided Engineering) tool is used, such as STATEMATE, to create a model of the product that is being developed, and the functional and performance requirements may be embedded directly in an executable model.

TRY SOMETHING DIFFERENT at this step of the product development process, when its functionality is being described, uses one or more of the following techniques:

- Try a different or additional function at each interface
- Move a required function to a different external interface
- Delete a function
- Add new functions
- Raise a performance parameter beyond best in class
- Lower a performance parameter to near zero
- Explore if functions can operate simultaneously which couldn't before (referred to as States and Modes)

This step is typically done very iteratively with the previous step, identifying external interfaces, and rightly so. Identifying new external interfaces can trigger new functions to add and vice versa.

3. Create a functional architecture. The functional architecture step is where you decide how you're going to transform the inputs to outputs. The top level functions get broken down into smaller, more manageable pieces. Although the decisions made in this step are not usually directly visible to the end-user, it does set the stage for the remainder of the product development process. TRY SOMETHING DIFFERENT at this step of the product development process uses one or more of the following steps:

- Try different top level functions
- Develop functions from the output backwards towards the input
- Break the functions down in a different manner
- Investigate functions that everyone knows just won't work
- Eliminate those functions that are currently costly or difficult
- Ask people with different engineering specialties, i.e. mechanical, hydraulic, software, how they would transform the inputs to the outputs

Some companies refer to the top level functions as the concept "strategy", and don't refer to the term function. In doing this, they entirely miss the benefits of viewing the problem space and the solution space in the functional domain, and walk away from keeping the Product Development Process grounded in science and logic.

4. Select technologies to use. As companies try to reduce cost and cycle times, they are increasingly looking toward off-the-shelf parts. Inherent to these part selections is the decision to utilize the technology contained therein. In highly integrated companies, there may be a tremendous amount of capital investment in a technology. In other cases, there is considerable investment in terms of the combined knowledge and experience of the employees.

TRY SOMETHING DIFFERENT at this step of the product development process uses one or more of the following steps:

- Consider using TRIZ or Goldfire, two software tools that catalog technologies with respect to functions
- Review other industries with similar functional requirements
- Patent searches
- Your R&D department
- Review trade journals
- Ask the "new" guy
- Ask the "old" guy for any inklings he's ever had on a different approach
- Consider what you would do if your favorite technology was banned, i.e. asbestos
- Try eliminating the expensive technologies
- Try incorporating inexpensive / recyclable / green technologies

Selecting technologies gives you a chance to look at new suppliers, new manufacturing options, and new materials as well as innovative solutions for your customers.

5. Create a physical architecture. The decisions made during this step are to identify the next lower elements and their interrelationships. How do you TRY SOMETHING DIFFERENT at this step? Again, just do things differently.

- Try one big part
- Overlay your competitor's physical architecture on your system
- Use lots of small, inexpensive parts
- Separate parts by engineering discipline
- Mimic something in nature

Typically, parts are created by trying to minimize interface complexity and along contractual lines. Other usual considerations are manufacturability, modularity, shipping, and serviceability. Many Design for X methodologies, such as Design for Manufacturability are useful in this time frame.

6. Allocate functions to physical architecture elements. Allocation of functions to the elements of the physical architecture is usually done very iteratively with the prior step, creating the physical architecture. The process basically follows the Scientific Principle, propose a hypothesis (physical architecture and allocated functions) and try to prove that it works, via analysis and test. If it doesn't work, either the physical architecture or the allocation of functions is changed until it proves feasible. The performance parameters are also allocated in this step. DFSS (Design for Six Sigma) and Robust Engineering are techniques employed in this timeframe. TRY SOMETHING DIFFERENT at this step can use the following techniques:

- The brute force morphological method is to try to allocate each function to each physical element and determine how good of a system can be built around each concept
- Allocate the functions similarly to your competitor's product
- Allocate the hardest function to the least expensive element
- Try breaking all the “rules” that say it can’t be done this way

As the physical architecture was developed with the functions in mind, innovations in this area will probably be done during the iterative process described above.

7. Allocate attributes (i.e. cost, mass, and volume) to physical architecture elements. There are many system-level attributes which get mathematically allocated to lower level components, these include cost, mass, volume, and reliability. Targets for these values must be developed early, and variations of these is one way to drive innovation during the other steps. Comparing the percentage of allocation to the perceived value of the part is one way to evaluate the allocations. TRY SOMETHING DIFFERENT at this step include the following options:

- Move up or down to the next price point
- Zero out the allocation
- Allocate much more than thought needed
- Mix allocation levels, i.e. low cost/low mass, high cost/low mass
- Allow the part owners to "horse trade" amongst themselves, one attribute for another, i.e. trade $5/unit for 3 kg/unit

Allocating the attributes late will drive rework, so make sure this step is done early in conjunction with the others.

METRICS

Adding a line item, name it Innovation Investigation perhaps, in each major step of your product development process is a way to keep this effort visible. From there measurements can be initiated and tracked. You can measure the total number of ideas generated, the number of ideas investigated, the number of ideas pursued, the number of Innovation Workshops, etc. These metrics do two things. One, by getting management to start tracking the metrics means by default, that they are allowing the Innovation Investigation to occur in the first place. Secondly, they will find that a structured systems engineering based process doesn't constrain creativity, it sets it free.

SUMMARY

The ideas expressed here are to make you recognize that product development does have a generic structure, and you can "play" with the design at each step. Most of the creativity techniques used today first try to get you to "escape" from your current thinking and then try to stimulate your thoughts. This paper has tried to put those techniques in perspective within a repeatable, overarching product development process based on classical systems engineering concepts.

BIOGRAPHY

Kevin J. Dutcher is a Staff Specialist at General Dynamics Land Systems with over thirty years of experience. He has been involved with Systems Engineering in both the Automotive and Defense Industries, and is a member of INCOSE and IEEE. Mr. Dutcher earned a certificate in Design for Customer Value and Market Success from Stanford University, an MSE in Integrated Vehicle Systems Engineering from Purdue University, and a BSEE from Michigan Technological University.