 DOI: 10.2514/6.2023-3772

Digital Engineering is too important to be left alone to MBSE Practitioners

Rao Mannepalli[1]
Leidos, 11464 Wildwood Ridge Dr. Colorado Springs, CO 80921

Weapon systems critical to national security are increasingly becoming more complex and expensive. The cost overruns alone are estimated to be about half-a-trillion Dollars, making it the second largest defense budget in the world. To address this problem, DOD introduced the Digital Engineering initiative in 2018. Even though its vision and scope are laudable, not all the areas/disciplines involved in realizing these weapon systems are excited. There is a certain poverty of imagination in naming the vision as "Digital Engineering." Most people misunderstood it to be an engineering initiative (in particular Systems Engineering). This paper redefines the term "Engineering" in Digital Engineering as a process (e.g., engineering a political coup rather than like Mechanical/Electrical/Systems Engineering), which then addresses all the five pillars of Digital Engineering. Another disturbing trend noticed by the author is that the Systems Engineering community equated Model Based Systems Engineering (MBSE) to Digital Engineering. This prevents new thinking and thereby limits or negates the vision of Digital Engineering. This paper also touches upon the origins of Systems Engineering at Bell Labs, obsession with the term "Model" to mean anything and everything, including a text document or a static picture, and Digital Twins. It gives practical and useful definitions of "Model" and "Digital Twin" and provides examples and some best practices followed in developing them. It also lists some of the disruptive technologies like ChatGPT which could potentially even eliminate the need to write computer programs. These systems can fuse Unstructured, Uncertain, Incomplete, Imprecise, and even Contradictory (UUIIC) information from all sources. This will make the current processes obsolete, including Systems Engineering. Finally, it presents the lessons learned and makes some recommendations for the near-term, including how to prevent Digital Twins from becoming "Evil Twins."

I. Introduction

Weapon systems critical to national security are increasingly becoming more complex and expensive. Most of the time, these are system-of-systems. Some reports [1, 2] estimated that the cost overruns of the top 10 weapons systems of the U.S. Department of Defense (DOD) alone make it the third largest defense budget in the world. A more recent study [3] estimated that the combined cost overruns for Major Defense Acquisition Program (MDAP) portfolio programs in 2015 was $468 billion, up from $295 billion in 2008, making it the undisputed second largest defense budget in the world. Adding to this, some of the systems developed more than half-a-century ago, during the Cold War, are still in use. PARCS radar, UWER radar, and B-52 are some examples [4-6]. Maintaining and sustaining these systems - designed with slide rules, pencils, and paper - is becoming exceedingly expensive. In the case of some systems like PARCS, where the documentation was partially destroyed as per the Anti-Ballistic Missile Treaty [7], the problem is even worse.

To address these problems, DOD, under the leadership of Dr. Mike Griffin, Undersecretary of Acquisition, Technology and Logistics (ATL), introduced DOD's Digital Engineering initiative in 2018 [8]. This is a far-reaching vision intended to revolutionize the way DOD weapons systems are designed, built, tested, maintained, and sustained. It is expected to result in greater efficiency and improve the quality of all acquisition activities.

The Digital Engineering vision has 5 goals/pillars. They are: (1) Formalize the development, integration, and use of models to inform enterprise and program decision making; (2) Provide an enduring, authoritative source of truth

[1] Chief Engineer, Lead, Digital Twins Special Interest Group (SIG) in Digital Engineering Center of Excellence (COE), rao.mannepalli@gmail.com, Associate Fellow of AIAA.

(through the Models created in the previous step); (3) Incorporate technological innovation to improve the engineering practice; (4) Establish a supporting infrastructure and environments to perform activities, collaborate, and communicate across stakeholders; and (5) Transform the culture and workforce to adopt and support Digital Engineering across the lifecycle.

Digital Engineering is often misunderstood. There is no confusion in the "Digital" part of it. Nowadays, everything is done digitally and people have even begun referencing "Information Technology (IT)" as "Digital Technology." The confusion is in the "Engineering" part of the name. The "Engineering" is in the sense of "Engineering a Political Coup" rather than "Engineering" as in "Mechanical Engineering.[2]" The "Engineering" here is a process that encompasses all 5 pillars and the main purpose is to develop and deliver the best weapons systems, which need not be fired in anger,[3] at an affordable cost.

Apart from Engineering, a large number of specialties are involved in realizing the weapons systems. Defense Acquisition Workforce Improvement Act (DAWIA) certifies the defense acquisition workforce in the following areas [11]:

- Auditing
- Business Cost Estimating
- Business Financial Management
- Contracting
- Facilities Engineering
- Industrial/Contract Property Management
- Information Technology
- Software
- Cybersecurity
- Life Cycle Logistics
- Production, Quality and Manufacturing
- Program Management
- Purchasing
- Small Business
- Systems Planning, Research, Development and Engineering
- Program Systems Engineering
- Science and Technology Management
- Test and Evaluation

From the above list, we can see that Systems Engineering is a small part in the whole scheme of things and MBSE is even smaller. Hence, the term "Digital Engineering" is a misnomer. To borrow the words of the creators of AWK programming language,[4] "there is certain poverty of imagination" in selecting that name. The scope envisioned in Digital Engineering is much broader. A better name should have been chosen which represents the breadth and depth of the vision. For example, Better Buying Power (BBP)[5] telegraphs its intent better. A name which represents its full vision/portfolio and for which every discipline can relate to (at least loosely) but no one owns it, will serve the purpose better.

II. Origin of "Systems Engineering" at Bell Labs

The term "Systems Engineering" originated at Bell Labs [14]. It is not the omnipotent role that is being made of now. In those days, the telephone system was called the Bell System. The folklore[6] is: there was a charge code for every major technology (e.g., Area-11 (fundamental research), electronics, switching, amplification, transmission, billing, etc.) [15-17]. There was one even for IEEE because shaping the standards and the future was considered a strategic initiative [18-20]. There were some people who worked on improving the system in general without any

[2] As Ritche and Thompson (creators of Unix Operating system) stated "In the context of UNIX operating system "Free text" is more like "Free" in Free Speech" rather than in "Free beer" [9]

[3] "Best weapon is one which need not be fired [10]

[4] AWK programming language was named after the first letters of names of its creators (Aho, Weinberger, and Kernighan of Bell Labs [12]

[5] The goal of the BBP is to obtain greater efficiency and productivity in defense spending by delivering the warfighting capabilities needed for the money available [13]

[6] Oral history of Bell Labs

specific specialty. Those people were called Systems Engineers (people who worked, in general, on the Bell System without an identifiable specialty).

III. Requirements Engineering at Bell Labs

Bell Labs used 4 tiers of requirements engineering[7] for large[8] systems like its 5th Generation Electronic Switching system (5ESS) [17-19]. The development effort for 5ESS required 5,000 employees, producing 100 million lines of system source code, mostly in the C language, with another 100 million lines of header files and make files. Evolution of the system took place over 20 years. Three releases were often being developed simultaneously with each taking about three years to develop. There was a very clear separation of roles and responsibilities: Requirements Engineering, Architecture, Design, Development, Testing, Deployment, Customer support. [20-23]. The roles and responsibilities of Requirements Engineering were:

- Tier-1: People who interact with customers and generate customer requirements (Tier-1 requirements)
- Tier-2: Take tier-1 requirements as input and generate requirements for the whole system (Tier-2 requirements)
- Tier-3: Take tier-2 requirements as input and generate the requirements for each subsystem (Tier-3 requirements)
- Tier-4: Using subsystem requirements as input and generate detailed requirements for each component/sub-sub-system (Tier-4 requirements)

These Tiers are similar to the A-spec, B-spec, C-Spec, etc. of MID-STD-490 and 490A [24-25]. The requirements are prepared using in-house developed typesetting tools like roof, nroff, troff and sexist[9] [26-31].

IV. Models and Digital Engineering

The crux of Digital Engineering is the creation of computer readable models to represent all aspects of the system and to support all the activities for the design, development, manufacture, and operation of the system throughout its lifecycle. These computer models have to be based on shared data schemata so that a digital thread integrates all the diverse stakeholders involved in the acquisition of new weapon systems [32]. Some of the models enumerated are [33]:

- Specialty Engineering Models
- Management Models
- Cost Models
- Design Models
- Manufacturing Models
- Verification and Validation Models
- Digital System Models
- Product Support Models
- Acquisition Reference Models
- Government Reference Models
- Performance Models
- Reliability Models
- System Safety Models
- Cybersecurity Models
- Maintainability and Sustainability Models
- Environment Models
- Quality, Safety, and Mission Assurance Models
- Human Performance Requirements Models

V. Current State of Digital Engineering

Problems in Systems Engineering resulted in quality, schedule and cost problems in DOD systems. For example, GPS-OCX breached the Nunn-McCurdy threshold in 2016. Factors that led to the critical Nunn-McCurdy breach

[7] Ultimate purpose of any analysis or systems engineering is to generate clear cut requirements for the next phase of development, testing and qualification.

[8] At Bell Labs any system with more than 100 million of code is considered large.

[9] Sexist is a tool that parses the documents for gender specific pronouns and converts them to gender neutral pronouns. It is a mandatory step for all the documents produced

include "inadequate systems engineering at program inception, Block-0 software with high defect rates and Block-1 designs requiring significant rework." [34]

There are significant efforts to improve Systems Engineering in DOD [35-38]. The Digital Engineering initiative is one of them.

After the publication of DOD's Digital Engineering initiative and its requirement that all new acquisition programs include the language of Digital Engineering, DOD components came up with their own vision of Digital Engineering. Some of their requirements are so broad, deep, and ambitious that trying to develop these digital system models for all the systems, including legacy systems, will not only make Norm Augustine's XVI Law[10] a reality, but probably will move the target date from the predicted 2054 to 2044.

Hence, most Defense and Aerospace companies are in a quandary. Moving to wholesale Digital Engineering does not justify the ROI and mostly cannot be funded. But DOD requirements mandate Digital Engineering.

The Systems Engineers found a solution. They stretched the definition of MBSE and equated it to Digital Engineering. Hence, whenever and wherever the government requests Digital Engineering, MBSE is used as the answer (which as described later, is a collection of mostly static SysML drawings). This is similar to the Flowchart Curse Boorks described in his "Mythical Man Month[11]. This is also similar to the switching from 4G to LTE by telecom companies.[12]

During this process, the spirit of Digital Engineering is lost. Only Systems Engineers are concerned with it. Even though most major aerospace companies have been using models encompassing all the domains enumerated in DOD's vision, there is no coordination among these development efforts and the synergies are not optimized. They were not considered as a part of Digital Engineering.

VI. Modeling & Simulation (M&S)

Before the dawn of Digital Engineering (and even MBSE), the most popular use of models was in the M&S community. Models are used to produce real and tangible results that cannot be obtained by modifying and experimenting on an existing system. They can also be used to obtain such results at a lower cost and in a more repeatable way. M&S played a major role in the development and analysis of military systems. These models are developed at different levels [13] (commonly known as the M&S pyramid [41]) with varying levels of fidelity and sophistication.

Highly complex systems-of-systems are analyzed by distributed simulation using independently developed models federated using High Level Architecture (HLA) and Distributed Interactive Simulation (DIS) [42-45]. M&S is mostly the domain of designers and operational analysts.

VII. Model-Based Systems Engineering (MBSE)

Model-Based Systems Engineering (MBSE) is a subset of Digital Engineering. MBSE supports the systems engineering activities of requirements, architecture, design, verification, and validation. These models have to be connected to the physics-based models used by other Engineering disciplines such as Mechanical and Electrical Engineering to be useful. One challenge remaining for Digital Engineering is the integration of MBSE with physics-based models [32].

Even though the term "MBSE" first appeared in the book "Model-Based Systems Engineering" in 1993 [46], it came into prominence during the early 2000s when it became associated with Systems Modeling Language (SysML) and was championed by INCOSE. Due to the composition of INCOSE membership (mostly Systems Engineers) and the prevailing problems with the document-based systems engineering methods used at that time, Systems Engineers embraced it. Tool vendors pushed it as the "Silver Bullet" for improving engineering efficiency and lowering development costs. Technology, in particular in software and systems domains, is no stranger to Silver Bullets. These are discussed in more detail later.

---

[10] "In the year 2054, the entire defense budget will purchase just one aircraft. This aircraft will have to be shared by the Air Force and Navy 3-1/2 days each per week except for leap year, when it will be made available to the Marines for the extra day"[39]

[11] The flowchart is a most thoroughly oversold piece of program documentation. Many programs don't need flowcharts at all; few programs need more than a one-page flowchart"[40]

[12] The joke in Bell Labs was "AT&T flipped a switch and converted their complete network to LTE. Interestingly the flipped switch was not in their Engineering department but in their marketing department"

[13] Campaign, Warfare Mission Area, Force or Force Level, Platform Level, System/Subsystem Level/Engineering and Physical Level

Models are not new concepts. Even before the advent of MBSE, there were Model Based Engineering (MBE), Model-Driven Engineering (MDE), and Model-Driven Development (MDD)

VIII. Origin of the Word "Model"

The term originally denoted the plans of a building in late 16th-century English and derived via French and Italian ultimately from Latin (from the word "modulus" meaning "measure"). Models can be divided into physical models (e.g. a model plane) and abstract models (e.g., mathematical expressions describing behavioral patterns) [47].

IX. What is a Model in MBSE?

The Systems Engineering community stretched the definition of Model. With the advent of tools like Cameo and Rhapsody, creating static pictures/drawings became very popular. They started defining the SysML artifacts as "Models" although they are mostly static pictures, not executable models, and cannot answer difficult questions like the ones shown later (Aegis Ballistic Missile Defense System, Solar power plant, and others).

Some MBSE practitioners started calling anything and everything a "Model." For example, system requirements uploaded to a DOORS [48] database became a "Model." Some even started calling the original text document a "Descriptive Model." Such an informal and permissive definition brings to mind the archer who shoots an arrow first and draws a target around it wherever it lands [49]. Working with these imprecise definitions is like performing surgery with blunt tools - everyone suffers. There are already too many "CAC cards"[14] and "A2/Ads"[15] in our business and they are not helping the cause. Even DODAF jumped on the Model bandwagon and started calling almost everything a Model[16] and UAF is not far behind[17]

The real Digital Engineering effort is greatly harmed by these lax definitions of "Models." When everything is a "Model," there is no need or motivation to develop real Digital Models that are useful. Moreover, developing real models that can answer difficult questions (like those shown in the examples later) needs effort, thought, and concentration in addition to resources (experts, time, and money). For example, a weather map displayed on the TV during weather bulletins is not a "Model." The real weather model runs on high performance computers. The blueprint of a house is not a Model. The real 3-D model generated using CAD software is the real model.

X. MBSE is No Silver Bullet, and in fact "There is no Silver Bullet"

Contrary to popular belief, MBSE is no silver bullet. In fact, there are no Silver Bullets in any discipline [55]. However, from time immemorial, people have always touted silver bullets.[18] Some of them are:

- Structured Programming ("GOTO" statement considered harmful: The unbridled use of the go to statement has as an immediate consequence that it becomes terribly hard to find a meaningful set of coordinates in which to describe the process progress. ... The go to statement as it stands is just too primitive, it is too much an invitation to make a mess of one's program [57, 58])
- Structured programming with GOTO statements [59]
- Literate Programming ("Someday Pulitzer prizes will be awarded to computer programs" [60, 61])
- ALGOL programming language ("Here is a language so far ahead of its time, that it was not only an improvement on its predecessors, but also on nearly all its successors [62]." "Algol, the language for those scholars among programmers" [63])
- Simple and systematic programming languages ("These baroque monstrosities, these conglomerations of idiosyncrasies, are really unmanageable, both mechanically and mentally. I see a great future for simple and systematic languages" [64])
- Proving programs to be correct ("I have not tested this program. Only proved it to be correct." [65])
- Beauty is our Business ("When we recognize the battle against chaos, mess, and unmastered complexity as one of computing science's major callings, we must admit that "Beauty is our Business" [66-68])

[14] Common Access Card (CAC) card)[50, 51]

[15] Anti-Access/Area Denial [52]

[16] The DoDAF-described Models that are available in DoDAF V2.0. The DoDAF Models are grouped into the viewpoints.[53]

[17] UAF architecture models provide a means to develop an understanding of the complex relationships that exist between organizations, systems, and systems-of-systems and enable the analysis of these systems to ensure that they meet the expectations of the user community.[54]

[18] "The nice thing about silver bullets is that there are so many to choose from. And if you do not like any of them, just wait a year or two [56]

- Software Engineering ("Software Engineering: An Idea Whose Time Has Come and Gone?" [69], "Everything is NOT, need NOT be, and should NOT be Engineering" [70], "Software engineering should be known as 'The Doomed Discipline,' and aimed to guide people who can't program." [71])
- Object Oriented Programming (OOP) ("Everything is an Object." "An exceptionally bad idea which could only have originated in California." [72])
- CORBA (Common Object Request Broker Architecture [73])
- CMM, CMMI [19]
- Agile ("Agile is no silver bullet" [75])
- UML ("Death by UML Fever: Self-diagnosis and early treatment are crucial in the fight against UML Fever" [76])
- Flowcharts ("90% of programmers cannot write a proper program without a flowchart. But 90% of the programmers think they are in the other 10%. Hence at least 80% of programs won't work first time.", Flowchart curse" [40])
- The Waterfall [20] method of software development was an innovation when it was introduced in the 1950s for producing the programs for the Semi-Automatic Ground Environment (SAGE) system [77]. ("Have you seen a waterfall" [78])
- Artificial Intelligence (AI) ("AI had a long and glorious history. It started in 1956 and ended on Oct 24" [79], Grandmother hack of ChatGPT" [80-83])

XI. Digital Twin

A Digital Twin is a related yet distinct concept to Digital Engineering. The Digital Twin is a high-fidelity model of the system which can be used to emulate the actual system. Modeling and Simulation (M&S) is very closely related to Digital Twins [32, 84]. There is a lot of confusion in what constitutes a Digital Twin. The simplest and most practical definition is: it is a computer program which can answer the questions about the real physical system without the need to modify and experiment with the real system. More often than not, the real system is either not available, not yet built, cannot be experimented with, and prohibitively expensive in terms of cost, schedule, and complexity. The best example is the software model developed by Lawrence Livermore Labs to check the effectiveness of nuclear warhead stockpiles. Real testing of them is no longer possible due to the Comprehensive Nuclear-Test-Ban Treaty (CTBT) [85]. As general rule of thumb, unless there are equations (often complex equations) it is either not a Digital twin or not a useful twin. Some examples of Digital Twins are given below.

A. Solar Power Plant

A digital Twin of the Concentrating Solar Power Plant (CSP) was developed for the design, analysis and proposal for a 295 MW plant. Several models developed independently, which were designed for interactive use, were integrated into a Digital Twin. Some of them are: Solar Advisor Model (SAM), ThermoFlow, GateCycle, UniSim, TRNSYS, HYSYS, Sol TRACE, FLUENT, Excelergy, CSDS, and Models by EDA. Using this Digital Twin, we reduced the time taken to generate the proposal for a new solar power plant from 3 weeks to 2 hours and the cost from $1.7 million to $20,000. This was also used to invent new methods and algorithms that can save hundreds of millions of dollars in capital costs for solar power plants and even increase the energy output of existing power plants by at least 8% while honoring the power purchase agreements (PPA) [86-89]. Some of the questions this Digital Twin answered are:

- How much will it cost to build a 295 MW (or any specified size) Solar Power Plant in Arizona (or any specified location (e.g., Ascension Island in the Atlantic Ocean)? [90]
- What would be the LCOE (Levelized Cost of Energy)?
- What would be the LCOE without incentives from the Federal Government?
- Can we improve the ROI of any power plant by at least 8% by just changing the software?
- Can a new Energy Dispatch Algorithm take into account the PPA (Power Purchase Agreement) and TOD (Time of the day) pricing?
- Do we really need NREL (National Renewable Energy Labs) recommended 6 hours of storage (of molten salt) which costs $80 million/hour?

[19] Currently we do not hear much about CMM or CMMI. My favorite is the CIMM Capability Immaturity Model which defines 4 more levels in addition to the traditional level-1 to Level-5. They are: 0 : Negligent, -1 : Obstructive, -2 : Contemptuous, and -3 : Undermining[74]

[20] When two much maligned waterfalls are put back to back, it becomes a much respected "V" of systems engineering

- How can we reduce the capital cost of a 295 MW power plant from $1.8 billion by $200 to 300 million?

B. Solar System Test and Engineering Site (SolSTES)

The Digital Twin of SolSTES predicted the performance of SolSTES (before it was built) to an accuracy of less than 0.25% (e.g., The temperature of the oil heated by the Concentrating Solar Panels was predicted to $\pm 1^0$C [91]).

C. Aegis Ballistic Missile System [92-94]

The Digital Twin of Aegis was used to answer the following questions:

- Can the Aegis BMD ship shoot down a satellite instead of a regular missile? If not, what changes are needed for Radar, Discrimination, SM-3 Missile, Kill Vehicle, and Terminal Guidance?
- If one ship with an Aegis Ballistic Missile defense system can handle "n" threats, how many threats can 2 ships can handle (2n, <2n, or > 2n)?
- How does the total number of threats handled change if CEC (Cooperative Engagement Capability) is used?
- How do these numbers change for "Shoot," "Shoot-Look-Shoot," "Shoot-Shoot-Look-Shoot," and "Shoot-Shoot-Look-Shoot-Shoot"?
- Sensitivity analysis for QOS (Quality of Service)
- How the P_{ssk} (Probability of Single Shot Kill) change if SPY-1 Radar is replaced with MADE (Air and Missile Defense Radar)?
- How does the Probability of Raid Annihilation (P_{RA}) change with 1 ship, 2 ships, .n ships?

D. Information Processing System

Developed, using Python Language and Spyder IDE. It was used to optimize the integration of two legacy systems. It was also used for trade studies between "Data Driven" and "Demand Driven" approaches [95]. Some of the questions this Digital Twin answered were:

- Given the choice between a single machine with speed s or n machines each with speed s/n, which should we choose for upgrading the communication system?
- If both the arrival rate and service rate double, will the mean response time stay the same for the system?
- Should the Enterprise Database system really aim to balance load, or is this a convenient myth?
- If a scheduling policy favors one set of jobs, does it necessarily hurt some other jobs or are these "conservation laws" being misinterpreted?
- Do greedy, shortest delay, routing strategies make sense in a server farm or is what is good for the individual may be disastrous for the system as a whole?
- How should one trade energy and delay in designing a computer system for enterprise?
- If 12 servers are needed to meet delay guarantees when the arrival rate is 9 jobs/s, we will need 12,000 servers when the arrival rate is 9,000 jobs/s?

XII. Multi-Mission Analysis Tool (MMAT) for DD(X) Developed Entirely Using Only UML

Throughout my professional life, I have seen only one example [96] where the requirements taken from the documents were used to create the UML artifacts using Rational Rose [97], which were then enhanced using Rational Rose Real-time [98], which were used to generate the executable code. The lessons learned were:

- The effort required is not less. In fact, in some cases, it requires much more effort than normal methods.
- The code generated is not human readable
- In spite of recent advantages in compiler optimizations, the efficiency of generated code is suboptimal both in terms of space and time [99]

XIII. Best Practices for Developing Digital Twins

- Don't get hung up on the tools (in particular vendor promoted Silver Bullets")
- Do it incrementally (optimum fidelity to answer the questions)
- Do not generate dead-end artifacts (e.g., static pictures which cannot be used in the next phase of development (e.g. Automatic code generation, Automatic text script generation))
- Follow Unix culture. The last bullet is the most important. If we can eliminate any step by using technological advances (pillar-3 of the Digital Engineering vision), eliminate it (e.g., if ChatGPT can generate the code from the documents (or even emails, voice memos, etc.), then use it).
- Keep "Model Developer's Dilemma" in mind [100]

XIV. Improving Efficiency and Productivity of Developing Digital Twins Using UNIX Culture

Bell Labs follows UNIX culture [101, 102]. It is summed up in the following steps:

- First, let the machine do the work (use grep, wc, awk.)
- Second, let other people do the work (use programs that already exist as building blocks in your programs, with the shell and the programmable filters that glue them together)
- Third, do the job in stages (build the simplest things that will be useful)
- Fourth, build the tools (write programs that mesh with existing environments, enhancing it rather than merely adding to it)
- Lastly, "The most efficient way of doing any job is: showing that it need not be done"

XV. Preventing Digital Twins from Becoming Evil Twins

Digital Twins are very useful in finding the answers which could not be obtained easily using the real systems (Physical Twins). However, after some initial successes, people tend to trust them blindly and make critical decisions based on their results. This is where the Digital Twin becomes an evil twin [103,104]. Some of the steps to guard against this are:

- Be skeptical[21] about the results
- Remember the worlds of Aristotle[22]
- Try to cross-check the results from a system of lesser fidelity (if available)
- If possible, use data obtained from the real system to periodically update the underlying model
- If the system is really critical (like validating the effectiveness of nuclear weapons), use the method used for Space Shuttle Avionics System [105]

XVI. Incorporating Technological Innovations

This is the 3rd and probably most important initiative in the Digital Engineering vision. A number of disruptive technological innovations are coming onto the scene. They will revolutionize the way systems are designed and developed. For example, the field of Computer Science is headed for a major upheaval with the rise of large AI models such as ChatGPT that are capable of performing general-purpose reasoning and problem solving [80-82]. We are headed for a future in which it will no longer be necessary to write computer programs. These technologies can fuse Unstructured, Uncertain, Incomplete, Imprecise, and even Contradictory (UUIIC) information from all sources (documents, audio, video, web, emails, hidden web, etc.) and generate the final systems probably using 3-D printing. This will make the current processes obsolete, including Systems Engineering [106].

XVII. Conclusion

Digital Engineering is a misnomer and there is certain poverty of imagination in choosing that name. As presented in this paper, it is not specific to Engineering (much less Systems Engineering) but encompasses all the specialties involved. Engineering is a small part of it and System Engineering is a small part of Engineering. We need to come up with a new name for it which all disciplines can relate to. At present, only Systems Engineers are paying any attention to it and unfortunately, they are equating it to MBSE. This way, we will fail to realize its full potential. We should refrain from calling everything a "Model." MBSE is not a silver bullet and should not be equated to Digital Engineering as is currently being done in government, industry and academia. The main motivation or justification for MBSE is "Document based process is bad." However, I am yet to see a program where the original requirements are in any form other than documents. If AI can parse those documents and generate the requirements or even final executable software, we do not need Systems Engineering, MBSE or even Software Engineering. We must not flip a switch, like AT&T did in its marketing department, and make MBSE is Digital Engineering. There is no Royal Road to Digital Engineering.

Acknowledgments

I want to acknowledge the inspiration derived from Alex Bell's "Death by UML fever," Tom Demarco's "Software engineering an idea whose time has come and gone" and Dijkstra's "Software engineering should be known as 'The

[21] As they say in Instrumentation, good measurement always needs someone who is spectacle about it

[22] "It is the mark of an instructed mind, to remain satisfied with the precision allowed by nature to that subject, and not to ask for exactness when an approximation of truth alone is possible"--Aristotle

Doomed Discipline' and should have its charter 'How to program if you cannot.'" I also want to express my sincere thanks to Mr. Norm Augustine and Dr. Paul Kaminski who reviewed the draft and provided valuable comments.

References

[1] Hofbauer, J., Sanders, G., Ellman, J., and Morrow, D., Center for Strategic and International Studies, Cost and Time Overruns for Major Defense Acquisition Programs: An Annotated Brief, Center for Strategic and International Studies,

[2] Berteau, D., "Cost and time overruns in Major Defense Acquisition Programs (MDAPs)", NPS 8th Annual Acquisition Research Symposium May 11-12, 2010

[3] Deloitte Study: Cost Overruns Persist in Major Defense Programs, "The combined cost overrun for Major Defense Acquisition Program (MDAP) portfolio programs in 2015 was $468 billion, up from $295 billion in 2008" https://www.prnewswire.com/news-releases/deloitte-study-cost-overruns-persist-in-major-defense-programs-300349671.html

[4] AN/FPQ-16 PARCS, Perimeter Acquisition Radar Attack Characterization System (PARCS), https://en.wikipedia.org/wiki/AN/FPQ-16_PARCS

[5] Upgraded Early Warning Radars (UEWR), Missile Defense Advocacy Agency (MDAA), https://missiledefenseadvocacy.org/defense-systems/upgraded-early-warning-radars-uewr/

[6] Boeing B-52 Stratofortress fact sheet, USAF, https://www.af.mil/About-Us/Fact-Sheets/Display/Article/104465/b-52h-stratofortress/

[7] Anti-Ballistic Missile (ABM) Treaty, https://en.wikipedia.org/wiki/Anti-Ballistic_Missile_Treaty

[8] DOD Digital Engineering Strategy, 2018, https://ac.cto.mil/digital_engineering/

[9] Ritche, D., and Thompson, K., "UNIX operating System),"Bell Labs Technical Journal, 1972

[10] IEEE Spectrum, "Technology in War and Peace", October 1982

[11] DAU, Defense Acquisition Workforce Improvement Act (DAWIA), https://en.wikipedia.org/wiki/Defense_Acquisition_Workforce_Improvement_Act

[12] Aho, A., Weinberger, P., and Kernighan, B., "The AWK programming Language", 1970, https://en.wikipedia.org/wiki/AWK

[13] "Better Buying Power (BBP)", https://acqnotes.com/acqnote/tools/better-buying-power-2

[14] Schlager, J., "Systems engineering: key to modern development". IRE Transactions, July 1956. EM-3 (3): 64–66. doi:10.1109/IRET-EM.1956.5007383. S2CID 51635376.

[15] Gertner, J., The Idea Factory: Bell Labs and the Great Age of American Innovation, Penguin Books,– February 26, 2013, ISBN-978-1-59420-328-2

[16] Endlich. L., "Optical Illusions: Lucent and the Crash of Telecom", Simon & Schuster, – October 7, 2004, ISBN-0-7432-2667-4

[17] Forbes, "Why Bell Labs Was So Important To Innovation In The 20th Century", Jul 19, 2017, https://www.forbes.com/sites/quora/2017/07/19/why-bell-labs-was-so-important-to-innovation-in-the-20th-century/?sh=78f585817015

[18] 5ESS Switching System, https://en.wikipedia.org/wiki/5ESS_Switching_System

[19] AT&T Bell Laboratories, Best Current Practices (BCP), 1980

[20] Howard, W.,(ed) "The 5ESS Switching System," vol. 64, AT&T Technical Journal, July-August 1985, Special Issue on the 5ESS Switch.

[21] Zave, P., and Jackson, M., "Four Dark Corners of Requirements Engineering" AT&T Research, ACM Transactions on Software Engineering and Methodology, Vol. 6, No. 1, January 1997.

[22] Mannepalli, R., "How can we achieve Level-3 on SEI's CMM, with some simple process Improvements that can save $640 million/year and add $0.32 to Lucent EPS, Lucent Technologies, 1999, (unpublished)

[23] Mannepalli, R., "Roles and Responsibilities: Changes to Systems Engineering, Development and Testing", Lucent Technologies, 2001 (unpublished)

[24] MIL-STD-490, 3-Oct-1968

[25] MIL-STD-490A, 4-Jun-1985

[26] roff, Bell Labs developed tool for typesetting, https://en.wikipedia.org/wiki/Roff_(software)

[27] nroff, Bell Labs developed tool for typesetting (new roff), https://en.wikipedia.org/wiki/Nroff

[28] troff, Bell Labs developed tool for typesetting (newer nroff), https://en.wikipedia.org/wiki/Troff

[29] sexist, Bell Labs developed tool to identify gender specific pronouns and convert them to gender neutral

[30] Bell System Technical Journal (BSTJ) i , https://en.wikipedia.org/wiki/Bell_System

[31] Bell Labs Technical Journal, https://en.wikipedia.org/wiki/Bell_Labs_Technical_Journal

[32] INCOSE-SEBOK (SE Book of Knowledge), https://sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK)

[33] Digital System Model, DI-SESS-82364, 2022-02-01, https://publishers.standardstech.com/stgnet

[34] Gruss, M., "Air Force declares Nunn-McCurdy breach on GPS ground system", Space News, https://spacenews.com/air-force-declares-nunn-mccurdy-breach-on-gps-ground-system/

[35] Griffin, M. D., "How do we fix System Engineering", 2010, https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwiN6bmis7n-

AhXLFDQIHTWhC6EQFnoECBEQAQ&url=https%3A%2F%2Fwww.nasa.gov%2Fsites%2Fdefault%2Ffiles%2Fatoms%2Ffiles%2F3_griffin_how_do_we_fix_systems_engineering.pdf&usg=AOvVaw0EOxlyvnQOcWB3RmbkX3g9
[36] Griffin, M. D., "System Engineering", https://www.youtube.com/watch?v=8l4J38D7VaE
[37] DAU, "Art & Science" of systems engineering: ENG 301 Leadership in Engineering Defense Systems, Student Book Version 1.3 https://icatalog.dau.edu/onlinecatalog/courses1.aspx?crs_id=1996
[38] Defense Acquisition University (DAU), Systems Engineering Guidebook , Feb 2022, https://www.dau.edu/cop/dmsms/_layouts/15/WopiFrame.aspx?sourcedoc=/cop/dmsms/DAU%20Sponsored%20Documents/Systems-Eng-Guidebook_Feb2022-Cleared.pdf&action=default&DefaultItemOpen=1
[39] Augustine's Law XVI (on exponential growth of aircraft cost) https://en.wikipedia.org/wiki/Augustine%27s_laws
[40] Brooks, F. P., "The Flow-Chart Curse", Mythical Man Month, Essays on Software Engineering, Page: 167
[41] Naval Simulation System-21 (NSS-21), Use of Modeling and Simulation (M&S) in Support of Joint Command and Control Experimentation: Naval Simulation System (NSS) Support to Fleet Battle Experiments, https://apps.dtic.mil/sti/citations/ADA461113
[42] IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)— Framework and Rules. IEEE Computer Society. 18 August 2010. ISBN 978-0-7381-6251-5.
[43] IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)— Federate Interface Specification. IEEE Computer Society. 18 August 2010. ISBN 978-0-7381-6247-8.
[44] IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)— Object Model Template (OMT) Specification. IEEE Computer Society. 18 August 2010. ISBN 978-0-7381-6249-2.
[45] IEEE Standard 1278.1-2012, for Distributed Interactive Simulation (DIS) (for conducting real-time platform-level wargaming) - Application Protocols. IEEE. doi:10.1109/IEEESTD.2012.6387564. ISBN 978-0-7381-7310-8.
[46] Wayne, W. A., "Model-based systems engineering: an introduction to the mathematical theory of discrete systems and to the tricotyledon theory of system design", 1993
[47] Model, https://en.wikipedia.org/wiki/Model
[48] IBM, Dynamic Object Oriented Requirements System (DOORS), https://www.ibm.com/products/requirements-management?mhsrc=ibmsearch_a&mhq=doors
[49] Mannepalli, R., "Rao's Method and Chord-Midpoint Method for Burning Time Computation of Solid Rocket Motors, AIAA Publishing and in the AIAA Propulsion and Energy Forum- August 9-11, 2021. DOI: 10.2514/6.2021-3700
[50] Common Access Card, Wikipedia, https://en.wikipedia.org/wiki/Common_Access_Card
[51] Mannepalli, R., There are too many CAC cards and A2/ADs in our field, Raytheon, 2017 (unpublished)
[52] Anti-access/area denial, https://en.wikipedia.org/wiki/Anti-access/area_denial
[53] DOD Architecture Framework Version 2.02 https://dodcio.defense.gov/Library/DoD-Architecture-Framework/dodaf20_models/
[54] Unified Architecture Framework (UAF), https://www.omg.org/uaf/
[55] Brooks, F. P., "No Silver Bullet -- Essence and Accident in Software Engineering", https://en.wikipedia.org/wiki/No_Silver_Bullet
[56] Tanenbaum, A., and David Wetherall , D., "Computer Networks (5th ed), Pearson, ISBN-13 978-0132126953
[57] Structured programming, https://en.wikipedia.org/wiki/Structured_programming
[58] Dijkstra, E. W "Go To Statement Considered Harmful" Communications of the ACM. 11 (3): 147–148. doi:10.1145/362929.362947. S2CID 17469809, (March 1968).
[59] Knuth, D. E., Structured Programming with go to Statements, ACM Computing Surveys Volume 6 Issue 4 Dec. 1974 pp 261–301 https://doi.org/10.1145/356635.356640
[60] Knuth, D. E., Literate Programming, Stanford, California: Center for the Study of Language and Information, 1992), xvi+368pp. (CSLI Lecture Notes, no. 27.) ISBN 0-937073-80-6, https://www-cs-faculty.stanford.edu/~knuth/lp.html
[61] Literate Programming, https://en.wikipedia.org/wiki/Literate_programming
[62] Hoare, C. A. R., "Hints on Programming Language Design" (December 1973). p.27.http://www.eecs.umich.edu/~bchandra/courses/papers/Hoare_Hints.pdf
[63] Mannepalli, R., "ALGOL, the language for those scholars among programmers', Indian Space Research Organization, 1993 (unpublished)
[64] Dijkstra, E. W., Turing Award Lecture, 1972
[65] Knuth, D. E., I have not tested this program. Only proved. it to be correct
[66] Dijkstra, E. W., Quotes, When we recognize the battle against chaos, mess, and unmastered complexity as one of computing science's major callings, we must admit that "Beauty is our Business
[67] Mannepalli, R., Beauty is our Business: Elegance in Software Development, A tribute to Dijkstra", Bell Labs Lecture, 2003 (unpublished)
[68] Feijen, W. H. J., van Gasteren, A. J. M., Gries, D., and Misra, J., (ed) " Beauty Is Our Business: A Birthday Salute to Edsger W. Dijkstra on his 60th birthday", Springer Verlag, 1990, ISBN-0-387-972999-4
[69] Demarco, T., "Software Engineering: An idea whose time has come and gone", IEEE Software, Volume: 26, Issue: 4, July-Aug. 2009.
[70] Mannepalli, R., "Everything is NOT, need NOT be, and should NOT be Engineering" (in response to 'Software Engineering is Engineering' in ACM Communications), Lockheed Martin, 2012 (unpublished)

[71] Dijkstra, E. W., Quotes, Software engineering should be known as "The Doomed Discipline", and aimed to guide people who can't program.
[72] Dijkstra, E. W., Quotes, "Object-oriented programming is an exceptionally bad idea which could only have originated in California."
[73] Common Object Request Broker Architecture (CORBA), https://en.wikipedia.org/wiki/Common_Object_Request_Broker_Architecture
[74] "Capability Immaturity Model", https://en.wikipedia.org/wiki/Capability_Immaturity_Model
[75] Mannepalli, R., "Agile is no Silver bullet", Leidos, 2022 (unpublished)
[76] Bell, A., "Death by UML Fever: Self-diagnosis and early treatment are crucial in the fight against UML Fever." ACM Queue, March 2004
[77] Bennington, H. D., "Production of Large Computer Programs for the Semi-Automatic Ground Environment (SAGE) system," Office of Naval Research in June 1956.
[78] Mannepalli, R., "Have you seen a waterfall?", Lockheed Martin, 2007 (unpublished)
[79] Mannepalli, R., Artificial intelligence (AI) had a long and glorious history. It started in 1956 and ended on Oct 24, 2011' (in response to 'AI: Past and future' in ACM Communications), Lockheed Martin, 2012 (unpublished)
[80] ChatGPT, https://en.wikipedia.org/wiki/ChatGPT
[81] OpenAI, https://en.wikipedia.org/wiki/OpenAI
[82] Welsh, M., "Large Language Models and the End of Programming", ACM Tech Talk, May 9, 2023
[83] Brinkmann, M., "Here is how you talk with an unrestricted version of ChatGPT (Grandmother Hack)", https://www.ghacks.net/2023/04/24/here-is-how-you-talk-with-an-unrestricted-version-of-chatgpt/
[84] AIAA, Digital Twin: Reference Model, Realizations & Recommendations. An AIAA, AIA, and NAFEMS Implementation Paper January 2023
[85] Comprehensive Nuclear-Test-Ban Treaty, https://en.wikipedia.org/wiki/Comprehensive_Nuclear-Test-Ban_Treaty
[86] Mannepalli, R., and Albertelli, P., "Two-step, Two-Pass Method' and Apparatus for Dispatching Thermal Energy in Concentrating Solar Power Plants to get Maximum Revenue and Return on Investment," US-8903561, US-8095245-B1
[87] Mannepalli, R., and Albertelli, P., New method and device to optimize the dispatch of solar thermal energy - in capacity constrained Thermal Energy Storage (TES) systems - which can be used to design optimum Concentrating Solar Power (CSP) plants that could save hundreds of millions of dollars in capital costs, 2010
[88] Mannepalli, R., and Albertelli, P., "Renewable Energy Solar Thermal Objective Resource Modeling (RESTORM), 2009, Lockheed Martin (unpublished)
[89] Mannepalli, R., and Yeh, L., "New Method to Simplify the Operations and to Improve the Productivity of Modeling and Simulation of Solar Power Plants" US-8373291-B1
[90] Mannepalli, R., Designing a Renewable Energy solution for USAF Space Fence Radars and supporting facilities on Ascension Island in the Atlantic Ocean (Optimum combination of Photovoltaic, Concentrating Solar Power, Wind, Geothermal, Ocean Thermal (OTEC) and Nuclear)", Lockheed Martin, 2010 (unpublished)
[91] Mannepalli, R., "Solar System Test and Engineering Site (SolSTES) Model for prediction, testing and validation", 2010, Lockheed Martin (unpublished)
[92] Aegis Missile Defense system, MDA", ,https://www.mda.mil/system/aegis_bmd.html
[93] USS John Finn Takes Out ICBM in Test off Hawaii, MDA, https://www.youtube.com/watch?v=PLqZY-CfFwc
[94] CEC - Cooperative Engagement Capability, US Navy, https://www.navy.mil/Resources/Fact-Files/Display-FactFiles/Article/2166802/cec-cooperative-engagement-capability/
[95] Mannepalli, R., "Rao's rate for CCIC2S, 2022, Leidos (unpublished)
[96] Mannepalli, R., Yeh. L, and Cho, J., "Multi-Mission Analysis Tool (MMAT) for DD(X)", 2006, Lockheed Martin (unpublished)
[97] Rational Rose, https://www.ibm.com/support/pages/ibm-rational-rose-enterprise-7004-ifix001
[98] Rational Rose Real-time, https://www.ibm.com/docs/en/rtr/8.0.1?topic=SSSHUF_8.0.1/com.ibm.rational.testrt.studio.doc/topics/troseworking.html
[99] Aho. A., Monia, M. S., Sethi, R., and Ullman, J., "Compilers: Principles, Techniques, and Tools 2nd Edition, Addison Wesley, 2011, ISBN-0-321-48681-1, https://www.amazon.com/Compilers-Principles-Techniques-Tools-2nd/dp/0321486811
[100] Mannepalli, R.,' Model Developer's Dilemma: problem definition, proposed solution, and reference implementation', Lockheed Martin, 2006 (unpublished)
[101] Mannepalli, R., "Improving the performance of Optical Network Management Systems by following Plain Old UNIX Culture", Bell Labs, 2004 (unpublished)
[102] Mannepalli, R., "Reducing the time required for discovering an optical ring from 80 minutes to less than 2 minutes and reducing the garbage (Java) generated from 400 GB to less than 2 GB", Bell Labs, 1999 (unpublished)
[103] Mannepalli, R., "Experiences in developing light footprint, optimum fidelity, Digital Twin of a legacy system, incrementally (and preventing it from becoming an Evil-twin), International conference on Vibration Engineering, Science and Technology, INVEST-2022, Dec 9-10, 2022 (Invited paper)
[104] Linthicum, D., "When a digital twin becomes the evil twin", InfoWorld, Sep 18. 2020 https://www.infoworld.com/article/3574905/when-a-digital-twin-becomes-the-evil-twin.html
[105] Hanaway, J. F., and Moorehead, R. W., "Space Shuttle Avionics System", NASA SP-504, 1989

[106] Mannepalli, R., "Fusing, analyzing, and learning All-source, Unstructured, Uncertain, Incomplete, Imprecise, and even Contradictory (UUIIC) information,", Raytheon Technologies Machine Learning workshop, 2020 (unpublished)