Correspondence

The Difficulty in Distinguishing Product from Process

Christiano Danny Abadi[*] and Terry Bahill[†]

Systems and Industrial Engineering, University of Arizona, Tucson, AZ 85721-0020

Received 27 May 2002; Accepted 19 November 2002
DOI 10.1002/sys.10035

ABSTRACT

When engineers design a system, they must design both the product and the process that will create it. Accordingly, systems engineers must write requirements for the product and the process. Stating these requirements in separate documents might make it easier to get the requirements right and manage the requirements when either the product or the process requirements change. But, of course, these two sets of documents must be intricately interrelated, integrated, and produced with extensive feedback loops. This paper shows the results of an experiment that was designed to investigate the difficulty in distinguishing between the product and the process. © 2003 Wiley Periodicals, Inc. Syst Eng 6: 106–115, 2003

1. THE DIFFERENCE BETWEEN PRODUCT AND PROCESS

The product documents deal with the product, whereas the process documents deal with the design and development system, the testing system, the production and manufacturing system, the operating system, the maintenance system, the performance evaluation system, the customer service system, and the retirement and replacement system. Some authors [e.g, Buede, 2000] have argued that there should be a document for each of these systems. Boehm, Port, and Basili [2002] use four models: product, process, property, and success. However, in this paper we will only discuss two: the product and the process.

Which comes first the chicken or the egg? Or in this case, which should be written first, the product documents or the process documents? Neither. The two sets of documents must be developed in a parallel and iterative fashion. Figure 1 shows a general-purpose process that could be adapted for writing these two sets of documents. The most important point of the figure, for this discussion, is the large number of iterative feedback loops. The following is an explanation of this general system life-cycle model.

State the problem. Stating the problem is the most important systems engineering task. It entails identifying customers, understanding customer needs, estab-

[*]Present affiliation: Eastern University, St. Davids, PA
[†]Author to whom all correspondence should be addressed (e-mail: terry@sie.arizona.edu

Systems Engineering, Vol. 6, No. 2, 2003

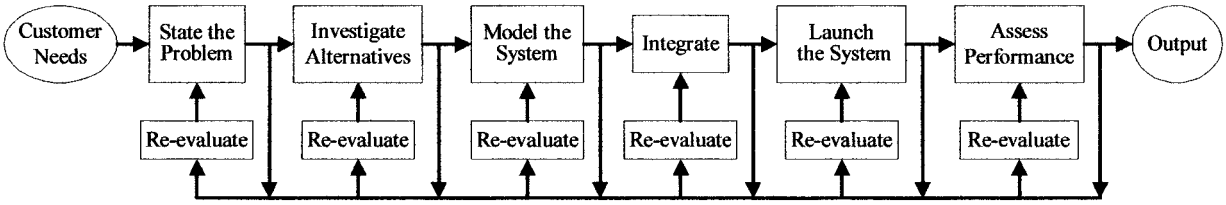

Figure 1. The SIMILAR process from Bahill and Gissing [1998]. The most important aspect is the multiple, iterative, parallel feedback loops.

lishing a need for change, discovering requirements and defining system functions. For a product, the main customer is the end user. For the process, the main customers are the stockholders.

Investigate alternatives. Alternative concepts are evaluated based on schedule, cost performance, and risk measures of effectiveness. For the design of complex systems, analyzing alternative designs reduces project risk. Investigating innovative alternatives helps clarify the problem statement.

Model the system. Models will be developed for most alternative product designs. The model for the preferred alternative will be expanded and used to help manage the system throughout its entire life cycle. Process models allow us, for example, to study scheduling changes, create dynamic PERT charts, and perform sensitivity analyses to show the effects of delaying or accelerating certain subprojects. Running the process models reveals bottlenecks and fragmented activities, reduces cost, and exposes duplication of effort. Product models help explain and improve the system. These models are also used in tradeoff studies and risk management.

Integrate. Systems, businesses and people must be integrated so that they interact smoothly with one another. Integration means bringing things together so they work synergistically as a whole. Interfaces between subsystems must be designed. Processes of coevolving systems also need to be integrated. The consequence of integration is a system that is built and operated using efficient processes.

Launch the system. Launching the system means running the system and letting it produce outputs. In a manufacturing environment this might mean buying commercial off-the-shelf hardware or software, or it might mean actually making things. Launching the system means allowing the system do what it was intended to do.

Assess performance. Figures of merit, technical performance measures, and metrics are all used to assess performance. Figures of merit are used to quantify requirements in the tradeoff studies. They usually focus on the product. Technical performance measures are used to mitigate risk during design and manufacturing. Metrics (including customer satisfaction comments, productivity, number of problem reports, or whatever you feel is critical to your business) are used to help manage a company's processes. Measurement is the key. If you cannot measure it, you cannot control it. If you cannot control it, you cannot improve it.

Reevaluation. Reevaluate is arguably the most important of these functions. For a century, engineers have used feedback to help control systems and improve performance. It is one of the most fundamental engineering tools. Reevaluation should be a continual process with many parallel loops. Reevaluate means observing outputs and using this information to modify the system, the inputs, the product, or the process. The re-evaluate function produces retirement, replacement, and recycle at the end of the system life cycle.

This process can be summarized with the acronym SIMILAR [Bahill and Gissing, 1998]. It is important to note that this process is not sequential: The functions are performed in a parallel and iterative manner.

Figure 2, based on Rechtin and Maier [1997] and Bahill and Gissing [1998], shows the intersection of the product and process life cycles. They are similar, although a product life cycle may last months or years, whereas a life cycle for a manufacturing facility may be decades. Figure 2 would have multiple columns for a process that manufactured multiple products. For illustration simplicity, we have omitted the multiple, parallel, iterative feedback loops in this figure.

In this paper, we use the same documentation scheme for the product and the process. However, in industry the documentation is usually different. The product documents are often in the form of a Systems Engineering Management Plan (SEMP) [Grady, 2000; Blanchard and Fabrycky, 1998], whereas the process is often described with, for example, an ISO 9000 description or a Capability Maturity Model Integration® (CMMI^{SM}) model, although Ahern, Clouse, and Turner

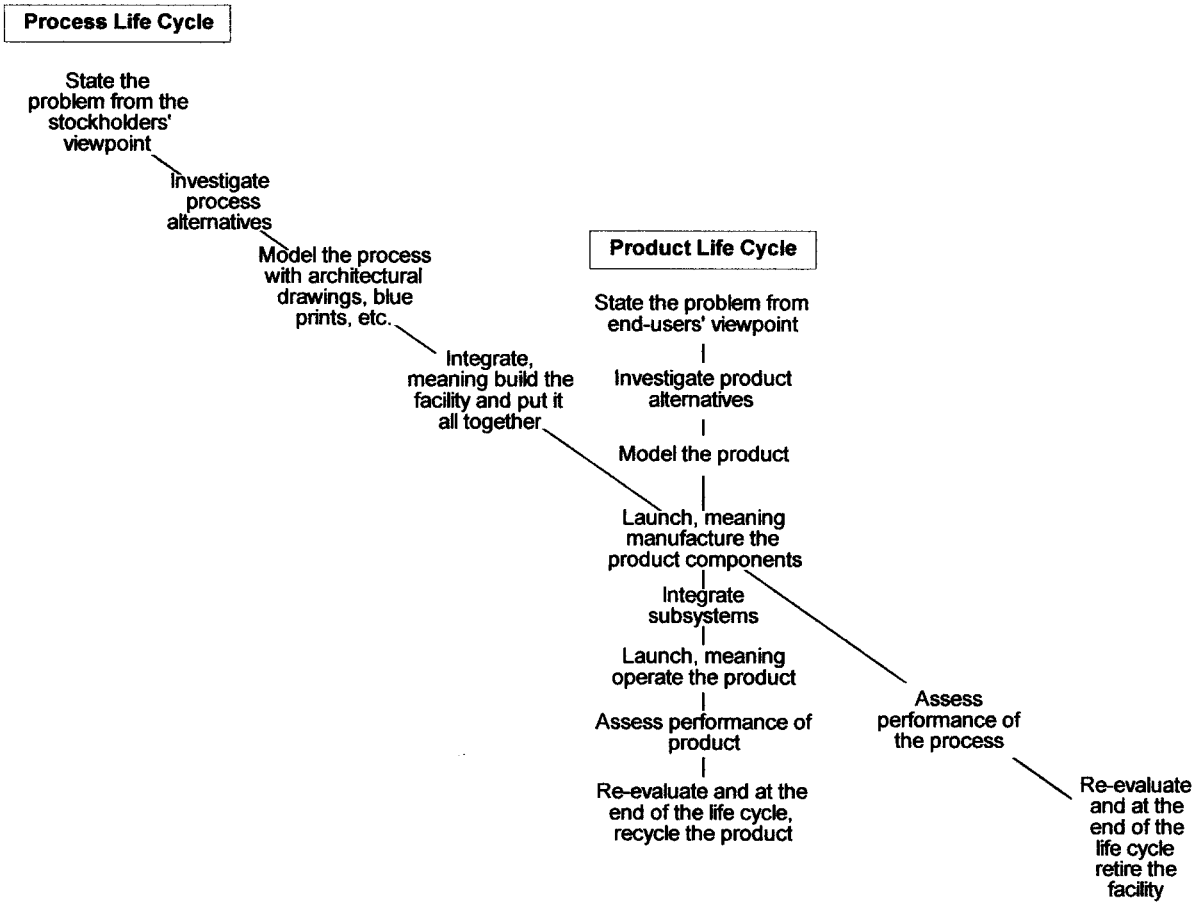

Figure 2. Intersection of the product and process life cycles.

[2001: 123] state, "In OID [Organizational Innovation and Deployment], however, both product- and process-related technology are deployed to improve the total organization." And the CMMI[SM] model [2001] itself states, "In this process area [OID], the term 'process and technology improvements' refers to incremental and innovative improvements to processes and also to process or product technologies [PA161.N107]," and "Solutions, design, and implementations encompass products, product components, and product-related life-cycle processes ... [PA160]." System engineers are not involved, nor should they be, in writing many types of process specifications, .e.g., legal processes, sales processes, contractual processes, supplier processes, etc. There is no reason why these process specifications should have the same or a different format as our systems engineering product and process requirements.

The rest of this paper shows the results of an experiment where we tried to separate the product and process documents.

We wrote the complete set of Systems Engineering Documents [Wymore, 1993; Chapman, Bahill, and Wymore, 1992; Daniels, Werner, and Bahill, 2001] for both the product and the process for one satellite in the University of Arizona Student Satellite Program (SSP). These documents can be seen at http://www.sie.arizona.edu/sysengr/SSP/uasat/uasat.html. The **product** documents describe UASat, a student-designed satellite that is planned to be launched from the Space Shuttle's Hitchhiker Ejection System in 2004 [Abadi, 2001]. The **process** documents describe the Student Satellite Program, which is the organization that is designing UASat and other satellites.

As in most applications, the product and process requirements for UASat are intricately interdependent. The Student Satellite Program has three top-level functions:

1. Educate students,
2. Design, build, and launch satellites, and
3. Publish results in scientific journals.

In addition, each individual satellite has a top-level function, namely to obtain significant scientific data. Figure 3 shows the interconnections of these functions. The output of one function can be the input for another function, or it can become a function that transforms another input into an output. In Figure 3, for example, the semidashed lines indicate that educated students, who are the products of the highest function, operate all other functions.

The SSP has its criteria for success, and each individual satellite project has its individual success criteria that evaluate how well the satellite satisfies its top-level function of obtaining significant scientific data. For example, the UASat must Detect Sprites and Lightning, Record Stellar Data, and Characterize Laser Communication.

Such a cascade of functions of product and process can also exist in industry where (besides manufacturing the product) training employees, obtaining patents and possibly publishing papers are important. We suggest that the product and process can be documented separately, but with the same format.

2. RELATIONSHIPS BETWEEN PRODUCT AND PROCESS

This section will discuss the relationships between the product and process in three areas: requirements, risk, and test.

2.1. Requirements

Requirements are the objectives that the product or process must achieve. There are three main types of requirements in the Systems Engineering documents: Schedule, Cost, and Performance. In some Systems Engineering documentation schemes, these three requirements are rolled into two requirements [e.g., Wymore, 1993]. The Cost and Schedule requirements are grouped into Utilization of Resource Requirements.

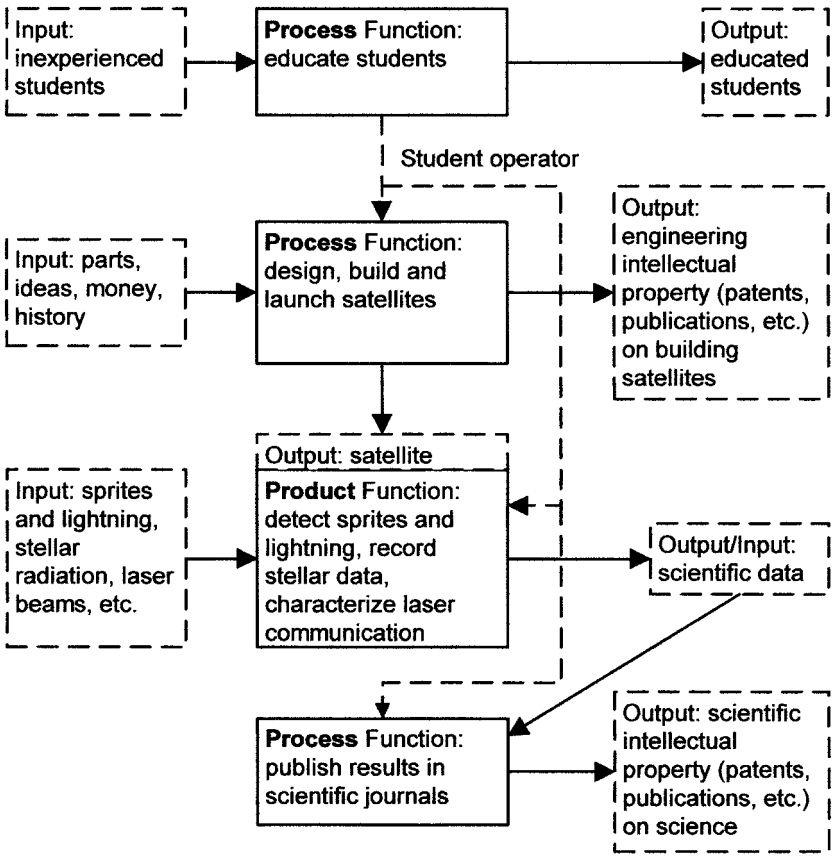

Figure 3. Cascade of Functions for UASat (the Product) and the Student Satellite Program (the Process).

2.1.1. Cost

The cost of one product is determined by the following equation, where n is the number of product items being produced:

$$\text{Product Cost} = \text{Materials} + \text{Manufacturing} + \text{Assembly Test} + \text{Operations} + \text{Facility}/n + \text{Design}/n + (\text{Design Test})/n + \text{Overhead}/n + \text{Retirement}/n. \quad (1)$$

Each element in the equation is explained below:

- Materials refers to the cost of purchasing the raw materials for one product item.
- Manufacturing refers to costs such as labor for manufacturing one product.
- Assembly Test is the cost to test a product unit to make sure all components work and are assembled correctly.
- Operations is the cost of operation and maintenance.
- Facility is the cost of buying and running the factory (or office) and all the tools needed to make the product.
- Design is the cost of paying the engineers who design the product.
- Overhead includes communications, management, human resources, legal, education, ethics, sexual harassment, diversity, sales, marketing, etc.
- Design Test is the cost to test the design before the design is finalized for production.
- Retirement is the cost of retirement, disposal and recycling. Although it could be allocated to both the product and the process, in this equation we only charged it to the process.

The cost of the process is determined by the following equation:

$$\text{Process Cost} = \text{Facility} + \text{Design} + \text{Design Test} + \text{Overhead} + \text{Retirement}. \quad (2)$$

By inserting Eq. (2) into Eq. (1), the Product Cost becomes

$$\text{Product Cost} = \text{Materials} + \text{Manufacturing} + \text{Assembly Test} + \text{Operations} + (\text{Process Cost})/n. \quad (3)$$

Figure 4 visually describes these equations.

These equations are simplistic. They describe the case of a facility dedicated to one product. If the facility is shared by many products, then the situation is more complex. Furthermore, this simplistic model has the Process Cost relatively fixed: It does not change directly with the number n. Whereas the rest of the Product Costs are variable; they are proportional to n. This model is appropriate for processes producing a small number of products, such as our satellites.

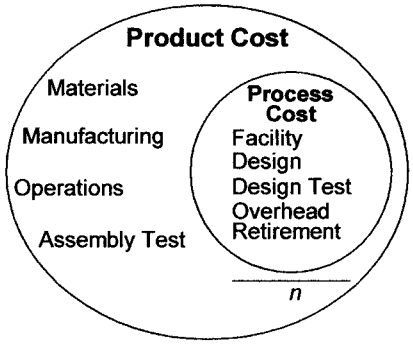

Figure 4. Venn diagram for product and process cost.

Process Cost includes the cost required to design the product before it is manufactured. It is incorporated into product cost by dividing it by the number of production units. If management decides to buy commercial off the shelf (CotS) components instead of designing the components from scratch, then Process Cost is reduced.

In determining the cost requirement for the product and process, the systems engineer first determines the target price for the product by determining how much money the customer is willing to pay for the product. The systems engineer will then determine the Product Cost. The target price must be larger than the Product Cost to allow an acceptable profit. Otherwise, the company does not bid on the proposal.

The following is an example of a cost requirement for the product:

1. The satellite shall cost no more than $100,000, to model, design, build and test.

The following are examples of cost requirements for the process:

1. The cost of paying students shall be $25,000 per year.
2. Faculty/mentor times should be on average 0.5 person-years per year.

2.1.2. Schedule

The top-level schedule requirement is usually the deadline for delivering the final product, as determined by the customer. This is broken down into a set of deadlines, or milestones, or steps toward the completion of

the final product. This is done by working backward from the Delivery Date. Examples of milestones are (in inverse chronological order): Final System Test, Test Readiness Review, Critical Design Review (CDR), Preliminary Design Review (PDR), System Functionality Review, System Requirements Review, and Mission Concept Review.

The following is an example of a schedule requirement for the product:

1. The satellite shall be ready for delivery to NASA no later than March 2004.

The following are example schedule requirements for the process:

1. The time to train new students should be on average 4 weeks.
2. The facilities for building the satellite must be ready by CDR.

More process schedule requirements can be derived from the Work Breakdown Structure (WBS), Gantt charts, PERT charts, Systems Engineering Management Plan (SEMP), Integrated Master Plan (IMP), and the Integrated Master Schedule (IMS).

2.1.3. Performance

For the product, the performance requirements state how well the system must perform certain functions. For example, some of the performance requirements for the UASat are:

1. The satellite, while in orbit but not operating, shall withstand temperatures from –20ºC to +50°C without degradation.
2. The system shall maintain the operating temperature of the electronics between 10°C and 20°C.
3. Components of the satellite with mass more than 9 kg shall withstand test accelerations of ±11 *g*, where *g* is the acceleration of gravity.
4. Components of the satellite with mass 9 kg or less shall withstand test accelerations of ±40 *g*.

For the process, the performance requirements are the criteria that the organization or company that makes the product should achieve. These are some of the performance requirements for the Student Satellite Program.

1. In this decade, the Student Satellite Program shall publish, on average, two student conference papers per year.
2. In this decade, the Student Satellite Program shall produce, on average, one satellite per year.
3. The Student Satellite Program shall produce, on average, 20 alumni per year.

2.1.4. Relationship between the Product and Process Requirements

The main schedule requirement is determined by the date on which the product must be delivered. Almost all other schedule requirements are process-based. For the cost requirements, the process requirements are for the most part a subset of the product requirements. However, there is no fixed relationship between the product and process performance requirements. Stated differently, it is useful to specify whether a requirement is a requirement on the product or the process. For performance requirements this is easy. But for Cost and Schedule requirements it is more difficult, because they are interrelated.

2.2. Risk

The second area to be explored is risk. Risk items may jeopardize the product or process. The risk for product and process is easy to separate. An example of product risk is, “If the wire connecting the solar panel to the power supply breaks, then the batteries cannot be recharged.” An example of process risk is, “If the student with the most knowledge about the satellite suddenly leaves the project, the project will be seriously set back.”

2.3. Test

The third area to be explored is the test. There are two kinds of tests: one for the product and one for the process. The test for the process will measure the design of the product to check whether it meets all the requirements, before it is mass-produced. The result of the test will be used to improve the design. The cost of this test is included in the cost requirement of the process. For the product, we will test the wiring and connections of a product that has been manufactured. It will not result in redesign of the product, but only in correction of errors or withdrawal of products that have been shipped to market. The cost of this test will be included in the product’s cost requirement.

3. EXAMPLES OF MIXING PRODUCT AND PROCESS

The following sections detail two instances in which engineers have mixed the concepts of product and process in the Systems Engineering Documents.

3.1. From the UASat and the Student Satellite Program

The Systems Engineering Documents of the UASat and the Student Satellite Program [2001] can be seen at http://www.sie.arizona.edu/sysengr/SSP/uasat/uasat. html. In the **product** documents, three alternative concept analyses were presented: the ejection system, the science experiment, and the satellite. When this document was first written in May 2000, only the ejection system and the science experiments were explored. However, as time went by, we realized that these two concepts are not related to the product (i.e., the satellite); rather they are related to the process. At that time, we were not aware of the issue of the separation of product and process documentation. After realizing the mistake, we added the third concept exploration, which is for the satellite itself. However, we left in the mistake of including process alternatives in product documents so that people could see the mistake.

The **process** documents deal with the process, i.e., the organization model of the Student Satellite Program. During the time this document was written, we already had a clear concept of the separation of product and process and thus this document truly describes the process of making the satellite.

3.2. From SIE 554 Students

A second example illustrating the difficulty in separating the product and the process comes from studying documents that were written by students who took SIE 554 (The Systems Engineering Process) taught by Professor Bahill at the University of Arizona.

In the fall of 2000, the semester project was to create the documents that describe a system that will choose the best bat for a softball or baseball player (or some other sports implement for hitting moving balls). The students were required to write two sets of documents, one for the product and the other for the process that would make the product. An unexpected finding was that even though the students were given clear instructions to separate product and process requirements into two sets of documents, they still had product requirements in the process documents and process requirements in the product documents.

The following three requirements were in the cost section of the product documents of the best project that was submitted.

1. Acquisition Time (in months) baseline = 3 months. The number of months required to complete the design in order to get the product to market.
2. Acquisition Cost (in dollars). The project should be completed within the budget established for this effort: $425,000.
3. Manufacturing Cost (in dollars/racket) Baseline = $24. The tennis racket design must consider the expense of manufacturing.

The second of these requirements is clearly a process requirement and should not be in the product documents.

Table I illustrates the types and number of mistakes made in mixing product and process for the experiment conducted in SIE 554. The rows contain data for the

Table I. Mistakes Detected in Mixing Product and Process in Systems Engineering Documents

Team		System Inputs	Functions	System Outputs	Performance Requirements	Cost Reqs	Schedule Reqs	Alternative Concepts	*Errors*
	Prod	0/8	0/1	0/1	0/5	2/2	1/3	0/6	*2*
1	Proc	0/6	0/1	0/1	1/3	0/2	1/3	0/4	*2*
	Prod	0/3	0/1	0/1	0/4	0/1	0/1	0/5	*0*
2	Proc	2/4	0/1	2/2	0/0	0/0	0/0	0/0	*2*
	Prod	0/6	0/1	0/2	0/3	2/3	2/3	0/6	*2*
3	Proc	0/2	0/1	0/1	0/2	1/1	1/1	5/5	*2*
	Prod	0/10	0/1	0/2	0/3	0/1	0/2	0/5	*0*
4	Proc	9/9	0/1	2/2	2/2	0/1	0/2	0/4	*3*
	Prod	3/3	1/1	2/2	0/3	0/2	0/1	0/5	*3*
5	Proc	0/3	0/1	0/1	0/3	0/1	0/1	0/4	*0*
	Prod	0/11	0/1	0/1	0/4	1/2	0/2	0/4	*1*
6	Proc	0/1	0/1	0/1	0/2	0/2	1/1	0/2	*1*
7	Prod	0/2	0/1	0/1	0/5	0/5	0/1	0/4	*0*
8	Prod	0/4	0/1	0/1	0/6	0/1	0/1	0/3	*0*
9	Prod	0/4	0/1	0/4	0/4	0/1	0/1	0/3	*0*
10	Prod	0/4	0/1	0/1	0/4	0/1	0/0	0/3	*0*
Errors		*3*	*1*	*3*	*2*	*4*	*5*	*1*	

Table II. Mistakes Detected in Mixing Product and Process in Systems Engineering Documents in the Second Year

Team		System Inputs	Functions	System Outputs	Performance Requirements	Cost Reqs	Schedule Reqs	Alternative Concepts	*Errors*
1	Prod	0/5	0/38	0/5	0/17	0/1	0/1	0/6	*0*
	Proc	0/3	0/3	0/3	0/7	0/2	0/3	0/6	*0*
2	Prod	0/4	0/1	0/4	0/18	0/1	0/1	0/5	*0*
	Proc	0/3	0/3	0/3	0/7	0/2	0/2	0/5	*0*
3	Prod	0/4	0/1	0/4	0/21	0/0	0/0	0/5	*0*
	Proc	0/3	0/1	0/3	0/7	0/2	0/2	0/5	*0*
4	Prod	0/2	0/3	0/2	0/13	0/1	0/1	0/2	*0*
	Proc	0/4	0/4	0/3	0/10	0/2	0/3	0/2	*0*
5	Prod	0/5	0/1	0/2	0/4	0/1	0/1	0/3	*0*
	Proc	0/7	0/1	0/6	0/4	0/1	0/1	0/3	*0*
Errors		*0*	*0*	*0*	*0*	*0*	*0*	*0*	

product and process documents of each of the teams. The columns contain the categories where mistakes were feasible: System Inputs, System Functions, System Outputs, Performance Requirements, Cost Requirements, Schedule Requirements, and Concept Exploration Alternatives. Each cell shows the number of mistakes and the total number of items listed in each team's documents. If the students put a process requirement in the product document, it is counted as a mistake, and vice versa.

The column on the right contains the count of how many areas (e. g., Input, Function, etc.) contain a mistake for each row. Once an area has a mistake, it counts as one error, and the number of mistakes found in an area is irrelevant. The bottom row contains the count of how many documents contained mistakes. The procedure for counting is the same as the right column; once a document has a mistake, it is counted as one error. A "0/0" in this table means that there were no data in the documents, because these teams failed to write these sections.

Summary of the Statistics. The data of Table I can be summarized as follows:

- The most mistakes were made in the Schedule Requirements (five mistakes), followed by Cost Requirements (four mistakes), Input and Output (three mistakes), Performance Requirements (two mistakes), Alternative Concepts (one mistake), and Functions (one mistake).
- The teams who made input mistakes most likely made output mistakes as well.
- Although all teams were asked to write two sets of documents, teams 7, 8, 9, and 10 only wrote the product documents.

These were not merely careless mistakes, because the students reported that, on average, each team spent 100 person-hours writing their documents.

One year later, another study was conducted. This time, the students were given the task of writing product and process systems engineering documents for CubeSat-3, another satellite project from the Student Satellite Program. To assist them in separating the documents, the students were provided a preliminary version of this INCOSE paper, a lecture on it and the documents for UASat [2001], which clearly separate product and process (except for the mistake of including process alternatives in the product documents, a mistake that was deliberately left in these documents for heuristic reasons).

Table II shows the results. There were no mistakes in separating the product and process. Two errors of omission were made by a team that failed to write product cost and schedule requirements.

These data seem to show that it is hard to avoid including process requirements in product documents, and vice versa. But the task becomes easier if the two sets of documents are written formally.

4. DETAILS ABOUT THE SATELLITES

This project has generated many exciting ideas that are out of the scope of this paper. But here we will briefly mention a few.

4.1. Reusability

The need to reuse systems has caused a design change in the satellites. Our newer satellites will be CubeSats: 10 cm cubes. Each will start with a standard bus containing the frame, power supply, communications, and

computers. Then a unique science experiment will be designed into each satellite.

4.2. Ground Station

Each satellite will communicate with our scientists through a ground station at the University of Arizona. There was some controversy as to whether the ground station should be a part of the product or the process. In the UASat [2001] documentation, we made it a part of the product. This decision was arbitrary. All systems engineers know that specifying the boundary of the system is one of our most difficult tasks.

4.3. Science Experiments

A more difficult issue was deciding whether the science experiments were a part of the product or the process. In the present SSP, each science experiment is being designed and implemented by an independent team. The science experiments are interchangeable. Therefore, the satellite design may be reused for several launches, each with a different science experiment. This suggests that the product system boundary could exclude the science experiments. However, the science experiments produce scientific data, which is a defined output of the process. In an earlier part of this paper, we said that including an investigation of alternative science experiments in the product documents was a mistake. However, the mistake was subtler than we implied. The mistake was including an investigation of alternative science experiments in the *top level* of the product design. The top-level investigation of alternatives should include things like balloons, distant mountains (for laser communication), rockets, CubeSats, and particle accelerators (for semiconductor irradiation).

4.4. Challenges

The biggest challenge in the SSP is getting the satellites launched. The original plan was to use the Space Shuttle. The newer plan is to have 30 cube sats launched simultaneously on a Russian SS-19 rocket. Our budget is about $100,000 per year, and we plan to launch a satellite per year culminating in a flight to Mars in 2010. Readiness for launch is determined by the Project Manager and the launch organization, i.e., NASA or the company organizing the cube sats. Consequences of failure are difficult to define. There can be no total failure, because the students have and will gain valuable project experience. And the students are the most important component of the SSP. Each of the satellites is designed so that useful scientific information can be collected even if most of the satellite fails. For example, the corner reflectors will still function even if all of the electronics fails. Additional details about the SSP can be found at http://www.sie.arizona.edu/sysengr/SSP/cubesat3/readme.txt.

5. CONCLUSION

Separating product and process is a difficult task, as shown in the results of the experiments summarized above. In these experiments, the most difficult tasks were separating the Cost and Schedule requirements.

Despite the difficulty inherent in keeping these requirements in separate documents, we believe that it is important for systems engineers to understand the differences between product and process. Writing separate documents for product and process should (1) make it easier to get the requirements right, (2) help manage the requirements when either the product or the process requirements change, (3) make omissions less likely, (4) make it easier to specify the system boundary, (5) make it easier to manage programs that have one process producing multiple products, and (6) help identify which costs are dependent on the number of product items being produced.

Requirements are easier to verify and validate if each requirement states who must do what. Separating requirements into the categories of product and process is a step in this direction.

Most companies have a method for doing the systems engineering for a product. It might be located in their Integrated Product Development Process or some similarly named computer system. Most companies also have a process for how they do business. What we have shown in this paper is that you can use the same systems engineering documentation format for both the product and the process.

In industry, when there are two sets of documents, the product documents are written by Systems Engineering and the process documents are often written by Program Management. But there is a trend to change this. The progression from a SEMP to an Integrated Master Plan and an Integrated Master Schedule (IMP/IMS) shows the progression from Systems Engineering writing only the product documents to Systems Engineering writing both sets of documents.

Finally, the process is performed by a system. We should apply systems engineering tools to the design of this system.

REFERENCES

C.D. Abadi, Product and process: An innovative approach for a Student Satellite Program from the Systems Engineering point of view, 15th Annual AIAA/USU Small Satellite

Conference, Logan, UT, August 2001, http://tucson.sie.arizona.edu/sysengr/SSP/uasat/document/AIAAPaper.pdf (October 2001).

D.M. Ahern, A. Clouse, and R. Turner, CMMI[SM] distilled: A practical introduction to integrated process improvement, Addison-Wesley, Reading, MA, 2001.

A.T. Bahill and B. Gissing, Re-evaluating systems engineering concepts using systems thinking, IEEE Trans Syst Man Cybernet C Appl Rev 28(4) (1998), 516–527.

B.S. Blanchard and W.J. Fabrycky, Systems engineering and analysis, Prentice Hall, Englewood Cliffs, NJ, 1998.

B. Boehm, D. Port, and V. Basili, Realizing the benefits of the CMMI[SM] with the CeBASE method, Syst Eng 5(1) (2002), 73–88.

D.M. Buede, The engineering design of systems, Wiley, New York, 2000.

W.L. Chapman, A.T. Bahill, and A.W. Wymore, Engineering modeling and design, CRC Press, Boca Raton, FL, 1992.

Capability Maturity Model® Integration (CMMI[SM]), Version 1.1, (CMMI-SE/SW/IPPD, V1.1 Staged Representation, Software Engineering Institute, Pittsburgh, 2001, http://www.sei.cmu.edu/cmmi/.

J. Daniels, P.W. Werner, and A.T. Bahill, Quantitative methods for tradeoff analyses, Syst Eng 4(3) (2001), 199–212.

J.O. Grady, System engineering deployment, CRC Press, Boca Raton, FL, 2000.

E. Rechtin and M.W. Maier, The art of systems architecting, CRC Press, Boca Raton, FL, 1997.

Student Satellite Program 10 year Plan, October 2001, http://tucson.sie.arizona.edu/sysengr/SSP/uasat/document/SSP10year.htm.

UASat and Student Satellite Program Systems Engineering Documents, October 2001, http://tucson.sie.arizona.edu/sysengr/SSP/uasat/uasat.html.

A.W. Wymore, Model-based Systems Engineering, CRC Press, Boca Raton, FL, 1993.

Christiano Danny Abadi received his M.S. in Systems Engineering from the University of Arizona in 2001. He has a B.S. in the Electrical and Computer Engineering. He was the Systems Engineer for the Student Satellite Program that designed UASat, a student-designed satellite that is planned to be launched from the Space Shuttle's Hitchhiker Ejection System in 2004. He described this work in a paper he presented at the 15th Annual AIAA/USU Small Satellite Conference, Logan, UT, August 2001.

Terry Bahill has been a Professor of Systems Engineering at the University of Arizona in Tucson since 1984. He received his Ph.D. in electrical engineering and computer science from the University of California, Berkeley, in 1975. He holds U.S. Patent Number 5,118,102 for the Bat Chooser, a system that computes the Ideal Bat Weight for individual baseball and softball batters. He is a Fellow of the Institute of Electrical and Electronics Engineers (IEEE), of Raytheon and of INCOSE. He is chair of the INCOSE Fellows Selection Committee. This photograph of Bahill is in the Baseball Hall of Fame's exhibition *Baseball As America*.