Applying Model-Driven Engineering to the Development of Smart Cyber-Physical Systems

Benoit Combemale and Jean-Michel Bruel
University of Toulouse, IRIT CNRS Lab, France

Key words: Model-Driven Engineering, Software Language Engineering, Model Composition

Abstract

Various disciplines use models for different purposes. An engineering model, including a software engineering model, is often developed to guide the construction of a non-existent system. A scientific model is created to better understand an existing phenomenon (i.e., an already existing system or a physical phenomenon). An engineering model may incorporate scientific models to build a smart cyber-physical system (CPS) that requires an understanding of the surrounding environment to decide of the relevant adaptation to apply. Sustainability systems, i.e., smart CPS managing resource production, transport and consumption for the sake of sustainability (e.g., smart grid, city, farming system), are typical examples of smart CPS. Due to the inherent complex nature of sustainability that must delicately balance trade-offs between social, environmental, and economic concerns, modeling challenges abound for both the scientific and engineering disciplines. In this talk, we will present a vision that promotes a unique approach combining engineering and scientific models to enable informed decision on the basis of open and scientific knowledge, a broader engagement of society for addressing sustainability concerns, and incorporate those decisions in the control loop of smart CPS. We will introduce a research roadmap to support this vision that emphasizes the socio-technical benefits of modeling, especially in computational sciences.
Software is eating the world, including the increasing development of smart cyber-physical systems (CPS) which assist citizens and companies in their daily life and businesses (e.g., smart cities/farming/building, industry 4.0 and transportation systems). But, may the world drive software? Modern CPS require more than ever to integrate a proper understanding of the world in which they evolve in order to take informed and open decisions during their dynamic adaptations that act on the physical world.

Various disciplines use models for different purposes. Engineers, e.g., software engineers, use engineering models to represent the system to implement, and scientists, e.g., environmentalists, use scientific models to represent the complexity of the world to understand and reason over it for analysis purpose. While the former tries to integrate all the properties in between the various engineering involved in the development process, the latter use models to internalize all the possible externalities of any changes, and later perform trade-off analysis. With the advent of smart CPS, the combination of scientific and engineering models becomes essential, respectively for openly and freely involving massive open data and predictive models in the decision process (either for trade-off analysis or dynamic adaptation purposes), and engineering models to support the smart design and reconfiguration process of modern CPS. It urges to provide the relevant facilities to software engineers for integrating into the future CPS the various models existing from the scientific community, and thus to support informed decisions, a broader engagement of the various stakeholders (incl. scientists, decision makers and the general public), and dynamic adaptations with regards to the expected impact of the smart CPS. This is particularly true for smart CPS dedicated to resource and energy management (e.g., water and electricity) for the sake of sustainability, namely sustainability systems, i.e., dynamically adaptable resource and energy management systems that aim to improve the techno-economic, social, and environmental dimensions of sustainability. Sustainability issues have been primarily described by scientific models (e.g., mathematical models, incl. analysis and predictive models) that enable scientists to understand the impact of changes in one or more of these three dimensions of sustainability. Engineering models have been used by (software) developers to construct CPS that support various aspects of sustainability systems, such as ecosystem monitoring, power grid management, and climate-control in smart buildings. In this context, scientific models help promote understanding of sustainability concerns and evaluate alternatives, while engineering models support the development and runtime management of smart CPS in charge of the resources. As the complexity of these systems increases, many challenges are posed to the modeling community to support the continuous integration of the various models and data, to support trade-off analysis among various stakeholders, to make the analysis results more accessible, and to continuously feed the control loop of smart CPS.

In Figure 1, we introduce a conceptual framework, termed Sustainability Evaluation ExperienceR (SEER), whose objective is to support sustainability-centered decision making for its different types of stakeholders who have a broad range of views and interests. SEER will enable to explore and identify the key enabling capabilities that are needed by a software system to support sustainability, and to eventually provide a decision-making tool for sustainability, to enable broader engagement of the community (e.g., scientists, engineers, group of interests, policy makers, general public), to facilitate more informed decision-making through what-if/for scenarios, and, to directly use those decisions to drive the automatic and dynamic adaptation of smart CPS.

Figure 1 depicts the integrated approach where each model provides feedback to the other for addressing the aforementioned overall ambition. In this approach, engineering models (included into the Software) are dynamically adapted by interpreting the resulting impact on the sustainability system, while scientific models are impacted by the use of the engineering models in a control loop that continuously adapts the system to reflect the trade-offs and changes in priorities among the different sustainability dimensions (economic, social and environmental). Other stakeholders (e.g., individuals, community leaders, policy makers, industrial organizations) can select specific (personalized) views of sustainability to explore the impact of changes in social behavior, policies, and resource consumption.
SEER must provide facilities for the curation and monitoring of data sets and models and enable flexible (open) data and model integration, e.g., physical laws, scientific models, regulations and preferences, possibly coming from different technological foundations, abstractions, scale, technological spaces, and worldviews. This also includes the continuous, automated acquisition and analysis of new data sets, as well as automated export of data sets, scenarios and decisions. The main function is to support the generation of what-if scenarios to project the effects on the different sustainability dimensions, and support the evaluation of externalities, especially for non-rapidly renewable resources. Since the predictions are necessarily probabilistic, the system must be able to assess the uncertainty inherent in all its actions and provide suitable representations of uncertainty understandable by users. Next to generating what-if scenarios, the tool should be capable of generating suggestions on how to reach user specified goals including quantifiable impacts (what-for scenarios), and driving the dynamic adaptation of sustainability systems. These powerful services must be made accessible to the population at large, regardless of their individual situation, social status and level of education. Several foundational concepts are used by engineering modeling approaches to handle size and tame complexity: decomposition and separation of concerns. Tools and methods from Model-Driven Engineering (MDE) help to reduce the accidental complexity associated with developing complex software-intensive systems [7]. A primary source of such accidental complexity is the gap between the high-level concepts used by domain experts to express their specific needs and the low-level abstractions provided by general-purpose programming languages [4, 5]. Manually bridging this gap is costly, both in time and effort. MDE approaches address this problem by automatically generating the major system artifacts from models. MDE models focus on how system functionality and domain-specific concepts are modeled relative to how their behavior may be specialized for the respective domain. Domain-specific modeling languages (DSML) provide a vocabulary and modeling primitives to facilitate experts in developing models describing the intended system behavior [5].

Based on previous experiments that demonstrate the usefulness, though with a minimum effort, of DSML for both engineering and scientific domains (see Figure 2), we review in the following the various dimensions of the underlying challenges for integrating models in SEER (for more details, we refer the reader to [6, 2]).

**Different foundations:** in traditional MDE approaches there are foundational notions used in constructing models, such as hierarchy, containment or reference coming from object orientations; different notions are used in other modeling spaces (e.g., multi-scales or multi-view) and the integration process must somehow acknowledge and align these different notions.

**Different abstractions/layers/granularity:** integrating models involves more than just establishing consistent vocabulary. The disparate models - created by stakeholder groups with a range of domain knowledge - will use different abstractions (e.g., patterns specific to the type of model), different layers and layering structures (e.g., networking layers versus atmospheric chemistry model layers), and different forms of granularity (e.g., a grid of rainfall observations over a large area at a single point in time, versus a series of measurements of cumulative water flow at one location in a river).

**Different technological spaces:** models may be constructed using different technologies from different technological spaces (e.g., databases, grammars, ontologies, ODEs, logics, structured natural languages, metamodels), which make different assumptions about: the basic building blocks of modeling; how those building blocks can be composed; how the well-formedness of the resultant models can be established; and how the well-formed models can be manipulated.

**Different worldviews:** a model is constructed with its own (often implicit) worldview. Model integration requires integration of these views, but they can contradict. Multiple contradictory worldviews need to be made explicit and embraced.
References


