Systems Engineering: Art and Science in an International Context

by
John S. MacDonald
MacDonald Dettwiler
Richmond, B. C., Canada

Good morning, and allow me to bid you welcome to one of the world's most attractive cities. It is a genuine pleasure and source of pride for us here on Canada's west coast to host this important gathering which, as I understand it, is being held for the first time outside of the U.S. We want you all to know that world class systems engineering practices are alive and well here in the northwest corner of the continent and all of us from this area look forward to informative and lively discussions with our colleagues from other places over the next few days.

As a practicing engineer, I have lived through the formative stages of systems engineering as a formal discipline and have seen it evolve from quite ad-hoc origins to the rigorous discipline we practice, or at least like to think we practice, today. I have also spent a substantial part of my career involved in the development and marketing of advanced systems in a global marketplace mostly in Australasia, Europe, the Middle East and Latin America as well as in North America. In my remarks to you this morning, I will try to draw on that collective experience and perhaps, if I have analyzed the requirements correctly, designed an intelligent approach, implemented my solution within schedule and budget and tested it rigorously, I will satisfy the requirements of you the users of this product by giving each of you a little something new to think about as you work your way through the rest of the week's activities. If not, well, maybe I can show up in Brighton next year with a thoroughly debugged keynote speech.

Mr. Jaswinder Madhur of Rational Software Corporation, Chair of the 1998 INCOSE Symposium (left) with
Dr. John MacDonald of MacDonald Dettwiler and Associates Ltd. (right).

Presented at the 1998 INCOSE Symposium.
INCOSE 1-206-441-1164, 800-366-1164 (toll free U.S.) incose@halcyon.com and www.incose.org

One of the things that has always fascinated me about engineering in general, and systems engineering in particular, is the incredible creativity involved. It has always been my belief that those of us who practice this craft, in particular those who create great engineering works, achieve a degree of creativity that is, without a doubt, in the same league with the great artists who have who have inspired the human spirit with their creations. Yet engineers are not generally regarded in this way. A bridge is a bridge, spacecraft are spacecraft and the elegance and beauty in an excellent piece of software are forever hidden from the public view. Indeed the creativity in these things in many cases manifests itself not in some obvious elegance that everyone can see and appreciate, but in the very existence of the thing in question. Spacecraft, automobiles, telephones, computers, the internet ... the list goes on and on and on ... exist because the creative talents of engineers enabled them to find ways to bring these things into being within the constraints, both financial and material, of the resources that were available to them.

When one thinks of creativity in engineering, one is tempted to draw parallels between the creation of engineering works and the creation of works of art as we understand it in the normal context. To one who can appreciate the aesthetics of it, an excellent engineering work is like a beautiful painting, an outstanding sculpture or a Mozart symphony. But there are differences. Engineering works are meant to be used, not just appreciated. They have to, as we say in the systems engineering business, "satisfy the requirements of the user." They are also the result of teamwork; rarely are they the creative inspiration of a single individual and they are never implemented by one person alone. In all cases the objective is to create a system within schedule and budget constraints as well as those imposed by the natural world. That objective is even achieved on occasion! Engineering and art are therefore quite different, but they do share an aesthetic sense which is a source of pleasure and satisfaction to those who do the creating.

In a sense, one can argue that systems engineering involves a higher degree of creativity than does a work of art. The constraints one works under are more confining and each of the steps one takes requires the ability to solve a set of problems in new ways. This is true not just in the initial design or architecting stage; it is also true in the implementation stage as well as the testing stage. The challenge is always to be able to balance the creative instincts of the engineer with the schedule and financial constraints that the team must work under -- to balance creativity with discipline. The formation of the team itself is a highly cerebral process. The first step is to select the best people one can pry out of other areas of the organization. The next and more difficult step is to place those people in the positions where they will be most productive and effective. Creative people are not all the same. This is certainly true for engineers; we are not identical and interchangeable. On top of that, in order to construct an effective team one must have access to a broad range of talents. In the architecting stage, one needs those who are good communicators, who are innovative, imaginative and creative BUT who also are cost conscious and are capable of resisting the urge to continually improve things without consideration of schedule and cost. These are the rarest of birds, but it is from this group that the leaders are drawn. For analysis we need those who are mathematical modelers, whose whole life revolves around highly technical concepts and who care about little else. These are what we call today the "nerds", a term which I use in the most positive context. In the implementation phase we need detail-oriented engineers with good writing skills and a broad experience base. In system testing we need skeptics with the ability to

look beyond the obvious, to solve puzzles and sometimes break things. This is quite a mixture, and is clearly an oversimplification of reality, but it serves to illustrate the challenge of team-forming, the creativity involved in that process itself and the creativity involved in the contributions made to the project by each of these diverse talents. It also illustrates one of the most important principles of engineering management: The success of a project, indeed of a manager, is critically dependent on the quality of talent employed, and, more importantly, the manner in which that talent is deployed on the project.

We live in an increasingly complex world., and nowhere is this more true than in the systems engineering business. The relentless advance of modern technology enables us to realistically solve ever more complex problems. Technological advancement in computing hardware together with the reduction in cost of the modern computer is the largest single cause of this phenomenon. This, in turn, leads to the fact that the development of large systems today is increasingly dominated by software. Indeed, my first exposure to many of the principles we now accept as the fundamental maxims of systems engineering was a small book on software engineering I read around twenty years ago. It was whimsically titled "The Mythical Man-month" and it was written by Fred Brooks, who was the first manager of the OS/360 development team at IBM in the 1960's. Those of us old enough to recall this project will remember that it was not one of engineering's finer hours, and this fact prompted Fred to write the book. I am sure some of you have read this book, but nevertheless I would like to share a few of its passages with you this morning. The bits of wisdom contained here are as true today as when they were written over twenty years ago.

The first of these is on the optimism of programmers:

> "All programmers are optimists. Perhaps this modern sorcery especially attracts those who believe in happy endings and fairy godmothers. Perhaps the hundreds of nitty frustrations drive away all but those who habitually focus on the end goal. Perhaps it is merely that computers are young, programmers are younger, and the young are always optimists. But however the selection process works, the result is indisputable: "This time it will surely run," or " I just found the last bug." So the false assumption that underlies the scheduling of systems programming is that *all will go well*, that *each task will take only as long as it "ought" to take*."

Nothing much has changed in the intervening twenty or so years regarding the optimism of programmers, but hopefully we have developed tools, procedures and techniques that better equip us to deal with this optimism through realistic estimation and management approaches.

A second pearl of wisdom:

> "The second fallacious thought mode is expressed in the very unit of effort used in estimating and scheduling: the man-month. Cost does indeed vary as the product of the number of people and the number of months. Progress does not. *Hence the man-month as a unit for measuring the size of a job is a dangerous and deceptive myth*. It implies that people and months are interchangeable.
>
> People and months are interchangeable commodities only when a task can be partitioned among many workers *with no communication among them*. This is true of reaping wheat or picking cotton; it is not even approximately true of systems programming."

Or, might I add, systems engineering.

This, of course speaks to the point I made a few moments ago that creative people are not all the same and that they are certainly not interchangeable. It also speaks however to the arts of team-forming and project management: the art of assembling the right team for the job at hand and the art of managing a process which demands intense creativity and intense intercommunication among team members if success is to be achieved.

One cannot overemphasize the importance of team-forming in the ultimate success of a project. Getting the team right at the beginning and creating a solid workplan are the two major prerequisites for ultimate success. These two are intimately tied together: The plan determines the team and the team does the plan. It is thus an iterative process. It is worth remembering in all of this that the actual coding of a successful project takes much less time than the planning, design and testing phases. Typical numbers for the software business are one-third planning and design, one-sixth coding, one-quarter component testing and early system test and one-quarter system testing with all components in hand. Thus one can see that typically half the time is spent testing and debugging, one-third of the time designing and planning and the bit of time remaining to actually code the thing. Since different skills and personalities are required to perform each of these tasks, it can be readily understood that there is no substitute for careful planning and team formation if overruns and later confusion, not to mention disaster, are to be avoided. The disproportionate amount of time that should be allocated to the testing phase is particularly critical. If you get this wrong, schedule slippage does not appear until late in the project. The delivery date is looming and bad news late and without warning is unsettling to everybody, especially the customer. There is a tendency to panic -- which leads me to one final quotation from Fred Brooks:

"Oversimplifying outrageously, we state Brooks's Law:

Adding manpower to a late software project makes it later.

This then is the demythologizing of the man-month. The number of months of a project depends upon its sequential constraints. The maximum number of people depends upon the number of independent subtasks. From these two quantities one can derive schedules using fewer people and more months. (The only risk is product obsolescence). One cannot, however, get workable schedules using more people and fewer months. More software projects have gone awry for lack of calendar time than for all other causes combined."

When you read a book and remember it for the rest of your life, quite often it is because of one thing you learned from that book. So it was when I read "The Mythical Man-Month". I was motivated to read it because the company was in some serious financial trouble due to a large project overrun. Brooks's Law "*Adding manpower to a late software project makes it later."* was the important thing I learned. We "toughed it out", we stuck with the original team and survived the experience. We became a leading practitioner and proponent of rigorous system engineering principles, and grew by another order of magnitude over the ensuing sixteen years during which time most of our software development projects have been done on a fixed price basis. Our success is evidenced by the fact that we are still here and still growing.

Around half of our business has typically been outside of North America, and this brings me to the second aspect of my message this morning.

All of us live and operate in an increasingly global economy. The opportunities within this economy are diverse and expanding. A great many of these opportunities are in what is loosely referred to as the "developing world", and are generated by the need to acquire the tools of modern civilization. But as far as systems engineering is concerned, there is, by and large, little understanding of the issues that will be discussed here in the next few days and the business environment has characteristics which are quite foreign to many of us. At the same time, there is increasing pressure for "cooperative projects" driven by the associated lure of "technology transfer". Thus, projects to be executed in such places must, like all projects, be carefully planned but, in addition to all the usual considerations, they must be planned with full knowledge of the characteristics of the cultures in which one must operate and the implications of those characterisitics as they will affect the project.

Doing business in this marketplace requires an understanding of, and an ability to operate in, cultures which are different from our own. To me this has always been the greatest challenge of doing business in the global market. It is also the thing which has made the exercise most interesting. Other cultures interpret things differently from the way we do. Even things as fundamental as "right" and "wrong" differ from culture to culture. The idea of a "fixed" requirement is a concept foreign to many cultures. Systems engineering is a culture too and it is intimately tied to our North American value-set and way of doing things. Thus when one ventures into the big world out there to places where a schedule slip is a huge loss of face and must be swept under the rug in spite of the obvious passing of time, or where contractual terms are regarded as "flexible" throughout the project, or where fixed requirements are a confusing impediment in the eyes of the customer or where the passage of time is of minor importance, to mention but a few of the characteristics of such places, a whole new gamut of interesting contingencies arises. Of course all of this is happening in a competitive environment where the customer's procurement system is as leaky as a sieve and final negotiations make the concept of a BAFO as practiced in North America look like some quaint relic of the Victorian era or the Age of Chivalry.

When one speaks of inter-cultural skills, it is important to remember that business practices which, in our culture, we would consider as being wrong, in another culture can be regarded as quite acceptable or even the standard way of doing things. It is important to remember that these things are not necessarily "wrong" or "unfair" as they would be if passed through our value-set, but are simply "different" and have to be accounted for in the planning process. Attempting to impress the North American value-set upon another culture is an exercise in futility and will lead to nothing more than an expensive marketing exercise with no return. The first prerequisite of any business arrangement is to show respect for the culture and value-set of your customer, no matter how strange they may seem when viewed from the vantage point of your own cultural background.

To a person whose main experience has been within the protective cocoon of the defence business, this must seem like a daunting challenge, but if one's organization is to succeed and

prosper in the world of the next century, acquiring the inter-cultural skills necessary to do business in the emerging global economy is an investment that simply must be made.

As I stated at the beginning of this address I have always been fascinated by the degree of creativity in the engineering process. I also said that one is tempted to compare systems engineering with art from the point of view of the creativity involved. Perhaps the best parallel we can draw from the world of art is with the creation of a symphony. While, unlike most modern systems, a symphony is the creation of a single mind, it is implemented over and over again by a large team of people whose output is determined by the physical constraints of the instruments they use, it is tested in rehearsal, and each performance of the work gives pleasure to a greater or lesser degree to an audience. The performers and the audience interact with the mind of the maker of the work. So it is with systems engineering: the users interact with the minds of the makers of the system. We are those makers; and it is from this that many of us draw our greatest satisfaction. Clearly, systems engineering is more art than science and each of us is an artist who uses his or her knowledge of science to create something of usefulness.

So as each of you sits through the sessions over the next few days and interacts with your colleagues, take pride in your art and the creativity which it manifests, be it technical or organizational. As you create your own symphonies over the coming years remember that from all historical accounts Mozart was a nerd. But he created fine things.

Thank you for your attention.

Biography of **Dr. John MacDonald**: As one of the founders of MacDonald Dettwiler and Associates Ltd. (MDA) and currently the Chairman of the Board, INCOSE '98 is honoured to have Dr. John MacDonald as our keynote speaker. Dr. MacDonald's professional interests helped pioneer the development of advanced digital systems engineering, remote sensing and image processing. He also led the design team for the first LANDSAT ground processing system produced by MDA and was involved in the early development of synthetic aperture radar processing at the company. He is also active as an advisor to government, serving as Canada's member of the Eminent Persons Group of Asia Pacific Economic Cooperation (APECO) from 1993 to 1995. In the industrial sector, Dr. MacDonald is a Director of several companies including the Geosat Committee in the U.S., Analytical Spectral Devices Inc. of Boulder, Colorado, ST Systems of San Mateo, California, Radarsat International Inc. of Richmond, B.C., and Kinetic Sciences of Vancouver, B.C. He is also a Director of the Earth Space Institute of Paris, France. Dr. MacDonald is a registered Professional Engineer, a Fellow of the Institute of Electrical and Electronic Engineers (IEEE), a Founding Fellow of the Canadian Academy of Engineering, and a Fellow of the Canadian Aeronautics and Space Institute.

The mission of the International Council on Systems Engineering (INCOSE), a professional society, is to *foster the definition, ever increasing understanding and practice of World Class Systems Engineering in industry, academia. and the public sector.* Learn about the INCOSE at **www.incose.org**.