Do We Always Want to Integrate Tools to Create the Digital Thread?

STEVEN H. DAM, PH.D., ESEP

OCTOBER 24, 2019

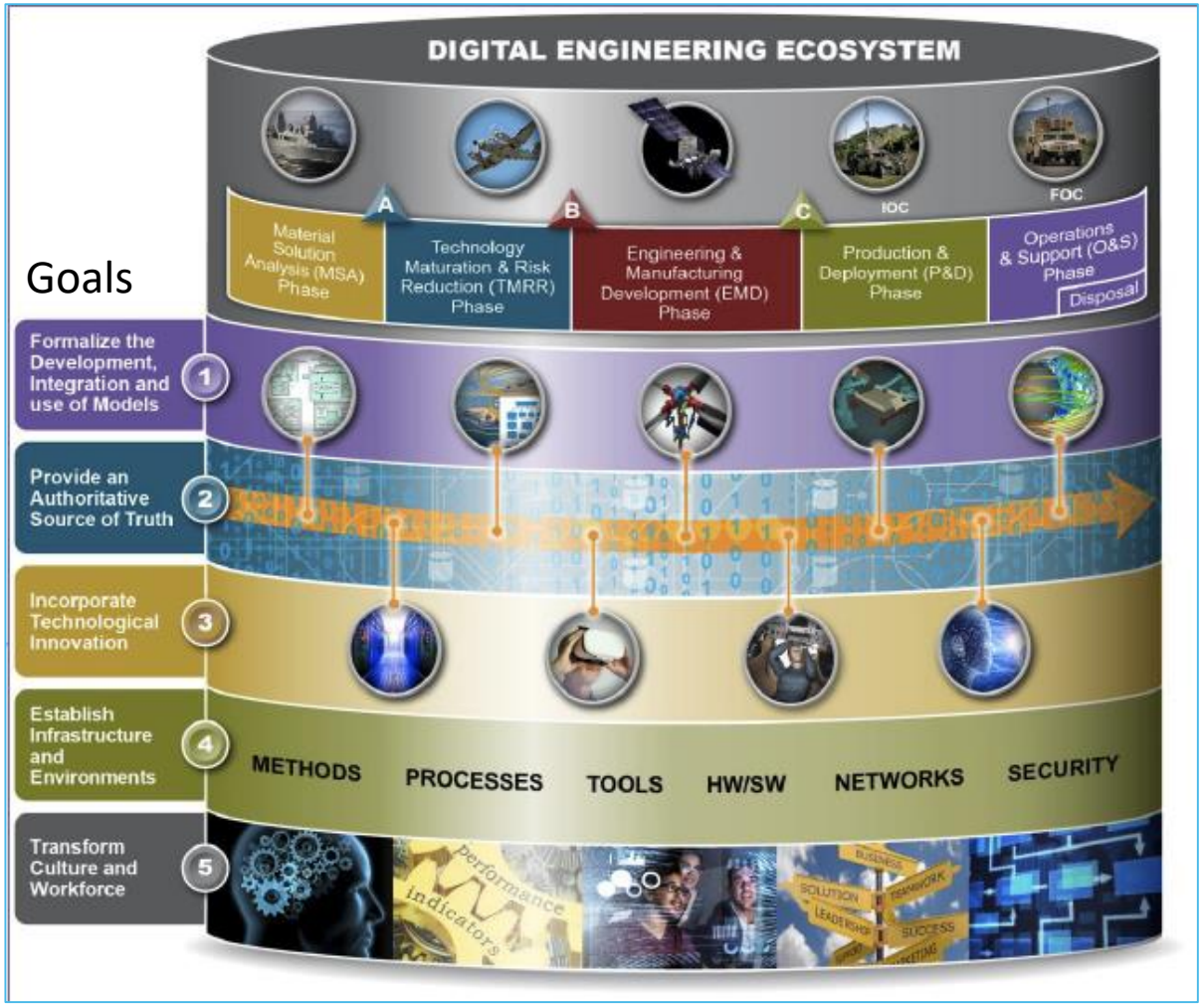
SPEC INNOVATIONS

Agenda

- What is the “Digital Thread?”
- Why Do We Want to Integrate Tools?
- How Can We Integrate Tools?
- When Should We Integrate Tools?
- When Should We Not Integrate Tools?

What is the “Digital Thread?”

- Defined in the DoD’s Digital Engineering Strategy
- Objectives:
  - Deliver high payoff solutions to the warfighter at the “speed of relevance”
  - Reform business practices
    - Digital enterprise connects people, processes, data, and capabilities
    - Improves technical, contract, and business practices through an authoritative source of truth and digital artifacts

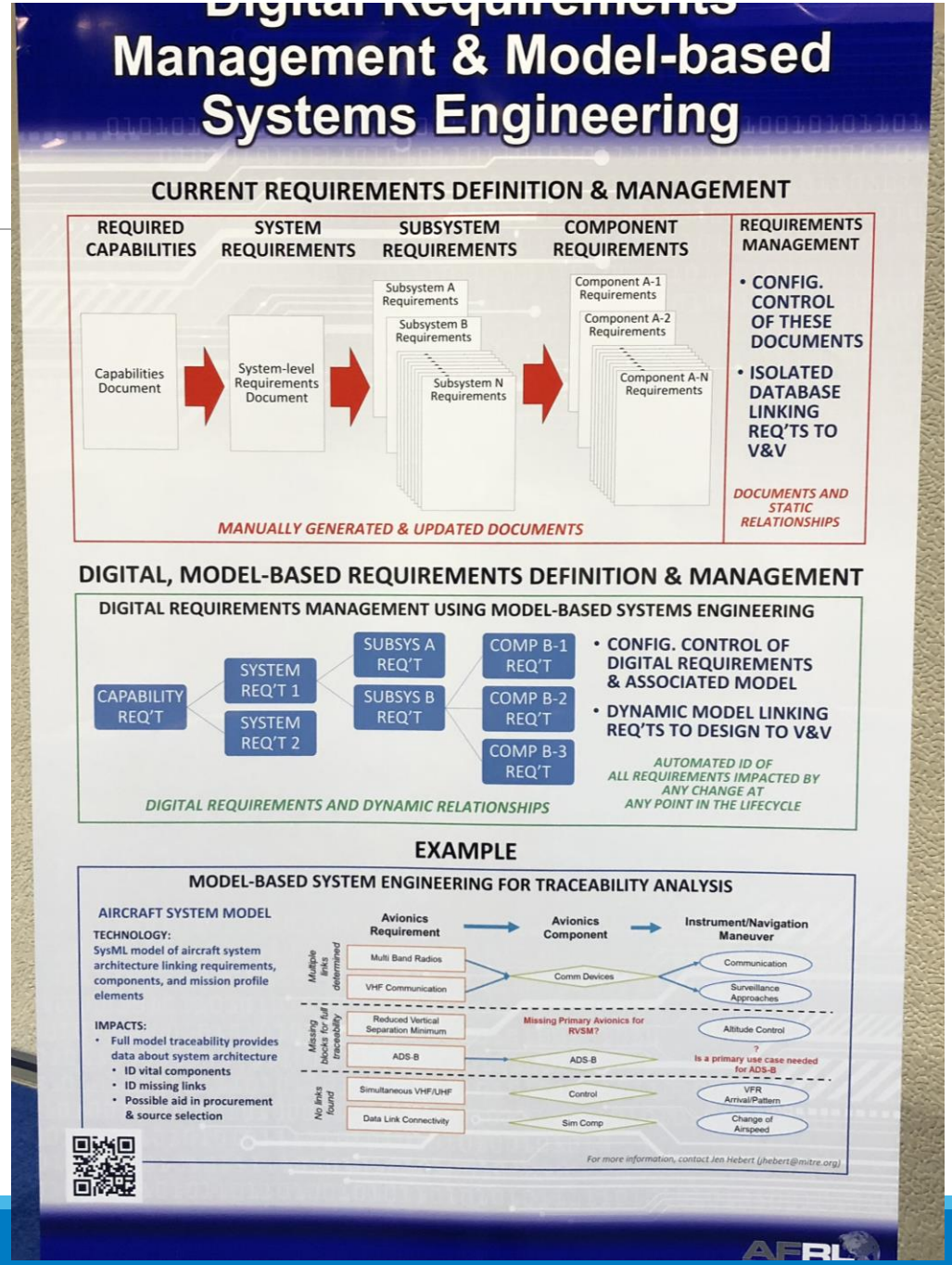


From NDIA Presentation by Philomena Zimmerman. October 24, 2018

Combines model-based techniques, digital practices, and computing infrastructure

How Does Systems Engineering "Play" in the Digital Thread?

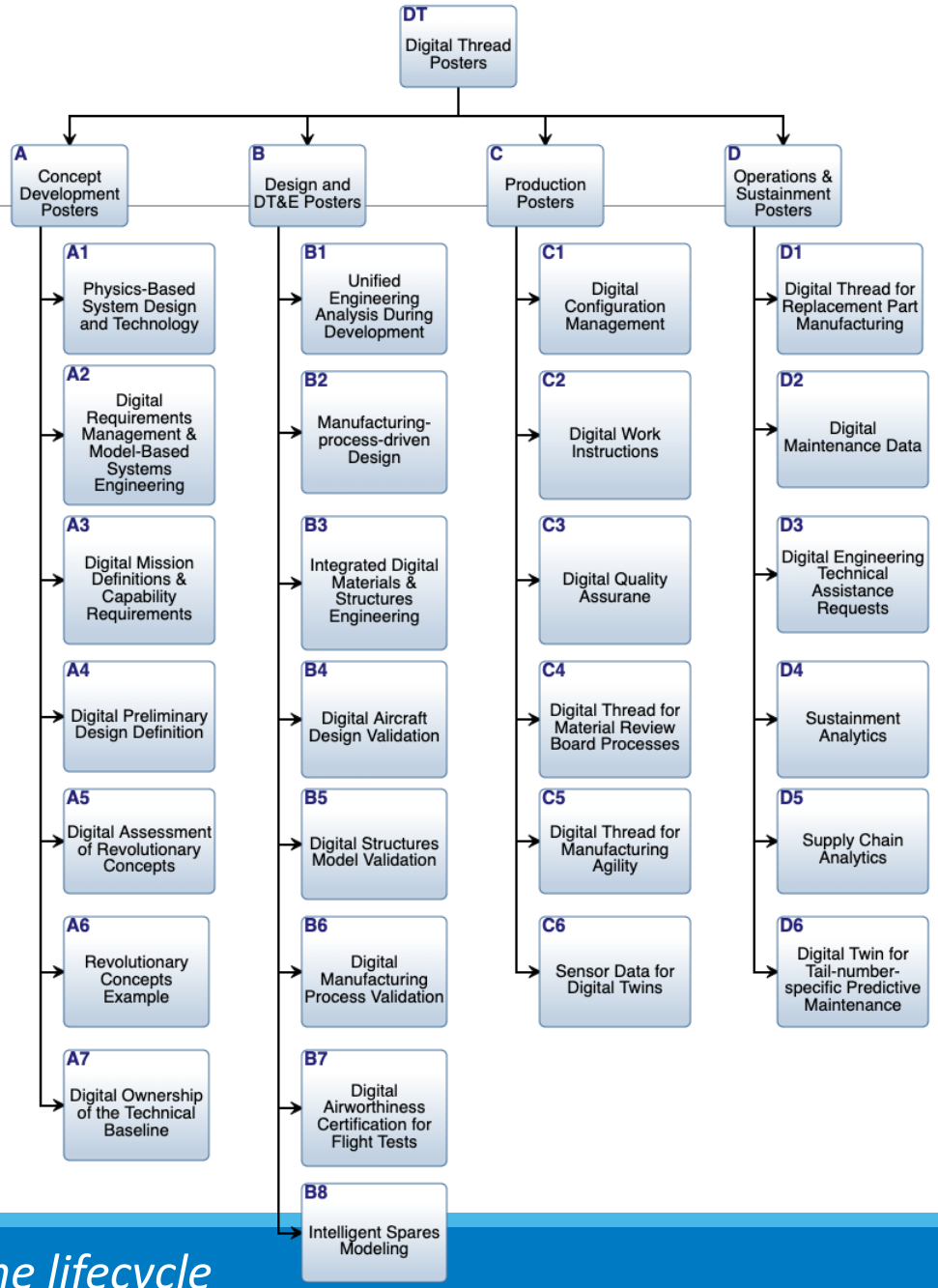
- DoD commissioned AFRL to develop a set of posters to explain the Digital Thread (DT)
- The current requirements process shows using MS Office products to create documents manually
- The Digital process shows a model-based approach
  - We have been using tools to do this for over 30 years
  - What is needed is the ability to use models in place of documents or create living documents to enable model-based reviews



But this is not the whole picture

MBSE Is Only One of Many DT Posters

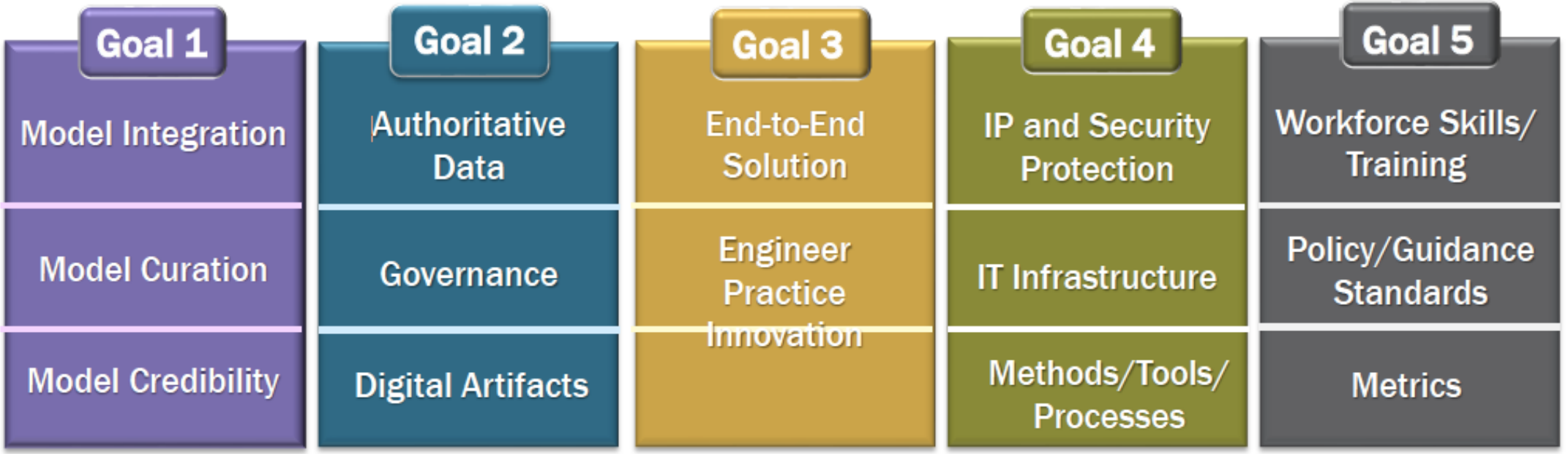
- Hierarchy shows the large number of DT posters used to describe how models support the entire lifecycle
- A complete SE approach should play a significant role in orchestrating the information developed in other models
- We clearly need to at least be able to trace design engineering results back to requirements to create a closed-loop system



We need to ensure that systems engineering is part of all phases of the lifecycle

Why Do We Want to Integrate Tools?

- Model integration is a key goal of the DES
- Tool interoperability has been a historic problem



But is it model integration we need or data interoperability?

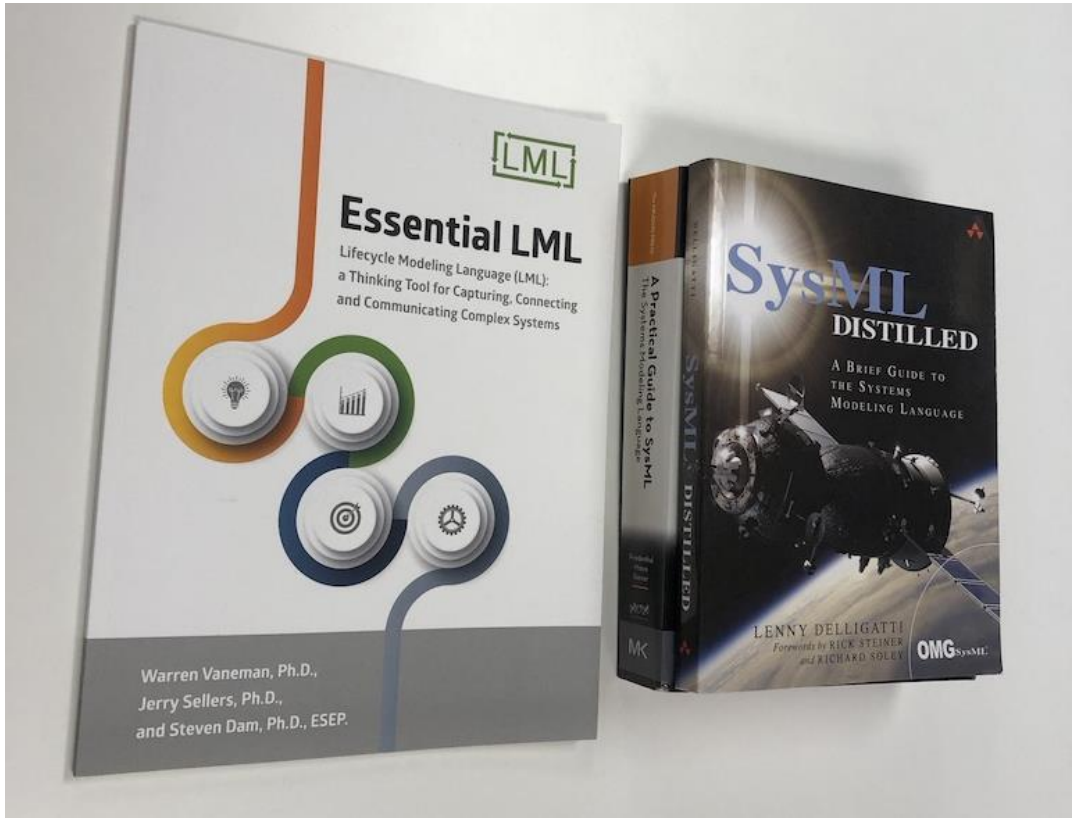
How Can We Integrate Tools?

- The fundamental problem is the lack of a common ontology for data interoperability
  - If we don't understand what the data elements represent, then we will likely not be able to create an interface
  - Without a common ontology, we would have to create one-to-one interfaces between tools – this can be done, but then you have to pick a single set of tools
- The problem has been that no common ontology had been developed
  - Lifecycle Modeling Language (LML) provides an open standard to address this issue
    - LML already provides an ontology for both DoDAF and SysML
    - It can easily be extended to support any domain
  - But as usual, very few vendors have adopted it or anything else
  - Instead, a group is trying to build an ontology for SysML, but it will likely go the way of the DoDAF Meta Model 2.0, Core Architecture Data Model, and many others before it
    - These ontologies failed because they were unnecessarily complex

The Government could adopt LML and solve this problem immediately!

Lifecycle Modeling Language (LML)

- LML was developed by a group of systems engineers who realized that SysML was not meeting the needs of the systems engineering and program management communities
- The group is led by Dr. Warren Vaneman, USN CAPT (retired) and Professor of Practice at the Naval Postgraduate School (NPS)
- LML is taught in over 200 Universities around the world, including MIT, George Mason University, Stevens Institute of Technology, West Point, NPS, Air Force Academy
- LML is easy to learn, use, and extend



LML has proven to provide a strong ontology for systems engineering ands program management

When Should We Integrate Tools?

- Lacking a common ontology, we need to carefully select the tools we want to integrate with
- We currently are selecting such a toolset for various projects
- But for the most part, we can rely on the export of a common format, such as CSV or XML or ReqIF, to move data between tools; APIs can also be used to move the data between tools
- Note that this data flows usually in one direction well, but not bi-directionally
- But we as systems engineers we mainly want to produce specifications for the design engineers and then take the results of the design engineering tools and import them into our SE database
- A completely seamless integration between our SE tools and DE tools many be a bridge too far at this time, but that should not stop us from trying to make it work

Careful integration is critical to avoiding "garbage in, garbage out"

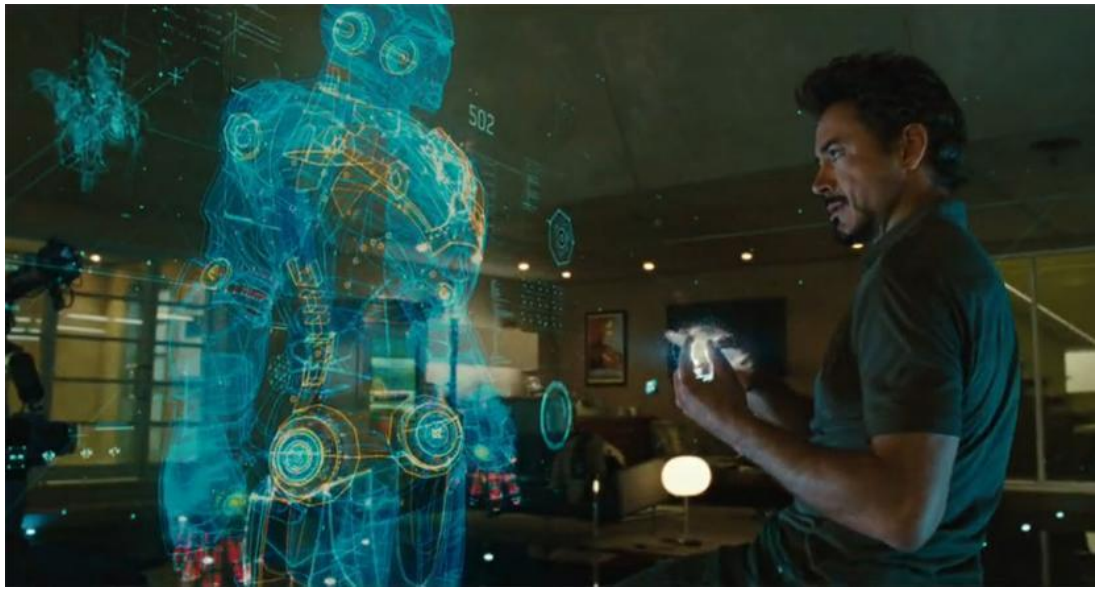
When Should We Not Integrate Tools?

- Physics-based simulations often apply only a portion of the physics needed to fully represent a system and its environment
  - Limited theoretical basis for the physics itself
  - Approximations used due to our lack of math
  - Application to areas where we don't know the exact physics
- Physics-based simulations require extensive calibration to experimental data to be useful
  - Simulation codes use "knobs" to adjust to results to match experiments
  - Users need to understand the boundary conditions where the results are valid
- Pushing poorly or non-analyzed simulations results up to the systems level may introduce new errors

In these cases we need an "air gap" or at least an "analyst-in-the-loop"

Summary

- LML provides a path to the future of systems engineering
- If digital engineering is the future, then we need to have a modern cloud-based tool that applies NLP/ML and other technologies
- We recommend including LML in any and all SE tools
- We cannot design future systems using old methods and tools
- We need to embrace these new technologies, not hamper them



We look forward to helping make "Tony Stark's lab" a reality!