Simulation in Produktion
und Logistik 2021
Jörg Franke & Peter Schuderer (Hrsg.)
Cuvillier Verlag, Göttingen 2021

Von der Simulation zum Experimentierbaren Digitalen Zwilling und zurück

From Simulation to Experimentable Digital Twins and back

Michael Schluse, Jürgen Roßmann, RWTH Aachen University, Aachen (Germany),
schluse@mmi.rwth-aachen.de, rossmann@mmi.rwth-aachen.de

Abstract: This contribution proposes a new methodology allowing for the in-depth analysis of complete systems of Cyber-Physical Systems in their operational environment which is the basis for various simulation-based methods in engineering and operation. Key structuring element and semantic entity is the Experimentable Digital Twin (EDT) which describes a system and its interfaces from a Model-based Systems Engineering (MBSE) perspective and wraps Simulation Models or Experimentable Models describing its behavior. EDTs bridge the gap between MBSE and simulation, leveraging the potential of state-of-the-art modeling, simulation and co-simulation approaches in a comprehensive and easy-to-use workflow.

1 Einleitung

Technische Systeme entwickeln sich immer mehr zu komplexen **Systemen von Cyber-Physischen Systemen**. Dieser Trend ist in nahezu allen Anwendungsbereichen zu beobachten, auch und insbesondere in Produktion und Logistik, und wird aktuell z.B. durch Modularisierung, moderne Sensorik, das Internet der Dinge, Industrie 4.0-Konzepte und Cloud-/Edge-Computing verstärkt. Diesen Trend müssen aktuelle Entwicklungen in der Simulationstechnik spiegeln, z.B. indem Modularisierung und IT-technische Vernetzung von der physischen auf die virtuelle Ebene übertragen werden (u.a. Rosen et al. 2021). Zudem erfordern heute drängende Fragestellungen die **simulationsgestützte Betrachtung vollständiger Systeme**, um z.B. die vielfältigen Wechselwirkungen zwischen Systemkomponenten untersuchen und damit das Verhalten ganzer Systeme in unterschiedlichen Betriebssituationen analysieren, optimieren und funktional validieren zu können. Konzepte der Virtuellen Inbetriebnahme können so umfassend angewendet oder aussagekräftige Trainings- und Validierungsdaten für Maschinelles Lernen generiert werden. **Emergenz** ist hier von zentraler Bedeutung, denn aus dem Verhalten einzelner Systembestandteile kann meist nur in Grenzen auf das Verhalten des Systems geschlossen werden.

Technische Systeme müssen hierzu direkt in ihren Einsatzumgebungen in unterschiedlichen Einsatzsituationen mit einer großen Bandbreite an sinnfälligen

Eingangsstimuli umfassend simulationsgestützt untersucht werden. Dabei reicht es häufig nicht mehr aus, diese Untersuchungen nur auf höheren Skalenebenen, z.B. mittels Ereignisdiskreter Simulation, durchzuführen. Vielmehr muss auch auf Systemebene das Verhalten der beteiligten Teilsysteme detailliert betrachtet werden. Dies leisten zeitkontinuierliche Simulationsverfahren für Kinematik, Starrkörperdynamik oder Sensorik bis hin zu FEM-Methoden. Zudem müssen Datenverarbeitungs- und Kommunikationssysteme berücksichtigt werden. Grundlage einer Systemsimulation sind entsprechend **vielfach gekoppelte Multi-Skalen-Simulationsmodelle, deren Modellierung, Orchestrierung und Simulation neue Strukturen, Methoden und Prozesse erfordern**, in denen existierende Simulatoren anwendungsspezifisch miteinander gekoppelt und mit Hard-/Softwaresystemen und Menschen verbunden werden.

Wie aber können in verteilten Prozessen von unterschiedlichen Akteuren derartige Simulationen konzipiert, die notwendigen Simulationsmodelle erstellt, durch eine Kombination der am besten geeigneten Simulationsalgorithmen simuliert und das Ergebnis (mehrfach) genutzt werden? Dieser Beitrag schlägt hierfür geeignete Strukturen/Prozesse vor und untersucht hierzu die weitgehende Trennung der Modellierungsebenen durch Kombination von Model-based Systems Engineering (MBSE) mit etablierten simulationstechnischen Untersuchungen auf Komponentenebene. Ziel ist also eine Methodik, die mit bestehenden Mitteln umgesetzt werden kann.

Kapitel 2 fasst zunächst den Stand der Technik zur Systemsimulation zusammen. Kapitel 3 führt den Experimentierbaren Digitalen Zwilling (EDZ) ein und zeigt, auf welchen Ebenen mit diesem Systeme modelliert werden können. Kapitel 4 skizziert, wie EDZ System- und Simulationsbetrachtung zusammenführen. Kapitel 5 legt diesen Workflow gegen die Randbedingungen der Praxis. Dies führt zum in Kapitel 6 vorgestellten Workflow, in dem EDZ den Simulationsprozess strukturieren. Der Beitrag schließt in Kapitel 7 mit einer Zusammenfassung.

2 Stand der Technik: Modellierung auf Simulatorebene

Auch wenn Simulationen „immer komplexere und umfassendere Untersuchungen auch auf Systemebene" erlauben, ist der Stand der Technik aktuell stark durch **„detaillierte Analysemöglichkeiten auf Komponentenebene, jedoch limitierte Fähigkeiten auf Systemebene"** geprägt (Rosen et al. 2020). Zur Untersuchung von Systemen werden **Simulationsmodelle häufig entlang der Grenzen der jeweils eingesetzten Simulatoren entwickelt** und ggfls. anschließend miteinander gekoppelt. Alternativ kombinieren anwendungsspezifische Simulatoren ausgewählte Simulationsverfahren für vorab festgelegte Simulationsaufgaben. Leistungsfähige Simulationen auf Systemebene benötigen allerdings häufig die Zusammenführung der jeweils am besten geeigneten Simulatoren. Gleiches gilt, wenn Simulationen verteilt durch unterschiedliche Akteure (z.B. Abteilungen, Komponentenanbieter, Systemintegratoren) entwickelt werden. Entsprechend rückt Co-Simulation zur Integration von Teilmodellen in eine Simulation des Gesamtsystems nach Rosen et al. (2020)/Kuhn (2017) mehr und mehr in den Mittelpunkt. Gleichzeitig führt eine verteilte Werkzeuglandschaft heute zu diversen Medienbrüchen (Blumör et al. 2017).

Ausgangspunkt der Systemsimulation ist die Modellierung auf Systemebene. Hierzu steht eine Vielzahl von Sprachen und Ansätzen u.a. aus dem MBSE-Kontext zur Verfügung. SysML hat sich zu einem Standard entwickelt, Alternativen sind z.B. CAEX, AutomationML, das Referenzmodell zur Digitalen Fabrik nach IEC 62832 sowie die

Arbeiten zu IEEE P2806 und zur Verwaltungsschale. Auf Seite der Simulatoren ist die Situation gekennzeichnet durch herstellerspezifische Kombinationen aus Datenmodell/Datenformat für das Simulationsmodell und dem zugehörigen Simulator. Darüber hinaus existieren wenige übergreifende Ansätze wie Modelica oder SRML. Ein neutrales Austauschformat fehlt (Weigert et al. 2017). Systeme in einem einzigen Simulationsmodell vollständig zu beschreiben erscheint daher ausgeschlossen.

Deutlich besser stellt sich die Situation hinsichtlich des Austauschs Experimentierbarer Modelle dar (Gomes et al. 2017). Der De-Facto-Standard (Schweiger et al. 2019) ist hier das Functional Mockup Interface (FMI), daneben existiert eine Vielzahl herstellerspezifischer Lösungen. Im Bereich der verteilten Simulation haben sich die IEEE-Standards HLA und DIS etabliert. Allerdings sind Limitierungen zu beachten. So unterstützt z.B. FMI ausschließlich gerichtete Ein-/Ausgänge. Co-Simulationsansätze wurden bereits in vielfältiger Weise zur Realisierung von Simulationen auf Systemebene eingesetzt. Beispiele sind Scheifele et al. (2019) oder Härle et al. (2020). Letztere kombinieren ein Topologiemodell (AutomationML), Teilmodelle (FMU) und einen Orchestrator (OpenModelicaSimulator). Der Orchestrator führt die Teilmodelle zusammen, steuert den Zeitfortschritt und den Datenaustausch. Als Herausforderung wurden Modelle identifiziert, die sich über mehrere Module erstrecken. Scheifele et al. 2021 fassen unterschiedliche Ansätze und Vorteile zur Co-Simulation zusammen. Ein vielversprechender neuer Standard ist SSP (System Structure and Parameterization) zur Beschreibung von Systemen aus FMU.

Es gibt diverse Überlegungen zur Zusammenführung von detaillierten Simulationen und MBSE (z.B. mit SysML) und detaillierter Simulation (z.B. Kuebler et al. 2018; Kapos et al. 2019) oder zur Ableitung von Simulationsmodellen aus Systembeschreibungen (z.B. Scheifele 2019), die weitgehende Integration liegt allerdings noch in der Zukunft (INCOSE 2014). Dabei ist die hierdurch mögliche durchgängige modellbasierte Systementwicklung zentral (Stark et al. 2020). Als ein Ansatz erscheint das Konzept des Digitalen Zwillings (DZ), der MBSE in ein Digital Twin-based Systems Engineering überführen soll (Rosen et al. 2020). Herausforderung sind hier die vielen unterschiedlichen Sichten auf Begriff, Konzept, Nutzung und technische Umsetzung.

Im Ergebnis sind die Fragen, wie Systeme mit MBSE modelliert und Komponenten im Kontext von Simulation modelliert und simuliert werden, jeweils umfassend beantwortet. Die Zusammenführung bleibt aber immer noch eine Herausforderung. Dies gilt insbesondere, wenn unterschiedliche Modellierungskonzepte und -sprachen sowie Werkzeuge zusammentreffen. Ein umfassender methodischer und anwendungsübergreifender Ansatz zur Modellierung und Orchestrierung von Co-Simulationsanwendungen ausgehend von der Modellierung von Systemen existiert bislang nicht.

3 Grundidee: EDZ beschreiben Systeme und deren Interaktion

Zur übergreifenden Umsetzung von Simulation auf Systemebene sollte sich die Modellierung an der physikalischen Architektur des betrachteten Systems ausrichten. Im Mittelpunkt sollten Komponenten und Teilsysteme, deren mechanische, elektrische, hydraulische, thermale, IT-technische Schnittstellen und deren Verbindungen und Interaktion stehen – und nicht das einzelne Simulationssystem, dessen spezifische Abstraktion der realen Welt und das hierfür mit meist großem Aufwand

von Experten anwendungsspezifisch erstellte Simulationsmodell. Eine zentrale Frage ist jetzt, wie die entstehenden Modelle, Methoden und Prozesse beschrieben und zusammengeführt werden können. Das Konzept des **Digitalen Zwillings** liefert hier neue Ansätze. Ein Digitaler Zwilling im Sinne dieses Beitrags ist eine unter gewählten Gesichtspunkten betrachtete virtuelle digitale 1-zu-1-Repräsentation seines Realen Zwillings (RZ, insbesondere in Bezug auf Semantik, Struktur, Verhalten und Interaktion), die eine Interaktion auf Grundlage aktueller Digitaler Artefakte ermöglicht.

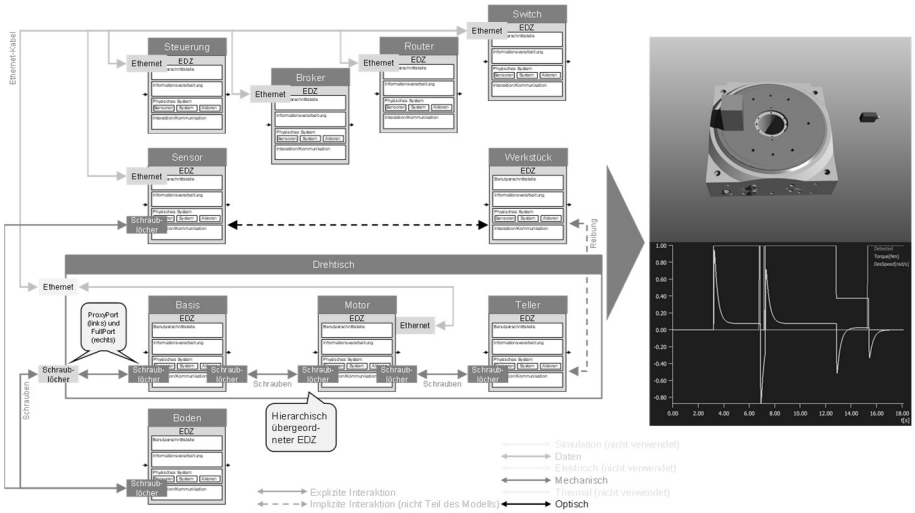

Abbildung 1: *Grundlage für das Experimentierbare Systemmodell des Drehtisch-Beispiels rechts ist die Modellierung der Anwendung auf Ebene des Systems Engineerings links als Netzwerk interagierender EDZ. Die Struktur der beteiligten Komponenten orientiert sich an der physikalischen Architektur des Systems, die „Verschaltung" der EDZ erfolgt entsprechend der Interaktion in der realen Welt.*

Simulation erweckt Digitale Zwillinge zum Leben, macht sie und ihr Verhalten erfahrbar und analysierbar. Aus Digitalen Zwillingen (DZ) werden **Experimentierbare Digitale Zwillinge** (EDZ), DZ zusammen mit aktueller Simulationstechnik bzw. DZ betrachtet aus der Perspektive der Simulationstechnik (Schluse et al. 2018). EDZ repräsentieren RZ, d.h. Physische, Informationstechnische, Mechatronische oder Cyber-Physische Systeme: **Ein EDZ ist eine virtuelle digitale 1-zu-1-Abbildung (Struktur, Verhalten, Interaktion) seines RZ in einem Experimentierbaren Modell.** EDZ/RZ interagieren miteinander in der virtuellen/realen Welt sowohl auf **physischer Ebene** durch Austausch von Energie und Materie (z.B. mechanisch, elektrisch, hydraulisch) als auch auf **informationstechnischer Ebene** durch Austausch von Nachrichten (z.B. über OPC UA, MQTT, Ethernet, CAN-Bus, serielle Schnittstelle). Aus Sicht von Modellierung und Simulation wird die Interaktion entweder **explizit** modelliert (z.B. Verschraubung oder Verkabelung) oder erfolgt **implizit** im Rahmen der Simulation (z.B. Kollision oder Detektion durch Sensor). Implizite Interaktionen sollten im Modell nicht formuliert werden; diese herzustellen ist Aufgabe der Simulation. Damit sind EDZ ideale Strukturierungselemente zur Beschreibung

von Systemen von Systemen. Die dargestellte Vorgehensweise konnte bereits für unterschiedliche Anwendungen in unterschiedlichen Domänen eingesetzt werden.

Die EDZ-Struktur wird durch Komponenten beschrieben, deren Schnittstellen durch Ports festgelegt sind und die durch Konnektoren verbunden sind (Abb. 2 links). Komponenten können selbst wieder EDZ sein. Auf oberster Hierarchieebene steht das EDZ-Szenario, welches ein konkretes Anwendungsszenario beschreibt. Die grundlegende Vorgehensweise/Darstellung orientiert sich an MBSE/SysML (Abb. 2 rechts). Die Empfehlung ist, die Modellierung an der physikalischen Architektur des Systems auszurichten. Dies senkt die Modellabstraktion, unterstützt die 1-zu-1-Beziehung zwischen EDZ und RZ und beantwortet hierdurch bereits eine Vielzahl von Fragen nach einer geeigneten Modellstruktur („Modelliere einfach so wie in der Realität!“). Dies erfolgt im Idealfall mit Ports und Konnektoren, die genauso auch in der realen Welt zu finden sind (Schraublöcher/Schrauben, Ethernet-Buchsen/Kabel etc.). Allerdings zeigt die Praxis, dass häufig auf Simulationsverfahren-spezifische Ports zurückgegriffen werden muss, z.B. zur Modellierung von Verbindungen für spezielle Kinematik-/Dynamik-Simulatoren oder bei Verwendung spezifischer Simulationssprachen wie Modelica. Schließlich benötigen Orchestratoren oft eigene Konstrukte zur Modellierung von Kommunikation (z.B. innerhalb der Komponenten auf funktionaler Ebene).

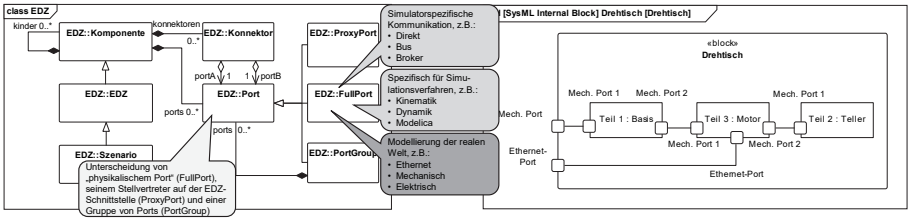

Abbildung 2: *Grundelemente des abstrakten Datenmodells für EDZ (links), SysML-Modell des Drehtisch-Beispiels (rechts)*

4 Konzept: EDZ führen System- und Simulationsbetrachtung zusammen

Auf dieser Grundlage führen EDZ System- und Simulationsbetrachtung zusammen (Abb. 3). Rechts unten ist der übliche Workflow der **Komponenten-/Simulator-spezifischen Modellierung und Simulation** dargestellt. Ausgehend von der (bestehenden oder neu zu entwickelnden) Realen Komponente wird ein *Konzeptuelles Komponentenmodell* aufgestellt, eine software-unabhängige Beschreibung eines Simulationsmodells, die dessen Ziele, Ein- und Ausgänge, Inhalte, Annahmen und Vereinfachungen beschreibt (Robinson 2008). Dieses wird in ein *Simulationsmodell* überführt, welches einem (meist Simulator-spezifischen) Meta-Modell folgt. Ein Simulator überführt das Simulationsmodell dann in ein *Experimentierbares Komponentenmodell*, welches ausführbar und damit Grundlage von Simulationsexperimenten ist und meist standardisiert (z.B. als FMU) zur Verfügung gestellt werden kann.

Zur Weiterverwendung dieser Ergebnisse auf einer Simulator-unabhängigen Systemebene werden die hier erzielten Ergebnisse in semantischen und insbesondere hinsichtlich ihrer Schnittstellen möglichst Simulator-unabhängigen Einheiten, den **EDZ**,

gekapselt (oben rechts). Deren Schnittstelle wird mit den Simulator-spezifischen Modellen „im Inneren“ verbunden, welche entweder als Experimentierbare Modelle oder als Simulationsmodelle zur Verfügung stehen. Aus Sicht objektorientierter Programmierung stellt der EDZ eine Fassade für die in seinem Inneren geeignet angeordneten Experimentierbaren Modelle/Simulationsmodelle dar und kapselt diese.

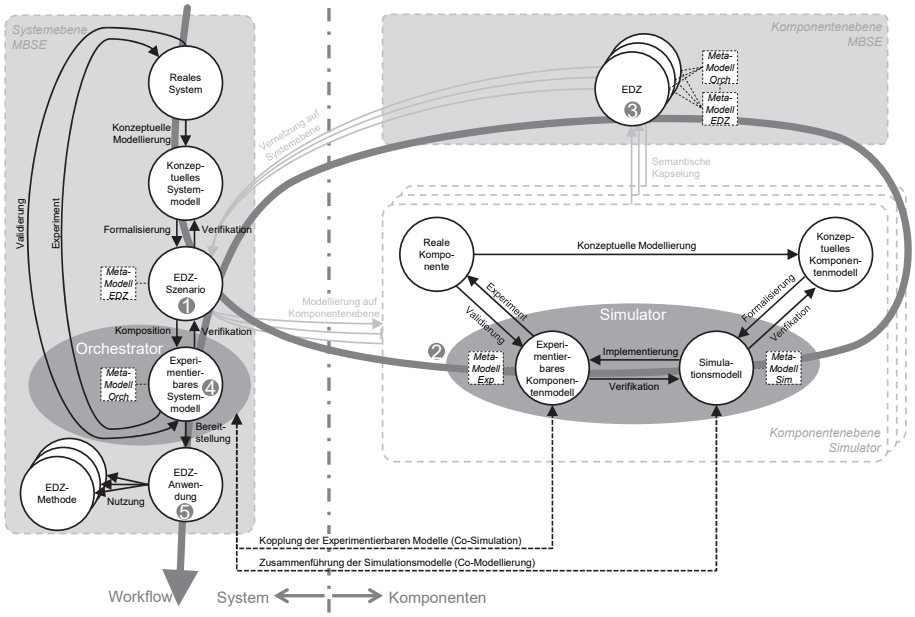

Abbildung 3: *Zusammenführung von Modellierung auf Systemebene und Modellierung/Simulation auf Komponentenebene durch EDZ*

Im Gegensatz hierzu sollte auf der **Systemebene** links Simulator-unabhängig gearbeitet werden. Im Idealfall werden etablierte MBSE-Methoden eingesetzt, die vollständig unabhängig von einer (später stattfindenden) Simulation und insbesondere unabhängig von einer (auf dieser Ebene möglichst zu vermeidenden) Simulationsabstraktion und einer Simulator-spezifischen Modellierung sind. MBSE und Simulation treffen sich auf Ebene der o.g. 1-zu-1-Repräsentation (hier Struktur/Interaktion). Das EDZ-Szenario legt fest, welche Systemkomponenten (d.h. welche EDZ) auf welchen Interaktionsebenen miteinander interagieren. Zur Erstellung des *Experimentierbaren Systemmodells* stellt das EDZ-Szenario den „Bauplan“ für dieses Modell zur Verfügung, entsprechend dem die über die EDZ bereitgestellten Experimentierbaren Modelle/Simulationsmodelle der Komponenten durch einen Orchestrator zusammengeführt und zur Verfügung gestellt werden. Die Komposition des Experimentierbaren Systemmodells erfolgt auf zwei Varianten. Einerseits können die EDZ Teil-Simulationsmodelle bereitstellen, die einem gemeinsamen Meta-Modell folgen. Diese werden zu einem Gesamt-Simulationsmodell zusammengeführt und durch einen Simulator in einem Experimentierbaren Modell implementiert (Co-Modellierung). Diese Zusammenführung erfolgt typischerweise entlang der Grenzen der Meta-Modelle, wobei theoretisch auch eine Überführung zwischen Modellen unterschiedlicher Meta-Modelle möglich wäre, wenn diese auf gleichen Meta-Meta-Modellen aufbauen

(Modelltransformation). Andererseits kann die Zusammenführung durch Kopplung der Experimentierbaren Komponentenmodelle erfolgen, die (weich (mehrere Solver) oder stark (ein Solver) miteinander gekoppelt) vernetzt werden (Co-Simulation).

Zur Umsetzung dieses Konzepts wird mindestens die **Interoperabilität** 1) der Simulatoren/Orchestratoren, 2) der Konzeptuellen Modelle (System/Komponenten), 3) der Meta-Modelle und 4) der Experimentierbaren Modelle/Simulationsmodelle der Komponenten/EDZ/EDZ-Szenarien benötigt. Zu deren Analyse haben u.a. Wang et al. (2009) im „Levels of Conceptual Interoperability Model (LCIM)" Ebenen definiert:

0. **Keine Verbindung**
1. **Technisch (Physisch):** Kommunikationsverbindung besteht
2. **Syntaktisch (Daten):** Daten können auf syntaktischer Ebene interpretiert werden
3. **Semantisch (Information):** Daten können semantisch zugeordnet werden
4. **Pragmatisch (Verwendung):** Gleiches Verständnis der Verwendung der Daten
5. **Dynamisch (Verhalten):** Gleiches Verständnis des Verhaltens
6. **Konzeptuell:** U.a. Randbedingungen, Einschränkungen, Validität bekannt

LCIM ≥ 1 wird für grundsätzliche Integrierbarkeit benötigt, LCIM ≥ 4 für Interoperabilität, LCIM = 6 für Kompositionsfähigkeit. Es muss aber unterschieden werden zwischen der grundsätzlichen „Erstellbarkeit" eines Experimentierbaren Systemmodells (Stufe 1), der Automatisierbarkeit dieses Prozesses (Stufe 2) und der Beurteilung der Validität des Ergebnisses vor dem Hintergrund des geplanten Einsatzes (Stufe 3). Für **Stufe 1** wird LCIM 3 benötigt, d.h. es müssen gemeinsame Kommunikationsprotokolle, Datenformate (z.B. Modelica, FMU) und Meta-Modelle (z.B. von Modelica oder FMU-Modellbeschreibungen) genutzt werden, anhand derer die Experimentierbaren Modelle/Simulationsmodelle in EDZ und diese dann in EDZ-Szenarien integriert werden können. Hierdurch sind die Grundlagen für die (ggfls. auch manuelle) Zusammenführung des Experimentierbaren Systemmodells gelegt. Der Verzicht auf die semantische Ebene erscheint hier nicht zeitgemäß. Für **Stufe 2** wird LCIM 4 benötigt. Hierdurch ist insbesondere eine Zuordnung zwischen den Ports der Teil-Modelle möglich (z.B. Ausgang „b" der Regler-FMU ist die Reglerausgangsgröße und Eingang „u" der Aktor-FMU ist der entsprechende Eingang). Für **Stufe 3** ist LCIM 6 notwendig, wobei ggfls. Anforderungen der LCIM 4 und 5 entfallen können.

5 Anwendung: Was bedeutet das in der Praxis?

LCIM 3 kann heute meist erreicht werden. Teilweise stehen Informationen zu den Experimentierbaren Modellen/Simulationsmodellen auf Ebene des Konzeptuellen Komponentenmodells zur Verfügung. LCIM 4 und 5 sind allerdings noch spärlich belegt. In der technischen Realisierung sollten und können die Meta-Modelle der EDZ/EDZ-Szenarien identisch ausgeführt werden. Die Kapselung von Experimentierbaren Komponentenmodellen in EDZ ist technisch einfach umzusetzen. Notwendig ist lediglich die Verbindung der EDZ-Ports mit den Experimentierbaren Komponentenmodellen. Hierzu stellt der Orchestrator eigene Ports/Konnektoren bereit (Abb. 4). Allerdings entsteht hierdurch eine direkte Abhängigkeit zwischen dem eingesetzten Orchestrator und der Modellierung der EDZ, die im Idealfall zu vermeiden ist. Hier besteht Bedarf einer Standardisierung.

Komplexer ist die Situation bei der Zusammenführung Simulator- und Komponenten-spezifischer Simulationsmodelle. Dies ist sinnvoll, um z.B. maximal performante und

stabile Kinematik- oder Starrkörpersimulationen zu erreichen oder eine komponentenübergreifende Sensorsimulation zu ermöglichen. Hierzu müssen Modellelemente auf EDZ-Ebene (z.B. starre Verbindungen zwischen EDZ) mit Modellelementen auf Ebene der Komponenten-Simulationsmodelle (z.B. Starrkörper) zusammengeführt werden. Notwendig ist hierfür eine allgemeine Sicht z.B. auf eine mechanische Verbindung, die dann verlustfrei auf unterschiedliche Simulatoren umgesetzt werden kann, deren Sichtweise auf mechanischen Verbindungen entspricht (z.B. gerichtete vs. ungerichtete Verbindungen) und gleichzeitig alle benötigten Parameter zur Verfügung stellt. Ansätze hierfür liefert z.B. AutomationML. Ein allgemeiner Ansatz ist aktuell nicht in Sicht, auch hier besteht Bedarf einer Standardisierung – wenn sie denn überhaupt möglich ist (vgl. z.B. die Anforderungen an die Beschreibung einer mechanischen Verbindung aus Sicht von Kinematik, Starrkörperdynamik und FEM). Ein Ausweg ist, Simulator-spezifische Verknüpfungen auch auf Ebene der EDZ zu erlauben. Dies kann häufig durch „Anwendungsprofile“ für etablierte EDZ-Meta-Modelle wie z.B. SysML oder AutomationML erreicht werden, führt allerdings eine starke Abhängigkeit zwischen den eigentlich Simulator-unabhängigen EDZ und den Simulatoren ein. Die auf semantischer Ebene im EDZ-Meta-Modell notwendigen Änderungen halten sich allerdings in Grenzen und führen meist nicht zu einem benötigten tiefen Simulations-Know-how auf Ebene der Systemmodellierung (siehe Abb. 4).

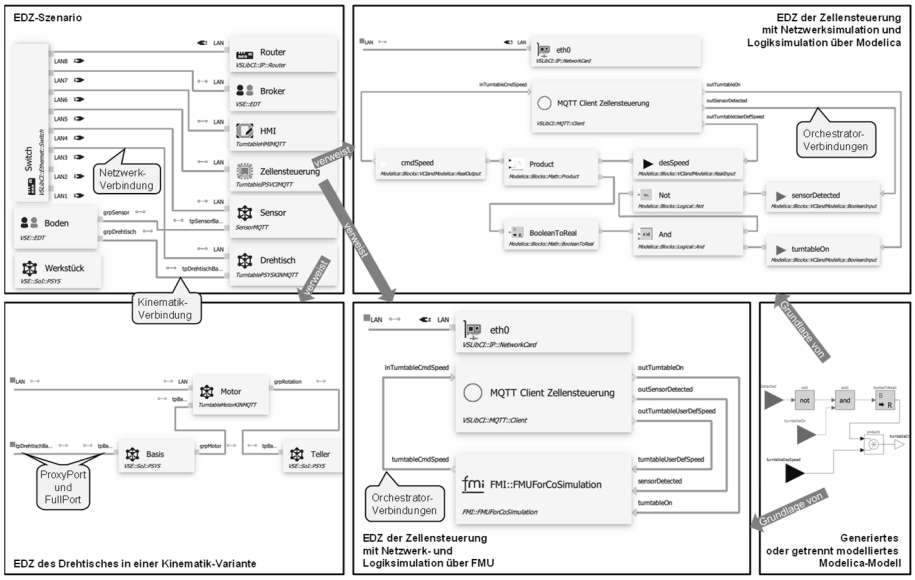

Abbildung 4: *Zusammenführung von Komponentenmodellen im EDZ am Beispiel: EDZ-Szenario (l.o.), EDZ des Drehtisches in Kinematik-Variante (l.u.), EDZ der Zellensteuerung modelliert über Modelica (r.o.) oder integriert als FMU (r.u.)*

6 Workflow: EDZ strukturieren den Simulationsprozess

Umgesetzt wird die skizzierte Grundidee in einem Workflow, der in fünf Stufen von der EDZ-basierten Systembeschreibung zur fertigen EDZ-basierten Simulationsanwendung führt (Blockpfeil in Abb. 3). In Schritt ① wird das EDZ-Szenario auf

übergeordneter (Gesamtsystem-) Ebene einschl. Beschreibung der Teilsysteme, deren Schnittstellen und deren Interaktion beschrieben. Diese Modellierung erfolgt im Idealfall in einem von Simulation unabhängigen Format wie z.B. SysML. Davon getrennt erfolgt in ② die Komponenten-/Simulator-spezifische Detailmodellierung. Das Ergebnis wird in ③ in EDZ zusammengefasst, die hierzu die erstellten Modelle beinhalten oder referenzieren. In ④ werden die jeweiligen Experimentierbaren Modelle/Simulationsmodelle automatisiert zusammengestellt, im Fall der Zusammenführung von Simulationsmodellen noch in Experimentierbare Modelle überführt und in standardisierten Simulationseinheiten, deren Struktur sich am FMI-Standard (Modelica 2014) orientiert, bereitgestellt. Auf dieser Grundlage wird ein Ausführungsplan abgeleitet, zur parallelen Ausführung optimiert und gekoppelt simuliert. Das Ergebnis wird dann in ⑤ in die EDZ-Anwendung integriert und so für unterschiedliche EDZ-gestützte Methoden wie Analyse, Test, Validierung, Optimierung, Inbetriebnahme, Training, Bedienung oder Steuerung genutzt.

7 Zusammenfassung

Im Ergebnis hat das dargestellte Konzept das Potenzial, die Komponenten-spezifische Modellierung auf Simulator-Ebene, die Systemmodellierung auf MBSE-Seite und den eingesetzten Orchestrator weitgehend und in Zukunft möglicherweise vollständig voneinander zu trennen und dennoch eine detaillierte Simulation auf Systemebene zu ermöglichen. Es legt weitere Grundlagen für die Überführung der Modularisierung und IT-technischen Vernetzung in die virtuelle Welt. Quasi als Nebeneffekt kann die Vernetzung über die Grenzen der virtuellen und realen Welten erfolgen, wodurch diese Welten im Sinne der Konvergenz weiter zusammenwachsen. Dies wurde z.B. für IP-basierte Kommunikation oder KNX-Bussysteme erfolgreich umgesetzt. Damit ist die Schleife vom EDZ zur Komponentensimulation und über den EDZ zurück zur Systemsimulation und zur EDZ-gestützten Anwendung konsistent geschlossen.

Der dargestellte Ansatz konnte erfolgreich in unterschiedlichen Anwendungsbereichen eingesetzt werden und hat sich darin als Grundlage für den Einsatz EDZ-gestützter Methoden bewährt. Zur praktischen Umsetzung heute muss das Konzept an den beiden beschriebenen Stellen aufgeweicht werden, durch die Abhängigkeiten zwischen Orchestrator-, Simulator- und MBSE-Meta-Modellen entstehen. Daher sind die Entscheidungen, welcher Orchestrator eingesetzt und welche Simulator-spezifischen Ports/Verbindungen eingesetzt werden sollen, elementar und zentral für den Aufbau einer EDZ-Bibliothek. Standardisierung im Kontext der Integration von Experimentierbaren Modellen/Simulationsmodellen in EDZ sowie im Bereich Simulator-spezifischer Ports/Verbindungen im EDZ-Szenario kann diese Abhängigkeit reduzieren und zu mehr Flexibilität führen, ist aber aktuell nicht in Sicht. Die Lücken bzgl. LCIM 4 und 5 verhindern eine vollautomatisierte Integration von Simulationsmodellen in EDZ, fehlende Informationen auf LCIM 6 erschweren insbesondere die Validierung des Experimentierbaren Systemmodells. Drei wichtige Empfehlungen sind, dass die Systemmodellierung sich an der physikalischen Systemarchitektur ausrichten, auf die Modellierung impliziter Interaktion verzichten und so weit wie möglich unabhängig von der späteren simulationstechnischen Umsetzung erfolgen soll.

Literatur

Blumör, A.; Pregitzer, G.; Bothen, M.: Werkzeuge für die Entwicklung mechatronischer Systeme mit Methoden des MBSE. In: Tag des Systems Engineering, Carl Hanser Verlag, 2017.

Gomes, C.; Thule, C.; Broman, D.; Larsen, P. G.; Vangheluwe, H.: Co-simulation: State of the art. http://arxiv.org/abs/1702.00686, 2017.

Härle, C.; Barth, M.; Fay, A.: Simulationsmodellgenerierung im modularen Maschinen- und Anlagenbau - Assistenzsystem zur automatischen Komposition und Konfiguration von Co-Simulationen. atp magazin 09 2020.

INCOSE: Systems Engineering Vision 2025. http://www.incose.org/docs/default-source/aboutse/se-vision-2025.pdf?sfvrsn=4, 2017.

Kapos, G.-D.; Tsadimas, A.; Kotronis, C.; Dalakas, V.; Nikolaidou, M.; Anagnostopoulos, D.: A Declarative Approach for Transforming SysML Models to Executable Simulation Models. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2019.

Kübler, K., Scheifele, S., Scheifele, C., & Riedel, O.: Model-Based Systems Engineering for Machine Tools and Production Systems (Model-Based Production Engineering). Procedia Manufacturing, 24, 216–221, 2018.

Kuhn, T.: Digitaler Zwilling. Informatik-Spektrum, 40(5), 2017.

Modelica Association Project “FMI.”: Functional Mock-up Interface for Model Exchange and Co-Simulation - version 2.0. 2014.

Rosen, R.; Jäkel, J.; Barth, M.; Stern, O.; Schmidt-Vollus, R.; Heinzerling, T.; … Röhler, M.: Simulation und digitaler Zwilling im Anlagenlebenszyklus - Standpunkte und Thesen. VDI-Statusreport Februar 2020.

Rosen, R.; Heinzerling, T.; Schmidt, P. P.; Jäkel, J.; Kübler, K.; Ristic, M.; … Stern, O. (2021): Die Rolle der Simulation im Kontext des Digitalen Zwillings. atp magazin 04 2021.

Scheifele, S.: Generierung des Digitalen Zwillings für den Sondermaschinenbau mit Losgröße 1. Fraunhofer Verlag, 2019.

Scheifele, C.; Härle, C.; Rosen, R.; Heinzerling, T.; Röhler, M.: Co-Simulation als Realisierung digitaler Zwillinge. atp magazin 04 2021.

Schluse, M.; Priggemeyer, M.; Atorf, L.; Rossmann, J.: Experimentable Digital Twins—Streamlining Simulation-Based Systems Engineering for Industry 4.0. IEEE Transactions on Industrial Informatics, 14(4), 2018.

Schweiger, G.; Gomes, C.; Engel, G.; Hafner, I.; Schoeggl, J.; Posch, A.; Nouidui, T.: An empirical survey on co-simulation: Promising standards, challenges and research needs. Simulation Modelling Practice and Theory, 95, 2019.

Stark, R.; Anderl, R.; Thoben, K.-D.; Wartzack, S.; Krause, F.-L.; Grässler, I., … Göckel, N.: WiGeP - Positionspapier: Digitaler Zwilling, 2020.

Weigert, D.; Aurich, P.; Reggelin, T.: Durchgehende Modellerstellung zwischen Simulations-, Visualisierungs- und Konstruktionswerkzeugen für die gesamtheitliche Planung von Produktions- und Intralogistiksystemen. In: S. Wenzel & T. Peter (Eds.), Simulation in Produktion und Logistik 2017.

Robinson, S.: Conceptual modelling for simulation Part I: Definition and requirements. Journal of the Operational Research Society, 59(3), 278–290, 2008.

Wang, W.; Tolk, A.; Wang, W.: The levels of conceptual interoperability model: Applying systems engineering principles to M&S. In: Spring Simulation Multiconference 2009