Designing Information Systems with System Dynamics: A C2 example

Corey Lofdahl
Science Applications International Corporation (SAIC)
20 Mall Road, Suite 130
Burlington, Massachusetts USA
781.221.7610
clofdahl@bos.saic.com

Abstract

It has long been thought that simulation could be used to design Command and Control (C2) system architectures, but simulation's benefits so far have not matched their promise. Instead Enterprise Architecture Planning (EAP) tools have become ascendant in the design of C2 systems, though problems remain. EAP tools break down proposed systems into their low-level, constituent parts and place the details into relational databases. The resulting architectures however yield little intuitive sense of whether the proposed system actually solves the motivating problem. Consequently, fundamental issues continue to emerge deep into the design process.

This study proposes using simulation early in the design process to envision the total system and avoid problems by generating requirements and metrics early in the design process. Issues regarding an Air Force Air Operations Center (AOC) are explored, most notably flow of control and the coordination of sensor, decision, and operator assets.

0. Introduction

The funding, design, and management of modern command and control (C2) systems presents technical, organizational, and operational challenges to those who are tasked with their acquisition. The designation of a Lead System Integrator, a contractor who coordinates the system's subcontractors and technical development, has recently been tried but has not proven totally successful. Correct system design cannot be mandated or imposed as the design of complex C2 systems is a process, one in which organizational relationships and various technologies can support but do not define. For example, the government must still decide whether to make or buy key components, and compromises must be reached among the various proprietary solutions offered by contributing contractors. Even the most basic questions, such as when the various design phases are complete, is open to interpretation.

Beyond the technical there are *organizational* complexities including involvement by the Office of the Secretary of Defense (OSD) in joint programs that span the services, changing service priorities, and the jostling of funding based on these factors. Most importantly, C2 systems inherently cross the boundary between the technical and the behavioral (Glasow 2004) – such systems are designed to coordinate distributed, social organizations tasked with an operational mission – which adds additional dimensionality to the technical design space. Common operational understandings and leadership relationships impact system design as do ongoing joint concerns. The earlier such issues are thought out and addressed in the design cycle, especially by the government before contract award, the better for the program, system, and eventual users.

Operational innovations such as increased force mobility, decreased force footprint, and Network Centric Warfare or NCW (Alberts et al. 1999) place additional pressure on C2 systems. Designers help give operators the tactical edge with layered architectures that separate applications from infrastructures, modern IT technology that puts cursors on targets, and high bandwidth connections that connect operator platforms. This constellation of technical, policy, and social challenges combines with the larger paradigm shifts from an operator's "need to know" to "need to share" along with more traditional questions of system costs, performance, and force protection. The result of all these factors is a complex engineering trade space thick with hard-to-understand tradeoffs. Designers, users, and funding agencies all need to know how to represent considered systems, where the decision points are, and how to integrate the relevant data.

This paper proposes using System Dynamics (SD) simulation early in the design process to envision the total C2 system, generate system requirements, and answer design questions. This argument is developed first by presenting a general overview of SD's application to the design of C2 systems. Second, an example simulation is created to examine a proposed F2T2EA[1] Air Operations Center (AOC). Third, simulation is contrasted and compared with more traditional Enterprise Architecture Planning, with the two techniques found to be complementary. In conclusion, programmatic and technical steps are discussed.

[1] find, fix, track, target, engage, assess

1. System Dynamics Model

When C2 systems are first envisioned, they are usually introduced and discussed in a highly graphical, Power-point supported, and quantitatively challenged manner. This study instead proposes using simulation early in the design cycle to enable a more quantitatively grounded description of the system.

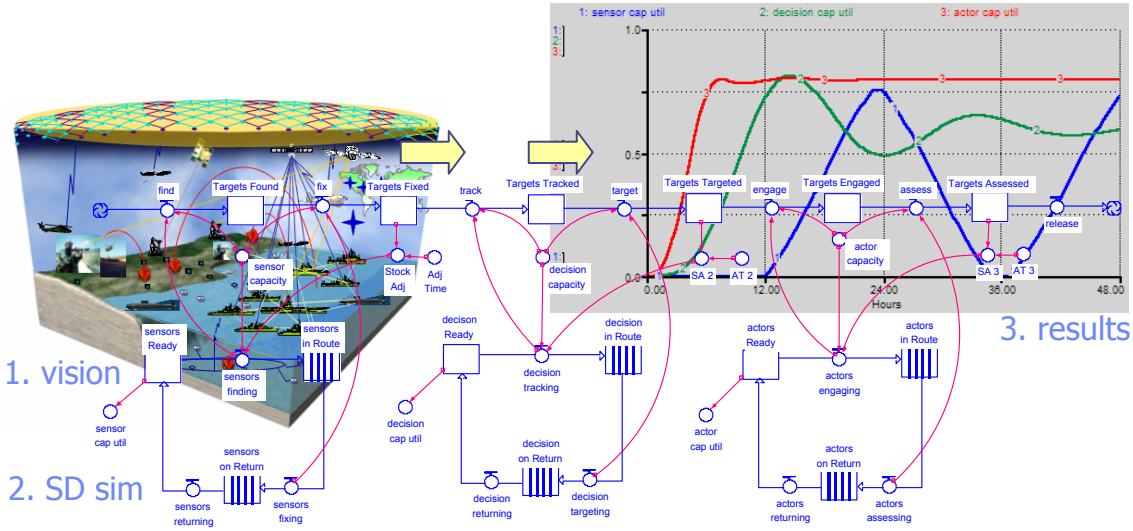

Figure 1 – Early C2 System Simulation Process

To the left of figure 1 is a typical, high-level system depiction graphic. In the center is an SD simulation of the C2 system, an F2T2EA architecture comprised of sensor, decision, and operator assets. To the right is an output from the simulation that shows how it reacts given certain initial conditions and operational assumptions. The goal here is not to simulate perfectly the proposed system but to determine quickly its likely timing and processing requirements. This can be done in a matter of weeks rather than years, which can benefit greatly those acquiring, building, funding, and using the system.

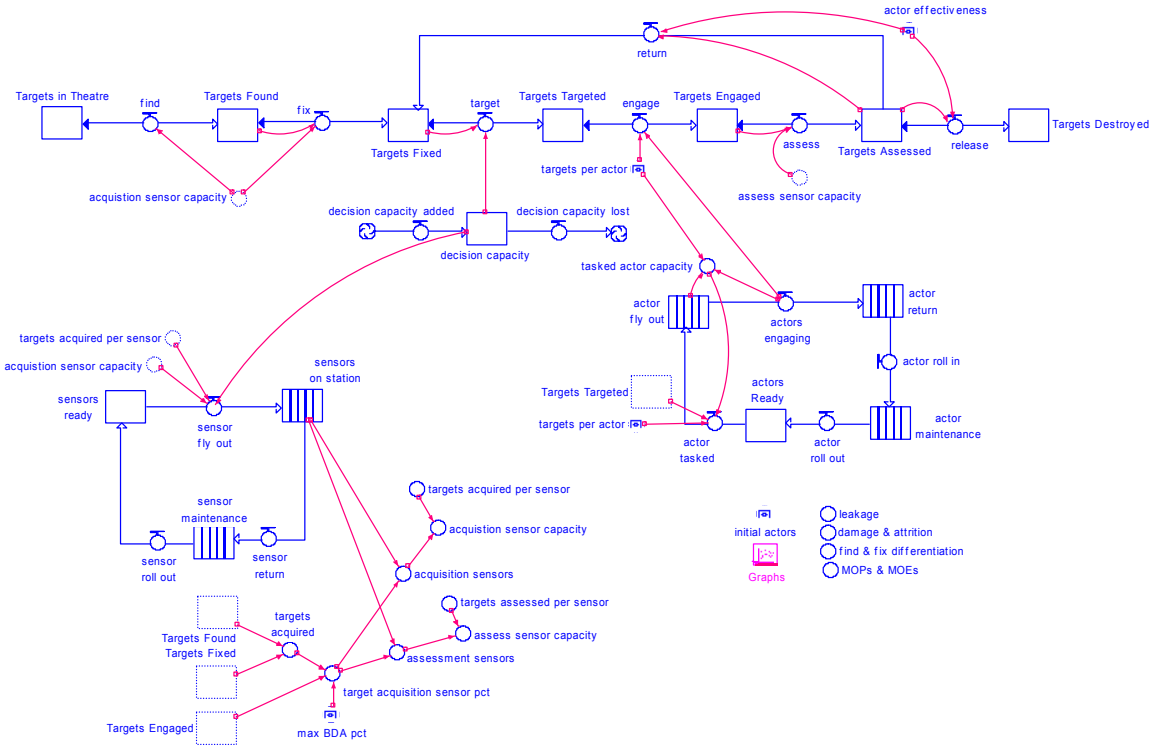

Figure 2 -- Detailed F2T2EA Architecture

Figure 2 presents an architecture with more detail incorporated into the sensor and operator portions of the system. Creating such a simulation provides several benefits. First, since the simulation in inherently quantitative, metrics in the form of Measures of Performance (MOPs) and Measures of Effectiveness (MOEs) fall out naturally from the simulation effort. Second, the simulation also delivers a more graphical and analytically rich description of the system. Third, it provides a method of system analysis separate from and complementary to system architectures (cf. Section 3). Fourth, the resulting simulation provides support for analytically grounded gap analyses.

These capabilities, in turn, support several aspects of system design. First, such simulations allow for the quick analysis of Systems of Systems (SoS). In Figure 2, sensor systems, decision systems, and operator systems – each analytically rich by themselves – are described and combined in such a way as to allow the whole system to

be analyzed. Thus stovepipes can be encompassed and the contributions of individual systems with their proposed changes and contributions can be evaluated in a methodologically defensible fashion. Second, simulation supports program management and choreography. Should an acquisition agency be faced with multiple proposals and investment opportunities, simulation provides a way to think them through without the time and effort of actually funding them. Third, simulation can be used to capture investments as systems that are described more clearly and quantitatively are likely to be viewed more favorably by funding agencies. The details and sample outputs of such an effort are discussed in the next section.

2. System Dynamics Simulation

In this section, the Figure 2 model is run to demonstrate the type of information that can reasonably be obtained from a simulation. This section shows that such simulations can be created quickly and yield volume and timing information in support of an initial, high-level C2 system vision. In defining the simulation, technical gaps and key data items can be identified early in the design process enabling more focused experimentation and directed data acquisition. Moreover, simulation helps close the gap between social and technical systems (Glasow 2004), thus supporting dynamic examination of interactions between enterprise and system. And since simulations are developed using commercial tools, time spent coding rather than analyzing the problem is minimized.

Figure 3 shows the base run for the F2T2EA model developed in the preceding section. Three different panels are shown. The first represents the system's *stocks*, the containers of various key measures within the model – in this case, records denoting

items of system interest or *targets*.[2] The diagram shows the third value, *targets targeted*, increasing in relation to the other values denoting a bottleneck in the system. The second panel denotes *flows* within the system, that is, the rate at which records move from one stock to the next. There is no real pattern in the third panel except that the fourth measure, *engage*, jerks up and down. The third panel shows some aggregate measures of system performance – *targets in theatre*, *targets destroyed*, targets *released* from the system, and targets *returned* to the system. In this way, measures of system performance and effectiveness can be developed and the overall performance of the system can be determined.

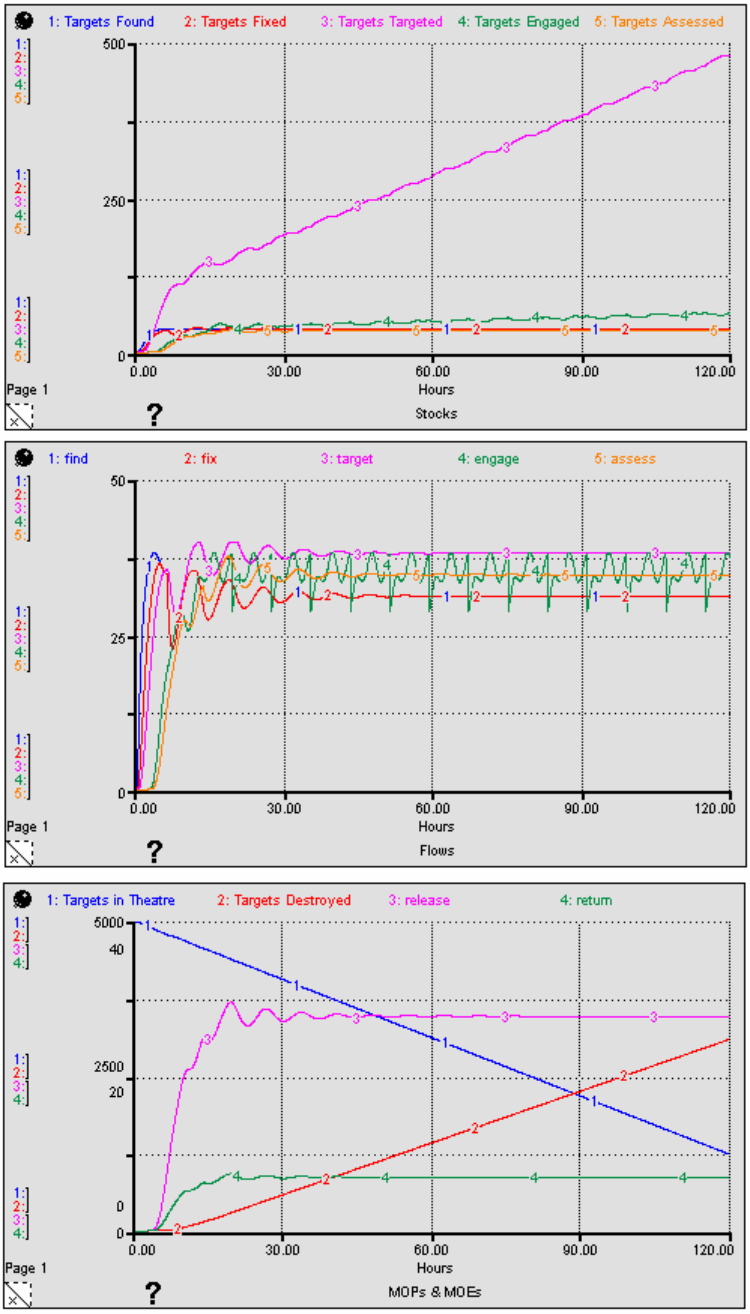

Figure 3 -- Base Run Output

[2] The stocks depicted are Targets Found, Targets Fixed, Targets Targeted, Targets Engaged, and Targets Assessed.

Figure 4 shows a set of values obtained trying to correct the bottleneck observed in Figure 3, which showed one system value growing without bound. The number of operators was increased and the simulation was run again. The top panel shows that the bottleneck has simply moved down the chain so that the fourth line, *engaged* now grows above the others. The second panel denoting system flows has also changed with the fourth measure, *engage*, having straightened out.

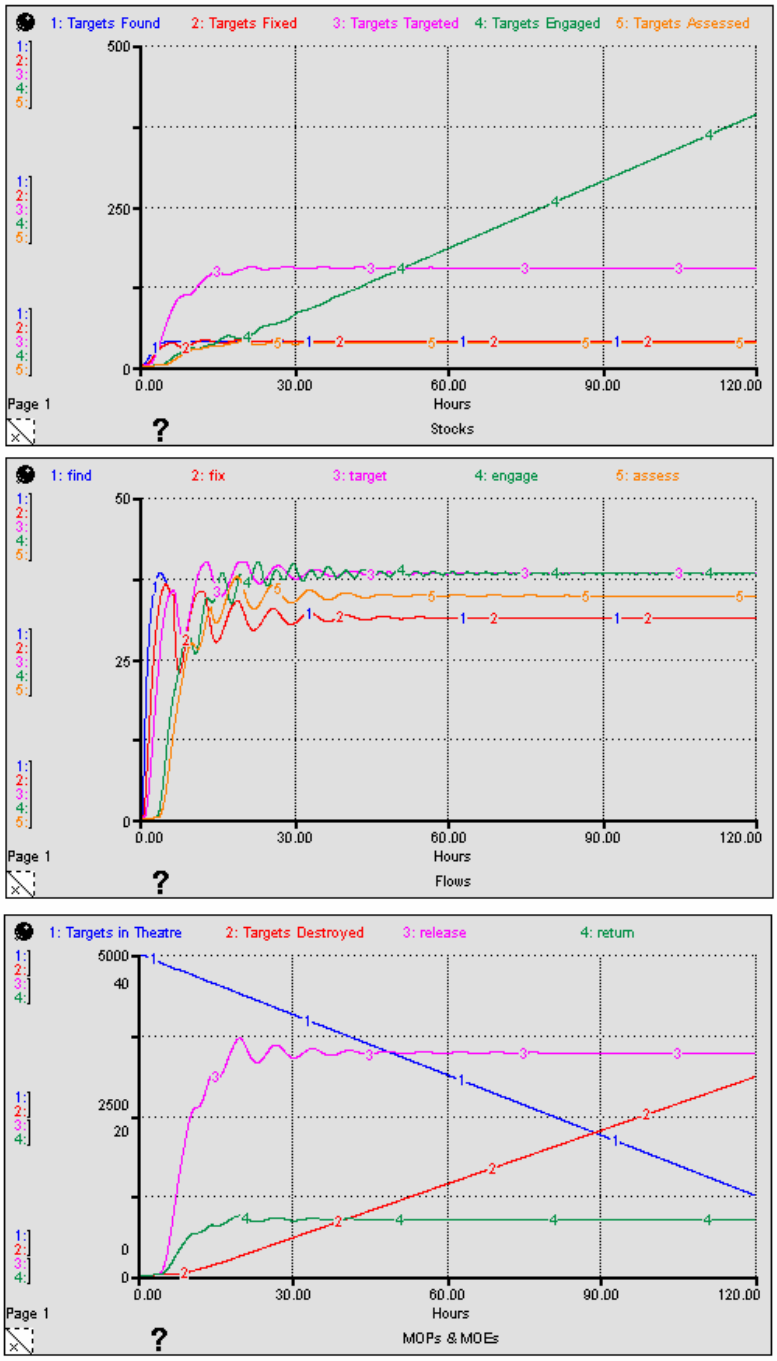

Figure 4 -- Increased Operator Run

To fix the system problem of Figure 4, sensor assets are reallocated so that there is more assessment of targets already in the system rather that searching for targets to enter into the system. The Figure 5 results now show that no stock grows without bound, denoting the problem has been fixed. In the second panel, there is a clear separation between the flows at the right and left of Figure 2 due to records being returned back into the system for further processing. In this manner, the magnitude or volume of the system's flows can be determined.

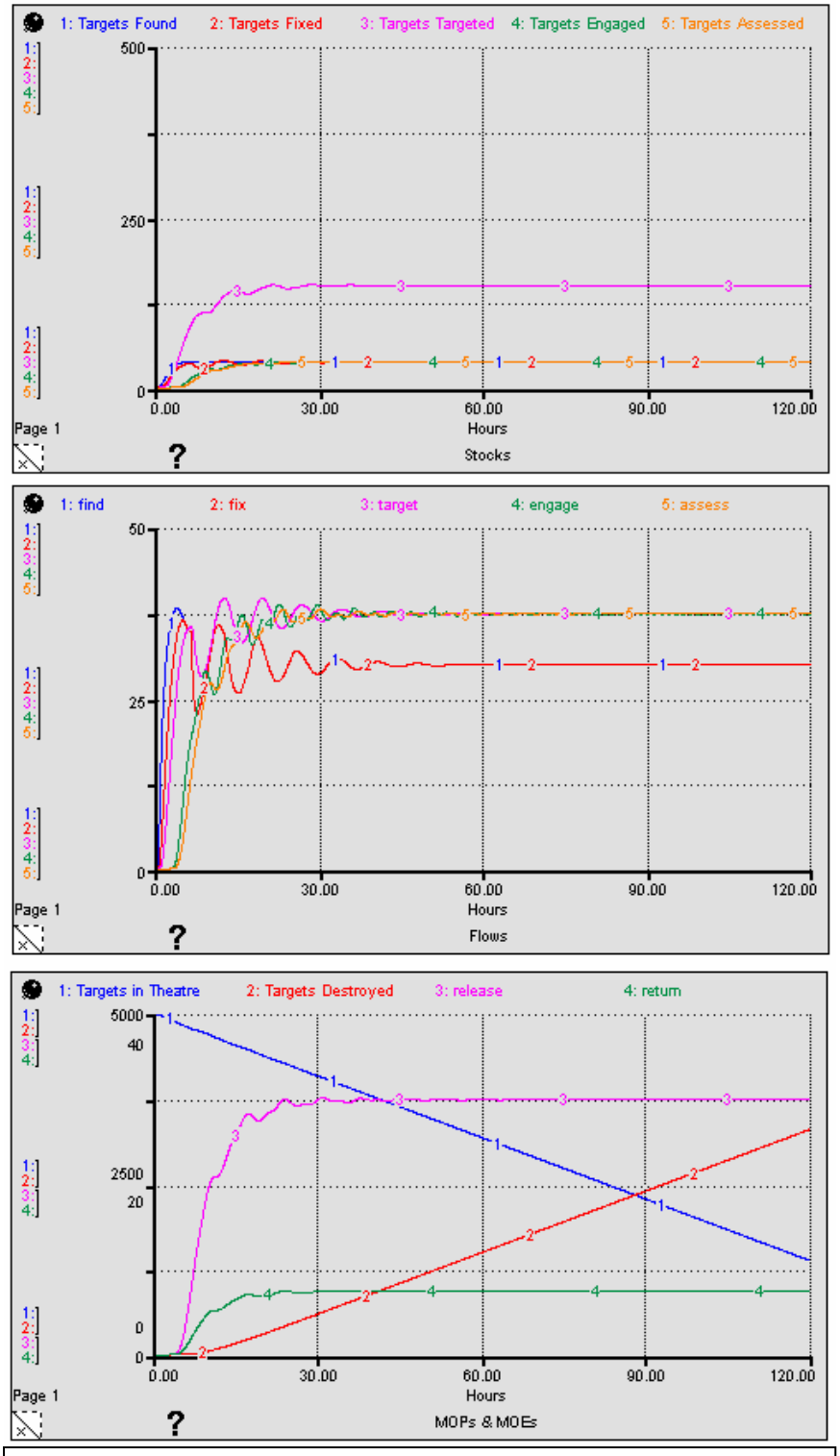

Figure 5 -- Sensor Reallocation Run

3. Enterprise Architecture Modeling

To understand how simulation can contribute to the design of information systems generally and C2 systems specifically, it must first be considered what tools are currently being used to design such systems:

> Until recently, it has been almost a fundamental article of faith that as we got more advanced technologically and organizationally, we would be able to tame complexity by insightful decomposition and massive amounts of processing power. (Alberts, Garstka, and Stein 1999, 151)

Alberts et al. observe that the current state of the system design art entails system decomposition driven by ever more powerful computers and that this combination is not up to the task of taming system complexity. *Enterprise Architecture Planning* (EAP) tools best characterize this system design state-of-the-art (e.g., Spewak 1995, Fowler 2003, Popkin 2004).

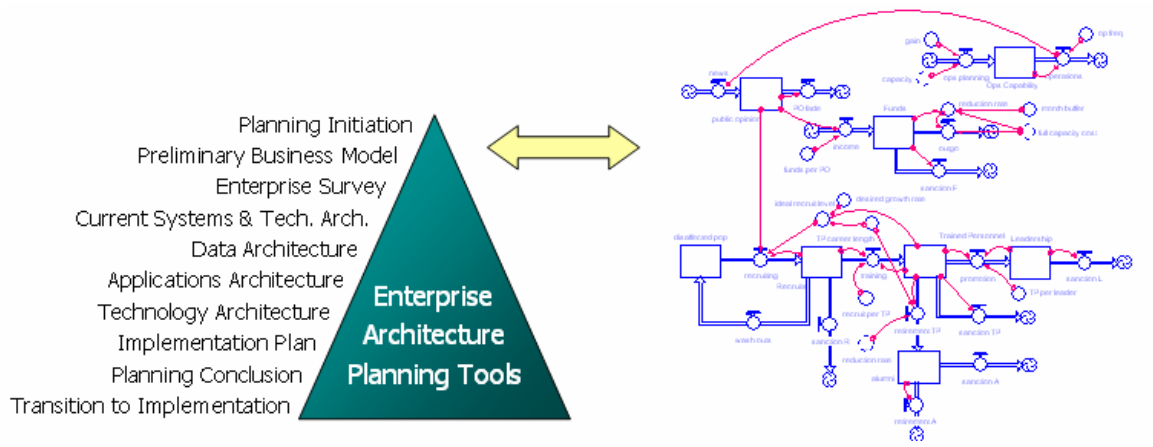

Figure 6 -- Enterprise Architecture Planning and System Dynamics Models

EAP is built on top of relational databases that capture the myriad architectural details of a proposed information system. What is gained in the understanding of system detail however comes at the expense of seeing how the whole system is likely to work, how it will interact with other systems, and whether or not the resulting architecture solves the motivating problem. System dynamics simulation, in contrast, is more abstract. It does

not seek to represent every system detail but instead strives to capture key features that span and impact the system through dynamic experimentation. In short, if EAP seeks to understand the trees, then system dynamics seeks to understand the forest. More technically, Sterman (2000) contrasts two types of system complexity, *detail* and *dynamic*. With regard to the methodologies discussed here, EAP is better at detail complexity, while SD's strength is dynamic complexity. Combined correctly, EAP and SD have the potential to complement each other.

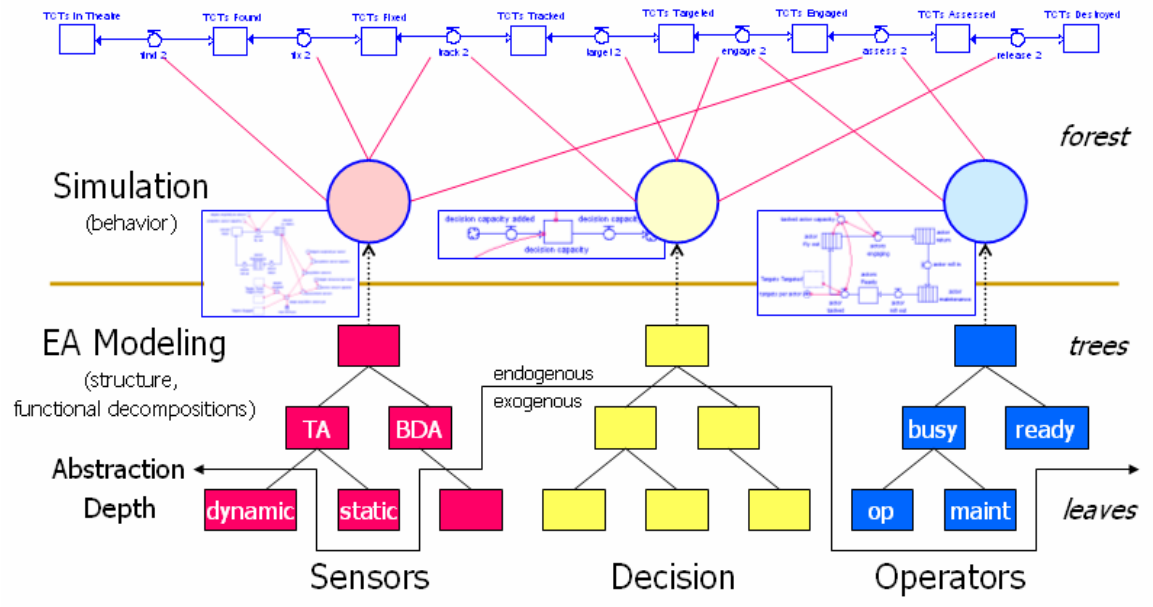

Figure 7 -- Complementary EAP and SD Models

Figure 7 shows how SD and EAP models can combine. The Figure 2 system is comprised of three separate subsystems – sensors, decision elements, and operators – each represented its own decomposition. The SD model selects some but not all details from the EAP model for incorporation into the simulation. Note that for the sensor and operator systems, many details have been incorporated into the SD simulation. A "thinner slice" has been taken from the decision system, making this more of a gap

analysis or requirements generation exercise. Learning that takes place early through simulation will positively impact the subsequent system development effort by generating requirements and metrics, exposing design flaws, and saving taxpayer dollars.

4. Conclusion

The U.S. Department of Defense has a long history of using simulation for training, but the use of simulation to design complex technical systems, while long thought possible (for example, Simulation Based Acquisition) has not yet lived up to its promise. Part of the problem is technical with shortcomings in hardware and software limiting simulation's contributions, but more serious problems regard how simulation has been conceptualized. Very large simulation efforts have historically been undertaken with the intent that the end product apply across a wide range of problems. These efforts have not proven successful due to their inability to represent, organize, and process the huge amount of data included in the simulation. The core problem of simulation is not one of more data and computing power but of *abstraction*. No computer, regardless of its power, will ever be able to process all the details and relationships of reality. This study proposes a different way of envisioning simulation: smaller, more directed efforts focused by a single question with the understanding that not all details and data will be incorporated. Note that this view of simulation is not dependent on or limited to system dynamics, although it is developed in term of this methodology. These observations can also be applied to discrete event, distributed, and agent-based simulations as well as EAP-like methodologies (Tignor 2004). Simulation can thus help design complex C2 systems as well as inform the selection and development of other system design tools.

References

Alberts, David S., John J. Garstka, Frederick P. Stein. 1999. *Network Centric Warfare: Developing and Leveraging Information Superiority*. Washington: CCRP.

Fowler, Martin. 2003. *Patterns of Enterprise Architecture*. Boston: Addison-Wesley.

Glasow, Priscilla A. 2004. "How Cognitive and Behavioral Factors Influence Command and Control." *Phalanx* (March).

Popkin. 2004. "Building an Enterprise Architecture: The Popkin Process (ver. 1.0)." Technical Report, Popkin Software, New York.

Spewak, Steven H. 1995. *Enterprise Architecture Planning: Developing a blueprint for data, applications, and technology*. New York: Wiley-QED.

Sterman, John. 2000. *Business Dynamics*. New York: Prentice Hall.

Tignor, Warren. "System Engineering and System Dynamics Models." Oxford, UK: International Conference of the System Dynamics Society, July 2004.