This is an annotated briefing.
See the Notes Pages for paper.

"*So…,*" asks the Chief Engineer "*What do I go do?*"[1]*

Douglas O. Norman
Brian E. White
(781) 271-8218
bewhite@mitre.org

8 April 2008
Session 2D1–Systems Engineering 2

2nd Annual IEEE Systems Conference
7-10 April 2008
Marriott Chateau-Champlain
Montreal, Quebec, Canada

Public Release Case Number - 07-1347

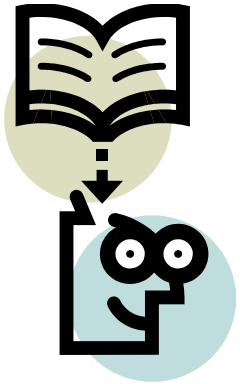
MITRE

** See References at end of briefing*

Introduction

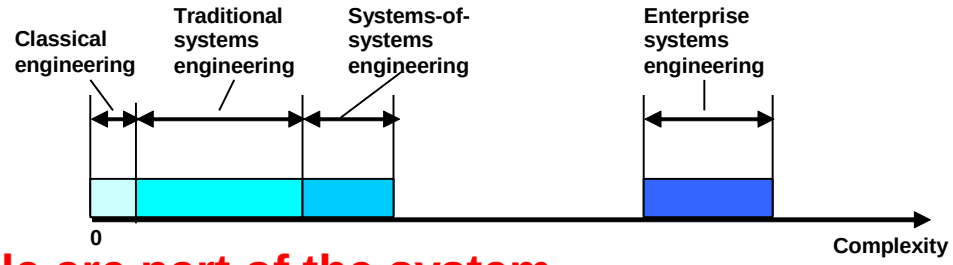
See Notes Page

- We're focusing on the world of systems acquisition
  - Particularly in a military-industrial (complex!) context.
- Using the language of complexity causes problems with most systems engineers
  - It's advisable to have considerable discussion about terminology to reach a "meeting of the minds" as to what our words mean.
- A personal example of this is exhibited on the next chart.



Introduction

See Notes Page

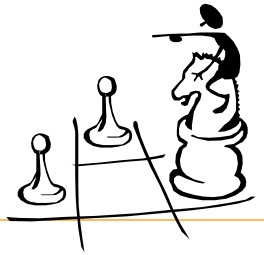


- Complexity is a continuum
  - From complicated
  - To (really) complex, i.e., people are part of the system
- We're focusing on the "messy frontier" of
  - Extreme complexity, i.e.,
  - The most difficult systems engineering problems
- Each system of a system of systems (SoS)
  - Can operate on its own in the environment of the SoS
  - Is independently developed and managed
- Typically enterprises
  - Have elements of homeostasis; and are complex systems
  - But complex systems are not necessarily enterprises; or SoSs
- One is not in control but can only hope to influence
  - Enterprise systems engineers are more concerned with engineering the environment of the systems
  - The gardener or urban planner metaphors apply

Introduction (Concluded)

See Notes Page

- Traditional/conventional systems engineering techniques are great but only in environments that are relatively stable.
  - There, Reductionism and Constructionism techniques can be used
  - Some things we love don't work well in more dynamic situations
- We need to
  - Recognize that processes of natural evolution work well in open, volatile environments
  - Identify and bring these principles into our systems engineering work to improve our practice
- Harnessing complexity of human interactions is key to progress
  - E.g., shape incentive structures and encourage interactions
- So the Chief Engineer (and others) ask, So What (do we do)?
- The main assertion of this talk
  - Systems acquisition can become more successful by developing composable elements that can be fit together quickly by the users to satisfy their immediate mission needs.



See Notes Page

What We Do Today

- Programs tend to be insular, why? Because of the way the acquisition world works! Because of
  - Existing expectations and incentive structures
  - The manner of (and rigidity of) thought (It takes a crisis to change.)
  - What good engineers do
- But, what do engineers do? (What feeds the insularity?)
  - First set of questions asked by a Chief Engineer
    - What are the boundaries for this system?
    - What is the limiting case this system?
  - Second set questions:
    - What is expected? (requirements question)
    - When is it expected? (schedule question)
    - What resources do I control? (cost question)
  - Third (and perhaps the most important) question:
    - How will I be judged?

This is a poor fit to the enterprise goal!

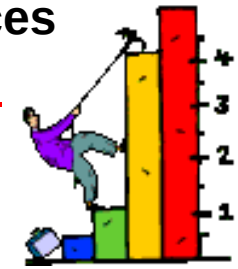
What We Do Today (Concluded)

See Notes Page

- **Unfunded mandates from above**
  - **E.g., for greater interoperability and horizontal integration**
  - **Are dwarfed by**
    - **The stovepiped mentality of legal/financial constraints**
    - **And *especially*, how one is measured and rewarded (or penalized).**

- **Despite the imperative needs, the prevailing culture of most organizations is <u>against</u> the sharing of information.**
  - **Usually there are penalties (not rewards) for sharing!**

- **Similarly, if a service, e.g., of either an infrastructure or application nature, is to be provided for the common good, who ultimately bears the burden?**
  - **Rather than being rewarded, such a provider often experiences**
    - **Greater pressure to continue supplying that service to <u>additional</u> users, or to the same users with <u>enhancements</u>**
    - **But without compensation!**



See Notes Page

What Might We Do Differently?

- From a system Chief Engineer's point of view:
  - Know my system's ***fundamental unique value*** (FUV) which is offered to the enterprise
    - How does one recognize *FUV*?
  - Know how others will/could interact with this fundamental unique value
    - Concept of *technical intimacy*
  - Implement casual interaction mechanisms initially
    - Reduces needed *a priori* agreements among offeror and users
    - Increases likelihood of being composed into flows
    - Provides guideposts on using limited resources for more-intimate interaction mechanisms
      - Apply where most valued
  - Provide a mechanism for reducing the integration barrier
    - Developers networks

Here's An Analogy

See Notes Page

- Suppose each offeror concentrates on their FUV and strips down their offering to that + a good interface(s).
- Pretend the collection of such potential capabilities are like LEGO™ blocks.
- Most users then should be able to assemble a collection of LEGO blocks to create a capability they need in the near term for an important mission.
  - Because of the rough edges they won't be perfect but it will do the job.
  - The process can be iterated taking different needed LEGOs and either warehousing or returning those no longer relevant for the moment.
- Proposal: The producers of the LEGOs that get used are rewarded, and thereby are better able to continue developing more of the same LEGOs or even new LEGOs for potential future use.

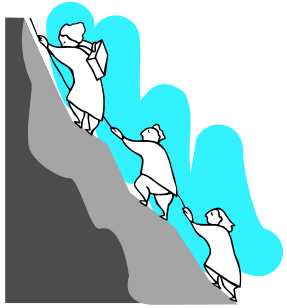
See Notes Page

Reducing The Integration "Barrier"

- **Put in place a *developers' network*, i.e., points-of-presence with offered functionality exposed as live services.**
- **Integration is an existing, open-ended problem for achieving the agility, demanded by the push for net-centricity.**
  - **To date, when we expect to integrate with an existing system, we usually get a copy for our own development environment.**
  - **This requires a long-term commitment to this foreign system.**
  - **This is resource intensive and is untenable when there are many such systems with which to integrate.**
- **As *developers' networks* are stood-up and used, a new dynamic for design, development, and use may emerge.**
  - **Given the lower barriers to use and integration, this dynamic will support exploration and discovery.**
  - **Systems may be de-constructed into composable parts, and then (re-)integrated into collections providing user capabilities.**
  - **Initially, those elements of unique value, being exposed and made available in venues such as developers networks, create new opportunities for those close to the network "edges" to assemble the approximations (or realizations) of the needed capabilities.**

See Notes Page

A Large Unmet Challenge



- These aims will not be achieved on a large scale anytime soon since the prevailing economic and business structures really don't support them well.
- Nevertheless, we should endeavor to continue experimenting with these techniques in pockets of opportunity.
- Such efforts might be successful and will, by example, gradually change, for the better, the way the world works.
- There are many problems to be overcome; it's going to take a different mindset and creative leadership to accomplish change.
- Collections of composable elements of valuable functionality should help the end users to go beyond what they have now, i.e.,
  - Self-service (i.e., building their own tools and solutions using local talent and discretionary funds)
  - Reciting their perceived needs in the form of written requirements that then enter the formal acquisition system (that has long delays).

See Notes Page

Here's An Example of Fundamental Unique Value

- What's a FUV of the U.S. Air Force's Air Operations Center's (AOC's) Theater Battle Management Core System (TBMCS)?
- It's the Air Tasking Order (ATO)—*but is it?*
- What if you're a pilot about to enter your aircraft with a compact disk, or whatever, that contains the information needed to complete a successful mission.
  - If I come along with a laptop, or whatever, ask for your CD, then change the code, and hand it back, how would you feel?
  - Chances are you would not accept it and would refuse to fly the mission because your success may be jeopardized.
- So the fundamental unique value of is the integrity of the ATO, it's credibility based on the authoritative, competent source
  - Not the software in or sheet of paper on which the ATO is written!

See Notes Page

Summary

- "So…," asks the Chief Engineer "What do I go do?"
  - Know and concentrate on your system's fundamental unique value provided to the enterprise
  - Have multiple ways to interact with your FUV
    - Favor casual methods initially
    - Then more intimate methods where demonstrated need exists
    - Make it easy for others to "integrate" with your offering(s)
  - Design with replaceability, not reuseability, in mind
    - This also favors your use of others' offerings
  - Monitor the actual use of your offerings
    - Collect field experiences and let that inform change

It's an achievable start!

Back Up Chart

See Notes Page

MITRE

Back Up Chart (Concluded)

See Notes Page

References

[1] Norman, D. O., *"So…," asks the Chief Engineer "What do I go do?,"* Invited Keynote Talk, Complex07, The 8th Asia-Pacific Complex Systems Conference, Brisbane, Queensland, Australia, 3-5 July 2007

[2] The International Council on Systems Engineering's (INCOSE's) Systems Science Enabler Group (SSEG), for example, http://www.incose.org/practice/techactivities/wg/sseg/

[3] White, B. E., "On Interpreting Scale (or View) and Emergence in Complex Systems Engineering," 1st Annual IEEE Systems Conference, Honolulu, HI, 9-12 April 2007

[4] Dawkins, R., *The Selfish Gene*, Oxford University Press, 1976

[5] Kuras, M. L., "An Introduction to Complex-System Engineering," 7th International Conference on Complex Systems, New England Complex Systems Institute (NECSI), Boston, MA, 28 October - 2 November 2007

[6] Norman, D. O., and M. L. Kuras, "Engineering Complex Systems," Chapter 10, *Complex Engineered Systems: Science Meets Technology*, D. Braha, A. Minai, Y. Bar-Yam (Eds.), New England Complex Systems Institute, Cambridge, MA, Springer, 2006

[7] Systems Engineering Handbook, Version 3.1, INCOSE at URL: http://www.incose.org/ProductsPubs/products/sehandbook.aspx

[8] Csetel, Marie E., and John C. Doyle, "Reverse Engineering of Biological Complexity," DOI: 10.1126/science.1069981, Vol. 295, Science, 1 March 2002, pp. 1164-1169 http://www.sciencemag.org

[9] Kuras, M. L., and B. E. White, "Engineering Enterprises Using Complex-System Engineering," INCOSE Symposium, Rochester, NY, 10-15 July 2005