SNE SIMULATION NOTES EUROPE

Journal on Developments and Trends in Modelling and Simulation

EUROSIM Scientific Membership Journal

Vol. 34 No.4, December 2024 ISSN Online 2306-0271 DOI 10.11128/sne.34.4.1071
ISSN Print 2305-9974

ARGESIM

ASIM Fachtagung

CALL FOR PAPERS

24. bis 26. Sept.

2025

„Haus der Kirche“

Dresden

21. ASIM Fachtagung „Simulation in Produktion und Logistik“

Als größte europäische Tagung zum Thema Simulation im Bereich Produktion und Logistik gibt die ASIM Fachtagung alle zwei Jahre einen Überblick der zukunftsweisenden Trends, aktuellen Entwicklungen und erfolgreichen Projekte. Präsentiert und diskutiert werden wissenschaftliche Arbeiten sowie interessante Anwendungen aus der Industrie.

Themen der Tagung

Supply Chain Simulation & Logistik

- Produktionsnetzwerke & -logistik
- Intralogistik; Lieferketten
- Transport & Verkehr
- Innovative Materialflusstechnik
- Produktionsplanung & -steuerung
- Systemresilienz; (Cyber-)Sicherheit

Simulationsanwendungen in Industrie und Dienstleistung

- Automobilindustrie
- Maschinen- & Anlagenbau
- Halbleiterindustrie
- Simulation as a Service
- Energieeffizienz; Nachhaltigkeit

Methoden, Werkzeuge & Simulationstechnik

- Virtuelle Inbetriebnahme; Digitaler Zwilling; Digitaler Schatten
- Simulationsbasierte Optimierung
- Verifikation & Validierung
- Data Science; Visual Analytics; Virtual Reality; Augmented Reality
- Künstliche Intelligenz; Maschinelles Lernen
- Ereignisdiskrete Simulation; Agentenbasierte Simulation
- Interoperabilität; verteilte Simulation; Cloud-basierte Simulation

Einreichung von Beiträgen

Zweistufiges Verfahren

Anwender, Anbieter und Wissenschaftler werden zur Einreichung eines Exposés gebeten. Auf dessen Grundlage wird zur Ausarbeitung eines Full-Papers aufgefordert und begutachtet. Die Veröffentlichung erfolgt Open-Access mit persistenten Identifier (bspw. DOI).

Willkommen sind auch Beiträge ohne Full-Paper zur Präsentation auf der Tagung. Die Entscheidung zur Annahme erfolgt auf Basis des Exposés.

Termine

Einreichung Exposé (Full-Paper oder nur-Präsentation)	14.02.2025
Benachrichtigung Autoren	28.03.2025
Einreichung Full-Paper	16.05.2025
Annahme Full-Paper (nach Begutachtung)	20.06.2025
Vorlage druckreifes Manuskript	18.07.2025

Weitere Infos, Tagungssprache und Links

Die Einreichung erfolgt online via ConfTool in deutscher oder englischer Sprache. Pro Beitrag muss eine Person zur Tagung angemeldet und das Thema präsentiert werden. Alle Infos sind auf der Website zur Tagung zusammengefasst.

[https://asim-gi.org/spl2025]

GI | asim | ASIM Fachtagung | Institut für Technische Logistik und Arbeitssysteme | TECHNISCHE UNIVERSITÄT DRESDEN

Editorial

Dear Readers, This fourth issue of SNE Volume 34, 2024, SNE 34(4), surprises with its cover: a graphic picture of Lord Kelvin – but with a strong link to modelling and simulation, as the Overview Note 'Kelvin and Bush: Their Contributions to Today's System Modelling and Simulation Methods' explains. This year we celebrated Lord Kelvin's 200th anniversary, and David Murray-Smith, former president of UKSIM, prepared for SNE this note, reviewing the development from Lord Kelvin's proposal that a type of mechanical integrating device could be used directly to solve ordinary differential equations until Vannevar Bush's development of 'integraph' devices leading to the mechanical differential analyser. Murray-Smith concludes, that these early developments represent important historical milestones in simulation methods, and that many of the ideas and principles established by Kelvin and Bush remain valid today but their origins are seldom fully recognised.

Four contributions in this issue conclude the postconference publications from ASIM's STS/GMMS Magdeburg Workshop 2023, with the improved and English versions of the articles: C. Fink et al. on coupled simulation of the injection and combustion of a Diesel Engine, B. Wörrlein and S. Straßburger on dynamic time warping and synthetic data for validating Seq2Seq, S. Jacobitz et al. on a model library for the low-cost development of mechatronic systems, and X. Liu-Henke et al. on a holistic, highly flexible HiL test system for autonomous driving functions. And last but not least, the CEA group from university Wismar submitted a paper on simulation-based and vendor-independent multi-robot control development (B. Freymann et al.).

I would like to thank all authors for their contributions, especially David Murray-Smith for the Overview Note, and many thanks to the SNE Editorial Office for layout, typesetting, preparations for printing, electronic publishing, and much more. And have a look at the info on EUROSIM-related simulation events of the next year 2025: ASIM Conference on Simulation in Production and Logistics in Dresden, I3M 2025 conference in Fes, Morocco, WinterSim 2025 in Seattle, and further conferences and workshops of the EUROSIM societies.

Felix Breitenecker, SNE Editor-in-Chief, eic@sne-journal.org; felix.breitenecker@tuwien.ac.at

Contents SNE 34(4)

e-SNE 34(4), DOI 10.11128/sne.34.4.1071

www.sne-journal.org/sne-volumes/volume-34

SNE Basic e-Version with Open Access

SNE Full e-Version for Members of publication-active EUROSIM Societies: ASIM, CEA-SMSG, CSSS, DBSS, KA-SIM, LIOPHANT, NSSM, PTSK, SIMS, SLOSIM, UKSIM

print-SNE for members on demand (printer INTU TU Wien)

Cover: Lord Kelvin, grafic picture, ID 54758644
© Lukaves | *Dreamstime.com*

SNE Contact & Info

SNE Online ISSN 2306-0271, SNE Print ISSN 2305-9974

→ *www.sne-journal.org*

office@sne-journal.org, eic@sne-journal.org

SNE Editorial Office
Felix Breitenecker (Organisation, Author Mentoring)
Irmgard Husinsky (Web, Electronic Publishing),
ARGESIM/Math. Modelling & Simulation Group,
Inst. of Analysis and Scientific Computing, TU Wien
Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria

SNE SIMULATION NOTES EUROPE
WEB: → *www.sne-journal.org*, DOI prefix 10.11128/sne

Scope: Developments and trends in modelling and simulation in various areas and in application and theory; comparative studies and benchmarks (documentation of ARGESIM Benchmarks on modelling approaches and simulation implementations); modelling and simulation in and for education, simulation-based e-learning; society information and membership information for EUROSIM members (Federation of European Simulation Societies and Groups).

Editor-in-Chief: Felix Breitenecker, TU Wien, Math. Modelling Group
Felix.Breitenecker@tuwien.ac.at, *eic@sne-journal.org*

Print SNE: INTU (TU Wien), Wiedner Hauptstrasse 8-10, 1040, Vienna, Austria – *www.intu.at*

ARGESIM Publisher: ARBEITSGEMEINSCHAFT SIMULATION NEWS c/o Math. Modelling and Simulation Group, TU Wien / 101, Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria
www.argesim.org, *info@argesim.org*
on behalf of ASIM *www.asim-gi.org* and
EUROSIM → *www.eurosim.info*

SNE - Aims and Scope

Simulation Notes Europe (SNE) provides an international, high-quality forum for presentation of new ideas and approaches in simulation - from modelling to experiment analysis, from implementation to verification, from validation to identification, from numerics to visualisation (www.sne-journal.org).

SNE seeks to serve scientists, researchers, developers and users of the simulation process across a variety of theoretical and applied fields in pursuit of novel ideas in simulation. **SNE** follows the recent developments and trends of modelling and simulation in new and/or joining areas, as complex systems and big data. **SNE** puts special emphasis on the overall view in simulation, and on comparative investigations, as benchmarks and comparisons in methodology and application. For this purpose, **SNE** documents the **ARGESIM Benchmarks** on *Modelling Approaches and Simulation Implementations* with publication of definitions, solutions and discussions. **SNE** welcomes also contributions in education in/for/with simulation.

SNE is the scientific membership journal of **EUROSIM**, the *Federation of European Simulation Societies and Simulation Groups* (www.eurosim.info), also providing Postconference publication for events of the member societies. **SNE**, primarily an electronic journal **e-SNE** (ISSN 2306-0271), follows an open access strategy, with free download in basic version (B/W, low resolution graphics). Members of most **EUROSIM** societies are entitled to download **e-SNE** in an elaborate full version (colour, high resolution graphics), and to access additional sources of benchmark publications, model sources, etc. (via group login of the society), **print-SNE** (ISSN 2305-9974) is available for specific groups of **EUROSIM** societies.

SNE is published by **ARGESIM** (www.argesim.org) on mandate of **EUROSIM** and **ASIM** (www.asim-gi.org), the German simulation society. **SNE** is DOI indexed with prefix 10.11128.

Author's Info. Individual submissions of scientific papers are welcome, as well as post-conference publications of contributions from conferences of **EUROSIM** societies. **SNE** welcomes special issues, either dedicated to special areas and/or new developments, or on occasion of events as conferences and workshops with special emphasis.

Authors are invited to submit contributions which have not been published and have not being considered for publication elsewhere to the **SNE** Editorial Office.

SNE distinguishes different types of contributions (*Notes*), i.e.

- **TN** Technical Note, 6–10 p.
- **SN** Short Note, max. 5 p.
- **SW** Software Note , 4–6 p.
- **BN** Benchmark Note, 2–10 p.
- **ON** Overview Note – only upon invitation, up to 14 p.
- **EN** Education Note, 6–8 p.
- **PN** Project Note 6–8 p.
- **STN** Student Note, 4-6 p., on supervisor's recommendation
- **EBN** Educational Benchmark Note, 4–10 p.

Further info and templates (doc, tex) at **SNE**'s website, or from the Editor-in-Chief

www.sne-journal.org

office@sne-journal.org, *eic@sne-journal.org*

SNE Editorial Board

SNE - Simulation Notes Europe is advised and supervised by an international scientific editorial board. This board is taking care on peer reviewing of submission to SNE (and extended for special issues and Postconference publication):

Simulation-Based and Vendor-Independent Multi-Robot Control Development

Birger Freymann, Thorsten Pawletta*, David Jammer

Research Group Computational Engineering and Automation, University of Applied Sciences Wismar, Philipp-Müller-Straße 14, 23966 Wismar, Germany
*birger-freymann@hotmail.de, *thorsten.pawletta@hs-wismar.de, david.jammer@hs-wismar.de*

SNE 34(4), 2024, 185-194, DOI: 10.11128/sne.34.tn.10711
Submitted: 2024-06-28; Revised: 2024-07-30
Accepted: 2024-10-01
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. The Simulation-Based Control (SBC) approach and the Task-Oriented Control (TOC) development, which have already been investigated for individual articulated robots, is extended to teams of articulated robots, hereinafter also referred to as Multi-Robot System (MRS).
The basics of SBC, TOC and client-server approach for vendor-independent robot programming are introduced. Possible interactions in an MRS are then analyzed and classified, and a TOC description of interactions is developed. The TOC description is based on reusable, atomic tasks that can be aggregated sequentially and hierarchically via interfaces.
For three interaction classes, the TOC specification and its transformation into executable robot commands using a client-server based robot midleware is discussed. The control dynamics is described with the Discrete Event System Specification (DEVS) and represented using DEVS diagrams.

Introduction

Robots have been established in industry for decades as powerful and flexible tools. With new areas of application, as defined in the context of Industry 4.0, the requirements for the efficient development of robot controls are growing. The methods of Rapid Control Prototyping (RCP) can make a significant contribution here (Abel and Bollig [1]).

RCP requires a consistent process model, an end-to-end tool chain from the design phase through to operation, and the consistent use of modeling and simulation methods to continuously verify development steps. Almost all robot manufacturers offer RCP-based development environments. However, these are manufacturer-specific and only support their own robots.

The methodological and software differences make it difficult to set up Multi-Robot Systems (MRSs), consisting of robots from different manufacturers.

Another method for the efficient development of robot controls are high-level specifications, such as the Task-Oriented Control (TOC) approach (Siciliano [2]). The principle of TOC is to break down complex problems into a set of reusable, atomic tasks that can be aggregated sequentially or hierarchically via interfaces. To execute a TOC, the tasks must be translated into executable robot commands.

This paper proposes a vendor-independent RCP approach for robotic control development. We build on the Simulation-Based Control (SBC) approach introduced by Maletzki [3] for Single-Robot Systems (SRS) and extend its application to MRS.

The SBC approach corresponds to the RCP method and enables manufacturer-indepent end-to-end development from the control design phase through to the operational use. Furthermore, the SBC approach defines a framework for practical implementation. A simulation environment with real-time process interface and client-server based robot midleware form its core.

In contrast to an SRS, the interactions between robots must be described when controlling an MRS. For this purpose, interactions between articulated arm robots are analyzed and classified. We examine how the interactions can be specified and processed using reusable tasks. To do this, we define a case study and discuss for three interaction types the TOC specification and its transformation into executable robot commands.

We use a Discrete Event System Specification (DEVS) based modeling and simulation environment and describe the tasks and their transformation into robot code using DEVS diagrams. Details on the DEVS environment can be found in Freymann [4].

1 Fundamentals

In this section, the basics of the SBC and TOC approach are presented and some methods for manufacturer-independent control programming of MRS are discussed.

1.1 Simulation Based Control Approach

The SBC approach is a methodology for control development and defines a framework for practical control implementations (Maletzki et al. [5]).It supports a continuous simulation-based development process from the early specification of a control logic through to the operational use of a control system on the basis of an integrated tool chain. A schematic representation of the SBC approach is shown in Figure 1.

A Simulation Model (SM) developed in the design phase is extended step by step to a Control Software (CS) and continuously tested by simulation during the development process. This approach eliminates the need to reimplement simulation code in control code. In this way, errors are avoided, development time is saved and overall development costs are reduced. The consistent testing of development steps with SM makes it possible to identify and correct errors at an early stage.

As shown in Figure 1, it is necessary to consistently distinguish between the Control Model (CM) with the control logic and the Process Model (PM) with the image of the real process as early as possible during the development. The operational mode requires an interface to the real process, which records measured sensor values and conversely sends actuator commands to the components of the real process. For this reason, an Interface Model (IM) must be developed for the operational phase. This IM should support simulative testing with a robot simulation and at the same time act as an interface to the real process. In contrast to most other RCP approaches, the PM remains a component in the control system even in the operating mode. This enables the calculation of non-measurable or poorly measurable process variables and the realization of observer concepts.

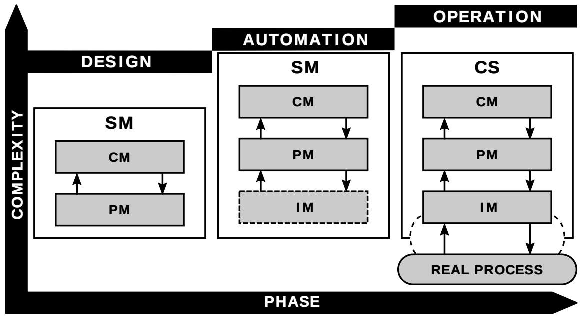


Figure 1: Schematic represenation of the SBC approach.

1.2 Task-oriented Control Design

TOC is an established concept for control design (Siciliano et al. [2]) and is used for programming SRS in many applications [3, 6]. The procedure for creating a TOC corresponds to the human way of thinking when solving complex problems.

The basic principle is the representation of control problems by a sequence of tasks. Tasks are usually independent work steps. The tasks should be formulated universally in order to be able to represent different control problems by linking tasks in different ways. Tasks can be linked purely sequentially, conditionally or in loops. Aggregated tasks can be created by links on one level or in hierarchies. The *Closure Under Coupling* principle according to Zeigler et al. [7] should apply to both atomic and aggregated tasks, i.e. a task composed of subtasks cannot be distinguished from an equivalent atomic task. This property is the basis for modularizing task-based descriptions and implementing reusable tasks. In addition, solving a problem may require processing tasks in parallel.

According to Figure 2, a task-based control is specified within the CM. Such a control specification is not directly executable because tasks are an abstract description of work steps. A task only describes the *what* but not the *how* or *with what* something is to be implemented. A *transformation method* is needed to execute tasks. It transforms tasks into control commands for real devices using a *world model*. The result is an executable control code in a language specific to the application. In robotics, for example, this can be a control code in a manufacturer-specific robot programming language. In the SBC approach, the PM corresponds to the world model and the IM to the transformation method.

1.3 Vendor-independent Control Development

In addition to different types of robots, which are optimised for certain areas of application in terms of hardware, the software solutions for programming robots also differ greatly and are often manufacturer-specific.

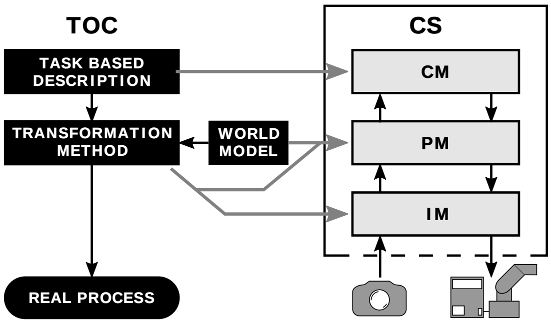


Figure 2: Implementation of TOC within the SBC approach.

Manufacturers are creating their own self-contained ecosystem that exclusively supports the robots, sensors and actuators of their own brand. The robots are mostly programmed using a specific programming language. Therefore, Robot Oriented Middleware from different manufacturers is generally incompatible.

Manufacturer-specific software and hardware systems complicate the development of robot applications, especially for MRS. Once developed, control solutions cannot simply be ported to robot systems from other manufacturers, even if the robots have almost the same functionality. Long-standing efforts to standardize robot programming languages have failed.

An alternative approach for robot programming is based on the client-server model, as shown schematically in Figure 3. Here, the robot controller acts as a server and gives a client access to all manufacturer-specific services of the robot. Client and server communicate via interfaces (Serial, Ethernet, etc.) of the robot controller. Control commands from the client are translated into a data word and then transmitted to an interpreter via a data connection. The interpreter is developed in the robot manufacturer's specific language and executed on the robot controller. The interpreter executes received data words and sends back data words to the client. In this way, sensor data and robot movement commands are communicated and executed.

Based on this concept, Deatcu et. al [8] developed a toolbox in MATLAB (client side) and interpreter (server) for KUKA, Kawasaki and virtual robots. Physical and virtual robots from both manufacturers can be programmed, tested and deployed using a uniform MATLAB-based command set. The integration of physical or virtual robots from other manufacturers only requires the implementation of an interpreter in their specific language.

With the virtual robots, the toolbox comprehensively supports the requirements of the RCP with regard to an end-to-end software chain and the use of simulation models from the early planning phase through to operation.

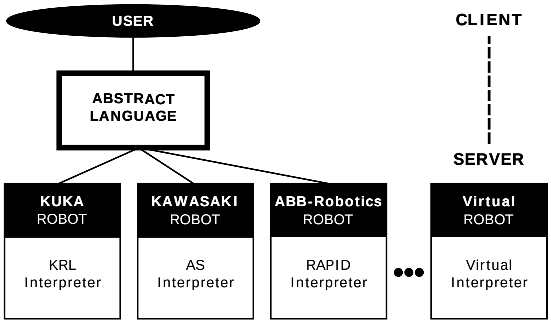


Figure 3: Robot control based on client-server approach.

2 Interaction in MRS and its Implementation Within the SBC Approach

In this section, interactions of joined-arm robots in an MRS are analyzed based on the interaction classes formulated by Lüth [9].

Subsequently, it is examined how the interaction type *coordination* can be realized within the SBC approach. According to section 1.2, the task specification, the *WHAT*, is done in the CM layer and the task transformation, the *HOW* and *WHO*, in the PM layer. We start our consideration with the task transformation and then devote ourselves to the task specification.

2.1 Interaction Classes

According to Lüth [9], the interactions of industrial robots in MRS can be divided into six classes. For better illustration, these are discussed below using the example of a transport problem.

Figure 4 shows the structure of the transport problem using an SRS. Parts are transported from an Input Buffer (IB) to an Output Buffer (OB). Based on this basic structure, Figure 5 shows derived system structures for MRS according to the six interaction classes by Lüth [9].

Class 0: The structure in Figure 4 shows an SRS that does not involve any interaction. The robot R1 can be understood as a server S with the capacity of one. It has to move one part at a time from the IB to the OB.

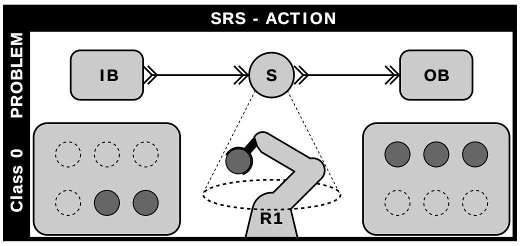


Figure 4: Structure of the transport problem with an SRS (interaction class 0).

Class 1: The MRS consists of two robots (R1, R2) with separate workspaces. Both robots have identical tools and capabilities.

The overall task consists of the transport of parts from the IB to the OB. Due to the separate workspaces and the same identity of the robots, they do not have to interact to solve their partial task.

Class 2: The structure is analogous to class 1, but each workspace contains a different part type, and robot R2 is adapted to the new part type. Due to the separate workspaces, the robots do not have to interact and do not need to be coordinated. In contrast to class 1, the robots are not mutually interchangeable.

Class 3: The robots are no longer spatially separated from each other. Their workspaces overlap, making it necessary to coordinate their movements in order to avoid collisions. *Coordination* requires an exchange of information, a *communication*.

Class 4: Another part type is added, which can only be moved by both robots together. This requires *cooperation* and *coordination* between the robots. The timing of the cooperation must be planned in order to avoid unnecessary waiting times for the robots.

Class 5: Each robot can handle the other robot's part types to support the partner in an overload situation. There follows no principally new requirement regarding communication and coordination.

Class 6: There is a third type of part to be transported that cannot be handled with the standard configuration of the two robots R1 and R2. This results in the need for a new interaction, referred to here as *dynamic function expansion*. This can be done by changing tools or temporarily integrating a specific robot R3.

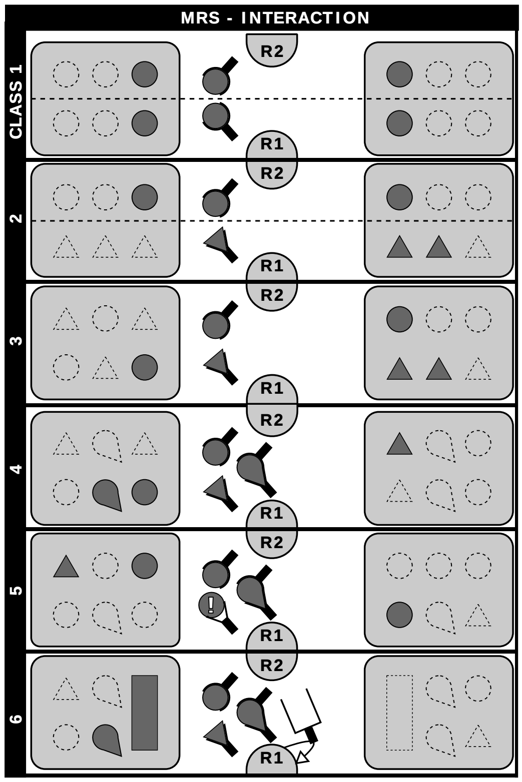
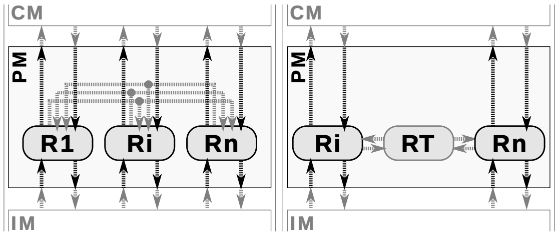


Figure 5: Illustration of the six interaction classes according to Lüth [9] using the transportation problem.

2.2 Task Transformation on the PM Layer

Interaction class 3 requires *temporal coordination* of robots in order to avoid collisions in the shared workspace. Figure 6 shows two variants for bidirectional *communication* of robot model components on the PM layer. In Figure 6a, the communication is solved on the basis of direct couplings between the robot components. The structure created by the couplings resembles a peer-to-peer (P2P) network as is known from the field of network technology. The effort required to couple components with each other in this way increases greatly with their number.

For this reason, the introduction of a new component, called Robot-Team (RT), is proposed in Figure 6b. This serves as a *middleman* between the robot components. The number of input and output ports required for communication is reduced to one input and output port per robot component R. The structure of a robot component R is independent of the number of components communicating with each other. RT stores all information relevant for team coordination. This significantly reduces the complexity of modeling at PM level.



(a) Peer-to-Peer (P2P) (b) Robot Team (RT)

Figure 6: Two variants for modeling communication on the PM layer.

2.3 Task Specification on the CM Layer and Relationship to the PM Layer

Figure 7 shows the basic implementation of a TOC within the SBC for interaction classes 1 to 3 according to the problem illustrated in Figure 5. The task sequences of the two robots (R1, R2) are specified as concurrent processes on the CM layer.

As already described, the task transformation takes place in the PM layer. The notation Ri.STS and Ri.CMD stands for coupling relationships between the CM and PM layers to exchange control commands (CMD) and status information (STS).

The interfaces to the robot middleware are implemented in the IM layer (INTF1 and INTF2). The RTC component in the IM layer stands for *Real Time Clock* and realizes the real-time synchronization of the executing simulator required during the operational phase.

Due to the separate workspaces, the robots can perform their tasks independently of each other in the case of interaction class 1 and 2. There is no need for interaction between the robots. The *Lock* and *UnLock* tasks on the CM layer and the RT component on the PM layer are not required.

A task sequence starts with the part identification task *IdPrT*, which provides position parameters (Pos). The Pos are passed to the task *Move*, which is followed by the task *PickPrt* to pick up a part, and so on.

In the case of interaction class 3, IB is a common workspace for both robots. It must be ensured that the robots do not move into the IB at the same time. To map a mutual exclusion (mutex) during the execution of the *Move(IB)* task for both robots, the tasks *Lock* and *UnLock* are introduced. These specify the start and end of a task coordination.

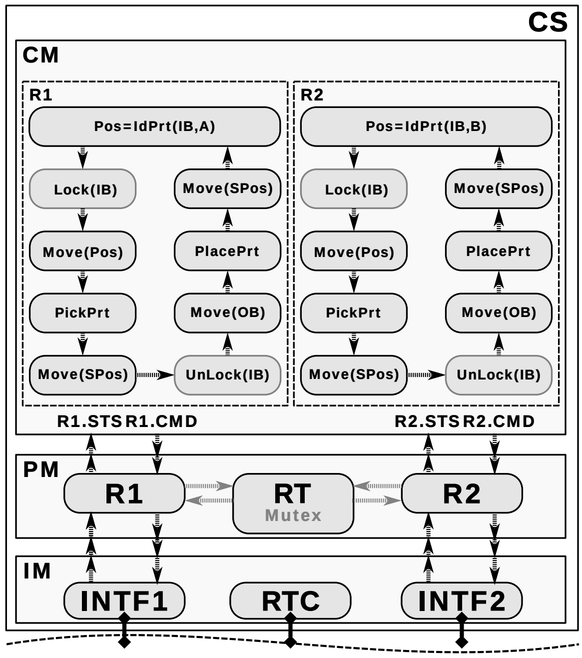


Figure 7: Concept of a TOC within the SBC approach by the example of interaction class 3 (for classes 1 and 2, the Lock and UnLock tasks as well as the RT component are omitted).

The resulting coordination of the two robots is implemented by the RT *Mutex* component. Mutex can be implemented using inter-process communication according to Tanenbaum and Bos [10].

When the task *Lock(IB)* is called, a resource with the name IB is reserved by the respective robot and blocked for others. The information about which robot owns which resource is stored in the RT component and exchanged via the coupling relationships between RT and the robot components R1 and R2. The robot that executes the *Lock* task first becomes the owner. Another robot cannot initially complete the *Lock* task.

However, a robot's wish to own the resource next can already be saved in RT. The reservation only becomes valid when the resource is released again with the *UnLock* task. If a reservation has already been made, ownership is immediately transferred to the reserving robot. Before a robot that is in ownership of a resource releases it again, it should leave the critical workspace. The *Move(SPos)* task is defined for this purpose, which describes the movement to a safe position.

3 DEVS-based Implementation of a TOC Within the SBC Approach

This section discusses the implementation of TOC according within the SBC approach using a DEVS simulation environment with a real-time process interface. It starts with the definition of tasks for the CM layer using DEVS. Then, we consider a DEVS-based transformation of tasks without interactions as well as a DEVS-based implementation of the interface to the robot middleware.

Finally, a DEVS-based transformation of tasks with interactions is described using the example of interaction type *Coordination* (class 3). The DEVS specifications are presented using extended DEVS diagrams. The practical implementation was carried out using DEVS-RCP-V2, a MATLAB-based DEVS environment. Details on the extended DEVS diagrams and on DEVS-RCP-V2 can be found in Freymann [4].

3.1 Specification of Tasks for the CM Layer

Figure 7 shows the task sequences of the case study on the transportation problem for interaction classes 1 to 3. With the exception of the task *IdPrt* for part identification, all other tasks can be defined according to the uniform pattern in Figure 8. The ports *STS* and *CMD* form the interface to the PM layer (see Figure 7)), while the ports *BEG* and *NXT* are used to link tasks in the CM layer. The task pattern defines the two phases *Passive* and *Active*.

In DEVS diagrams, phases are values of the state variable *phase*, which are represented with a box. These state values are often referred to as main states. Each state has a dwell time, including zero and infinity. The dwell time is notated with @*Variable* or @*Value*.

In the example in Figure 8, the initial phase is *Passive*. An external event *'next'* at the input port BEG (*BEG?'next'*) causes a change to the phase *Active* and the state transition $\sigma = 0$ schedules an immediate internal event. Internal events can cause state changes and trigger output events. In this case, an output event $(tid, p1, p2)$ at port CMD (*CMD!(tid,p1,p2)*) is generated. Parameter *tid* encodes the ID of the current task and the identifiers *p1, p2* are placeholders for specific task parameters. In the *Move* task, for example, the position to be approached is coded in *p1*.

Due to the state transition $\sigma = \infty$, the task remains in the phase *Active* until the next external event. When an external event *'done'* at the port STS (*STS?'done'*) occurs, the task changes to phase *Passive* and triggers immediately an internal event due to $\sigma = 0$. As a result, an output event *'next'* is generated at port NXT (*NXT!'next'*) and a follow-up task is activated. Due to the state transition $\sigma = \infty$, the current task remains in phase *Passive* until a new external event *BEG?'next'* occurs.

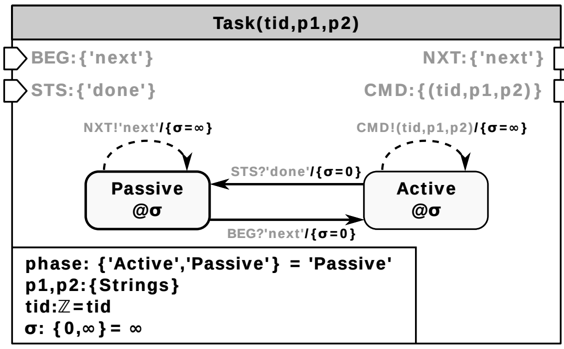


Figure 8: DEVS specification of a general task pattern.

3.2 Transformation of Tasks Without Interactions

The task transformation takes place by the robot components R1 and R2 in the PM layer (see Figure 7). Both components have an identical behavior. Figure 9 shows the DEVS specification for translating the task sequences of the case study for classes 1 and 2. It defines the three phases *Passive, Error* and *Active*, with the initial phase *Passive*. If component R is to execute a task, it must be in the phase *Passive*.

The task to be executed is received as an external event $(tid, p1, p2)$ via the input port CM_CMD (*CM_CMD?(tid,p1,p2)*) and an internal event is immediately scheduled with $\sigma = 0$. The external event codes a task ID in *tid* and task specific values in $p1$ and $p2$. The task ID decides which state transition, and therefore which task transformation, is executed.

The part identification task *IdPrt* is realized with a table, which defines part types and position coordinates. The part is selected depending on the task parameters $p1$ and $p2$, the coordinates are assigned to the state variable *position*, and the state variable *sts='done'* is set.

If no part can be identified, *sts='none'* is set. As there is no communication with the real process, there is a return to the phase *Passive*. The scheduled internal event triggers an output event (*CM_STS!sts*) that sends the value of the state variable *sts* to the CM layer via the port CM_STS. The state transition $\sigma = \infty$ schedules no further internal event, the component R is back in its initial phase, and ready to execute a new task again.

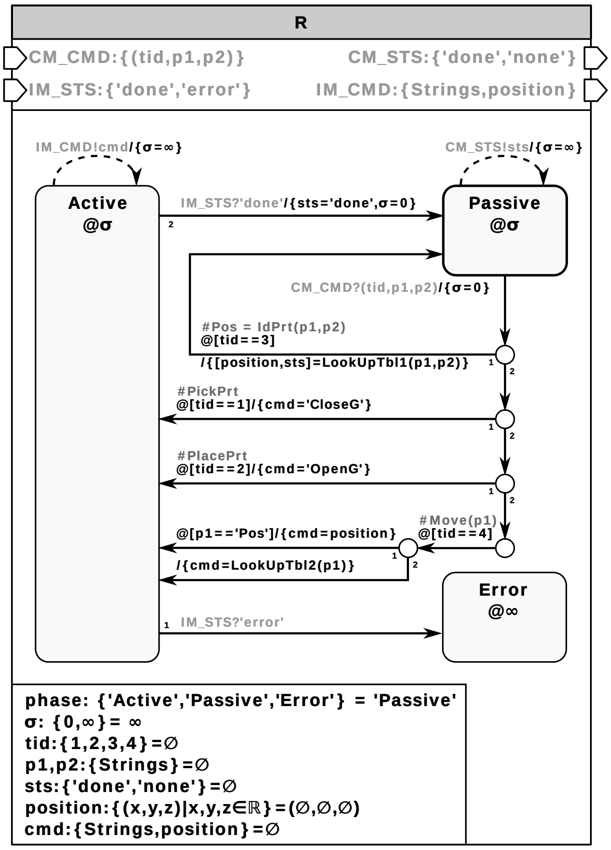


Figure 9: DEVS specification of the robot components R1 and R2 for translating the task sequences of the case study for classes 1 and 2.

The execution of the tasks *PickPrt, PlacePrt* and *Move* always leads to a transition to the phase *Active*. However, before this, the state transition caused by the external event (*CM_CMD?(tid,p1,p2)*) sets the state variable *cmd* to the task-specific value and schedules an internal event with σ=0.

For the task *Move*, a distinction is made between two transition variants. In the first case, the coordinates in the state variable *position* are assigned to the variable *cmd*. In the second case, the coordinates are determined via a lookup table. The internal event is then triggered in the phase *Active*.

This results in an output event *cmd* at port IM_CMD (*IM_CMD!cmd*), which is sent to the IM layer (see Figure 7). The R component remains in phase *Active* due to $\sigma = \infty$ until an external event, a status message from the IM layer, occurs at port IM_STS.

If a status event *'error'* (*IM_STS?'error'*) is received, the phase *Error* is entered, and an error handling routine is called, which is omitted in Figure 9. In the case of receiving a status event *'done'* (*IM_STS?'done'*), a transition to the phase *Passive* takes place, the state variable *sts='done'* is set, and with $\sigma = 0$ an internal event is scheduled. This triggers an output event *CM_STS!sts* to send the status information to the CM layer and sets $\sigma = \infty$. Now, the component R is ready to execute a new task. The transformation of the tasks *PlacePrt* and *PickPrt* are carried out in the same way.

3.3 Interface to the Robot Middleware

DEVS models require an interface to interact with real processes. In the DEVS-RCP-V2 formalism according to Freymann [4], activities are introduced for this purpose. According to Zeigler et al. [7], activities are Function Specified System (FNSS). An activity is characterized by a start and end event and has a duration. Permissible time windows can be defined for the execution times of activities. During the execution of activities, status queries can be made about the process.

In DEVS diagrams, activities are defined in the bottom right-hand field. For the interface component, an evaluation order of the internal state transitions (dashed lines) must be specified. The order is set with priorities, with value 1 as the highest priority. Otherwise, the notation is identical to the previous diagrams. The following explanation of the DEVS specification focuses on the special features with regard to activities.

Figure 10 shows the DEVS specification of the component INTF, which acts as interface between a robot component R in the PM layer and the robot middleware (see Figure 7). Like R, INTF defines the three phases *Passive, Active* and *Error*, with the initial phase *Passive*. In the phase *Passive*, INTF can receive input events from R and converts these into process interactions. This means that INTF must start and monitor activities and sends status messages back to R as output events.

INTF defines three activities. Two activities are based on application programming interface functions of the robot middleware.

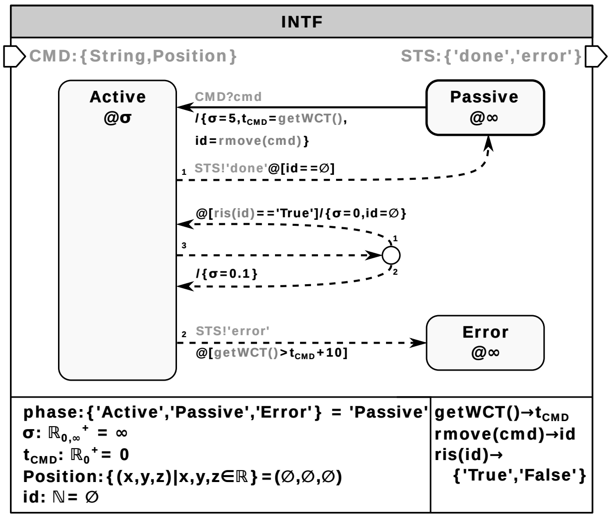


Figure 10: DEVS specification for the robot interface components INTF1 and INTF2.

The activity *rmove* starts a robot or gripper movement and returns an *id* as an identifyer. This *id* can be used with the activity *ris* to determine whether the movement has been completed or not. The activity *getWCT* provides the wall-clock-time (WCT) and is used to monitor the execution time of a movement activity.

When a movement activity starts, i.e. a state transition from *Passive* to *Active* is triggered by an external event *CMD?cmd*, the WCT is written to the state variable t_{CMD}. It is assumed that a movement takes at least 5 seconds (σ=5) and is completed after a maximum of 10 seconds, which is defined by the condition $@[getWCT > t_{CMD} + 10]$. If the upper time limit is exceeded, a transition from phase *Active* to *Error* occurs and an output event *'error'* is sent to the PM layer via port STS (*STS!'error'*).

The sampling time for status queries to the process with the activity *ris* is set to 0.1 seconds (σ=0.1). If the *ris* activity returns the status *'True'* in the permitted time interval, the condition $@[ris(id) == \textit{'True'}]$ applies.

An immediate internal event is scheduled with σ=0. This means that the output event *STS!done* is triggered immediately in phase *Active* and INTF switches to phase *Passive* with $\sigma = \infty$.

For real-time synchronization, DEVS-RCP-V2 defines an RTC component on the IM layer (Figure 7). The DEVS specification of RTC can be found in Freymann [4].

3.4 Transformation of Tasks with Interactions

The interaction class 3 in Figure 5 defines that the robots move into a common workspace. Access to the IB is critical and must be *coordinated* by mutual exclusion (mutex). Mutex has been discussed in Subsection 2.3 and the concept of a related TOC structure is shown in Figure 7. The coordination of the robots via Mutex is implemented in the RT component on the PM layer.

The robot components R on the PM layer must be adapted to communicate with RT. The tasks *Lock* and *UnLock* on the CM layer can be specified as described in Section 3.1.

Figure 11 shows the DEVS specification of the adapted component R. Communication with RT takes place via the input/output ports RT_IN and RT_OUT. In addition, a new phase *PassiveRT* is introduced. An external event *CM_CMD?(tid,p1,p2)* with *tid=-1* or *tid = -2* encodes the execution of the newly introduced tasks *Lock* or *UnLock*. σ = 0 schedules an internal event and R switches to phase *PassiveRT*. In *PassiveRT* an output event *RT_OUT!(tid,rid,p1)* is triggered and send to the RT component.

The event contains the task ID in *tid*, the robot ID in *rid* and the identifier of the resource to be locked or unlocked in the parameter *p1*. Due to $\sigma = \infty$, R remains in the phase *PassiveRT* until a *'done'* event is received from RT, which causes the transition of R to the phase *Passive*.

Here, R schedules an internal event ($\sigma = 0$) that triggers an output event *CM_STS!sts* with *sts='done'* to the parent CM layer. R waits in phase *Passive* ($\sigma = \infty$).

Figure 12 shows the specification of component RT.[1] It processes incoming events from the robots R (R1_IN?(tid1,rid1,p1) or R2_IN?(tid2,rid2,p2)) and responds to them with a *'done'* event if a task can be executed.

Robots R can simultaneously request the reservation of a resource from RT. In this case, robot R1 is given priority because its reservation request is checked first. The reservation is made by means of a resource identifier, such as *IB* in the transportation problem.

The information about which robot reserves which resource is stored in a resource list *RcsList*. A resource can only be added to the list if it does not already exist in it.

[1] The specification of RT is incorrect in Freymann [4].

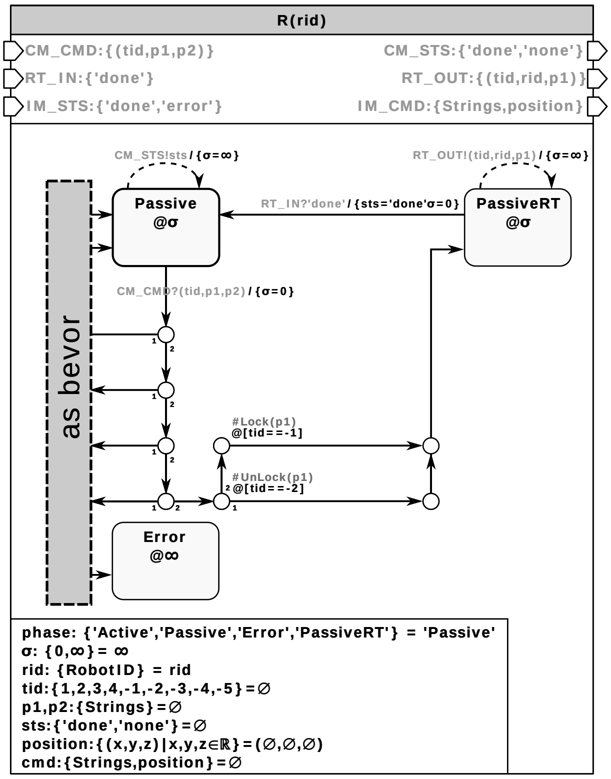


Figure 11: Extension of the DEVS specification for R to support coordination (interaction class 3)

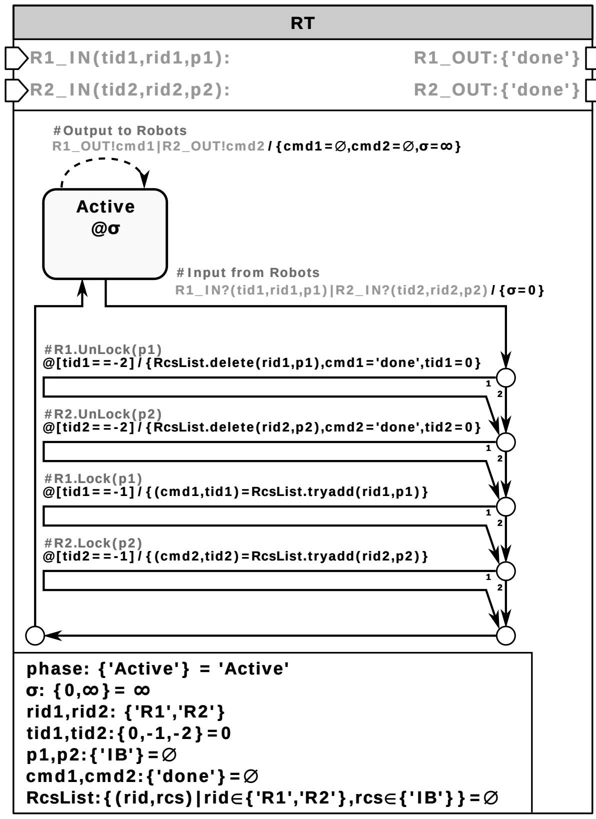


Figure 12: DEVS specification of RT component for interaction class 3

(cmd,tid) = **RcsList.tryadd**(rid,p)
if p is not in RcsList
  cmd = ’done’; tid = 0; RcsList.add(rid,p);
else
  cmd=∅; tid = -1;
with:
  cmd ∈ {’done’}, tid ∈ {0,-1}
  rid ∈ {’R1’,’R2’}, p ∈ {’IB’}
  logging of rid within RcsList is just for debugging

A successful reservation is communicated to the robot components R via a *’done’* event. If a resource is released again its ownership can immediately pass to another robot. It follows from this that, due to concurrency, the task *UnLock* must be evaluated first. Only then is the task *Lock* evaluated.

4 Summary

It has been shown that the classification of interactions in MRS enables their systematic mapping in the form of reusable tasks. In this way, controls for MRS can be specified in a completely task-oriented manner, analogous to SRS. The task transformation takes place analogously to SRS, with the complexity increasing when interactions are added.

Using an application example, it has been shown that the SBC approach can be used to develop manufacturer-independent and consistent simulation-based controls from the early planning phase to the operational phase. In the course of implementation, generic components were derived that were consistently reused and gradually developed further. The prerequisite for the approach is appropriate robot midleware.

The DEVS formalism has been used as the basis for the SBC approach. This enabled the modular task concept to be implemented step by step and consistently. With the help of DEVS diagrams, even partially complex dynamics could be clearly represented. For the practical tests, the diagrams were implemented one-to-one in a DEVS based MATLAB environment.

Interactions between robots and humans and with the environment were not taken into account. However, the work can serve as a blueprint. The implementation of the other interaction classes will be presented in a follow-up article.

References

[1] Abel D, Bollig A. *Rapid Control Prototyping, Methoden und Anwendung*. Springer. 2006.

[2] Siciliano B, ed. *Springer handbook of robotics*. Berlin: Springer. 2008.

[3] Maletzki G. Rapid control prototyping of complex and flexible robot controls based on the SBC approach (in German). Dissertation, Univ. of Rostock & Wismar Univ. of Appl. Sciences. 2014. *Advances in Simulation Vol. 25*. DOI: 10.11128/fbs.25

[4] Freymann B. Task-oriented multi-robot controls based on the SBC framework and DEVS (in German). Dissertation, Technical Univ. Clausthal & Wismar Univ. of Appl. Sciences. 2022. *Advances in Simulation Vol. 40*. DOI: 10.11128/fbs.40

[5] Maletzki G, Pawletta T, Dünow P, Manemann P. Simulation-based control of a robot cell (in German). *Frontiers in Simulation - Simulationstechnique 18th Symposium, Erlangen, Germany*. 2005;pp. 305–310.

[6] Schwatinski T, Pawletta T, Pawletta S. Flexible Task Oriented Robot Controls Using the System Entity Structure and Model Base Approach. *SNE Simulation Notes Europe*. 2012;22(2):107–114. DOI 10.11128/sne.22.tn.10135

[7] Zeigler BP, Muzy A, Kofman E. *Theory of Modeling and Simulation. 3rd Edition*. Academic Press. 2019.

[8] Deatcu C, Freymann B, Schmidt A, Pawletta T. MATLAB/Simulink Based Rapid Control Prototyping for Multivendor Robot Applications. *SNE - Simulation Notes Europe Journa*. 2015;25(2):69–78. DOI 10.11128/sne.25.tn.10293

[9] Lüth T, Längle T. Multi-agent systems in robotics and artificial life (in German). In: *GMD Workshop Artificial Life*. Sankt Augustin, Germany. 1995; pp. 1–10.

[10] Tanenbaum AS, Bos H. *Modern Operating Systems*. Pearson, 5th ed. 2022.

Coupled Simulation of the Injection and Combustion Process of an Industrial Diesel Engine

Daniel Jörss[1], Maximilian Ringel[1], Bert Buchholz[2], Christian Fink[1*]

[1]Department of Mechanical / Process and Environmental Engineering, Wismar University of Applied Sciences, Philipp-Müller-Str. 14, 23966 Wismar, Germany; *Christian.Fink@hs wismar.de

[2]Chair of Piston Machines and Internal Combustion Engines, Rostock University, Albert-Einstein-Straße 2, 18059 Rostock, Germany

SNE 34(4), 2024, 195-202, DOI: 10.11128/sne.34.tn.10712
Selected ASIM WS 2023 Postconf. Publication: 2023-10-15
Rec. Improved English Vs.: 2024-08-06; Accepted: 2024-09-15
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. In order to comply with future exhaust emission limits, very precise control of the injection quantity and injection timing is of crucial importance. The injection system fulfillling these requirements in the best way is the common rail system.

The model presented here describes a 1D hydraulic injector model in Simscape™, which is coupled to a zero-dimensional combustion engine model. Measurements were carried out using an injection rate analyser to validate the injector model. Good results were achieved between simulation and experiment.

In the second step, a model developed in Matlab® Simulink® [1] was extended with a phenomenological combustion model according to Barba [2]. The injection rate is transferred to the combustion model as an input parameter. This is used to calculate the heat release rate by means of the combustion engine model, taking other sub-models into account. Good agreements were achieved between the heat release rate calculated from the pressure curve analysis and the simulation.

Introduction

The introduction of common-rail injection technology has made it possible to decouple pressurisation and injection in terms of timing, pressure and number of injections. The necessary injection pressure in the system is generated by a separate high-pressure pump. The pressurised fuel is then accumulated in a high-pressure rail which is placed between the pump and the injectors. Instead of pressure-controlled injection nozzles, electrically actuated injectors are used in common-rail systems. The timing and duration of the injection is determined by the electrical control of the injector.

This allows for wide flexibilities with regard to multiple injection strategies and injection pressure depending on the load point. This makes it possible to develop injection strategies in such a way that considerable emission reductions can be achieved by engine internal measures.

The timing, the injection rate and the atomisation quality of the fuel injected into the combustion chamber prove to be important parameters. Extensive test bench trials are often necessary for the best possible adaptation of each operating point in the engine map. With today's simulation methods it is possible to model the complex processes in a combustion engine in detail to better understand the operating and emission behaviour and to analyse the effects of different applications.

This paper describes the model of a common-rail injector connected to a zero-dimensional combustion engine model [1] and compares the simulation results obtained with test bench measurements.

1 Model Description

The physical object-orientated toolbox Simscape™ was selected in Simulink® to simulate the common-rail injector. Conventional common-rail injectors consist of three main assemblies: the control valve, the injection nozzle and a mechanical-hydraulic coupling element. With reference to Figure 1, the pressure in the valve control chamber (2) determines whether the injection nozzle opens or closes or remains opened or closed. The magnitude of the pressure in the control chamber depends on the rail pressure on the one hand and the ratio of the inlet and outlet throttle on the other.

If the control valve is open, fuel flows out of the control chamber via the outlet throttle and the pressure drops.

A falling pressure reduces the force acting on the control piston on the nozzle needle, which is subjected to a force equilibrium. When the pressure falls below a certain level, the nozzle needle opens and the injection process begins. When the control valve closes the outlet throttle, the pressure in the control chamber rises and increases the force acting on the nozzle needle via the valve control piston until it closes and ends the injection process.

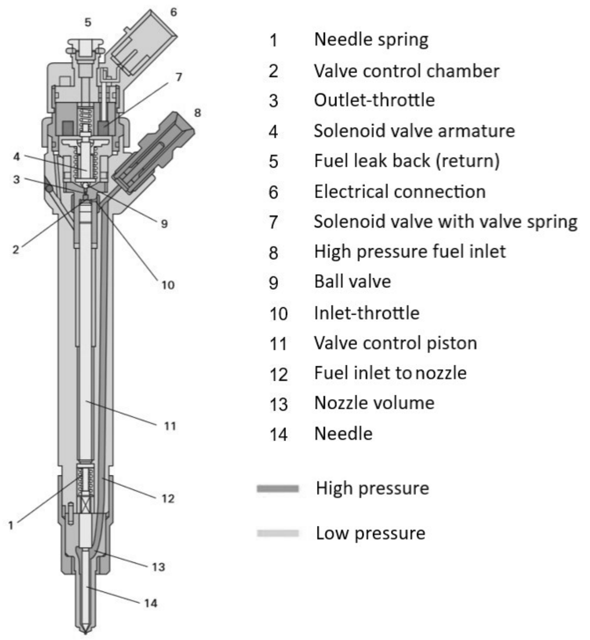


Figure 1: Structure of a typical solenoid valve injector [3].

In the simulation environment, the injector is divided into subsystems in which different domains of Simscape^{TM} are used. The geometric dimensions of the individual components and the volume of the injector to be simulated were determined in advance by measurements. Parameters of the solenoid valve were taken from [4], [5].

1.1 Solenoid Valve Model

The solenoid valve model comprises a total of four domains from the Simscape^{TM} library. Figure 2 shows how the individual blocks are connected to each other. The structure of this submodel includes a pulse width modulated voltage signal as input variable in accordance to [4]. Taking into account the ohmic resistance and the inductance, this results in the solenoid current.

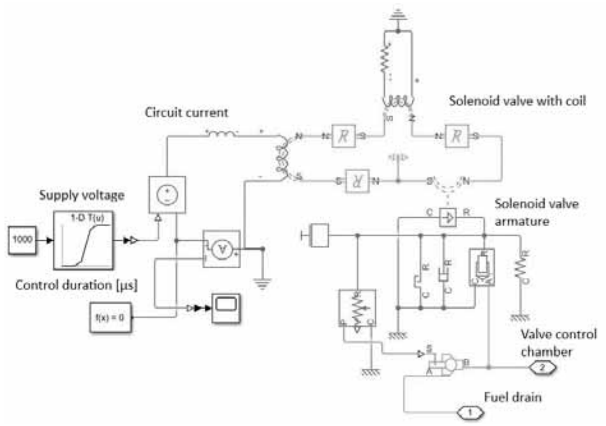


Figure 2: Structure of the solenoid valve in Simscape^{TM}.

The actuation time for the solenoid valve results from the time curve of this signal. This relationship is shown as an example in Figure 3 for a solenoid actuation duration of 1000 µs. The current creates a magnetic field in the coil of the solenoid valve, which attracts the solenoid valve armature. If the magnetic force outweighs the opposing spring force and the pressure force acting on the ball valve, the ball valve starts to lift out of the seat.

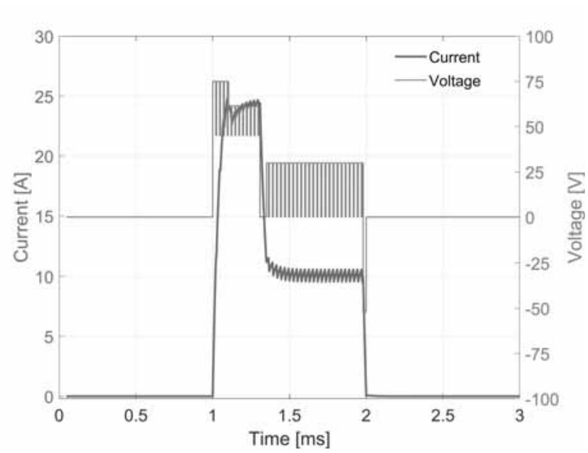


Figure 3: Simulated current- and voltage curve.

Figure 4 shows the actuation current and the valve lift movement of the solenoid valve armature versus time. The negative portion of the valve lift can be attributed to compression and elongation of the components caused by the high system pressure.

The high current in the pull-in phase causes the valve to open quickly.

After a short time, the hold phase begins, during which the ball valve remains open. At the end of actuation, the solenoid valve spring pushes the solenoid armature downwards, closes the valve and thus stops the flow through the outlet throttle.

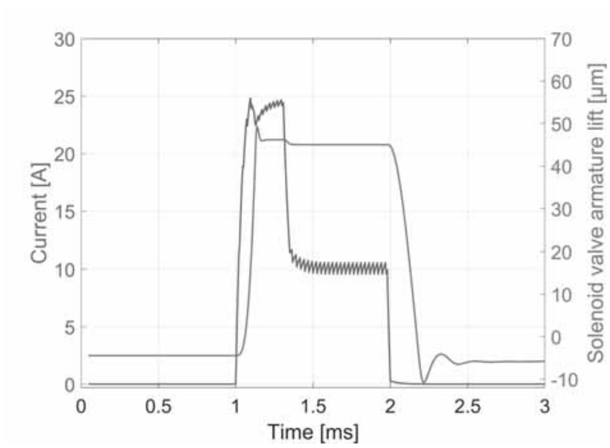


Figure 4: Control current and solenoid valve lift.

1.2 Mechanic-hydraulic Model

This model describes the coupling between the hydraulic and mechanical components in the injector. The lift movement of the solenoid valve armature from the solenoid valve model and a specified fuel pressure are used as input variables.

Figure 5 shows the structure of the injector model in Simscape™. The two hydraulic cylinder blocks represent the nozzle needle and the valve control piston with their corresponding masses.

As in [4], the nozzle holes are modelled by a combination of cross-sectional constriction and a throttling point. In the non-actuated state, the same pressure is present in the control chamber and in the nozzle volume. A force acts on the surface of the control piston and the pressure shoulder of the nozzle needle. Due to the surface ratios, a larger force is created towards the needle seat and the injector remains closed. When the solenoid valve is activated, the ball valve lifts out of the seat and opens the outlet throttle.

As a result, the fuel flows out of the valve control chamber, causing the pressure to drop. So that the pressure difference between the control chamber and the nozzle volume creates a force on the needle in upwards direction causing the needle to open.

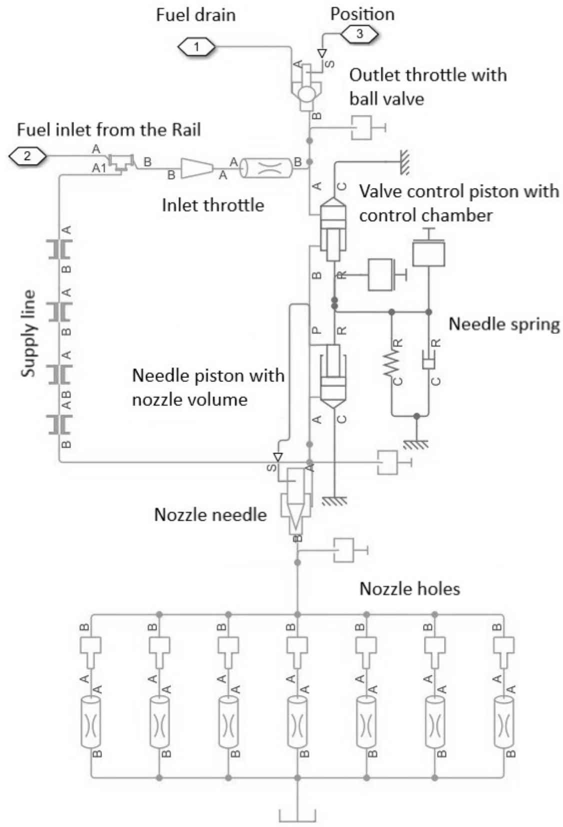


Figure 5: Structure of the injector model in Simscape™.

The opening speed at which the nozzle needle moves depends on the cross-sectional ratio between the inlet and outlet throttle of the control chamber since this ratio influences the resulting pressure in the control chamber.

Figure 6 presents simulation results of the control valve lift, control chamber pressure, needle lift and injection rate for a typical load point revealing the relation of the different parameters. At the end of the solenoid valve actuation, the valve closes the outlet throttle and stops the flow. This causes the pressure in the control chamber to build up, resulting in an increased downward force on the control piston.

The closing speed is largely determined by the inlet throttle. If the hydraulic force from the valve control chamber and the force of the needle spring exceeds the opposing force on the nozzle needle, the nozzle needle begins to close. Injection ends when the nozzle needle reaches the nozzle body seat and closes the nozzle holes.

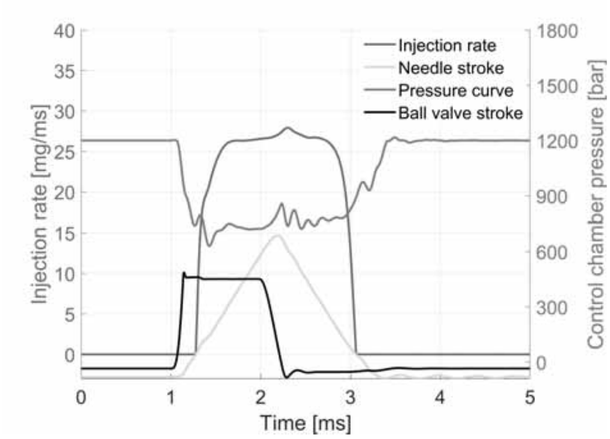


Figure 6: Simulation results injector model.

2 Model Validation

To validate the simulation model, the calculation results are compared with measured data from the injection rate analyser. A comparison of injection rates for two different operating points is shown in Figure 7. In both cases, the measured data curve is the average curve from 150 consecutive injection processes. A comparison of the injection rates shows that the simulation and measurement data are in good agreement. The deviation of the injected quantities between simulation and experiment is around 1%. Similarly good results were also observed at other operating points, which confirms the applicability of the developed injector model.

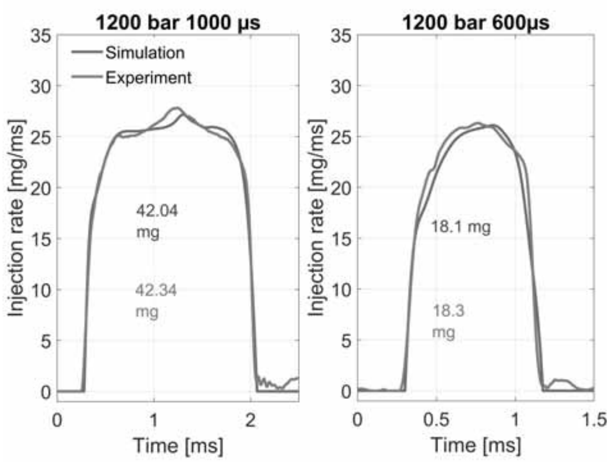


Figure 7: Comparison of injection rates between measurement and simulation at two different injection durations.

3 Combustion Model

The approach presented here describes a global phenomenological combustion model for calculating the heat release rate according to Barba [2].

This model uses the injection rate as an input variable, which opens up more possibilities for parameter studies than conventional empirical models.

As in [6], the basic idea here is that the fuel conversion is controlled by the ignition delay and the fuel vaporisation. The first model component for fuel vaporisation is the calculation of the primary droplet diameter d_{Tr0} [7] according to equation 1.

$$d_{Tr0} = c \cdot d_{D.eff} \cdot (Re \cdot We)^{-0.28} \quad (1)$$

The Reynolds and Weber numbers are calculated using equations 2 and 3.

$$Re = \frac{u_{Tr0} \cdot d_{D.eff}}{\nu_{B.fl}} \quad (2)$$

$$We = \frac{{u_{Tr0}}^2 \cdot d_{D.eff} \cdot \rho_Z}{\sigma_{B.fl}} \quad (3)$$

The output droplet velocity u_{Tr0} is calculated directly from the injection rate according to equation 4.

$$u_{Tr0} = \frac{\frac{dm_E}{dt}}{\rho_{B.fl} \cdot A_{D.eff}} \quad (4)$$

Using the d^2-law in equation 5, the vaporisation time can be calculated for each calculation step based on the output droplet diameter and an empirical factor β.

$${d_{Tr}}^2 = {d_{Tr0}}^2 - \beta \cdot t \quad (5)$$

Literature values for the coefficient c are 8.7 [7], 4.0 [2] and 25.0 [8]. For the vaporisation coefficient β, the value is given as $7 \cdot 10^{-6}$ [2][10]. In addition to the vaporisation time, the ignition delay must also be taken into account.

The ignition delay is generally the time between the start of injection and the first significant release of heat. According to equation 6, Barba [2] divides this into a physical and a chemical component.

$$\tau_{ZV} = \tau_{ZV,phy} + \tau_{ZV,chem} \quad (6)$$

The proportions can be calculated using equations 7 and 8.

$$\tau_{ZV,phy} = c_0 \cdot u_{Tr0}{}^{-1.68} \cdot d_{D.eff}{}^{0.88} \tag{7}$$

$$\tau_{ZV,chem} = c_1 \cdot \left(\frac{p_z}{p_0}\right)^{c_2} \cdot \lambda_{Zn}{}^{c_3} \cdot e^{\frac{T_A}{T_Z}} \tag{8}$$

As the temperatures and pressures change during the calculation of the ignition delay, the ignition integral according to equation 9 is used.

$$1 = \int_{t_{EB}}^{t_{VB}} \frac{1}{\tau_{ZV}}\, dt \tag{9}$$

The ignition event occurs when the integral has reached the value 1. Figure 8 shows the simulation results of the vaporisation model.

The droplet diameter and the corresponding vaporisation time are calculated from the injection rate for each calculation step.

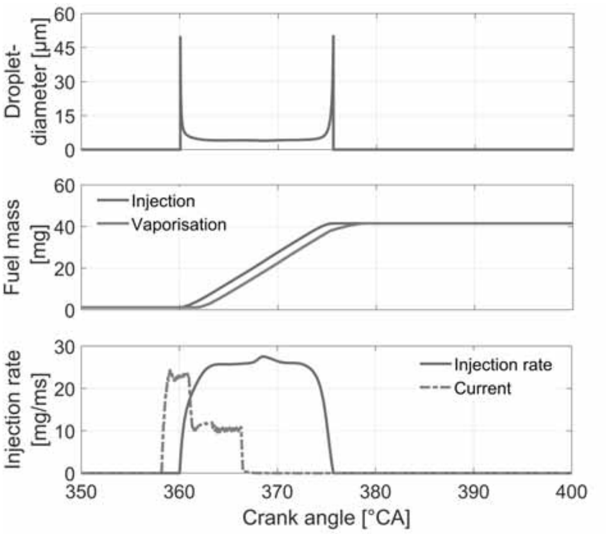


Figure 8: Simulation results vaporisation model.

In Simulink®, the vaporisation time and the ignition delay are transferred to a "Transport Delay" as a time delay. This allows the cumulative injected fuel mass to be shifted by this time period as an input variable.

The most important parameter for the combustion model is the available unburnt mass of vaporized fuel.

The description of diffusion combustion according to equation 10 is based on a frequency approach which, in addition to a characteristic mixing length, also uses a mixing velocity which stands for the turbulence in the combustion chamber [2].

$$\frac{dm_{B.V}}{dt} = \frac{u'}{l_{Diff}} \cdot m_{B.UV} \tag{10}$$

According to equation 11, the mixing length l_{Diff} is made up of the combustion chamber geometry, the current combustion air ratio and the number of nozzle spray holes.

$$l_{Diff} = \sqrt[3]{\frac{V_Z}{\lambda \cdot Anz_D}} \tag{11}$$

Due to the turbulence occurring in the combustion chamber, the mixing velocity u' is divided into a base turbulence and an injection turbulence.

The effects that occur due to inflow or turbulence are recorded in the base turbulence and mapped via the mean piston speed c_m, which represents a rough simplification.

By vectorial addition of the base turbulence and the injection turbulence, equation 12 can be written as follows [2].

$$u' = \sqrt{c_G \cdot c_m{}^2 + c_{Kin} \cdot k} \tag{12}$$

The two parameters c_G and c_{Kin} allow a weighting of the two components as well as an adjustment of the diffusion model to measurement data [2]. Equation 10 can therefore be formulated as follows:

$$\frac{dm_{B.V}}{dt} = \frac{\sqrt{c_G \cdot c_m{}^2 + c_{Kin} \cdot k}}{\sqrt[3]{\frac{V_Z}{\lambda \cdot Anz_D}}} \cdot m_{B.UV} \tag{13}$$

The specific kinetic turbulence energy k summarises the diffusion of the fuel and the kinetic energy supplied to the system as a result of the injection.

This highly simplified $k - l_l$ model is described according to equation 14.

$$\frac{dk}{dt} = -c_{Diss} \cdot \frac{1}{l_l} \cdot k^{\frac{3}{2}} + c_E \cdot \frac{dk_E}{dt} \tag{14}$$

The kinetic energy supplied results from the injection rate over time and the corresponding output droplet velocity.

In addition, Barba takes a global view of the cylinder by using the entire cylinder filling as a scale for the required kinetic energy density [2]. This can be described according to equation 15 as follows:

$$\frac{dk_E}{dt} = \frac{1}{2} \cdot \frac{dm_E}{dt} \cdot u_{Tr0}{}^2 \cdot \frac{1}{m_Z} \tag{15}$$

Using the burnt fuel mass from equation 13 and the lower calorific value for diesel fuel, the heat flux supplied is calculated according to equation 16.

$$\frac{dQ_B}{dt} = H_U \cdot \frac{dm_{B,V}}{dt} \tag{16}$$

For the simulation calculation, this model replaces the previously used combustion model according to Vibe [9] in the energy balance.

Figure 9 shows the comparison of the heat release rate from the pressure curve analysis of experimentally determined measurement data and the phenomenological modelling approach. The second part of the main combustion in particular shows that the phenomenological model delivers a good result. Due to the consideration of the injection rate, the phenomenological model can be used to investigate the impact of further partial injections and variations of hydraulic parameters of the injection system.

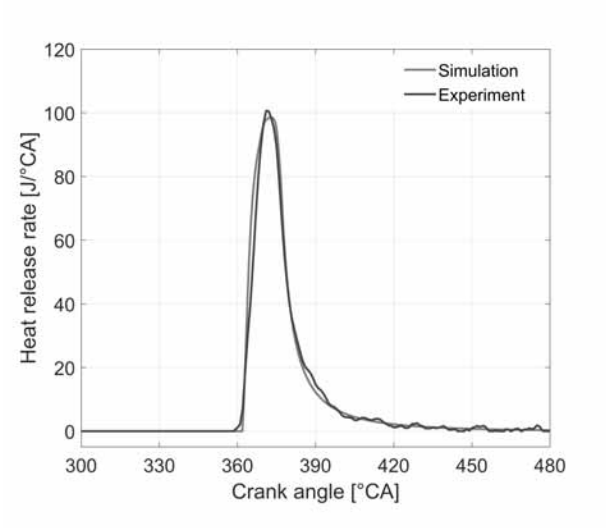


Figure 9: Comparison of heat release rate from cylinder pressure measurement and simulation model.

4 Overall Model Validation

To validate the simulation model, the calculation results are compared with measurement data from the engine test bench. The input parameters such as injector current duration, rail pressure and boost pressure are modelled according to the test bench data.

Figures 10 and 11 show a comparison of the rate of heat release at two different load points. It can be seen that the coupled simulation model shows a high level of agreement with experimentally obtained data even when multiple injection is used.

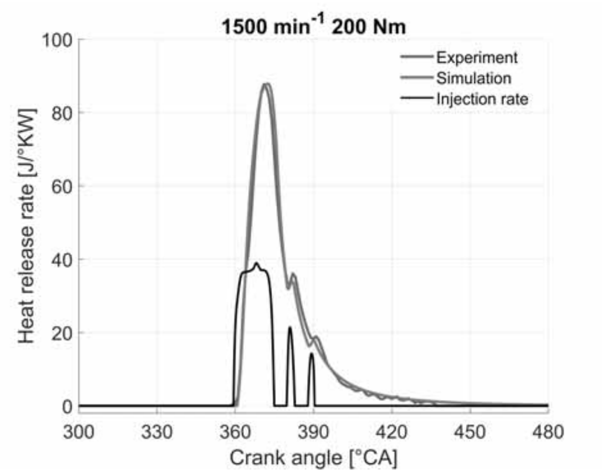


Figure 10: Comparison of the heat release rate determined experimentally and simulatively when using multiple injection at high load

The heat release peaks that occurs after the main injection can clearly be recognised and referred to the post injection events. Further validation is carried out by means of the corresponding cylinder pressure curves at these operating points, which are shown in Figure 12. It can be seen that the combustion behaviour is reproduced very well.

A comparison of the mean pressure, as an integral parameter for characterising the engine load, shows a deviation of 2 % between measurement and simulation. Similarly, good results were also observed at other operating points with single and multiple injection, which confirms the applicability of the simulation approach and the selected models.

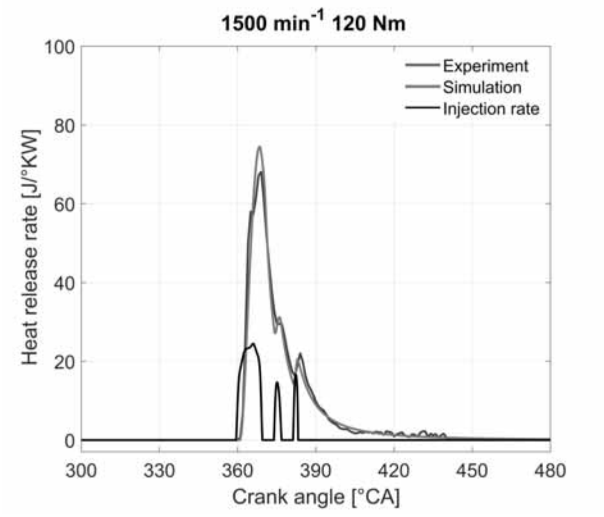


Figure 11: Comparison of the heat release rate determined experimentally and simulatively when using multiple injection at medium load.

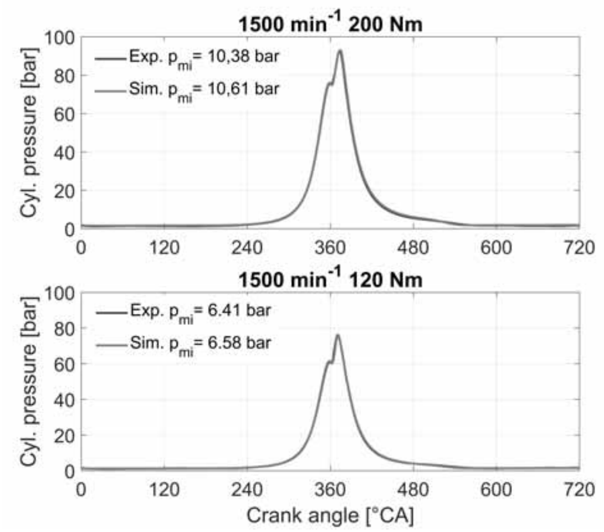


Figure 12: Comparsion of the cylinder pressure curve between simulation and measurement for two different load points.

5 Conclusion and Outlook

The modelling approach presented in this article describes a 1D hydraulic injector model in Simscape™, which was coupled with a phenomenolgical combustion model implemented in Simulink®. The injector model is a powerful tool which allows for a detailled description of the injector behaviour.

The combustion model provides a very good basis for calculating the heat release in the combustion chamber considering the injection rate. A comparison of the simulation results showed very good agreement with the measurement results from the engine test bench, which supports the applicability of these modelling approaches.

A key objective for future work is the further development of the model with regard to dynamic operating scenarios. Furthermore, a focus is on implementing a nitric oxide emission model in order to analyse the emission behaviour in parameter studies and to derive optimisation measures.

Acknowledgement

The results presented here were obtained within the framework of the publicly funded project SIDYN (FKZ:13FH043PX8) and the doctoral program of Wismar University of Applied Sciences.

The authors want to thank the BMBF and Wismar University for the provided financial support.

Nomenclature

β - vaporisation coefficent [m^2/s]
λ - air fuel ratio [-]
λ_{Zn} - local air fuel ratio [-]
$\nu_{B.fl}$ - liquid fuel viscosity [m^2/s]
$\rho_{B.fl}$ - liquid fuel density [kg/m^3]
ρ_Z - density in the combustion chamber [kg/m^3]
$\sigma_{B.fl}$ - liquid surface tension [N/m]
τ - time [s]
τ_{ZV} - ignition delay [s]
$A_{D.eff}$ - effective nozzle cross-section [mm^2]
c - universal constant [-]
c_m - mean piston speed [m/s]
$d_{D.eff}$ - effective nozzle diameter [mm]
H_U - lower calorific value [J/kg]
k - specific kinetic turbulence energy [m^2/s^2]
l_{Diff} - mixing length [m]
l_l - length scale [m]
m_B - fuel mass [kg]
m_E - injected fuel mass [kg]
p_0 - ambient pressure [bar]
p_{mi} - indicated medium pressure [bar]
p_z - cylinder pressure [bar]
Q_B - energy released during combustion [J]
Re - Reynolds number [-]
T_A - activation temperature [K]
t_{EB} - start of injection [s]
t_{VB} - start of combustion [s]
T_Z - gas temperature in the combustion chamber [K]
u' - turbulence intensity [m/s]
u_{Tr0} - output droplet velocity [m/s]
V_Z - cylinder volume [m^3]
We - Weber number [-]

References

[1] Jörss, D., Herrmann, B., Fink, C. *Modellierung des Betriebsverhaltens eines Diesel-Industriemotors als Stromerzeuger*, Virtueller ASIM Workshop der Fachgruppe STS/GMMS & EDU, 2021, DOI: 10.11128/arep.45

[2] Barba, C. *Erarbeitung von Verbrennungskennwerten aus Indizierdaten zur verbesserten Prognose und rechnerischen Simulation des Verbrennungsablaufes bei Pkw-DE-Dieselmotoren mit Common-Rail-Einspritzung*, Zürich, ETH Zürich, Diss., 2001.

[3] Audi AG, *http://www.volkspage.net/technik/ssp/ssp/SSP_227.PDF*, letzter Zugriff: 08.12.2022

[4] Vermeiden, J.C. *Fuel injection equipment characterisation for short and multiple injections experimental setup design and model-based sensitivity analysis* , Eindhoven University of Technology, 2020.

[5] Payri, R., Salvador, FJ., Carreres, M., De la Morena, J. *Fuel temperature influence on the performance of a last generation common-rail diesel ballistic injector. part ii: 1d model development, validation and analysis* Energy Conversion and Management, 114:376-391, 2016.

[6] Constien, M., R., Woschni G. *Vorausberechnung des Brennverlaufes aus dem Einspritzverlauf für einen direkteinspritzenden Dieselmotor*, MTZ 53, Nr.7/8, 1992.

[7] Varde, K.S., Popa, D.M., Varde L.K. *Spray Angle and Atomization in Diesel Sprays*, SAE 841055, 1984.

[8] Kožuch, P. *Ein phänomenologisches Modell zur kombinierten Stickoxid- und Rußberechnung bei direkteinspritzenden Dieselmotoren*, Stuttgart, Universität Stuttgart, Diss., 2004.

[9] Vibe, I. *Brennverlauf und Kreisprozess von Verbrennungsmotoren*, VEB Verlag Technik, Berlin, 1970.

[10] Grill, M. *Objektorientierte Prozessrechnung von Verbrennungsmotoren*, Stuttgart, Universität Stuttgart, Diss., 2006.

Using Dynamic Time Warping and Synthetic Data for Validating Seq2Seq in Simulation

Benjamin Wörrlein, Steffen Straßburger

Fachgebiet Informationstechnik in Produktion und Logistik, Technische Universität Ilmenau, Max-Planck-Ring 12, 98693 Ilmenau, Deutschland; *benjamin.woerrlein@tu-ilmenau.de; steffen.strassburger@tu-ilmenau.de*

SNE 34(4), 2024, 203-213, DOI: 10.11128/sne.34.tn.10713
Selected ASIM WS 2023 Postconf. Publication: 2023-10-15
Rec. Improved English Vs.: 2024-08-03; Accepted: 2024-09-20
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. Seq2Seq is a machine learning method that allows to translate sequences into other sequences. This method has been tried in hybrid simulation of machine tools. The method has been used to generate time series of energy consumption of jobs from the corresponding numerical control code that runs on a machine tool. Seq2Seq suffers from various problems. Firstly, the creation of training data is costly. Secondly, standard Seq2Seq metrics only allow for the evaluation of a prediction of one timestamp at a time, not an entire time series. Thirdly, training metrics are failing when vanilla data is used, as two identical numerical control codes can result in deviating time series. This causes confusion for the model in the training loop, as it is not clear which time series should be considered correct.

Here we propose a holistic framework to all three problems, that contains synthetic data, additional metrics for time series and dynamic time warping.

Introduction

Sequence-to-Sequence (Seq2Seq) [3, 16] is a class of machine learning (ML) methods that enables mapping two sequences of different lengths and different descriptions to each other. Seq2Seq utilizes artificial neural networks and is categorized under deep learning [4].

In the context of simulation, Seq2Seq has already been successfully used to translate numerical control codes (NC codes) of a machine tool (MT) into a time series of the energy consumption of the same machine. For this, a training dataset is first created based on individual manufacturing orders (MO).

This dataset contains the NC code and the corresponding measured time series for each MO. The Seq2Seq model is then trained with this dataset. The trained model can now be activated with an NC code and subsequently outputs a time series based on the NC code [25]. Furthermore, the model can be used within a hybrid simulation to predict, for example, the duration of a MO or the energy consumption of machines [22, 26].

The Seq2Seq method suffers from several issues:

1. Training Data:

The acquisition of training data is cost and time-intensive [13, 21]. This effect is particularly strong for ML methods, as the amount of training data is equated with improved learning performance [5]. As a solution, we propose the use of synthetic data [13, 21].

2. Lack of Metrics for Ambiguous Datasets

Training data for machine learning must generally be unambiguous. *Unambiguous* here means that there is exactly one solution for each sample in the training dataset. This condition is violated for training data that includes NC codes and time series, where slightly different time series are measured for the same NC code. We propose introducing a new learning metric that compares the generated time series at the end of a training period with a comparison time series.

3. No Comparison Time Series:

Comparing generated time series with time series from training data seems trivial at first glance. The problem lies in two points. First, during the training of a Seq2Seq model each data point in the time series is compared individually to one another [3, 16]. A comparison of time series in their entirety does not take place. Second, for ambiguous datasets, there is no single comparison time series that can be used to compare entire time series. We propose solving this problem with *Dynamic Time Warping* (DTW) [1, 15].

The following chapters will propose the fundamentals of the suggested method and a concept based on it to solve all three of those problems. The core of the concept is to generate several time series based on all possible NC codes after each training run. Subsequently, the generated time series are compared with reference time series created by DTW. This is done iteratively at the end of each training run. Therefore, this approach is referred to as *Iterating over Metrics* (IOM).

Next, the components necessary for the implementation of the concept will be explained in detail, and the results of the derived method will be presented.

1 Fundamentals of the IOM-approach

For better understanding, the methods used in the presented IOM- approach will be briefly explained.

1.1 Sequence-to-Sequence

Artificial neural networks (ANNs) are used to identify patterns in complex data structures. When patterns change over time, this temporal sequence of patterns is understood as a sequence.

To process temporal patterns, recurrent connections in ANNs are necessary, allowing feedback of abstracted knowledge [27]. Such feedback or recurrent neural networks (RNNs) are particularly suitable for data in sequential form [4].

When the data consists of sequences, these are referred to as Sequence-to-Sequence (Seq2Seq) architectures. The input sequence is encoded through the input layer of an ANN.

When the input sequence is encoded into a neural layer, it is called an encoder. When a target sequence is generated from a neural layer, this part is referred to as a decoder [4].
Such a topology is also commonly referred to as recurrent encoder-decoder networks (RNN-ED) [4]. Figure 1 depicts an ANN for the application of NC codes and time series.

During training, all data of a sequence pair $\{X_i, Y_i\}$ are processed sequentially and mapped to each other. First, the input sequence X_i is encoded into a so-called context vector *C*. This context vector is then decoded into the output sequence Y_i.

The ML model learns the progression of Y_i by updating the state of the context vector for each entry $y_{i,T}$.

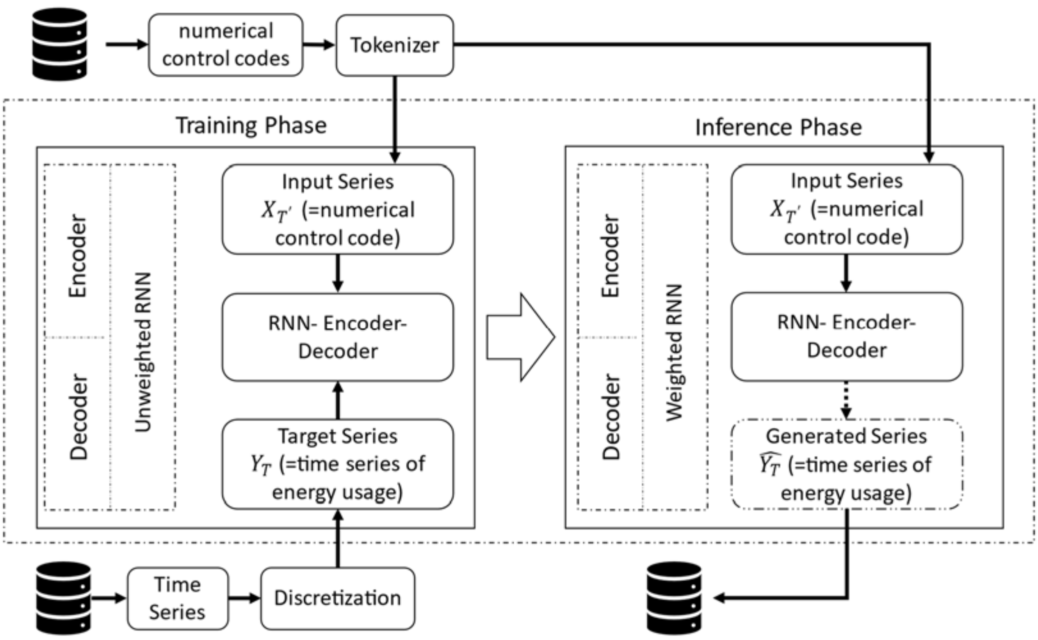


Figure 1: Components of an RNN Encoder-Decoder Topology.

The change in the state of the ANN is determined by a categorical cross-entropy loss function. The model then attempts to minimize this error [4].

The error value describes the difference between the target y_i and the predicted result $\hat{y}_i$. The prediction $\hat{y}_i$ is made in the final layer of the ANN, which uses a so-called *softmax* activation function. However, this has the drawback that it can only compare one entry of y_i with the generated time series at a time [4].

Sequence-to-Sequence models led to the development of *Large Language models* (LLM). This was made possible through the evolution from encode-decoder architectures to *Transformer* architectures [28].

Further explanations of the encoder-decoder network used here can be found in [3, 4, 16].

1.2 Synthetic data from Simulation

A fundamental problem in applying deep learning is the availability of training data. This data must be collected, checked for inconsistencies, preprocessed with analytical methods, and so on. This process often involves significant costs and time. This is particularly evident in the example of the dataset of NC codes and energy consumption time series for manufacturing orders.

If the machine tool in question does not have a suitable interface, the NC code and the period during which it operates must be manually recorded and transferred into a data format. The creation of the energy time series dataset is even more complex.

It requires identifying, implementing, and ultimately applying suitable measurement technology and software. This demands significant resources, including available personnel, as well as the availability of machine tools and measurement equipment.

Once the prerequisites for recording the NC code and time series are met, the next question is whether the machine tool in the observed period has sufficient utilization to generate a large number of training data points. Given the size of datasets typically used in deep learning (>10k), this seems unlikely.

Synthetic data can address this issue. Synthetic data are artificially generated data that resemble the original data in the relevant structure (length, features, feature frequency, etc.). The advantage of synthetic data is that they can be created cost-effectively, transparently, and reproducibly. Synthetic data can be generated through a simulation model and then used as a training basis for ML models.

The use of synthetic data in machine learning is not new and is an established research field [20]. For example, *Melo et al.* [13] demonstrated that synthetic data can be used to train image recognition models. Other applications of synthetic data in deep learning include non-destructive testing of steel [2], object detection [23], creating vehicle boundary frames for autonomous driving [20], and pedestrian detection in image data [7].

Creating synthetic data through simulation techniques is also not novel. Data farming has been used to create a data basis for deep learning in the simulation of production systems [9] or object detection for robots [19]. Additional applications include generating mobility data [8, 12], image data for heart tissue determination [10], or data for manufacturing planning and control [6].

1.3 Dynamic Time Warping

Time series describe the progression of a feature over time, typically for a fixed measurement interval, meaning the same interval between individual data points. As a result, a time series is essentially a collection of temporally ordered data points.

Comparing time series might initially seem like a trivial problem. For instance, if comparing time series of a fixed period, such as the temperature over a day, one could simply overlay the time series of two days, calculate their average to determine the mean temperature, or subtract them to find the temperature difference between two days.

However, comparing time series that do not follow a fixed period is more complex. For example, in the case of time series for the energy consumption of a MO. Machine tools are dynamic systems. This means their state changes with each event, resulting in different appearances for time series of the same NC code and the same machine. The differences can be observed in both dimensions of the time series: the number of data points and the feature values of the time series.

When comparing time series from dynamic systems, the following issues arise:

Firstly, the time series will have a different number of data points. The length of a time series is determined by its number of data points. If there is a discrepancy, the additional data points in one time series cannot be compared with the data points of the other time series because they are missing. If one were to remove the additional data points, it would make the time series comparable again, but potentially important data points could be lost.

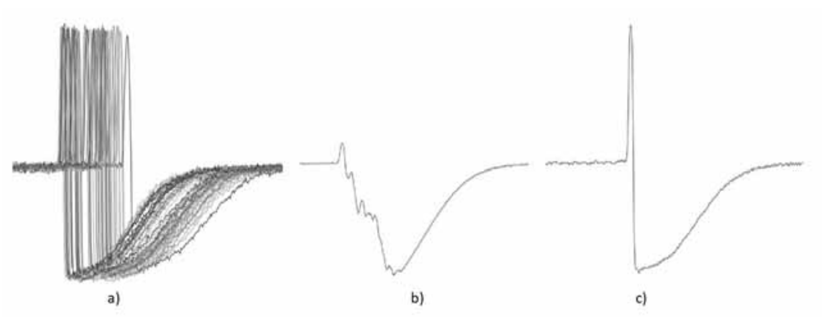


Figure 2: Comparison of a) a set of time series, b) the mean of that set and c) a time series that was generated from the set via DTW (Figures taken from [14]).

Alternatively, one could fill the shorter time series with values until it matches the length of the longer time series. The challenge here is finding a value that can fill the shorter time series without adversely affecting it.

Secondly, the characteristic patterns in the time series might shift due to missing data points. For example, these patterns could be a daily temperature peak at noon or, as in the mentioned case, recurring patterns in the tension profile of a machine tool.

Considering Figure 2, the above-mentioned problems are further illustrated. Figure 2 a) shows a set of time series, all of which are slightly different in length and whose characteristics (such as slope, gradient, and return to the starting value) differ in both position and magnitude. If the time series were simply filled with values (e.g. the mean of the time series), one could compute the mean of all the time series. Figure 2 b) shows the result of this averaged time series. It is evident that this averaged time series, except for its length, is no longer comparable to the original time series.

A method that can compare time series of different lengths with varying positions of their characteristics is called *Dynamic Time Warping* (DTW) [1, 15]. DTW is an algorithm that compares a set of time series using a so-called distance matrix.

The distance matrix compares the data points of two (or a set of) time series using a distance measure, e.g. *Euclidean* distance, *Manhattan* distance, etc. When comparing two data points, their difference is recorded in the distance matrix.

Then, a path is drawn through the distance matrix, known as the *Warping Path* (see the line in Figure 3). The optimal *Warping Path* is that path that, in total, has the shortest distance through the distance matrix compared to other paths.

If the data points are close to each other, the *Warping Path* increases uniformly in each dimension (see Figure 3, *Warping Path* – bottom left). If the data points of two time series deviate from each other, the *Warping Path* either flattens out or rises sharply.

Once a *Warping Path* is found, it can be used to create an averaged time series through DTW. For example, Figure 2 c) shows a time series that is clearly comparable in length and shape to the individual time series from Figure 2 a).

Further literature on Dynamic Time Warping can be found in [1, 14, 15, 17].

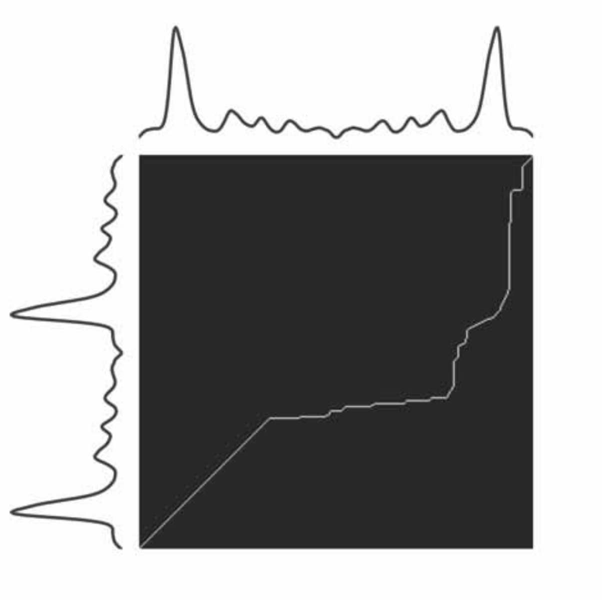


Figure 3: Distance matrix of two time series compared using *Dynamic Time Warping* (shown on the left and at the top) [17].

2 IOM-Concept

The IOM- approach consists of several steps (compare Figure 4). First, to address the cost and availability issues of training data, it will be generated using a simulation model.

In the second step, reference time series are created. These are used to make the time series generated by the model comparable.

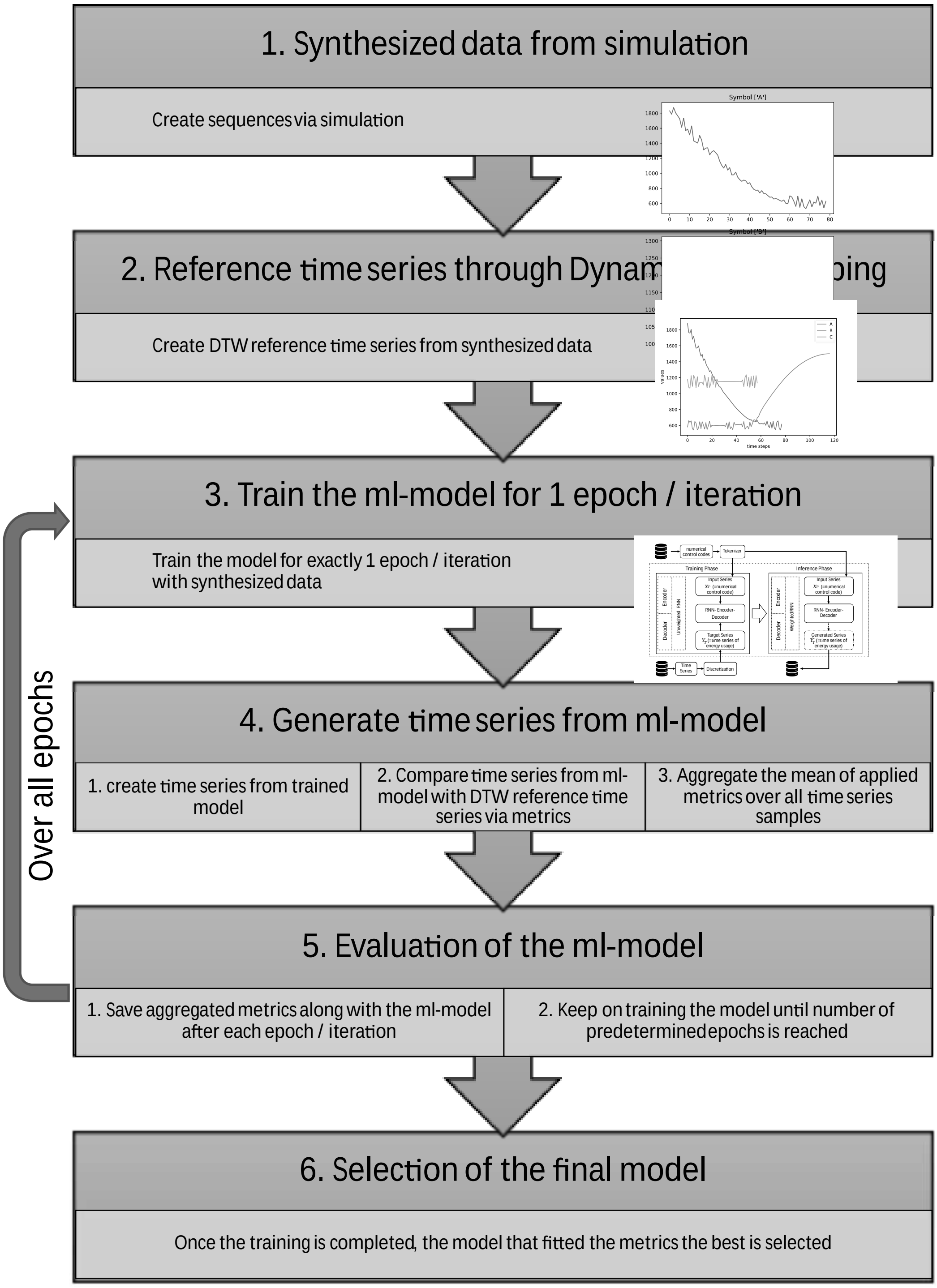


Figure 4: Concept of the IOM-approach

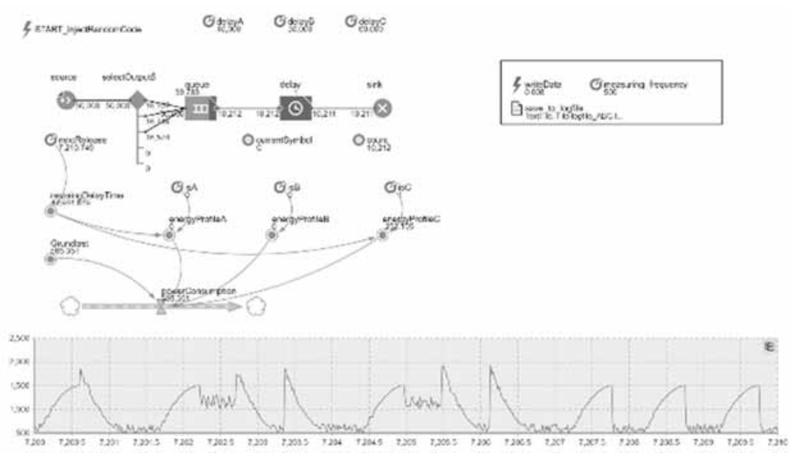


Figure 5: AnyLogic Simulation Model.

This is done using *Dynamic Time Warping*, with the data foundation being the previously generated synthetic data that will also be used for training.

Third, the training of the ML model is initiated. The model is trained for exactly one iteration (Note: To reduce computation limitations, 10 epochs are combined into one iteration). In the fourth step, the trained model is used to generate time series.

Comparing entire time series is necessary to address the problem described in Chapter 1.1, where the model's *softmax* function only compares individual data points step by step, not entire time series.

The quality of the generated time series is then assessed by comparing them with the reference time series produced by DTW. After metrics comparisons are made for all considered time series, these are aggregated. The average of each individual metric across all time series is computed.

In the subsequent fifth step, the model undergoes further training and repeats steps 3 and 4 until a predetermined number of epochs is reached. Again, a summary of metrics is created in each iteration, and the trained model is saved. In the sixth and final step, the model that achieved the best results in the evaluated metrics is selected.

3 Experimental Preparation

To implement the IOM- concept, several preparatory steps are necessary. For once there is the generation of synthetic time series data through the simulation model. Additionally, the creation of comparison time series using *Dynamic Time Warping*. Finally, the metrics defined in the previous chapter must be developed and integrated into the model training process.

3.1 Creation of synthetic time series data

To create synthetic time series, the simulation tool *AnyLogic* (see Figure 5) is used. In *AnyLogic*, a discrete-event simulation model is set up. In the model's source contains manufacturing orders (A, B, or C) that are generated randomly.

These orders enter a queue and then move to a waiting area. Only one (MO) can be in the waiting area at a time. Each MO is assigned its own waiting time, which follows a continuous uniform distribution of [0.95;1].

Once the waiting area is occupied by an MO a function associated with that MO is executed. These functions differ for each of the three MO types (see Figure 6).

They are mathematical functions that use the remaining waiting time of the MO as an input parameter. The outputs of these functions thus vary over time. The computed value is then altered through another uniform distribution function and finally output.

The uniform distribution in the waiting area and in the output functions ensures that the time series for the same type of MO will slightly differ each time. This addresses the problem described initially, where two measured time series of the same MO never match exactly in length and value progression. Thus, it is ensured that two MO with the same label are represented with comparable, but not identical time series. Figure 6 illustrates samples of the generated time series. The uniform distribution in the output values is clearly visible here.

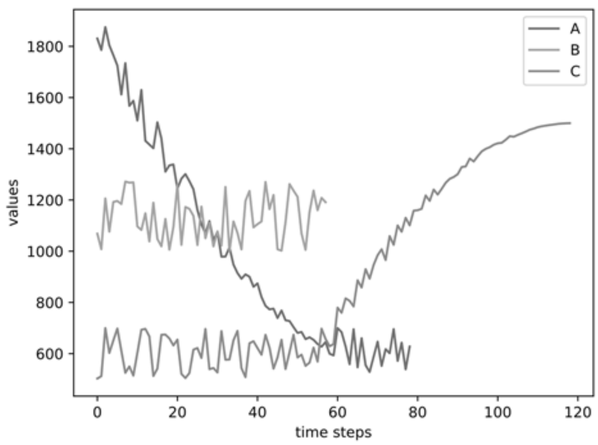


Figure 6: Sample of time series created by simulation model.

3.2 Generation of reference time series through Dynamic Time Warping

To create reference time series, *Dynamic Time Warping* is used. DTW generates a reference time series from a set of time series, which exhibits minimal overall error in length and structure compared to the original set.

The calculation is performed using the Python library *tslearn* [17]. The specific DTW algorithm used here is *softDTW* [11]. *softDTW* has a higher error tolerance in the creation of the *Warping Path*, which positively affects the generation of averaged time series [11].

Figure 7 shows a time series generated by *softDTW* compared to two samples of manufacturing order C. The *softDTW* time series effectively summarizes the samples but does not merely replicate them. Instead, it emulates their shape and length.

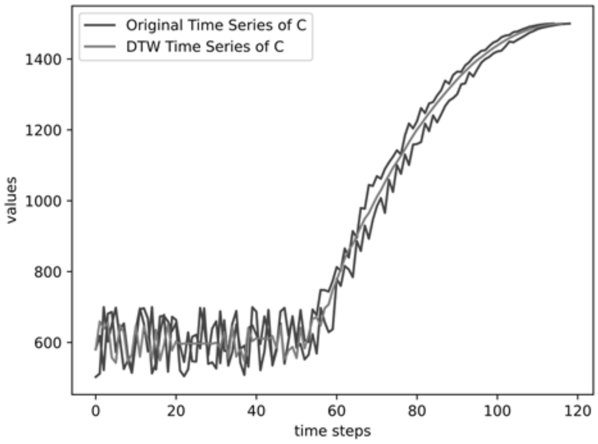


Figure 7: Comparison of 2 synthetic time series of type C with the time series created for type C through softDTW.

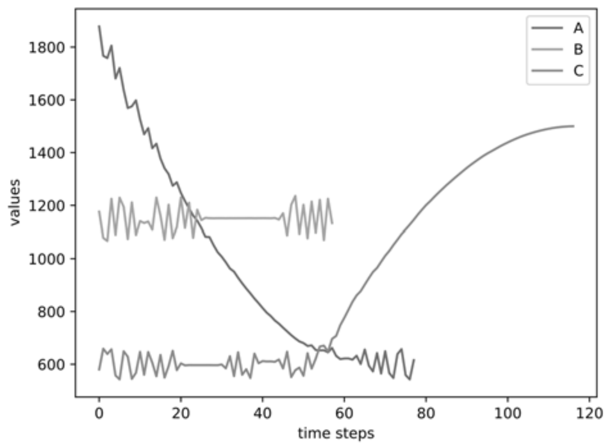


Figure 8: Sample of time series created via softDTW.

Figure 8 shows the *softDTW* time series for all three manufacturing orders. Compared to Figure 6, the difference between the original and DTW-generated time series is evident. The time series all exhibit a similar length and shape but differ slightly, as they are derived from the entire set of time series.

3.3 Implementation of metrics into the Sequence-to-Sequence Model

During model training, the generated time series are compared with the DTW-generated time series as a reference. Subsequently, the values of the metrics for the individual time series are averaged over an entire training epoch. Two metrics were used for this purpose.

First, the mean squared error (*MSE*) is applied. This compares the time series by calculating the difference between each individual data point and then squaring these differences.

All differences are then summed to provide a measure of the overall quality of the time series. Squaring the differences has two advantages. First, all differences become positive, which means that negative and positive error values do not cancel each other out in the subsequent addition. Second, squaring gives more weight to large errors compared to small ones.

The second metric, *sigma length*, considers only one dimension of the time series, more precisely its length. This metric calculates the number of data points in both time series and then divides one by the other. The closer the result is to 1, the more similar the generated time series is to the reference time series in terms of length.

```
def results(ml_ts, dtw_ts, iteration):
        mean_squared_error_accuracy =
        mean_squared_error(ml_ts, dtw_ts)
        sigma_length_accuracy =
        len(ml_ts) / len(dtw_ts)

        results =
        {"Iteration":iteration,
        "MSE":mean_squared_error_accuracy,
        "Sigma_Length":sigma_length_accuracy}
        return results
```

Code 1: Pseudocode of the MES and *sigma length* metrics to compare two time series. *ml_ts* marks the time series that was created through the machine learing model. *dtw_ts* is the reference time series that was created via DTW.

When a training run is completed, time series are generated with the trained model. These are then compared using the metrics described in Code 1, and the results for each time series are saved.

Next, a summary for the entire iteration is calculated. This aims to provide a comprehensive assessment of the model at a particular training stage. Hence, all *MSE* and *sigma length* results are averaged.

The mean allows for a more robust statement about the model's ability to generate time series than examining individual time series.

4 Experiment Execution

The creation of synthetic time series was carried out in *AnyLogic*. Python was used as the programming language. The calculation of DTW-generated time series was performed in *tslearn* [17], and the creation of the ML model in TensorFlow [18].

The hardware available included a Ryzen 7 2700 X processor, an RTX 2080Ti graphics card with 4352 CUDA cores, and a 480 GB SSD.

The IOM- approach involves saving the ML model after each epoch. However, this could not be implemented with the available storage capacity. Therefore, it was decided to save the trained the model after 10 epochs (here: equal to 1 iteration) and only collect metrics for the last epoch of an iteration.

The model was trained for 1000 epochs (subsequently 100 iterations).

The training data as described in Chapter 3 consisted of labeled sequence pairs formed from 2 manufacturing orders each to increase the variations in the dataset. The dataset was comprised of 10,000 labeled sequence pairs.

4.1 Comparison of metrics

As described in the IOM- concept, averaged metrics are to be generated at the end of each iteration. The progression of these metrics is plotted in Figure 9. The results of *MSE* and *sigma length* are shown.

To select the optimal model, the optimum of both metrics is determined. The optimum of a metric is the value at which the metric achieves the best result. For *MSE*, this would be 0, as *MSE* considers the difference between all data points. For *sigma length*, it would be 1, as this metric examines the ratio of the length of one time series to another.

To illustrate, the optimum for *sigma length* is graphically represented (blue line in Figure 9). All iterations that lie on the blue line represent models that have achieved the optimal value in the *sigma length* metric.

In a second step, iterations where *sigma length* $= 1$ are examined for their value in the *MSE* metric. The green line in Figure 9 at iteration 91 (=epoch 910) corresponds to the iteration where the values for both metrics are the lowest. Therefore, the model from epoch 910 is therefor selected.

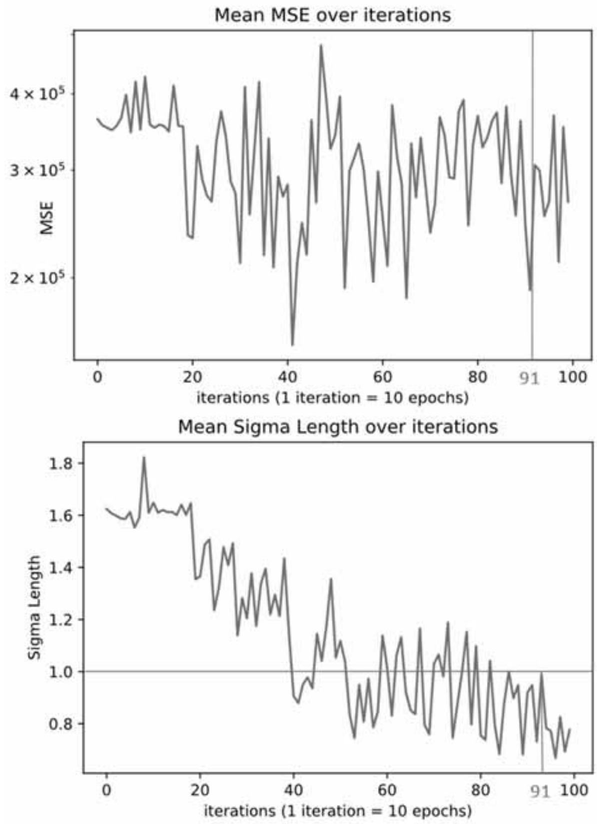


Figure 9: Metrics of the IOM Approach. Top: Averaged *MSE*. Bottom: Averaged *sigma length*. The blue line marks the optimum of *sigma length* (=1). The green line indicates the selected iteration.

4.2 Comparison of metrics

The model can now be used to generate time series. Figure 10 shows all 9 possible variants on which the ML model was trained.

The blue time series was generated by the ML model. The orange time series is the reference time series created by DTW. The visual comparison confirms the functionality of the IOM- approach.

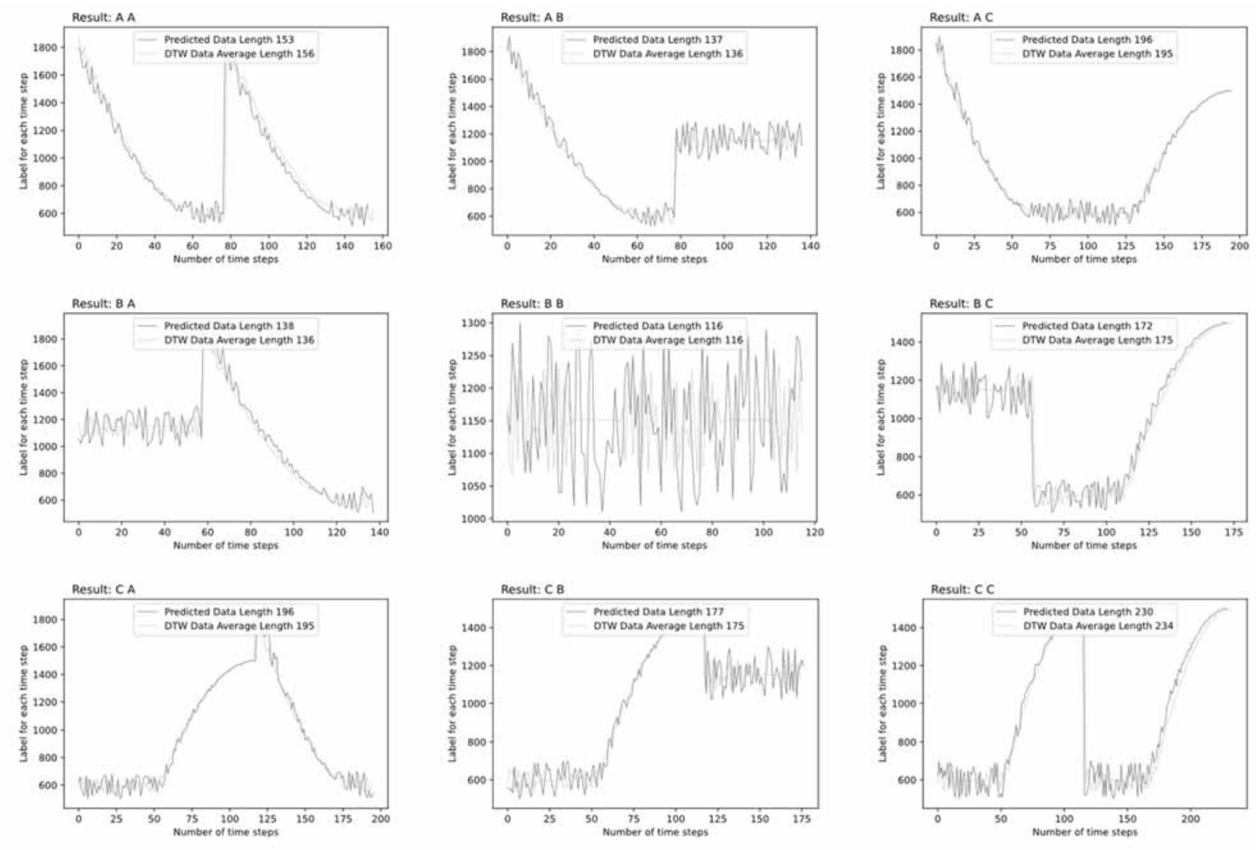


Figure 10: Comparison of all time series created at epoch 910 with the DTW reference time series. The legend in the variants indicates the lengths of the two time series.

5 Critical Review

The presented IOM- approach was able to meet the requirements set for it.

On the one hand, it was shown that synthetic data generated through simulation can be used for training ML models. These synthetic data could be generated cost-effectively, transparently, and appropriately for the specific application.

Furthermore, the implementation of specialized metrics ensured that entire time series could now be compared, not just individual data points via the *softmax* function as in the Seq2Seq base model.

It was subsequently demonstrated that DTW time series can be used to form reference time series for comparison via the metrics. The use of DTW was necessary to address the issue of ambiguity in sets of time series with identical descriptions (e.g., the same NC code).

Finally, it was shown that time series can be generated that hardly differ from the analytically generated *Dynamic Time Warping* time series. The time series are almost identical in form and length.

The method itself offers potential for further research questions. For instance, it could be further validated by significantly enlarging the dataset or generating several different manufacturing orders in the simulation.

Additionally, the ML model here only generated time series from NC codes that were already present in the training data. It remains to be investigated to what extent the model can generate time series in the case of an unknown NC code. This is a general challenge for all generative AI applications as they lack labels for unlabelled data to compare its output to.

A disadvantage of the presented method is the requirement for computational and storage resources. A model must be saved for each iteration, although after determining the optimal model, all others can be deleted. This drawback can be mitigated by applying the comparison of metrics during the training and not just once afterwards. This way models that underperform could be identified and removed sooner. Also, the calculation of the metrics exceeds the time spent here for training the model when done on an epoch-by-epoch basis. This offers room for improvement aswell.

References

[1] Berndt DJ, James Clifford J. Using dynamic time warping to find patterns in time series. 1994, in KDD workshop, 359–370.

[2] Boikov A, Payor V, Savelev R, Kolesnikov A. Synthetic Data Generation for Steel Defect Detection and Classification Using Deep Learning. 2021, Symmetry 13, 7, 1176. DOI 10.3390/sym13071176.

[3] Cho K, van Merrienboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. 2014, in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, Doha, Qatar.

[4] Goodfellow I, Bengio Y, Courville A. 2016, Deep Learning. MIT Press.

[5] Hestness J, Narang S, Ardalani N, Diamos G, Jun H, Kianinejad H, Patwary MA, Yang Y, Zhou Y. 2017, Deep Learning Scaling is Predictable, Empirically.

[6] Jain S, Narayanan A, Lee YT. Infrastructure for Model Based Analytics for Manufacturing. 2019, in 2019 Winter Simulation Conference (WSC). IEEE. DOI 10.1109/wsc40007.2019.9004893.

[7] Ekbatani HK, Pujol O, Segui S. Synthetic Data Generation for Deep Learning in Counting Pedestrians. 2017, in Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods. SCITEPRESS - Science and Technology Publications. DOI 10.5220/0006119203180323.

[8] Kutyev AT, Bekeneva YA. Simulation Software for Generating Data in Monitoring Systems. 2021, in Proceedings of the 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus). January 26-28, 2021 : St. Petersburg and Moscow, Russia, 2021. IEEE, Piscataway, NJ, 486–489. DOI 10.1109/ElConRus51938.2021.9396098.

[9] Lechler T, Sjarov S, Franke J. Data Farming in Production Systems - A Review on Potentials, Challenges and Exemplary Applications. 2021, Procedia CIRP 96, 230–235. DOI 10.1016/j.procir.2021.01.156.

[10] Loecher M, Perotti LE, Ennis DB. Using synthetic data generation to train a cardiac motion tag tracking neural network. 2021, Medical image analysis 74, 102223. DOI 10.1016/j.media.2021.102223.

[11] Cuturi M, Blondel M. Soft-DTW: a Differentiable Loss Function for Time-Series. 2017, in Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research. PMLR, 894–903.

[12] Martí P, Jordán J, Palanca J, Vicente Julian V. Charging stations and mobility data generators for agent-based simulations. Neurocomputing 484, 2022, 196–210. DOI 10.1016/j.neucom.2021.06.098

[13] de Melo CM, Torralba A, Guibas L, DiCarlo J, Chellappa R, Hodgins J. Next-generation deep learning based on simulators and synthetic data. 2022, Trends in cognitive sciences 26, 2, 174–187. DOI 10.1016/j.tics.2021.11.008

[14] Petitjean F, Forestier G, Webb GL, Nicholson AE, Chen Y, Keogh E. Faster and more accurate classification of time series by exploiting a novel dynamic time warping averaging algorithm. 2016, Knowl Inf Syst 47, 1, 1–26. DOI 10.1007/s10115-015-0878-8.

[15] Petitjean F, Forestier G, Webb GL, Nicholson AE, Chen Y, Keogh E. Dynamic Time Warping Averaging of Time Series Allows Faster and More Accurate Classification. 2014, in 2014 IEEE International Conference on Data Mining, 470–479. DOI 10.1109/ICDM.2014.27.

[16] Sutskever I, Vinyals O, Quoc LV. Sequence to Sequence Learning with Neural Networks. 2014, in NIPS'14: Proceedings of the 27th International Conference on Neural Information Processing Systems. MIT Press, Cambridge, MA, USA.

[17] Tavenard R, Faouzi J, Vandewiele G, Divo F, Androz G, Holtz C, Payne M, Yurchak R, Rußwurm M, Kolar K, Woods E. Tslearn, A Machine Learning Toolkit for Time Series Data. 2020, Journal of Machine Learning Research 21, 118, 1–6.

[18] TensorFlow Developers. TensorFlow, 2022, Zenodo.

[19] Tobin J, Fong R, Ray A, Schneider J, Zaremba W, Abbeel P. Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World, 2017.

[20] Tremblay J, Prakash A, Acuna D, Brophy M, Jampani V, Anil C, To T, Cameracci E, Boochoon S, Birchfield S. Training Deep Networks with Synthetic Data: Bridging the Reality Gap by Domain Randomization. 2018.

[21] Triastcyn A, Faltings B. Generating Artificial Data for Private Deep Learning. 2018. http://arxiv.org/pdf/1803.03148v3

[22] Woerrlein B, Strassburger S. A Method for Predicting High-Resolution Time Series Using Sequence-to-Sequence Models. 2020, in 2020 Winter Simulation Conference (WSC). IEEE, 1075–1086. DOI 10.1109/WSC48552.2020.9383969.

[23] Wong MZ, Kunii K, Baylis M, Ong WH, Kroupa P, Koller S. Synthetic dataset generation for object-to-model deep learning in industrial applications. 2019. *PeerJ. Computer science* 5, e222. DOI 10.7717/peerj-cs.222

[24] Woerrlein B, Strassburger S. On the Usage of Deep Learning for Modelling Energy Consumption in Simulation Models. 2020, SNE 30, 4, 165–174. DOI 10.11128/sne.30.tn.10536.

[25] Wörrlein B, Strassburger S. Sequence to Sequence Modelle zur hochaufgelösten Prädiktion von Stromverbrauch. 2020, in Proceedings ASIM SST 2020. ARGESIM Publisher Vienna, 149–157. DOI 10.11128/arep.59.a59021.

[26] Wörrlein B, Strassburger S. Hochaufgelöste Energieprofile durch hybride Simulation. 2022, in ASIM SST 2022 Proceedings Langbeiträge. 26. ASIM Symposium Simulationstechnik, 25.07.-27.07.2022, TU Wien. ASIM Mitteilung, 180. ARGESIM Verlag, Wien, 243–251. DOI 10.11128/arep.20.a2004.

[27] Zell A. Simulation neuronaler Netze. 2003, 4. unveränd. Nachdruck, Oldenbourg, München

[28] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I. Attention Is All You Need. DOI 10.48550/arXiv.1706.03762

Design of a Model Library for the Low-Cost Functional Development of Mechatronic Systems

Sven Jacobitz*, Jie Zhang, Xiaobo Liu-Henke

Institute for Mechatronics, Ostfalia University of Applied Sciences, Salzdahlumer Str. 46/48, 38302 Wolfenbüttel, Germany; *sve.jacobitz@ostfalia.de

SNE 34(4), 2024, 215-223, DOI: 10.11128/sne.34.tn.10714
Selected ASIM WS 2023 Postconf. Publication: 2023-10-15
Rec. Improved English V.: 2024-08-12; Accepted: 2024-09-20
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. The increasing complexity of mechatronic systems demands a structured and systematic approach to their development, thereby highlighting the importance of Rapid Control Prototyping (RCP) as an essential methodology.
Consequently, it is imperative to have a comprehensive Computer Aided Engineering (CAE) platform to facilitate support throughout the development process. Traditional CAE platforms, however, are often prohibitively expensive, leading to the development of LoRra at Ostfalia as a low-cost alternative.
Central to this paper, the LoRra model library supports the entire RCP development process through robust data management that includes advanced version and configuration management capabilities. By utilizing a systematic, hierarchical organization of models within a tree-like structure and enabling multi-user access, the library promotes consistent and traceable development processes. This framework not only enhances the reusability and reliability of development artifacts but also lowers barriers to adopting rapid control prototyping methodologies for small and medium-sized enterprises (SMEs).
This paper discusses the design, functionality, and implementation strategies of the LoRra library, demonstrating its potential to significantly influence the future of mechatronic system development by improving accessibility and systematic management.

Introduction

The complexity and functional scope in mechatronic systems is constantly increasing. This trend represents a major challenge for small and medium-sized enterprises (SMEs). To remain competitive, they must integrate more and more intelligent hardware and software into their products. This can be attributed to the increasing number of functions, as well as the growing number of complex and interdependent software components [1].

A structured, systematic development of such systems is indispensable to handle ever shorter development times with higher quality requirements [2]. Holistic model-based Rapid Control Prototyping (RCP) methodology that is frequently employed in this context. It is imperative that a Computer Aided Engineering (CAE) platform provides seamless support in order to achieve a high degree of automation, which is a fundamental aspect of RCP. The cost of established CAE platforms represents a significant barrier to the introduction of the RCP process, particularly for small and medium-sized enterprises (SMEs). As part of the EU-funded research project *Low-Cost Rapid Control Prototyping-System with Open-Source Platform for the Functional Development of Embedded Mechatronic Systems (LoCoRCP)*, the low-cost development platform LoRra was created at Ostfalia [3].

In order to guarantee the reusability of models and functions, a systematic approach to data management is employed, encompassing associated sub-processes such as version and configuration management. This is indispensable due to the high diversity, flexibility, and short lifetime [4]. The initial regulatory framework for this was established at NASA in the early 1960s, as documented in [5]. As Sax et Al. [6]observe, inconsistencies in function development are increasingly caused by the high number of variants. This issue can be avoided by using appropriate configuration management. In the context of RCP, this requires a CAE-based model library that manages all relevant artifacts (such as models, program source code, or documentation) [7].

Version and configuration management are widely utilized in the domain of classical software development. For an overview, see [8]. One of the most commonly utilized open-source tools in versioning is GIT [9]. Nevertheless, the primary objective of this tool is to facilitate the change-based management of text files [10].

Nevertheless, the application of this change-based approach to data formats commonly utilized in RCP is not particularly practical, as evidenced by [11]. Consequently, it is essential to implement suitable adaptations for utilization in a model library.

To identify the requisite adaptations, Kruse and Shea conducted systematic investigations [7]. In this approach, a model is constructed in a standardized way from metadata, interface information, and parameters. Based on such a standardized structure, version and configuration management for complex, composite models can also be carried out using a system entity structure (SES) [12].

In the following, the CAE-based LoRra model library is conceived, designed and realized as an example. The rest of the paper is structured as follows: Section 1 summarizes the RCP-development methodology and introduces the LoRra-platform. In section 2, the basics of the solution are designed based on an analysis of the requirements, which is then concretized into a design in section 3. Finally, a approach of realization is provided in section 4. Section 5 summarizes the results and provides an outlook for future work.

1 Development Methodology and Platform

Due to the high system complexity of modern interconnected mechatronic systems, the structured, model-based, verification-oriented RCP process is used for development and validation. This comprises the following process steps: modeling, analysis/synthesis, automated generation of source code, automated implementation on real-time hardware, and online experimentation. Model-in-the-Loop (MiL), Software-in-the-Loop (SiL), and Hardware-in-the-Loop (HiL) simulations are employed to support this methodology [13].

The methodology delineated in this work is distinguished by its high level of consistency and automation, encompassing modeling, model-based functional design, automated code generation, and real-time implementation, as depicted on the left side of Figure 1. This approach is supported by a fully automated, seamless CAE platform. Specifically, the LoRra development platform represents a modular, low-cost example of such a CAE system.

Figure 1 demonstrates the RCP development process and the comprehensive integration facilitated by LoRra (cf. [3]).

Central to this plattform is a model library that ensures a consistent and traceable development status throughout all phases of the development process.

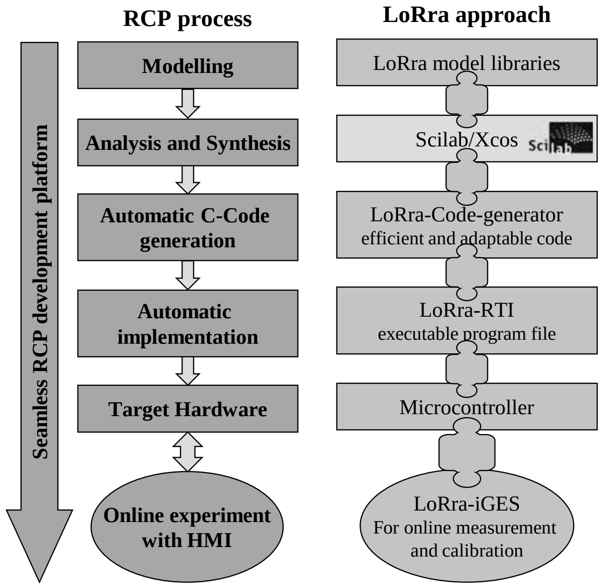


Figure 1: RCP development process with seamless support by the LoRra platform [3].

A comprehensive multi-domain model library is utilized throughout the modeling process. This library facilitates the clear assembly and management of model variants through robust version and configuration management systems. The open-source CAE tool Scilab/Xcos is employed for the analysis and synthesis of functions, offering functionalities comparable to those found in the commercially frequently utilized Matlab/Simulink.

Subsequently, the resultant functional model can be seamlessly integrated into the model library. The LoRra Application Programming Interface (API)'s open interfaces enable straightforward incorporation of additional modeling programs and interface drivers, enhancing the system's versatility. MiL simulations are leveraged to optimize and validate the functions at an early developmental stage.

The LoRra code generator implements model-to-text transformation techniques to automatically produce efficient, modular C source code derived from the functional model. This generator is designed with open functional descriptions of the model's basic elements, called basic blocks, allowing for flexible extensions.

The resulting source code can be reintegrated into the Xcos model for further optimization and testing via SiL simulations without the need for manual intervention [14].

Upon achieving a satisfactory functional status, the interfaces with the controlled system models or other functions are substituted with blocks from the LoRra Real-Time Interface (RTI). This substitution facilitates the configuration and programming of real-time hardware without manual coding.

The hardware-specific RTI framework includes a real-time operating system and standardized interface drivers, enabling automated deployment on real-time hardware platforms such as low-cost microcontrollers, for instance, those from the STM32H7 series. HiL simulations are employed to optimize and evaluate the developed functions under real-time conditions [15]. Additionally, the integrated graphically-supported experimentation software (iGES) serves as an intuitive Human-Machine Interface (HMI) for controlling, monitoring, and recording data during online experiments.

2 Conception of the Library

This section is dedicated to the concept of the model library and begins with an analysis of the requirements. Various development approaches for graphical user interfaces (GUIs) are then examined.

A basic solution approach is developed on the basis of these analyses. The detailed consideration of these aspects lays the foundation for the subsequent steps in the design and realization of the model library, described by the next sections.

2.1 Development Approaches for Graphical User Interfaces

Standardized architectural styles play a pivotal role in structured software design, facilitating reusability, organizing design processes, and establishing a uniform vocabulary [16]. Notably, over a quarter of these styles are employed in designing user interfaces [17]. The Model/View/Controller (MVC) principle is particularly significant in the context of this work.

Based on [18], the MVC architectural style is depicted in Figure 2. This framework segregates the visualization (View), control, and data model into distinct components with clearly defined interfaces. The control component responds to user interactions within the GUI and modifies the model as needed.

Additionally, the model may be altered by other software components. It communicates any modifications to the control component, enabling the latter to refresh the view. This architecture's low coupling between components makes it especially apt for HMIs that operate across various platforms [19]. For instance, the graphical visualization specific to an operating system can be entirely isolated from both the controller and the model, enhancing adaptability and maintainability.

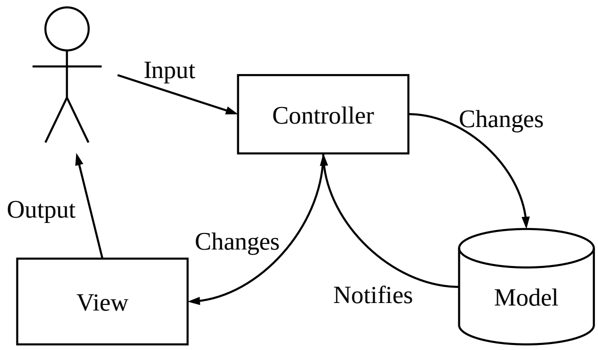


Figure 2: Idea of the architectural style Model/View/Controler according to [18].

Applying his MVC principle ensures a modular reusable software design.

2.2 Requirements on the Model Library

As delineated in section 1, the model library serves as a crucial instrument for data management throughout the development process. To fully enhance version and configuration management, the model library must adhere to several stringent requirements:

1. **Version Control:** It must support comprehensive versioning of all data it contains, including functionalities for version management processes such as checking and releasing versions.
2. **Configuration Management:** The library should facilitate the necessary data structures for configuration management across all stages of the development process, along with associated processes (e.g., checking, release) to maintain consistent data status throughout these stages.
3. **Hierarchical Organization:** Models should be categorically and hierarchically structured, allowing for configurable categories and levels that enhance organization and retrieval.

4. **Search Capability:** A robust search function is essential to efficiently locate specific models within the library.
5. **Collaboration Features:** The library should support collaborative work in distributed teams, enabling seamless access and modification of a shared model base.
6. **Comprehensive Display of Information:** It should provide a clear and detailed overview of all relevant model information to facilitate easy access and understanding for all users involved in the project.

These requirements are essential to ensure that the model library effectively supports the intricate and dynamic needs of modern software development environments.

2.3 Basics of the Solution

To fully address the requirements of the model library, it is imperative to examine the structure of a model in greater depth. The distinction between generic and aggregated models forms a crucial part of this analysis.

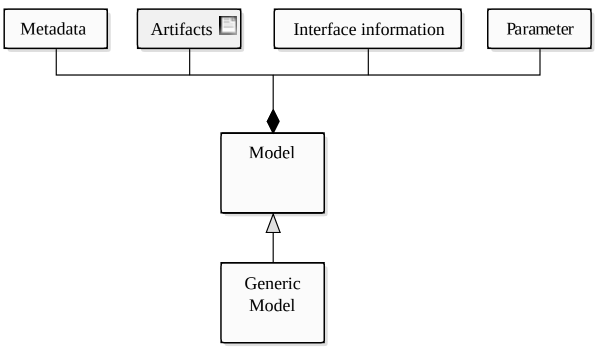


Figure 3: Structured setup of a generic model.

Generic Model: This represents the smallest coherent unit within the model library and operates at the lowest level of hierarchy. It is indivisible and does not contain any submodels structured hierarchically below it. An example of a generic model is the electrical part of a DC motor, represented by equation (1). Figure 3 visually details the structure of a generic model, which encompasses the following components:

$$u = Ri + L\frac{\mathrm{d}i}{\mathrm{d}t} - u_i \quad (1)$$

- **Metadata:** These attributes detail overarching characteristics such as the model's name, author, and a general description.
- **Interface information:** This includes the data structure, units, and other pertinent details about the model's inputs and outputs. For instance, using equation (1), the terminal voltage u in volts (V) as input and the motor current i in amperes (A) as output.
- **Parameters:** Specifications and values for the model's parameters, such as the resistance R in ohms (Ω) and the inductance L in henries (H).
- **Artifacts:** These may include the simulation file (e.g., in Xcos format), generated C code, or documentation associated with the model.

Aggregated Model: This type of model comprises multiple sub-models and represents a higher hierarchical level within the library. Aggregated models are structured as configurations in the model library, formed by integrating specific versions of part models. The concept of assembling these configurations is illustrated in Figure 4.

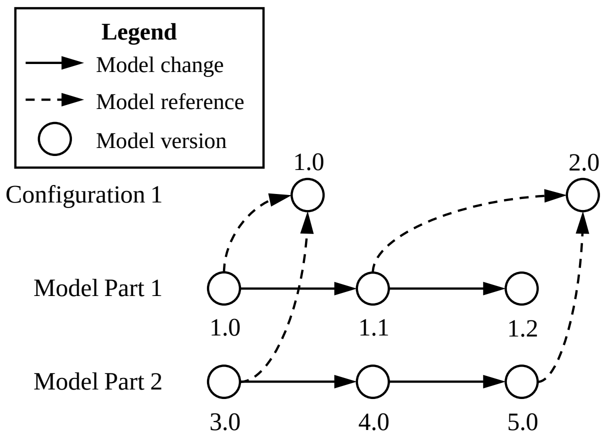


Figure 4: Basic concept of assembling a configuration.

In terms of organization, models should be structured hierarchically in a tree-like format within the library. This structure includes folders (grouping hierarchy element) and model elements (both generic and aggregated models). User rights management is crucial, allowing permissions to be assigned both to groupings and individual model elements.

To facilitate multi-user access, a central storage principle is utilized, as shown in Figure 5. A local working copy is employed to access and modify model artifacts. Any changes made are then synchronized back to the central repository, which functions as a comprehensive database.

Following synchronization, users can update their local copies with the modified data, ensuring that all team members have access to the most current versions of models and configurations.

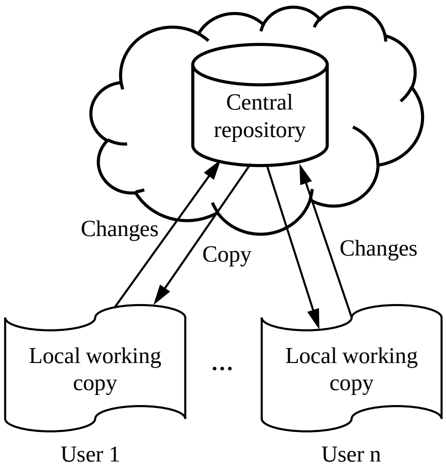


Figure 5: Concept for the central storage of models.

Employing this approach enables the systematic design of a model library seamlessly integrated into the RCP process. This integration facilitates efficient and coherent management of model data throughout the development cycle.

3 Design of the Library Functions

Building on the concepts introduced in section 2, this section provides a detailed elaboration of the proposed system.

Initially, we examine the data structures and interfaces, as well as the mechanisms for data management. Subsequently, the design of the GUI is addressed to ensure a seamless and efficient user experience. This structured approach ensures that both the backend and frontend components of the system are robustly developed and integrated.

3.1 Data Structures

The design of the model library requires intricate data structures and interfaces to support its functionality. At the core of the library is the hierarchical model tree, which also serves as the foundational database for the GUI developed according to the MVC principle. This section elaborates on the design of the data structure and interfaces that underpin the model tree.

The data structures are designed using object-oriented principles. The abstract class *AModelElement* acts as the foundational structure for each element within the tree, encapsulating essential data such as the element's title, its path within the tree, and its parent element.

Derived from *AModelElement* are two key classes: *HierarchyElement* and *ModelElement*.

- **HierarchyElement:** This class manages a collection of subordinate elements, effectively organizing them within the tree structure to facilitate navigation and management.
- **ModelElement:** This class is more specialized and includes detailed interface information, parameters, and storage details of the model, among other pertinent data. This encapsulation ensures that each model within the library is both self-contained and richly described for easy access and modification.

Figure 6 provides a visual representation of these relationships through a UML class diagram, illustrating how these elements are interconnected to form a robust and scalable model tree structure. This structure is critical for supporting the complex interactions and data management requirements of the model library in the RCP process.

The object-oriented design of the data structures ensures that the model library is both scalable and adaptable, supporting complex hierarchies and detailed model management. This architectural choice enhances functionality and integration within the RCP process, facilitating efficient navigation and modification of models.

3.2 Data Management

The data management within the model library primarily hinges on robust version and configuration management systems.

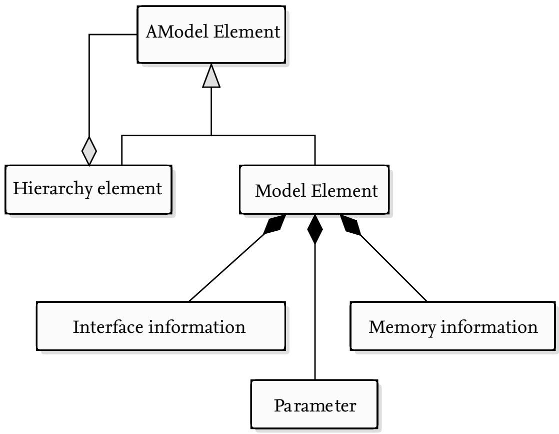


Figure 6: UML diagramm of the hierarchical model tree.

To facilitate consistent versioning and thereby enable structured reuse in configurations, the library must support a version management process that includes the formal release of new versions. Modifications to a model necessitate its reincorporation as a new version following company-specific approval procedures. For the LoRra model library, versions suggested by users are only made publicly available after obtaining consent from designated groups, as mandated by organization-specific protocols.

Version identification within the model library utilizes a semantic versioning approach by the numerical format, x.y, where 'x' represents the major version and 'y' the minor version.

Minor version increments occur when updates (such as bug fixes) do not alter model compatibility (i.e., behavior and interfaces remain unchanged). Conversely, major version increments – accompanied by resetting the minor version to zero – are employed when changes impact model compatibility, such as modifications to interfaces or functionality enhancements. Models are integrated into configurations by referencing these version numbers, as depicted in Figure 4, which outlines this versioning principle.

Effective data management within the model library is achieved through meticulous version and configuration management, ensuring that models are consistently updated and released in alignment with defined organizational processes. This systematic approach supports the seamless integration and reuse of models across various configurations, enhancing the library's utility and reliability.

3.3 Graphical User Interface

The GUI of the LoRra model library is designed in accordance with the MVC principle, as outlined in section 2.1.

The foundation for this design (the model component) is the hierarchical model tree developed in section 3.1, complemented by various control classes tasked with specific functions.

Figure 7 depicts the comprehensive layout of the GUI. The main window is segmented into three primary parts:

- **Navigation Area:** This section hosts the hierarchical model tree, enabling users to perform various actions such as opening or editing models, and provides an initial overview of the current model status. It also includes a search function for navigating the model tree.
- **Display Area:** This part offers different views for the presentation and modification of information, including metadata, model artifacts, and version history.
- **Tool Area:** This section features a context-sensitive toolbar and a menu structure designed to facilitate the operation of the library.

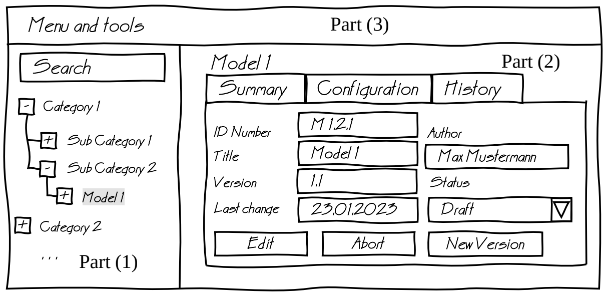


Figure 7: Draft of the model library GUI.

This structured approach ensures a clear and effective interaction with the model library, enhancing user experience and operational efficiency.

4 Realization

The initial implementation of the model library utilizes Java within an object-oriented Eclipse Rich Client Platform framework (cf. [20]).

Listing 1: Exemplary model tree in JSON format.


```
{
  "title" : "root",
  "relPath" : "",
  "children" : [ {
    "title" : "Vehicle models",
    "relPath" : "vehicle_models/",
    "children" : [ ... ]
  }, {
    "title" : "Function models",
    "relPath" : "function_models/"
      ,
    "children" : [ {
      "title" : "VMS",
      "relPath" : "VMS/",
      "children" : [ ... ]
    },
    {
      "title" : "AMS",
      "relPath" : "AMS/",
      "children" : [ {
        "relPath" : "efm/",
        "metaFileName" : "efm.json
           ",
        "repoUrl" : "https://
           tinyurl.com/repo_efm/"
      }, ... ]
    } ]
  }, ... ]
}
```


The Eclipse platform provides numerous built-in mechanisms essential for development, including the *Standard Widget Toolkit* and event-driven, low-coupling communication among various graphical components. Moreover, it supports a wide range of extendable plug-ins with open interfaces.

Version management is integrated using the open-source tool GIT (cf. [9]), which offers established versioning protocols. The LoRra setup facilitates the avoidance of the limitations identified earlier by employing structured, text-based model descriptions.

Initially, user authentication is set up for the Atlassian service Bitbucket, with potential for future enhancements.

The model descriptions are structured in JSON format, aligning with standards outlined in [21]. Listing 1 illustrates an example of a model tree stored in this format.

Hierarchy elements are designated by attributes such as *title* (the display title of the element), *relPath* (the relative file path to the parent hierarchy element), and *children*. Model elements are characterized by fields like *relPath*, *metaFileName*, and *repoUrl* (the URL to the online GIT repository), with all pertinent metadata stored in the file identified by *metaFileName*.

The integration of sub-models into configurations is managed through XML-based SES, facilitating the establishment of an abstract structure for new configurations as a preliminary phase. This is followed by the generation of a specific configuration by referencing the appropriate sub-models and specifying the required version and variant.

This implementation strategy leverages the robust capabilities of proven tools for efficient version management and flexible integration, ensuring a scalable and modular architecture. By utilizing open text based formats for structured data representation, the model library effectively supports complex configurations and facilitates seamless data management and access.

5 Summary and Outlook

The paper discusses the design and implementation of a model library within the Low-Cost Rapid Control Prototyping platform LoRra, aimed at enhancing the functional development of embedded mechatronic systems. The LoRra library offers a cost-effective alternative to traditional CAE platforms, particularly beneficial for SMEs. It incorporates systematic data management with robust version and configuration management to ensure reusable, reliable, and consistent access to development artifacts across the system's development cycle. The library utilizes a hierarchical, tree-like structure for organizing models and supports multi-user environments through central storage and local working copies, ensuring that all team members access the most recent data.

The model library's foundational framework is robust, but enhancements are planned to optimize functionality and user experience. These include adding undo/redo functions, improving navigation in the GUI, and integrating advanced graphical editors for better model configuration and visualization. Enhancing the GIT tool's differential capabilities for Xcos models will allow more effective visualization of changes, aiding in version tracking and management.

A new user authentication system will support integration with central storage, boosting collaboration across distributed teams.

Automated consistency checks will ensure model configurations are coherent before deployment, complemented by comprehensive documentation and training materials to help new users.

Performance will be optimized through algorithm improvements and efficient data handling, particularly for large-scale projects. Establishing a community for sharing models and best practices, along with collaborative development features like shared workspaces and real-time tools, will increase the library's utility and reach.

Focusing on these improvements, the LoRra model library aims to significantly enhance its capabilities and user satisfaction, supporting the advancement of mechatronic systems and adapting to user needs.

Acknowledgement

Funded by the Lower Saxony Ministry of Science and Culture under funding number ZN3495 in the Lower Saxony Advance of the Volkswagen Foundation and supervised by the Center for Digital Innovations Lower Saxony (ZDIN).

References

[1] Liu-Henke X, Scherler S, Fritsch M, Quantmeyer F. Holistic development of a full-active electric vehicle by means of a model-based systems engineering. In: *Proceedings of 2016 IEEE International Symposium on Systems Engineering (ISSE)*, edited by Rassa B, Carbone P. 2016; pp. 1–7. DOI: 10.1109/SysEng.2016.7753142.

[2] Jacobitz S, Göllner M, Zhang J, Yarom OA, Liu-Henke X. Seamless validation of cyber-physical systems under real-time conditions by using a cyber-physical laboratory test field. In: *Proceedings of the International Conference on Recent Advances in Systems Science and Engineering (RASSE)*. IEEE. 2021; pp. 1–8. DOI: 10.1109/RASSE53195.2021.9686844.

[3] Jacobitz S, Liu-Henke X. The Seamless Low-cost Development Platform LoRra for Model based Systems Engineering. In: *Proceedings of the 8th International Conference on Model-Driven Engineering and Software Development*. SciTePress. 2020; pp. 57–64. DOI: 10.5220/0008993500570064.

[4] Kittlaus HB. *Software Product Management*. Berlin, Heidelberg: Springer. 2022. DOI: 10.1007/978-3-662-65116-2.

[5] Fahmy S. The Evolution of Software Configuration Management. *International Journal of Advanced Trends in Computer Science and Engineering*. 2020; 9(1.3):50–63. DOI: 10.30534/ijatcse/2020/0891.32020.

[6] Guissouma H, Klare H, Sax E, Burger E. In: *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, edited by IEEE. 2018; pp. 298–305. DOI: 10.1109/SEAA.2018.00056.

[7] Kruse B, Shea K. Design Library Solution Patterns in SysML for Concept Design and Simulation. *Procedia CIRP*. 2016;50:695–700. DOI: 10.1016/j.procir.2016.04.132.

[8] Ratti N, Kaur P. Case Study: Version Control in Component-Based Systems. In: *Designing, Engineering, and Analyzing Reliable and Efficient Software*, edited by Singh H, Kaur K, pp. 283–297. IGI Global. 2013;DOI: 10.4018/978-1-4666-2958-5.ch016.

[9] Eriksson H, Sun J, Tarandi V, Harrie L. Comparison of versioning methods to improve the information flow in the planning and building processes. *Transactions in GIS*. 2021;25(1):134–163. DOI: 10.1111/tgis.12672.

[10] Nugroho YS, Hata H, Matsumoto K. How different are different diff algorithms in Git? *Empirical Software Engineering*. 2020;25(1):790–823. DOI: 10.1007/s10664-019-09772-z.

[11] Schmitz D, Deng W, Rose T, Jarke M, Nonn H, Sanguanpiyapan K. Configuration Management for Realtime Simulation Software. In: *Proceedings of the 35th Euromicro Conference on Software Engineering and Advanced Applications*. IEEE. 2009; pp. 229–236. DOI: 10.1109/SEAA.2009.69.

[12] Durak U, Pawletta T, Oguztuzun H, Zeigler BP. System entity structure and model base framework in model based engineering of simulations for technical systems. In: *Proceedings of the Symposium on Model-driven Approaches for Simulation Engineering*, edited by D'Ambrogio A, ACM Digital Library. Society for Computer Simulation International. 2017; pp. 1–10. DOI: 10.22360/springsim.2017.mod4sim.001.

[13] Liu-Henke X, Jacobitz S, Scherler S, Göllner M, Yarom O, Zhang J. A Holistic Methodology for Model-based Design of Mechatronic Systems in Digitized and

Connected System Environments. In: *Proceedings of the 16th International Conference on Software Technologies (ICSOFT)*, edited by Fill HG, van Sindern M, Maciaszek L. SciTePress. 2021; pp. 215–223. DOI: 10.5220/0010566702150223.

[14] Jacobitz S, Liu-Henke X. Automatic Code Generation for a Seamless Low-cost Development Platform. In: *10th International Conference on Model-Driven Engineering and Software Development (MODELSWARD)*, edited by Pires LF, Hammoudi S, Seidewitz E. 2022; pp. 294–301. DOI: 10.5220/0010894300003119.

[15] Jacobitz S, Liu-Henke X. A Real-Time Interface for Xcos – Demonstrated on a Battery-management System. In: *2nd Scilab Conference*. 2019; pp. 1–8.

[16] Bass L, Clements P, Kazman R. *Software architecture in practice*. SEI series in software engineering. Upper Saddle River NJ u.a.: Addison-Wesley, 3rd ed. 2013. ISBN: 978-0321815736.

[17] Henninger S, Corrêa V. Software pattern communities: current practices and challenges. In: *Proceedings of the 14th Conference on Pattern Languages of Programs*, edited by Aguiar A, Yoder J. ACM Press. 2007; pp. 1–19. DOI: 10.1145/1772070.1772087.

[18] Adams S. MetaMethods: The MVC paradigm. *HOOPLA!* 1988;1(4).

[19] Liu Z, Li F, Liu H, Wu C, Zhang J. A Study of Cockpit HMI Simulation Design Based on the Concept of MVC Design Pattern. In: *Proceedings of the 3rd International Conference on Modelling, Simulation and Applied Mathematics (MSAM 2018)*. Atlantis Press. 2018; pp. 82–84. DOI: 10.2991/MSAM-18.2018.19.

[20] Vogel L. *Eclipse Rich Client Platform: The complete guide to Eclipse application development*. Hamburg: Vogella, 3rd ed. 2015. ISBN: 978-3943-747133.

[21] International Organization for Standardization. ISO/IEC 21778:2017: Information technology - The JSON data interchange syntax. 2017.

Concept and Realisation of a Holistic, Highly Flexible HiL Test System for Testing Autonomous Driving Functions

Xiaobo Liu-Henke*, Marian Göllner, Sven Jacobitz, Jie Zhang

Institute of Mechatronics, Ostfalia University of Applied Sciences, Salzdahlumer Str. 46/48, 38302 Wolfenbüttel, Deutschland; *x.*liu-henke@ostfalia.de*

SNE 34(4), 2024, 225-232, DOI: 10.11128/sne.34.tn.10715
Selected ASIM WS 2023 Postconf. Publication: 2023-10-15
Rec. Improved English V.: 2024-11-23; Accepted: 2024-11-25
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. Autonomous driving and networked cyber-physical transport systems pose ever-increasing challenges for the development and validation of advanced driver assistance systems and autonomous driving functions. Real-time optimisation and testing in particular are associated with enormous effort and risk. A holistic, flexibly configurable, real-time-capable test bench for the entire vehicle would provide a remedy here. The following article describes the concept of the holistic, highly flexibly configurable real-time test system for intelligent vehicles in co-operating cyber-physical traffic systems (ERAGON), which is currently being developed at the Ostfalia University of Applied Sciences.

Introduction

Mobility is in the midst of a disruptive change due to the increasing digitalisation and networking of vehicles. The autonomous driving of electric hybrid vehicles in highly networked cyber-physical systems (CPS) is one of the core technologies in the digital transformation process of mobility.

The variety of applications for autonomous vehicles requires ever more diverse sensors and ever more complex and intelligent algorithms from the fields of modern control technology and artificial intelligence (AI). This results in systems that are even more extensive and complex than the already existing highly networked electronic vehicle functions [1].

The development of such systems is highly complex and requires multidisciplinary, interdisciplinary design processes based on proven rapid control prototyping (RCP) methods from mechatronics research. In a top-down process, the system complexity is first reduced by structuring using modularisation and hierarchisation. The upper hierarchy levels of the networked mechatronic system (VMS) and the autonomous mechatronic system (AMS) are increasingly characterised by intelligent autonomous driving functions and driver assistance systems (ADAS) such as electronic vehicle management and intelligent, cooperative guidance.

The model-based design of each individual subsystem is then carried out in a bottom-up process incorporating the validation processes model-in-the-loop (MiL), software-in-the-loop (SiL) and hardware-in-the-loop (HiL). Such a structured approach is essential for the design and validation of networked mechatronic systems [2].

In order to investigate the integrated overall functionality of autonomous, networked vehicles, a holistic vehicle test bench that accurately maps the overall system consisting of the road, networked vehicles and networked driving environment and at the same time stimulates the vehicle's sensors is therefore indispensable.

An evaluation of the following state of knowledge and research shows that there are many partial solutions for reliable verification for ADAS, highly automated and autonomous driving functions at AMS and VMS level and that these are also being continuously further developed. What is missing, however, is a holistic test system that enables flexible configuration of both the test specimen and the test environment under realistic and reproducible conditions at all levels of mechatronic structuring.

This article therefore presents the concept of the holistic, highly flexibly configurable real-time test system for intelligent vehicles in co-operating cyber-physical transport systems (ERAGON). Work is currently underway on this new system at Ostfalia University.

1 State of the Art

According to [3], valid functional validation is a major challenge on the progression towards autonomous driving. It must be ensured that the designed functions in the overall system are verifiably safe in terms of output quality and probability of misinterpretation [4]. It is necessary to reproducibly test as many situations as possible with which the vehicle could be confronted. If this task is to be carried out under real road traffic conditions, hundreds of millions of test kilometres will be required [5].

It therefore makes sense to supplement the test under real conditions with simulation-based procedures so that certain rare situations can still be tested safely and reproducibly. This requires a complex, real-time-capable test system that generates test conditions that are as realistic as possible using a mixture of real and simulated systems and environmental components.

To this aim, Chen et. al [6] use an integrated simulation and test platform for self-driving vehicles. Their platform offers the possibility of testing a real vehicle on a restricted test track. The special feature of their approach is that the sensor signals (GPS, IMU, lidar and camera) originate from a high-precision virtual simulation scenario and are processed by a real control unit in the vehicle to generate real driving commands. In the course of the test, the sensor technology is therefore mapped completely virtually, which means that unwanted aggregation effects can occur even though a real vehicle is present [7].

The developed driving functions can also be tested in advance without the real components in the simulation environment in offline and online simulations. The platform presented in Chen et. al. is not flexible on the hardware side. It is tied to a specific research vehicle with a prototypical control unit with fixed interfaces. This means that neither subordinate functions nor functions based on V2X communication can be secured with this structure. Due to the hardware architecture, the platform is also not suitable for incorporating AI algorithms.

The vehicle-in-the-loop methodology for evaluating automated driving functions in virtual traffic from Solmaz [8] is based on a very similar structure to Chen. Here too, a real vehicle is used on a restricted test track combined with a simulated environment. The main difference, however, is the design of the control unit for the driving functions.

In contrast to the control unit from Chen, the MicroAutoBox II from Solmaz allows flexible modification of the system under test by RCP. However, this is limited to the AMS level. Various algorithms, including those from the field of AI, can be implemented and additional signals, e.g. from V2X communication, can theoretically also be integrated. However, the test system itself is neither flexibly configurable nor free from aggregation effects. The reproducibility of the test scenarios is always questionable with test systems on real test tracks, as tyre behaviour, for example, changes depending on the weather and temperature despite simulated sensor signals.

Kanchwala [9] presents a real-time HiL vehicle simulator that clamps a real vehicle into a stationary test setup. Here, driving resistances are simulated via four electric motors that are directly coupled to the wheels. The associated target torques are determined by a real-time computer on which the test scenario is simulated. However, the focus of this system is on analysing the longitudinal dynamic driving characteristics, e.g. in virtual off-road environments. Investigations of steering behaviour or consideration of sensors and communication systems are not possible. The flexibility of the vehicle simulator is limited to testing different vehicles. Flexible configuration of the test setup is not provided.

Ying et al [10] use a vehicle-in-the-loop simulation and test platform to validate the functionality of autonomous vehicles. The associated vehicle test bench has four individual excitation units, which make it possible to map three degrees of freedom onto a real vehicle. The vehicle's sensors (camera, lidar and radar) are stimulated with signals generated on the basis of virtual traffic scenarios.

As a result, the test environment enables reproducible and fully controllable test scenarios. The simulation environment allows the integration of virtual V2X communication. However, this is not modelled in reality. This means that it is not possible to fully validate the driving functions at VMS level.

Furthermore, the vehicle to be tested cannot be modified. Although it enables the validation of various driving functions in different test bench configurations, it does not allow the prototyping of freely definable functions.

2 Methodology

The holistic model-based, verification-oriented RCP methodology is used for the development of complex interconnected mechatronic systems. By means of mechatronic structuring, the overall system complexity is handled. For this, the linked CPS is divided into hierarchically arranged subsystems with four hierarchical levels through modularization and hierarchization: mechatronic function modules (MFM), mechatronic function groups (MFG), autonomous mechatronic systems (AMS) and interconnected mechatronic systems (VMS) [11].

The result of the structuring is a functional decomposition of the overall system into encapsulated modules. The hierarchically arranged modules with their sub-functions have uniquely defined physical and information-technological interfaces in horizontal and vertical direction and form the basis for the later integration into the overall system [12].

After the hierarchical structuring and definition of all interfaces, the model-based and function-oriented design of each individual module follows in a bottom-up process. Starting with the lowest and at the same time most vital level MFM, the integration into higher-level functions then is performed on all hierarchy levels. The design as well as integration of these functions into the overall system (mechatronic composition) is done using the model-based mechatronic development cycle.

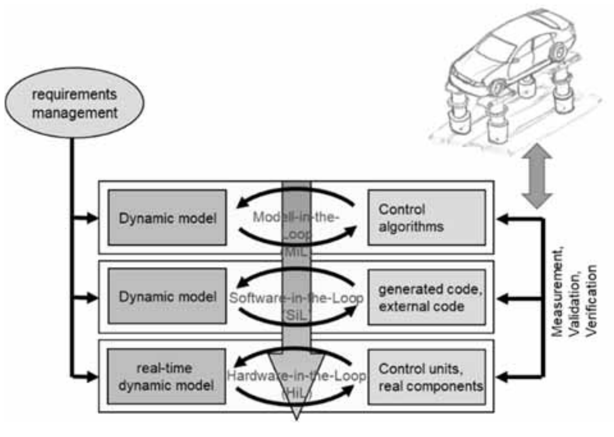


Figure 1: Seamless model-based development and validation process [11].

This is followed by Model-in-the-Loop (MiL) simulations, where control algorithms and AI are developed based on a physical or mathematical equivalent model and tested on a vehicle model.

An executable program code is generated from the simulatively tested algorithms within the context of Software-in-the-Loop (SiL) simulation by means of automatic code generation and tested offline in a virtual test bench.

This is followed by Hardware-in-the-Loop (HiL) simulations, in which a real-time vehicle model supplemented by physical subcomponents is used online to validate and optimise the algorithms and intelligent functions under real-time conditions [13]. Figure 1 illustrates the process flow.

3 Concept of the Test System

The real-time test system for intelligent vehicles in cooperating cyber-physical traffic systems ERAGON mainly consists of four modules: test bench module, real-time information processing, software and MMI communication module.

This allows the system to be flexibly configured and used for different Vehicles under Test (VUTs). Figure 2 illustrates the interaction.

All subordinate test systems are arranged hierarchically via defined interfaces. Thus, the ERAGON is able to be used flexibly in different variants for different testing purposes.

All subordinate test systems are arranged hierarchically via defined interfaces. Thus, the ERAGON is able to be used flexibly in different variants for different testing purposes.

3.1 Test Bench Module

The test bench module essentially consists of a complex excitation unit for the simulation and stimulation of a VUT by the simulated environment. An extensive environment simulation is used to create a virtual 3D world and to represent the real objects in the environment of the vehicle. The sensors can be stimulated in different ways with the data from the simulation.

One possibility is the over-the-air stimulation of the calculated obstacles by means of a physical stimulus, e.g. ultrasonic systems using ultrasonic waves via an oscillating membrane.

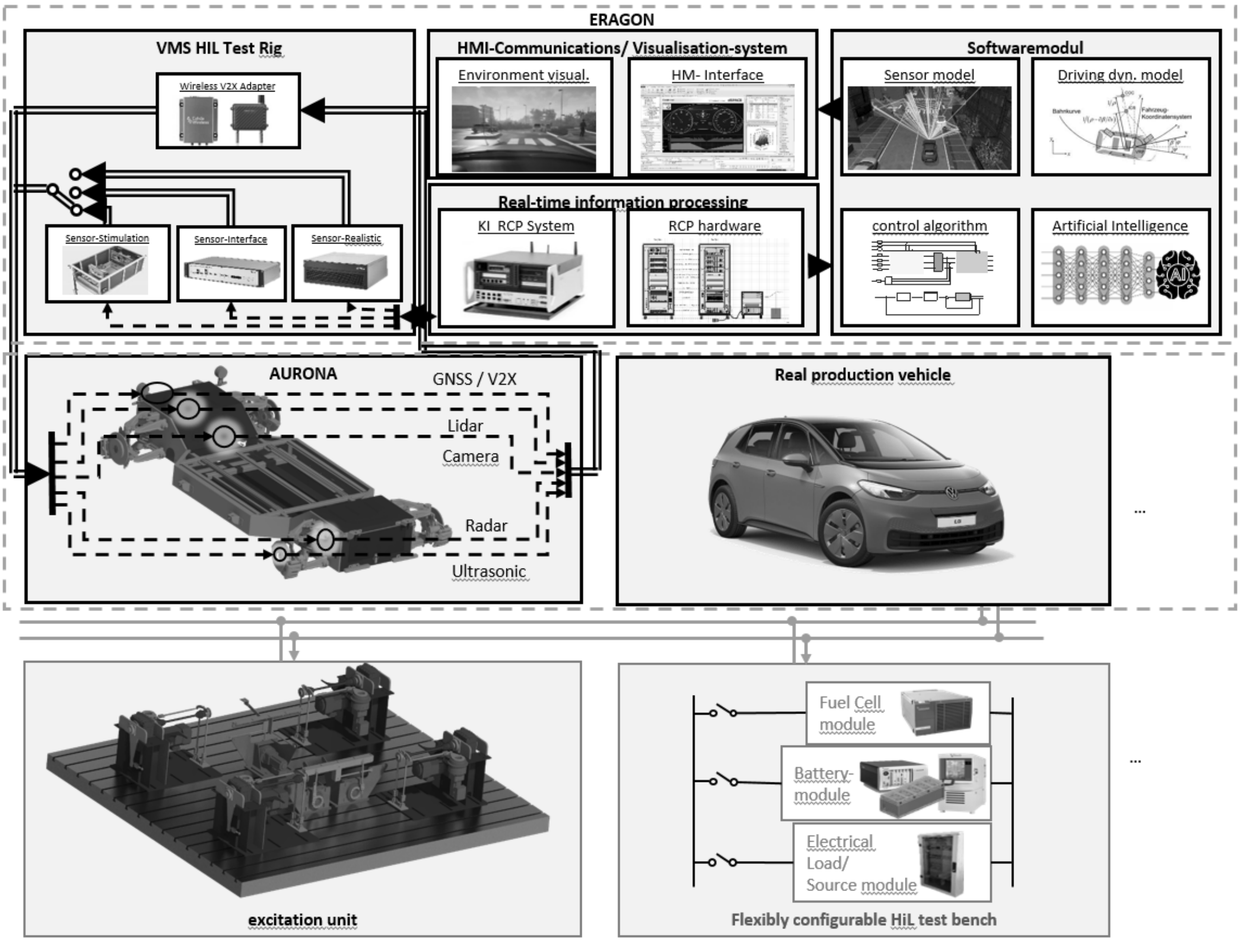


Figure 2: Concept of the ERAGON holistic testing system.

Another channel for feeding environmental signals is an additional V2X development platform for easy access to V2X communication, so that no implementation of specific communication protocols and software layers is required.

These systems provide a broad platform for implementing and coupling comprehensive traffic simulation. The core module can thus simulate a complete traffic scenario in a reproducible and realistic manner. This high degree of augmentation of reality and simulation is indispensable for testing higher-level autonomous and interconnected functions at the AMS and VMS structural levels. It makes comprehensive testing of the driving function in a CPS traffic environment at the AMS/VMS structural levels possible for the first time.

3.2 Software Module and HMI

The software module contains all the controller functions and virtual systems to be tested, like the sensor and vehicle models. In addition, calibration, scaling and signal conditioning take place here.

The software module is digitally computed and processed as a distributed calculation on several processors of the real-time information processing module under real-time conditions.

The communication module with human-machine interface, which is also used for visualization, is used for measurement and calibration tasks. Appropriate user interfaces (GUIs) are provided for this purpose. The RTI (Real-Time Interface) of the target hardware is used to control and automate the test stand.

3.3 Real-Time Information Processing

The module of real-time information processing with its various RCP systems ensures data processing. The module is equipped with several RCP systems including AI RCP system for fast prototyping of intelligent control functions, machine learning of artificial intelligence using powerful processor technology for demanding real-time requirements, and has a fast data processing and storage system for storing and managing large amounts of data.

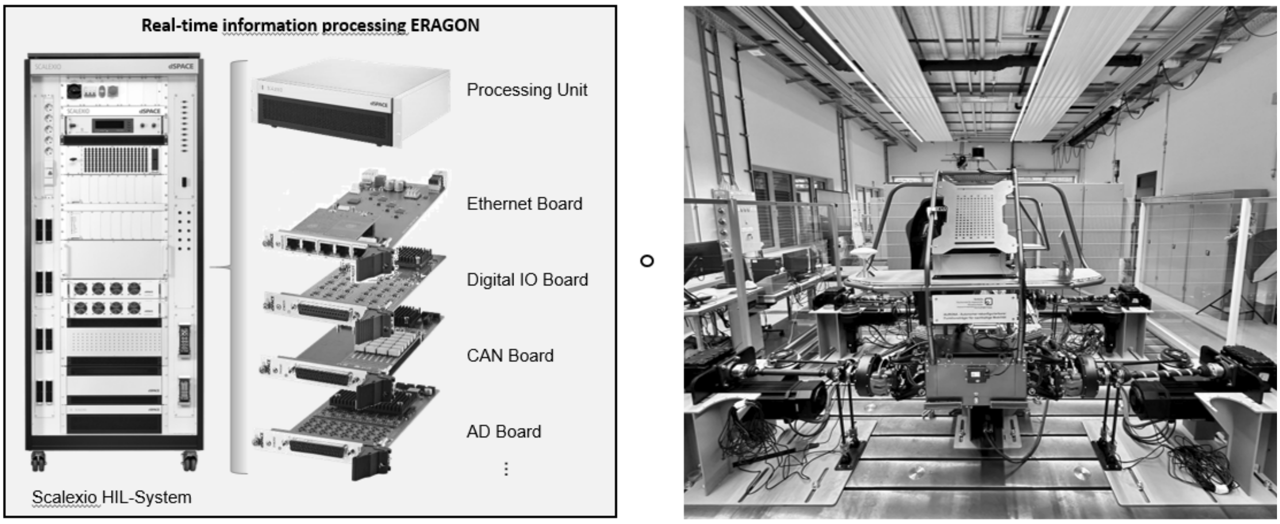


Figure 3: Concept of the ERAGON holistic testing system.

3.4 Device / Vehicle under Test

The test system is designed for flexible use. Defined interfaces to various test items are provided for this purpose. For example, real vehicles, function carriers or special test setups such as a single-track model can be used here.

Initially, an Autonomous Reconfigurable Functional Carrier for Sustainable Mobility (AURONA) is provided as a VUT.

As an RCP function carrier, AURONA is equipped with all typical environment sensors (camera, radar, ultrasound, lidar, etc.) required for highly automated or autonomous driving functions, whose interfaces are opened at various points.

In this way, the hardware components of the sensor system can be bypassed by means of RCP bypassing. The interfaces are configured in such a way that a flexibly configurable excitation of the sensors via the ERAGON can be selected. It is possible to excite the physical sensor input via sensor stimulation as well as to feed raw data directly to the electronics of the same sensor.

Furthermore, AURONA is equipped with a 5G V2X communication unit via which it can communicate bidirectionally with the V2X development platform of the ERAGON, but also with other V2X units. Thus, cooperative driving functions can also be investigated. The AURONA is also equipped with a drive and steering system.

This makes it possible to test the influence of the actuator elements in an autonomous vehicle. In this way, it is able to implement the commands for operation in a completely realistic way. These are counter actuated via the excitation unit of the ERAGON, resulting in realistic and closed-loop driving behaviour of the AURONA or also of another DUT.

4 Realisation

The test system is implemented step by step in a bottom-up process. Starting from the lowest and at the same time most vital level, the functions are implemented and subsequently integrated.

Figure 3 illustrates the partial realization of the test system. On the left is the heart of the test-rig – the Real time information processing system – with the necessary processing units and the highly flexible interchangeable interface system.

On the right side is the excitation unit in a coupled state with the RCP function carrier AURONA. The motors and gears for dynamical excitation on the vehicle chassis are exposed.

The test field itself is surrounded by a safety fence. A release control prevents people from being inside the safety area during the test operation.

Figure 4 shows the RCP function carrier AURONA. The vehicle is equipped with four direct drives and a break-by-wire system. All four wheels can be driven, braked and steered individually. GPS and LIDAR are used for position detection.

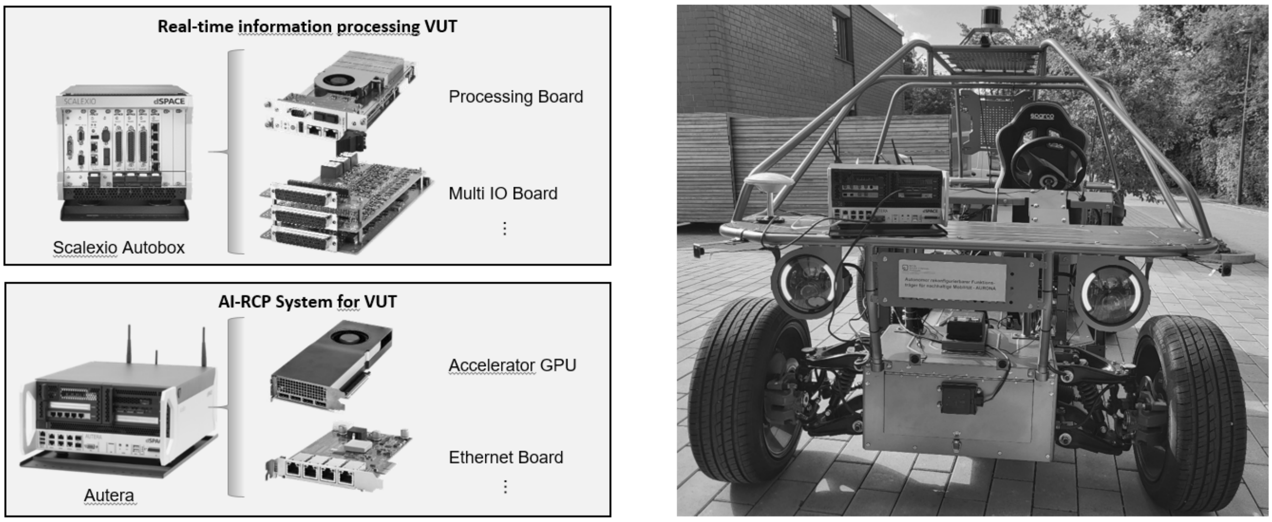


Figure 4: Concept of the ERAGON holistic testing system.

Objects are detected via camera, LIDAR, ultrasound and RADAR. Sensor data fusion with V2X data enables the creation of a complete dynamic map of the environment. On the left, the real-time data processing (Scalexio system from dSPACE) and the vehicle computer of the AURONA are shown.

Figure 5 finally shows the data coupling mechanism for the simulation, emulation und stimulation of vehicle sensors using the ERAGON test bench system.

Depending on the type of sensor excited and the measuring principle to be stimulated, a distinction is made here between the pure simulation of measurement results, the emulation of sensor measurement data and the stimulation of the physical sensor with electromagnetic radiation.

The usual approach is to completely bypass the sensor data within the simulation by assuming an omniscient system.

However, since it is highly likely that the complete capturing of all system states of a complex system will only be possible through a state estimation based on sensor data evaluation and fusion, an important component of the ERAGON test bench is the simulation of sensors and their dynamic physical behaviour.

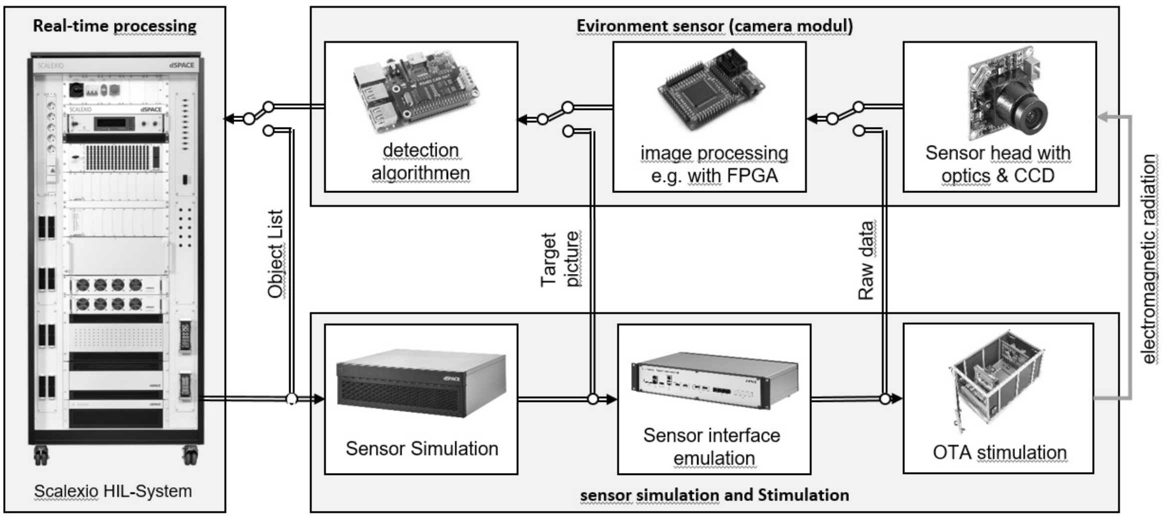


Figure 5: Concept of the ERAGON holistic testing system.

The basis is the simulation of the signal paths of the physical signals recorded by the sensors; in the types frequently used for autonomous driving functions, these are electromagnetic waves. For example, the path of an electromagnetic wave emitted by a radar sensor is tracked within the simulated driving environment using ray tracing in order to reproduce the reflections on objects and surfaces as realistically as possible.

A downstream emulation generates the exact signal that would be emitted by the real sensor measuring head, taking into account the physical properties of the sensor.

In the final step, this signal can now also be ‘shown’ to the real sensor head over-the-air by converting it into the electromagnetic radiation on which the measuring principle is based. This ensures the highest possible fidelity to reality.

5 Summary and Outlook

In this paper, the concept of a holistic, highly flexible configurable HiL test system ERAGON for testing autonomous driving functions was presented. In a closed loop together with the function carrier AURONA, this system is able to simulate and stimulate the entire autonomous vehicle system, starting from the infusion of raw sensor data via the development and testing of AI functions up to the stimulation of realistic driving situations.

The information processing of AURONA was integrated and the vehicle sensors are connected to the environment and sensor simulation.

Currently, the provided concept is under realization. In the next steps, the test system will be successively implemented and put into operation. Therefore, simulation scenarios with defined area of application are used.

Acknowledgement

Funded as part of the EU response to the Covid 19 pandemic. Funding was provided by REACT-EU resources from the EFRE under grant number ZW 7-85195402. Responsibility for the contents rests with the authors.

EUROPÄISCHE UNION
Europäischer Fonds für regionale Entwicklung

Als Teil der Reaktion der EU auf die Covid-19-Pandemie finanziert

EUROPA FÜR NIEDERSACHSEN

References

[1] Liu-Henke X, Yarom O. Scherler S. *Virtual Development and Validation of a Function for an Automated Lateral Control using Artificial Neural Networks and Genetic Algorithms.* 91st IEEE Vehicular Technology Conference (VTC2020-Spring), Antwerp, Belgium, May 25 - 28, 2020.

[2] Liu-Henke X, Jacobitz S, Scherler S, Göllner M, Yarom O, Zhang J. *A Holistic Meth-odology for Model-Based Design of Mechatronic Systems in Digitized and Connected System Environments.* 16 th International Conference on Software Technologies (ICSOFT), Online, July 6-8, 2021.

[3] Aeberhard M, et al. *Experience, Results and Lessons Learned from Automated Driving on Germany's Highways*, in IEEE Intelligent Transportation Systems Magazine, vol. 7, no. 1, pp. 42-57, 2015.

[4] Mitsch S, Ghorbal K, Platzer A. *On Provably Safe Obstacle Avoidance for Autonomous Robotic Ground Vehicles.* Robotics Science and Systems (RSS), 2013.

[5] Wachenfeld W, Winner H. *The Release of Autonomous Vehicles.* In: Maurer M, Gerdes JC, Lenz B, Winner H*: Autonomous Driving,* Eds. Springer Berlin Heidelberg, Berlin, Heidelberg, 425–449, 2016. DOI 10.1007/978-3-662-48847-8_21

[6] Chen S, Chen Y, Zhang S, Zheng N: *A Novel Integrated Simulation and Testing Platform for Self-Driving Cars with Hardware in the Loop.* In IEEE Transactions on Intelligent Vehicles, vol. 4, no. 3, pp. 425-436, Sept. 2019. DOI 10.1109/TIV.2019.2919470

[7] Jacobitz S, Göllner M, Zhang J. Yarom OA, Liu-Henke X. *Seamless Validation of Cyber-Physical Systems Under Real-Time Conditions by Using a Cyber-Physical Laboratory Test Field.* IEEE International Conference on Recent Advances in Systems Science and Engineering (RASSE), Online, December 12-14, 2021.

[8] Solmaz S, Rudigier M, Mischinger M. *A Vehicle-in-the-Loop Methodology for Evaluating Automated Driving Functions in Virtual Traffic.* 2020 IEEE Intelligent. Vehicles Symposium (IV), 2020, pp. 1465-1471. DOI 10.1109/IV47402.2020.9304811

[9] Kanchwala H, Dhillon JS. *A real-time hardware-in-the-loop vehicle simulator.* 2020 IEEE 18th International Conference on Industrial Informatics (INDIN), 2020, pp. 182-187. DOI 10.1109/INDIN45582.2020.9442138

[10] Ying G, Zhao X, Xu Z, Cheng J, Wang W: *An Indoor Vehicle-in-the-Loop Simulation Platform Testing Method for Autonomous Emergency Braking*. April 2021, Journal of Advanced Transportation 2021(3):1-12. DOI 10.1155/2021/8872889

[11] Liu-Henke S, Scherler S, Göllner M. *System Architecture of a full active Autonomous Electric Vehicle*. IEEE Transportation Electrification Conference and Expo (ITEC), Chicago, IL, USA, June 22 - 24, 2017.

[12] Liu-Henke X. *Mechatronische Entwicklung der aktiven Feder-/Neigetechnik für das Schienenfahrzeug RailCab*. VDI-Fortschritt-Berichte, Reihe 12, Nr. 589, VDI-Verlage. Düsseldorf, 2004.

[13] Jacobitz S, Liu-Henke X. *A Real-Time Interface for Xcos – Demonstrated on a Battery-management System*. 2nd Scilab Conference, Berlin, Germany, October 14, 2019.

Kelvin and Bush: Their Contributions to Today's System Modelling and Simulation Methods

David J. Murray-Smith

James Watt School of Engineering, University of Glasgow, G12 8QQ, United Kingdom, *David.Murray-Smith@glasgow.ac.uk*.

SNE 34(4), 2024, 233-239, DOI: 10.11128/sne.34.on.10716
Invited: 2024-04-04; Submitted: 2024-08-20
Revised: 2024-11-03; Accepted: 2024-11-26
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. In 1876 Professor Sir William Thomson, who later became Lord Kelvin, published a series of papers that are seen as the basis for subsequent developments in analogue simulation and for many of today's simulation software tools. Sir William proposed that a type of mechanical integrating device designed by his brother, Professor James Thomson, could be used directly to solve ordinary differential equations of any order. Although not implemented in the nineteenth century, the ideas resurfaced in the 1920s and 1930s in the work of a team at the Massachusetts Institute of Technology (MIT) under the direction of Vannevar Bush. At that time there was much interest in the possible use of mechanical methods for system analysis and design in the field of electrical power systems. This led to the development of "integraph" devices and to the mechanical differential analysers for which Bush is rightly famous. As we approach the 150th anniversary of the publication of the key papers by Sir William Thomson and his brother in 1876 and also the centenary of the developments at MIT in the mid-1920s, it is appropriate to review the significance of these early developments which represent important historical milestones in simulation methods. Many of the ideas and principles established by Kelvin and Bush remain valid today but their origins are seldom fully recognised.

Introduction

In the early 1870s, Sir William Thomson, Professor of Natural Philosophy at the University of Glasgow (later to become Lord Kelvin), was working on the development of harmonic analyzers and their application to tidal prediction.

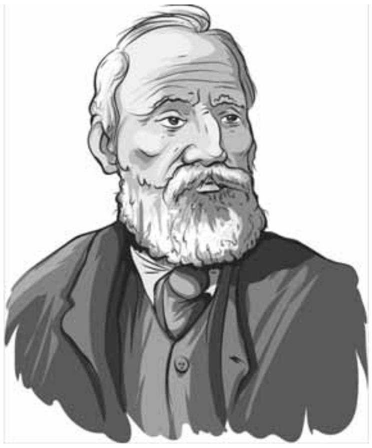


Figure 1: Lord Kelvin, Graphic Picture , ID 54758644 © Lukaves | Dreamstime.com

In 1873 Sir William's older brother, James, was appointed to the Regius Chair of Civil Engineering and Mechanics at Glasgow.

One of Professor James Thomson's research interests concerned "planimeters", which were devices used to find the area of a closed figure by tracing over it using a mechanical linkage. He recognised the accuracy limitations of previous planimeters that depended on rolling and slipping actions and developed a design that operated using rolling action only and involved a disk, ball and cylinder. James Thomson was encouraged in this work by the famous Scottish physicist, Professor James Clerk Maxwell who also had a strong interest in planimeters [1]. The new form of mechanical device developed by Thomson had potential advantages in terms of accuracy, compared with earlier designs.

Sir William Thomson was fully aware of his brother's research on planimeters and the Thomson brothers collaborated on the use of these devices within a harmonic analyser based on the evaluation of Fourier series coefficients. This formed an important area of research for Sir William, leading to highly successful longer-term developments in his tidal predictors [2], [3] .

1 Sir William Thomson and the Solution of Differential Equations using Mechanical Integrators

Early success with the harmonic analyser applications led Sir William to consider other possible uses of mechanical integrating devices, with the publication of four key papers in the Proceedings of the Royal Society of London early in 1876.

The first of these was written by James Thomson [4]. It described his improved type of planimeter, which he called an "integrator", and he is now widely credited with being the first person to use that word. The other three papers, which were all by Sir William as the sole author, were concerned with possible applications of these integrator devices to the solution of ordinary differential equations. The first step was a process which led to an iterative form of solution [5], [6]. A more direct approach followed from this and, as Sir William wrote on the use of two integrators for the solution of second-order equations [6]: "So far I had gone and was satisfied, feeling I had done what I wished to do for many years. But then came a pleasing surprise. Compel agreement between the function fed into the double machine and that given out by it". This process of forcing agreement required a mechanical feedback connection and led to a closed-loop system described by an equation identical to the given differential equation.

Sir William had immediately realised that implementation of the closed-loop configuration would automatically allow a solution to be found for any set of initial conditions and any input function. The final paper [7] extended this idea to the solution of ordinary differential equations of any order.

This can be illustrated by an example involving the linear motion of a simple mechanical system having a mass M, a spring of stiffness, K, and a damping element with viscous resistance R, as shown in Figure 2.

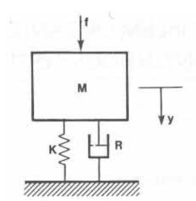


Figure 2: A schematic diagram of a mechanical system with mass, spring and viscous damping elements.

A mathematical description of this system based on a linear ordinary differential equation is:

$$M\frac{d^2y}{dt^2} + R\frac{dy}{dt} + Ky = f(t) \qquad (1)$$

where $y(t)$ represents the displacement of the mass and $f(t)$ represents a time-varying force applied to the mass M. Rearranging this equation gives

$$\frac{d^2y}{dt^2} = \frac{1}{M}\left[f(t) - R\frac{dy}{dt} - Ky\right] \qquad (2)$$

If it is assumed that the acceleration $\frac{d^2y}{dt^2}$ is available, the variables representing the velocity $\frac{dy}{dt}$ and the position $y(t)$ can be found directly using two integrators connected in cascade as shown in Figure 3.

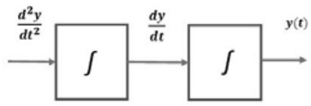


Figure 3: A block diagram showing two connected integrator units.

Negative feedback pathways need to be established from the variables representing $y(t)$ and $\frac{dy}{dt}$ through constant coefficient units set to values K and R respectively.

As shown in Figure 4, the force $f(t)$ forming the right-hand side of Equation (1) needs to be added to the feedback terms and the result is fed to the first integrator through a coefficient unit set to a value value $\frac{1}{M}$.

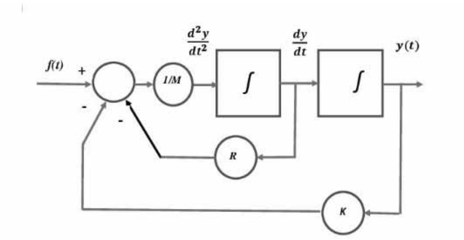


Figure 4: The block diagram corresponding to Equation (2), representing the coupled mass, spring and damper system of Figure 2.

This feeding back of the terms involving $Ky(t)$ and $R\frac{dy}{dt}$ and the addition of the forcing term $f(t)$ is the procedure described by Thomson as compelling "agreement between the function fed into the double machine and that given out by it".

It should be noted that, although this example involves a linear second-order ordinary differential equation, Sir William Thomson had correctly observed that the approach could be applied to ordinary differential equations of any order and that the feedback pathways could include varying coefficients [7].

The difficulty inherent in the implementation of this closed-loop configuration was that the outputs from the mechanical integrators lacked the power to generate the necessary feedback quantities and Sir William did not discuss how to achieve this in his published papers.

Although the mechanical integrators developed by James Thomson could not be used immediately in the solution of differential equations, they were of central importance in the harmonic analysers used in Sir William's tidal predictors. At a later date, on Kelvin's advice, James Thomson's type of integrator was incorporated into a naval gun fire-control system being developed by Arthur Pullen and completed about 1912 [8]. However, the solution of ordinary differential equations using mechanical integrators was not achieved in Kelvin's lifetime and it was well into the twentieth century before that development took place.

2 The Development of the Mechanical Differential Analyser at MIT

The most significant step towards the development of practical mechanical simulation systems was in the mid-1920s when Professor Vannevar Bush and his colleagues at the Massachusetts Institute of Technology (MIT) began work on systems involving coupled integrating devices. The key stimulus for the work carried out by Bush and his team came from practical problems of transient performance and stability within electrical power systems.

These issues were very important in the United States during the early years of the twentieth century (e.g. [9]) and research was being carried out at MIT to try to gain a better understanding of the problems. Bush had a background in electrical circuit theory and electrical power systems and, in the early 1920s, was involved in the construction at MIT of electrical network models representing generators, transmission lines and electrical loads of different kinds within complex power systems.

Figure 5: Vannevar Bush with his Differential Analyzer, c. 1935. MIT Museum https://www.britannica.com/biography/Vannevar-Bush#/media/1/86116/19210

His interests then moved from the model networks to a more mathematical method of approach, based on the solution of ordinary differential equations.

The first step involved building a machine in which the integrating devices were watt-hour meters similar to those that measure household electrical energy usage. Although the first machine had limited capabilities in that it was intended only to evaluate the integral of the product of two given functions plotted on paper, it was a significant development in that it employed a servomechanism type of arrangement so that the rotation of the watt-hour meter disk could be followed precisely and the servo motor provided the energy needed to drive a pen to display the required integral. The group at MIT initially described their machine as a "continuous integraph".

An early publication describing this first machine [10] was followed very quickly by a paper describing a similar system which incorporated a feature termed "back coupling" [11]. This allowed integrator outputs to be fed back through coefficient units to the input and allowed the concepts suggested by Sir William Thomson to be applied for the first time. The servo-motor approach was used, once again, to allow the rotation of the disks on watt-hour meter integrator units to be fed into the inputs of other units. This was the first system capable of solving ordinary differential equations and involved six integrators.

In 1931, Bush announced the development of a more powerful machine [12] involving more integrators and the use of a form of torque amplifier which had been developed in 1925 by Henry W. Niemann at the Bethlehem Steel Corporation.

This provided the power needed to couple the elements of the mechanical simulator system together and the use of this form of torque amplification appears to have been suggested by Hazen. Bush termed this new machine a "differential analyser" and it provided far more flexibility than the earlier "integraph" systems as it eliminated previous restrictions in the form of the equations being represented. It was a general-purpose simulation system and its development was followed by other similar designs in the United Kingdom and elsewhere.

At MIT a further development programme began in 1935 leading to an improved and much more powerful differential analyser which went into service in 1942. A key paper by Bush and Caldwell [13], published in 1945, described the development of that machine and included a statement: "It is interesting to observe that it is the feedback connection which 'mechanizes' the equal sign in the equation because it applies the constraint which forces the machine to operate so as to equalize the two sides of the equation".

This is very similar to Kelvin's statement about having to "compel" agreement. In the introductory section of their 1945 paper, Bush and Caldwell [13] acknowledged the contribution made by Lord Kelvin when they stated "The machine … was placed in operation at the Massachusetts Institute of Technology in I925, and the first comprehensive differential analyzer was introduced in 1930. Subsequently, the early papers of Lord Kelvin were found in which he described a method for using a machine of this type".

3 The Fundamental Importance of Models as seen by Thomson and Bush

The study of governors and the analysis of their dynamic properties was, by the 1860s, considered an important area of research that had direct engineering applications. In November 1868 Sir William Thomson addressed the Institute of Engineers and Shipbuilders of Scotland on the subject of a new form of centrifugal governor [14].

This presentation led to a lively discussion with Professor William Maquorn Rankine (which was fully reported in the Transactions of the Institute [14]). Rankine was, at that time, Professor of Civil Engineering and Mechanics at the University of Glasgow.

He acknowledged the importance of further developments in governors generally, and his remarks were broadly supportive of the research being reported by Sir William Thomson. It is interesting to note that in February of that same year, Professor James Clerk Maxwell published his famous paper entitled "On governors" [15] which is now widely regarded as the first paper on the mathematical analysis of an engineering control system.

Although there is no evidence that Sir William Thomson's interest in the use of mechanical integrators in the solution of ordinary differential equations was directly linked to the work being carried out on governors, the growth of interest in the use of mathematical methods for the analysis of complex engineering systems, such as these, was likely to have been important in terms of his thinking at that time.

Kelvin was well known for his use of models and analogies and is famous for his remark, said to have been made during a lecture at the Johns Hopkins University in Baltimore in 1894, that: "I am never content until I have constructed a mechanical model of the subject I am studying. If I succeed in making one, I understand. Otherwise, I do not." [16] This appears to have been central to his research and teaching activities.

His strong interest in mechanical models and analogies is reflected in items that are now on display or stored within the Hunterian Museum at the University of Glasgow and in documents and manuscripts within the Kelvin Collection in the Archives and Special Collections section of the University Library.

This understanding of the potential role of mechanical integrators for the modelling of dynamic systems, as opposed to calculation, is further emphasised in a record of discussions that followed on from a presentation in 1885 by Professor Henry Selby Hele-Shaw in which he described and reviewed various forms of mechanical integrators.

Professor Hele-Shaw had designed several different mechanical integrating devices while working at the Royal School of Mines in London (now part of Imperial College, London). As outlined by Care [17], the contributors to the discussion included Major General Henry Prevost Babbage who was the younger son of Charles Babbage and had carried out work on his father's inventions after retiring from his army career. In the course of that discussion, Babbage expressed views that strongly favoured his father's type of mechanical calculating machine rather than devices based on mechanical integrators, such as those developed and applied by the Thomson brothers.

However, Professor Hele-Shaw responded with vigour and pointed out that there was a fundamental difference between applications involving the use of calculating machines producing precise numerical outputs and applications involving the modelling of engineering systems.

4 Discussion and Conclusions

The papers published by the Thomson brothers in 1876 are of great interest, both because of Sir William's comments about the moment of his discovery of the use of feedback and his recognition of the important part played by his brother James. Indeed, it is quite possible that without the development of Professor James Thomson's mechanical integrating devices, Sir William would not have spent so much time thinking about the solution of ordinary differential equations and would not have made his important discovery about linked networks of integrating units.

The technique for the automatic solution of ordinary differential equations of any order by interconnecting integrating devices in a closed loop and thus avoiding any type of iterative solution proved to be central to the development of dynamic system simulation techniques in general. Analogue computers, whether mechanical, or electronic, or based on other physical principles, all depend on the fact that many different types of real-world systems can be modelled using sets of equations having the same structure and involving linear or nonlinear ordinary differential equations. This means that, whatever the physical variables of the original system, the model can be represented in a mechanical implementation by displacements or shaft velocities or, in electronic analogues, by electrical voltages or currents.

Analogue computers allowed users to interact in a very direct fashion with the model of the system being investigated and to carry out experiments that would otherwise be difficult, expensive, or dangerous. The development of mechanical differential analysers was primarily a development in modelling technology rather than in computing technology as we view it now. The main applications were associated with dynamic systems in many different fields. For such applications, simulation techniques based on general-purpose digital computers and continuous system simulation software tools only became dominant when software technology and digital computer hardware had developed sufficiently to provide practical tools that offered levels of interaction between the model and the user similar to those possible using more costly analogue computer hardware.

Developments arising from the work at MIT in the 1920s and 1930s, and especially the move towards electronic analogue computers in the 1940s, are reviewed in more detail elsewhere (e.g. by Care [18], Mindell [19], Small [20], and Williams [21]). The connections between differential analysers, analogue computers and developments in control systems technology are also important and are explored in a paper by Paynter [22]. Applications of electronic analogue and hybrid computers to engineering problems in various fields, including control engineering, along with a discussion of the links to the developments by the Thomson brothers, are also explored in a recent paper by the current author [23].

Care [18] and Mindell [19] both argue strongly that analogue computing should be thought of as a development in system modelling technology, in which the interactions between the user and the machine are of central importance, rather than being viewed as a branch of computer technology. Care [18], Small [20] and Williams [21] also provide much interesting information about the history of mechanical and electro-mechanical differential analysers in the 1930s and 1940s, together with useful accounts of progress in the development of special-purpose and general-purpose electronic analogue computers following the introduction of the operational amplifier.

In a paper published in 1936 [24], Bush stated that differential analysers provided a "suggestive auxiliary to precise reasoning". This statement may remind former users of general-purpose analogue systems of how these machines were often applied in later decades. It may well also be a phrase that rings true today for those applying modern simulation tools to real-world problems where uncertainties about system boundaries, structure and parameter values are important.

By the 1980s low-cost personal computers were becoming available and, although their run-time speeds in large simulation applications could be slow in comparison with high-performance analogue machines, they were easy to use. Simulation developers and users required little additional specialist training and simulation programs could be written using well-established numerical techniques and general-purpose programming languages.

However, digital computer technology did not become dominant in simulation applications until it provided practical tools that offered modelling insight, levels of user interaction and speed that began to compete with analogue computer technology based on continuous variables.

Many of the modern simulation software tools that are now readily available on personal computers provide graphical user interfaces which allow problems to be defined in terms of a diagram that is very similar in principle to diagrams used to show the interconnections between units on an electronic analogue computer patch panel or mechanical differential analyser.

Such user interfaces, together with other developments in digital technology have greatly enhanced the levels of interaction possible between the user and continuous system simulation models on inexpensive digital hardware. However, the principles of simulation model development and applications have remained largely unchanged, with a direct link back to the work of Bush and his colleagues and thus to the Thomson brothers.

As we approach the 150th anniversary of the publications that established these key aspects of continuous system simulation methods, it is appropriate to review the part played by the Thomson brothers and by Vannevar Bush and his colleagues at MIT. However, other key figures must not be forgotten, especially Professor James Clerk Maxwell and his work on governor systems which was an early example of a practical application of mathematical modelling to investigate a complex engineering control problem. Simulation methods have grown to become of central importance in solving problems in many areas of engineering, science and medicine and the philosophy and terminology of analogue computing methods continue to be a feature of modern simulation tools. Real-world systems are inherently nonlinear and dynamic and, in mathematical terms, they can be described fully only using nonlinear dynamic models for which analytical methods of solution are not generally available.

Sir William Thomson and Vannevar Bush were both experimentalists at heart and one of Kelvin's famous quotations is highly relevant "….when you can measure what you are speaking about and express it in numbers, you know something about it, but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meagre kind: it may be the beginning of knowledge…."[25]. However, Professor Sir William Thomson and Professor P.G. Tait of the University of Edinburgh also included a statement in a preface to their well-known textbook [26], that "nothing can be more fatal to progress than a too confident reliance upon mathematical symbols, for the student is only too apt to take the easier course and consider the formula and not the fact as the physical reality".

In other words, modelling should never be separated from observations and measurements.

Both Kelvin and Bush were strongly motivated by practical problems. Kelvin's interest in models was linked to his underlying determination to understand the world around him and use the knowledge so gained to help solve important engineering design problems. Equally, Bush saw system modelling as a key to finding ways to overcome fundamental problems in the development of electrical power systems, as encountered in the late 19^{th} and early 20^{th} centuries. It can be argued that, without this strong motivation arising from practical problems, these early developments in simulation methods would have taken much longer and it is likely that computational tools for simulation would have evolved in different ways. Those working in system modelling and simulation today owe much to the work of these pioneers.

References

[1] Maxwell JC. Description of a new form of the planimeter, an instrument for measuring the areas of plane figures drawn on paper. *Trans. Roy. Scottish Soc. Arts*. 1856; 4(4): p420-429.

[2] Thomson W. Report on tidal instruments, *Proc. Roy. Soc. London*. 1878; 27: p371.

[3] Thomson W. The tide gauge. Tidal harmonic analyser and tide predictor. *Proc. Inst. Civil Eng.* 1881; 45: p2-25.

[4] Thomson J. On an integrating machine having a new kinematic principle. *Proc. Roy. Soc. London.* 1876; 24: p262-265.

[5] Thomson W. On an instrument for calculating the integral of the product of two given functions. *Proc. Roy. Soc. London*, 1876; 24: p266-269.

[6] Thomson W. Mechanical integration of the linear differential equations of the second order with variable coefficients. *Proc. Roy. Soc. London*; 1876: 24: p269-271.

[7] Thomson, W., Mechanical integration of general linear differential equation of any order with variable coefficients. *Proc. Roy. Soc. London*; 1876: 24: p 271- 275.

[8] Care C. A multi-stranded chronology of analogue computing. In Care, C. Technology *for Modelling, Electrical Analogies, Engineering Practice, and the Development of Analogue Computing*, London, Springer ; 2010, Section 2.3.4.2, 34-35. ISBN 978-184882-47-3.

[9] Bush V, Booth RD. *Power* system transients. *Trans. Am. Inst. Elec. Eng*. 1925; 22(4): p229-249.

[10] Bush V, Gage FD, *Stewart* HR. A continuous integraph, *J. Franklin Inst*. 1927; 211(1): p63-84.

[11] Bush V, Hazen H. Integraph solution of differential equations, *J. Franklin Inst*. 1927; 211: p575-613.

[12] Bush V. The *differential* analyzer: a new machine for solving differential equations, *J. Franklin Inst*. 1931; 212(4): p447-488.

[13] Bush V, *Caldwell* SH. A new type of differential analyzer, *J. Franklin Inst*. 1945; 240: p255-346.

[14] Thomson W. On a new form of centrifugal governor. *Trans. Inst. Eng. & Shipbuilders Scotland.* 1868; 12: p 67-71.

[15] Maxwell JC. On governors, *Proc. Roy. Soc.London,* 1868; 16: p270-283.

[16] Kelvin, William Thomson, Baron Kelvin. *Baltimore Lectures on Molecular* Dynamics *and the Wave Theory of Light*. *Founded on Mr. A.S. Holloway's Stenographic Report of Twenty Lectures Delivered in Johnss Hopkins University, Baltimore, in October 1884. Followed by Twelve Appendices on Allied Subjects*, London, C. J. Clay, 1904.

[17] Care C. A multi-stranded conology of analogue computing. In Care C. *Technology for Modelling, Electrical Analogies, Engineering Practice, and the Development of Analogue Computing*. London: Springer; 2010. Section 2.3.3.1, p31-32. ISBN 978-184882-47-3.

[18] Care C. *Technology for Modelling, Electrical Analogies, Engineering Practice, and the Development of Analogue Computing*, London, Springer; 2010. ISBN 978-1-84882-947-3.

[19] Mindell DA. *Between Human and Machine: Feedback, Control and Computing before Cybernetics*, Baltimore, Johns Hopkins University Press; 2002. 439p. ISBN 0-8018-6895-5.

[20] Small .S. *The Analogue Alternative: The Electric Analogue Computer in* Britain *and The USA, 1930–1975.* London, Routledge; 2000. 336p. ISBN 9780415862998.

[21] Williams RW. *Analogue Computation, Techniques and Components.* London, Hey-Wood; 1961, 271p.

[22] Paynter HM. The differential analyzer as an active mathematical instrument. *IEEE Control Syst. Mag.* 1989; 9(7): p3-8. DOI 10.23919/acc.1989.4790497

[23] Murray-Smith DJ. Early developments in computing: the Thomson brothers' contribution to today's software tools for dynamic system simulation. *Inst. of Eng. in Scotland, J. Eng*. 2023; 162: p21-27, Paper 1714.

[24] Bush V. Instrumental analysis. *Bull. Am Math. Soc.*1936; 42(10): p649-669.

[25] Thomson W. Electrical units of measurement, Lecture to Inst. Civil Eng., 3rd May 1883. From *Popular Lectures and Addresses* . Inst. Civil Eng. 1889; 1: p80-81.

[26] Thomson W, Tait G. *Treatise on Natural Philosophy*, Vol 1, Oxford, Clarendon Press; 1867. 727p

EUROSIM Data and Quick Info

MATHMOD 2025 11th Vienna International Conference on Mathematical Modelling
February 19 – 21 , 2025, Vienna, Austria www.mathmod.at

Simulation in Production and Logistic - 21st ASIM Expert Conference ASIM SPL
September 24-26, 2025, Dresden, Germany www.asim-gi.org/spl2025

I3M 2025 International Multidisciplinary Modeling & Simulation Multiconference
Fes, Morocco, September 17-19, 2025 www.msc-les.org/i3m2025

EUROSIM Congress 2026
12th Congress of the European Simulation Societies
July 2026, Italy
www.eurosim.info

ASIM STS/GMMS/EDU Workshop 2025
April 10-11, 2025, Oberpfaffenhofen, Germany
www.asim-gi.org/ws2025

EUROSIM – the **Federation of European Simulation Societies** was set up in 1989.

The purpose of **EUROSIM** is to provide a European forum for simulation societies and groups to promote modelling and simulation in industry, research, and development – by publication and conferences.

www.eurosim.info

EUROSIM members may be national simulation societies and regional or international societies and groups dealing with modelling and simulation.

Full Members are ASIM, CEA-SMSG, CSSS, DBSS, KA-SIM, LIOPHANT, LSS, PTSK, NSSM, SIMS, SLOSIM, UKSIM. Observer Members are ALBSIM and ROMSIM. Former Members (societies in re-organisation) are: CROS-SIM, FRANCOSIM, HSS, ISCS.

EUROSIM is governed by a B**oard** consisting of one representative of each member society, president, past president, and **SNE** representative.

President	Agostino Bruzzone (LIOPHANT) *agostino@itim.unige.it*
Past President	M. Mujica Mota (DBSS), *m.mujica.mota@hva.nl*
Secretary	Marina Massei (LIOPHANT), *massei@itim.unige.it*
Treasurer	Felix Breitenecker (ASIM) *felix.breitenecker@tuwien.ac.at*
Webmaster	Irmgard Husinsky (ASIM), *irmgard.husinsky@tuwien.ac.at*

SNE – Simulation Notes Europe is EUROSIM's membership journal with peer reviewed scientific contributions about all areas of modelling and simulation, including new trends as big data, cyber-physical systems, etc.

The EUROSIM societies distribute **e-SNE** in full version to their members as official membership journal. The basic version of **e-SNE** is available with open access. Publishers are **EUROSIM**, **ARGESIM** and **ASIM**,

www.sne-journal.org
office@sne-journal.org

SNE-Editor: Felix Breitenecker (ASIM)
felix.breitenecker@eic@sne-journal.org

EUROSIM Congress and Conferences

Each year a major **EUROSIM** event takes place, as the **EUROSIM CONGRESS** organised by a member society, **SIMS EUROSIM Conference**, and **MATHMOD Vienna Conference (ASIM)**.

On occasion of the **EUROSIM Congress 2023**, the *11th EUROSIM Congress* in Amsterdam, July, 2023, a new **EUROSIM** president has been elected: we welcome Agostino Bruzzone, well known simulationist, as new president. His society **LIOPHANT** will organize the next EUROSIM Congress in 2026 in Italy.

Furthermore, **EUROSIM Societies** organize local conferences, and **EUROSIM** co-operates with the organizers of **I3M Conference** and **WinterSim Conference Series.**

EUROSIM Member Societies

ASIM
German Simulation Society
Arbeitsgemeinschaft Simulation

ASIM is the association for simulation in the German speaking area, servicing mainly Germany, Switzerland and Austria.

President	Felix Breitenecker, *felix.breitenecker@tuwien.ac.at*
Vice President	Sigrid Wenzel, *s.wenzel@uni-kassel.de* Thorsten Pawletta, *thorsten.pawletta@hs-wismar.de* Andreas Körner, *andreas.koerner@tuwien.ac.at*

ASIM is organising / co-organising the following international conferences: **ASIM SPL** Int. Conference 'Simulation in Production and Logistics' (biannual), **ASIM SST** 'Symposium Simulation Technique' (biannual), **MATHMOD** Int. Vienna Conference on Mathematical Modelling (triennial). Furthermore, **ASIM** is co-sponsor of *WSC* - Winter Simulation Conference and of the *I3M* and conference series.

ASIM Working Committees

GMMS: Methods in Modelling and Simulation
U. Durak, umut.durak@dlr.de
SUG: Simulation in Environmental Systems
J. Wittmann, wittmann@informatik.uni-hamburg.de
STS: Simulation of Technical Systems
W. Commerell, commerell@hs-ulm.de
SPL: Simulation in Production and Logistics
S. Wenzel, s.wenzel@uni-kassel.de
EDU: Simulation and Education
A. Körner, andreas.koerner@tuwien.ac.at
Working Group Big Data: Data-driven Simulation in Life Sciences, N. Popper, niki.popper@dwh.at
Other Working Groups: Simulation in Business Administration, in Traffic Systems, for Standardisation, etc.

Contact Information

www.asim-gi.org

info@asim-gi.org, admin@asim-gi.org

ASIM – Office Germany, Univ. Bundeswehr Munich, Inst. für Technische Informatik, Tobias Uhlig, Werner-Heisenberg Weg 39, 85577 Neubiberg, Germany

ASIM – Office Austria, dwh Simulation Services, F. Breitenecker, N. Popper, Neustiftgasse 57-59, 1070, Wien, Austria

CEA-SMSG – Spanish Modelling and Simulation Group

CEA is the Spanish Society on Automation and Control. The association is divided into national thematic groups, one of which is centered on Modeling, Simulation and Optimization (**CEA-SMSG**).

President	José L. Pitarch, *jlpitarch@isa.upv.es*
Vice President	Juan Ignacio Latorre, *juanignacio.latorre@unavarra.es*

Contact Information

www.ceautomatica.es/grupos/

simulacion@cea-ifac.es

CEA-SMSG / Emilio Jiménez, Department of Electrical Engineering, University of La Rioja, San José de Calasanz 31, 26004 Logroño (La Rioja), Spain

CSSS – Czech and Slovak Simulation Society

CSSS is the Simulation Society with members from the two countries: Czech Republic and Slovakia. The CSSS history goes back to 1964.

President	Michal Štepanovský *michal.stepanovsky@fit.cvut.cz*
Vice President	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*

Contact Information

cssim.cz

michal.stepanovsky@fit.cvut.cz

CSSS – Český a Slovenský spolek pro simulaci systémů, Novotného lávka 200/5,
11000 Praha 1, Česká republika

DBSS – Dutch Benelux Simulation Society

The *Dutch Benelux Simulation Society* (**DBSS**) was founded in July 1986 in order to create an organisation of simulation professionals within the Dutch language area.

President	M. Mujica Mota, *m.mujica.mota@hva.nl*
Vice President	A. Heemink, *a.w.heemink@its.tudelft.nl*
Secretary	P. M. Scala, *paolo.scala@fedex.com*

Contact Information

www.DutchBSS.org

a.w.heemink@its.tudelft.nl

DBSS / A. W. Heemink, Delft University of Technology, ITS – twi, Mekelweg 4, 2628 CD Delft, The Netherlands

KA-SIM Kosovo Simulation Society

The Kosova Association for Modeling and Simulation (**KA-SIM**) is closely connected to the University for Business and Technology (UBT) in Kosovo.

President	Edmond Hajrizi, *ehajrizi@ubt-uni.net*
Vice President	Muzafer Shala, *info@ka-sim.com*

Contact Information

www.ubt-uni.net

ehajrizi@ubt-uni.net

Dr. Edmond Hajrizi
Univ. for Business and Technology (UBT)
Lagjja Kalabria p.n., 10000 Prishtina, Kosovo

LIOPHANT Simulation

LIOPHANT Simulation is a non-profit association born in order to be a trait-d'union among simulation developers and users; **LIOPHANT** is devoted to promote and diffuse the simulation techniques and methodologies; the Association promotes exchange of students, sabbatical years, organization of International Conferences, courses and internships focused on M&S applications.

President	A.G. Bruzzone, *agostino@itim.unige.it*
Director	E. Bocca, *enrico.bocca@liophant.org*

Contact Information

www.liophant.org

info@liophant.org

LIOPHANT Simulation, c/o Agostino G. Bruzzone, DIME, University of Genoa, Savona Campus, via Molinero 1, 17100 Savona (SV), Italy

LSS – Latvian Simulation Society

The Latvian Simulation Society (LSS) has been founded in 1990 as the first professional simulation organisation in the field of Modelling and simulation in the post-Soviet area.

President	Artis Teilans, *Artis.Teilans@rta.lv*
Vice President	Oksana Kuznecova, *Oksana.Kuznecova@rtu.lv*

Contact Information

www.itl.rtu.lv/imb/

Artis.Teilans@rta.lv, Egils.Ginters@rtu.lv

LSS, Dept. of Modelling and Simulation, Riga Technical University, Kalku street 1, Riga, LV-1658, Latvia

NSSM – National Society for Simulation Modelling (Russia)

NSSM – The National Society for Simulation Modelling (Национальное Общество Имитационного Моделирования – НОИМ) was officially registered in Russia in 2011.

President	R. M. Yusupov, *yusupov@iias.spb.su*
Chairman	A. Plotnikov, *plotnikov@sstc.spb.ru*

Contact Information

www.simulation.su

yusupov@iias.spb.su

NSSM / R. M. Yusupov, St. Petersburg Institute of Informatics and Automation RAS, 199178, St. Petersburg, 14th line, h. 39

PTSK – Polish Society for Computer Simulation

PTSK is a scientific, non-profit association of members from universities, research institutes and industry in Poland with common interests in variety of methods of computer simulations and its applications.

President	Tadeusz Nowicki, *Tadeusz.Nowicki@wat.edu.pl*
Vice President	Leon Bobrowski, *leon@ibib.waw.pl*

Contact Information

www.ptsk.pl

leon@ibib.waw.pl

PSCS, 00-908 Warszawa 49, ul. Gen. Witolda Urbanowicza 2, pok. 222

SIMS – Scandinavian Simulation Society

SIMS is the Scandinavian Simulation Society with members from the five Nordic countries Denmark, Finland, Norway, Sweden and Iceland. The **SIMS** history goes back to 1959.

President	Tiina Komulainen, *tiina.komulainen@oslomet.no*
Vice President	Erik Dahlquist, *erik.dahlquist@mdh.se*

Contact Information

www.scansims.org

vadime@wolfram.com

Vadim Engelson, Wolfram MathCore AB,
Teknikringen 1E, 58330, Linköping, Sweden

SLOSIM – Slovenian Society for Simulation and Modelling

The Slovenian Society for Simulation and Modelling was established in 1994. It promotes modelling and simulation approaches to problem solving in industrial and in academic environments by establishing communication and cooperation among corresponding teams.

President	Goran Andonovski, *goran.andonovski@fe.uni-lj.si*
Vice President	Božidar Šarler, *bozidar.sarler@fs.uni-lj.si*

Contact Information

www.slosim.si

slosim@fe.uni-lj.si, vito.logar@fe.uni-lj.si

SLOSIM, Fakulteta za elektrotehniko, Tržaška 25,
SI-1000, Ljubljana, Slovenija

UKSIM - United Kingdom Simulation Society

The UK Modelling & Simulation Society (**UKSim**) is the national UK society for all aspects of modelling and simulation, including continuous, discrete event, software and hardware.

President	David Al-Dabass, *david.al-dabass@ntu.ac.uk*
Secretary	T. Bashford, *tim.bashford@uwtsd.ac.uk*

Contact Information

uksim.info

david.al-dabass@ntu.ac.uk

UKSIM / Prof. David Al-Dabass, Computing & Informatics, Nottingham Trent University, Clifton lane, Nottingham, NG11 8NS, United Kingdom

Observer Members

ROMSIM – Romanian Modelling and Simulation Society

ROMSIM has been founded in 1990 as a non-profit society, devoted to theoretical and applied aspects of modelling and simulation of systems.

Contact Information

florin_h2004@yahoo.com
ROMSIM / Florin Hartescu, National Institute for Research in Informatics, Averescu Av. 8 – 10, 011455 Bucharest, Romania

ALBSIM – Albanian Simulation Society

The Albanian Simulation Society has been initiated at the Department of Statistics and Applied Informatics, Faculty of Economy at the University of Tirana, by Prof. Dr. Kozeta Sevrani.

Contact Information

kozeta.sevrani@unitir.edu.al

Albanian Simulation Goup, attn. Kozeta Sevrani, University of Tirana, Faculty of Economy , rr. Elbasanit, Tirana 355, Albania

Former Societies / Societies in Re-organisation

- **CROSSIM** – *Croatian Society for Simulation Modelling*
  Contact: Tarzan Legović, *Tarzan.Legovic@irb.hr*
- **FrancoSim** – Société Francophone de Simulation
- **HSS** – Hungarian Simulation Society
  Contact: A. Gábor, *andrasi.gabor@uni-bge.hu*
- **ISCS** – Italian Society for Computer Simulation

The following societies have been formally terminated:

- **MIMOS** –Italian Modeling & Simulation Association; terminated end of 2020.

Association Simulation News

ARGESIM is a non-profit association generally aiming for dissemination of information on system simulation – from research via development to applications of system simulation. **ARGESIM** is closely co-operating with **EUROSIM**, the Federation of European Simulation Societies, and with **ASIM**, the German Simulation Society.
ARGESIM is an 'outsourced' activity from the *Mathematical Modelling and Simulation Group* of TU Wien, there is also close co-operation with **TU Wien** (organisationally and personally).

→ **www.argesim.org**

→ *office@argesim.org*

→ ARGESIM/Math. Modelling & Simulation Group,
Inst. of Analysis and Scientific Computing, TU Wien
Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria
Attn. Prof. Dr. Felix Breitenecker

ARGESIM is following its aims and scope by the following activities and projects:

- Publication of the scientific journal **SNE** – *Simulation Notes Europe* (membership journal of **EUROSIM**, the *Federation of European Simulation Societies*) – *www.sne-journal.org*
- Organisation and Publication of the **ARGESIM Benchmarks** for *Modelling Approaches and Simulation Implementations*
- Publication of the series **ARGESIM Reports** for monographs in system simulation, and proceedings of simulation conferences and workshops
- Publication of the special series **FBS Simulation** – Advances in Simulation / Fortschrittsberichte Simulation - monographs in co-operation with ASIM, the German Simulation Society
- Support of the Conference Series **MATHMOD Vienna** (triennial, in co-operation with **EUROSIM**, **ASIM**, and **TU Wien**) – www.mathmod.at
- Administration of **ASIM** (German Simulation Society) and administrative support for **EUROSIM** *www.eurosim.info*
- Simulation activities for TU Wien

ARGESIM is a registered non-profit association and a registered publisher: **ARGESIM Publisher Vienna**, root ISBN 978-3-901608-xx-y and 978-3-903347-xx-y, root DOI 10.11128/z…zz.zz. Publication is open for **ASIM** and for **EUROSIM Member Societies**.

SNE – Simulation Notes Europe

The scientific journal **SNE – Simulation Notes Europe** provides an international, high-quality forum for presentation of new ideas and approaches in simulation – from modelling to experiment analysis, from implementation to verification, from validation to identification, from numerics to visualisation – in context of the simulation process. **SNE** puts special emphasis on the overall view in simulation, and on comparative investigations.
Furthermore, **SNE** welcomes contributions on education in/for/with simulation.

SNE is also the forum for the **ARGESIM Benchmarks** on *Modelling Approaches and Simulation Implementations* publishing benchmarks definitions, solutions, reports and studies – including model sources via web.

SNE Editorial Office / ARGESIM

→**www.sne-journal.org**

office@sne-journal.org, eic@sne-journal.org

Irmgard Husinsky (Web, Electronic Publishing)
Felix Breitenecker EiC (Organisation, Authors, Layout)
ARGESIM/Math. Modelling & Simulation Group,
Inst. of Analysis and Scientific Computing, TU Wien
Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria

SNE, primarily an electronic journal, follows an open access strategy, with free download in a basic version (B/W, low resolution graphics). **SNE** is the official membership journal of **EUROSIM**, the *Federation of European Simulation Societies*. Members of (most) **EUROSIM Societies** are entitled to download the full version of e-**SNE** (colour, high-resolution graphics), and to access additional sources of benchmark publications, model sources, etc. (group login for the 'publication-active' societies; please contact your society). Furthermore, **SNE** offers **EUROSIM Societies** a publication forum for post-conference publication of the society's international conferences, and the possibility to compile thematic or event-based **SNE Special Issues**.

Simulationists are invited to submit contributions of any type – *Technical Note*, *Short Note*, *Project Note*, *Educational Note*, *Benchmark Note*, etc. via **SNE**'s website:

→ **www.sne-journal.org**,

MATHMOD 2025

February 19-21, 2025, Vienna, Austria
www.mathmod.at

ASIM SUG Workshop 2025

March 26-28, 2025, Göttingen, Germany
fa-ui.gi.de

ASIM STS/GMMS/EDU Workshop 2025

April 10-11, 2025, Oberpfaffenhofen, Germany
www.asim-gi.org/ws2025

I3M 2025
International Multidisciplinary Modeling & Simulation Multiconference

September 17-19, 2025, Fes, Morocco
www.msc-les.org/i3m2025/

Simulation in Produktion und Logistik 2025

September 24-26, 2025, Dresden, Germany
www.asim-gi.org/spl2025

EUROSIM Congress 2026

July 2026, Italy
www.eurosim.info