COTS: What You Get
(In Addition to the Potential Development Savings)

James E. Long
Vitech Corporation
2070 Chain Bridge Road, Suite 320
Vienna, Virginia 22182
(703) 883-2270

Abstract. It comes as no surprise that incorporating an existing component into a system means you save in having to develop the component yourself—savings in schedule and development costs. What often is surprising is the impact that incorporating an existing component has on the system engineering process.

This paper reviews the system engineering process and shows where the incorporation of COTS (Commercial-off-the-shelf) and GOTS (Government-off-the-shelf) fits into the process. It discusses the savings you get from a COTS component as well as the cost of these savings. Also described are the issues of interfaces and life cycle maintenance and modification. Once you incorporate a COTS component, how do you handle the upgrades to the COTS? Before jumping in to take advantage of COTS products, take time to consider whether the system and design process are robust enough to handle the additional constraints.

Keywords. COTS, GOTS, Systems Architecture

INTRODUCTION

A goal of system engineering a system is to define the set of cooperating components that comprise the system and satisfy the originating requirements, including cost, schedule, and performance. An overview of the system engineering process is shown in **Figure 1**. The purpose of this paper is to discuss how the consid-

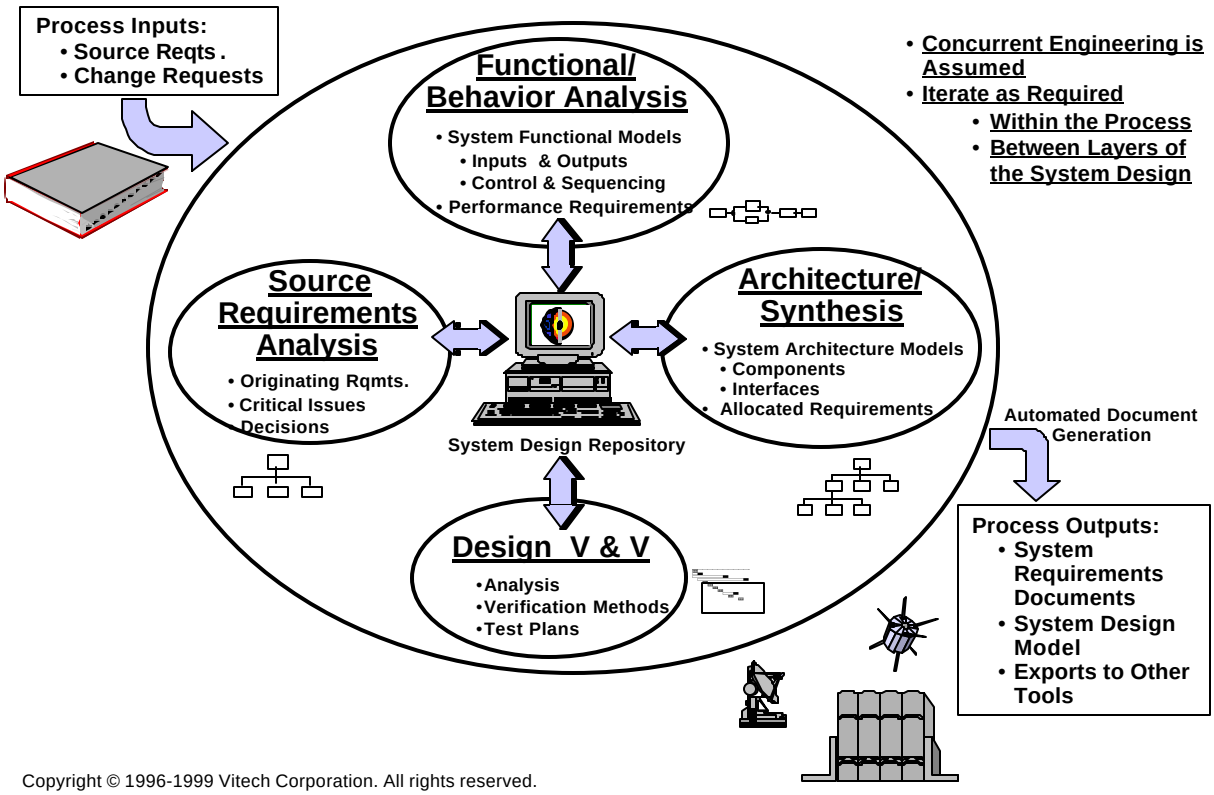

Figure 1. The System Engineering Process: An Overview

eration of COTS (Commercial-off-the-shelf) and GOTS (Government-off-the-shelf) system components impact the system engineering process. The emphasis here is on the top-down and middle-out process, since the reverse engineering of an existing system starts with a system consisting of legacy components.

The consideration of COTS has received added emphasis in recent years because of the potential to significantly reduce the time-to-market and cost-to-market of a system under development. However, this approach does not come for free. Since COTS elements already exist independent of your system, they were probably built to satisfy a set of requirements that may vary widely from those you need for a given component of your system. Matching the COTS interfaces, behavior, and life-cycle characteristics to your system may be challenging.

BACKGROUND AND CONTEXT

In order to put the subject in context, let's look at where the COTS element affects the system engineering process as shown in **Figure 1**.

The COTS element is involved in the Architecture/Synthesis activity where the physical architecture of the system is defined. The physical architecture is comprised of those hardware, software, and human elements and their interconnectivity that define the deliverable system. **Figure 2** shows the results of the activities in each domain.

DECIDING TO USE COTS

The decision to use COTS components can arise in two ways: it can be given as an original requirement from the customer, or it can arise later, as you are adding layers of detail to your physical component hierarchy diagram. In the case where the decision to use COTS is directed from the customer, it may appear to some that the inclusion of the COTS component in the system engineering process requires a major change to the process, but this is not the case. Looking at the system engineering process as shown in **Figure 3**, this capability has always been provided in *Activity 3, Capture Originating Architecture Constraints*. The requirement to use a specific COTS as a component in the system is viewed as an Originating Constraint from the customer and is handled just like other originating architecture constraints such as a specific printer or specific word processing tool.

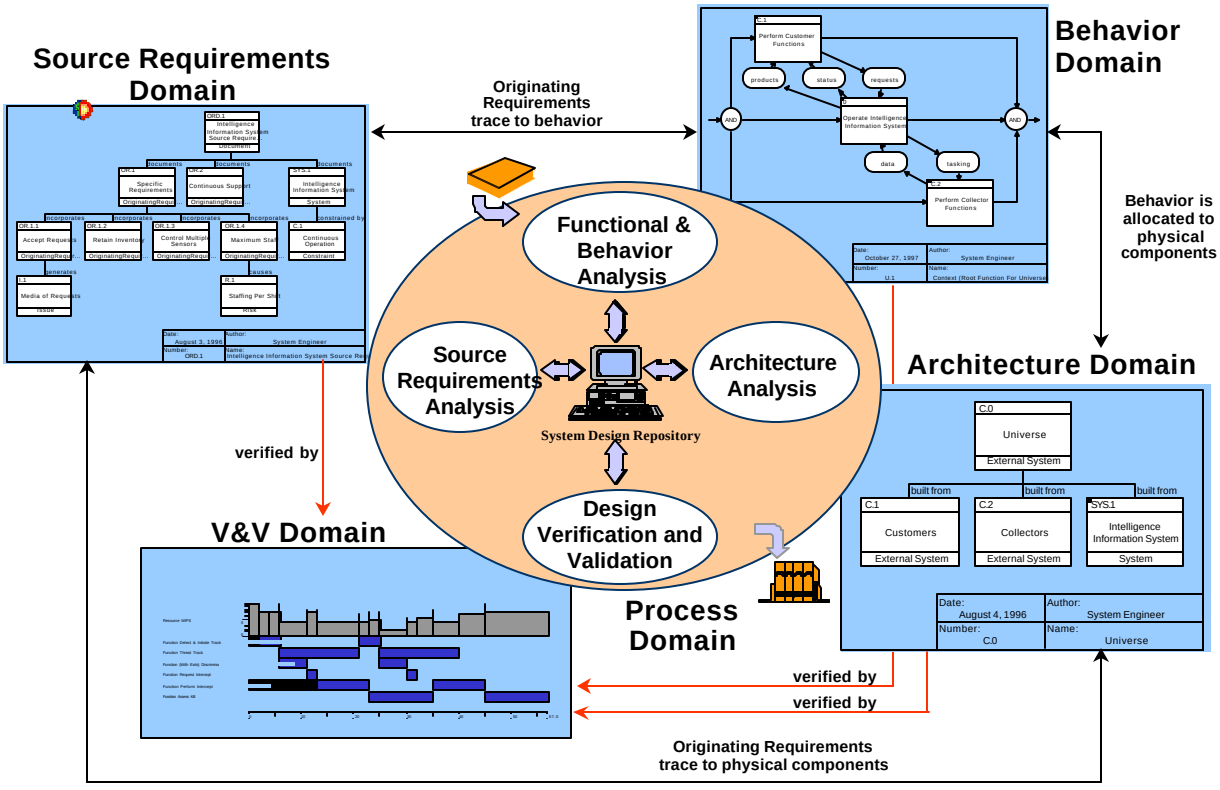

Figure 2. The System Engineering Process: The Domains

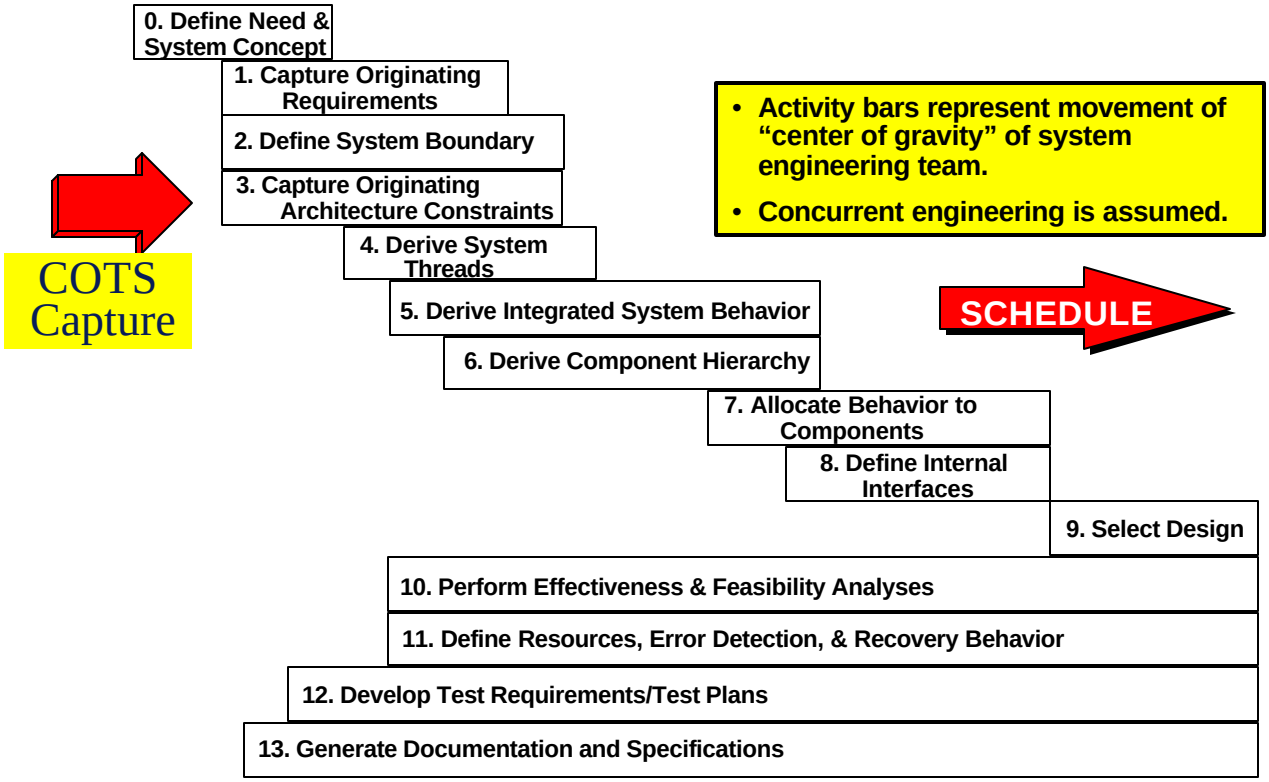

Figure 3. Where a Requirement for COTS Fits On The Road Map

Often the decision whether to use COTS isn't an original requirement and doesn't arise until you are adding layers of detail to your physical component hierarchy as represented by *Activity 6, Derive Component Hierarchy* of **Figure 3**. Each layer of the component hierarchy represents concurrent engineering activities. Beginning with Layer 1, as shown in **Figure 4**, you add detail to each of the four primary activities and you do not go to the next layer until the current layer is completed and verified.

As you add components to each layer of the decomposition, you make a develop-or-buy decision for each component. If there is an existing element that fits the needs (functional, physical, cost, and political) you have the option to use it as a part of your system.

Figure 5 shows an example of the physical architecture for one layer of the Automated Personal Assistance System (APAS) system.

As you develop the physical architecture and select or identify those components that could be satisfied by a COTS product, you have the potential to save development schedule and cost by eliminating the need to further specify and develop that element in-house.

WHAT IS PROVIDED?

A commercial product (COTS) or government product (GOTS) comes with two predefined features: the physical interface and the behavior (stimulus-response characteristics).

The physical interface of the COTS has already been defined and implemented, and your system must be designed to integrate with it. Modifying the COTS interface to match your design violates the concept and reason for using the COTS component in the first place.

In addition, the COTS behavior has already been defined. Its response to each stimulus and combination of stimuli is already implemented. Modifying this behavior modifies the COTS product, which would not result in the desired development schedule and cost savings. Although, since the documentation of the COTS may be less than complete, it's possible you may be required to analyze and document this alien component.

You want the COTS product to be just as much an element of your system as if you had designed and developed it. This means that the test plans should be

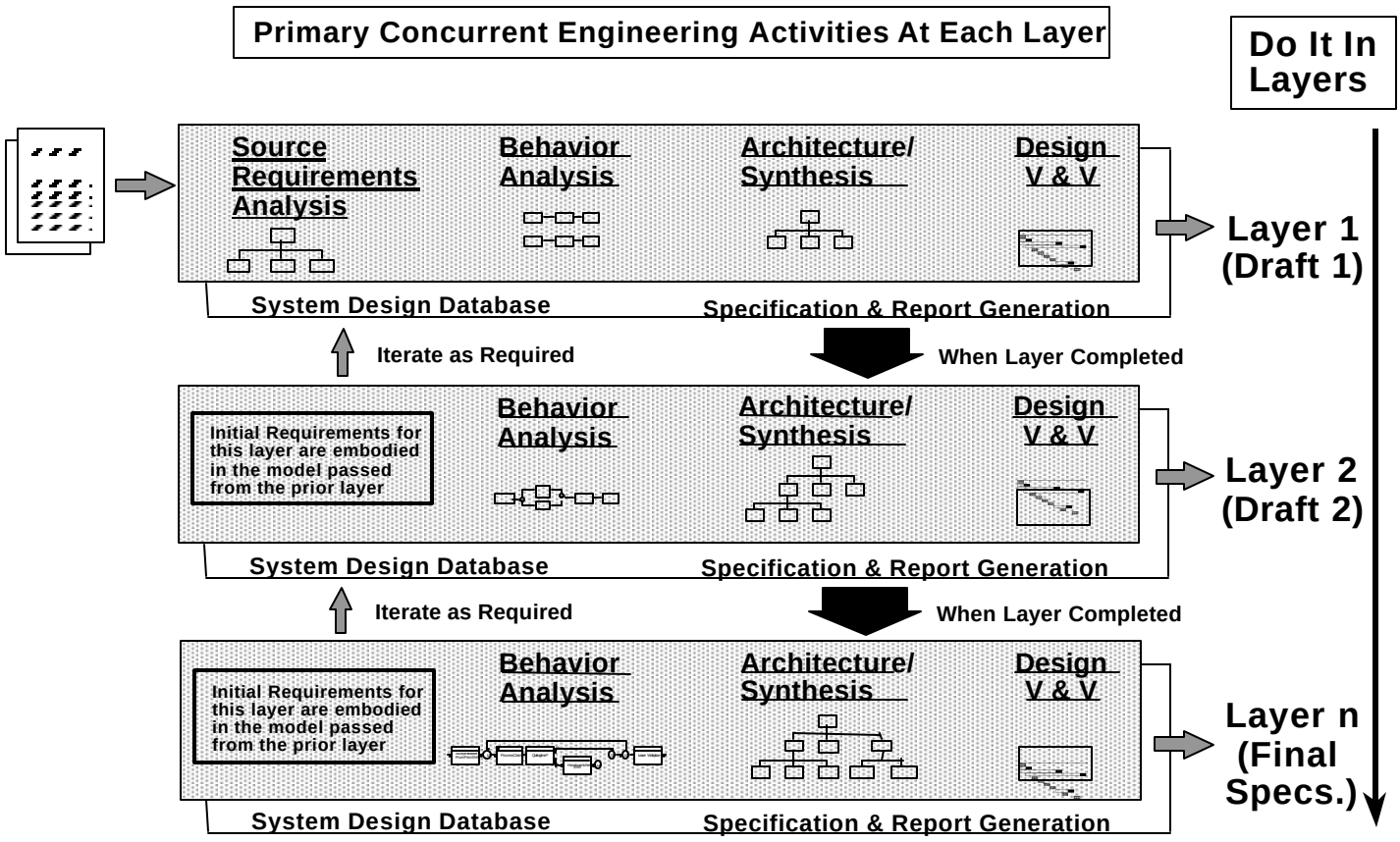

Figure 4. The Onion Model. Do Your System System Engineering in Increments/Layers

expanded to incorporate COTS and to catch COTS inconsistencies. An ideal system consists of a collection of cooperating components, including any COTS.

LIFE CYCLE PROBLEMS AND OPTIONS

You should expect new releases of the COTS component to occur during the operational life of your system. These updates potentially involve changes to the interface or changes to the COTS behavior. Sometimes these changes are minor, but sometimes they can include a complete overhaul of functionality. You need to be prepared to deal with these changes should they occur.

Your engineering options to handle new releases of COTS components are to:

- Design enough flexibility into the system to absorb these future and unknown changes.
- Decide to not accept any future releases of the COTS. This means that you have an element in your system that does not have supplier support during part of your system's operational life.
- Plan Product Improvement Cycles in your product's life cycle for accepting COTS upgrades.

Whichever option you choose, dealing with new releases of the COTS component means increased Configuration Management and Logistics effort during the Operations and Maintenance phase of your system, especially if multiple copies of the system exist.

DESIGN PROCESS PROBLEMS?

If the COTS decision is made too early in the system design cycle, it may reduce the designer's ability to produce a "balanced" system. This is like earlier days in which the design process was crippled with a "Hardware First" philosophy.

OTHER OBSERVATIONS

As you decide whether to use COTS, keep in mind,

- The COTS documentation is usually not adequate or completely accurate, causing you to test/verify the real COTS interfaces and stimulus/response behavior

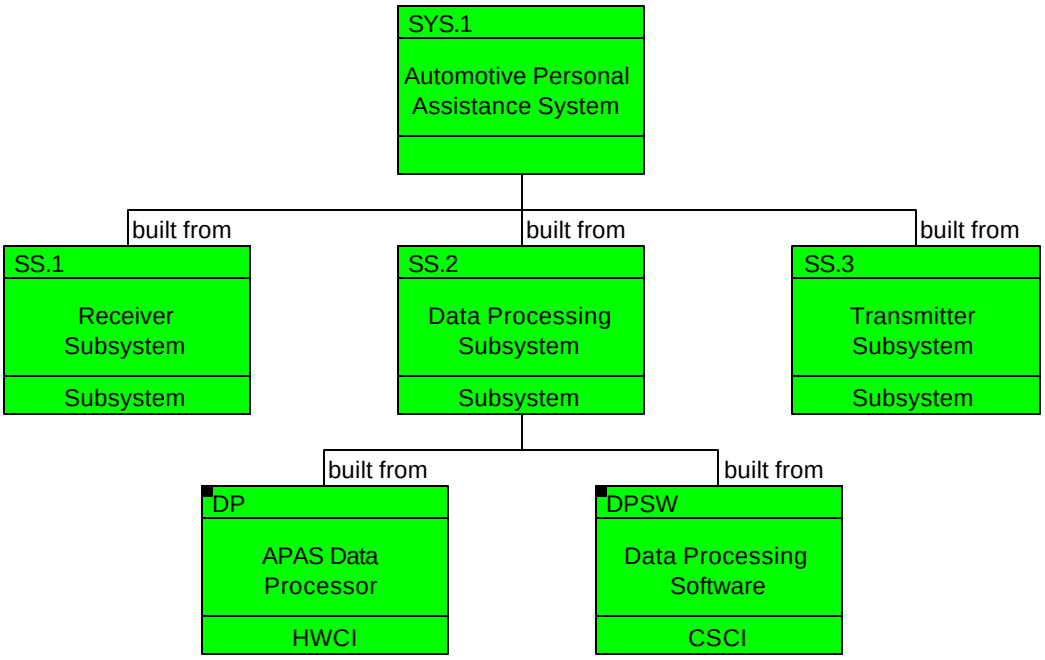

Figure 5. Component Hierarchy For The APAS

- The COTS product you need is often not released yet
- If you spend time and effort to modify COTS, it isn't COTS anymore

The COTS approach often results in shifting problems into future support in order to get reduced time-to-market now

CONCLUSION: SHOULD YOU USE COTS?

Yes! You should take advantage of COTS products. But make sure that the system and the design process are robust enough to handle the additional constraints. COTS alone is not the "silver bullet".

REFERENCES

Baker, Loyd and Long, James E. "Specifying A System Using ERA Information Models," *Proceedings of the Sixth Annual International Symposium of the International Council on Systems Engineering*, Volume 1, 1996, pp. 399-405.

Buede, Dennis M. and Long, James E., "System Engineering – The Complete Process," *A Tutorial for the Washington Metropolitan Area Chapter of the International Council on System Engineering*, (Vienna, VA, March 28, 1998).

BIOGRAPHY

Mr. James Long is the President of Vitech Corporation – the developer of the system engineering support tool CORE®. He has been a performing system engineer and innovator since creating the first behavior diagrams (then called Function Sequence Diagrams) at TRW in late 1967. He played a key technical and management role in the maturing and application of that system engineering process and technology at TRW. Both CORE® and RDD-100® are based on the TRW developments.

He is a member of INCOSE and served as vice-president and then president of the Washington Metropolitan Area Chapter, the largest chapter of INCOSE.

Mr. Long has been selected as an Eminent Engineer by Tau Beta Pi, the honorary engineering scholastic society. The Eminent Engineer designation is recognition for career achievement in engineering.

This page intentionally left blank.