Towards Adaptive Enterprises

A Generic Architecture and Its Model-Based Realization

TATA CONSULTANCY SERVICES

Vinay Kulkarni

IN BRIEF

Adapting enterprises suitably and effectively in increasingly dynamic environments remains a time-, effort- and intellectually intensive endeavor. To address this critical need, we propose a knowledge-guided, data-driven, simulation-aided, model-based, and evidence-backed approach to dynamic adaptation of a complex system of systems.

Executable modeling will be necessary to arrive at the right analysis and simulation models for the enterprise and model-driven engineering (MDE) will be used to transform them into a software implementation. As pioneers in bringing software product line ideas to MDE, we are extending the present capabilities of our model-driven engineering platform to support rigorous analysis and simulation. Simulatable nature of these models enables "in silico" validation of the proposed change, rather, a set of relevant changes. As these models are amenable to model-based code generation, the changes can be automatically reflected into implementation.

Research motivation

Modern enterprises are complex systems of systems operating in highly dynamic environments that need to respond quickly to a variety of change drivers. Determining the right response often requires a deep understanding of various organizational aspects such as goals, structure, business-as-usual operational processes, and more. The large size of the organization, its socio-technical characteristics, and fast business dynamics make this a challenging endeavor. The current industry practice of relying principally on human expertise for arriving at a suitable response has turned out to be inadequate.

Several enterprise modeling languages have been proposed to aid human-in-the-loop decision-making. However, none seem to address the need suitably. Moreover, the desired response needs to be implemented through suitable modifications to enterprise IT systems, business processes, and strategies. Little help exists to identify *what*

Several enterprise modeling languages have been proposed to aid human-in-the-loop decision making. However, none seem to address the need suitably.

Fact File

TCS Research: Software Systems & Services

Outcomes: Pertaining only to the article: Enterprise modeling & simulation platform, Automated compliance checking platform, Ontology mining platform, Actor-based simulation language

Principal Investigators: Vinay Kulkarni

Academic Partners: Middlesex University London UK, Aston University Birmingham UK

Techniques used: Executable models, Meta modeling, Model transformation, Logic, Natural Language Processing, Machine Learning, Text mining, and Schema integration

Industries benefited: All industries that rely heavily on computing & software systems

Patents: 4

Papers: 40

needs to be modified *where* and *how*. Effective and efficient implementation demands precise understanding of *what* needs to change *where* and *how*.

Having identified the desired set of modifications, technology exists to introduce these changes albeit in a widely varying spectrum of efficacy. For instance, modern IT systems are relatively more amenable to such modifications than legacy systems. However, adapting enterprises suitably and effectively in increasingly dynamic environments remains a time-, effort- and intellectually intensive endeavor. To address this critical need, we propose a knowledge-guided, data-driven, simulation-aided, model-based, and evidence-backed approach to dynamic adaptation of complex system of systems. Though it is being validated in spaces where the mechanistic worldview holds, we believe that the approach, when augmented with human behavioral modeling, will work equally well for socio-techno-economic spaces too.

The proposed line of attack

Our line of attack draws its inspiration from a well-studied and widely used concept from control theory—Model Reference Adaptive Control (MRAC) shown in Figure 1. The *Model* captures the desired reference behavior of the enterprise. The *Enterprise* is the complex system of systems to be controlled—viewed principally as input-output transfer function. The *Monitoring & Sense Making* component constitutes primordial technology infrastructure to observe and discern the current (as well as reference) output of enterprise. The *Controller* component constitutes primordial technology infrastructure that, together with the *Monitoring & Sense Making* component, nudges the *Enterprise* as close to the *Model* as possible, thus achieving a model-based, data-driven, justification-based, and human-in-the-loop adaptive response.

We also borrow from the idea of the digital twin that is gaining ground,

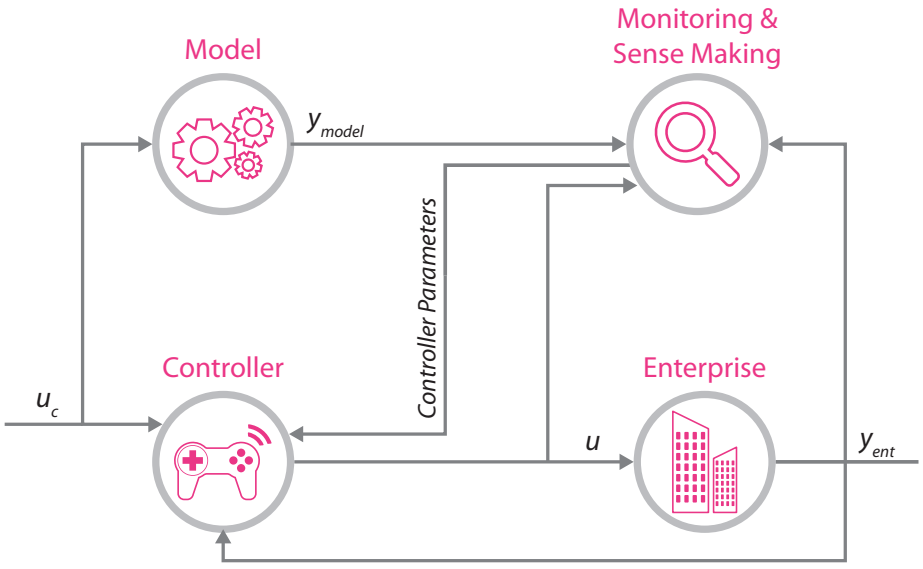

Figure 1: *Model-driven adaptive enterprise*

especially in the manufacturing domain. A digital twin mimics the behavior of a system in order to support what-if analyses and to arrive at appropriate responses to various contingencies that may arise in a plant. Therefore, a digital twin of an enterprise is best viewed as a specification capturing the aspects of *why*, *what*, *how,* and *when* in a formal machine form that can be easily manipulated. Thus, it is capable of modeling the relevant aspects of an enterprise to support *run-the-enterprise* as well as *change-the-enterprise* activities. Future enterprises are systems of systems with complex interactions operating in a dynamic environment. Given the structural and behavioral complexity, detailed understanding is possible only in localized contexts. At the same time, events occurring in one context influence the outcomes in others. Lack of complete information, coupled with inherent uncertainty, makes holistic analysis of systems intractable. As a result, decisions pertaining to system design and implementation are unlikely to be globally optimal. Non-availability of complete information and inherent uncertainty make traditional optimization approaches impractical. Therefore, simulation-based approach is often the only recourse available for arriving at a "good enough" solution by navigating the design space [1]. However, considering the open nature of the problem space, an exhaustive navigation of the design space is infeasible. It calls for intelligent navigation of the design space guided by domain knowledge and learning from past experience. Figure 1 provides a pictorial description of a possible line of attack that hinges on: (i) a layered view of enterprise; (ii) model-based machinery to help arrive at the right systems; (iii) model-based machinery to help implement the system correctly; and (iv) a mechanism to map the two sets of models and means to derive one from the other.

The architecture of Figure 2 supports realization of a model reference adaptive enterprise. Enterprise Digital Twin is the model of MRAC. It is a set of analyzable models that can be easily simulated to help arrive at the

right responses through scenario playing. A number of approaches can help achieve the desired architecture: *natural language processing* (*NLP*) *and machine learning* can be used to process enterprise data, information, and knowledge to construct the desired twin. *Executable modeling* will be necessary to arrive at the right analysis and simulation models for the enterprise, and *model-driven engineering* will be used to transform them into a software implementation. *System adaptation* will be used to capture data about system execution and interactions among sub-systems, and subsequent *data mining* will be used to derive suitable insights to refine software system models, which serve as the basis for *software system re-engineering*. The traces can also be mined to derive insights to refine the digital twin itself. Thus, the proposed architecture supports not only runtime adaptation but also design space exploration.

Enterprise digital twin

Enterprise digital twin is a set of purposive analyzable and simulation-ready models representing the enterprise in order to mimic real-world phenomenon. A variety of enterprise modeling (EM) languages exist that provide information-capture and analysis support across a wide spectrum of sophistication. A majority of these languages can be traced to the Zachman Framework [2] advocating that the capture of the *why*, *what*, *how*, *who*, *when*, and *where* aspects leads to the necessary and sufficient information for addressing a given problem. Thus, it can be argued that the complete specification of an enterprise is possible using the Zachman Framework. However, there exists no support for automated analysis as the

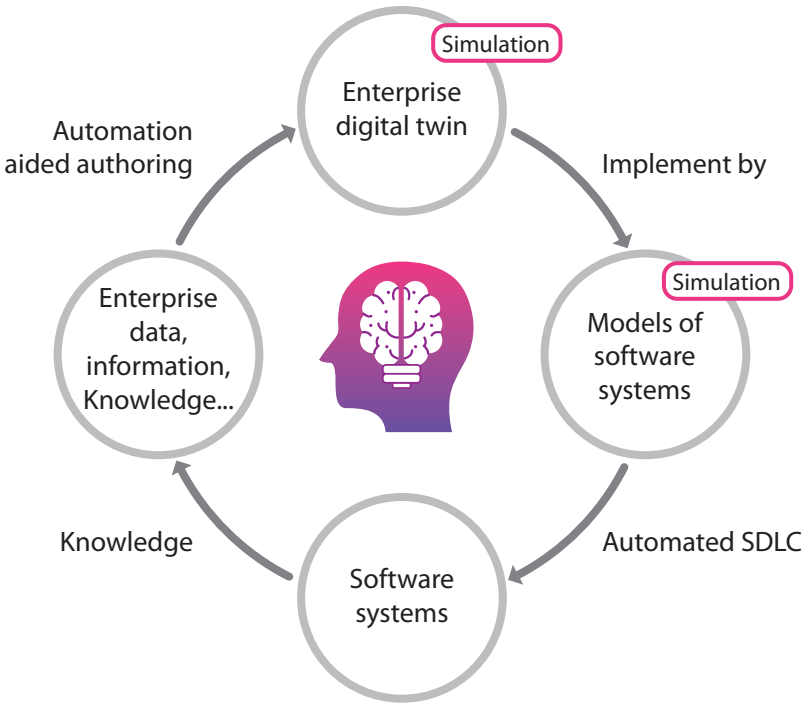

Figure 2: *Model-based architecture for adaptive enterprise*

information is captured typically in the form of texts and pictures. It can be observed in the existing EM languages that the languages capable of specifying all the relevant aspects of the enterprise for organizational decision-making lack support for automated analysis [2, 3, 4], and the languages capable of automated analysis can cater to only a specific subset of the aspects required for decision-making [5, 6, 7]. Moreover, the system of systems nature and the large size of modern enterprises mean that understanding of enterprise—structural and behavioral—is available only locally from which the overall behavior needs to be derived. Even the local understanding can have an element of uncertainty.

We have developed an actor-based modeling language, ESL, to specify enterprise as a set of autonomous encapsulating units that interact with each other by exchanging messages [8]. We have developed a component abstraction to model the system of systems nature of modern enterprises [9]. Our unique contribution is to enable modeling of uncertainty using probability [10]. Through a set of case studies, we have validated that ESL is necessary and sufficient to specify the complex system of systems for decision-making [11, 12].

Accelerating the creation of digital twin models

The lack of knowhow required to create the digital twin models is a serious concern. These models are typically large with the required information spread across the breadth of an enterprise in various forms, such as databases, execution logs, standard operational procedure notes, policy documents, and so on. As a result, an army of experts from across fields of expertise is required to manually create the digital twin models using appropriate model editors—clearly a non-workable proposition. We

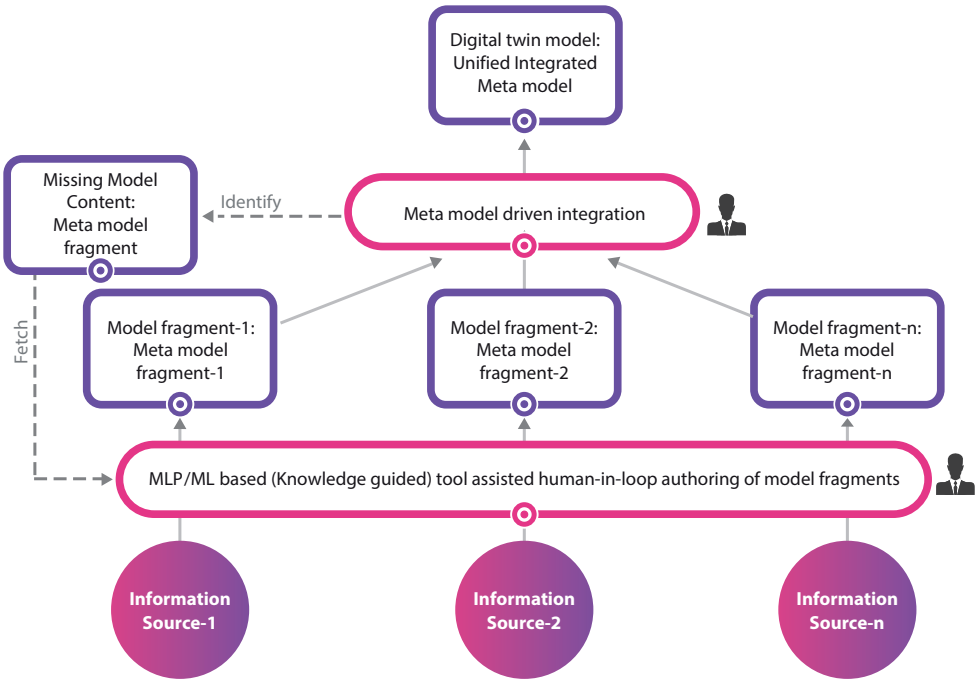

Figure 3: *Accelerated creation of digital twin models*

We have come up with a human-in-control method that stitches together the various automation aids to support accelerated creation of enterprise digital twin models.

have developed a framework for accelerated creation of digital twin models from information available in semi-structured, unstructured, and structured forms. It comprises automation aids based on: (i) NLP or machine learning techniques for gathering the desired information from a given information source; (ii) metamodel-driven techniques for integration and reconciliation of the model fragments; and (iii) model validation-based techniques for identifying the missing model fragments. We have come up with a human-in-control method that stitches together the various automation aids to support accelerated creation of enterprise digital twin models as shown in Figure 3.

We provide two ways of validating the digital twin models: (i) certification of correctness by experts, and (ii) through simulation wherein the models (initialized suitably) are subjected to known past events leading to simulation trace, which is then examined to ascertain whether the results are identical to the ones from the past. Our simulation engine generates a rich execution trace containing all the relevant information. We have developed a pattern language to specify the desired behavior and a pattern-matching engine to look for the specified patterns in the simulation trace [13]. This generic solution to ascertain correctness can be further augmented by manual validation of the input, output, and control variables of the simulation. This, we believe, should cover a wide range of digital twin models.

Simulation-aided design/ decision space exploration

The pervasiveness of computing in businesses has led to the availability of large volumes of data with high variability and volatility. This enables the use of rich analytic techniques to derive insights using a human-in-the-loop process. However, the recommendations could be suboptimal or even incorrect, as they are based on the past only, i.e., information regarding "possible situations" is totally missing. We use what-if simulations on the digital twin models to augment information pertaining to "what can happen" to the information pertaining to "what has happened" as shown in Figure 4.

The availability of more complete datasets for analysis leads to better recommendations. Moreover, we check efficacy of the proposed recommendation "in silico" through simulation of the digital twin models thus helping improve the "observe ⇒ analyze ⇒ recommend ⇒ implement" process in terms of time and cost. Ideally, analysis of "all possible situations" will lead to the best recommendation. However, given the open-ended nature of typical industrial problems, it is practically impossible to obtain the knowledge of "all possible situations" through simulation. Some guidance to navigate the design/decision space so as to arrive at sufficient knowledge of "possible situations" emerges as a critical need. We are experimenting with various AI techniques to meet this requirement.

Models of software systems

The analysis of enterprise digital twin models produces recommendations to be implemented in the enterprise or to be precise in the part of the enterprise supported by software systems This calls for an abstract view of the enterprise as a complex system of systems wherein the various layers such as strategy, processes, and IT systems are

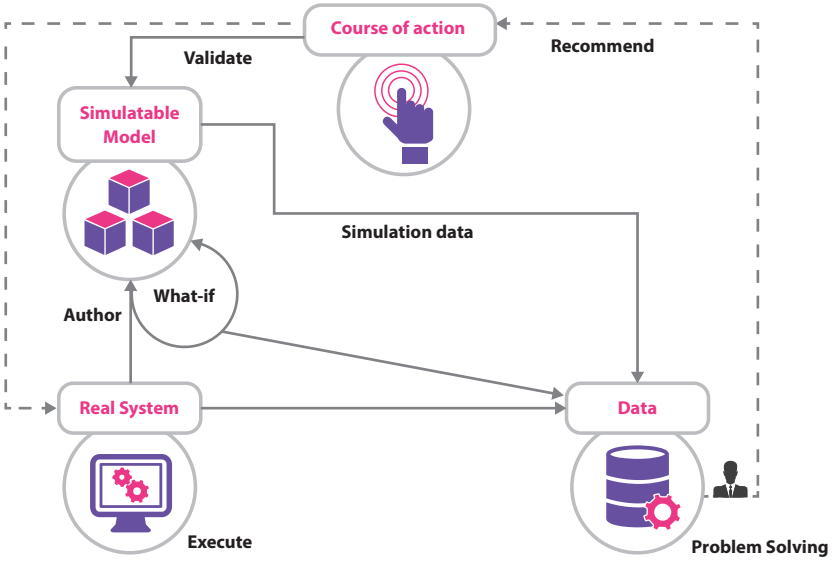

Figure 4: *Simulation-aided design/decision space exploration*

separate, and the interdependencies across and within these layers clearly brought out. We use popular EM languages such as ArchiMate for specifying this abstract view. Each of these layers needs to be further detailed out using specific modeling languages, for instance, a business process model and notation (BPMN) for the process layer, a unified modeling language (UML) for the systems layer, and so on. However, some of these modelling languages do not provide rich enough simulation support. Therefore, there is a need to come up with suitably rich languages amenable to simulation. This, in total, constitutes the necessary and sufficient specification machinery to model enterprises supported by software systems.

At present, we are relying on human experts to identify the footprint of a recommendation in terms of these models, but some sort of automation support is also necessary. The nature of these models is such that they can be easily simulated, and enables "in silico" validation of the proposed change, or rather, a set of relevant changes. As these models are amenable to model-based code generation, the changes can be automatically reflected into implementation. Change-driven development ensures the cost of implementing a recommendation into software system infrastructure supporting enterprise is minimal.

We already have much of the specification infrastructure required [14]. It will be augmented with additional modeling infrastructure as we address concrete use cases.

Model-based development and integration of software systems

Modeling helps address software development and integration from a higher level of abstraction. The most prominent benefits delivered so far have been through automated code generation. The software system is specified in terms of the *what* and *how* aspects leaving out details such as architectural decisions, design strategies, and implementation technology platforms. A set of

Change-driven development ensures the cost of implementing a recommendation into the software system infrastructure supporting enterprise is minimal.

appropriate code generators transform these specifications into a platform-specific implementation while adding details pertaining to the chosen architectural decisions and design strategies. The same specification can be used to deliver another solution with different choices for architectural decisions, design strategies, and implementation technology platform. Model-driven software development has delivered on the promises of platform independence, enhanced productivity, and uniformly high code quality [15]. We are one of the pioneers in the development of core technology for model-driven software development and using it to deliver several business-critical software systems across a wide range of technology platforms worldwide [16]. In MasterCraft,[1] we use proven MDE technology to cater to the needs of the implementation space. Much of this research has also found a way into core international MDE standards.[2, 3]

We are one of the pioneers in bringing software product line ideas to MDE through support for variability modelling and resolution at model level [17, 18]. As a result, it is possible to manage over 70 software systems delivered so far as a few domain-specific product families. Another illustration is available in the form of our model-based code generators, which are generated from their model-based specifications [19].

We are extending the present capabilities of our model-driven engineering platform to support rigorous analysis and simulation. We intend to use model checking techniques for the former and a clutch of simulation languages for the latter. Given the diverse nature of modeling languages, we are investigating if techniques such as cosimulation might be useful.

We have developed an actor-based language for specifying a complex system of systems as a set of interacting autonomous actors [8]. Though primarily designed as a specification of intentional modular reactive systems that is amenable to simulation and characterized by uncertainty, it can also be used for eliciting functional requirements. Since it is amenable to simulation, it can help with quick finalization of requirements. Moreover, the language can be easily transformed into the existing higher level specification languages. Thus, it seems a good choice for implementing recommendations.

Adaptive software systems

Software systems should reflect the characteristics of the enterprises they support. As enterprises are evolving into complex system of systems that need to quickly adapt to a variety of changes taking place in the operating environment, so should their supporting software systems. This puts new requirements of dynamic adaptation on enterprise software systems. Though self-adaptation is desired, auditory and similar requirements would demand human-in-the-loop adaptation.

In general, sense-n-respond architecture seems to have called for the Monitor-Analyze-Plan-Execute over a shared Knowledge (MAPE-K) architecture pattern; and this has been much in discussion as a solution, although it has not seen commensurate adoption in the industry [20]. The pervasiveness

1 https://mastercraft.tcs.com/

2 http://www.omg.org/spec/QVT/1.3

3 http://www.omg.org/spec/MOFM2T/1.0/PDF

of computing in general and advance of IoT in particular is expected to address the "sense" part adequately albeit at the finest level of granularity. There is a need for "sense-making" machinery to discern whether a higher level "domain event" has occurred or not. We intend to build further upon existing complex event recognition techniques to meet this need. A look-up table or a rule base may suffice in meeting rudimentary adaptation needs, but mechanisms to populate the look-up table with candidate adaptation responses would still be required. The responses can be identified either through simulation or learnt from past data using artificial intelligence (AI) techniques. We have developed core machinery for the former and intend to use it in tandem with learning techniques for the latter. We are also working on a simulation-aided learning architecture for dynamic adaptation. With cloud emerging as the preferred computing platform, we are keen on cloud-based implementation of the architecture.

Realizing Model-based Architecture through Possible Use Cases

We discuss a few use cases for the machinery outlined. Each of them illustrates a partial realization of a part of the vision depicted in Figure 2.

Digital manufacturing

Manufacturing is expected to be more and more customer-centric to the extent that the product gets manufactured per the specifications provided by customers. As a result, manufacturing supply chains need to be more dynamic, wherein the production, scheduling, and inventory-control processes need to be significantly more

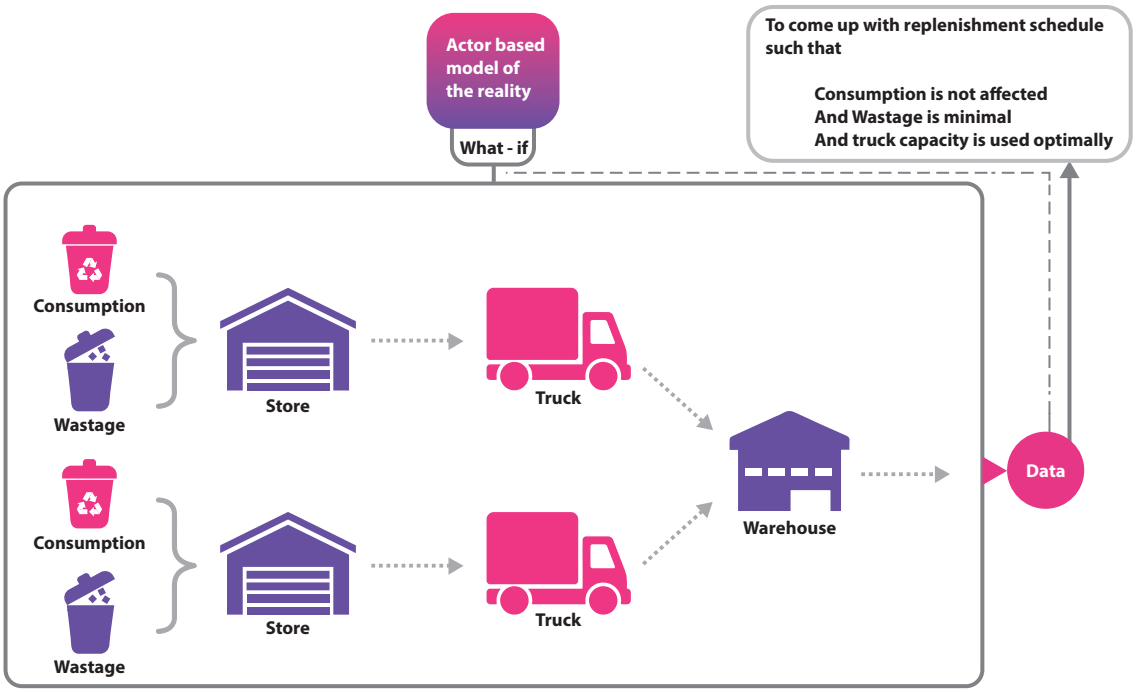

Figure 5: *Adaptive stock replenishment*

responsive. For instance, the mobile handset industry has seen several disruptions in the last decade—from feature phones to touchscreens to smartphones capable of high quality video experience. The ability to rapidly adapt to changing market demands was essential for survival.[4] A digital manufacturing organization would model the production processes, supply chains, the market, and the competition among other factors, and use these models individually and together to play out the scenarios of interest so as to devise suitable interventions. This helps the digital manufacturing organization to be more responsive with a degree of control on the certainty of outcomes [21, 22].

Figure 5 illustrates a small subset of store replenishment problems we have addressed in real-life context. The distribution center caters to replenishment of a set of stores at a regular frequency of, for instance, every 6 hours. Each store houses a fixed set of items, some of which are perishable, and has a known consumption pattern for each of them. A reinforcement learning-based algorithm is used to come up with replenishment orders for each store while meeting a set of constraints such as: (i) no item should go out of stock; (ii) no shelf should look empty; (iii) no shelf should display an item that is past its expiry date; and (iv) there should be no overstocking of any item at the store. The replenishment orders of all stores are combined at the distribution center to guide loading of trucks used for transporting the items to various stores. A model of the replenishment scenario is used for what-if scenario playing to generate data pertaining to "possible situations," leading to improvement in reinforcement learning algorithms. Moreover, corrective recommendations are validated "in silico" using simulation, thus leading to adaptive replenishment process.

Integrated computational materials engineering (ICME)

ICME is a new paradigm for designing products, materials, and manufacturing processes in an integrated manner [16]. A product's performance depends on the properties of the material it is made from. A material's properties in turn depend on the processes performed on it. Thus, there is a need to design the product, material, and manufacturing processes in an integrated manner. Traditional design approaches involve a lot of trial and error, and experimentation, leading to long cycle times and suboptimal design decisions. ICME proposes a modelling and simulation approach to explore the design space "in silico" in a systematic and efficient manner. However, the design space is usually so vast that exhaustive exploration is simply infeasible. To address this, the ICME platform draws heavily on domain knowledge and machine learning to guide the exploration process intelligently [23, 24]. The approach has been validated for problems, such as gear design, advanced high strength steel design [25].

Model-driven organization (MDO)

MDO is a new paradigm for modeling all relevant aspects of an enterprise so as to support run-the-enterprise as well as change-the-enterprise processes [26]. It brings the ideas of model reference adaptive control [27] to enterprise modeling as shown in Figure 6.

4 https://www.wired.com/2012/04/5-reasons-why-nokia-lost-its-handset-sales-lead-and-got-downgraded-to-junk

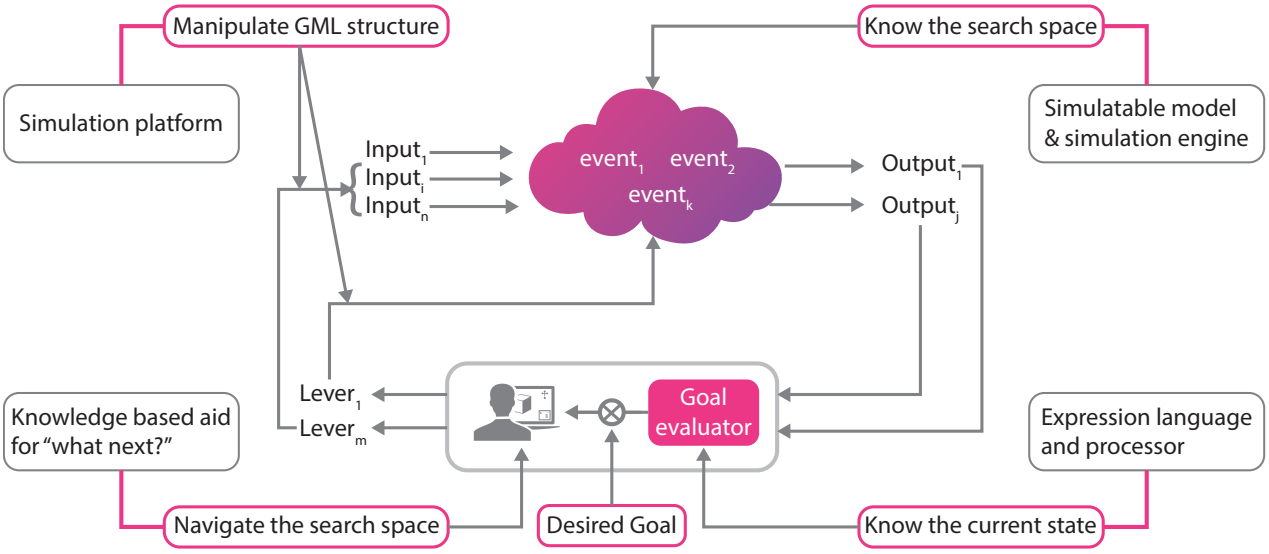

Figure 6: *Supporting adaptation under human supervision*

It enables modeling of an enterprise as a socio-technical system wherein the overall goal can be decomposed into subgoals, sub-subgoals, and so on, to the desired level of granularity. It identifies a set of variables (i.e., measures) that need to be observed in order to determine whether the finest level goal is met. It identifies a set of system parameters (i.e., levers) that influence a given measure, and the lever-measure influence is specified in a manner that can be processed by machines. The dependencies among levers, measures, and goals are also made explicit in a manner that can be processed by machines. Decision makers can now determine if the desired goal can be reached through application of the available levers, thus leading to a data-driven decision-making process. In a significant advance over the current state of practice, it is possible to provide evidence-based justification for the decision arrived at using what-if and if-what scenario playing [9]. It provides means for storing history, i.e., execution traces that can be mined for monitoring and sense-making [28] so that the domain expert can take meaningful adaptation decisions [29]. This approach has been validated on industry-critical problems in laboratory setting [11].

Having arrived at the right system models, they need to be realized through the right system implementations. Model-based software development approaches can be used for this purpose [30, 31, 32]. There is a need for a bidirectional mapping between system models and implementation models. Such a mapping will enable derivation of implementation models from system models and will also help feed back the insights extracted from data produced by implemented systems into the system models. Thus, the proposed modeling machinery not only helps at design time but also at run-time, facilitating a continuous adaptation loop. For problem-spaces that are well-defined and well-bounded, the proposed line of attack can support self-adaptation.

References

1 Gosavi A. Simulation-based optimization. parametric optimization techniques and reinforcement learning. 2003.

2 Zachman, J.A., 1987. A framework for information systems architecture. IBM systems journal, 26(3), pp. 276–292.

3 Krogstie, J., 2008. Using EEML for Combined Goal and Process Oriented Modeling: A Case Study,[w:] T. Halpin, J. Krogstie, E. Proper. In Proceedings of EMMSAD'08. Thirteenth International Workshop on Exploring Modeling Methods for Systems Analysis and Design.

4 Jonkers, H., Lankhorst, M., Van Buuren, R., Hoppenbrouwers, S., Bonsangue, M. and Van Der Torre, L., 2004. Concepts for modeling enterprise architectures. International Journal of Cooperative Information Systems, 13(03), pp. 257–287.

5 Yu, E., Strohmaier, M. and Deng, X., 2006, October. Exploring intentional modeling and analysis for enterprise architecture. In Enterprise Distributed Object Computing Conference Workshops, 2006. EDOCW'06. 10th IEEE International (pp. 32–32). IEEE.

6 Meadows, D. and Wright, D., 2008. Thinking in systems: A primer. Chelsea green publishing.

7 White, S, 2004. Introduction to BPMN. IBM Cooperation, 2(0).

8 Tony Clark, Vinay Kulkarni, Souvik Barat, Balbir Barn. ESL: An Actor-Based Platform for Developing Emergent Behaviour Organisation Simulations. PAAMS 2017: 311–315

9 Vinay Kulkarni, Tony Clark and Balbir Barn, 2014. A Component Abstraction for Localized, Composable, Machine Manipulable Enterprise Specification. BMSD Conference, 24–26 Jun 2014, Luxembourg.

10 Vinay Kulkarni, Souvik Barat, Tony Clark and Balbir Barn, 2017. Supporting Organisational Decision Making in Presence of Uncertainty. The European Symposium on Modeling and Simulation (EMSS 2017).

11 Souvik Barat, Vinay Kulkarni, Tony Clark, Balbir Barn, 2017. An actor-model based bottom-up simulation - An experiment on Indian demonetisation initiative. WSC 2017: 860–871.

12 Souvik Barat, Asha Rajbhoj, Prashant Kumar, Vinay Kulkarni, 2017. A Case Study Exploring Suitability of Bottom Up Modelling and Actor-based Simulation for Decision Making. DIAS/EDUDM@ISEC 2017.

13 Tony Clark, Balbir Barn, Vinay Kulkarni, Souvik Barat, 2017. Querying Histories of Organisation Simulations. ISD 2017.

14 Vinay Kulkarni, 2016. Model driven development of business applications: a practitioner's perspective. ICSE (Companion Volume) 2016: 260–269.

15 Hutchinson, John, Mark Rouncefield, and Jon Whittle. "Model-driven engineering practices in industry." In Proceedings of the 33rd International Conference on Software Engineering, pp. 633–642. ACM, 2011.

16 Vinay Kulkarni, Sreedhar Reddy, Asha Rajbhoj, 2010. Scaling Up Model Driven Engineering - Experience and Lessons Learnt. MoDELS (2) 2010: 331–345.

17 Vinay Kulkarni, 2010. Raising family is a good practice. FOSD 2010: 72–79.

18 Vinay Kulkarni, Souvik Barat, 2011. Business process families using model-driven techniques. IJBPIM 5(3): 204–217 (2011).

19 Vinay Kulkarni, Sreedhar Reddy, 2008. An abstraction for reusable MDD components: model-based generation of model-based code generators. GPCE 2008: 181–184

20 Arcaini, Paolo, Elvinia Riccobene, and Patrizia Scandurra. "Modeling and analyzing MAPE-K feedback loops for self-adaptation." In Proceedings of the 10th international symposium on software engineering for adaptive and self-managing systems, pp. 13–23. IEEE Press, 2015.

21 Kyle Cooper, Gautam Sardar, Jeffrey D. Tew, Erick Wikum: Reducing inventory cost for a medical device manufacturer using simulation. Winter Simulation Conference 2013: 2109–2115.

22 Kyle Cooper, Erick Wikum, Jeffrey D. Tew: Evaluating cost-to-serve for a retail supply chain. Winter Simulation Conference 2014: 1955–1964.

23 BP Gautham, S Reddy, P Das, C Malhotra: Facilitating ICME Through Platformization, Proceedings of the 4th World Congress on Integrated Computational Materials Engineering (ICME 2017): 93–102.

24 Prasenjit Das, Raghavendra Reddy Yeddula, Sreedhar Reddy: A Model Driven Framework for Integrated Computational Materials Engineering. ModSym+SAAAS@ ISEC 2016: 27–32.

25 BP Gautham, NH Kulkarni, JH Panchal, JK Allen, F Mistree: A method for the preliminary design of gears using a reduced number of American Gear Manufacturers Association (AGMA) correction factors Engineering Optimization 49 (4), 565–582.

26 Tony Clark, Vinay Kulkarni, Balbir Barn, Robert B. France, Ulrich Frank, Dan Turk: Towards the Model Driven Organization. HICSS 2014: 4817–4826.

27 Isermann R, Matko D, Lachmann KH. Adaptive control systems. Prentice-Hall, Inc.; 1992 Jan 1.

28 Tony Clark, Vinay Kulkarni, Souvik Barat, Balbir Barn: Actor Monitors for Adaptive Behaviour. ISEC 2017: 85–95.

29 Souvik Barat, Vinay Kulkarni, Tony Clark, Balbir Barn: A Model based Realisation of Actor Model to Conceptualise an Aid for Complex Dynamic Decision-making. MODELSWARD 2017: 605–616.

30 Selic B. The pragmatics of model-driven development. IEEE software. 2003 Sep; 20(5):19–25.

31 Hailpern B, Tarr P. Model-driven development: The good, the bad, and the ugly. IBM systems journal. 2006; 45(3): 451–61.

32 Kulkarni V, Reddy S. Separation of concerns in model-driven development. IEEE software. 2003 Sep; 20(5): 64–9.

Vinay Kulkarni

Vinay Kulkarni is a Chief Scientist and Head of Software Systems Research at Tata Consultancy Services (TCS). His research interests include model-driven software engineering, enterprise modelling and software engineering for uncertain world. His work in model-driven software engineering has led to a toolset that has been used to deliver several large business-critical systems over the past 20 years. Much of this work has found way into OMG standards, some of which Vinay contributed to in a leadership role. This work also received fair mention in respected international print media. He has several patents to his credit and has authored more than 100 papers in scholastic journals and conferences worldwide. He has served as the conference and program chairperson for the premier ACM and IEEE international conferences in the area of software engineering; and is on technical program committees of many international conferences. Recently, he was inducted as Fellow of Indian National Academy of Engineering. An alumnus of Indian Institute of Technology Madras, Vinay also serves as Visiting Professor at Middlesex University, London.