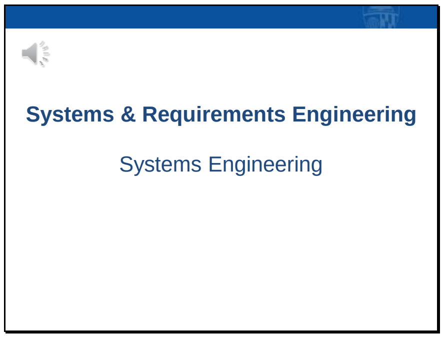
1

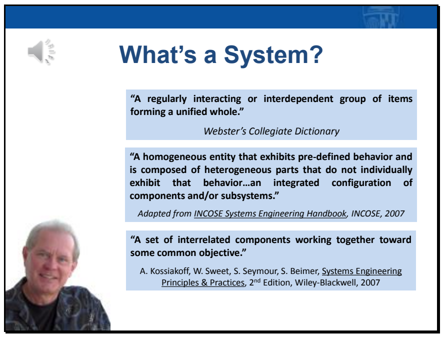


Systems & Requirements Engineering

Systems Engineering

In this lecture we'll discuss some basic concepts in systems engineering .

2



What's a System?

"A regularly interacting or interdependent group of items forming a unified whole."

Webster's Collegiate Dictionary

"A homogeneous entity that exhibits pre-defined behavior and is composed of heterogeneous parts that do not individually exhibit that behavior…an integrated configuration of components and/or subsystems."

Adapted from INCOSE Systems Engineering Handbook, INCOSE, 2007

"A set of interrelated components working together toward some common objective."

A. Kossiakoff, W. Sweet, S. Seymour, S. Beimer, Systems Engineering Principles & Practices, 2nd Edition, Wiley-Blackwell, 2007

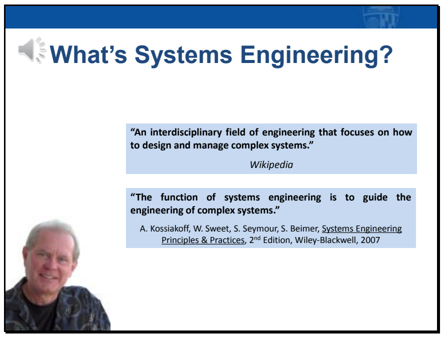
While the focus of this course is on software engineering, I want to take a little time to talk about putting software engineering into the larger context of systems engineering. So...let me start by discussing the term system.

What's a system anyway? Well...there are many definitions of the term system, even within the systems engineering profession.

Let me start with a dictionary definition. Webster's Collegiate Dictionary defines a system as a regularly interacting or interdependent group of items that form a unified whole. The International Council on Systems Engineering defines a system as a homogeneous entity that exhibits pre-defined behavior and which is composed of heterogeneous parts that do not individually exhibit that behavior...and which is an integrated configuration of components and/or subsystems. That's kind of a mouthful. And there are quite a few more definitions. The IEEE has one, the International Standards Organization...ISO...has one, and many other organizations. There are things I like in both of these definitions...the concept of a unified whole, the concept of components and subsystems. The definition I like the best, since it's very easy to remember, is this one. A set of interrelated components working together toward some common objectives.

There are lots of different kinds of systems that exist in the world. There's the solar system, the number system, ecological systems, weapons systems, and everyday products like automobiles that are, in fact, systems.

3



What's Systems Engineering?

"An interdisciplinary field of engineering that focuses on how to design and manage complex systems."

Wikipedia

"The function of systems engineering is to guide the engineering of complex systems."

A. Kossiakoff, W. Sweet, S. Seymour, S. Beimer, Systems Engineering Principles & Practices, 2nd Edition, Wiley-Blackwell, 2007

Now that we've established a definition or two about what a system is, what then is software engineering?

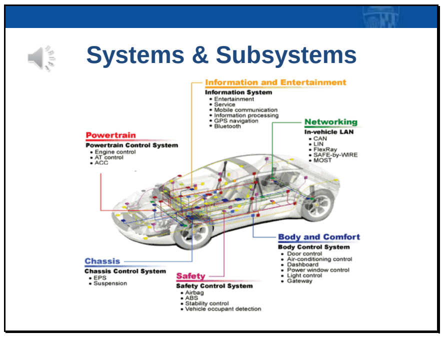
According to Wikipedia, it's an interdisciplinary field of engineering that focuses on how to design and manage complex systems. And, according to the late Alexander Kossiakoff, a former director of the Johns Hopkins Applied Physics Laboratory, its purpose is to guide the engineering of complex systems.

Systems engineering typically requires the collaboration of many different types of disciplines, such as electrical engineering, mechanical engineering, perhaps physics or other scientific disciplines, and also software engineering.

Note that the word complex is used in both of these definitions. According to Kossiakoff, complexity restricts systems engineering to systems in which the elements are diverse and have intricate relationships with one another. What he meant by that is systems engineering, as a discipline, is not likely to be applied to smaller, simpler systems.

4

Systems & Subsystems



Another characteristic of systems is that they contain components that work together to contribute to the system's overall functionality. These components are often called subsystems.

As a real and concrete system example, let's take an automobile. The automobile itself is a system, with well-defined behavior. It consists of numerous subsystems, some of which are identified in this schematic. There's a powertrain system, a safety control system, and so forth. And quite a few more, like a fuel control system, a brake control system...I could go on and on, but I'm hoping you see my point. Each of the automobile's subsystems are designed to provide well-defined behavior that contributes to the overall functioning and behavior of the entire automobile. Some of these subsystems may be relatively independent, and some might be quite interdependent on other subsystems.

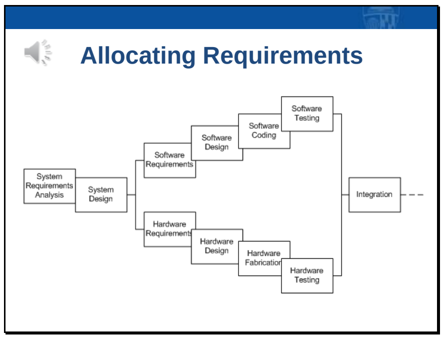
At this point, you might be wondering how all this relates to software engineering. It relates directly to software engineering. If you drive a reasonably modern car, you are most likely aware that many features of your car are controlled, at least in part, by software.

Your car's brake control system, for example, may include sensors that monitor wheel speed driving angle and send signals to an onboard computer to automatically adjust braking and engine power to ensure that you always remain in control of the vehicle.

When automobiles are designed, some of the decisions that are made in the systems engineering process is which subsystems will be responsible for various functions, which functions will be satisfied by hardware components, and which will be satisfied by software.

So...before any software engineering takes place...system responsibilities...or requirements...must be allocated to the software components that make up the system and subsystem elements.

5



Allocating Requirements

Here's a piece of a systems engineering life cycle. The first two phases in this life cycle consist of system requirements analysis and system design. System requirements analysis determines the requirements for the overall system. Some of these requirements will be fulfilled by hardware, some by software and some by both...as an example, software may be embedded in firmware for certain functions.

A step within these phases, sometimes called functional analysis or functional allocation, will allocate the system functionality to the system's hardware components and its software components. Then, the software engineering process begins...as indicated by the upper part of the diagram...as does the hardware engineering process...indicated in the lower portion of the diagram. Then, the hardware components are integrated, tested, and so forth.

So...what I wanted to accomplish here is how software engineering fits into the overall systems engineering process. Now, there can be quite a few variations on the diagram I used here. For example, this diagram is showing a step in which hardware is actually fabricated. For some types of systems, component parts might just be purchased and interfaced to the software.

In the next lecture, I'll discuss some of the criteria that might be used to allocate system requirements to software.