





Infrastructure and tool support for MDE in the petrochemical industry automation

Carlos E. Xavier Correia   [Federal University of Santa Catarina | carloscorreia0002@gmail.com]

Leandro Buss Becker  [Federal University of Santa Catarina | leandro.becker@ufsc.br]

Fabio Paulo Basso  [Federal University of Pampa | fabio basso@unipampa.edu.br]

 Department of Automation and Systems Engineering, Federal University of Santa Catarina, R. Eng. Agrônomo Andrei Cristian Ferreira, s/n, Trindade, Florianópolis, SC, 88040-900, Brazil.

Received: 30 May 2024 • Accepted: 10 February 2025 • Published: 10 July 2025

Abstract Automation is essential for productivity and safety in the oil industry, but designing control systems often requires multiple software tools, leading to redundant plant modeling (costing time and money) and creating possible inconsistencies. The Model-Driven Engineering for Petrochemical Industry Automation (M4PIA) approach streamlines automation design by enabling the interoperability of models between different tools. Currently, M4PIA integrates EMSO for process simulations and MPA for deploying real plant applications. Besides, it supports high-level, graphical automation models design, also providing a component library. This paper aims to detail M4PIA, positioning it within the state-of-the-art, and illustrate its application through the deployment of an oil and gas automation system. This case study highlights M4PIA's ability to handle complex, real-world systems, demonstrating the platform's capability to optimize modeling stages through domain-specific languages (DSLs) and automated transformations. The results show that M4PIA not only reduces development time but also enhances system reliability and maintainability. By bridging simulation and deployment tools, M4PIA establishes a solid foundation for efficient and robust application development in the oil and gas sector. This platform represents a significant step forward in advancing model-driven engineering for industrial automation.

Keywords: Model-Driven Engineering, Graphical Domain-Specific Language, M4PIA, MPA, EMSO

1 Introduction

It has long been recognized that automation systems are a crucial element in modern production environments. As pointed out in Frohm *et al.* [2006], many companies have invested in automation modernization as a differentiator to remain competitive in the market. They have sought to eliminate their productivity deficiencies by improving process automation, with the aim of reducing production costs, increasing efficiency, and expanding production.

This is also true within the oil and gas industry. All processes are closely tied to automation systems, which are essential for maintaining high production levels, quality, and process safety [Klatt and Marquardt, 2009]. By working to keep the system operating at its optimal point, these control systems not only provide the aforementioned benefits but also enable the anticipation of issues that could lead to process shutdowns, potentially causing significant financial losses [Seborg *et al.*, 2010].

Overall, automating a production plant is of high complexity. Supervision, control, and sensing & actuation are the minimal (sub)systems that must be developed to support real-time operation. Besides, production control is also needed to close the automation loop. It happens that no existing software tool is capable of performing all these tasks. Instead, several different tools are available on the market to perform these tasks, whether through simulation, process supervision, or even deployment in real environments. Therefore, given that multiple software tools are required to automate an industrial plant, there is a need to model the target system (plant)

at least once within each adopted tool [Gesser *et al.*, 2022].

In the petrochemical industry specifically, a wide range of processes, components, and software tools are used [Devold, 2008]. As there is rarely interoperability between these software applications, it becomes necessary to create new models for each tool that represents the plant. Consequently, this not only leads to wasted time (and money) to design the models, but also results in high operational costs to maintain the consistency among such models. Thus, it becomes evident that the oil and gas industry faces significant challenges in modeling its processes.

To mitigate the aforementioned challenges, we observe that model-driven approaches, such as Model-Driven Engineering (MDE), are not only gaining popularity, but also proving to be highly suitable for the development of industrial applications [Vepsäläinen *et al.*, 2010]. By adopting such approaches, it becomes possible to address inherent deficiencies in the industry and incorporate or enhance important characteristics of this sector.

These facts contextualize the present paper, which is an extended version from Correia *et al.* [2023]. In essence, the paper outlines the Model-Driven Engineering for Petrochemical Industry Automation (M4PIA) approach as a possible solution to overcome the previously mentioned models' interoperability problems. M4PIA's main ideas and its base implementation, including the metamodels and model transformation routines, were presented in Damo *et al.* [2019]. Afterwards, additional features were incorporated, as follows. In Silva Cruz *et al.* [2020], a reverse engineering strategy was created to allow reusing existing source code. Finally,

in Basso *et al.* [2023], a Domain-Specific Language (DSL) was proposed with the intention of simplifying the modeling process using M4PIA through a high-level graphical representation.

As far as we know, there is no other related work that promotes the interoperability among heterogeneous tools, specially within the petrochemical domain. Besides, no other tool showed to be capable of handling bidirectional model transformations within heterogeneous tools.

This paper also presents a real-life application of the M4PIA approach, applying it to the gas compression plant presented in Gesser *et al.* [2022]. This plant represents a common (sub)system present in the oil and gas industry. The developed elements highlight how M4PIA can be leveraged to address industry-specific challenges and optimize system design and implementation.

The remainder parts of the paper are structured as follows: Section 2 presents the technological background that forms the basis of the present work. Section 3 explores the existing research related to our work. Section 4 clarifies the development of the M4PIA platform. Section 5 reports on the procedures and results of the proposed DSL. Section 6 presents a novel application in the context of the gas and oil industry. Finally, Section 7 presents the conclusions drawn from the study, as well as future perspectives for the work.

2 Technical Background

This section is dedicated to revisiting the primary technological concepts addressed in the current study. Additionally, it provides an overview of the two software tools utilized by our petrochemical collaborators and targeted in our transformation process.

2.1 Model-Driven Engineering

According to Selic [2003], Model-Driven Engineering (MDE) is a software engineering transformative approach where the primary focus is on creating and refining models rather than writing code directly. Similar to traditional engineering disciplines that rely on models to design complex systems, MDE uses abstraction to better understand problems and solutions. By emphasizing platform-independent models, MDE reduces dependency on specific implementation technologies, making systems easier to design, understand, and evolve. MDE's potential lies in breaking free from low-level coding constraints and empowering domain experts to directly contribute to system creation, potentially overcoming the industry's inertia caused by decades of investment in legacy technologies.

The application of MDE principles allows the development process to focus solely on achieving the system's objectives, such as automation, without requiring a deep understanding of the complexities of a program's execution [Schmidt, 2006]. This approach reduces the risk of errors, boosts productivity, and promotes artifacts reusability [Somerville, 2011].

Based on MDE principles, models at different levels of abstraction are connected through transformation processes

[Brambilla *et al.*, 2017]. During Model-to-Model (M2M) transformations, it is feasible to employ both direct and reverse engineering techniques, enabling transitions between higher and lower abstraction levels in either direction. Direct engineering, specific to Model-to-Text (M2T) transformations, results in the generation of platform-specific source code. In contrast, the Text-to-Model (T2M) transformation, derived from reverse engineering methodologies, produces models based on the source code of a particular platform.

2.2 Model-Driven Reverse Engineering

Reverse Engineering (RE) is an important methodology within software analysis, employed to unravel the intricacies of software systems, facilitating subsequent tasks such as maintenance, documentation, and reengineering [Rugaber and Stirewalt, 2004]. Its enduring significance is evidenced by its widespread adoption, especially aiding maintenance teams in unraveling the architectural complexities inherent in software systems.

Model-Driven Reverse Engineering (MDRE) applies model-driven methods to address reverse engineering challenges. This approach involves creating descriptive models that capture the behaviors and characteristics of legacy systems [Favre, 2005]. According to Raibulet *et al.* [2017], the MDRE process consists of two key stages: (i) converting legacy software into models with minimal loss of information, and (ii) using these models to generate the desired output models through transformation processes.

2.3 Domain-Specific Language

A Domain-Specific Language (DSL) denotes a linguistically structured system tailored to address a particular application domain, contrasting with general-purpose languages [Mailund, 2018]. DSLs offer a precise mechanism for executing tasks and attaining specific objectives within a defined scope. As noted by van Deursen *et al.* [2000], the utilization of DSLs contributes to enhanced productivity, reliability, user-friendliness, and adaptability.

Another key benefit of DSLs is improving communication among various project stakeholders, making it easier for everyone to understand the software [Otto, 2017]. This improved understanding fosters teamwork and collaboration, ultimately increasing the efficiency of the system development lifecycle.

2.4 Targeted Software Tools

2.4.1 MPA

The Automated Processing Module (MPA) tool [Satuf *et al.*, 2009] was developed by Tecgraf/PUC-Rio upon request and in collaboration with Petrobras. It supports the development and execution of automation systems in oil platforms. MPA adopts a graphical language based on flowcharts to define procedures for monitoring, diagnostics, and action within the plants. Initially developed to assist in the commissioning of oil extraction platforms at the company's research center (Cenpes) [Guisasola and Maia, 2009], it is now used in

various sectors such as level control in vessels and anti-roll protection.

MPA consists of an execution server and a configuration and supervision application. In the application, industrial plants are modeled using objects configured in the LUA programming language [PUCRJ, 1993], and the diagrams to be executed are created. The server is responsible for executing the diagrams and communicating with the supervisory system through a specific bridge. In this way, MPA can not only monitor but also act upon the different variables that constitute the plant under control.

2.4.2 EMSO

The Environment for Modeling, Simulation, and Operation (EMSO) [Soares and Secchi, 2003] is a simulation tool based on algebraic-differential equations. It provides a comprehensive development environment where users can describe equipment, model processes, perform optimizations, simulate, and visualize results. Currently, it is maintained by an alliance of national universities and petrochemical companies.

EMSO contains three main elements: models, devices, and flowcharts. Models are used to describe devices mathematically. Devices are instances of the models, used to represent the actual equipment. The flowchart pertains to the process being analyzed, depicting the interaction between the various devices that compose a given plant.

3 Related Works

Our approach is a novel contribution in applying MDE to the petrochemical industry; therefore, we were unable to find works that can be directly compared to it. This section is divided into two parts: the first presents a broad view of MDE usage in industry, while the second presents works that inspired our DSL and that could eventually become another tool targeted by our transformation process.

3.1 MDE in Industry

AUKOTON Hästbacka *et al.* [2011] is an interesting tool that supports the development of industrial control applications based on the UML Automation Profile. It includes concepts to represent the requirements, functionality, and structure of control applications, including requirements from stakeholders, alarm events, and control algorithms such as PID, fuzzy logic, and MPC. MindCSP Vidal *et al.* [2015] is a related solution that focuses on specifying the autonomic behavior of cyber-physical systems through sensor-actuator networks and autonomic control loops (monitor/analyze/plan/execute). None of them deals with model transformations.

Building Information Modeling (BIM) Wang *et al.* [2013] has introduced a significant transformation in the construction industry, with well-documented advantages in construction management. It provides interesting inspiration for metamodels construction.

Taghaddos *et al.* [2016] developed API codes that automate construction estimation. It streamlines the BIM process by automatically filtering items within specific disciplines and work areas, enabling systematic quantity take-off

across different project segments. This approach allows for precise 3D work area definitions and accurate estimations of required materials and man-hours. The study validates the automated approach by comparing its results with manual calculations and those obtained through standard BIM software interfaces, including a case study of an actual petrochemical project, demonstrating its practical applicability.

Neis *et al.* [2023] presents a model-based approach for developing applications within the Energy Management System (EMS). Despite the widespread use of models in electrical systems, EMS application development often heavily relies on these models for support, with a primary focus on source code. This approach leads to high costs and increased error risks, necessitating specialized skills in both software engineering and power systems. To address these limitations, the authors propose D-SPADES, a Model-Driven Engineering approach tailored to the EMS domain. This method employs a specific modeling language, EMSML, and tools to automatically translate models into EMS application source code. The authors demonstrated the feasibility of D-SPADES by successfully modeling and implementing two applications from the Itaipu power plant, validating their effectiveness through energy system simulations. Despite being a solid MDE approach, it does not cover the interoperability among different target tools.

In Teixeira *et al.* [2020], the authors address a problem related to the fact that many different tools are used in MDE initiatives. It shows that the significant diversity of tools, each adopting specific artifacts, leads to results that affect the interoperability of systems. In the context of MDE, such a problem poses a considerable risk to the methodology, as MDE is commonly used due to the benefits of artifact reuse, which increases productivity. To address this issue, the authors conducted research to categorize the different tools adopted in MDE. This study contributed by identifying and classifying nine types of MDE toolboxes, highlighting distinct properties compared to traditional MDE environments.

Kourouklidis *et al.* [2021] propose a model-driven engineering (MDE) approach to monitor machine learning (ML) systems, particularly for detecting and responding to performance-affecting events, like concept drift. The approach allows ML experts to define system behavior without deep technical knowledge, relying on a domain-specific language (DSL) to communicate with software engineers responsible for implementation. By leveraging MDE principles, the solution aims to simplify the design and deployment of ML monitoring systems in non-stationary environments, reducing technical barriers. Key aspects of the proposed solution include scheduling drift detection tasks, improving drift detection algorithm descriptions, expanding possible responses to drift, and enabling incremental updates to deployed systems. However, it does not deal with the interoperability among heterogeneous tools.

Lukács and Bartha [2022] introduce the FMBRSE methodology, a model-based specification and verification framework designed to support railway engineers in developing high-quality functional specifications for interlocking systems. The goal is to reduce the need for advanced computer science knowledge, making formal methods more accessible to domain engineers. By utilizing well-known tech-

niques such as model-driven development, model checking, and structured formalisms like UML, the methodology improves the quality of specifications, ensuring correctness, completeness, and consistency. It also aims to balance development costs throughout the system's lifecycle and facilitates better collaboration between engineers and end-users. The approach allows for automated code generation after formal model verification, enhancing efficiency in software design. Again, it does not deal with the interoperability among heterogeneous tools.

Wang *et al.* [2020] addresses the challenge of establishing traceability links between natural language requirements (NLRs) and AADL models in the context of model-driven development (MDD) for safety-critical systems (SCS). To bridge the gap between ambiguous NLRs and AADL models, the authors propose a requirement modeling method using a restricted natural language (RM-RNL), which reduces ambiguity while maintaining engineers' specification practices. The approach automates the generation of initial AADL models from RM-RNLs and establishes traceability links between the two. Additionally, it introduces refinement patterns to update the traceability links when requirements change or models are refined. The effectiveness of the approach is demonstrated through industrial case studies and evaluation experiments, highlighting its potential to improve MDD in safety-critical domains. The paper also evaluates the practicability of RM-RNL in terms of understandability, usability, and effectiveness based on feedback from engineers. This approach also does not tackle the need for the interoperability among heterogeneous tools.

Ponsard and Darquennes [2023] explores the challenge of selecting tools for Model-Based Systems Engineering (MBSE), focusing on the needs of both methodology and domain-specific requirements. While MBSE offers benefits like improved precision, earlier validation, and better complexity management, its successful implementation depends on choosing the right tools to support the methodology. The authors compare an industrial tool selection process with a tool selection proposal from the literature (LFMTS), applying it to a railway case study involving the modernization of a train control system. This comparison allows for the validation of the LFMTS approach, identifying gaps and complementarities. The paper also proposes guidelines to improve the tool selection process. The authors plan to extend this approach in the future to evaluate entire toolchains rather than individual tools, enhancing early assessments.

Paz *et al.* [2021] proposes CHECSDM, a systematic approach based on Model-Driven Engineering (MDE) for ensuring consistency in heterogeneous design models of safety-critical systems, which often require multiple modeling languages like UML, Simulink, and Stateflow. The approach follows an iterative three-phase process: elicitation to define requirements, mapping rules, and design guidelines; codification, where a tool framework is used to create a toolchain that checks for consistency errors; and operation, where the toolchain is applied to real system designs. Two case studies with different modeling languages demonstrated the approach's effectiveness in identifying inconsistencies with higher recall than manual verification. An assessment workshop with industrial practitioners confirmed the feasibility,

cost-effectiveness, and strong potential for the adoption of CHECSDM in real-world safety-critical system design and certification.

3.2 Relevant Petrochemical Industry Tools

The use of simulation has been prominent in analyzing and enhancing project performance indicators, capturing their dynamic and hierarchical complexities. Shafieezadeh *et al.* [2020] introduces a dynamic simulation model that explores the impact of changes in project dynamics throughout their life cycle, evaluating the effectiveness of scope and change management policies. Validated through a project in the petrochemical industry, the model highlights influential factors in a project's performance, revealing optimization opportunities in resource allocation to enhance performance indicators. The decision model can be considered a predictive and flexible tool in the construction industry, providing insights into the dynamic behavior of projects.

At the catalytic reforming unit of the Kaduna Refining & Petrochemical Company (KRPC), a model of the catalytic naphtha reforming process is adopted and simulated in Yusuf *et al.* [2019], using gPROMS, an equation-oriented modeling software. The modeling is presented in graphical, textual, tabular, and mathematical forms. The tool's metamodel is based on the Extensible Markup Language (XML) Schema. This model was used to track variations in temperature and concentrations of paraffins, naphthenes, and aromatics concerning changes in reactor heights. Increasing the temperature leads to higher production of certain compounds but reduces the yield of the final product. Conversely, increasing pressure has the opposite effect. The hydrogen-hydrocarbon ratio slightly influences the quality of the final products.

In Boukouvala *et al.* [2016], a data-oriented model was developed to optimize the planning operations of a large petrochemical complex, consisting of a petrochemical plant and two ethylene plants. Unit operation models were developed for all the processes present in the industrial superstructure, integrated with mass balance, property specification, demand, capacity, and unit selection constraints to form the overall planning problem. The authors' solution focused on the design of a tool with a methodology and technique view from the perspective of control and simulation. The developed platform is Excel-based and communicates with the planning model developed in the GAMS modeling language.

Tracking simulators operating in parallel in oil plants represent a significant evolution in the management and control of these complex environments. In Nakaya and Li [2013], a tracking simulator has been developed and is currently undergoing validation at a commercial site of a petrochemical plant. Although the Kalman filter is applied to software sensors to improve model estimation quality, precision may decrease in the face of abrupt changes in plant behavior and analyzer failures. To address these issues, Just-In-Time modeling was adopted, based on the technique with historical data from the tracking simulator, expanding the scope of the plant model across various processes.

The Routed DEVS (RDEVS) formalism provides a formalization for routing process simulation. Espertino *et al.* [2022] proposes an association between constrained network mod-

els obtained from textual specifications of routing processes and RDEVs simulation models in Java as part of developing M&S tools for the RDEVs formalism. This allows modelers to obtain simulation models without the need to encode routing implementation, offering significant benefits such as reduced implementation times and ensuring the correctness of the simulation model in line with the RDEVs formalism.

Based on the same principles of RDEVs, Blas *et al.* [2021] introduces a context-free grammar for defining routing processes as a specific case of a constrained network model. This grammar, based on a metamodel that defines the semantic elements of the syntax, enables a direct connection between its concepts and RDEVs simulation models. Additionally, a Java implementation is provided for this grammar as a plugin for the Eclipse IDE. The authors emphasize that the main benefit of this software tool is the ability to acquire a simulation model without the need for programming skills.

Table 1 characterizes each approach devoted to petrochemical tools, comparing their intents according to their purpose, modeling focus, relevance and interoperability.

4 M4PIA Infrastructure

The MDE approach under consideration in the present work is referred to as M4PIA, which stands for Model-Driven Engineering for Petrochemical Industry Automation. It was developed to support the use of MDE in applications for simulation, control, and supervision of petrochemical platforms. M4PIA was initially presented in Damo *et al.* [2019] and further detailed in Damo [2019]. Additional related works helped complement the proposal, as further detailed. Broadly, it consists of the following elements: (i) metamodels that support model development and transformation; (ii) four sets of model transformations allowing automated integration across models; (iii) two sets of code generators; and (iv) two sets of model generators.

The entire M4PIA infrastructure was constructed within the Eclipse environment, leveraging the Eclipse Modeling Framework (EMF) [Steinberg *et al.*, 2008] for metamodeling, the QVTo framework [Foundation, 2023b] to execute model transformations, the Acceleo engine [Foundation, 2023a] for code generation, and the Antlr framework [Parr, 2023] for extracting models from source code.

In MDE applications, higher-level abstraction models are preferred at the start of the modeling stages because they can more accurately and easily describe the processes. Thus, two levels of abstraction were proposed: Platform Independent Model (PIM) and Platform Specific Model (PSM). PIM models are responsible for describing the technical details of the systems, while PSM models describe the behaviors provided by the PIM models in the context of a specific software or program. Through model refinement, PIM models are transformed into PSM models via the defined transformations. Transformations in the opposite direction (from PSM to PIM) are also supported in the proposed approach.

As described in [Damo *et al.*, 2019; Damo, 2019], three different metamodels were created to support the development and usage of PIM and PSM throughout the proposed process. The PIM, also known as M4PIA, is the primary ele-

ment responsible for representing the platform-independent application. The MPA and EMSO represent PSMs that support the design of the target application. The following subsection provides more details on these metamodels.

4.1 Platform-Independent and Platform-Specific Metamodels

4.1.1 M4PIA PI-Metamodel

The M4PIA, depicted in Figure 1, is a PIM. It represents, in its highest abstract level, the structure and requirements of the process through classes, interfaces, and relationships between different elements. It is capable of capturing and generalizing aspects of both PSMs further developed. Therefore, there are similar metaclasses between it and the PSMs. Follows a brief description of this PIM.

The `Project` class stores and relates the different models present in the development. It can be composed of two different types of `File`: `ImportedFile` or `GeneratedFile`. The first type corresponds to files imported as libraries, while the second type represents groups of entities generated by the infrastructure itself.

Promoting the basic structure to more specialized classes, the `Entity` class is at the top of the modeling hierarchy. For example, the `Equipment` class is of type entity and contains attributes and methods, defined to represent the characteristics and dynamics of the equipment present in the platform.

The `Function` class is used to represent an operation and can contain a language and code. It is also associated with the `Variable` class, which can act as either a parameter or a result of the function. It can be further specialized, such as the `Method` class, which is defined to have an exclusive connection with `Equipment`.

The `Variable` class represents a logical variable that can be of type `NonTyped` or `Typed`. Typed variables can be of `EquipmentType` or `BasicType`, such as `Real`, `Integer`, or `Boolean`. The `Attribute` class is a specialization of the `Variable` class and is related to the `Access` class to define the read/write operations of objects.

Finally, the M4PIA metamodel allows recursion through the `Attribute` class. This means that it allows an equipment or function to be part of another equipment as attributes, for example, creating a hierarchical relationship between elements in the model.

4.1.2 MPA and EMSO PS-Metamodels

The MPA PSM is presented in Figure 2. In contrast to the M4PIA metamodel, which was developed targeting generic characteristics, the MPA metamodel was created to specifically represent the MPA software. The `Component` class directly corresponds to the `Entities` element in M4PIA. In compliance with MPA software, the `Variable` and `Attributes` classes are not abstract and should be associated with `Type`. This can be `BasicType`, `PointClass`, and `Equipment`, where the MPA `BasicTypes` are: `Real`, `Integer`, `Logical`, and `Textual`. Unique to MPA, codes are not only composed of `Function` or `Class` but also

Table 1. Comparison of relevant petrochemical industry tools.

Approach	Main Purpose	Modeling Focus	Industry Focus	Interoperability Focus
M4PIA	Interoperability and automation design via MDE	Graphical DSL, model transformations	Oil and gas automation, case-based validation	High - bridges multiple heterogeneous tools
Blas <i>et al.</i> [2021]	Simulation of routing processes via DEVS	Metamodel + constrained network grammar	Simulation of routing in constrained networks	Medium – models based on shared grammar
Esperino <i>et al.</i> [2022]	Mapping text-based routing specs to RDEVS models	Textual DSL, RDEVS Java simulation	Automated Java model generation for RDEVS	Medium – consistent transformation rules
Nakaya and Li [2013]	Real-time plant behavior tracking & prediction	Hybrid physical/statistical Just-In-Time modeling	Live petrochemical plant tracking & optimization	Low – primarily internal hybridization
Boukouvala <i>et al.</i> [2016]	Global optimization of petrochemical planning	Data-driven MINLP models with fitted parameters	Enterprise-level operational planning	Low – integrated in a single platform
Yusuf <i>et al.</i> [2019]	Catalytic reforming modeling and simulation	Equation-oriented modeling using gPROMS	Naphtha catalytic reforming optimization	Low – standalone simulation focus

of Code, used to import DLL, which will be stored in `DLLFunction` and `DLLParameter`.

The second implemented PSM is the EMSO metamodel, depicted in Figure 3. The class `Model` is equivalent to `Entities` in M4PIA and is responsible for instantiating the mathematical models of equipment or `Device`, recalling that the EMSO platform is focused on simulation and optimization. The `Equation` class relates functions to their respective models. `Parameter`, like `Variable`, is a specialization of `Attributes` and represents, respectively, the constant and variable attributes of the model. As a mathematical software, `Variable` cannot be textual, while `Parameter` can be, through the `Switcher` class. The classes `Flowsheet`, `Estimation`, `Optimization`, `Device`, and `Connection` are represented but not specified in the metamodel, as the infrastructure's intention is exclusively for modeling.

4.2 Model Transformations

Initially, in Damo *et al.* [2019] it was developed support for direct engineering transformations, starting with a M4PIA PIM model and transforming it into PSM models (MPA or EMSO), and ultimately into source code. To further enhance the tool, in [Silva Cruz *et al.*, 2020; Cruz, 2021] it was introduced a set of transformations necessary for promoting reverse engineering, which generates the PSM model from the source code and the M4PIA PIM model from the PSM. This addition completed the tool by supporting both forward and reverse engineering processes, significantly enhancing its capabilities and versatility.

Currently, M4PIA supports eight different transformations, including: (i) four M2M transformations, (ii) two M2T transformations, and (iii) two T2M transformations. The transformations let available in the URL presented in the Appendix B. These transformations allow the generation of the EMSO software source code starting from the MPA software source code. The process involves first transforming the MPA source code into the MPA PSM model, then into the M4PIA PIM model, followed by the EMSO PSM model,

and finally to the `.emso` extension file. The procedure is illustrated in Figure 4.

Through the developed transformations, the M4PIA infrastructure facilitated the reuse of projects created in either MPA or EMSO. In other words, it allowed for the automatic conversion of a project from MPA to EMSO, or vice versa. This enabled interoperability between two widely-used software platforms within Petrobras (Brazilian oil company). As a result, a project that would have previously started from scratch now has a much more advanced starting point, significantly optimizing the overall design process.

5 Graphical DSL

The graphical DSL adopted in this work was proposed in [de Souza Moura, 2022; Basso *et al.*, 2023]. It includes two distinct levels of abstraction, a decision influenced by feedback from petrochemical industry experts who emphasized the need for different visualization modes of the model: one for development and another for maintenance and contextualization. As a result, two types of diagrams were developed: the Deployment Diagram (DD) and the Conceptual Diagram (CD).

The entire DSL structure was implemented using the Sirius framework [Foundation, 2023c], integrated within the Eclipse environment.

5.1 Conceptual Diagram

The Conceptual Diagram was designed to provide users with contextual insights into the model process. It supports model maintenance by offering transparent information, including the attributes and methods used. The Conceptual Diagram helps visualize the process clearly, illustrating how equipment is interconnected. This makes it easier to understand not only individual pieces of equipment but also the entire process. Figure 5 depicts a system modeled from the Conceptual Diagram perspective.

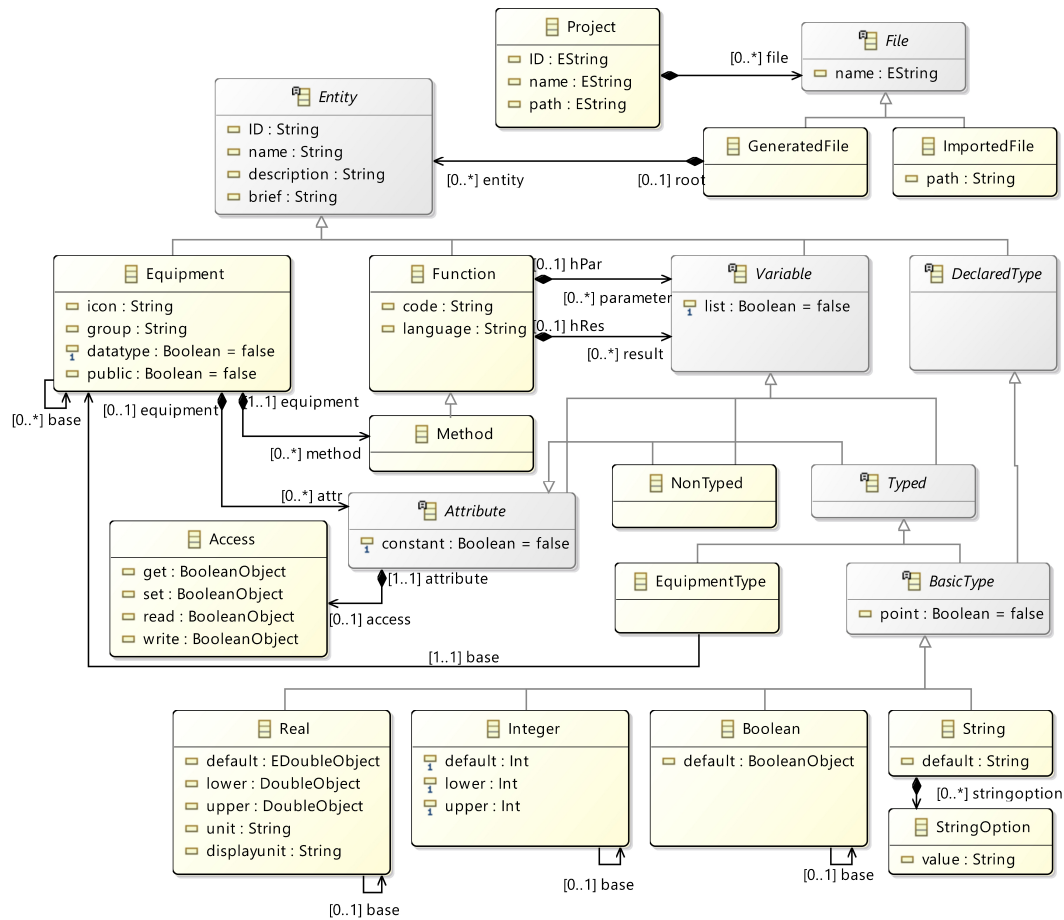


Figure 1. M4PIA Platform-Independent Metamodel (PIM) [Damo *et al.* [2019]]

In the Conceptual Diagram, equipment is represented using a parent container that houses three child containers arranged in a vertical stack. These child containers represent the compartments for basic attributes, equipment attributes, and methods. Each compartment contains child nodes displayed in lists, where specific properties are defined. These properties include the domain class and the expressions of fundamental semantic candidates to ensure accurate selection of the elements.

5.2 Deployment Diagram

The Deployment Diagram was designed specifically to implement the plant model. It streamlines the modeling process by providing a predefined structure for creating new objects in the M4PIA metamodel. The Deployment Diagram focuses solely on the components and their relationships, omitting other details to allow the developer to concentrate fully on process implementation. Figure 6 shows a model viewed from the perspective of the Deployment Diagram.

In the Deployment Diagram, equipment types are represented by a **node** using a default 3D cube format with spatial notation. A child artifact is represented as a *puzzle*-shaped border node to indicate composition by other equipment, which is displayed only when the equipment is composed of others. Additionally, the display characteristics of equip-

ment can be modified based on the textual attribute *icon* of the candidates themselves, which are obtained by applying a filter based on equipment type attributes using semantic candidates.

The relationships in both diagrams are represented by edges. However, the two diagrams differ in the style of the arrowhead decorator. The Deployment Diagram does not use a decorator for its edges, whereas the Conceptual Diagram employs a white triangular arrowhead to represent inheritance and a black diamond arrowhead to indicate composition, following UML class diagram notation.

6 Context and Example Application

In this section, we present the design of a classical application from the petrochemical industry: a gas compression plant, presented by Gesser *et al.* [2022]. It was modeled following MDE’s principles and using the tools described along the paper. We perform a classic case of conventional modeling, which involves direct engineering from an industrial plant.

6.1 Process Definition and Operation

Designed for oil and natural gas production, offshore platforms are units composed of a three-phase separator, an oil

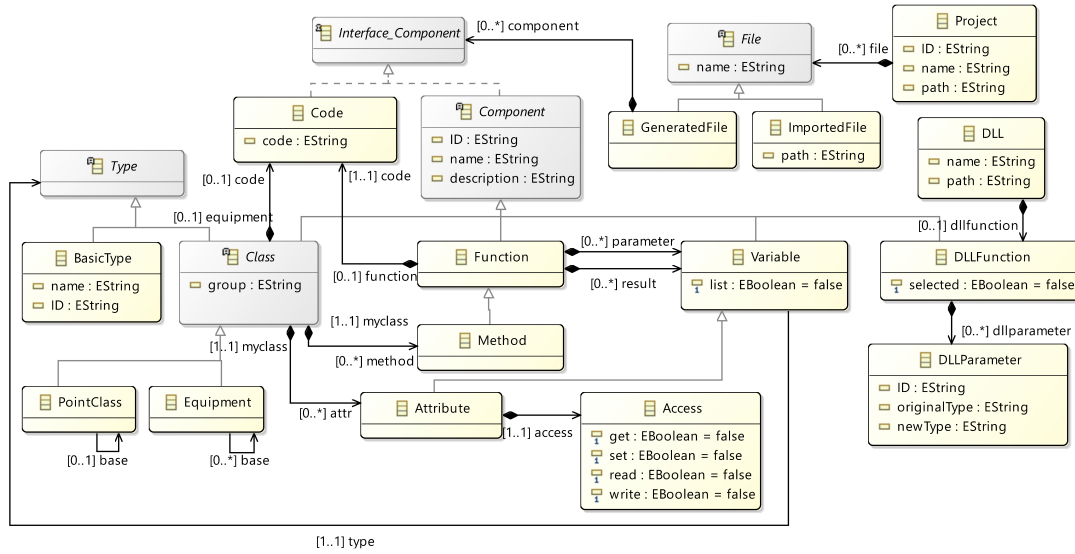


Figure 2. MPA Platform-Specific Metamodel (PSM) [Damo et al. [2019]]

and water treatment system, and a gas compression system. The separator is responsible for dividing the production from the wells into oil, water, and natural gas. The water produced must undergo treatment before being disposed of into the sea again, as it contains oil and other residues that can degrade the environment Liu *et al.* [2021]. Lastly, the gas compression system (GCS) plays a vital role in elevating gas pressure from the separator, ensuring a controlled gas flow at specific pressure, humidity, and temperature, by the desired operating point and the requirements of the following systems [Mokhatab *et al.*, 2015].

Typically composed of heat exchangers, tanks, and distribution pipes, the GCS unit has, as its main component, the centrifugal compressor responsible for increasing the gas pressure [Mokhatab *et al.*, 2015]. To maintain the desired conditions of pressure, temperature, and flow, and to reject disturbances arising from variations in inputs and gas properties, various local controllers are applied. In this sense, due to the number of equipment, each with its associated controller, and the nonlinear characteristics inherent in the process, the GCS is considered a complex multivariable system [Plucenio *et al.*, 2016].

The compressed gas produced by the GCS serves one or more purposes. The main one among them is to meet market demand, with the produced gas being directly exported through pipelines. Another significant application is in the injection of production wells to increase productivity [Sheng, 2015]. Finally, the compressed gas is also used to supply generators responsible for providing energy to the GCS.

As already mentioned, the simplified scheme of the GCS adopted in this work follows the model presented in Gesser *et al.* [2022] and is depicted in Figure 7. It comprises four compression units: the main, the exportation, the CO₂, and the injection units. The Main compressor is primarily responsible for increasing the pressure of gases coming directly from the separators. To remove the high CO₂ content present in the gases from the separators, immediately after the Main compressor, the compressed gas passes through a CO₂ sep-

aration membrane, where the natural gas is directed to the Export unit and the CO₂ to the CO₂ compression unit.

The exportation compression unit is responsible for increasing the pressure of the natural gas to be produced, which is then immediately directed to the processing unit, with the flow controlled by an output valve. In the CO₂ compression unit, the gas used to feed the injection unit has its pressure raised. Finally, the injection compression unit, powered by the CO₂ unit and the Export unit, depending on the production needs, is responsible for increasing the pressure of the gas to be used in the gas-lift operation in production wells.

To control the pressure in the GCS, flare valves are added to the inlet of each unit. This ensures a quick and effective response to mitigate not only local issues but also general concerns regarding excess pressure in the system.

Due to the complexity of the presented GCS, we will focus only on the Main compression system in the scope of this work. Also taken from Gesser *et al.* [2022], the scheme of the Main unit can be viewed in Figure 8. This system is composed of two parallel compressors, ensuring efficiency through load sharing even with only one compression stage [Xenos *et al.*, 2015].

The gas entering the compression unit initially passes through a Drum, responsible for removing water droplets present in the gas. Next, in the Header, control of the maximum pressure is performed with the aid of the Flare valve controlled by the local PID. After these processes, the gas is directed to the compressors with flow controlled by the outlet valves.

As the parallel systems are identical, here is an explanation of just one of their levels. A Scrubber is used at the inlet to remove any liquids and solids that may have formed in the gas during transport through the pipeline. The outlet of the Scrubber is connected to the compressor's inlet, driven by a turbine that is controlled by a PID to maintain the pressure at or near the set point. Afterward, the gas passes through a cooler to reduce its temperature, as the increase is inherent to the compression process [Bonjour and Bejan, 2006]. Part

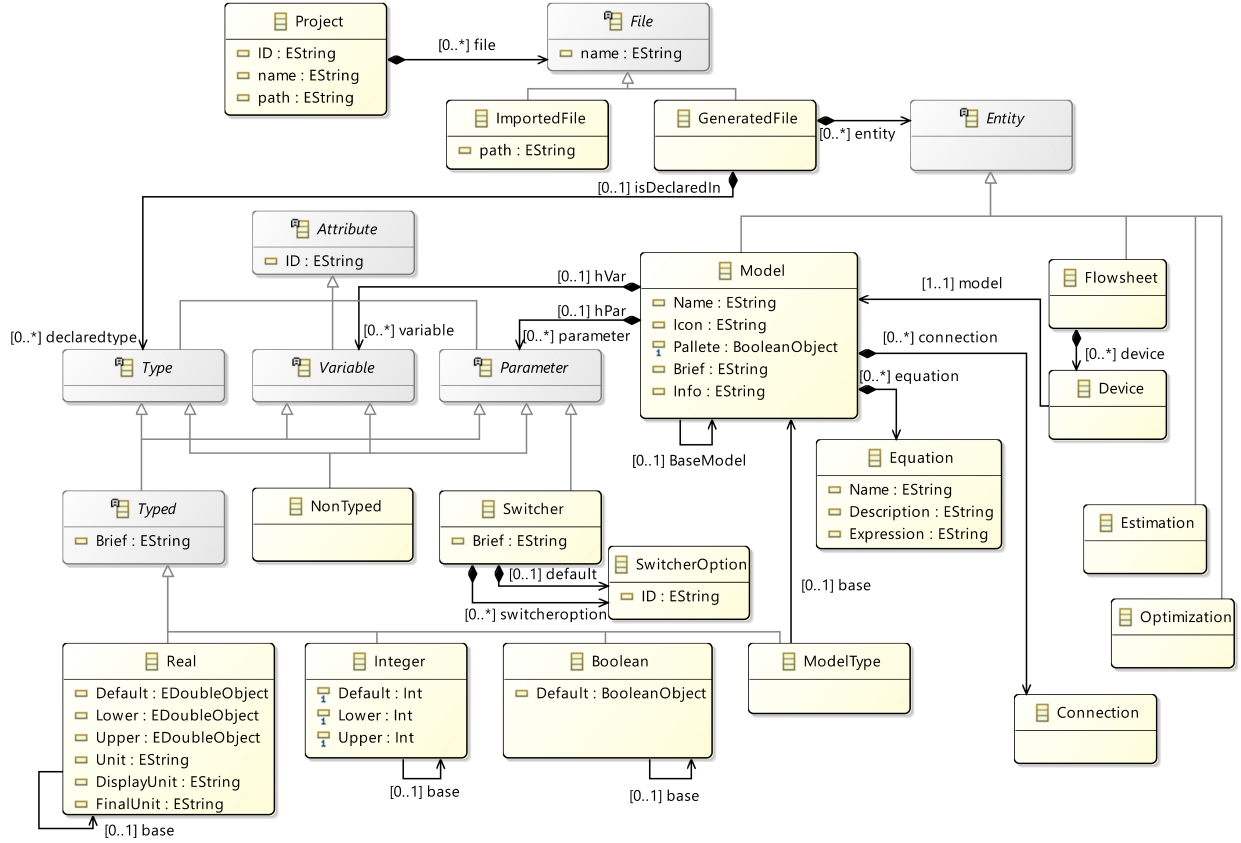


Figure 3. EMSO Platform-Specific Metamodel (PSM) [Damo et al. [2019]]

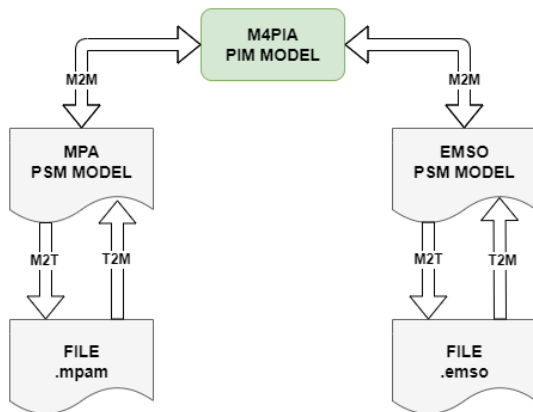


Figure 4. Set of supported transformations.

of the gas is recirculated through the Surge valve, depending on the compressor's operating conditions, to prevent surge problems that could lead to unit shutdown [Gravdahl et al., 2000]. Finally, the gas passes through another Scrubber for further filtration, ensuring an extremely clean gas at the Main compression plant outlet.

6.2 Process Modeling

As illustrates the graph in Figure 9, this section presents a feature from M4PIA that allowed us to reach level 6 of Technology Readiness Level (TRL). Maturity levels are discussed further in the section devoted to analysis of the application dimension. In the following, a conceptual demonstration is presented.

The modeling of the Main compressor, presented in Figure 8, was conducted using information extracted from the EMSO's built-in libraries and the MPA libraries provided by [PUC-Rio]. They provided details about the equipment to be modeled, including attributes, methods, and the relationships between them. The outcome model can be observed in both the Conceptual and Deployment diagrams, as depicted in Figures 10 and 11, respectively. Next, the steps employed to achieve these results are outlined.

The first modeling step targeted creating the M4PIA model, and involved using the Deployment Diagram provided by the DSL. In this stage, the focus was on creating the equipment and their relationships, aiming to accurately represent the so-called *Main* compressor. In a general sense, each piece of equipment in the Main compressor scheme becomes an *Equipment* component of the M4PIA metamodel. Thus, by the end of this stage, the model did include valves, PID controllers, compressors, and other minor components.

In contrast to the other components, the modeling of the Compressor Main equipment is not straightforward and presents some peculiarities. When analyzing the aforementioned libraries, it is evident that compressors are typically associated with other protective equipment and with a specific number of stages, aiming to ensure safety and transparency in the equipment's operation. Thus, to faithfully represent reality, Compressor Main 1 becomes a set of interconnected equipment, including the Compressor Main, Compression Stage, and Compression Protector.

In the system, various composition relationships were uti-

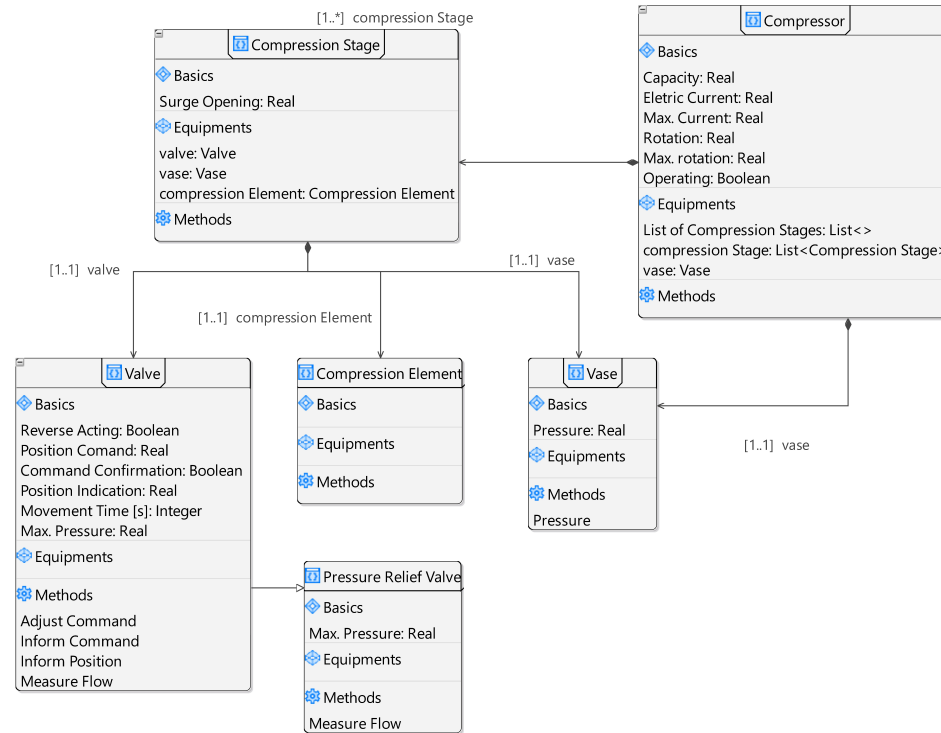


Figure 5. Conceptual modeling view.

lized to depict not only the individual operation of the equipment but also the overall functioning of the entire plant. Equipment, such as the control valve, must necessarily be composed with controllers for operation. The same principle applies to mixers and splitters, which integrate with valves to achieve the proper mixing or separation of the fluid. Finally, to represent the special modeling of the Compressor Main component, composition was utilized to incorporate the Compression Stage and Compression Protector into the main equipment.

The second and final stage of modeling was carried out by means of the Conceptual Diagram. Using the functionalities of this representation, basic attributes and methods were incorporated into the components modeled in the previous phase, obtained through consulting the literature and the documentation from the mentioned libraries. At the end of this stage, each piece of equipment contains the essential information to faithfully represent its real counterpart.

6.3 Direct Engineering

In the scope of this study, only the process of forward engineering was carried out, that is, starting from the PIM model and moving to the respective PSM models and source codes. This was done using the set of transformations provided by the M4PIA infrastructure. As follows, the steps used to achieve these results are described, just like depicted in Figure 4.

The first stage of this process involved obtaining the PSM models of the MPA and EMSO platforms. Using the previously developed M4PIA model and employing the M2M transformations provided by the infrastructure, the MPA and EMSO models were generated, representing the Main Com-

pressor system in the domains of MPA and EMSO software, respectively.

The following step consisted of transforming these PSM models into the source code of the platforms. For this purpose, M2T transformations were used, in which the MPA and EMSO models were converted into source files of type .mpam and .emso, respectively.

The last step consisted in the refinement process of the generated source codes. This is due to the fact that the transformations are still unable to generate a totally operational code, as there are artifacts that cannot be fully converted automatically. This refinement process is essential to maintain the syntax and compliance of the source code with the project. With the finalization of this stage, the source codes are ready for use, indicating the conclusion of the forward engineering process.

For those interested, the developed PIM and PSM models, as well as the source codes, are available in Appendix A.

6.4 Analysis of the Application Dimension

The Compression Main application addressed in this section aimed to extend the analysis in previous works of the applicability of the M4PIA infrastructure in the petrochemical industry domain, as well as to assess the overall functioning of the tool by identifying its strengths and bottlenecks. Furthermore, it also served as an additional proof of concept to the simplified compression system discussed in Damo [2019]; Cruz [2021].

To demonstrate the impact that the use of the proposed infrastructure can have on the development of domain applications, some dimensions are presented in Table 2. The table compares the developed Compressor Main application

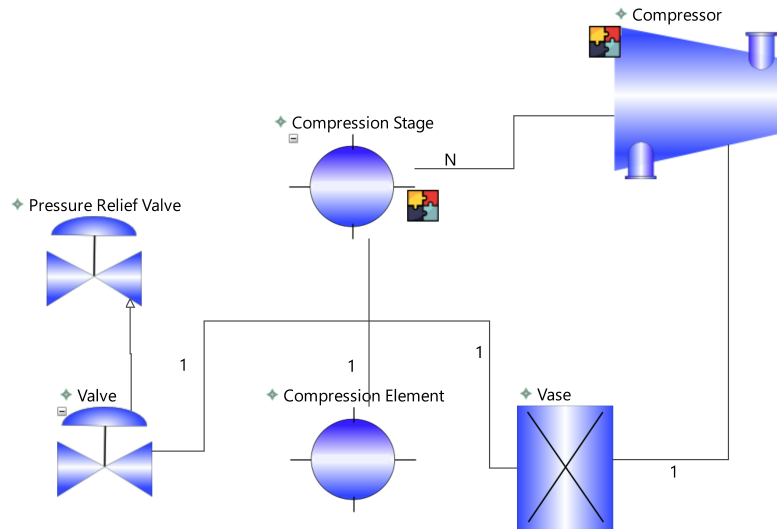


Figure 6. Deployment modeling view.

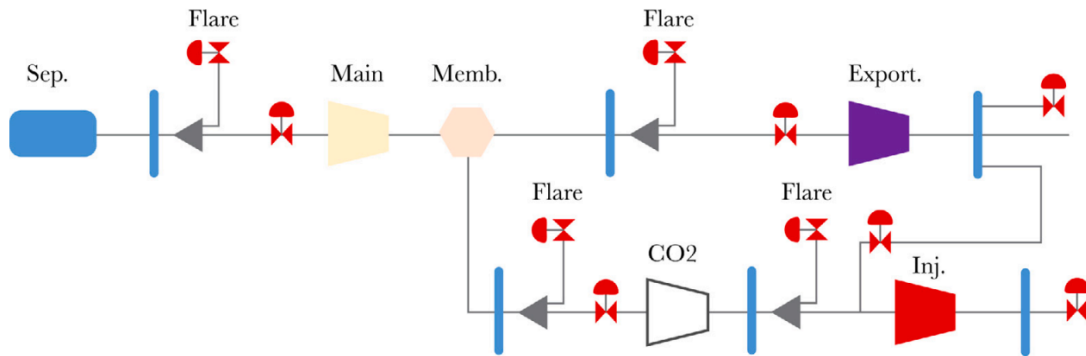


Figure 7. Schematic of the Gas-Compression System (GCS) [Gesser et al., 2022].

against a native example application from the MPA platform library. Such example application implements a system for detecting and breaking hydrates in a Floating Production, Storage, and Offloading unit (FPSO) for oil and natural gas. Hydrate formation in the pipelines that transport oil from the well to the surface causes blockages, resulting in production losses and increased overall maintenance costs Makwashi et al. [2019].

Analyzing the data from Table 2, it can be observed that the application developed in this work has a complexity level similar to the hydrate detection and breaking system. It has 30% fewer classes, but three times more attributes and twice the number of methods. However, it also has about 17 times fewer instantiated equipment.

Another important data to be analyzed is the number of lines of code (LOC) in the system, as it is a widely used quality indicator in Software Engineering evaluations. It represents the effort of development and maintenance, being associated with costs, development time, developer productivity, and code quality [Fenton and Bieman, 2014]. It is observed that the source code of the Compressor Main is 28% larger than the hydrate detection and breaking system for FPSO. In other words, we have a system that is not only larger but also more complex, not only in development but also in opera-

tional maintenance.

To evaluate the user experience when using the M4PIA, i.e. especially concerned in the context of integrated tools designed to fully support reverse and advanced engineering tasks, the User Experience and Usability Guidelines for Agile Project (UXUG-AP) technique was employed de Oliveira Sousa and Malveira Costa Valentim [2021]. According to de Oliveira Sousa and Malveira Costa Valentim [2021], this technique was developed with the aim of supporting Usability and User Experience (UX) design throughout the entire software development lifecycle. The methodology is structured into 11 categories, each comprising specific sub-categories, which in turn include guidelines that steer user-centered development.

We applied the guideline considering all the tools and requirement engineering phases conducted so far, and in special considering the controlled experiment presented by Basso et al. [2023], resulting in an analysis shown in Table 3. It can be observed that the application achieved 84% compliance with the evaluated subcategories, indicating a satisfactory level of performance regarding usability and user experience for this type of support tool.

The third conducted evaluation follows the six standard

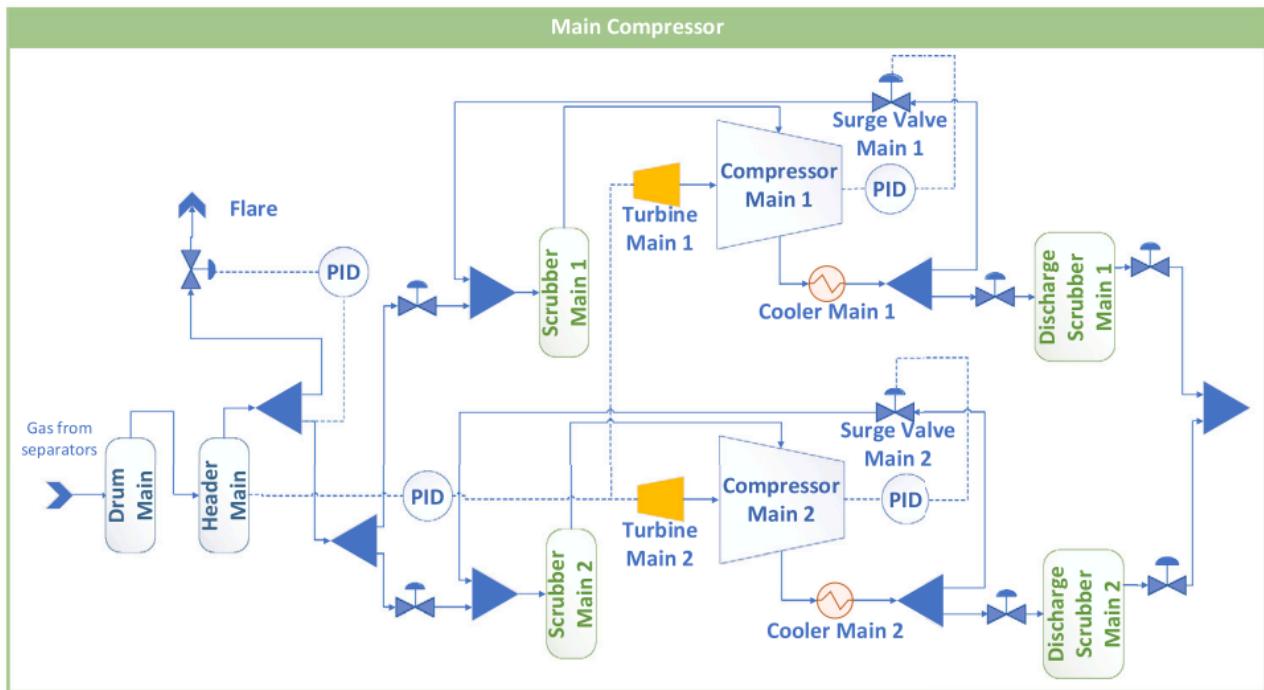


Figure 8. Schematic of the main compressor [Gesser et al., 2022].

Table 2. Comparison of domain applications.

	Main Compression System	Detection and Hydrate Break for FPSO-P31
Num. of Equipment Classes	13	17
Num. of Attributes	235	74
Num. of Methods	63	25
Lines of Code - MPA	1817	1297
Lines of Code - EMSO	897	-
Num. of Instantiated Equipment	36	621

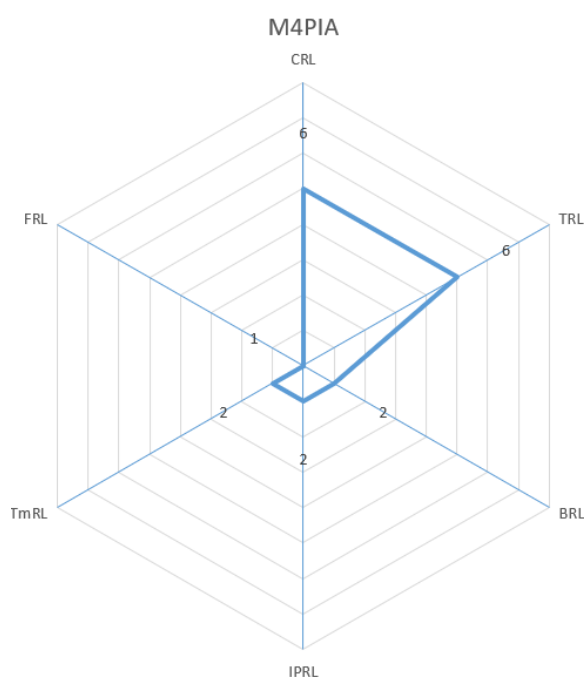


Figure 9. M4PIA metrics on KTH Innovation Readiness Level.

metrics of KTH Innovation Readiness Level¹. Figure 9 shows our current maturity for the M4PIA Infrastructure. KTH describes six areas or metrics scoping an innovation. Marking a maturity level implies that all the previous levels were once achieved by the innovators. In the following, we discuss these metrics with justification based on our experiences.

Metric 1 - Customer Readiness Level (CRL): We attributed the maturity level 6, corroborating with our findings of categories discussed in Table 3, such as Requirements (them all) Understanding User Needs (them all) and Emotional Connectivity. These categories were explored along more than five years of interaction with client needs. Our previous study was carried out with employees from Petrobrás, and now we are preparing to reach the maturity level 7 in CRL metric as follows:

- Level 1 – Hypotheses formulated about possible market needs;
- Level 2 – Specific market needs to be identified;
- Level 3 – Initial marketing feedback established;
- Level 4 – Problem/need confirmed through multiple customers or users;

¹ KTH Innovation Readiness - <<https://kthinnovationreadinesslevel.com/>>

Table 3. UXUG-AP Technique.

Categories	Subcategories	Included
Requirements	Information Exchange – Team/Client	Yes
	Interviews and Workshops	Yes
	Key Requirements	Yes
Understanding User Needs	Beginners and Experts	Yes
	Children, Youth, Adults, and Elderly	Yes
	Laypeople, Academics, and Professionals	Yes
Accessibility	Visual Impairments I	No
	Visual Impairments II	No
Ease of Use	Localization	Yes
Informative Feedback	Warning and Confirmation Messages	Yes
	Error Messages	Yes
	Loading Components	Yes
	Titles and Links	Yes
Error Prevention	Required Fields	Yes
	Field Restrictions	Yes
	Self-explanatory Presentation	Yes
Information Grouping	Information Independence	Yes
	Information Modularization	Yes
Sequence of Actions	Organization of Sequential Actions	Yes
	Behavior of Sequential Actions	Yes
Sense of Belonging	Emotional Connectivity	Yes
Degree of Importance	Information Positioning	Yes
	Relevant Terms	Yes
Privacy	Information Control	No
	Passwords	No

- Level 5 – Established customer interest and relationships;
- Level 6 – Benefits confirmed by initial customer test;
- Level 7 – Customers in extended tests or early sales stage. Small number of active users;
- Level 8 – Initial commercial sales and implemented sales process. Substantial number of active users;
- Level 9 – Widespread and scaling sales. Large number of active users with substantial growth.

Metric 2 - Technology Readiness Level (TRL): Also due to demonstration for Petrobrás employees, we reached level 6 of TRL, which categorizes a mature tool support but not yet ready for operational usage. To reach level 7 we plan the development of a methodology addressing how these models support operational safety and resilience in complex industrial control systems, thus providing real-company examples, but only for view of the company due to copyright issues, conform details the following TRL metrics:

- Level 1 – Interesting research results or initial technological idea;
- Level 2 – Technology concept and/or application formulated;
- Level 3 – Proof of concept of critical functions and/or characteristics in the lab;
- Level 4 – Technology validated in the lab;
- Level 5 – Technology validated in a relevant environment;
- Level 6 – Prototype technology demonstrated in relevant environment;

- Level 7 – Prototype technology demonstrated in operational environment;
- Level 8 – Complete technology demonstrated in real operations;
- Level 9 – Fully operational technology proven over time.

Metric 3 - Business Readiness Level (BRL): We were in Level 2 of BRL and hoped to reach Level 3 along the first six months of the Petrobrás project. The related work section is an example of Level 1 achievement, and Level 2 was reached through a funding project we wrote for Petrobrás. Due to high costs involved to expand the proposed solution for operational use, the project was rejected, for now. This implies in backward in some criteria of Level 2. BRL can be followed according to the nine levels below:

- Level 1 – Vague hypotheses of a possible business idea, market potential, and competition;
- Level 2 – Initial hypotheses of potential business concept (if any), full market and competitor identification;
- Level 3 – Description of a sustainable business model and target market, including competitors;
- Level 4 – Preliminary calculations indicate economically viable business model. Initial assessments indicate environmental and social sustainability;
- Level 5 – Key assumptions of the sustainable business model are tested in the market;
- Level 6 – Complete sustainable business model tested with customers, partners, suppliers (e.g., through sales testing), calculations demonstrate economic viability;

- Level 7 – Business model feasibility (pricing model, revenue model, etc.) validated through initial commercial sales;
- Level 8 – Sales and metrics show that a sustainable business model is viable;
- Level 9 – Sustainable business model meets internal and external expectations. In terms of profit, it presents scalability and long-term impact.

Metric 4 - Intellectual Property Rights Readiness Level (IPRL): We already discussed IPR, in special due to the Petrobrás project, and we intend to begin the registration of the software components. Next, we provided a brief description of IPRL levels:

- Level 1 – Hypotheses regarding possible IPRs;
- Level 2 – Identification of different forms of potential IPRs created. Some clarity on types of IPRs;
- Level 3 – Description of key potential IPRs in some detail. Initial assessment of protection potential;
- Level 4 – Team confirms that IPR protection is feasible with outlined purposes. Organization of IPRs by relevance;
- Level 5 – Draft IPR strategy to create business value. First forms for key technology registration are completed;
- Level 6 – Initial operational IPR strategy, considering different technologies. Positive responses in registrations and other formalizations;
- Level 7 – IPR registrations/formalizations completed for key technologies in countries/regions aligned with strategy;
- Level 8 – Full implementation of IPR strategy and management practices. Registrations and formalizations completed for all technologies;
- Level 9 – Strong IPR support and business protection. Ongoing IPR protection ensured in relevant countries.

Metric 5 - Team Readiness Level (TmRL): TmRL is a challenge for academic innovation projects. So far, we never could ensure that the same developer remains in the project. A total of four developers have contributed to M4PIA infrastructure, always as researchers, two graduate and two undergraduate students. Each one could contribute for at most two years. For this reason, we are still in Level 2, as exposed below:

- Level 1 – Lack of skills and resources to begin idea validation. Little awareness of team needs (typically an individual);
- Level 2 – Limited ongoing skills to begin idea validation. Initial awareness of the need for more skills and resources;
- Level 3 – Some ongoing skills to validate/develop the idea. Necessary competencies identified (and a plan to acquire them);
- Level 4 – A standout individual (champion) present with clear direction (startup/other format). Several dependent competencies in progress, with a complementary plan;

- Level 5 – Initial team with core competencies and capabilities. Team agrees on equity, roles, goals, and vision;
- Level 6 – Core team is committed, diverse, and has all necessary competencies to start building the business;
- Level 7 – Well-functioning team. Growth plan defined to expand the team and build the organization over time;
- Level 8 – Professional organization in progress (board, CEO, admin, staff);
- Level 9 – High-performing and well-structured organization at all levels for ongoing operations, development, and continuous improvement.

Metric 6 - Funding Readiness Level (FRL): The funding project declined by Petrobrás classified our innovation in FRL level 2. However, we are not sure if we should downgrade to Level 1 or 0, since our platform is devoted to their needs. In the following, the levels of FRL are introduced:

- Level 1 – No clear description of initial funding verification activities. No clear view of funding needs or options;
- Level 2 – Description of initial funding verification activities. Funding needs and sources identified for project milestones;
- Level 3 – Funding secured for an initial business verification plan;
- Level 4 – Funding secured for a more elaborate business verification plan;
- Level 5 – First funding pitch tested with a relevant audience. Short-term funding strategy defined;
- Level 6 – Improved funding pitch tested with a relevant audience. Initial contact with relevant funding sources;
- Level 7 – Initial discussions with potential external funding sources. Full pitch presented with ready-to-share materials;
- Level 8 – Detailed spreadsheet-level discussions with one or more external funding sources showing clear interest;
- Level 9 – Secured finances available for at least 6–12 months of operations. Financial monitoring and forecasting system fully implemented.

Through these discussions, we can conclude that the M4PIA infrastructure has demonstrated its capability to support a system with a complexity level comparable to a real application in the oil and gas sector. The streamlined and optimized modeling stage through the DSL and its transformations highlights the relevance of the proposed approach to meet a specific industrial need. Although we still have a long road to reach operational contexts, these outcomes lay a solid foundation for innovations in the oil industry.

7 Conclusions and Future Works

The purpose of this work was to review the papers Damo [2019]; Silva Cruz *et al.* [2020]; Basso *et al.* [2023], covering everything from the basic concepts to the development of the proposed solutions. It provided an opportunity not only to validate the feasibility of the proposed MDE and DSL approaches but also to identify deficiencies and issues that need to be further addressed.

To evaluate the proposed solutions in Damo [2019]; Cruz [2021], a comparative analysis of performance was conducted between the process utilizing the tool and the process that did not employ it. In de Souza Moura [2022], a quasi-experiment was carried out involving experienced professionals from the oil and gas sector. Based on the development of this work and the discussed evaluation sections, it is evident that the results obtained are satisfactory and that the developed tool is well received by stakeholders. The M4PIA infrastructure enabled interoperability between the MPA and EMSO software, optimizing plant modeling in the petroleum industry. The DSL, in turn, considerably facilitated the use of the M4PIA tool.

To test the M4PIA infrastructure and assess its applicability in a real oil and gas plant, as well as to evaluate its strengths and weaknesses, a modeling study was conducted on a two-stage parallel compression system. This study demonstrated that the infrastructure is not only capable of handling real, complex systems in the field but also scalable to accommodate larger applications. This confirms the initial assumption that the tool is capable of promoting optimizations for modeling stages in the petroleum industry.

However, some issues were noted during the application modeling phase. Concerning the M4PIA tool, there is a lack of integration between the developments of Damo *et al.* [2019] and Silva Cruz *et al.* [2020] since they were conceived in separate development environments. As a result, although the tool exists, it cannot be fully utilized due to the lack of integration. Regarding the DSL, although it facilitates the modeling of M4PIA models, there is still a need to model each piece of equipment from scratch for every new project. Considering that the oil industry uses a set of widely utilized equipment, something could be done to mitigate this issue and further streamline the modeling process.

The fact that M4PIA only supports two different platforms (or target tools) is indeed a limitation. However, it was conceived in a way that allows the creation of new PSMs and the related model transformations in a straightforward manner. Therefore, if concrete needs are specified and proper human resources are allocated, M4PIA's target tool set can get much broader.

For future work, solutions to address the aforementioned issues are proposed. For integration, the use of the same development environment with the help of plugins to ensure the proper functioning of the M4PIA infrastructure is suggested. Additionally, for the DSL, the creation of a library within the DSL itself, containing the most commonly used equipment in the oil industry, is recommended. This would not only make the tool more usable but also enhance its capabilities, making it even more powerful and efficient. Finally, another topic to be addressed regards maintaining the transformations throughout the system development lifecycle.

Declarations

Acknowledgements

The authors would like to thank the participation and cooperation of the original authors of the works on which this article was based.

Funding

This research was funded by CAPES and CNPq. Authors at UFSC are also supported by the PRH-ANP/MCT n° 2.1 program.

Authors' Contributions

LBB and FB contributed to the conception and supervision of this study. CEXC is the main contributor and writer of this manuscript. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

The material is available from the corresponding author on a reasonable request.

References

- Basso, F., de Souza Moura, J., Correia, C. E. X., Casola, K., and Becker, L. B. (2023). Graphical dsl devoted to ease the application of model-driven engineering in petrochemical industry automation. In *Prof. of Workshop on Requirements Engineering*, Porto Alegre, RS, Brasil. SBC. DOI: 10.29327/1298356.26-19.
- Blas, M., Espertino, C., and Gonnet, S. (2021). Modeling routing processes through network theory: A grammar to define rdevs simulation models. In *Anais do III Workshop em Modelagem e Simulação de Sistemas Intensivos em Software*, pages 10–19, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/mssis.2021.17255.
- Bonjour, J. and Bejan, A. (2006). Optimal distribution of cooling during gas compression. *Energy*, 31(4):409–424. DOI: 10.1016/j.energy.2005.04.004.
- Boukouvala, F., Li, J., Xiao, X., and Floudas, C. A. (2016). Data-driven modeling and global optimization of industrial-scale petrochemical planning operations. In *2016 American Control Conference (ACC)*, pages 3340–3345. DOI: 10.1109/ACC.2016.7525433.
- Brambilla, M., Cabot, J., and Wimmer, M. (2017). *Model-Driven Software Engineering in Practice*. Springer. DOI: 10.1007/978-3-031-02546-4.
- Correia, C., Becker, L., and Basso, F. (2023). A review on the infrastructure and tool support for model-driven engineering in the automation of the petrochemical industry. In *Anais do V Workshop em Modelagem e Simulação de Sistemas Intensivos em Software*, pages 31–40, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/mssis.2023.235679.
- Cruz, M. V. S. (2021). Engenharia reversa baseada em modelos para aplicações de simulação, controle e operação de plantas na indústria petroquímica. Master's thesis, Federal University of Santa Catarina (in Portuguese). Available at: <https://repositorio.ufsc.br/handle/123456789/231059>.
- Damo, T. P. (2019). Engenharia baseada em modelos para aplicações de simulação, controle e operação de plantas na indústria petroquímica. Master's

- thesis, Federal University of Santa Catarina (in Portuguese). Available at: <https://repositorio.ufsc.br/handle/123456789/206409>.
- Damo, T. P., Becker, L. B., and Basso, F. P. (2019). Model-Driven Engineering Infrastructure and Tool Support for Petrochemical Industry Automation. *Advances in Science, Technology and Engineering Systems Journal*, 4(4):174–187. DOI: 10.25046/aj040422.
- de Oliveira Sousa, A. and Malveira Costa Valentim, N. (2021). Designing usability and ux with uxug-ap: An observational study and an interview with experts. In *Proceedings of the XVII Brazilian Symposium on Information Systems*, SBSI '21, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/3466933.3466959.
- de Souza Moura, J. (2022). Uma dsl gráfica de suporte para a infraestrutura m4pia. Tcc (undergrad), Federal University of Pampa (in Portuguese). Available at: <https://repositorio.unipampa.edu.br/items/149343ef-3101-4a27-a50c-e9b18b8b07ff>.
- Devold, H. (2008). *Oil and Gas Production Handbook: An Introduction to Oil and Gas Production, Transport, Refining and Petrochemical Industry*. ABB Oil and Gas. Book.
- Esperitino, C., Blas, M., and Gonnet, S. (2022). Developing rdevs simulation models from textual specifications. In *Anais do IV Workshop em Modelagem e Simulação de Sistemas Intensivos em Software*, pages 41–50, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/mssis.2022.226306.
- Favre, J.-M. (2005). Foundations of Model (Driven) (Reverse) Engineering : Models – Episode I: Stories of The Fidus Papyrus and of The Solarus. In Bezivin, J. and Heckel, R., editors, *Language Engineering for Model-Driven Software Development*, volume 4101 of *Dagstuhl Seminar Proceedings (DagSem-Proc)*, pages 1–31, Dagstuhl, Germany. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. DOI: 10.4230/DagSem-Proc.04101.8.
- Fenton, N. and Bieman, J. (2014). *Software Metrics: A Rigorous and Practical Approach*. CRC Press, third edition edition. Book.
- Foundation, E. (2023a). Acceleo. Available at: <https://eclipse.dev/acceleo/>. Accessed on: 16 July 2023.
- Foundation, E. (2023b). Eclipse qvt operational. Available at: <https://projects.eclipse.org/projects/modeling.mmt.qvt-oml>. Accessed on: 16 July 2023.
- Foundation, E. (2023c). Sirius. Available at: <https://eclipse.dev/sirius>. Accessed on: 18 July 2023.
- Frohm, J., Lindström, V., Winroth, M., and Stahre, J. (2006). The industry's view on automation in manufacturing. *IFAC Proceedings Volumes*, 39(4):453–458. 9th IFAC Symposium on Automated Systems Based on Human Skill and Knowledge. DOI: 10.3182/20060522-3-FR-2904.00073.
- Gesser, R. S., Sartori, R., Damo, T. P., Vettorazzo, C. M., Becker, L. B., Lima, D. M., de Lima, M. L., Ribeiro, L. D., Campos, M. C., and Normey-Rico, J. E. (2022). Advanced control applied to a gas compression system of an offshore platform: From modeling to related system infrastructure. *Journal of Petroleum Science and Engineering*, 208:109428. DOI: 10.1016/j.petrol.2021.109428.
- Gravdahl, J., Willems, F., de Jager, B., and Egeland, O. (2000). Modeling for surge control of centrifugal compressors: comparison with experiment. In *Proceedings of the 39th IEEE Conference on Decision and Control (Cat. No.00CH37187)*, volume 2, pages 1341–1346 vol.2. DOI: 10.1109/CDC.2000.912043.
- Guisasola, T. and Maia, R. (2009). Mpa, um sistema de controle de plantas industriais. In *Lua Workshop 2009*, Rio de Janeiro, RJ, Brasil. Available at: <https://www.lua.org/wshop09/mpa.pdf>. Accessed on: 15 July 2023.
- Hästbacka, D., Vepsäläinen, T., and Kuikka, S. (2011). Model-driven development of industrial process control applications. *Journal of Systems and Software*, 84(7):1100–1113. DOI: 10.1016/j.jss.2011.01.063.
- Klatt, K.-U. and Marquardt, W. (2009). Perspectives for process systems engineering—personal views from academia and industry. *Computers Chemical Engineering*, 33(3):536–550. Selected Papers from the 17th European Symposium on Computer Aided Process Engineering held in Bucharest, Romania, May 2007. DOI: 10.1016/j.compchemeng.2008.09.002.
- Kourouklidis, P., Kolovos, D., Noppen, J., and Matragkas, N. (2021). A model-driven engineering approach for monitoring machine learning models. In *2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pages 160–164. DOI: 10.1109/MODELS-C53483.2021.00028.
- Liu, Y., Li, Y., Lu, H., Pan, Z., Dai, P., Sun, G., and Yang, Q. (2021). A full-scale process for produced water treatment on offshore oilfield: Reduction of organic pollutants dominated by hydrocarbons. *Journal of Cleaner Production*, 296:126511. DOI: 10.1016/j.jclepro.2021.126511.
- Lukács, G. and Bartha, T. (2022). Conception of a formal model-based methodology to support railway engineers in the specification and verification of interlocking systems. In *2022 IEEE 16th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pages 000283–000288. DOI: 10.1109/SACI55618.2022.9919532.
- Mailund, T. (2018). *Domain-Specific Languages in R: Advanced Statistical Programming*. Apress, USA, 1st edition. DOI: 10.1007/978-1-4842-3588-1.
- Makwashi, N., Zhao, D., Ahmed, T., and Paiko, I. (2019). Pipeline gas hydrate formation and treatment: A review. Available at: https://www.researchgate.net/publication/330262173_Pipeline_Gas_Hydrate_Formation_and_Treatment_A_Review.
- Mokhatab, S., Poe, W. A., and Mak, J. Y. (2015). *Handbook of Natural Gas Transmission and Processing: Principles and Practices*. Gulf Professional Publishing, third edition edition. DOI: 10.1016/C2017-0-03889-2.
- Nakaya, M. and Li, X. (2013). On-line tracking simulator with a hybrid of physical and just-in-time models. *Journal of Process Control*, 23(2):171–178. IFAC World Congress Special Issue. DOI: 10.1016/j.jprocont.2012.06.007.
- Neis, P., Wehrmeister, M. A., Mendes, M. F., and Pesente, J. R. (2023). Applying a model-driven approach to the development of power plant scada/ems software. *International Journal of Electrical Power Energy Systems*,

- 153:109336. DOI: 10.1016/j.ijepes.2023.109336.
- Otto, L. (2017). *DSL: Quebre a barreira entre desenvolvimento e negócios*. Casa do Código. Book.
- Parr, T. (2023). Antlr. Available at: <https://www.antlr.org/>. Accessed on: 16 July 2023.
- Paz, A., Boussaidi, G. E., and Mili, H. (2021). checsdm: A method for ensuring consistency in heterogeneous safety-critical system design. *IEEE Transactions on Software Engineering*, 47(12):2713–2739. DOI: 10.1109/TSE.2020.2966994.
- Plucenio, A., Vettorazo, C., Picon, B., Campos, M., and Lopes de Lima, M. (2016). Modeling and control of an oil platform gas compression station. *XXI Congresso Brasileiro de Automática*. Available at: https://www.researchgate.net/publication/309034010_MODELING_AND_CONTROL_OF_AN_OIL_PLATFORM_GAS_COMPRESSION_STATION.
- Ponsard, C. and Darquennes, D. (2023). Mbse tool selection process: Feedback from a railway case study. In *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pages 710–714. DOI: 10.1109/MODELS-C59198.2023.00114.
- (PUC-Rio), T. I. (2018). *MPA Programming Libraries from the Tecgraf Institute (PUC-Rio)*. Available at: <https://git.tecgraf.puc-rio.br/mpa/libs/pucrio-tecgraf>. Accessed on: 11/01/2023.
- PUCRJ (1993). The programming language lua. Available at: <https://www.lua.org/>. Accessed on: 15 July 2023.
- Raibulet, C., Arcelli Fontana, F., and Zaroni, M. (2017). Model-driven reverse engineering approaches: A systematic literature review. *IEEE Access*, 5:14516–14542. DOI: 10.1109/ACCESS.2017.2733518.
- Rugaber, S. and Stirewalt, K. (2004). Model-driven reverse engineering. *IEEE Softw.*, 21(4):45–53. DOI: 10.1109/MS.2004.23.
- Satuf, E., Pinto, S. F., and Dias, B. Q. (2009). Automatic alignment system for pra-1 pumping platform (in portuguese). In *5th Rio Automation Congress*. DOI: 10.5753/mssis.2023.235679.
- Schmidt, D. (2006). Guest editor’s introduction: Model-driven engineering. *Computer*, 39(2):25–31. DOI: 10.1109/MC.2006.58.
- Seborg, D. E., Mellichamp, D. A., and Edgar, T. F. (2010). *Process Dynamics and Control*. John Wiley & Sons. DOI: 10.1021/cen-v056n033.p045.
- Selic, B. (2003). The pragmatics of model-driven development. *IEEE Software*, 20(5):19–25. DOI: 10.1109/MS.2003.1231146.
- Shafieezadeh, M., Hormozi, M. K., Hassannayebi, E., Ahmadi, L., Soleymani, M., and and, A. G. (2020). A system dynamics simulation model to evaluate project planning policies. *International Journal of Modelling and Simulation*, 40(3):201–216. DOI: 10.1080/02286203.2019.1596779.
- Sheng, J. J. (2015). Enhanced oil recovery in shale reservoirs by gas injection. *Journal of Natural Gas Science and Engineering*, 22:252–259. DOI: 10.1016/j.jngse.2014.12.002.
- Silva Cruz, M. V., Damo, T. P., and Becker, L. B. (2020). Round-trip engineering for petrochemical industry automation. *IFAC-PapersOnLine*, 53(2):11824–11829. 21st IFAC World Congress. DOI: 10.1016/j.ifacol.2020.12.693.
- Soares, R. d. P. and Secchi, A. (2003). EMSO: A new environment for modelling, simulation and optimisation. In *Computer Aided Chemical Engineering*, volume 14, pages 947–952. Elsevier. DOI: 10.1016/S1570-7946(03)80239-0.
- Sommerville, I. (2011). *Software Engineering*. Pearson. Book.
- Steinberg, D., Budinsky, F., Paternostro, M., and Merks, E. (2008). *EMF: Eclipse Modeling Framework*. Pearson Education. Book.
- Taghaddos, H., Mashayekhi, A., and Sherafat, B. (2016). Automation of construction quantity take-off: Using building information modeling (bim). pages 2218–2227. DOI: 10.1061/9780784479827.221.
- Teixeira, P., Lebtog, B., and Basso, F. (2020). Diversity of mde toolboxes and their uncommon properties. In *Anais do II Workshop em Modelagem e Simulação de Sistemas Intensivos em Software*, pages 1–10, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/mssis.2020.12489.
- van Deursen, A., Klint, P., and Visser, J. (2000). Domain-specific languages: An annotated bibliography. *SIGPLAN Not.*, 35(6):26–36. DOI: 10.1145/352029.352035.
- Vepsäläinen, T., Sierla, S., Peltola, J., and Kuikka, S. (2010). Assessing the industrial applicability and adoption potential of the aukoton model driven control application engineering approach. In *2010 8th IEEE International Conference on Industrial Informatics*, pages 883–889. DOI: 10.1109/INDIN.2010.5549626.
- Vidal, C., Fernández-Sánchez, C., Díaz, J., and Pérez, J. (2015). A model-driven engineering process for autonomous sensor-actuator networks. *International Journal of Distributed Sensor Networks*, 11(3):684892. DOI: 10.1155/2015/684892.
- Wang, F., Yang, Z.-B., Huang, Z.-Q., Liu, C.-W., Zhou, Y., Bodeveix, J.-P., and Filali, M. (2020). An approach to generate the traceability between restricted natural language requirements and aadl models. *IEEE Transactions on Reliability*, 69(1):154–173. DOI: 10.1109/TR.2019.2936072.
- Wang, X., Love, P. E., Kim, M. J., Park, C.-S., Sing, C.-P., and Hou, L. (2013). A conceptual framework for integrating building information modeling with augmented reality. *Automation in Construction*, 34:37–44. Information Technologies in Safety Management. DOI: 10.1016/j.autcon.2012.10.012.
- Xenos, D. P., Ciccotti, M., Kopanos, G. M., Bouaswaig, A. E., Kahrs, O., Martinez-Botas, R., and Thornhill, N. F. (2015). Optimization of a network of compressors in parallel: Real time optimization (rto) of compressors in chemical plants – an industrial case study. *Applied Energy*, 144:51–63. DOI: 10.1016/j.apenergy.2015.01.010.
- Yusuf, A. Z., John, Y. M., Aderemi, B. O., Patel, R., and Mujtaba, I. M. (2019). Modelling, simulation and sensitivity analysis of naphtha catalytic reforming reactions. *Computers Chemical Engineering*, 130:106531. DOI: 10.1016/j.compchemeng.2019.106531.

A Main Compressor Model

<https://drive.google.com/drive/folders/1RbIiR78RVSSXLJR0AowGwMSJmN78gNro?usp=sharing>

B Model Transformations Source Code

<https://drive.google.com/drive/folders/1ffWAjkgpS8T0s0zswvpDtWkMGtI6SRxhP?usp=sharing>

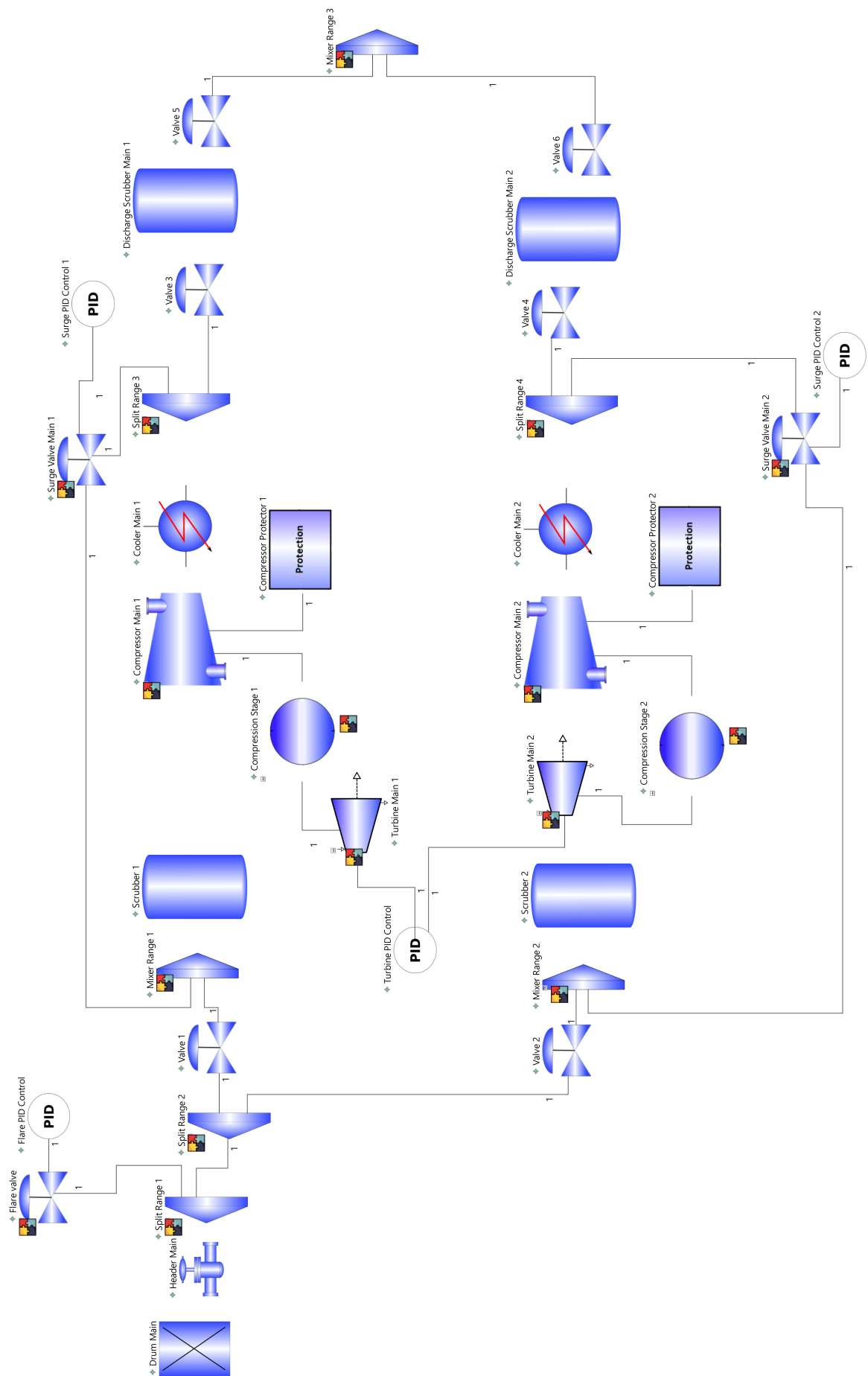


Figure 10. Main compressor in deployment diagram view.

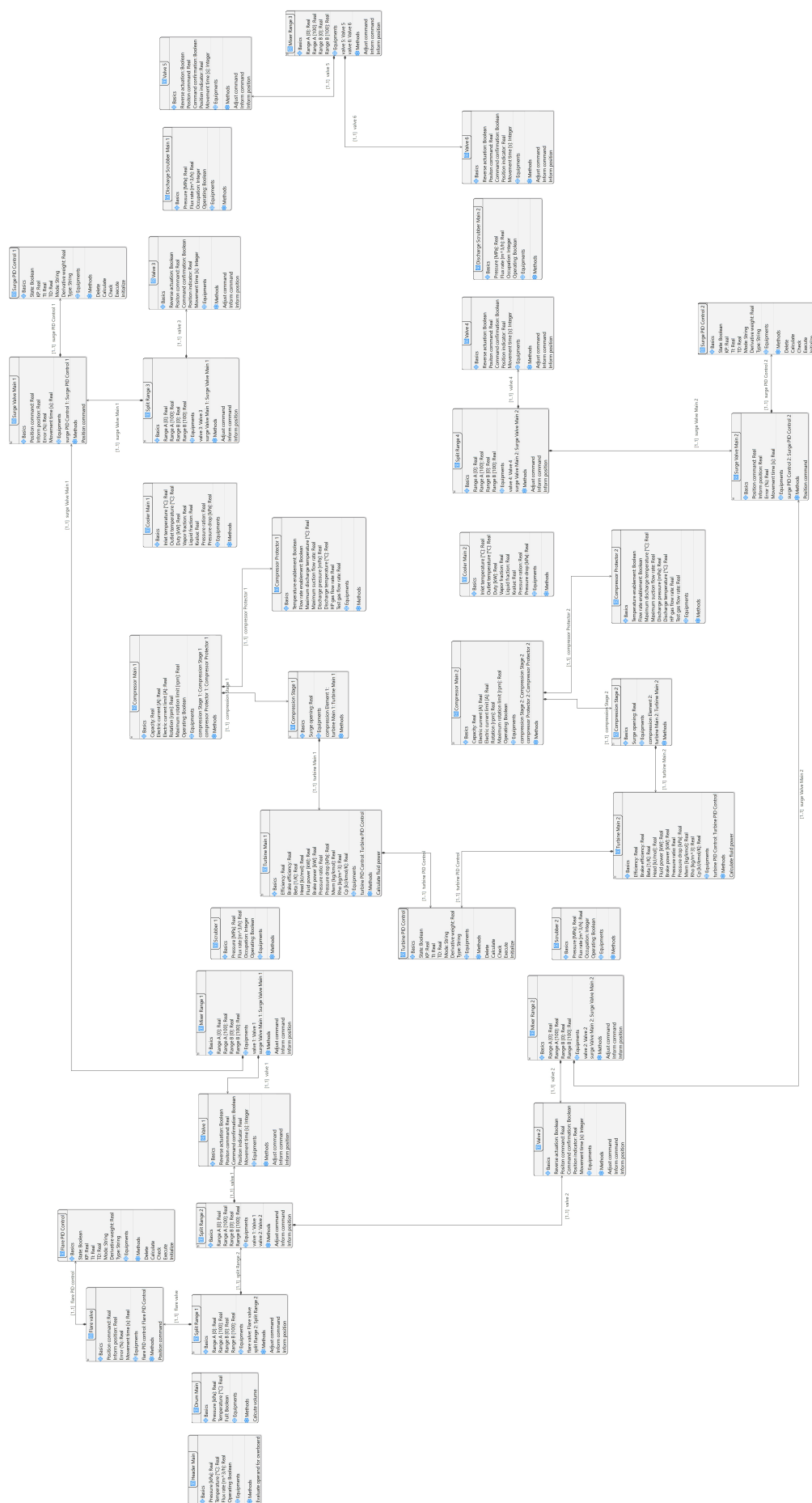


Figure 11. Main compressor in conceptual diagram view.