There Is Only One Time In Systems Engineering!

Towards a *Continuous (model-based) Engineering*

Prof. Benoit Combemale
Inria & University of Rennes
DiverSE team

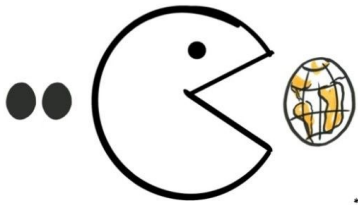
benoit.combemale@inria.fr
http://combemale.fr

Thanks to my students and all the colleagues from DiverSE, the Bellairs and WMM workshop series, the Inria/CWI Associate Team ALE, and the MDEnet International group (esp., AE group)

“Software Is Eating the World”

Digitalization of our society

- **personal context** (multimedia, health, transport, cities…)
- **professional context** (digitalization of numerous tools, processes and activities)



THE WALL STREET JOURNAL.
WSJ.com

ESSAY | AUGUST 20, 2011

Why Software Is Eating The World

By MARC ANDREESSEN

This week, Hewlett-Packard (where I am on the board) announced that it is exploring jettisoning its struggling PC business in favor of investing more heavily in software, where it sees better potential for growth. Meanwhile, Google plans to buy up the cellphone handset maker Motorola Mobility. Both moves surprised the tech world. But both moves are also in line with a trend I've observed, one that makes me optimistic about the future growth of the American and world economies, despite the recent turmoil in the stock market.

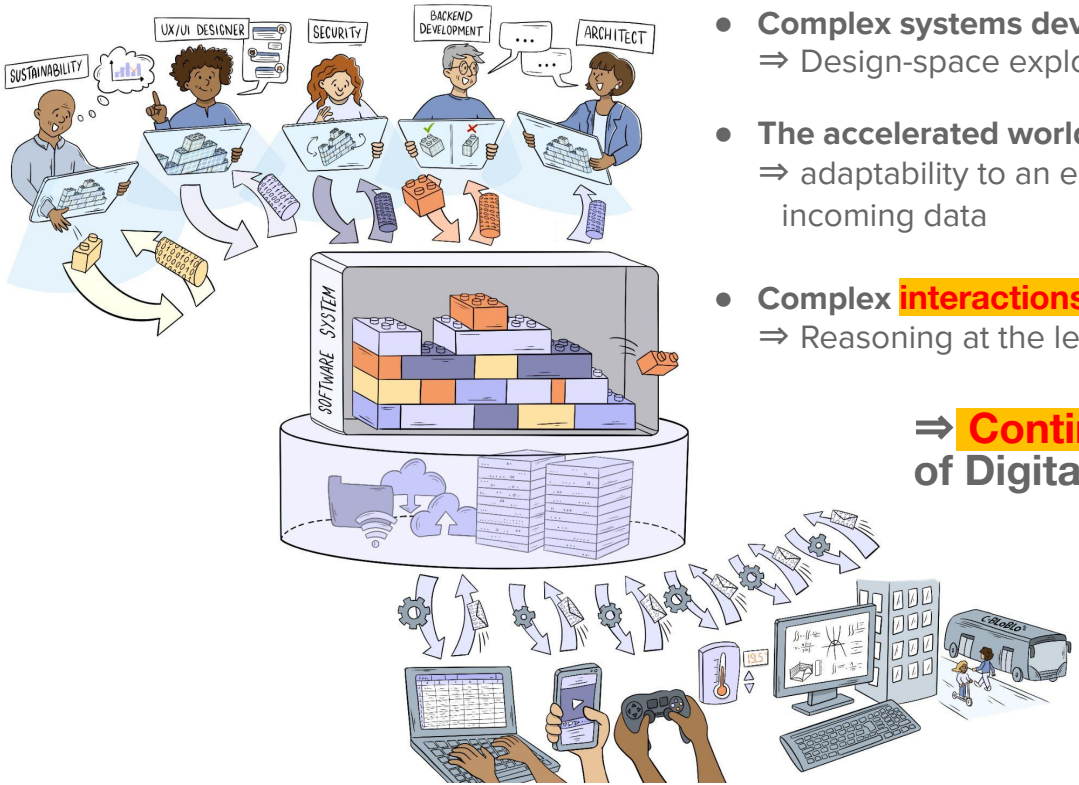
In short, software is eating the world.

More than 10 years after the peak of the 1990s dot-com bubble, a dozen or so new Internet companies like Facebook and Twitter are sparking controversy in Silicon Valley, due to their rapidly growing private market valuations, and even the occasional successful IPO. With scars from the heyday of Webvan and Pets.com still fresh in the investor psyche, people are asking, "Isn't this just a dangerous new bubble?"

I, along with others, have been arguing the other side of the case. (I am co-founder and general partner of venture capital firm Andreessen-Horowitz, which has invested in Facebook, pon, Skype, Twitter, Zynga, and Foursquare, among others. I am also personally an investor in LinkedIn.) elieve that many of the prominent new Internet companies are building real, high-growth, high-margin, ly defensible businesses.

h interview with WSJ's Kevin Delaney, Groupon and edIn investor Marc Andreessen insists that the nt popularity of tech companies does not constitute bble. He also stressed that both Apple and Google undervalued and that "the market doesn't like tech."

Towards a Digital World



- **Complex systems development belongs to a multi-dimensional space**
  ⇒ Design-space exploration, trade-off analysis & decision making
- **The accelerated world demands unprecedented levels of adaptability**
  ⇒ adaptability to an emerging sequence of requirements, often driven by incoming data
- **Complex interactions between natural and engineered systems**
  ⇒ Reasoning at the level of ecosystems

⇒ **Continuous Feedback-Driven Engineering of Digital Ecosystems**

One Time

Wait! What?

Design time
Analysis time
Development time
Training time
Run-time
Operation time
...

To what refer these “times” in Engineering?

A tool? an activity? a moment in the life cycle?

Design time and run time are no longer separable. The system evolves while it is operating. Requirements evolve while the system is deployed. There is only one time — continuous time.

Dogma of (Traditional) Software Engineering

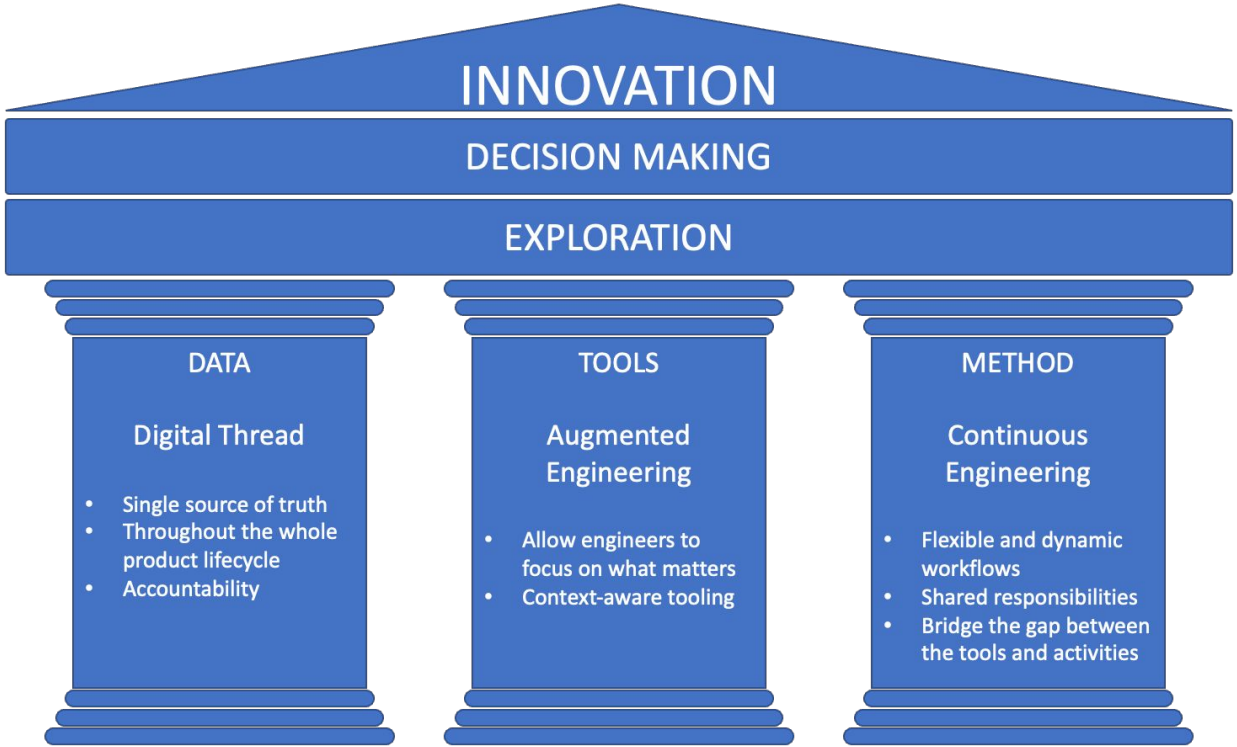
“The wealth of methods and tools that are used at development-time to forge software have no more use when the software enters the run-time stage” — Luciano Baresi and Carlo Ghezzi. The disappearing boundary between development-time and run-time. In Future of software engineering research (FoSER '10).

S,D ⊨ R, where D might continuously change, due to mobility,
but also more recently to socio interactions, wicked phenomena to consider, etc.

This prevents both

- a **seamless and continuous cross-fertilization** over the engineering processes, and
- to **explore new scenarios** beyond the ones captured in the established engineering processes

Facing Hyper Agility in CPS Development?



Towards a Continuous (model-driven) Systems Engineering

Towards a Continuous Systems Engineering

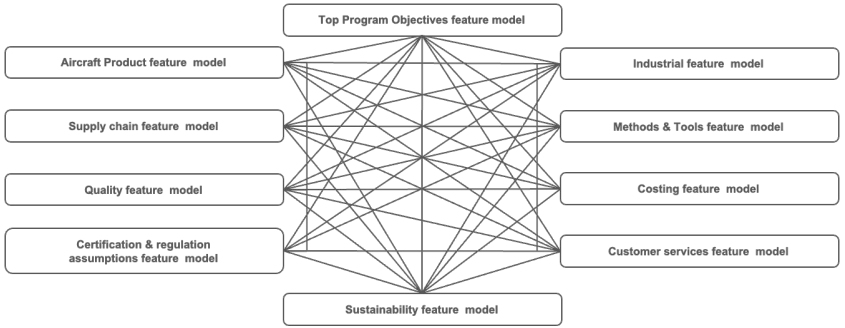
continuous systems engineering belongs to two dimensions: in space (deep variability, global decision making), and in time (digital twins, model hybridization).

Deep Variability

Variability occurs on all concerns

Variability showcases interdependencies

Variability impacts soft/sys properties



Some concerns in aircraft industry...

=> combinatorial explosion of the epistemic and ontological variability

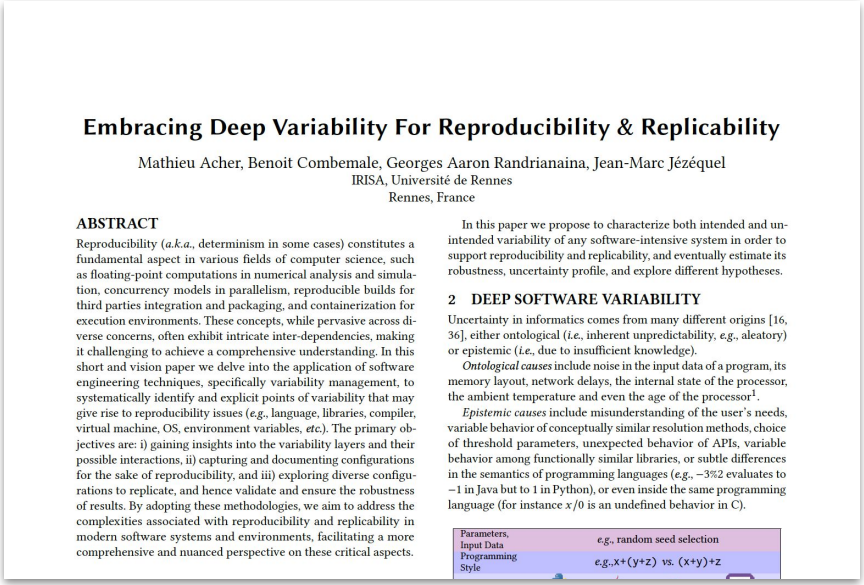
Deep Variability *refer to the interaction of all concerns modifying the behavior (including both functional and nonfunctional properties) of a complex cyber-physical, socio-technical, system*

Our Vision

Embrace deep variability!

Explicit modeling of the variability points and their relationships, such as:

1. Get insights into the variability concerns and their possible interactions
2. Capture and document configurations for the sake of **reproducibility**
3. Explore diverse configurations to **replicate**, and hence optimize, validate, increase the robustness, or provide better resilience



Embracing Deep Variability For Reproducibility & Replicability

Mathieu Acher, Benoit Combemale, Georges Aaron Randrianaina, Jean-Marc Jézéquel
IRISA, Université de Rennes
Rennes, France

ABSTRACT

Reproducibility (*a.k.a.*, determinism in some cases) constitutes a fundamental aspect in various fields of computer science, such as floating-point computations in numerical analysis and simulation, concurrency models in parallelism, reproducible builds for third parties integration and packaging, and containerization for execution environments. These concepts, while pervasive across diverse concerns, often exhibit intricate inter-dependencies, making it challenging to achieve a comprehensive understanding. In this short and vision paper we delve into the application of software engineering techniques, specifically variability management, to systematically identify and explicit points of variability that may give rise to reproducibility issues (*e.g.*, language, libraries, compiler, virtual machine, OS, environment variables, *etc.*). The primary objectives are: i) gaining insights into the variability layers and their possible interactions, ii) capturing and documenting configurations for the sake of reproducibility, and iii) exploring diverse configurations to replicate, and hence validate and ensure the robustness of results. By adopting these methodologies, we aim to address the complexities associated with reproducibility and replicability in modern software systems and environments, facilitating a more comprehensive and nuanced perspective on these critical aspects.

In this paper we propose to characterize both intended and unintended variability of any software-intensive system in order to support reproducibility and replicability, and eventually estimate its robustness, uncertainty profile, and explore different hypotheses.

2 DEEP SOFTWARE VARIABILITY

Uncertainty in informatics comes from many different origins [16, 36], either ontological (*i.e.*, inherent unpredictability, *e.g.*, aleatory) or epistemic (*i.e.*, due to insufficient knowledge).

Ontological causes include noise in the input data of a program, its memory layout, network delays, the internal state of the processor, the ambient temperature and even the age of the processor[1].

Epistemic causes include misunderstanding of the user's needs, variable behavior of conceptually similar resolution methods, choice of threshold parameters, unexpected behavior of APIs, variable behavior among functionally similar libraries, or subtle differences in the semantics of programming languages (*e.g.*, $-3\%2$ evaluates to -1 in Java but to 1 in Python), or even inside the same programming language (for instance $x/0$ is an undefined behavior in C).

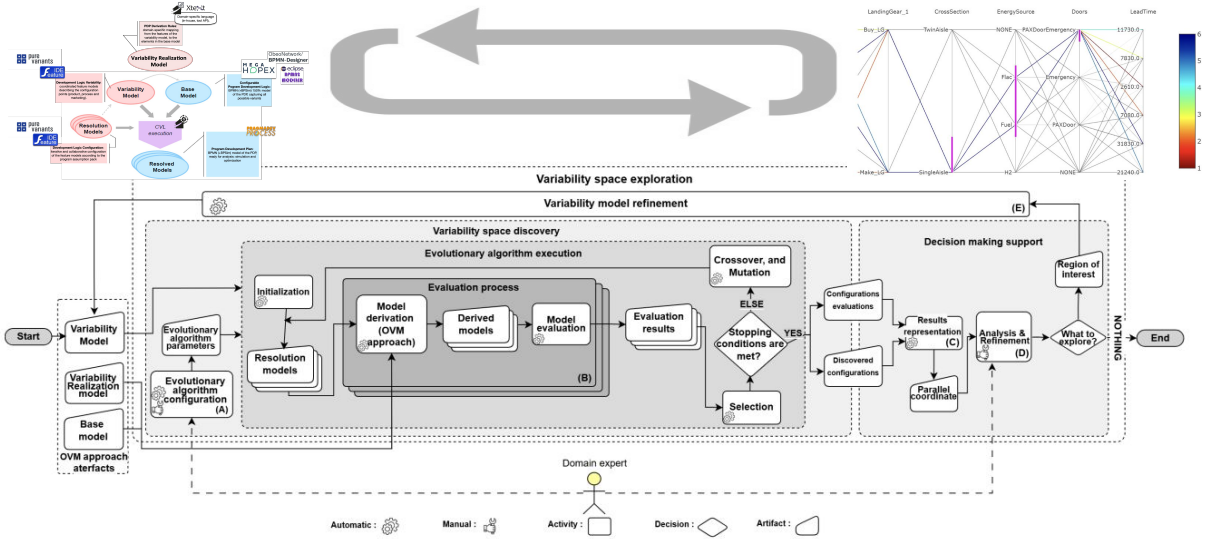
Parameters, Input Data	*e.g.*, random seed selection
Programming Style	*e.g.*, `x+(y+z)` vs. `(x+y)+z`

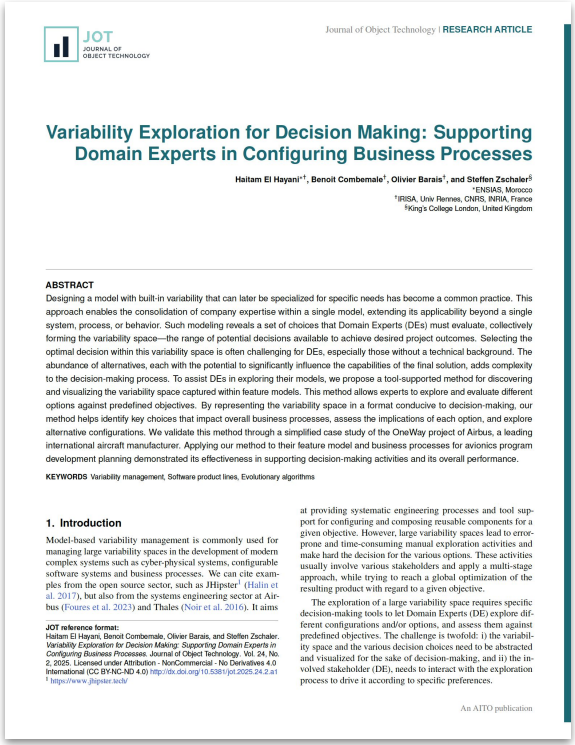
ACM REP 2024

⇒ We aim to **address the complexities associated with reproducibility and replicability in modern cyber-physical systems and environments**, facilitating a more comprehensive and nuanced perspective on these critical concerns.

Variability Exploration for Decision Making

- Automate the variability space exploration (genetic algorithms, *AI agents*)
- Interactivity with decision makers through parallel charts





JOT JOURNAL OF OBJECT TECHNOLOGY

Journal of Object Technology | RESEARCH ARTICLE

Variability Exploration for Decision Making: Supporting Domain Experts in Configuring Business Processes

Haitam El Hayani[*†], Benoit Combemale[†], Olivier Barais[†], and Steffen Zschaler[§]
[*]ENSIAS, Morocco
[†]IRISA, Univ Rennes, CNRS, INRIA, France
[§]King's College London, United Kingdom

ABSTRACT

Designing a model with built-in variability that can later be specialized for specific needs has become a common practice. This approach enables the consolidation of company expertise within a single model, extending its applicability beyond a single system, process, or behavior. Such modeling reveals a set of choices that Domain Experts (DEs) must evaluate, collectively forming the variability space—the range of potential decisions available to achieve desired project outcomes. Selecting the optimal decision within this variability space is often challenging for DEs, especially those without a technical background. The abundance of alternatives, each with the potential to significantly influence the capabilities of the final solution, adds complexity to the decision-making process. To assist DEs in exploring their models, we propose a tool-supported method for discovering and visualizing the variability space captured within feature models. This method allows experts to explore and evaluate different options against predefined objectives. By representing the variability space in a format conducive to decision-making, our method helps identify key choices that impact overall business processes, assess the implications of each option, and explore alternative configurations. We validate this method through a simplified case study of the OneWay project of Airbus, a leading international aircraft manufacturer. Applying our method to their feature model and business processes for avionics program development planning demonstrated its effectiveness in supporting decision-making activities and its overall performance.

KEYWORDS Variability management, Software product lines, Evolutionary algorithms

1. Introduction

Model-based variability management is commonly used for managing large variability spaces in the development of modern complex systems such as cyber-physical systems, configurable software systems and business processes. We can cite examples from the open source sector, such as JHipster[1] (Halin et al. 2017), but also from the systems engineering sector at Airbus (Foures et al. 2023) and Thales (Noir et al. 2016). It aims at providing systematic engineering processes and tool support for configuring and composing reusable components for a given objective. However, large variability spaces lead to error-prone and time-consuming manual exploration activities and make hard the decision for the various options. These activities usually involve various stakeholders and apply a multi-stage approach, while trying to reach a global optimization of the resulting product with regard to a given objective.

The exploration of a large variability space requires specific decision-making tools to let Domain Experts (DE) explore different configurations and/or options, and assess them against predefined objectives. The challenge is twofold: i) the variability space and the various decision choices need to be abstracted and visualized for the sake of decision-making, and ii) the involved stakeholder (DE), needs to interact with the exploration process to drive it according to specific preferences.

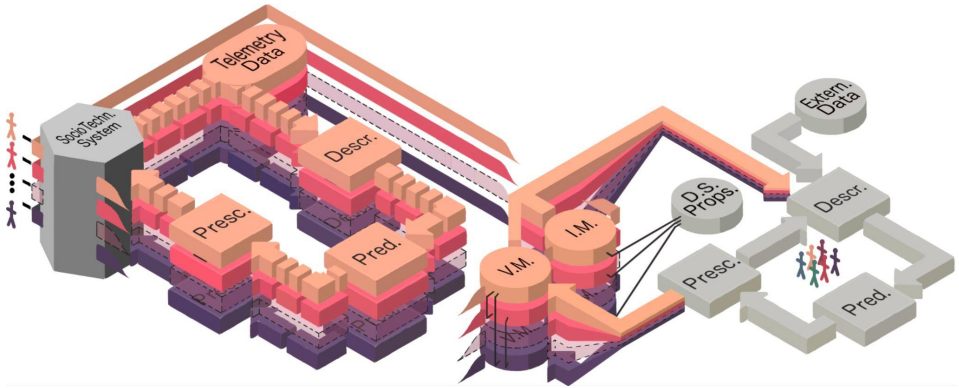
JOT reference format:
Haitam El Hayani, Benoit Combemale, Olivier Barais, and Steffen Zschaler. *Variability Exploration for Decision Making: Supporting Domain Experts in Configuring Business Processes.* Journal of Object Technology. Vol. 24, No. 2, 2025. http://dx.doi.org/10.5381/jot.2025.24.2.a1
[1] https://www.jhipster.tech/

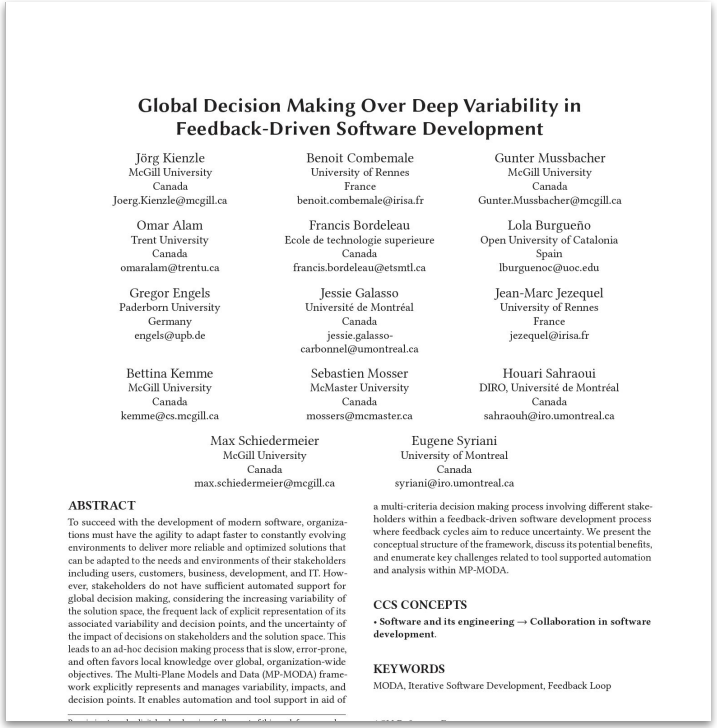
An AITO publication

JOT 2025, cf. https://inria.hal.science/hal-05029778

Feedback-Driven Software Development

- **Deep Software Variability needs decision-making support**
- **The MultiPlane MODA Framework** (Bellairs'22)
  - encapsulate the variability and impact intra-/inter- plane
  - **"Decision Space"** that derives from the dependencies in individual variability models and impact models
  - **Global feedback loop**





Global Decision Making Over Deep Variability in Feedback-Driven Software Development

Jörg Kienzle, McGill University, Canada, Joerg.Kienzle@mcgill.ca
Benoit Combemale, University of Rennes, France, benoit.combemale@irisa.fr
Gunter Mussbacher, McGill University, Canada, Gunter.Mussbacher@mcgill.ca
Omar Alam, Trent University, Canada, omaralam@trentu.ca
Francis Bordeleau, Ecole de technologie superieure, Canada, francis.bordeleau@etsmtl.ca
Lola Burgueño, Open University of Catalonia, Spain, lburguenoc@uoc.edu
Gregor Engels, Paderborn University, Germany, engels@upb.de
Jessie Galasso, Université de Montréal, Canada, jessie.galasso-carbonnel@umontreal.ca
Jean-Marc Jezequel, University of Rennes, France, jezequel@irisa.fr
Bettina Kemme, McGill University, Canada, kemme@cs.mcgill.ca
Sebastien Mosser, McMaster University, Canada, mossers@mcmaster.ca
Houari Sahraoui, DIRO, Université de Montréal, Canada, sahraouh@iro.umontreal.ca
Max Schiedermeier, McGill University, Canada, max.schiedermeier@mcgill.ca
Eugene Syriani, University of Montreal, Canada, syriani@iro.umontreal.ca

ABSTRACT

To succeed with the development of modern software, organizations must have the agility to adapt faster to constantly evolving environments to deliver more reliable and optimized solutions that can be adapted to the needs and environments of their stakeholders including users, customers, business, development, and IT. However, stakeholders do not have sufficient automated support for global decision making, considering the increasing variability of the solution space, the frequent lack of explicit representation of its associated variability and decision points, and the uncertainty of the impact of decisions on stakeholders and the solution space. This leads to an ad-hoc decision making process that is slow, error-prone, and often favors local knowledge over global, organization-wide objectives. The Multi-Plane Models and Data (MP-MODA) framework explicitly represents and manages variability, impacts, and decision points. It enables automation and tool support in aid of a multi-criteria decision making process involving different stakeholders within a feedback-driven software development process where feedback cycles aim to reduce uncertainty. We present the conceptual structure of the framework, discuss its potential benefits, and enumerate key challenges related to tool supported automation and analysis within MP-MODA.

CCS CONCEPTS

• **Software and its engineering → Collaboration in software development**.

KEYWORDS

MODA, Iterative Software Development, Feedback Loop

ASE 2022, cf. https://hal.inria.fr/hal-03770004

RE for Cyber-Physical Systems Development

1. There is a clear need for advanced global decision support that is **cross-discipline** and reduces information overload while prioritizing uncertain or hard areas
2. It is a challenge to **reduce information overload** while making balanced decisions that address uncertainty and maintain ecosystem equilibrium to achieve **continuous decision making**

Global Decision Making Support for Complex System Development

Lola Burgueño
ITIS Software
University of Malaga
Malaga, Spain
lolaburgueno@uma.es

Damien Foures
DDMS
Airbus Group
Toulouse, France
damien.da.foures@airbus.com

Benoit Combemale
ESIR & IRISA
University of Rennes
Rennes, France
benoit.combemale@irisa.fr

Jörg Kienzle
ITIS Software / School of Computer Science
University of Málaga / McGill University
Málaga, Spain / Montreal, Canada
joerg.kienzle@uma.es / mcgill.ca

Gunter Mussbacher
Department of Electrical and Computer Engineering
McGill University / INRIA
Montreal, Canada / Rennes, France
gunter.mussbacher@mcgill.ca

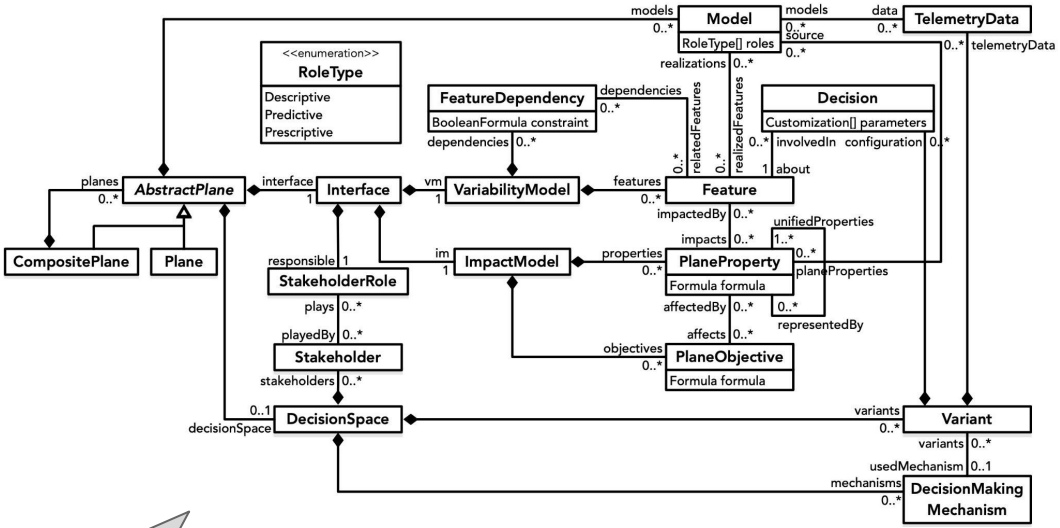
***Abstract*—To succeed with the development of modern and complex systems (e.g., aircrafts or production systems), organizations must have the agility to adapt faster to constantly evolving requirements in order to deliver more reliable and optimized solutions that can be adapted to the needs and environments of their stakeholders including users, customers, suppliers, and partners. However, stakeholders do not have sufficiently explicit and systematic support for global decision making, considering the vast decision space and complex inter-relationships. This decision space is characterized by increasing yet inadequately represented variability and the uncertainty of the impact of decisions on stakeholders and the solution space. This leads to an ad-hoc decision making process that is slow, error-prone, and often favors local knowledge over global, organization-wide objectives. As a result, one team's design decisions may impose too restrictive requirements on another team. In this paper, we evaluate our understanding of global decision making in the context of complex system development based on a conceptual model which explicitly represents and manages decision spaces including variability and impacts. We have conducted our evaluation by means of a case study where we interviewed domain experts with an average of 20 years of experience in complex system industries and report the key findings and remaining challenges. In the future, we aim at providing explicit and systematic tool-supported approaches for global decision making support for complex systems.**

***Index Terms*—Global Decision Making, Multi-Stakeholder, Variability, Impact, Requirements, Design.**

the organization from reaching a better global result. Global decision making that considers not only the solution space of each specialized team, but also the overall solution space of the organization is required. Furthermore, organizations must have the agility to adapt faster to constantly evolving requirements to succeed with complex system development by delivering more reliable and optimized solutions that can be adapted to the needs and environments of their stakeholders including users, customers, suppliers, and partners. Without automated support, teams have to revert to an ad-hoc decision making process for requirements and design that is slow, error-prone, and often favors local knowledge over global, organization-wide objectives. The vast decision space with ever increasing, but inadequately represented variability, and the uncertainty of the impact of decisions on stakeholders and the solution space further exacerbate this decision making problem. While there is a growing body of knowledge around variability management and decision making (see related work in Section VI), the challenges of global decision making in complex system development in an industrial context are not yet well understood.

Based on an initial conceptual model, we evaluate our understanding of global decision making by means of a case study in the context of complex system development and report our findings in this paper. The conceptual model explicitly

RE for Complex System Development



continuous decision making support?

uncertainty management?

Global Decision Making Support for Complex System Development

Lola Burgueño
ITIS Software
University of Malaga
Malaga, Spain
lolaburgueno@uma.es

Damien Foures
DDMS
Airbus Group
Toulouse, France
damien.da.foures@airbus.com

Benoit Combemale
ESIR & IRISA
University of Rennes
Rennes, France
benoit.combemale@irisa.fr

Jörg Kienzle
ITIS Software / School of Computer Science
University of Málaga / McGill University
Málaga, Spain / Montreal, Canada
joerg.kienzle@uma.es / mcgill.ca

Gunter Mussbacher
Department of Electrical and Computer Engineering
McGill University / INRIA
Montreal, Canada / Rennes, France
gunter.mussbacher@mcgill.ca

***Abstract*—To succeed with the development of modern and complex systems (e.g., aircrafts or production systems), organizations must have the agility to adapt faster to constantly evolving requirements in order to deliver more reliable and optimized solutions that can be adapted to the needs and environments of their stakeholders including users, customers, suppliers, and partners. However, stakeholders do not have sufficiently explicit and systematic support for global decision making, considering the vast decision space and complex inter-relationships. This decision space is characterized by increasing yet inadequately represented variability and the uncertainty of the impact of decisions on stakeholders and the solution space. This leads to an ad-hoc decision making process that is slow, error-prone, and often favors local knowledge over global, organization-wide objectives. As a result, one team's design decisions may impose too restrictive requirements on another team. In this paper, we evaluate our understanding of global decision making in the context of complex system development based on a conceptual model which explicitly represents and manages decision spaces including variability and impacts. We have conducted our evaluation by means of a case study where we interviewed domain experts with an average of 20 years of experience in complex system industries and report the key findings and remaining challenges. In the future, we aim at providing explicit and systematic tool-supported approaches for global decision making support for complex systems.**

***Index Terms*—Global Decision Making, Multi-Stakeholder, Variability, Impact, Requirements, Design.**

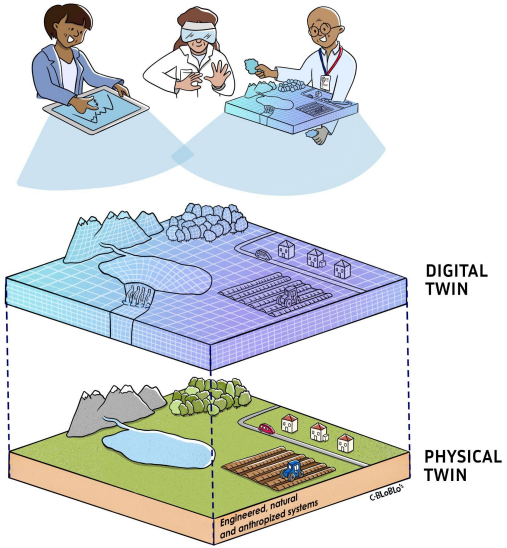
the organization from reaching a better global result. Global decision making that considers not only the solution space of each specialized team, but also the overall solution space of the organization is required. Furthermore, organizations must have the agility to adapt faster to constantly evolving requirements to succeed with complex system development by delivering more reliable and optimized solutions that can be adapted to the needs and environments of their stakeholders including users, customers, suppliers, and partners. Without automated support, teams have to revert to an ad-hoc decision making process for requirements and design that is slow, error-prone, and often favors local knowledge over global, organization-wide objectives. The vast decision space with ever increasing, but inadequately represented variability, and the uncertainty of the impact of decisions on stakeholders and the solution space further exacerbate this decision making problem. While there is a growing body of knowledge around variability management and decision making (see related work in Section VI), the challenges of global decision making in complex system development in an industrial context are not yet well understood.

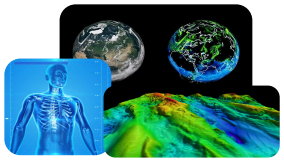
Based on an initial conceptual model, we evaluate our understanding of global decision making by means of a case study in the context of complex system development and report our findings in this paper. The conceptual model explicitly

Feedback-driven eng. with Digital Twins

"A digital twin is a software system that ***purposefully represents*** *an original,* ***accurately reflects*** *its changes, can* ***act upon*** *its original, and provides* ***added value to users or other systems****."*

-- Michael, Rumpe, Wortmann, Combemale, Jézéquel.





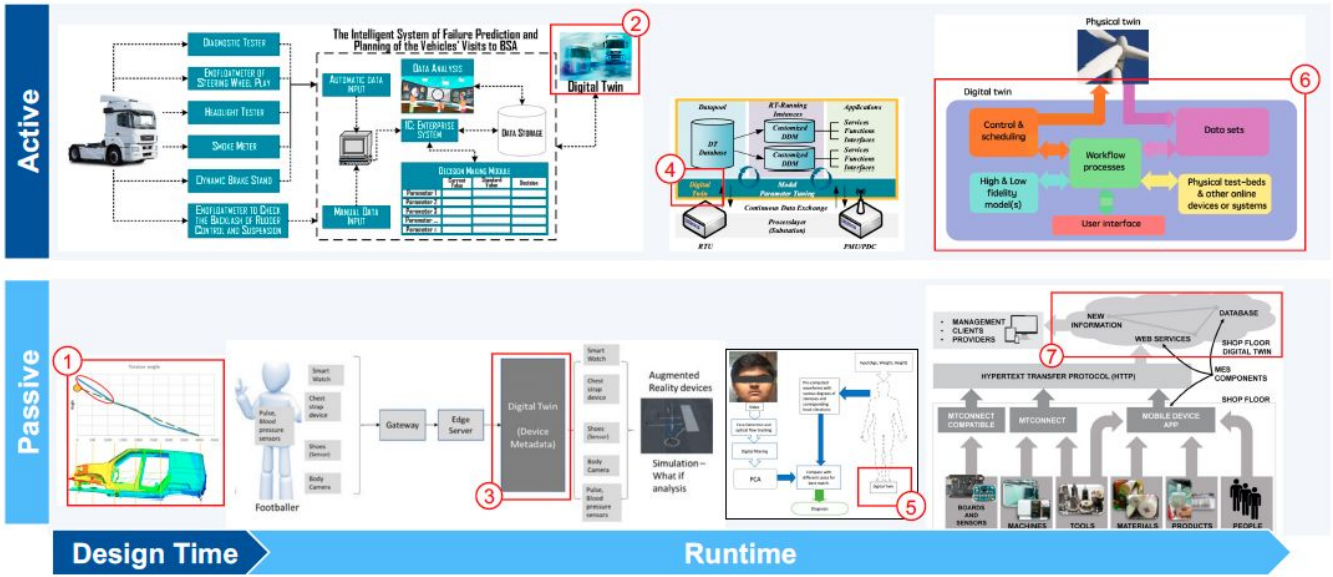
… natural systems

… anthropized systems

… engineered systems

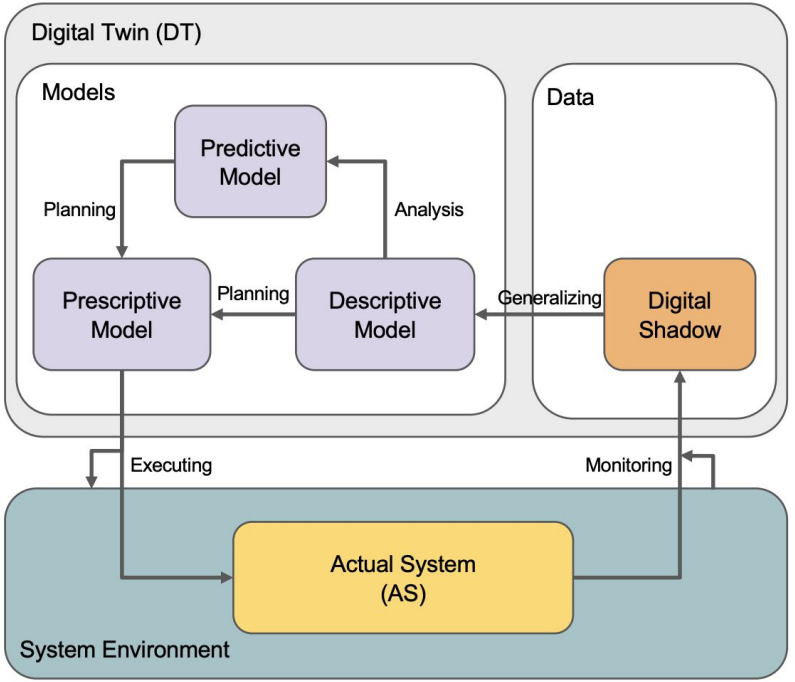
(Cyber-Physical Systems, infrastructures, and business processes)

Digital Twin: Seamless Continuum over Engineering Processes



Dalibor, Jansen, Rumpe, Schmalzing, Wortmann: *A Cross-Domain Systematic Mapping Study on Software Engineering for Digital Twins.* In: Journal of Systems and Software, 2022.

Digital Twin: The Role of Models and Data



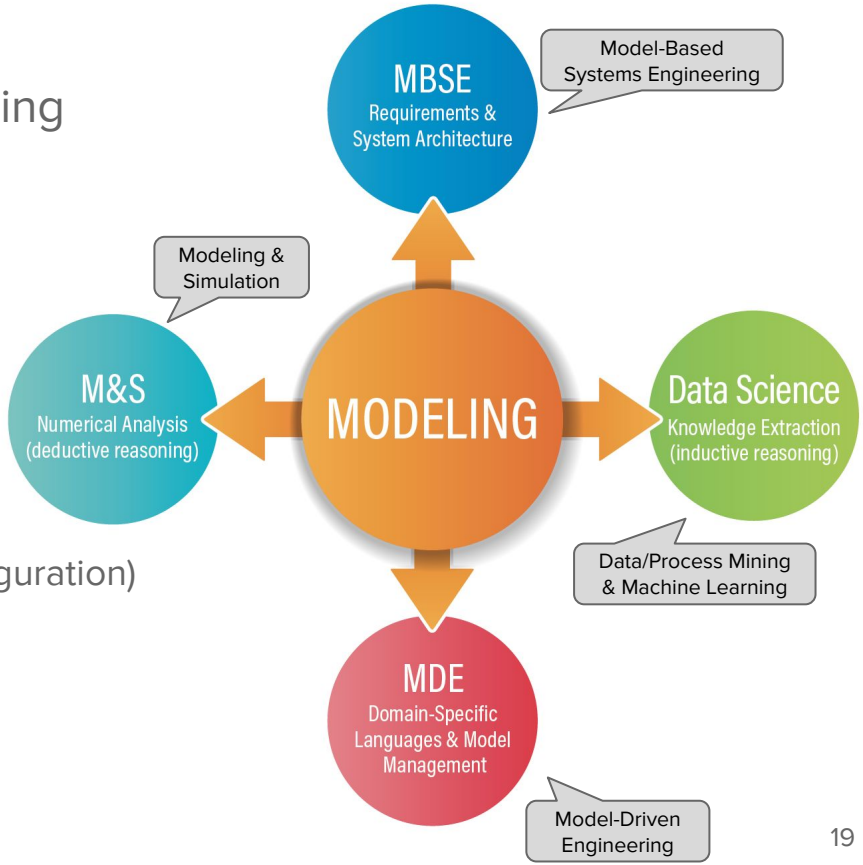
Conceptualizing Digital Twins. Romina Eramo, Francis Bordeleau, Benoit Combemale, et al.. IEEE Software, March-April 2022, pp. 39-46, vol. 39.

Model Hybridization

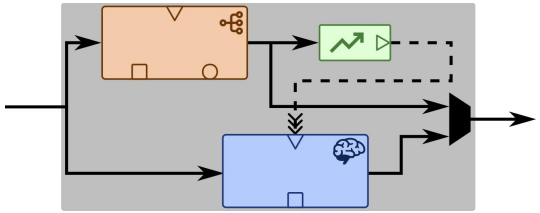
Man-made and inferred abstraction engineering

Towards a unifying theory for inductive and deductive reasoning?

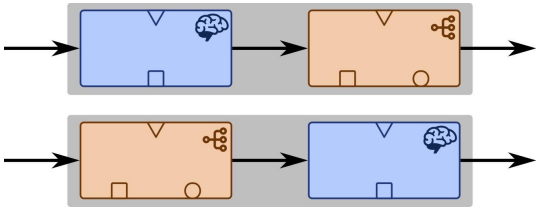
- **Hybrid modeling**
  - coordinated use of heterogeneous models
- **Adaptive modeling**
  - model adaptation (inference/refinement/configuration)



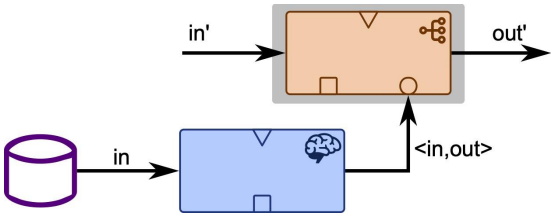
Towards
A Catalog of Model Hybridization Pattern



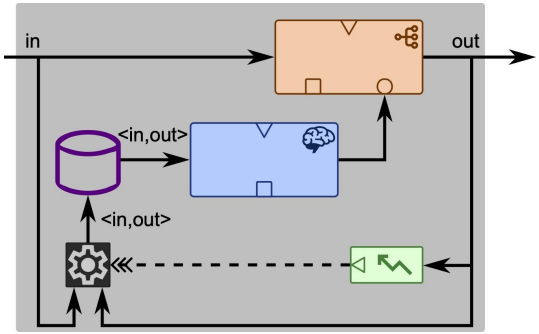
The **Shortcut** pattern sends inputs first to an inductive model acting as a replacement for the deductive model



The **Pre-/Post-Processing** pattern uses an inductive model to produce specific inputs or to refine outputs from a deductive model



The **Trainer** pattern enables modeling phenomena for which the knowledge is scarce/limited



The **Continual Learning** pattern combines a calibrated deductive model and a pre-trained inductive model to perform continual learning

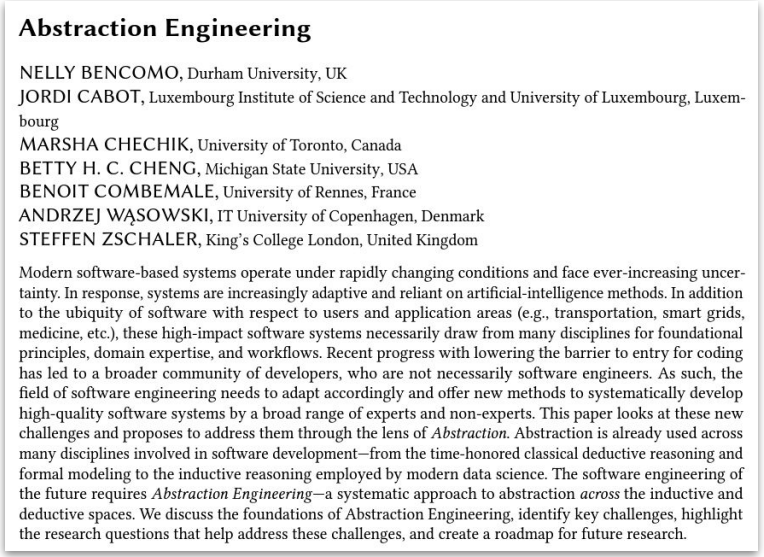
Abstraction Engineering

“The demands of future systems require an engineering approach to the ***systematic creation, maintenance, and coordinated use of abstractions”***

Abstraction Engineering: Wait, What?

An abstraction is a representation of a concept of concern in a particular context. Each abstraction, as a minimum, has a name and a purpose or intention.

Abstraction Engineering (AE) is the discipline of constructing and manipulating abstractions for a given purpose.



Abstraction Engineering

NELLY BENCOMO, Durham University, UK
JORDI CABOT, Luxembourg Institute of Science and Technology and University of Luxembourg, Luxembourg
MARSHA CHECHIK, University of Toronto, Canada
BETTY H. C. CHENG, Michigan State University, USA
BENOIT COMBEMALE, University of Rennes, France
ANDRZEJ WĄSOWSKI, IT University of Copenhagen, Denmark
STEFFEN ZSCHALER, King's College London, United Kingdom

Modern software-based systems operate under rapidly changing conditions and face ever-increasing uncertainty. In response, systems are increasingly adaptive and reliant on artificial-intelligence methods. In addition to the ubiquity of software with respect to users and application areas (e.g., transportation, smart grids, medicine, etc.), these high-impact software systems necessarily draw from many disciplines for foundational principles, domain expertise, and workflows. Recent progress with lowering the barrier to entry for coding has led to a broader community of developers, who are not necessarily software engineers. As such, the field of software engineering needs to adapt accordingly and offer new methods to systematically develop high-quality software systems by a broad range of experts and non-experts. This paper looks at these new challenges and proposes to address them through the lens of *Abstraction*. Abstraction is already used across many disciplines involved in software development—from the time-honored classical deductive reasoning and formal modeling to the inductive reasoning employed by modern data science. The software engineering of the future requires *Abstraction Engineering*—a systematic approach to abstraction *across* the inductive and deductive spaces. We discuss the foundations of Abstraction Engineering, identify key challenges, highlight the research questions that help address these challenges, and create a roadmap for future research.

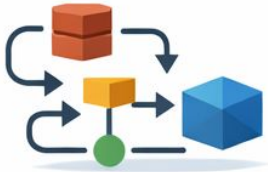
cf. https://arxiv.org/abs/2408.14074

⇒ See https://international.mde-network.org/top_navigation/abstraction_engineering.html

We are not starting from scratch

Many of the research questions identified have (partial or full) solutions developed in particular fields of Computer Science

Language-oriented programming & Software Language Engineering

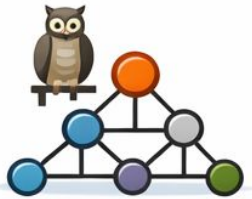


Model-driven engineering

Modeling & Simulation

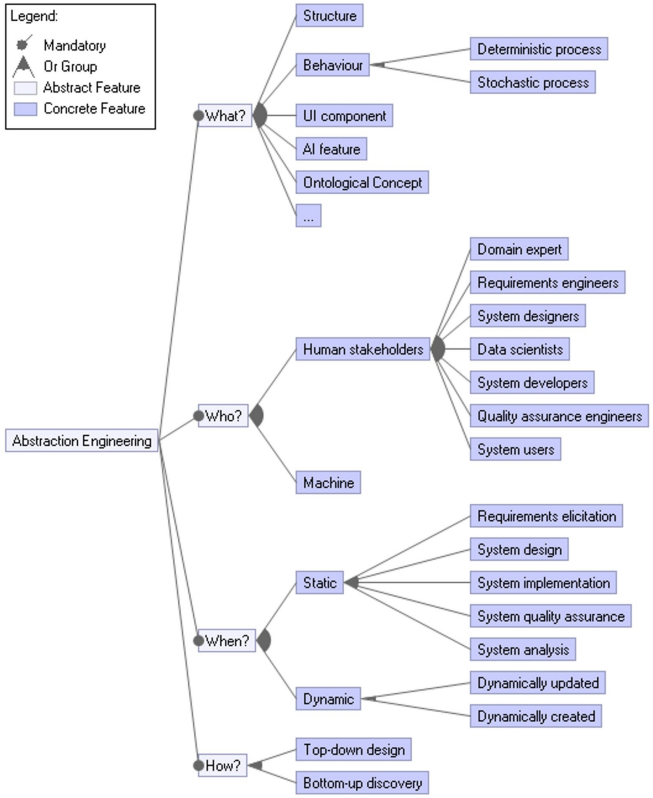
Verification / Formal methods

Artificial Intelligence



Foundational Ontologies

Anatomy of an abstraction



Abstraction Engineering

NELLY BENCOMO, Durham University, UK
JORDI CABOT, Luxembourg Institute of Science and Technology and University of Luxembourg, Luxembourg
MARSHA CHECHIK, University of Toronto, Canada
BETTY H. C. CHENG, Michigan State University, USA
BENOIT COMBEMALE, University of Rennes, France
ANDRZEJ WĄSOWSKI, IT University of Copenhagen, Denmark
STEFFEN ZSCHALER, King's College London, United Kingdom

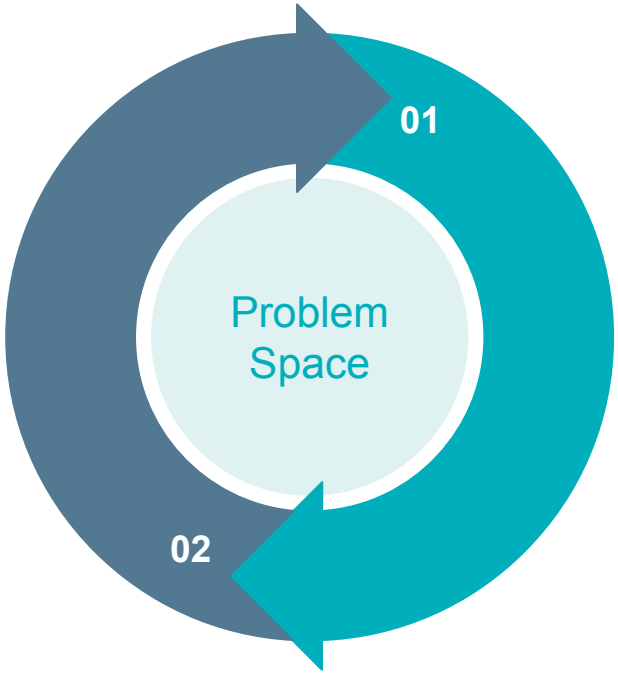
Modern software-based systems operate under rapidly changing conditions and face ever-increasing uncertainty. In response, systems are increasingly adaptive and reliant on artificial-intelligence methods. In addition to the ubiquity of software with respect to users and application areas (e.g., transportation, smart grids, medicine, etc.), these high-impact software systems necessarily draw from many disciplines for foundational principles, domain expertise, and workflows. Recent progress with lowering the barrier to entry for coding has led to a broader community of developers, who are not necessarily software engineers. As such, the field of software engineering needs to adapt accordingly and offer new methods to systematically develop high-quality software systems by a broad range of experts and non-experts. This paper looks at these new challenges and proposes to address them through the lens of *Abstraction*. Abstraction is already used across many disciplines involved in software development—from the time-honored classical deductive reasoning and formal modeling to the inductive reasoning employed by modern data science. The software engineering of the future requires *Abstraction Engineering*—a systematic approach to abstraction *across* the inductive and deductive spaces. We discuss the foundations of Abstraction Engineering, identify key challenges, highlight the research questions that help address these challenges, and create a roadmap for future research.

cf. https://arxiv.org/abs/2408.14074

Abstraction Engineering: Key Challenges

Complexity

- Manage abstractions dynamically discovered
- Combine designed vs discovered abstractions
- Contextualize abstractions
- Combine abstractions from different stakeholders



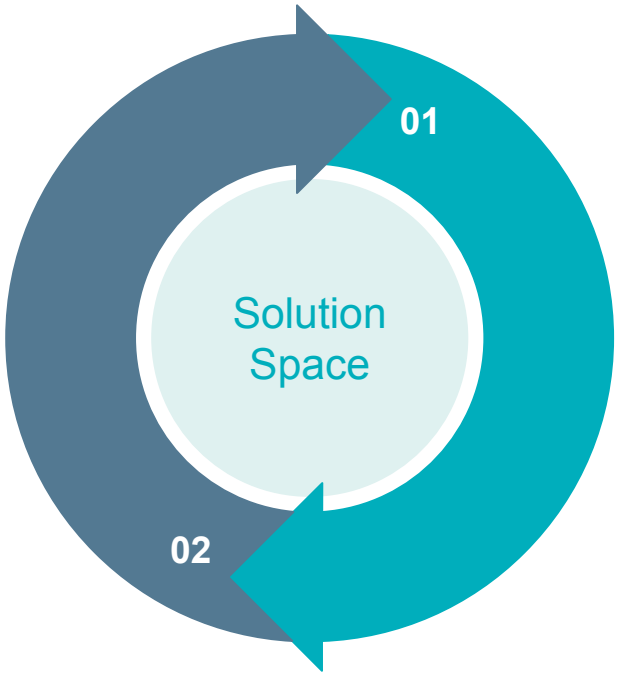
Uncertainty

- From precise to uncertain abstractions
- Measurement uncertainty
- Aleatoric uncertainty
- (ML) Model uncertainty
- Compose various sources of uncertainty

Abstraction Engineering: Key Challenges

Compositionality

- Composing AI and non-AI subsystems
- Reason about relationships between abstractions
- Agreement and conflict on abstractions coming from different stakeholders



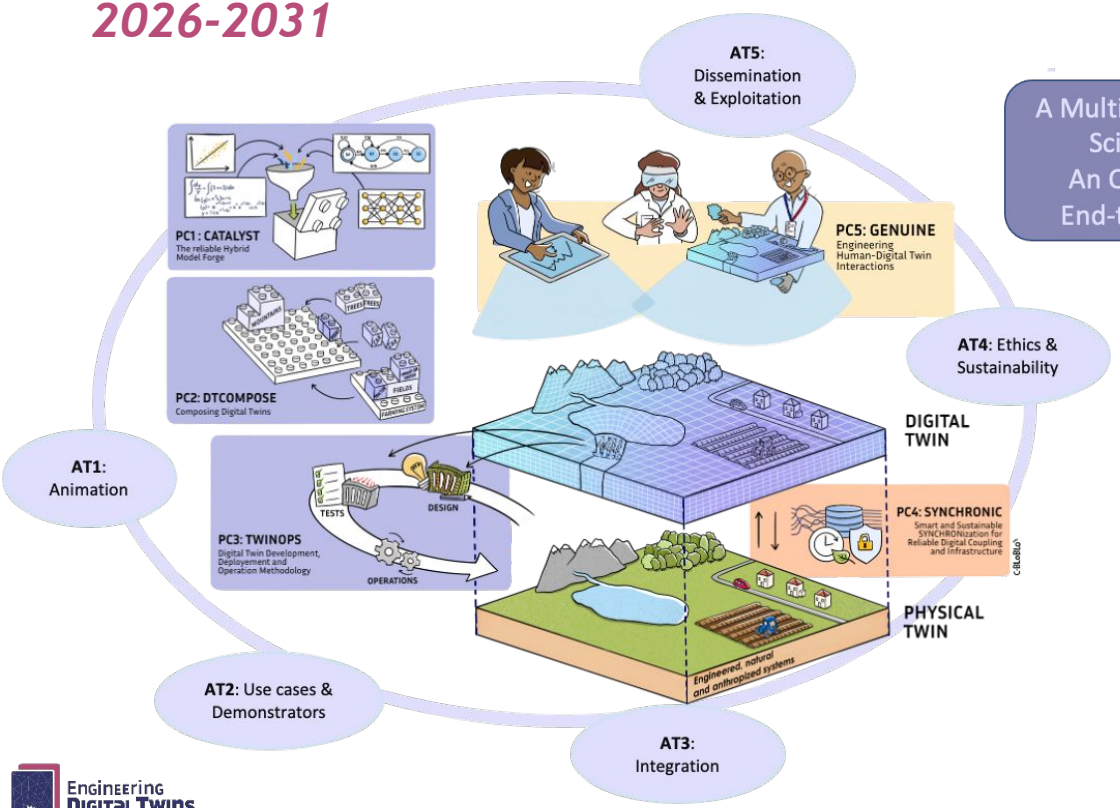
Reuse

- What quality assurances can be transferred
- How abstraction use shift over time
- Hierarchies of abstractions to create new abstractions for particular applications

French National Program on *Engineering Digital Twins* (EDT)

RESEARCH PROGRAM
Engineering Digital Twins

2026-2031



A Multidisciplinary Community
Scientific Foundations
An Open Source Platform
End-to-end Demonstrators

40+ PhDs / PostDocs
40+ Uses Cases
35+ Partners involved

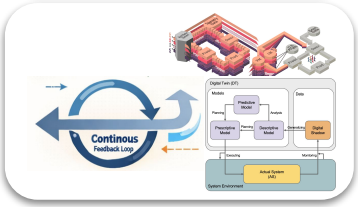
Focused Projects	9.2 M€
Cross-cutting Actions	3.4 M€
Governance	1 M€
Administrative fees	3.4 M€
Total	17 M€

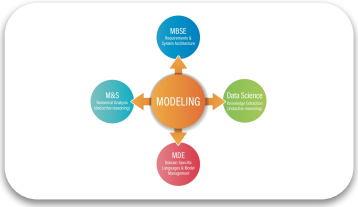
Engineering Digital Twins: A Research Roadmap.
Benoît Combemale, Pascale Vicat-Blanc, Arnaud Blouin, Hind Bril El Haouzi, Jean-Michel Bruel, Julien Deantoni, Thierry Duval, Sébastien Gérard, Jean-Marc Jézéquel. EDTConf, 2025. preprint: https://hal.science/hal-05223776

Engineering Digital Twins

FRANCE 2030
RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

Take Away Messages





- **There is only one time to tame *Systems Hyper-Agility***
  - **Innovation = Exploration** & Optimization
    - Breakthrough over incremental innovation
  - **New temporal adaptability**
    - Dynamic environment
- **Towards a *Continuous (model-driven) Software Engineering***
  - Deep Variability, Global Decision Making
  - Feedback-driven Development, Digital twins
- **Open challenges**
  - **Foundations**: abstraction engineering (e.g. model hybridization)
  - **Technologies**: context-aware dev tool, DT engineering…
  - **Businesses**: IP management, standards, patents…

We don't need faster cycles. We need systems engineering that accepts that change never stops — and that models must live with the system.

There Is Only One Time In Systems Engineering!

Towards a Continuous (Model-Based) Software Engineering

Systems engineering is a complex endeavor encompassing diverse socio-technical activities. Traditionally, these activities are organized along a development life cycle—from design to operation—applying engineering processes at both design and run time, and across application and domain levels. This organization has given rise to well-established life cycle models (e.g., V-model, Scrum) and practices (e.g., DevOps) designed to manage the complexity of software-intensive systems. It also shapes our tools, methods, and even communities within the software and systems engineering field—an illustration of Conway's Law applied to our own discipline! While this structured organization was crucial in the early days of the field ("divide and conquer"), I argue that it now limits the adaptability required to address what I call hyper agility. Modern systems evolve at an accelerating pace, operate in dynamic environments, and face increasing uncertainty. Meeting these challenges demands continuous, model-based engineering of complex cyber-physical and socio-technical ecosystems. In this talk, I will discuss challenges in variability management and abstraction engineering to better support feedback-driven development, and explore how the digital twin concept can act as a key enabler of this new engineering paradigm.