Seamless Digital Engineering:
A Grand Challenge Driven by Needs

👤 James S. Wheaton, Daniel R. Herber

🏛 Colorado State University – Department of Systems Engineering
✉ james.wheaton@colostate.edu, daniel.herber@colostate.edu

January 9, 2024
AIAA SciTech Forum, Orlando, Florida, USA

Outline

1. Introduction
2. The Problem Space
3. Defining Seamless Digital Engineering
4. A Grand Challenge Driven by Needs
5. Conclusions and Future Work

①

Introduction

→ Digital Engineering (Unfortunately) Depends on Digital Systems

- Modern networked computing systems exhibit unmitigated complexity.
- The deficiencies of modern computing are a *reverse salient*[1] to achieving *#DE*.
- DARPA has recognized the reverse salient and funded related efforts in the past:
  - Clean-Slate Design of Resilient, Adaptive, Secure Hosts (CRASH)[2]
  - META-II[3] – meta-language for design of complex systems
  - Circuit Realization At Faster Timescales (CRAFT)[4]
- That "set of critical problems"[5] is enumerable and may serve as design inputs for *#MBSE* of a clean-slate computing system.

> 💬 **Quote**
>
> *The current paradigm is so thoroughly established that the only way to change is to start over again.*[6]

[1] Hughes 1993; Dedehayir and Mäkineif 2008 [2] DARPA 2010a [3] DARPA 2010b [4] DARPA 2015 [5] Hughes 1993 [6] Norman 1999

Imagine Starting From a Clean Slate

Definition (Digital Engineering Ecosystem #*DEE*)

The "enterprises' interconnected digital environments, stakeholder-networks, and semantic data that allows the exchange of digital artifacts from an authoritative source of truth to serve the stakeholder communities' interests."[1a]

[a] DAU has since updated this definition.

Definition (Digital Engineering System #*DES*)

A theoretical engineered system-of-systems that provides the networked computing resources and human-computer interfaces for an enterprise to accomplish all digital engineering activities for successful realization of engineered systems.

Note: See the paper for definitions of tool, technology, engineering, systems engineering, model-based systems engineering, digital engineering, *and* authoritative source of truth #*ASoT*.

[1] Office of the Deputy Assistant Secretary of Defense for Systems Engineering 2023

→ Related Work

- "Seamless digital engineering" previously lacked definition, but correctly identified knowledge integration as a key component.[1]
- Seamless integration is the "deep, coherent, and comprehensive integration of models and tools".[2]
- Seamless model-driven systems engineering has three ingredients:[3]
  1. A modeling theory that serves as the basis of the *#ASoT* semantic domain
  2. A complete architectural model that describes product and process
  3. "A manner to build tools that conform to the modeling theory and allow the authoring of the product model"[4]

Quote

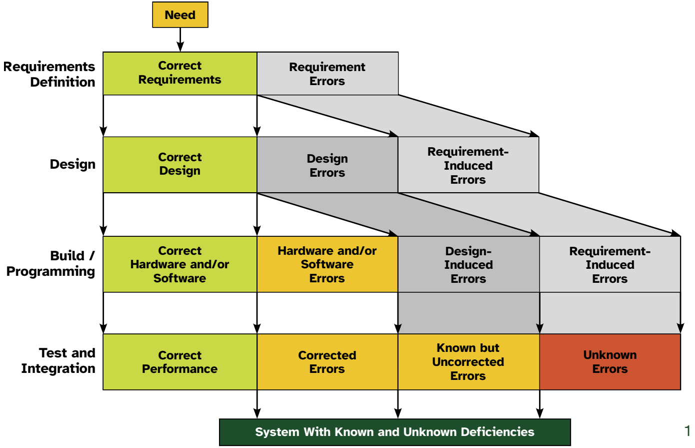
> *Those who require an integrated development environment—mainly the tool users—do not feel responsible for driving the integration. Those who seem to be responsible about tooling—mainly the tool vendors—do not profit.*[5]

[1] Perzylo et al. 2020 [2] Broy et al. 2010 [3] Broy 2009; Broy et al. 2010; Broy 2020 [4] Broy et al. 2010 [5] Broy et al. 2010

②

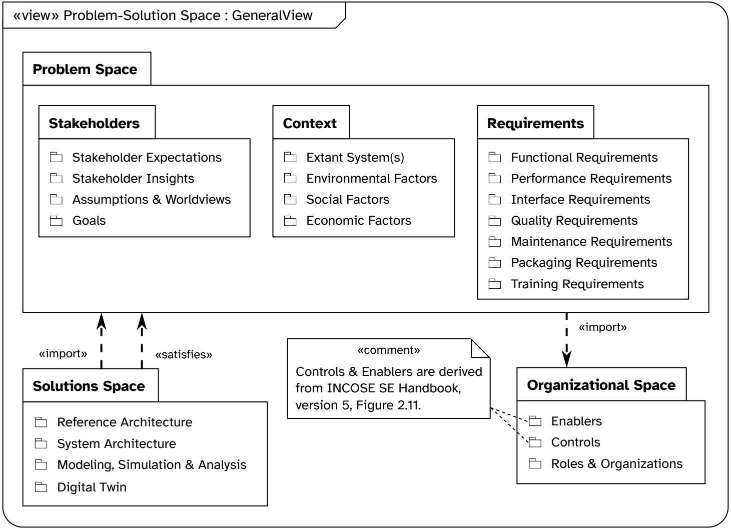
The Problem Space

→ Unmitigated Complexity Meets the Error Avalanche



[1] Claxton, Cavoli, and Johnson 2005

→ Generic Problem-Solution Space (SysML v2 depiction)



③

Defining Seamless Digital Engineering

→ Defining Seamless Digital Engineering

Definition (Seamless Digital Engineering *#SDE*)

A digital engineering tooling paradigm that guarantees model coherence and integrity by affording an elegant human-computer interface for systems modeling that is end-to-end formally verified down thru the computer hardware.

1. Digital tooling paradigm
2. Guarantees model coherence and integrity
3. Affords an elegant[1] *#HCI*[2]
4. End-to-end formally verified[3]

[1] Griffin 2010; Madni 2012 [2] Norman 1999 [3] Moore 2003

⟶ Seamless DE Architecture Tenets

Seamless Models Engineering models become seamless with consistent and coherent meta-models.

→ Seamless DE Architecture Tenets

Seamless Models Engineering models become seamless with consistent and coherent meta-models.

Composable Data Data objects become composable with commensurate data schemas.

→ Seamless DE Architecture Tenets

Seamless Models Engineering models become seamless with consistent and coherent meta-models.

Composable Data Data objects become composable with commensurate data schemas.

Live Objects Engineering artifacts maintain bidirectional traceability in a live object system.

Seamless DE Architecture Tenets

Seamless Models Engineering models become seamless with consistent and coherent meta-models.

Composable Data Data objects become composable with commensurate data schemas.

Live Objects Engineering artifacts maintain bidirectional traceability in a live object system.

Seamless Workflows Workflows become consistent and coherent with a unified HCI.

Seamless DE Architecture Tenets

Seamless Models Engineering models become seamless with consistent and coherent meta-models.

Composable Data Data objects become composable with commensurate data schemas.

Live Objects Engineering artifacts maintain bidirectional traceability in a live object system.

Seamless Workflows Workflows become consistent and coherent with a unified HCI.

Clean-slate Cybersecurity Only a clean-state computer architecture can satisfy these requirements while guaranteeing CIA properties.

Transforming Requirements from Tool Integration Questions

Dimension	**Aspect**	**Question** (Thomas and Nejmeh 1992)
Presentation Integration	Appearance and behavior	To what extent do two tools use similar screen appearance and interaction behavior?
	Interaction paradigm	To what extent do two tools use similar metaphors and mental models?
Control Integration	Provision	To what extent are a tool's services used by other tools in the environment?
	Use	To what extent does a tool use the services provided by other tools in the environment?
Process Integration	Process step	How well do relevant tools combine to support the performance of a process step?
	Event	How well do relevant tools agree on the events required to support a process?
	Constraint	How well do relevant tools cooperate to enforce a constraint?
Data Integration	Interoperability	How much work must be done for a tool to manipulate data produced by another?
	Nonredundancy	How much data managed by a tool is duplicated in or can be derived from the data managed by the other?
	Data consistency	How well do two tools cooperate to maintain the semantic constraints on the data they manipulate?
	Data exchange	How much work must be done to make the nonpersistent data generated by one tool usable by the other?
	Synchronization	How well does a tool communicate changes it makes to the values of nonpersistent, common data?

④

A Grand Challenge Driven by Needs

Evaluation Criteria for a Grand Challenge

Seamless Digital Engineering meets the criteria of a grand challenge[1]:

- Fundamental
- Astonishing
- Testable
- Inspiring
- Understandable
- Useful

- Historical
- International
- Revolutionary
- Research-directed
- Challenging
- Feasible

- Incremental
- Co-operative
- Competitive
- Effective
- Risk-managed

Note: See Table 4 in the paper for rationale.

[1] Hoare 2003

5

Conclusions and Future Work

→ Transdisciplinary Systems Engineering Is Essential

- Transdisciplinary Systems Engineering[1] is needed to tackle the mess[2] of *#DE*
- Seamless Digital Engineering draws on many interrelated fields:
  - model-based systems engineering *#MBSE*
  - computer science
  - computer engineering
  - software engineering
  - human-computer interaction *#HCI*
  - knowledge engineering
  - formal verification
- A new open-source systems engineering approach may be needed
- Systems praxis[3] is essential to success

[1] Madni 2019 [2] Ackoff 1997 [3] IFSR 2012

→ Conclusions and Future Work

Seamless Digital Engineering is a paradigm and grand challenge in *#DE* research. In conclusion:

- The reverse salient troubling *#DE* cannot be patched.
- Modern networked computing systems are non-viable in terms of cybersecurity, reliability, and usability.
- A clean-slate design is necessary to minimize complexity while meeting stakeholder needs.
- We have presented the definition and architecture tenets of *#SDE*.

Conclusions and Future Work

Seamless Digital Engineering is a paradigm and grand challenge in #*DE* research. In conclusion:

- The reverse salient troubling #*DE* cannot be patched.
- Modern networked computing systems are non-viable in terms of cybersecurity, reliability, and usability.
- A clean-slate design is necessary to minimize complexity while meeting stakeholder needs.
- We have presented the definition and architecture tenets of #*SDE*.

As a grand challenge, significant work lies ahead:

- eliciting and validating stakeholder needs;
- transforming those needs into a validated set of system requirements;
- developing a #*SDE* reference architecture with SysML v2; and
- constructing prototypes for early evaluation.

References

- R. L. Ackoff (1997). "Systems, Messes and Interactive Planning". *The Societal Engagement of Social Science* 3.1997. DOI: 10.9783/9781512819069-021
- M. Broy (2009). "Seamless Model Driven Systems Engineering Based on Formal Models". *Formal Methods and Software Engineering*. Ed. by K. Breitman and A. Cavalcanti. Springer Berlin Heidelberg. ISBN: 978-3-642-10373-5. DOI: 10.1007/978-3-642-10373-5_1
- — (2020). "Seamless Model-Based System Development: Foundations". *Engineering Trustworthy Software Systems: 5th International School, SETSS 2019, Chongqing, China, April 21–27, 2019, Tutorial Lectures*. Ed. by J. P. Bowen, Z. Liu, and Z. Zhang. Springer International Publishing. ISBN: 978-3-030-55089-9. DOI: 10.1007/978-3-030-55089-9_1
- M. Broy et al. (2010). "Seamless Model-Based Development: From Isolated Tools to Integrated Model Engineering Environments". *Proceedings of the IEEE* 98.4. DOI: 10.1109/JPROC.2009.2037771
- J. D. Claxton, C. Cavoli, and C. Johnson (2005). *Test and Evaluation Management Guide*. Tech. rep. Defense Acquisition University, Fort Belvoir, VA. URL: https://apps.dtic.mil/sti/pdfs/ADA436591.pdf
- DARPA (2010a). *Clean-Slate Design of Resilient, Adapative, Secure Hosts (CRASH)*. Broad Agency Announcement DARPA-BAA-10-70. Defense Advanced Research Projects Agency
- — (2010b). *META-II*. Broad Agency Announcement DARPA-BAA-10-59. Defense Advanced Research Projects Agency

References (Continued)

- DARPA (2015). *Circuit Realization At Faster Timescales (CRAFT).* Broad Agency Announcement DARPA-BAA-15-55. Defense Advanced Research Projects Agency
- O. Dedehayir and S. J. Mäkineif (2008). “Dynamics of Reverse Salience As Technological Performance Gap: an Empirical Study of the Personal Computertechnology System”. *Journal of Technology Management & Innovation* 3.3. DOI: 10.4067/S0718-27242008000100006
- M. D. Griffin (2010). “How Do We Fix Systems Engineering?” *61st International Astronautical Congress*. Vol. 27. September 27—October 1
- T. Hoare (2003). “The Verifying Compiler: A Grand Challenge for Computing Research”. *International Conference on Compiler Construction*. Springer. DOI: 10.1145/602382.602403
- T. P. Hughes (1993). *Networks of Power: Electrification in Western Society, 1880-1930.* JHU Press
- IFSR (2012). *The Systems Praxis Framework, Developed as a Joint Project of INCOSE and ISSS.* Tech. rep. International Federation for Systems Research (IFSR). URL: http://systemspraxis.org/framework.pdf
- A. M. Madni (2012). “Elegant Systems Design: Creative Fusion of Simplicity and Power”. *Systems Engineering* 15.3. DOI: 10.1002/sys.21209
- — (2019). “Transdisciplinary Systems Engineering: Exploiting Disciplinary Convergence to Address Grand Challenges”. *IEEE Systems, Man, and Cybernetics Magazine* 5.2. ISSN: 2333-942X. DOI: 10.1109/MSMC.2019.2899957

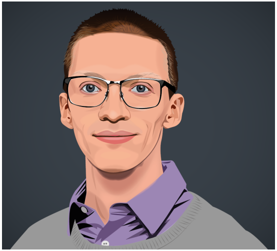
References (Continued)

- J. S. Moore (2003). "A grand challenge proposal for formal methods: A verified stack". *Formal methods at the crossroads. From panacea to foundational support*. Ed. by B. Aichernig, T. Maibaum, and A. Haeberer. Lecture Notes in Computer Science. Springer. ISBN: 9783540205272. DOI: 10.1007/978-3-540-40007-3_11
- D. A. Norman (1999). *The Invisible Computer: Why Good Products Can Fail, the Personal Computer is So Complex, and Information Appliances are the Solution*. MIT Press. ISBN: 9780262640411
- Office of the Deputy Assistant Secretary of Defense for Systems Engineering (2023). *Digital Engineering Ecosystem*. DAU Glossary Definition. Defense Acquisition University
- A. Perzylo et al. (2020). "Toward a Knowledge-Based Data Backbone for Seamless Digital Engineering in Smart Factories". *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. Vol. 1. DOI: 10.1109/ETFA46521.2020.9211943
- I. Thomas and B. A. Nejmeh (1992). "Definitions of Tool Integration for Environments". *IEEE Software* 9.2. DOI: 10.1109/52.120599

Questions?

Link to this Presentation

→ About the Presenter

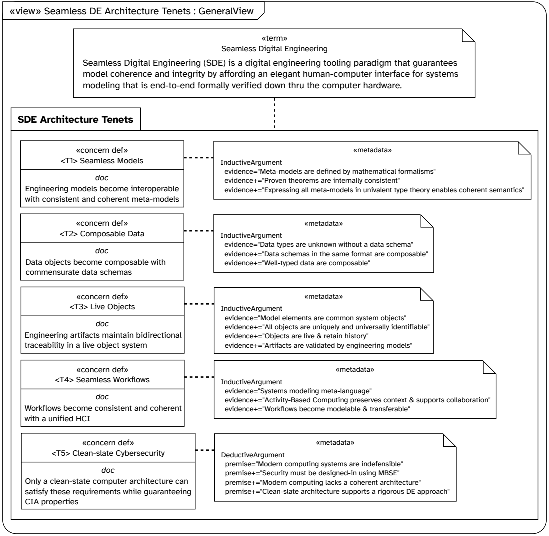


James S. Wheaton

Herber Research Group

- Ph.D. Candidate in Systems Engineering at Colorado State University, advised by Dr. Daniel R. Herber
- Current title of Ph.D. dissertation is ”Bootstrapping a Trustworthy and Seamless Digital Engineering System”
- Bachelor of Science, Mechanical Engineering from Purdue University
- Professional background in software engineering for e-commerce, and software engineering consulting for big data and blockchain systems at KPMG LLP
- Year-Round Student Intern at NASA Jet Propulsion Laboratory since January 2023 in the Robotic Systems Group, developing system architectures for Mars Sample Return projects

→ Seamless DE Architecture Tenets as packaged Concern definitions



«view» Seamless DE Architecture Tenets : GeneralView

«term»
Seamless Digital Engineering

Seamless Digital Engineering (SDE) is a digital engineering tooling paradigm that guarantees model coherence and integrity by affording an elegant human-computer interface for systems modeling that is end-to-end formally verified down thru the computer hardware.

SDE Architecture Tenets

«concern def»
<T1> Seamless Models
doc
Engineering models become interoperable with consistent and coherent meta-models

«metadata»
InductiveArgument
evidence="Meta-models are defined by mathematical formalisms"
evidence+="Proven theorems are internally consistent"
evidence+="Expressing all meta-models in univalent type theory enables coherent semantics"

«concern def»
<T2> Composable Data
doc
Data objects become composable with commensurate data schemas

«metadata»
InductiveArgument
evidence="Data types are unknown without a data schema"
evidence+="Data schemas in the same format are composable"
evidence+="Well-typed data are composable"

«concern def»
<T3> Live Objects
doc
Engineering artifacts maintain bidirectional traceability in a live object system

«metadata»
InductiveArgument
evidence="Model elements are common system objects"
evidence+="All objects are uniquely and universally identifiable"
evidence+="Objects are live & retain history"
evidence+="Artifacts are validated by engineering models"

«concern def»
<T4> Seamless Workflows
doc
Workflows become consistent and coherent with a unified HCI

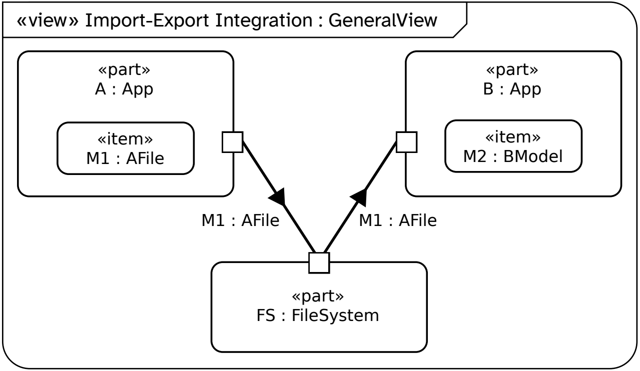
«metadata»
InductiveArgument
evidence="Systems modeling meta-language"
evidence+="Activity-Based Computing preserves context & supports collaboration"
evidence+="Workflows become modelable & transferable"

«concern def»
<T5> Clean-slate Cybersecurity
doc
Only a clean-state computer architecture can satisfy these requirements while guaranteeing CIA properties

«metadata»
DeductiveArgument
premise="Modern computing systems are indefensible"
premise+="Security must be designed-in using MBSE"
premise+="Modern computing lacks a coherent architecture"
premise+="Clean-slate architecture supports a rigorous DE approach"

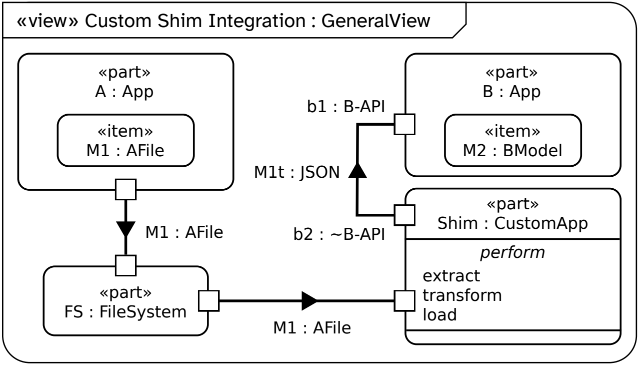
→ Import/Export Pattern of DE Tool Integration

- Manual operation
- Relies on competing or incommensurate file formats
- XMI and ReqIF designed to facilitate model data transfer
- Model information may be lost



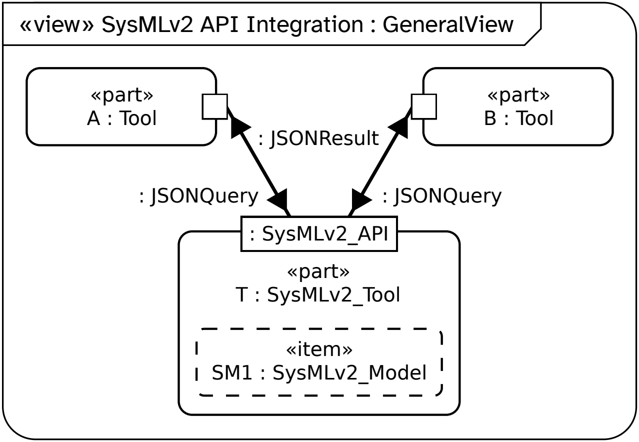
→ Custom Shim Pattern of DE Tool Integration

- Additional software engineering effort and maintenance
- File formats and APIs often change with software updates
- May be operationalized in data engineering infrastructure



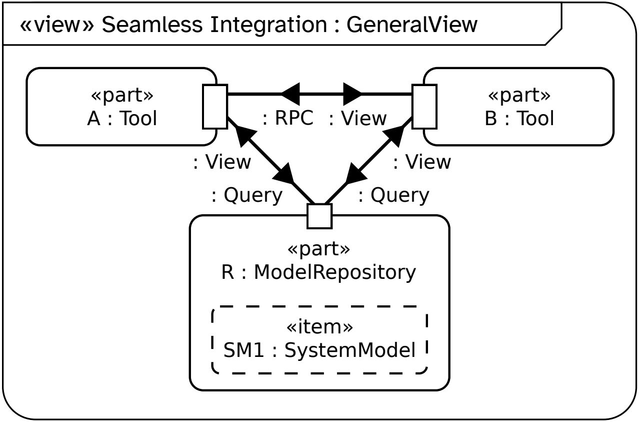
→ API Pattern of DE Tool Integration

- SysML v2 API standard places architecture model at center of *#ASoT*
- Encourages system model coherence
- Tools must add Web client functionality
- Tools must parse JSON-encoded API responses



→ Seamless Pattern of DE Tool Integration

- Employs system-wide standard interfaces, data schemas, and objects
- Interfaces are self-documenting
- Handles to live objects are provided, not copied model data
- Tools use transclusion of other tool's artifacts


«view» Seamless Integration : GeneralView
«part»
A : Tool
«part»
B : Tool
: RPC
: View
: View
: Query
: View
: Query
«part»
R : ModelRepository
«item»
SM1 : SystemModel