The Case for a Systems Focus in Healthcare

Richard Schrenker

As medical devices increasingly become interconnected systems within systems, the question looms: who is making sure that all of these systems work together? The stakes are high, as the effects of failure of any one part of a system are likely no longer isolated to a single device, but rather threaten to ripple outward, affecting disparate systems in ways ranging from minor to catastrophic.

Clinical engineers, trained in systems theory and uniquely situated at the nexus of these converging technologies, would appear to be ideally positioned to apply its systems-oriented focus to assuring patient safety through this transition. But it is not clear that the broader systems engineering capabilities of clinical engineering are fully recognized. For that to happen, clinical engineering may need to make the case explicitly to non-engineering healthcare leadership that clinical engineers can address the increasingly broad spectrum of engineering problems that will continue to face healthcare.

The following reads like something pulled straight out of a modern systems engineering text: “We need first to survey user needs and define the objectives of the system to be built, based on the system in use. Once we have those objectives, then we can define requirements. [The engineer] must engage in research, market analysis, and user consultation before attempting to make old systems efficient or to interface new ideas with the user…”

But just a few pages before, the following can be found: “The issue is really how to increase the desired one-to-one relationship in healthcare delivery. Only that result can maintain or increase the quality of healthcare delivery.”

Something from the current healthcare reform debate including a discussion of meaningful use? No, the source of the above is Cesar Caceres’ *The Practice of Clinical Engineering*, published in 1977. More recently, The International Council of Systems Engineering (INCOSE) described the concerns of systems engineering: “Systems engineering is concerned with the overall process of defining, developing, operating, maintaining, and ultimately replacing quality systems. Where other engineering disciplines concentrate on the details of individual aspects of a system (electronics, mechanics, ergonomics, aerodynamics, software, etc.), systems engineering is concerned with the integration of all of these aspects into a coherent and effective system. Systems engineers concentrate their efforts on the aspects of the engineering process (requirements definition, top-level functional designs, project management, life cycle cost analysis…) that serve to organize and coordinate other engineering activities.”

Although wordier, this is not all that different from the 1992 description/definition from the American College of Clinical Engineering (ACCE): “A Clinical Engineer is a professional who supports and advances patient care by applying engineering and managerial skills to healthcare technology.”

Clinical engineering is not yet broadly recognized as healthcare’s version of systems engineering. As the National Academy of Engineering and Institute of Medicine’s “Building a Better Delivery System: A New Engineering/Healthcare Partnership” notes: “…engineering educators face the challenge of cultivating demand for healthcare delivery-trained engineering graduates in an industry that has traditionally hired very few engineers and currently has no clearly defined career tracks for engineers…”

Among the disciplines noted as sorely needed were concurrent engineering, human factors engineering, failure analysis, and modeling and simulation.

Consider also the following from the National Acad-
emy of Sciences’ Computational Technology for Effective Healthcare: Immediate Steps and Strategic Directions:

“It is fair to say that the integration of healthcare IT into operational work processes has proven both more essential and more difficult than was first expected, at least in part because many attempts to deploy healthcare IT have not taken into account the systems engineering issues inherent in viewing healthcare as a complex, adaptive system…”

So why is there a gap between the recognition of the need for systems engineering professionals in healthcare and the recognition that they may already be present? Clues may be inferred from the roundtable article in the 2008 edition IT Horizons. While most of the discussion focused on the socio-technical issues associated with clinical engineering and IT convergence, Steve Grimes noted at least four times the importance of clinical engineering taking on more of a systems perspective. At one point Carol Davis-Smith noted “We are too focused on the ramifications of biomed/IT convergence…”

Davis-Smith’s comment reminded me of an old joke about a former world champion pool player named Minnesota Fats. A newspaper writer explained why he was so successful: “While the other guy is playing Minnesota Fats, Minnesota Fats is playing pool.”

Where should clinical engineering direct its attention and effort? Is its goal to achieve cultural equilibrium with IT or something else? Grimes had it right; the focus should be on addressing the problems our National Academies have identified as critical needs. And that reminds me of hockey great Wayne Gretzky’s crediting his success to “skating where the puck is going to be.”

If clinical engineers hope to claim their role as the guarantors of patient safety in this increasingly technologically intensive world, they must at least consider adopting as their top priority bringing rigorous engineering practice to ensuring that the systems being rapidly implemented in healthcare work safely and effectively.

Another compelling aspect to the call for a systems focus occurred to me during a workshop I attended this past June on the topic of assurance and safety cases and which included representation from the Federal Aviation Administration (FAA), Nuclear Regulatory Commission (NRC), and U.S. Food and Drug Administration (FDA). Early on we all realized the similarities across many of the systems-level technology issues facing us, but near the end one difference that jumped out at me was how FAA and NRC described how their regulatory authority extended across the use of technology as well as its supply, whereas FDA authority is essentially limited to the supply (manufacturing) side.

If FDA is not watching out for how these technologies are being used in healthcare, who is? Given the continuing “cottage industry” nature of the U.S. healthcare delivery system, this arguably suggests a need for even more rigorous attention to systems engineering the closer one gets to the point of care. And again, I would argue that clinical engineers are uniquely qualified to play this role.

What makes systems systems? And what’s the big deal?

We build systems to provide value beyond that which its constituent components can provide alone. Consider what occurs at the point of care. Patient status follows a trajectory that is not fully deterministic, and the associated dynamics both impact and are impacted by environmental, operational, and situational contexts that evolve with time. System components may be added or removed, clinical experts come and go, and patients move between more or less technology-intensive environments.

Over time, events emerge that need to be evaluated in terms of not only the current state of the patient and system but also their history. Implicit real-time patient care system construction and integration in the minds of clinicians is nothing new and goes on continuously. But the introduction of IT that leverages medical devices to inform and be informed by clinicians, support staff, and other devices and device-based systems without the actions of a human intermediary is quite new.

Intentionally, and all too often unintentionally, properties and functionality in addition to those associated with the constituent components are introduced. Only
by understanding and appreciating the system in its situational context can these problems be appreciated. Only by applying broad-based engineering skills can they be fully addressed.

The Need for Systems Engineering
It would be reasonable on being introduced to formal systems engineering to reject it with an argument of obviousness and a question of what’s to be gained from applying its principles, which certainly come at a cost. It may be helpful to consider but one of the stories in the public domain that cites some of the consequences of not adopting systems engineering practices and principles.

“Why have hospitals like Cedars-Sinai, which have made huge investments in IT, encountered enormous problems when it was implemented?… The answer, in part, is that no one thought about how the new system would change the work process. Doctors were being asked to use a brand new information system that added hours to their workday… [so] they know from experience that new systems do not always improve it.”

This may not seem pertinent to an end user community accustomed to being on the receiving end of a medical device regulatory process that usually results in the provision of safe and effective devices, including those that are software-based. But consider what consequences may result from not having clinical systems engineering oversight of the development of medical device networks.

In his book Normal Accidents: Living with High-Risk Technologies, Charles Perrow warned of what he argued would be inevitable (hence “normal”) significant accidents deriving from system designs as they grew in size, diversity of function, interactivity between components, and accessibility to indirect information sources. Perrow identified medical equipment, medical procedures, the Internet, telecommunications, and software as areas that should be evaluated for normal accident potential.

Emerging Requirements for Networks
The integration of systems in today’s multivendor environment move us beyond technical and cultural issues. The practice of healthcare is undergoing fundamental changes and we have the opportunity to make an impact on it.

Consider for instance the emergence of IEC 80001-1, Application of risk management for IT-networks incorporating medical devices—Part 1: Roles, responsibilities and activities. This new standard is being balloted as I’m writing this. This standard aims to apply the requirements of ISO 14791, Application of risk management to medical devices, to the use of medical devices on IT infrastructures.

In leveraging ISO 14971, a risk management standard specific to medical devices, it once again begs the question as to when does a network become a medical device? It also introduces the term “Responsible Organization,” which seems to be very much like what ISO 14971 calls a “Manufacturer.”

Note that the draft standard differentiates between what is expected of “Medical Device Manufacturers” and “IT Providers” in terms of supporting the risk management process. Why? Given the warnings of Perrow, why wouldn’t we expect the risks associated with the use of networks to be any less consequential than the medical devices using them to communicate?

But setting this aside, the standard most certainly changes the nature of technology management in hospitals in a direction oriented towards formal systems management in general and risk management in particular. Adoption of IEC 80001 will create demand for systems engineering services that will provide an opportunity to which clinical engineering must respond. Clinical engineering has to jump outside the proverbial equipment management box and learn from systems and software engineering experience external to healthcare.

Similar in many ways to modern systems engineering, software engineering shares many of its problems. Indeed, many of its standards echo one another, and the lessons of software engineering serve to equally inform systems engineering. Consider this basic understanding from software engineering as a pointer to some of the services clinical systems engineering could supply:

“The hardest part of building a software system is deciding precisely what to build. No other part of the conceptual work is so difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later… it is really impossible for clients, even those working with software engineers, to specify completely, precisely, and correctly the exact requirements of a modern software product before having built and tried some versions of the product they are specifying…”

Removing the word “software” from the above leaves an arguably valid general statement about systems. And while there is nothing explicit in this description of the
need for systems engineering services that points to a particular reason to turn to clinical engineering, findings from research in the subspecialty of requirements engineering offer a glimpse into what clinical engineering can offer:

“Safety-related functional faults are more likely than non-safety-related functional faults to be caused by requirements that have not been identified... unknown, undocumented, or wrong requirements are a greater cause of safety-related than non-safety-related errors...” 13

My work has provided me some opportunity along these lines, both in our interoperability systems research and development work in the Medical Device “Plug-and-Play” Interoperability Program (MD PnP, www.mdppn.org) and my role as software development manager for my department’s Systems Engineering group.

In the course of a requirements engineering project for MD PnP, we contracted with a software engineering consultant to advise us on aspects of requirements engineering. One piece of advice particularly stood out: Developers have to make thousands of decisions when developing a software product. If the requirements necessary to make a decision are not available when they need them, they will define them for themselves (consciously or otherwise). It is therefore critically important that all safety requirements be identified explicitly, completely, and correctly early, preferably before design. Who better to define these requirements than clinical engineering?

For a more thorough discussion of how as well as why to do requirements engineering, see Kevin Gary’s article on an open source software toolkit (see sidebar) and some other work.14-17

Validation and Verification
Another and perhaps more obvious broad spectrum engineering services opportunity for clinical engineering lies in the area of validation and verification. My department recognized the need when we experienced a very real upgrade-gone-bad event in 2004. Here is the epilogue to that story:

“An independent consultant was later hired by the department to assess the manufacturer’s software development and validation procedures. They found that [the manufacturer] had followed industry standards and met FDA regulations. However, they also recommended that testing strategies employ a more diverse set of operators, under the broadest possible methods of operation, and without following specific test procedures. In addition, beta testing at multiple client sites, in non-life-critical applications, could extend this level of testing... this is good advice for all manufacturers.

This event also gave the department and the entire Biomedical Engineering community a chance to consider how to improve their systems and operations. More comprehensive testing protocols based on detailed vendor descriptions of software changes needed to be developed. Taking this a step further, medical device software should be tested using clinical scenarios that represent diverse users and clinical settings. A safe venue for truly realistic testing would be in simulation laboratories. This approach would not only improve patient care, but lay a foundation to prevent similar events.”18

A close cousin of requirements engineering, the process of validation can be just as difficult, especially for software-based systems. Ultimately, paraphrasing the late U.S. Rep. Tip O’Neill of Massachusetts, all validation is local:

Verification: Ensuring you built the thing right.
Validation: Ensuring you built the right thing.

While verification can be done against a specification, validation can only be done with the people who will use or support what was built and who were (or should have been) the source of functional and non-functional requirements. With healthcare being a cottage industry, this effectively precludes completing validation before implementation on site, especially for increasingly complex systems being integrated into already technically and clinically heterogeneous environments. Of course, there is a great deal of commonality across the healthcare system, and not all features and services have to be validated at each and every site. Otherwise, there would be no medical device industry to speak of. So what should a site validate?

That seemingly simple question is not so easily answered. As part of our response to the event described above, our department has been evolving a set of software maintenance policies and practices. Early on we recognized the difficulty and cost involved in validating even relatively simple systems, and it occurred to us that access to information from the validation done by the manufacturer could inform our work tremendously. However, manufacturers have been extremely reluctant to provide us this information. They cite understandable business reasons, but we remain left with the problem of being unable to reasonably cite our degree of confidence
that implemented systems will be as safe and effective as we require. We hope that ongoing work along the lines of what is occurring with standards work like IEC 80001 and the Integrated Clinical Environment will yield tools to help address this dilemma.

My old-school habits being hard to break, I continue to prefer general tools to those that are specific to a device or system. Tools that are made hand-in-hand with the target device or system run the risk of inheriting any fallacious assumptions or analyses built into the design or implementation of the target. In December 2008 I reported in B&IT on an emerging family of general tools that I have hope for: safety and assurance cases. My confidence in their potential generalizability was strengthened at the workshop noted above that brought people from FAA, FDA, and NRC looking for the same thing. Case-based assurance methods are also generalizable in that they can be applied to all functional and non-functional claims, not just safety. Even evidence garnered from device-specific tools can be cited in a case, as long as the citation includes arguments that indicate (for example) any associated assumptions and the degree of confidence in validity.

The Role of Clinical Engineering

For the sake of healthcare, hopefully the focus on systems engineering in clinical engineering will continue, no matter where clinical engineering resides or what it comes to be called. What leads anyone to believe that information technology represents healthcare’s let alone clinical engineering’s final technological horizon? The details may change, but the nature of the need for systems engineering at the point of care will not.

References

Software Engineering Resources

Rick Schrenker served as guest editor for a special issue of IEEE Computer focused on clinical software engineering (April 2006). The following lists the articles from that issue:

I Lee et al, “High-Confidence Medical Devices Software and Systems”
S Rakitin, “Coping with Defective Software in Medical Devices”
V Gehlot and E Sloane, “Ensuring Patient Safety in Wireless Medical Device Networks”