15thIFK
Proc. of the 15th International Fluid Power Conference <Symposium/Conference>
<DOI>

System Simulation Meets Asset Administration Shell – Potential and Challenges of Standardized, Model-Based Development of Fluid Power Systems

Heiko Baum[1], Micha Koebele[1], Denis Ritz[2], Heiko Roecher[3], Jürgen Weber[2]

[1] *FLUIDON GmbH, heiko.baum@fluidon.com, micha.koebel@fluidon.com ,*
[2] *TU Dresden, Chair of Fluid-Mechatronic Systems, denis.ritz@tu-dresden.de, fluidtronik@mailbox.tu-dresden.de*
[3] *Röcher GmbH & Co. KG, heiko.roecher@roecher-pressen.de*

Abstract.

Dynamic simulation is an established tool in the development of fluid engineering systems. In combination with the Asset Administration Shell (AAS), new potential arises for more efficient, standardized, and automated modeling and parameterization. Using the example of a hydraulic forming press, this article shows how AAS-based data structures accelerate system simulation and improve its quality – and where existing AAS standards still need to be expanded in order to meet the requirements of fluid technology practice.

Keywords: Fluid 4.0, digitalization, asset administration shell (AAS), collaborative engineering, dataspace, sub-models, industry-relevant use cases

1. INTRODUCTION

The *dynamic system simulation method* [1], has established itself as a recognized tool in the development of technical systems. It enables complex physical relationships between individual components and their interactions to be visualized early in the development process – long before prototypes can be built or components procured. This means that digital prototypes are available for testing during the development phase. As a result, functional principles can be evaluated, design decisions can be validated, and potential system problems can be identified at an early stage.

1.1. Significance of dynamic system simulation in fluid technology

Despite its well-known advantages, system simulation has not yet been established as a standard tool in everyday development work in many companies. Instead, it is often used only selectively, for example to analyze specific issues that arise during the commissioning of a plant, but not continuously throughout the entire development process.

1.2. Challenges in practical application

One of the main reasons for the limited use of simulation-based methods is the high barrier to entry when it comes to model creation. In practice, the construction of simulation-capable models still involves considerable manual effort. In particular, the selection of suitable model components, their correct parameterization, and the documentation of relevant assumptions and simplifications require experience, specific domain knowledge, and a considerable amount of time.

These challenges not only affect the efficiency of the simulation process but also make it difficult to trace and reuse models. Decisions made during model creation are often implicit and inadequately documented, which makes it considerably more difficult to reuse or adapt existing models later on.

1.3. Administration shell as an enabler for automated simulation

Against this backdrop, the combination of dynamic simulation and Asset Administration Shell (*AAS*) [2] offers new perspectives for simulation-based development processes. AAS is a central concept of Industry 4.0 and enables the standardized digital description of technical assets throughout their entire life cycle. *Ritz* [3] explains the potential advantages and several norms and standards to be considered for the application of the administration shell throughout the product life cycle for hydraulic systems.

The structured and semantically described provision of information and metadata in machine-readable form allows numerous steps in model creation that were previously performed manually to be automated or at least systematically supported. This not only facilitates the initial model setup, but also improves the consistency, traceability, and reusability of simulation models across different use cases.

1.4. Objective and structure of the article

This article examines the interaction between system simulation and administration shells from a practical perspective. Using the example of simulating a hydraulic forming press, it shows how fluid power drive systems can be converted into a simulation-capable model in a largely automated manner with the help of AAS-based data structures. **Figure 1.1** shows the press in the simulation environment on the left side of the image and the list of administration shell submodels used in the application example on the right side. The legend distinguishes between submodels that have been standardized by the *IDTA* [4] and those that have been newly developed or expanded as part of the *Fluid 4.0* research project [5].

Using a simulation workflow from the *Fluidon Cube* development environment [6], the sequence of transformation steps from the machine's administration shell to the simulation model, its application in the design of the hydraulic system, and the return of results and models to the administration shell are explained.

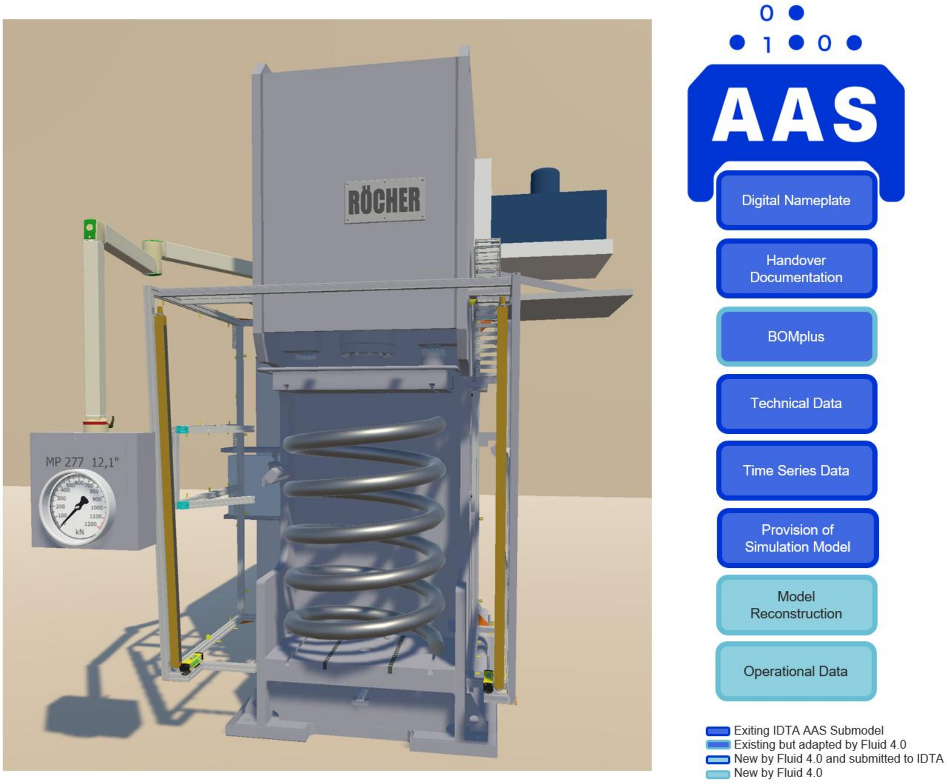


Figure 1.1. Fluid 4.0 example simulation of a hydraulic forming press

In addition, the article highlights where existing AAS standards are currently reaching their limits and which additional semantic descriptions and submodels are required to comprehensively support simulation-based engineering processes. The concepts developed in the Fluid 4.0 project are intended to provide impetus for the further development of existing AAS standards and contribute to establishing digital system simulation as a universal tool in mechanical and plant engineering.

2. STATE OF THE ART IN THE SIMULATION OF FLUID TECHNOLOGY SYSTEMS

2.1. Starting point: Fluid engineering circuit diagram and system complexity

In practice, a simulation project usually begins with a fluid technology circuit diagram containing all the components required to perform the drive task – often far more components than would be necessary for the actual simulation. One reason for this is that circuit diagrams are also used to generate the parts list. Accordingly, they contain additional components that are required for secondary functions but have no relevant influence on the dynamic behavior of the system.

2.2. Model design and selection of relevant system components

Figure 2.1 shows the typical steps in a simulation project, based on published descriptions [7], [8], [9].
Starting from the problem (see Figure 2.1-1) to be solved by means of simulation, the project engineer first has to ask himself the following question in the design phase (see Figure 2.1-2): Which components are actually relevant for solving the respective problem with the help of simulation?

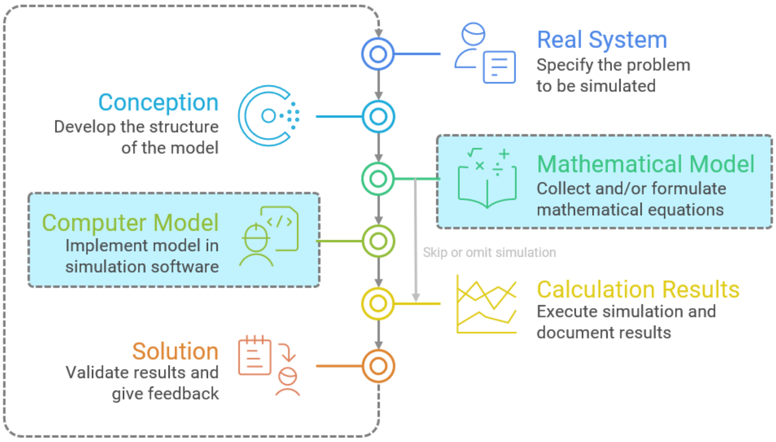


Figure 2.1. Sequence of work steps in a classic simulation process

This selection requires not only a deep understanding of the system but also experience in using the simulation environment and its model libraries and elements. In practice, this often leads to two typical error patterns. Either too many components are taken from the circuit diagram, resulting in unnecessarily large, computationally intensive, and numerically demanding models, or important components are omitted, meaning that crucial physical effects are not represented in the simulation.

2.3. Classic model creation process

Once the model concept has been defined, the classic simulation process begins with the creation of the mathematical model (see Figure 2.1-3) – regardless of whether a simulation will be performed later or not. The underlying equations are also required if the system is to be evaluated manually or with the aid of simple computer-aided design scripts, sometimes in the form of spreadsheets. In the past, the subsequent preparation of the equations for numerical calculation on a computer using a dedicated simulation program (see Figure 2.1-4) presented an almost insurmountable hurdle for many project engineers. As a result, simulations were often not used.

2.4. Modern simulation environments and model libraries

Nowadays, designers no longer have to manually create and code simulation models, as modern simulation environments take care of this task. Their model libraries contain many ready-made component descriptions that represent the dynamic properties of common components. These two process steps are therefore highlighted in color in Figure 2.1.

However, the designer must check whether the required components are available in the model library and whether their mathematical description meets the requirements. There is a high potential for error here, especially for occasional users, as it is usually difficult to choose between similar model types without in-depth background knowledge. Occasional users therefore often carry out additional, time-consuming test simulations to verify the behavior of individual component descriptions before using them in the overall system.

2.5. Parameterization of model components as a critical step

Another time-critical and error-prone step is the parameterization of the model (see Figure 2.1-5). As a rule, the required parameters must be manually researched and transferred from data sheets. For components that are mainly described by their geometry – such as cylinders – this can be done quickly. However, parameterization is much more complex for components with their own dynamics, such as control valves, regulated variable displacement pumps, or pilot-operated pressure valves. In practice, parameters such as step response times or frequency responses are often only available in graphical form. Their use in simulation therefore requires manual interpretation and conversion of the information obtained from diagrams, some of which are logarithmically scaled, into suitable model parameters.

In practice, this regularly leads to a time-consuming search for technical documentation on manufacturer websites and the need for internal consultations and validations using test simulations. However, the problem remains that some parameters and values are not provided by the respective manufacturers.

2.6. Motivation for an AAS-based simulation approach

The steps described above illustrate that, despite powerful simulation tools, the current state of the art in the simulation of fluid technology systems continues to be strongly influenced by manual activities, implicit expert knowledge, and individual decisions made by the project engineer. In particular, the selection of relevant system components, the interpretation of manufacturer information, and the parameterization of complex components are time-consuming and error-prone process steps.

At the same time, much of the information required for this – such as system structure, parts lists, or technical specifications – is available in digital form, but is often not structured consistently, lacks a unique semantic description, or is not linked in a machine-readable form. As a result, the consistent, automated use of this data throughout the entire simulation process has been limited to date. Concepts such as AAS open up new perspectives for data-driven, consistent, collaborative, and reproducible system simulation.

The following chapter shows how administrative shell data can be used to automate key steps in modeling and parameterization and to systematically support the simulation process.

3. SYSTEM SIMULATION REIMAGINED: AAS-BASED

In order to establish dynamic simulation as a standard tool across a wide range of fluid technology development processes, model design, model construction, and subsequent parameterization must be automated to a much greater extent and systematically supported. Only in this way can the full potential of simulation-supported development be exploited – both in early concept phases and in the detailed design and optimization of complex fluid engineering systems, as well as in the later further development of series products and the updating of machines designed in "batch size 1" for special applications.

The AAS-based approach opens up new possibilities here by structurally accessing existing engineering data and making it directly usable for the simulation process. **Figure 3.1** provides an overview of the essential steps in the AAS-based simulation process.

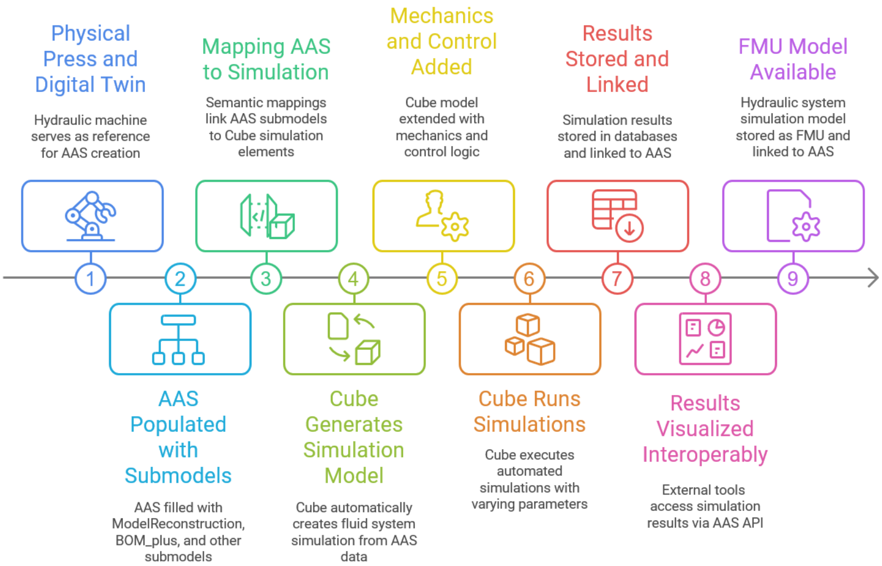


Figure 3.1. Digital Twin Workflow for a Hydraulic Forming Press

3.1. AAS-based system description

The starting point for the AAS-based simulation process here is, for example, the administration shell of a hydraulic forming press (see Figure 3.1-1), which the designer can use to access all relevant submodels. In addition to standardized submodels, the submodels newly developed or expanded in the *Fluid 4.0* project are also used (see Figure 3.1-2). This is necessary because existing IDTA submodels currently only provide incomplete simulation-relevant information.

The *ModelReconstruction* submodel plays a central role within the process. It contains the classification of components standardized in *ECLASS* [10] as well as their interconnection via hydraulic connections, thus defining the underlying system structure. The submodel is also linked to the parts list, which is mapped in the *BOM_plus* submodel. The *BOM_plus* submodel is the extension of the already standardized *BOM* submodel [11] proposed in the *Fluid 4.0* project.

The system structure and parts list for the application example were created using an external project planning tool (see [12]) and then stored in the administration shell of the hydraulic forming press. **Figure 3.2** shows the view of the *BOM_plus* submodel of the administration shell on the AAS repository server of the *Fluid 4.0* research project.

In the *BOM_plus* submodel, all components of the system are referenced to real existing components. The associated administration shells of these components in turn contain the parameters, model information, and settings required for simulation. This ensures that simulation-relevant information is always clearly assigned to a real, physical asset and can be used consistently across different engineering tools.

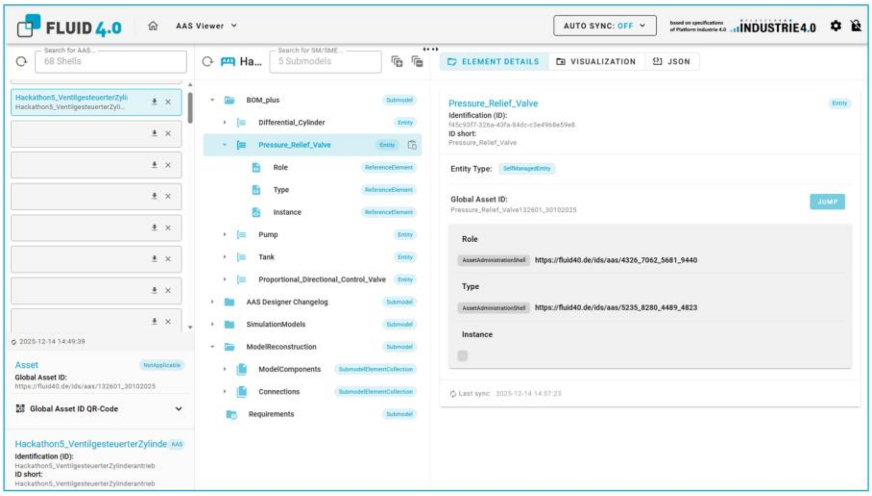


Figure 3.2. Administration shell of the hydraulic forming press

The classification into *role*, *type*, and *instance* describes the respective level of detail of a component in the development process. *Role* refers to an abstract component that merely represents the generic component class assigned in the *ModelReconstruction* submodel, for example, a pump. *Type* concretizes this abstract description to a real existing variant of the component class, for example, a gear, vane, or piston pump, which can be assigned to a specific type in a specific design from a manufacturer. If specific settings are then defined for such a component by the designer, this results in an *Instance* that is later integrated into the administration shell of the specific, real component. This shell is uniquely associated with a real, physical component with a unique serial number.

3.2. Mapping component behavior in the simulation

For each component contained in the system, there are basically two ways to map its fluid technical behavior in the simulation model:

1. **Direct model provision by the manufacturer:** A Functional Mockup Unit (*FMU*) [14] provided by the manufacturer can be loaded via the submodel *ProvisionOfSimulationModel* [13]. This contains an encapsulated mathematical description of the component and its parameterization and enables its use in the simulation – while preserving the manufacturer's know-how.
2. **Parameterization based on technical data:** If no FMU is available, the necessary parameters can be read from the *TechnicalData* submodel [15] and applied to a generic model from the library of the simulation environment used, which maps the real dynamic behavior as accurately as possible.

In the application example, the second option was deliberately chosen in order to demonstrate how flexibly technical component data can be assigned to existing generic mathematical model descriptions. **Figure 3.3** shows the *TechnicalData* submodel of the administration shell of a pump as an example. The parameters *NominalRotationalSpeed*, *Displacement,* and *MaxOperatingPressure,* among others, are relevant for the simulation.

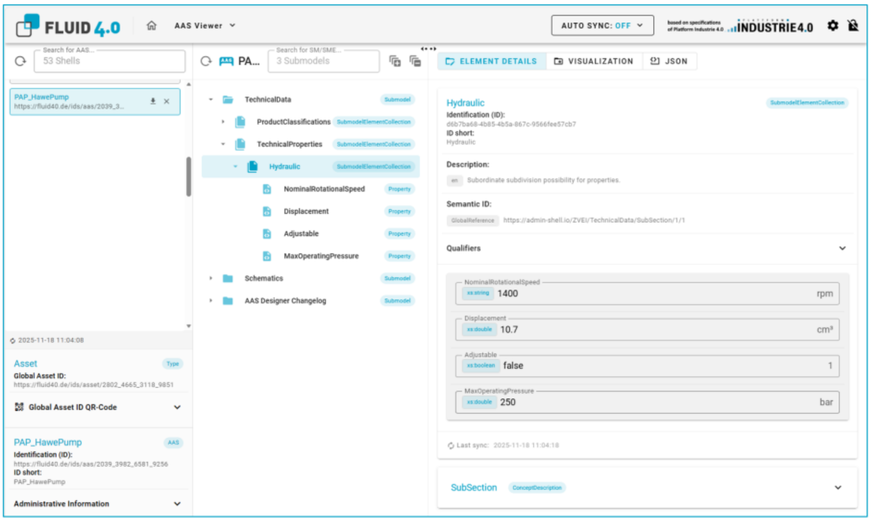


Figure 3.3. TechnicalData submodel of the administration shell of a pump

Access to this information and its further processing in the simulation workflow are explained in more detail in **Figure 3.4**.

First, the administration shell submodel is retrieved from the AAS server, read, and temporarily stored (see Figure 3.4-1). *ComponentName* references the unique component name from *ModelReconstruction*. In addition to name, value, and unit, each entry also contains a *GlobalReference* as an ECLASS-based semantic IRDI [16].

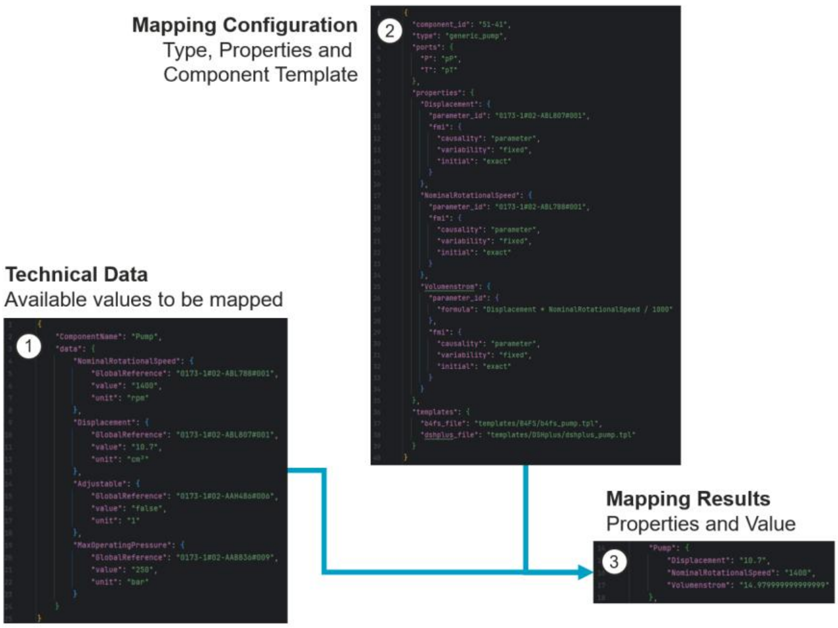


Figure 3.4. Mapping of technical data to the simulation model component

The declarative assignment of a real component type to a component from the simulation environment library is done via a mapping configuration (see Figure 3.4-2). In this configuration, the ECLASS IRDI is assigned to a simulation parameter as *parameter_id*.
Each parameter listed in the *properties* section also contains the semantic information *causality*, *variability,* and *initial*, whose meanings correspond to the definitions in the FMI standard [14].
The *templates* section references possible components of the simulation environment that can mathematically map the real component. The abbreviation *B4FS* refers to components that were developed in the *BaSys4FluidSim* project [17] to integrate component FMUs into the system simulation.

3.3. Formula-based parameter derivation in the mapping process

In the example of the pump, the real component is mapped to the simplest available pump model in the component library, which only provides a constant volume flow. However, the volume flow value required for this is not directly contained in the *TechnicalData* submodel. To enable consistent parameterization nonetheless, the Python-based mapping script allows formula-based derivation of missing parameter values (see Figure 3.4-2, line 27). Assuming that the pump is operated at its nominal speed, the required volume flow for the selected simulation model can be calculated in this way. The calculation rule for this is stored directly in the mapping configuration.
The results of this mapping step are again temporarily stored (see Figure 3.4-3) and transferred to the simulation model in a later process step.
Corresponding mapping configurations were created for all other components of the example system. For components not yet covered, users can independently define comparable mapping templates and integrate them into the workflow. Once created, these are then directly available for automatic model generation.

3.4. Automated model construction in the simulation environment

The actual model construction takes place within the *Cube* simulation environment (see Figure 3.1-4). A workflow script analyzes the structural information from the *ModelReconstruction* submodel, generates a network plan from it, and transfers it to a parameterized simulation model structure.
The entire process – from accessing the AAS repository to the fully parameterized simulation topology – is AAS-supported and requires no manual intervention. The hydraulic part of the simulation model is then immediately available for further analysis.

3.5. Extension to the multidisciplinary simulation model

However, for a meaningful evaluation – for example, in terms of energy efficiency, performance, or cycle times – the hydraulic simulation model must be supplemented with additional control and mechanical components. This extension is carried out manually by the user within the cloud environment (see Figure 3.1-5).
In the future, however, other domains, such as mechanics and control engineering, should also be mappable via corresponding administration shell submodels, so that fully automated multidisciplinary model generation will be possible at a later date.

3.6. Use of the generated model for simulation and optimization

With the completed simulation model (**Figure 3.5**), different simulation scenarios can then be carried out (see Figure 3.1-6). These range from the analysis of individual work cycles to automatic variant comparisons and optimization runs.

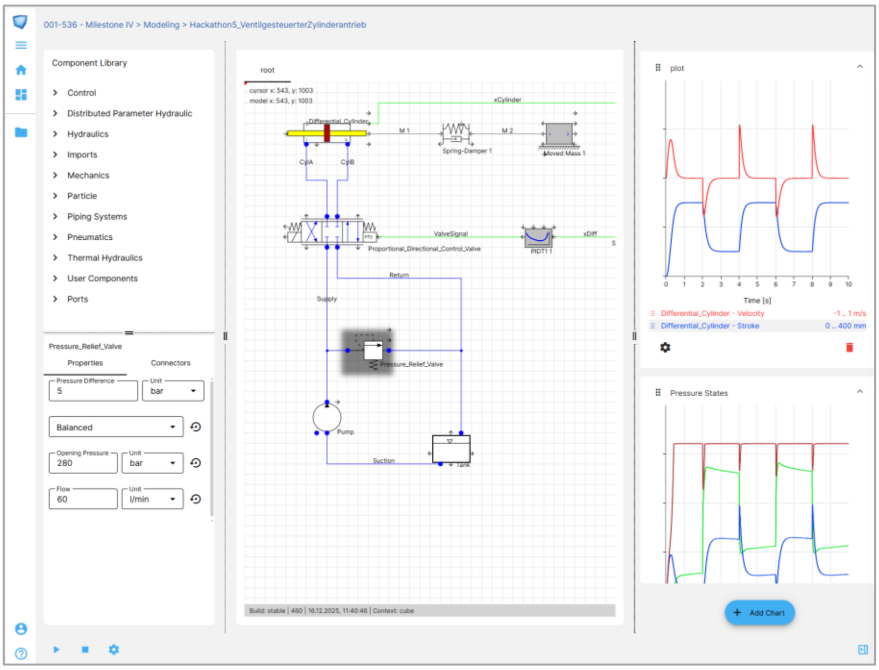


Figure 3.5. Simulation of the cylinder drive of the hydraulic forming press

The goal is to identify a system configuration that achieves the desired behavior while taking all technical constraints into account. At the same time, the simulation model should remain available as a consistently documented digital model in the AAS data ecosystem.

4. FEEDBACK OF RESULTS BACK TO THE ADMINISTRATION SHELL

Following the simulation, the simulation results are automatically checked and evaluated (see Figure 3.1-7).

4.1. Consistency check of the model parameters

A key step in this process is to check whether parameters that originally came from administration shells have been changed by the user. This check is made possible by the additional semantic information stored in the mapping configuration (see Figure 3.4-2). A distinction is made between *fixed product parameters* (e.g., piston diameter of a cylinder) and *freely adjustable settings* (e.g., opening pressure of a pressure valve or preload pressure of a hydraulic accumulator).
If the user changes parameters that are declared as *fixed*, the workflow generates a warning because the parameterization no longer matches the real component designed in *BOM_plus*. In the future – and already tested in a prototype as part of the *Fluid 4.0* research project –

the workflow can support an automated search for a component that matches the changed parameters. If a suitable alternative is found, *BOM_plus* can be updated accordingly so that the simulation and real project planning are consistent with each other again.

4.2. Rule-based verification of the system configuration

Beyond pure consistency checks, the Python scripts integrated into the workflow enable a rule-based evaluation of the system configuration. This checks whether parameter values comply with system-wide constraints and component-specific limits. **Figure 4.1** shows an example of this using the opening pressure of a pressure relief valve.

	ComponentName	MaxOperatingPressure	CrackingPressure
0	Differential_Cylinder	210	
1	Pressure_Relief_Valve	315	280
2	Proportional_Directional_Control_Valve	350	
3	Pump	250	

Figure 4.1. Checking the pressure resistance as a function of the selected system pressure

The opening (cracking) pressure is declared as *a variable* in the mapping configuration and can be set individually for each system by the project engineer in the real system. In the application example, a value of 280 bar was selected. However, a comparison with the maximum permissible pressure of the planned components shows that this value is not permissible for the cylinder and pump (see Figure 4.1).
Such rule-based checks could in principle also be implemented in today's project planning or circuit diagram creation tools, provided that they interact with the administration shell and use its data. In system simulation, however, this approach represents an innovation, as much of the component information required for it is not available in the classic simulation process. It is not necessary for numerical calculation and is therefore usually not represented in conventional simulation models.
In conjunction with dynamic simulation, the improved data available from the administration shell enables more advanced approaches to automated system evaluation. One example is the evaluation of the simulated cylinder speed, which checks whether permissible speed limits – for example, from the administration shells of the installed cylinder seals – are adhered to during the simulated work cycle.
Another example is the detection of critical pressure peaks in the simulated pressure curves, which can occur as a result of rapid reversal processes, for example. Such pressure peaks lead to material fatigue in hose lines or component housing. Based on the analyzed pressure curves, projections can be made regarding the number of critical load cycles in production operation. This information can in turn serve as a basis for condition-based maintenance strategies (predictive maintenance) or for analyses within the scope of the ongoing work on the Fluid 4.0 project on *the smart circular economy for the fluid industry* [18].

4.3. Feedback of results and models into the AAS data ecosystem

Once all tests have been successfully completed, the relevant results are automatically fed back into the administration shells. This includes, among other things:

- **Simulation documentation**, which is stored in the *HandOverDocumentation* submodel [19] and can be viewed by sales or the customer

- **Time series-based simulation results**, which are persisted in a time series database (e.g., *InfluxDB* [20]) and made available for detailed analysis via the *TimeSeriesData* submodel [21]
- **Settings determined in the simulation**, which are transferred to the *OperationalData* submodel developed specifically for this purpose as part of Fluid 4.0 and assigned to the instance AAS of the respective component so that they are available to the project engineer during commissioning
- **The generated simulation model**, which is stored as an FMU [14] on a Nextcloud instance [22] and referenced in the machine's AAS via the *ProvisionOfSimulationModel* submodel [13]. From there, it can be used, for example, for controller development or simulation-based optimizations.

The consistent feedback of all relevant data to the AAS ensures that digital project planning can be seamlessly integrated into downstream engineering processes. Project planning tools can then access the verified system configuration and its parameters to fully detail the plant – including all components required for implementation, such as screw connections, shut-off valves, or safety elements.
The administration shell thus functions as a continuous data hub between simulation, documentation, and actual machine construction.

5. Classification and Outlook

5.1. Simplification of the simulation process through AAS integration

The AAS-based system simulation presented in this article shows how the classic simulation process described in Chapter 2 can be fundamentally simplified and systematized. Many tool-specific and experience-based work steps that were previously performed manually are removed from the actual simulation process through the use of the administration shell and converted into standardized, interoperable data structures.
Figure 5.1 presents this new AAS-based simulation process, in which the remaining manual steps of the classic simulation process from Figure 2.1 are replaced by the automated steps presented in Chapter 3 and Chapter 4.

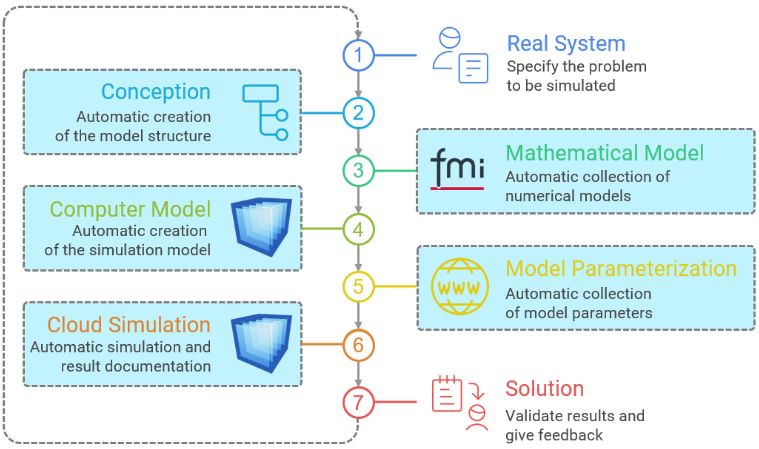


Figure 5.1. Automated simulation process with administration shell interaction

The central element of this simplification is the clear separation between *engineering data management* and *numerical simulation*. Information on the system structure, parts list, technical parameters, and semantic relationships is explicitly stored in the AAS and is available to the simulation workflow in a structured and machine-readable form. This makes model construction and parameterization largely automated and reproducible.
An example from the *Fluid 4.0* project (**Figure 5.2**) quantifies the difference to the classic approach (cf. Figure 2.1): A project engineer was tasked with researching, interpreting, manually reading, verifying component by component through test simulations, and transferring all component information required for the simulation from publicly available manufacturer data sheets based on the circuit diagram and parts list of the existing hydraulic forming press. The time required for this was recorded and forms the basis for comparison with the new AAS-based approach.

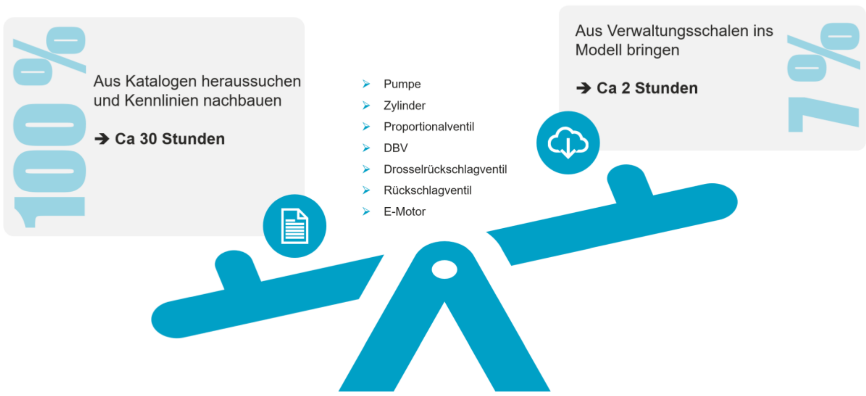


Figure 5.2. Time required for parameterization of components

Thanks to the AAS-based approach, the project engineer was able to begin his actual task – solving the problem or optimizing the system design – almost immediately after communicating the problem with the existing system design.

5.2. Insights gained

The practical implementation of the AAS-based simulation workflow in the *Fluid 4.0* project provides several important insights:

- Firstly, it shows that the combination of administration shell and dynamic system simulation goes far beyond mere automation of model creation. Feeding simulation results, settings, and models back into the AAS creates a consistent digital data space that links simulation, project planning, commissioning, and operation.
- Second, the improved data situation enables rule-based verification of the system configuration, which classic simulation approaches cannot achieve. Limit violations, inconsistent parameterizations, or potentially damaging operating conditions can already be identified and evaluated on the basis of simulated time sequences. This expands the scope of simulation from pure behavior analysis to an integral tool for system validation.
- Third, the AAS-based approach significantly improves the traceability and reusability of simulation-based development work. Decisions, assumptions, and results

are explicitly documented and clearly referable. This facilitates both internal collaboration and the transfer of engineering knowledge to downstream processes or external partners.

5.3. Gap analysis and contribution to standardization

The experience gained during the project reveals the following challenges and areas where action is needed:

- Available AAS submodels reach their limits in their current form and must be expanded in order to consistently map simulation-relevant information. For certain requirements – documentation of the system configuration or storage of plant-specific settings – there are currently no standardized submodels.
- The current AAS data situation for fluid engineering components is still insufficient. Many features relevant to simulation (e.g., efficiency data or characteristic curves) are either not included in current administration shells or are not described in the required granularity. In some cases, suitable semantic keys are missing in existing classification systems.

These gaps were closed in the *Fluid 4.0* project by means of project-specific extensions and newly developed submodels. All identified shortcomings and extensions deemed necessary will be systematically communicated to the IDTA and ECLASS after project completion so that they can be incorporated into future standardization activities. Among other things, the authors are already contributing their findings to the relevant IDTA submodel groups (Automation Engineering) or have already promoted the establishment and management of relevant working groups (Characteristic Curves) as a result of the project.

6. SUMMARY

This article shows how the dynamic system simulation of fluid power drive systems can be rethought and systematically simplified through the consistent use of the Asset Administration Shell. Based on the challenges of classic simulation processes, an AAS-based workflow was presented that integrates model design, parameterization, simulation, and result feedback in a continuous digital process.

Using the example of a hydraulic forming press, it was demonstrated how administration shell data can be used for automated model generation and how simulation results, validated settings, and simulation models themselves can be fed back into the AAS data ecosystem. This creates a consistent digital data space that establishes simulation not as an isolated tool, but as an integral part of modern engineering processes.

The results of the *Fluid 4.0* project highlight both the considerable potential of AAS-based system simulation and the need for further development of current standards. The insights gained and prototypical enhancements provide concrete impetus for standardization and contribute to establishing simulation-supported development as an integrated tool in mechanical and plant engineering.

7. ACKNOWLEDGEMENT

This work has been funded by the European Union (EU) and supported by the German Federal Ministry for Economic Affairs and Energy (BMWE) in the Fluid4.0 project – Revolutionizing Fluid Engineering: Implementation of digitalization for fluid power 4.0 in the cross-industry and cross-manufacturer data space using asset administration shells, submodels, and demonstrators (project numbers 13IK039E, 13IK039J, and 13IK039L). The views and opinions expressed are solely those of the authors and do not necessarily reflect the views of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.

8. REFERENCES

[1] 'Dynamical system simulation', *Wikipedia*. Feb. 23, 2025. Accessed: Jan. 03, 2026. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Dynamical_system_simulation&oldid=1277269006

[2] IDTA, 'Specification of the Asset Administration Shell - Part 1: Metamodel'. Industrial Digital Twin Association, 2025. [Online]. Available: https://industrialdigitaltwin.org/en/content-hub/aasspecifications/specification-of-the-asset-administration-shell-part-1-metamodel-idta-number-01001

[3] 'Accelerating Product Development: The Effort of Asset Administration Shells for aggregated Systems', in *The 19th Scandinavian International Conference on Fluid Power, SICFP'25*, Linköping University, Sweden: River Publishers, 2025.

[4] 'IDTA – The Standard for the Digital Twin', IDTA. Accessed: Jan. 03, 2026. [Online]. Available: https://industrialdigitaltwin.org/

[5] 'Joint Research Project Fluid 4.0 – Revolutionizing Fluid Engineering: Implementation of digitalization for fluid power 4.0 in the cross-industry and cross-manufacturer data space using asset administration shells, submodels and demonstrators.' Accessed: Jan. 03, 2026. [Online]. Available: https://fluid40.de/

[6] 'FLUIDON | Cube'. Accessed: Jan. 03, 2026. [Online]. Available: https://cube-lab.app/

[7] S. C. Brailsford, T. Eldabi, M. Kunc, N. Mustafee, and A. F. Osorio, 'Hybrid simulation modeling in operational research: A state-of-the-art review', *European Journal of Operational Research*, vol. 278, no. 3, pp. 721–737, Nov. 2019, doi: 10.1016/j.ejor.2018.10.025.

[8] S. Robinson, *Simulation: the practice of model development and use*. Chichester: Wiley, 2004.

[9] M. Landry, J.-L. Malouin, and M. Oral, 'Model validation in operations research', *European Journal of Operational Research*, vol. 14, no. 3, pp. 207–220, 1983, doi: https://doi.org/10.1016/0377-2217(83)90257-6.

[10] ECLASS, ‘ECLASS Standard’. Sep. 24, 2025. [Online]. Available: https://eclass.eu/en/eclass-standard

[11] IDTA, ‘AAS-SM - Hierarchical Structures enabling Bills of Material’. [Online]. Available: https://industrialdigitaltwin.org/en/wp-content/uploads/sites/2/2025/10/IDTA-02006-3-0-1_Submodel_Digital-Nameplate.pdf

[12] P. L. Remde *et al.*, ‘Digital fluid technology in the data space: From semantic component search to the development of mobile hydraulic systems using AAS’, presented at the 15th IFK 2026, Aachen, Mar. 2026.

[13] IDTA, ‘AAS-SM - Provision of Simulation Models’. Industrial Digital Twin Association, Frankfurt am Main, Germany, Oct. 2022. [Online]. Available: https://industrialdigitaltwin.org/en/wp-content/uploads/sites/2/2023/01/IDTA-02005-1-0_Submodel_ProvisionOfSimulationModels.pdf

[14] ‘Functional Mock-up Interface’. Accessed: Jan. 05, 2026. [Online]. Available: https://fmi-standard.org/

[15] IDTA, ‘AAS-SM - Generic Frame for Technical Data for Industrial Equipment in Manufacturing’. Frankfurt am Main, Germany, Mar. 2025. [Online]. Available: https://industrialdigitaltwin.org/en/wp-content/uploads/sites/2/2025/03/IDTA-02003_Generic-Frame-for-Technical-Data.pdf

[16] ECLASS, ‘IRDI’. Oct. 10, 2025. [Online]. Available: https://eclass.eu/support/technical-specification/structure-and-elements/irdi

[17] ‘Basic system Industry 4.0 | FLUIDON’. Accessed: Jan. 05, 2026. [Online]. Available: https://fluidon.com/knowledge-base/research-projects/basys4fluidsim

[18] ‘Fluid 4.0 – Smart Circular Economy for the Fluid Industry’, Supply Chain Services Division
of Fraunhofer IIS. Accessed: Jan. 05, 2026. [Online]. Available: https://www.scs.fraunhofer.de/de/referenzen/fluid-4-0.html

[19] IDTA, ‘AAS-SM - Handover Documentation’. Frankfurt am Main, Germany, Mar. 2023. [Online]. Available: https://industrialdigitaltwin.org/en/wp-content/uploads/sites/2/2023/03/IDTA-02004-1-2_Submodel_Handover-Documentation.pdf

[20] *influxdata/influxdb*. (Jan. 05, 2026). Rust. InfluxData. Accessed: Jan. 05, 2026. [Online]. Available: https://github.com/influxdata/influxdb

[21] IDTA, ‘AAS-SM - Time Series Data’. Frankfurt am Main, Germany, Mar. 2023. [Online]. Available: https://industrialdigitaltwin.org/en/wp-content/uploads/sites/2/2023/03/IDTA-02008-1-1_Submodel_TimeSeriesData.pdf

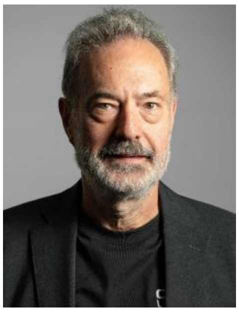
[22] ‘Nextcloud – The secure open source collaboration platform’, Nextcloud. Accessed: Jan. 05, 2026. [Online]. Available: https://nextcloud.com/de/

Biographies

Heiko Baum studied mechanical engineering at RWTH Aachen University and completed his doctorate in 2001. Since 2001, he has been the managing director of FLUIDON GmbH, where he focuses on model-based engineering, digital twins, and automated simulation workflows for hydraulic and pneumatic systems. His long-standing experience in the simulation and development of fluid power systems bridges industrial applications and research projects in the context of Industry 4.0 and the Asset Administration Shell.

Micha Koebele received his M.Sc. degree from RWTH Aachen University in 2021. Since 2024, he has been working at FLUIDON GmbH as a development engineer, focusing on dynamic system simulation, model-based engineering, and automated simulation workflows for hydraulic and pneumatic systems. His work particularly addresses the integration of simulation models with digital twin concepts and Asset Administration Shell–based data structures in the context of Industry 4.0.

Denis Ritz completed his mechanical engineering studies at TU Dresden in 2018 and is currently a research assistant at the Chair of Fluid Mechatronic Systems at TU Dresden. His research at centers on control strategies and the development of mobile hydraulic drive systems, with a strong emphasis on systems engineering. Prior to his academic studies, he completed vocational training as a farmer, providing him with practical insights into mobile machinery.



Jürgen Weber studied mechanical engineering at TU Dresden and successfully completed his doctorate in 1991. Until 1997, he was the active senior engineer at the former chair of Hydraulics and Pneumatics. This was followed by an approximately 13-year industrial phase. He was active in various positions in the R&D department of the agricultural and construction machinery manufacturer CNH. Effective March 1, 2010, Dr.-Ing. Jürgen Weber was appointed university professor and chair of Fluid-Mechatronic Systems (Fluidtronics) at TU Dresden. Since July 2018, he has been the head of the Institute of Mechatronic Engineering. He is also one of the founders and managing director of Construction Future Lab gGmbH.

Heiko Röcher is the managing director of Röcher GmbH & Co. KG, a family-owned company specializing in hydraulic presses and custom forming machinery. He combines engineering expertise with entrepreneurial leadership and is responsible for the strategic and operational development of the company, focusing on high-performance hydraulic press systems and customer-specific machine solutions.