Capturing and Exploiting Plant Topology and Process Information as a Basis to Support Engineering and Operationa...

Esteban Arroyo

Capturing and Exploiting Plant Topology and Process Information as a Basis to Support Engineering and Operational Activities in Process Plants

Von der Fakultät für Maschinenbau
der Helmut-Schmidt-Universität/Universität der Bundeswehr Hamburg
zur Erlangung des akademischen Grades eines Doktor-Ingenieurs (Dr. Ing.) genehmigte

DISSERTATION

von

M.Sc. Esteban Arroyo Esquivel
aus Alajuela, Costa Rica

Hamburg, 2017

Gutachter:	Univ. Prof. Dr.-Ing. Alexander Fay
Vorsitzender:	Prof. Nina F. Thornhill
Tag der mündlichen Prüfung:	9. Juni 2017

Information forms the basis for the analysis of any problem as well as for the creation of every model, method, or algorithm conceived to solve that problem.

A Dio.

Preface

Automation offers various fields for scientific advancement. At the Institute of Automation Technology at Helmut Schmidt University / University of the Federal Armed Forces Hamburg, new methods for the engineering of complex automated systems are elaborated. The application of these methods in industrial scenarios is the ultimate aim of automation science and a benchmark for success.

"Automation of Automation" (AoA) has been recently coined as a term to describe the automation of engineering tasks within the design, implementation, test, commissioning, maintenance, and modernisation of automated systems. Different approaches have been developed how this can be accomplished to increase the quality and reduce the effort and length of the engineering process. A crucial element of these approaches is a model of the system, which should contain structural but, ideally, also dynamical aspects. However, to create such model must not increase the engineering effort. Therefore, various methods have been developed to generate the required model from existing engineering artefacts, such as models and documents from early engineering phases.

Dr.-Ing. Esteban Arroyo Esquivel has devoted his research to the acquisition of a digital plant model for brownfield plants, i.e., existing plants which have been designed and built long before digital plant models were introduced. In his thesis, he describes a method to analyse documents of plants which usually exist as paper printouts or PDF files only, and to convert this information into a digital structural model. In the further chapters of his thesis, he elaborated how this model can be enriched, combined, and employed to achieve "Automation of Automation".

Univ. Prof. Dr.-Ing. Alexander Fay

Acknowledgments

The completion of any important project in life involves the collaboration of many helping hands at different stages. This doctoral dissertation –final outcome of over three years of research at the Institute of Automation Technology, Helmut Schmidt University– was certainly not the exception. During this time, I was lucky to count on several wonderful people, who supported me academically and emotionally. I would like to thank every one of them for their contribution.

My first thanks goes to Univ. Prof. Dr.-Ing. Alexander Fay for the opportunity to join his chair as a research assistant and thereby for the chance to get to know a new country, language, and culture, as well as to acquire cutting-edge knowledge on the field of automation technology. His guidance, scientific advice, and support along these years were fundamental for the completion of this work.

With the same gratitude, I thank Prof. Nina F. Thornhill from the Center for Process Systems Engineering at Imperial College London for her interest in my research, as well as for the honor of serving as the second reviewer of this thesis.

Another special thanks goes to Mario Hoernicke, research partner at ABB Corporate Research, for his support within the experimental stage of this work, especially for his input on topics related to automatic derivation of plant simulation models.

Thanks as well to Johanna Meisner, Frank Schumacher, Zen-Zen Yen, Lorenzo Stroppa, and Sebastian Schroeck for their exhaustive review of the manuscript and their valuable recommendations. Likewise, I thank my colleagues, research assistants, and students for their feedback and interesting discussions on topics related to the content of this thesis.

Last but not least, I express my sincere gratitude to my beloved ones –family, friends and (amazing) girlfriend– who supported me unconditionally during this journey.

Many thanks to all!

Cologne, June 2017

Esteban Arroyo Esquivel

Contents

List of Figures

List of Tables

List of Acronyms

ACM	Alarm Connectivity Matrix
AF	Associated Facet
AFDD	Automated Fault Detection and Diagnosis
AM	Alarm Management
AML	Automation Mark-Up Language (AutomationML)
AoA	Automation of Automation
API	Application Programming Interface
AUC	Automation Use Case
BMP	Bitmap File Format
BS	Basic Shape
BSA	Basic Shape Array
CA	Character Array
CAD	Computer-aided Design
CAE	Computer-aided Engineering
CAEX	Computer-aided Engineering eXchange
CAx	Computer-aided x
CG	Characteristic Graph
CIL	Component Inherent Link
CLD	Control Logic Diagram
CM	Connectivity Matrix
DCDG	Dynamic Causal Digraph
DCS	Decentralized Control System
DFX	Drawing Exchange File Format
DGN	MicroStation Design File
DL	Directed Link
DOM	Document Object Model
DRW	Micrografx Designer Vector Graphics File Format

DWG	AutoCAD Drawing Database File
EPS	Encapsulated PostScript
FAT	Factory Acceptance Test
FDD	Fault Detection and Diagnosis
FEED	Front End Engineering Design
GHT	Generalized Hough Transform
GM	Geometric Matching
HAZOP	Hazard and Operability
HF	High-fidelity
HMI	Human Machine Interface
HT	Hough Transform
IE	Integrated Engineering
IGL	Ideal Gas Law
IL	Information Link
ISA	International Society of Automation
IT	Information Technology
JPEG	Joint Photographic Experts Group
LF	Low-fidelity
MOC	Management of Change
OCR	Optical Character Recognition
OIM	Object Information Model
OO	Object-Oriented
OSR	Optical Symbol Recognition
P&ID	Piping and Instrumentation Diagram
PAM	Plant Asset Management
PCA	Principal Components Analysis
PCS	Process Control System
PCT	Process Control Technology
PDF	Portable Document Format
PFD	Process Flow Diagram

PIM	Primitive Information Matrix
PLUT	Propagation Look-Up Table
PNG	Portable Network Graphics
PRF	Propagation-related Factor
ROI	Region of Interest
SAMA	Scientific Apparatus Makers Association
SCD	System Control Diagram
SDG	Signed Directed Graph or Signed Digraph
ST	Spanning Tree
SVG	Scalable Vector Graphics
TIFF	Tagged Image File Format
VG	Vector Graphics
VLE	Vapor-Liquid Equilibrium
XCM	eXtended Connectivity Matrix
XML	eXtensible Markup Language

Notation

A	Image
AG_i	Alarm group
B	Structuring element
BSA_{db}	Basic Shape Array stored in the database
CA_{db}	Character Array stored in the database
C_i	Connectivity chain of a tree-graph node
COG_{BS}	Center of Gravity of the BS
COG_i	Center of Gravity of the i_{th} primitive in the BSA
E	Edge set
G	Signed digraph
I	Number of rows in the CA
J	Number of columns in the CA
K_j	Weighting factor applied to j_{th} property of the CA
$\overrightarrow{L}$	Orientation vector of a primitive
M	Number of rows of the PIM
N	Number of columns of the PIM
O_i	Object or node in a tree-graph
O_{C-b}	Set of tree-graph nodes representing merging points
O_N	Instance of a control object created in the integrated topology model
P	Accumulator array
$\overrightarrow{R}$	Projection traced from the COG_{BS} to the COG_i
S_P	Set of primitives
T_A	Time of occurrence of a given alarm A
T_{CLD}	Tree-graph of a CLD
T_{cs}	Lag caused by the definition of alarm-related parameters in the PCS
T_{dyn}	Time delay or lag caused by system dynamics
T_{exp}	Expected time delay or lag between two given alarms
T_{obs}	Observed time delay or lag between two given alarms
$T_{P\&ID}$	Tree-graph of a P&ID
T_{thr}	Time delay or lag related to the particular definition of alarm limits
V	Vertex set
a_{ij}^{CA}	Value of the i_{th} row and j_{th} column of the CA
$a_{ij}^{CA_{db}}$	Value of the i_{th} row and j_{th} column of the CA_{db}
a_{YX}	Cell located in the Y_{th} row and X_{th} column of the PIM
b	Line intercept
d_i	Relative direction between the center primitive and the i_{th} external primitive
l_i	Relative vector length
p_i	Penalization factor
s	Uniform scaling factor
t_{db}	Permitted variance between BSA and BSA_{db} values

t_l	Permitted relative length deviation
t_ϕ	Permitted angle deviation
$\tilde{x}_{vi}$	Steady-state value
α_i	Reference angle of the i_{th} primitive in the BSA
Δ	Set describing the possible qualitative statuses of a graph node
ε	Error threshold
θ	Angular position
λ	Set describing the possible qualitative values of a graph edge
ν	Resemblance difference value
τ	Expected time delay (lag) of a graph edge
Φ	Quantitative strength of a graph node
ϕ	Gradient
ω	Rotation angle

Referencing convention

- Literature quoted in the text as [<cite-key>] can be found in *General literature*.
- References cited in the document as [<cite-key>]* are listed in *Publications of the author*.
- Sources referenced in the manuscript as [<name>,<year>]# are presented in *Standards, norms, and guidelines*.
- References cited in the document as [<name>,<year>]@ are to be found in *Internet sources and software*.
- Literature referenced in the text as [<cite-key>]& can be found in *Supervised student works*.

Terminology

Alarm: Audible and/or visible means of indicating to the operator an equipment malfunction, process deviation, or abnormal condition requiring a response [ISO 18.2, 2009]#. The response may be, for example, manual intervention, increased watchfulness, or initiation of further investigation [NAMUR NA102, 2003]#.

Alarm flood: Situation when the number of alarm activations exceeds the ability of the operator to process them [Rot09]. Typically, events where 10 or more alarms occur within a 10-minute time frame [HH10][ISO 18.2, 2009]#.

Causality: Cause-effect relationship between process variables [YDSC14].

Chattering alarm: Alarm that repeatedly transitions between the alarm state and the normal state in a short period of time [ISO 18.2, 2009]#.

Connectivity: Material, information, and energy flow paths between process equipment, sensors, actuators, and controllers [YDSC14].

Distributed control system: System for process control purposes which, while being functionally integrated, consists of sub-systems which may be physically separated and remotely located from one another. These sub-systems are normally connected by a communication link (e.g., data bus) [ISO 3511, 1984]#.

Engineering: Creative application of scientific principles to design or develop structures, machines, apparatus, or manufacturing processes.

Failure: A permanent interruption of the ability of a system to perform a required function under specified operating conditions [RAM88].

Fault: An unpermitted deviation of at least one characteristic property or parameter of the system from the acceptable/usual/standard condition [RAM88]. Herein also referred to as disturbance.

Monitoring: A continuous real-time task of determining the conditions of a physical system, by recording information, recognizing and indicating anomalies in the behavior.

Process: Sequence of chemical, physical, or biological operations for the conversion, transport or storage of material or energy.

Production plant: Facility to produce one or more products which typically contains a number of regulatory control functions in a process control system [NAMUR NE152, 2014]#.

Re-engineering: Partial or total application of an engineering process to existing structures, machines, apparatus, or manufacturing processes.

Semantics: Branch of linguistics and logic concerned with meaning. The two main areas are logical semantics, concerned with matters such as sense and reference, and presupposition and implication; and lexical semantics, concerned with the analysis of word meanings and relations [Oxf14].

Symptom: A change of an observable quantity from normal behavior [RAM88].

Syntax: The arrangement of words and phrases to create well-formed sentences in a language. The structure of statements in a computer language [Oxf14].

1. Introduction

1.1 Motivation

The engineering of production plants is a complex process involving the execution of planning and design tasks by several parties from different domains [SDS08, SFJ15]. Among others, process, mechanical, electrical and control engineering departments are confronted with the need to cooperate and exchange information when planning or revamping an industrial facility. Engineering knowledge generated during this process is documented following existing industrial standards as a basis for the completion of subsequent activities within the plant lifecycle [DIN EN ISO 10628, 2001]# [IEC 62424, 2008]# [ISO 14617, 2005]# [IEC 62023, 2011]#. It is in this fashion how plant and process engineering documents such as piping and instrumentation diagrams (P&IDs), electrical schematics, and control logic diagrams are created in industrial practice.

Existing plant engineering documentation, herein also referred to as legacy engineering documents, embodies a valuable source of plant and process information [YDSC14] reflecting a fairly actual state of a production facility. The reason for this is that an important part of the originally engineered –and documented– plant characteristics does not change significantly along the years, and if it does, the respective documentation is accordingly updated. For instance, the overall process design together with the plant topology depicted in process P&IDs remain relatively constant during the operative phase of the plant. Should the process undergo major functional or structural modifications due to plant maintenance or in response to new production demands [BS10], the newly designed plant and process characteristics are recorded in updated versions of the respective engineering documents. Most undocumented changes relate to minor physical and software modifications, e.g., the replacement of sensors and actuators, *ad hoc* control code changes and plant aging [LHW+13]*, which do not have a drastic impact on the overall plant design. Thus, from a functional perspective, legacy engineering documentation can be regarded as an actual and, therefore, reusable source of plant and process knowledge.

Along with the amount of valuable information contained in legacy engineering documents, the high effort and respective costs demanded by (re)engineering and operational processes certainly justify the interest of plant owners and contractors in reusing existing documentation as a basis to support and boost the efficiency of plant modernization and operation activities [DGT15]. Particularly in process control technology (PCT) projects in the process industry, existing guidelines reveal that the combined effort demanded by the different engineering phases accounts for nearly a 70 % of the total project effort [NAMUR NA35, 2003]#. Likewise, a number of sources report on higher operational costs owing to inefficient access, exchange, and reuse of existing plant and process data [SDS08]. In the light of such figures, the reuse of existing engineering documentation appears to be not only technically but economically advantageous.

Unfortunately, documentation reuse is in general not a simple task. A number of factors concerning the nature of available engineering documents hinder the access of plant systems (e.g., engineering, control, and enterprise tools) to the underlying data content, thus confronting automation specialists and plant personnel with the need to retrieve information manually when conducting design and operational activities. The cause of such a limitation has a historical background. With the explosion and widespread adoption of CAx tools along the 80's and 90's [KCC98], design documentation –until then primarily maintained on paper records– began to be stored in elementary digital

formats such as CAD models and PDF files. A significant part of engineering knowledge, however, continued to be exchanged on paper media, mainly due to confidentiality issues and fear of losing process know-how. As a result, legacy engineering knowledge found today in existing industrial facilities resides in a large and heterogeneous mix of plant and process documents [TBS99], both on paper and elementary digital formats. Thus and so, heterogeneity and non-computer-interpretable nature of legacy documents pose significant obstacles for the effective reuse of existing information.

On account of the aforementioned constraints, it is evident that the first step towards the effective exploitation of legacy engineering data is the conversion of existing documents into formalized and uniform computer-interpretable descriptions on object-oriented formats [YSS97]. Admittedly, combining such models with formalized expert know-how (e.g., heuristics and first-physical principles) may enable the automation of time-consuming and error-prone engineering and operational tasks, which in current practice are executed by manual means. Extrapolating such notions allows concluding that the automatic capture, formalization and integration of existing engineering information and process knowledge may be an important driver for the effective application of modern paradigms such as automation of automation (AoA→ 1.2.1) in industrial practice. Based on this line of thinking, the hypothesis motivating the realization of this work is formulated as follows:

> *It is possible to automate the capture and formalization of plant and process information contained in legacy engineering documents, and exploit resulting computer-interpretable models in combination with expert know-how as a means to reduce the efforts required for automation of automation in plant modernization and operational activities.*

The ultimate aim of this contribution is, therefore, to propose a functional methodology to automate this process and allow thereby for the effective exploitation of legacy engineering knowledge in process industries.

1.2 Context

This section is dedicated to contextualize this contribution in the frame of automation of automation as well as to identify the main shortcoming of previous works in the field of automatic reuse of legacy engineering documentation.

1.2.1 Automation of Automation (AoA)

Originally defined by Schmitz et al. [SSE09] as *the automated execution of tasks supporting the engineering process of automation systems,* the scope of automation of automation has been progressively extended in the last years towards further automation and control areas. To state some examples, the applicability of the concept has been proven in theoretical use cases including fault detection and diagnosis [Chr15, DAIT11, Iyu11, LKJJ14], HAZOP studies [FSS09], alarm management [SCTF13], and derivation of simulation models [BF13, HFB15].

Until now, however, the use of AoA in industrial practice has been limited owing to the need of fundamental, yet usually inexistent, computer-interpretable plant and process models stemming from the process engineering phase [HFB15, Hoe15]. The approach for capturing and exploiting existing plant and process data proposed in this contribution aims at marking a step forward in the possibilities for industrial implementation of existing AoA methods that, to date, are constrained by the need of time-consuming manual steps.

1.2.2 Overview of solutions for automatic engineering information reuse

Motivated by the numerous potential benefits offered by the effective reuse of existing engineering knowledge, previous research initiatives have explored a number of methods to capture and digitize legacy documentation [AOC+00, Ber06]. In spite of the resulting advances on character and symbol recognition techniques [LWJ04, SCLC02, WZY07], up to now, existing approaches are incapable of supporting the reliable recognition and formalization of complex engineering documents, particularly technical drawings composed by characters, symbols, connectivity, and underlying semantic content. The main reason for this shortcoming relates to the lack of approaches combining graphic recognition techniques with appropriate semantic analysis methods. In fact, several existing methods perform well at identifying objects (i.e., symbols and text) but fail at recognizing their underlying contextual meaning, which in turn prevents the subsequent representation of inferred information as formal computer-interpretable descriptions.

The situation is similar in the commercial sector, where most existing software solutions are limited to document digitization (i.e., scanning, storing, and indexing) and basic graphical recognition [RadialSG, 2016]@. Despite reducing data storage problems, such solutions are incapable of providing effective means for automatic data access, as produced digital documents are not computer-interpretable. While more sophisticated commercial tools exist ([Nextspace, 2015]@, [Aveva, 2016]@, and [Autodesk, 2016]@), allowing for example the automatic conversion of digital CAD drawings into object-oriented representations, their applicability in practice is limited since this type of records represents only a small portion of documents found in process facilities.

Overall, no state-of-the-art method allows for effective reuse of engineering documentation by combining graphics recognition, semantic analysis, and subsequent data formalization. Aiming at bridging the existing gap, this work introduces a new functional methodology enabling the capture and exploitation of legacy engineering records as a basis for modernization and operational activities in process plants.

1.3 Scope

The methodology proposed herein is valid for application within different AoA use cases across the plant lifecycle. Aiming at concreteness, however, the contribution focuses on specific applications and accordingly on the analysis of particular information. This section is dedicated to discuss the scope of the approach.

1.3.1 Use cases

The applicability of the proposed methodology is illustrated in two independent use cases. First, it is demonstrated how computer-interpretable models can be applied on (re)engineering tasks, specifically for the derivation of low-fidelity simulation models supporting the validation of control functions. Second, it is shown how connectivity and causality information derived from computer-interpretable models can be used in conjunction with process knowledge and expert know-how during operational tasks, particularly within plant abnormal behavior analysis. The motivation for targeting such use cases and their respective scope are discussed in the following.

1.3.1.1 Automatic derivation of low-fidelity simulation models

In current industrial practice, simulation is applied in most phases of the plant lifecycle. Among others, process design, virtual commissioning, HAZOP studies, FAT, and condition monitoring, are examples of the manifold activities in which digital plant models are used for simulation.

Unfortunately, due to the high modeling efforts as well as the necessary know-how required for the derivation of functional simulation models, up to now the use of simulation has not been adopted as a standardized practice within the engineering of process plants. In an attempt to reduce manual modeling efforts, industry and academia have explored methods for the automatic and semiautomatic derivation of digital plant models. Barth and Fay [BF13], for instance, presented a method for the generation of plant simulation models based on CAE documents. Hoernicke et al. [HFB15] introduced a concept for automatic derivation of virtual plants from human machine interfaces (HMIs), and Oppelt et al. [ODLW13] proposed an approach to integrated simulation based on CAx data. All these approaches require as a basis the existence of plant information sources in computer-interpretable formats (e.g., CAEX/AutomationML [IEC 62424, 2008]#, or ISO 15926 [ISO 15926-1, 2004]#). Yet, such digital representations are commonly nonexistent or unavailable in most facilities, which constrains the industrial implementation of current methods [DAIT11].

Accordingly, in the context of this use case, the present contribution aims at making legacy plant and process documentation (on scanned media and elementary digital formats) available in computer-interpretable form as a basis for the automatic derivation of low-fidelity simulation models.

1.3.1.2 Plant abnormal behavior analysis

Plant condition monitoring and alarm management, as fundamental components of modern process control systems (PCS), are called to gain the maximum profit of interdisciplinary plant and process knowledge and improve thereby the reliability and resolution of plant diagnostics. Based on this notion, several authors have studied multi-knowledge monitoring approaches and reported on tangible benefits in different use cases. Thornhill, Cox and Paulonis [TCP03], for instance, presented evidence that data-driven root-cause analysis can be enhanced by incorporating process knowledge, specifically connectivity and directionality of material and heat flows. Iyun [Iyu11], and more recently Landman and Jämsä-Jounela [LJJ16], reported on improved diagnostics by combining process connectivity, cause and effect analysis, and process know-how. Likewise, Schleburg et al. [SCTF13] and Rodrigo et al.[RCHH16] noted an important reduction of superfluous alarm notifications by combining plant connectivity, process data, and alarm logs.

Despite the numerous benefits offered by multi-knowledge approaches, most state-of-the-art automated diagnosis methods do not exploit the richness of information found in process facilities. The main reason for that is the difficulty entailed for information collection, access, and timely retrieval, since data sources are typically heterogeneous, distributed, and non-computer-interpretable. In an effort to contribute towards the exploitation of such knowledge and provide additional methods for enhanced plant diagnostics, this work presents a new multi-knowledge approach to fault propagation analysis based on combined exploitation of plant and process information extracted from legacy engineering documents and expert know-how. The approach is intended to serve as a basis for alarm grouping as well as a front end for determining specific process signals in which complementary diagnosis methods should be applied during root-cause analysis.

1.3.2 Information

Information forms the basis for the analysis of any problem as well as for the creation of every model, method or algorithm conceived to solve that problem. Not any information type, however, is suitable or useful during the problem solving process, but only that one chosen and processed by considering the specific requirements of the targeted use case or application. Especially in the context of this contribution, which is fundamentally aimed at capturing and exploiting plant and process information to support specific plant activities, the definition of a clear information scope is therefore highly relevant. This subsection is accordingly dedicated to define the different types of information in the scope of the approach as well as the main criteria considered for their selection.

1.3.2.1 Documents

In what concerns plant documents, this contribution focuses specifically on the analysis of piping and instrumentation diagrams (P&IDs) and control logic diagrams (CLDs), both on scanned media and scalable vector graphics formats. The selection of these drawings as source documents is based on information requirements of the analyzed use cases as well as on applicability and complexity considerations. Among them:

- Previous research has shown that the use of connectivity and causality information depicted in P&IDs can significantly enhance the performance of plant abnormal behavior analysis [TCP03, Iyu11, SCTF13]. On this regard, Yang et al. [YDSC14] stated that P&IDs, as one of the main resources to capture plant connectivity, must be converted into standard, easily accessible and computer-interpretable formats aiming at enhanced plant diagnostics.
- A number of approaches have exploited plant topology information contained in P&IDs for the automatic derivation of process simulation models. Important reductions of engineering effort have been observed as a result of the use of derived simulation models within the validation of new control solutions in process plants [BF13, HFB15, AHFR16].
- As the main outcome of the Front End Engineering Design (FEED) phase, P&IDs embody the main source of plant and process information. Therefore, several authors have regarded these documents as the most important basis for the (re)engineering and implementation of control systems [Dra05, Bar11, Bur04, Ahr01, MBS$^+$11].
- The manner in which a process behaves during abnormal conditions is directly determined by the characteristics of its control system. Incorporating control information depicted in CLDs into root-cause analysis is, therefore, fundamental to explain observed plant symptoms.
- Studies have shown that the propagation of disturbances along a process does not occur only through material flow connections, but also through information links within control structures [YAB$^+$06, Chr15, DGT15]. Detailed descriptions of control loops and their interconnections with process variables can be extracted from CLDs.
- The graphical complexity of P&IDs and CLDs makes their manual recognition and formalization a time-consuming and error-prone task.
- Up to now, no existing academic or industrial approach offers the required capabilities to capture and formalize the plant and process information depicted by P&IDs and CLDs.

For the purpose of this approach, the content of interest contained in the source documents is restricted to:

- Passive components (e.g., tanks, heat exchangers, and columns)
- Instrumentation and control devices (e.g., sensors, actuators, and control functions)
- Control logic blocks (e.g., logical operators and controllers)
- Connectivity among components, devices, and blocks; specifically pipelines and information connectors
- Tags (text identifiers) of all named components, devices, and blocks

1.3.2.2 Plant data

Plant component-specific data (e.g., power and asset types) and plant dimension-specific information (such as lengths and diameters of pipelines) found in manufacturer manuals and other process design records can be used –if available– to enrich the information captured from analyzed legacy documents (→ 1.3.2.1). A concept for identification and description of use-case specific information is proposed to facilitate the collection of relevant plant data. Due to its unstructured nature, however, the aggregation of this information into generated object-oriented plant models is left to the user as a manual task. Although, plant data can increase the accuracy of derived simulation models as well as reliability of plant diagnostics, the developed methods are designed to cope with the lack of this source of information in cases of no availability.

1.3.2.3 First physical principles and expert know-how

In addition to the information provided by target documents and plant data, further knowledge concerning the physics of chemical processes is required during plant abnormal situation analysis. Such knowledge typically stems from first physical principles and process know-how gathered by experts while dealing with and solving faulty-operation cases. Previous research has shown that the consideration of process know-how in form of rules can increase the reliability of automated plant diagnostics [SGC+99, Chr15]. Existing approaches, however, have concentrated mostly on the capture of process-specific know-how, which hinders their effective applicability in other processes. Aiming at less level of detail but greater generality and extended applicability, the approach presented herein targets the incorporation of general process know-how, concretely first principles and heuristics of general validity.

1.3.2.4 Online process data

As a further information source, online process data, specifically process signals collected from the PCS, a data server, or a simulation environment, can be incorporated to increase the accuracy and resolution of plant diagnostics. Static addresses of process tags can be aggregated in object-oriented plant models and be used to access online process values when required.

1.3.3 Systems and processes

The methods for automatic model derivation and plant abnormal behavior analysis presented in this contribution are intended for use in continuous production systems typically found in the chemical, petrochemical, and power generation industries. Focus is set on processes operated for long periods of time, implying thereby that batch processing, start-ups and shut-down procedures are out of the scope of the approach.

1.4 Case study

This section presents a model of a middle-scale chemical process satisfying the scope requirements previously discussed in this chapter. The presented model has been accordingly chosen as the case study used to evaluate the applicability of the methods proposed along the contribution. In the following, a detailed description of the model structure and process characteristics is provided.

1.4.1 Overview

The chosen case study is the Tennessee Eastman Process (TEP), a non-linear model of a complex multi-component chemical system [BRJ15] originally proposed by Downs and Vogel in 1993 [DV93]. Although several years have past since its introduction, the TEP remains an important test-bench for the purpose of comparative studies and validation of algorithms throughout a broad range of disciplines including system identification [JSL01], plant-wide control design [MY94] and fault detection and diagnosis [YY15, YDH$^+$12].

1.4.2 Plant topology

1.4.2.1 Plant structure and instrumentation

The TEP consists of five main operation units, namely: a two-phase reactor (R003) with cooling and agitator systems, a product condenser (C115), a vapor-liquid separator (S012), a stripping column (E005), and a reboiler (G111). In addition, it comprises eleven valves (V160-V170), two pumps (P101, P102), and a compressor (P100).

The original version of the model [DV93] has 41 process measurements including flow (F), pressure (P), temperature (T), level (L), composition (A), and rotational speed (S). In recent years, the model was revised and provided with 32 additional measurements, specifically further reactant compositions and temperatures, as well as flows and temperatures of process utilities [BRJ15].

For the purpose of this work, process alarms and component tag names have been included in the revised TEP version. The resulting plant schematic, presented in Appendix A, is the process P&ID used in the sequel to evaluate the different proposed methods.

1.4.2.2 Control

The TEP is an integrating and open-loop unstable process showing strong interactions and nonlinearity in its fluid and chemical dynamics. The process has twelve degrees of freedom: the position of valves V160 to V170, and the agitation speed of reactor R003.

Along the years, several regulatory approaches have been proposed to control the TEP [MY94, Luy96, LG95, Ric95]. Existing control structures have targeted different control objectives and accordingly have deployed different regulatory strategies. McAvoy et al. [MY94] and Ricker et al. [Ric95] for instance, explored advanced process control structures for reducing feed variability and satisfying economic objectives, whereas Luyben [Luy96] proposed a basic structure to allow for rapid changes in production rate.

Due to its topological and algorithmic simplicity, Luyben's basic regulatory structure [Luy96] was chosen as the TEP control scheme to be used in this work. The reason for the selection of a simple control structure relies on the fact that the validation of the proposed method for plant abnormal behavior analysis (→ 6) requires a detailed manual observation of simulation signals to explain observed symptoms and determine whether identified propagation paths are correct or incorrect. A simple control structure can facilitate this task by not obscuring the traceability of events.

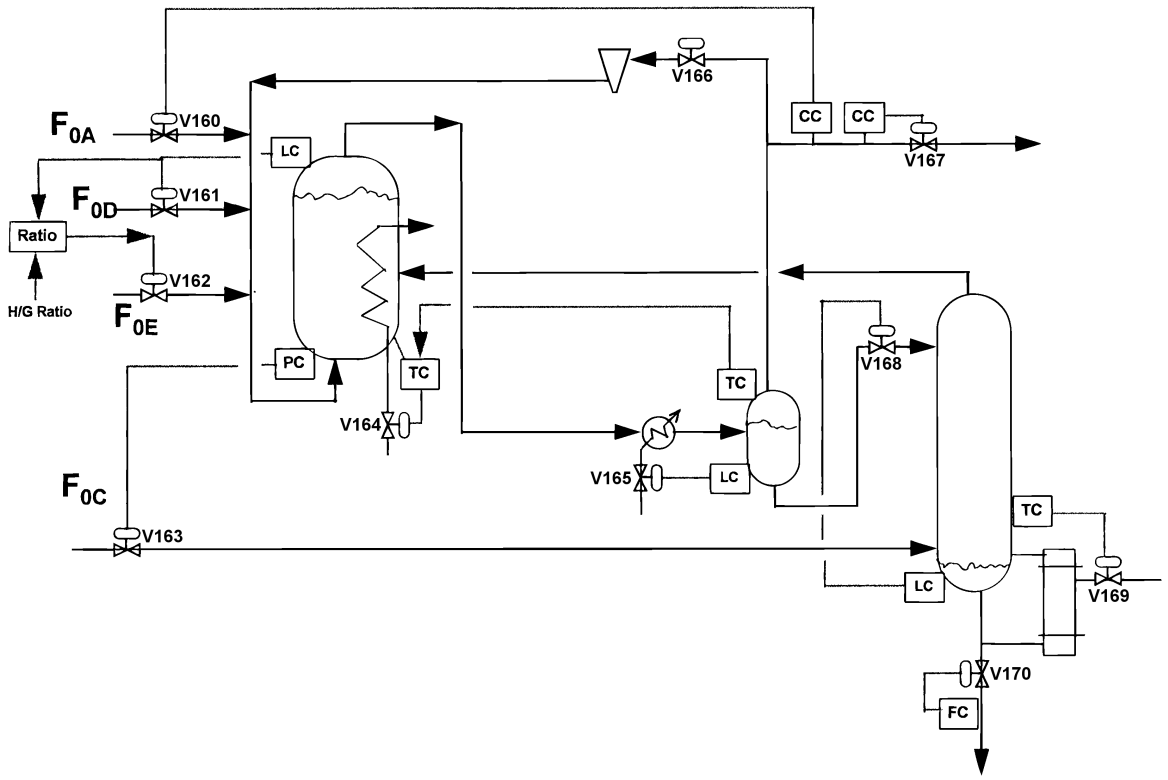


Figure 1.1: Luyben's regulatory control structure. Source [Luy96]

a. Lyuben's control structure

Luyben's basic regulatory control structure (see Figure1.1) uses ten degrees of freedom to control the TEP: 1 to control production rate (V170), 1 to control product quality (V169), 1 to satisfy pressure constraints (V163), 3 to control liquid levels (V161, V165, V168), 1 to define mixing ratios (V162), 2 to balance component concentrations (V160, V167), and 1 to control the reactor temperature (V164). The remaining two degrees of freedom, i.e., agitator speed and the position of the gas recycle valve (V166), are kept constant. As depicted in the control logic diagram of Appendix B, the control structure comprises proportional-only controllers in single, cascade, and ratio control configurations.

1.4.3 Process description

1.4.3.1 Reaction stoichiometry

The TEP produces two products (G and H) and a byproduct (F) from four reactants (A, C, D, and E) and one inert B. The reaction stoichiometry of the process is presented in equations 1.4.1 to 1.4.4:

$$A_{(g)} + C_{(g)} + D_{(g)} \rightarrow G_{(l)} \qquad Product\,1 \tag{1.4.1}$$

$$A_{(g)} + C_{(g)} + E_{(g)} \rightarrow H_{(l)} \qquad Product\,2 \tag{1.4.2}$$

$$A_{(g)} + E_{(g)} \rightarrow F_{(l)} \qquad Byproduct \tag{1.4.3}$$

$$3D_{(g)} \rightarrow 2F_{(l)} \qquad Byproduct \tag{1.4.4}$$

All reactions are irreversible, exothermic, and approximately first-order with respect to the reactant concentrations. The reaction to produce G has a higher activation energy resulting in more sensitivity to temperature. Reaction rates are a function of temperature through an Arrhenius expression.

1.4.3.2 Process

Gaseous reactants are fed into a reactor to form liquid products. The gas phase reactions are catalyzed by a nonvolatile catalyst dissolved in the liquid phase. Products leave the reactor along with the unreacted components as vapors. The product stream passes through a condenser and from there to a vapor-liquid separator. Vapors leaving the top of the separator recycle back through a centrifugal compressor to the reactor feed. The product stream exiting at the base of the separator is forced by a pump into a stripping column, which removes gaseous reactants by stripping with reactants A and C. Removed vapors are recirculated to the reactor, while products G and H together with remaining light components exit at the stripper base. The former are forced by a pump into a downstream refining section for further separation, whereas the latter are routed to a reboiler and recycled back to the stripper. The byproduct F is primarily purged from the system as a vapor from the vapor-liquid separator.

1.4.3.3 Model flow dynamics

Within the TEP model, flow dynamics are determined by pressure differences between volume objects (i.e., reactor, condenser, stripper, and separator). Modeled pipes have negligible length and roughness, and accordingly no pressure drops.

1.5 Thesis layout

The methodology proposed in this contribution comprises concepts of diverse nature including image processing, code pattern analysis, object-oriented modeling, simulation, and causal reasoning. Accordingly, this thesis is structured in a modular format in which the different methodology phases are addressed in individual but functionally related chapters. Each chapter represents a

self-contained unit with its own theoretical background, review of the state of the art, proposed method, validation and assessment, which can be used alone or in combination with other chapters according to the formalized description of Figure 1.2. In the following, a brief description of each chapter as well as an overview of the complete methodology are provided.

Chapter 2 addresses the first phase of the methodology: *information capture*. This chapter starts with a brief overview of the current problematic on documentation management in industrial practice to be followed by a review of relevant theoretical background and related work in the field of automatic document analysis. Based on the identified shortcomings of the state of the art, two novel methods for the automatic recognition of legacy engineering documents are presented. The first one, raster-based document recognition, is designed for the analysis of scanned paper documents, whereas the second, vector-based document recognition, is aimed at the processing of documents in elementary digital formats. The recognition performance of both methods is demonstrated based on the analysis of the considered TEP schematics.

The second phase of the methodology, *information formalization*, is discussed in **Chapter 3**. Here, a concept for the object-oriented (OO) modeling of information captured by the recognition methods is presented. As a result of this phase, single OO models of the analyzed documents are obtained.

Chapter 4 deals with *data integration*, the third phase of the proposed methodology. This chapter shows how the information contained in available single object-oriented models can be merged into an integrated plant topology model. Furthermore, concepts allowing for the aggregation of additional plant and process information, as well as for effective data retrieval, are introduced in this chapter.

Aiming at demonstrating the applicability of the resulting integrated plant topology model, **Chapters 5** and **6** address the last phase of the methodology, *information exploitation*. Along these chapters two examples illustrating the possible use of the captured, formalized, and integrated plant and process information are presented. **Chapter 5** discusses how the integrated plant topology model can serve as a basis to support plant modernization activities, specifically through the automatic derivation of low-fidelity process simulation models. **Chapter 6**, in turn, presents a use case of plant abnormal behavior analysis showing how connectivity, causality, and other relevant information derived from the integrated plant topology model can be used in conjunction with formalized expert know-how to support fault diagnosis and alarm management in process plants.

At last, **Chapter 7** draws the main conclusions of the contribution and points out relevant considerations for future work.

Figure 1.2 presents a formalized description of the overall methodology in accordance to VDI/VDE 3682 [VDI/VDE 3682, 2014]#. As it can be observed, the diagram depicts the sequence of processes $(O_1, ..., O_5)$ required to transform source plant and process information $(I_1, ..., I_{10})$ into aimed simulation and diagnosis results (I_{11}, I_{12}). Note that input information items $(I_1, I_2, I_7, I_9, I_{10})$ were previously specified in subsection 1.3.2, and that processes $(O_1, ..., O_5)$, corresponding to the different methodology phases, are introduced along **Chapters 2** to **6**.

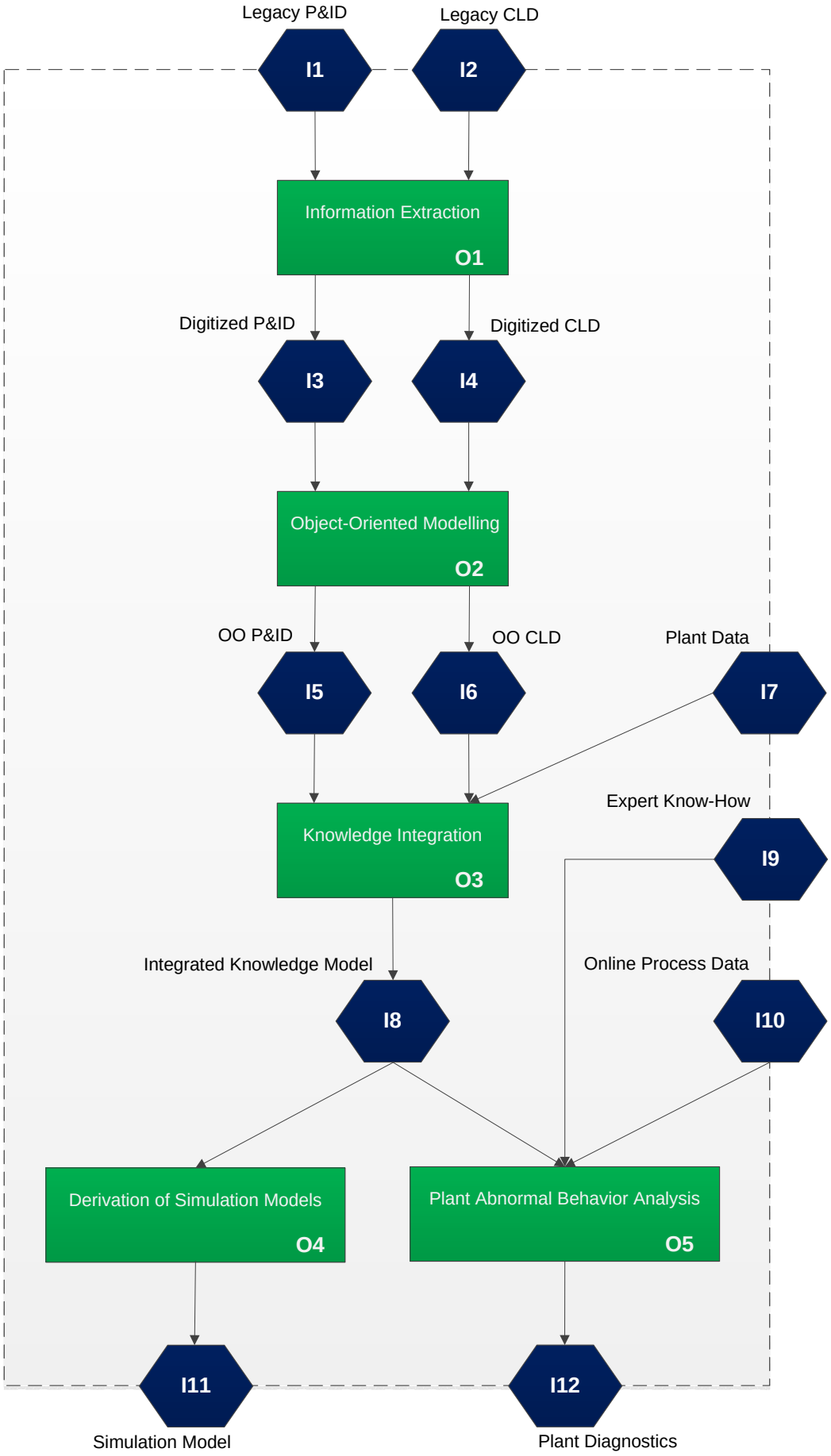


Figure 1.2: Formalized concept description

2. Automatic information extraction from engineering documents

2.1 Documentation in process industries

The size and complexity of existing process plants go hand-in-hand with the vast amount of documentation filed along their lifecycles. Structural models, process descriptions, piping and instrumentation diagrams (P&IDs) and control schematics are only a few examples of the broad range of legacy plant documents typically stored as rudimentary data formats such as paper, scanned media, and basic digital files.

As discussed in Chapter 1, existing legacy documentation embodies a rich source of plant and process knowledge which can serve as an important basis to support plant modernization and operational activities. As of today, nevertheless, the utilization of this information source in most industries has been hindered by the obstacles entailed for its timely localization and subsequent data extraction. Main reason for this stems from the heterogeneous, non-computer interpretable, and in occasions non-standard-compliant nature of existing documents.

Within the last years, information reuse and integration has acquired increasing relevance in the process industry [DM07]. Evidence of this are the significant efforts invested by companies in the oil and gas, and chemical sectors to digitize and consolidate existing legacy plant and process documentation [Intergraph, 2016]@. This tendency is expected to continue in the context of the ongoing Industry 4.0, where the availability of information stemming from all instances involved in the value chain is fundamental. To date, however, most currently used digitization methods are limited to scanning, storing and indexing [RadialSG, 2016]@. Digitized documents resulting from the application of such methods are not more than rigid digital artifacts, which reduce the complexity of data storage and visualization but cannot be fully exploited for automation tasks since they are not computer-interpretable.

While it is true that other tools allowing for the automatic conversion of digital files (e.g., CAD designs) into object-oriented models exist (e.g.,[Nextspace, 2015]@, and [Intergraph, 2016]@), such files account only for a small portion of those documents available in process facilities. As a consequence, automation specialists and process experts are confronted with the need to retrieve and consolidate information manually when conducting daily activities. This tedious procedure has a significant impact on the timely execution of required tasks and thus on the overall efficiency of (re)engineering and operational processes.

With the aim of allowing for a more effective reuse of legacy engineering data and alleviate thereby current difficulties for data retrieval, this chapter is dedicated to introduce a new approach for the automatic digitization and computer-interpretable description of legacy design documents, specifically piping and instrumentation diagrams (P&IDs) and control logic diagrams (CLDs), both on raster images and vector-graphics formats.

The remainder of this chapter is organized as follows: Section 2.2 presents relevant concepts on the areas of engineering diagrams, graphic formats, and graphic processing techniques. Section 2.3 reviews existing approaches to content recognition in legacy engineering documents, while Section

2.4 discusses their main shortcomings. Considering the identified knowledge gap, Section 2.5 proposes two novel methods for the automatic recognition and analysis of legacy P&IDs and CLDs. The applicability of the proposed methods on the examination of existing diagrams of the TEP is demonstrated in Section 2.6, and, based on experimental results, Section 2.7 assesses the major advantages and limitations of the methods, as well as discusses their potential applicability in industrial practice. Finally, Section 2.8 rounds up the chapter with a brief summary.

2.2 Theoretical framework

2.2.1 Engineering diagrams

2.2.1.1 Piping and Instrumentation Diagram (P&ID)

A piping and instrumentation diagram also referred to as piping and instrument diagram (P&ID) is a document representing the technical realization of a process by means of graphical symbols [DIN EN ISO 10628, 2001]#. P&IDs typically include information regarding function or type of equipment, process measurements and control functions, identification numbers for equipment and instrumentation (tags), indication of nominal diameters and pressure ratings, and specifications of material and type of piping. Figure 2.1 illustrates two examples of P&IDs.

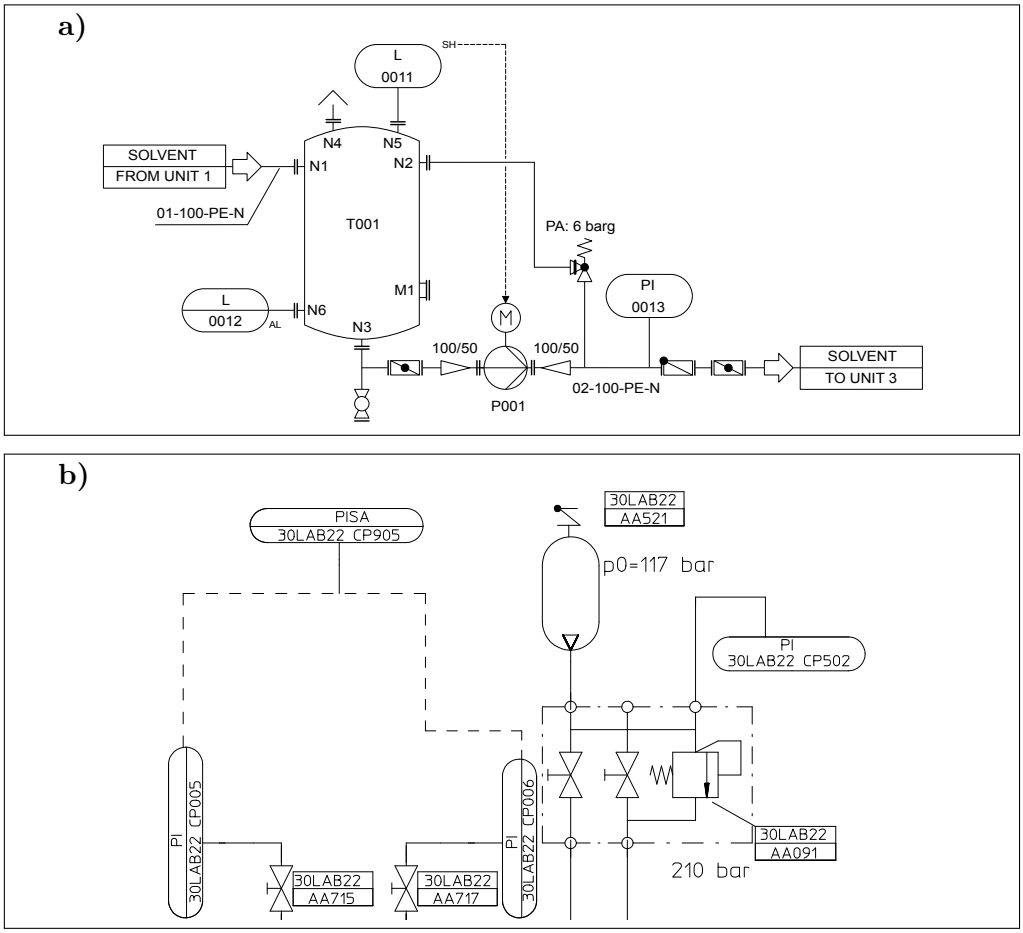


Figure 2.1: Excerpts of P&IDs found in the process industry. (*a*) Fragment of a chemical facility [Wikipedia Foundation, 2017]@ (*b*) Segment of a power plant

Common symbology and designations used in P&IDs can be found in different standards. ISO 14617 [ISO 14617, 2005]#, for instance, specifies graphical symbols for common pieces of equipment as well as clauses on identification and reference designation for sensors and actuators (see Appendix C). It is worth noticing, however, that despite the wide acceptance of standards in process industry, a large number of legacy P&IDs found in existing facilities are not fully standard-compliant. Differences of symbology, line strokes, and position of tag identifiers are typical variations encountered among legacy documents. The diagram excerpts depicted in Figure 2.1 and the TEP schematic presented in Appendix A are examples of non-fully-compliant P&IDs found in the process industry.

2.2.1.2 Control Logic Diagram (CLD)

A Control Logic Diagram (CLD) commonly referred to as Block Diagram (BD) is a graphical schematic providing a simple representation of the operation of individual equipment controls using basic symbols [LANL, 2006]#. These symbols functionally relate control input-output actions without implying a specific hardware realization or describing involved low-level instrument signals.

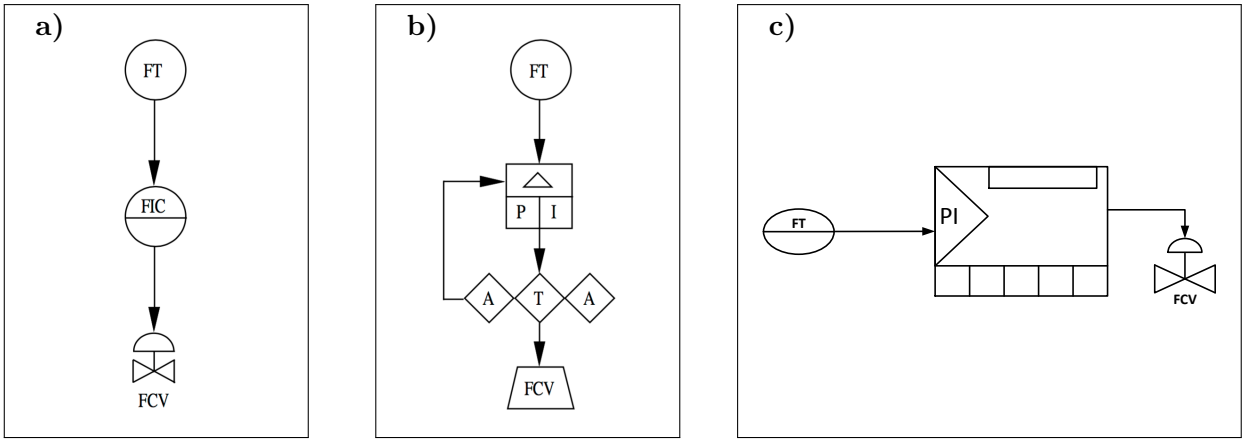


Figure 2.2: Representation of a single flow control loop in different CLD standards. (*a*) ISA diagram (*b*) SAMA diagram (*c*) Company-specific diagram

In practice, CLD standards typically vary from industry to industry. For instance, while the ISA standard [ISA 5.2, 1992]# is widely used in the chemical and petrochemical sectors, the SAMA [SAMA PMC 22-1, 1981]# standard is preferred in power generation [Wad04]. As in P&IDs, existing CLDs are in many cases not fully standard-compliant. In fact, CLD drawing rules are often defined internally by each company based on both legacy company standards and guidelines used by control equipment manufacturers and automation providers. Figure 2.2 shows a simple flow control loop using ISA, SAMA, and company-specific standards.

2.2.2 Graphic types and data formats

2.2.2.1 Raster graphics

In computer graphics, a raster image or bitmap is a dot matrix data structure representing a generally rectangular grid of pixels (see Figure 2.3a). A raster is technically characterized by the width and height of the image in pixels and the number of bits per pixel (intensity). Raster images are stored in image files with varying formats such as PNG, BMP, JPEG, and TIFF.

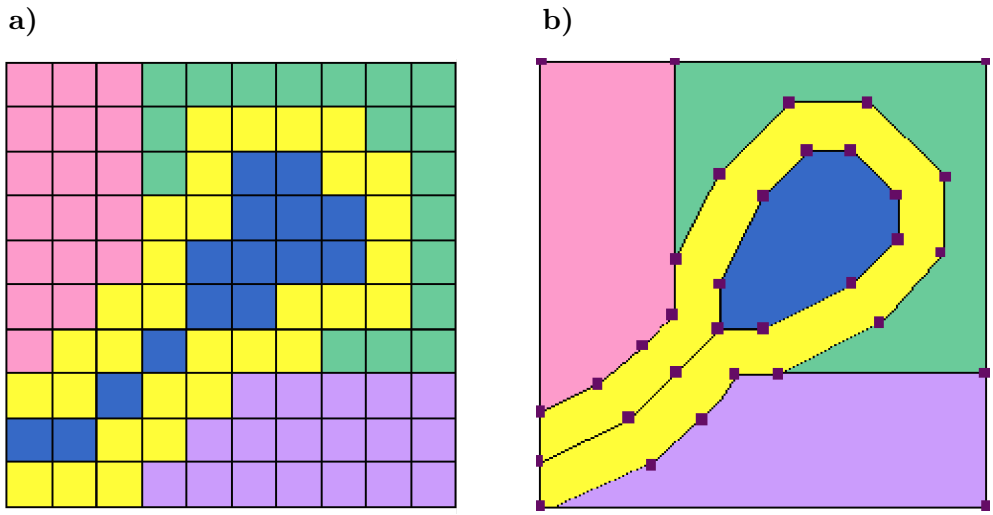


Figure 2.3: Structures of raster and vector graphic images. (*a*) Raster image (*b*) Vector graphics image. Source [Pagin, 2016]@

2.2.2.2 Vector graphics (VG)

As depicted in Figure 2.3b, vector images use geometric shapes such as points, lines, curves, and polygons to represent objects in an image. This structural property allows vector-based images to be scaled without loss of quality, in contrast to raster images. Common types of vector graphics include EPS, DRW, and SVG (→ 2.2.2.4).

2.2.2.3 Computer Aided Design (CAD) graphics

CAD graphics designates a family of specialized vector-based drawing formats largely used in engineering and architectural design. In such formats, drawings are constructed based on basic geometric elements such as points, lines, circles and curves, or most commonly based on objects or templates defined in standard and custom databases. Objects contain not only shape information but also may specify application-specific attributes such as materials and dimensions.

Unlike common vector graphics, CAD data formats are often proprietary. However, most off-the-shelf CAD tools (e.g., AutoCAD, MicroStation, and CATIA) offer built-in functionalities for exporting native files to standard exchange formats such as raster and vector graphics PDF (→ 2.2.2.5). Common CAD extensions include DWG, DXF, and DGN.

2.2.2.4 Scalable Vector Graphics (SVG)

Scalable Vector Graphics (SVG) is an XML-based open standard format for the representation of vector graphics [W3C, 2015b]@. The format provides a large set of elements such as lines, rectangles, circles, ellipses, polylines, polygons, paths, and text. Primitives are created by instantiating elements and specifying coordinates and attributes. The appearance of primitives (e.g., stroke width, and color) is defined by style attributes, whereas structural modifications (e.g., rotation, translation, scaling, and skew) can be declared by transformation attributes. Figure 2.4 depicts a fragment of SVG code.

```
<path id="path3547"
d= "M 484.04, 380.70, 484.04, 386.25"
style= "fill:none; stroke:#000000;stroke-width:0.88492644"/>
<polyline id="polyline23"
points="208.9, 459.53,208.9,494.28"
style="fill:none;stroke:#FFB90F;stroke-width=0.25"/>
<polygon id="polygon45"
points="374.531,379.531, 379.536, 374.536"
style="fill:none; stroke:lightblue; stroke-width:0.25"/>
<text style:"fill:blue; stroke:blue"
transform: scale(0.25)"x="1496" y="2124">V</text>
```

Figure 2.4: Excerpt of SVG code

Compared to CAD drawings, SVG-based schematics are less formal in terms of semantic content, as they are limited to represent the graphic structure of symbols but not their domain-specific properties. Document information formalism is discussed in more detail in Chapter 3.

2.2.2.5 Portable Document Format (PDF)

Portable Document Format (PDF) is a file format used to present data content in an application-independent manner [Adobe, 2008]@. A PDF file encapsulates a description of a fixed-layout flat document, including text, fonts, graphics, and additional information. Both raster and vector graphics can be embedded in PDF files. Due to its versatility, the PDF format has been widely used in practice for storing digitized legacy engineering documents.

2.2.3 Graphic recognition techniques

2.2.3.1 Hough Transform and Generalized Hough Transform

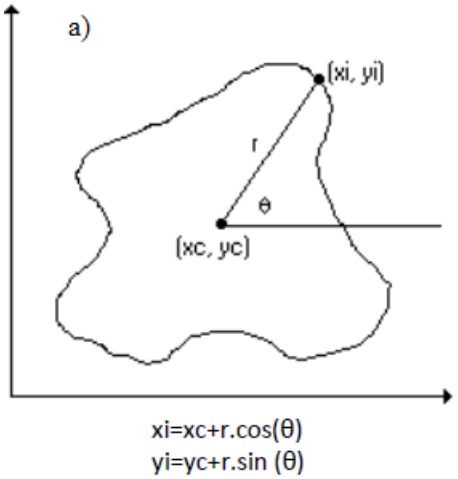
Hough Transform (HT) is a standard raster image analysis method for finding curves (i.e., series of interconnected pixels) that can be defined in parametrical forms [Bou09]. The method relies on the notion that certain curves can be analytically described by few parameters. Straight lines in a plane, for instance, are fully determined by their slope m and y-axis intercept b (eq. 2.2.1), or alternatively by their magnitude ρ and angular position θ (eq. 2.2.2).

$$y = mx + b \tag{2.2.1}$$

$$\rho = xcos(\theta) + ysin(\theta) \tag{2.2.2}$$

The first step of the HT for lines detection is the formation of a two-dimensional parameters space, i.e., a matrix or accumulator array P whose columns represent a finite number of values for the first parameter (e.g., θ), and whose rows represent the corresponding values for the second parameter (e.g., ρ). The actual transformation step consists in running through all the pixels (x, y) of the input image, incrementing a counter in those accumulator cells which represent straight lines passing through the pixel which is currently being looked at. The result of the transformation is a filled accumulator.

All those accumulator cells, whose values are larger than one, represent a straight line which contains at least two pixels of the original image. The larger the cell value, the more pixels are contained in the line represented by the cell. The aim of the analysis is to determine the maxima in P, as those cells indicate the position of significant straight lines in the image [BFRR01]. An analogous algorithm is used for locating other parametric shapes such as circles and ellipses.



b)

Gradient Value (ϕ)	(r, θ) Values
Φ_1	(r_{10}, θ_{10})
Φ_2	$(r_1, \theta_1), (r_5, \theta_5), (r_9, \theta_9)$
Φ_3	$(r_3, \theta_3), (r_4, \theta_4)$

Figure 2.5: Generalized Hough Transform. (*a)* Edge description (*b)* R-table

Based on Hough's findings, Ballard [Bal81] proposed years later an algorithm generalizing the HT for matching arbitrary non-parametric shapes. Ballard observed that the key for generalization was the use of directional information, and that this could be best described in form of *R-tables*. Accordingly, in the generalized Hough Transform (GHT), every point (x, y) in the contour of the shape has a gradient ϕ and can be described by a magnitude (r) and an angle (θ) with respect to a reference point (x_c, y_c), usually defined as the centroid of the shape. In this way, the information of every contour pixel can be indexed under discretized gradient values ϕ_i in an R-table, as depicted in Figure 2.5. Thus, the resulting R-table fully characterizes the particular shape and allows recomputing the location of the reference point during matching.

The actual shape matching consists of 4 steps: (*1*) find a gradient ϕ_i for each contour pixel (x, y) in the image, (*2*) retrieve from the R-table all (r, θ) values indexed under ϕ_i, (*3*) compute for every (r, θ) all possible candidate reference points (x_c, y_c) by using equations 2.2.3 and 2.2.4, and (*4*) increment the corresponding entries in the accumulator array $(P[x_c][y_c])$. As a result of this process, possible instances of the shape can be localized by finding the maxima in P.

$$x_c = x + rcos(\theta) \tag{2.2.3}$$

$$y_c = y + rsin(\theta) \tag{2.2.4}$$

To allow for the identification of rotated and scaled shapes, the GHT is extended to the general case of rotation and scaling. Supposing that the shape has undergone a rotation ω and a uniform scaling s, candidate reference points are computed by varying ω and s in accordance with equations 2.2.5 to 2.2.8, and incrementing the respective entries of the new accumulator array $(P[x_c][y_c][\omega][s])$. Correspondingly, the maxima of P reveal the possible instances of the shape as well as the respective rotation angles and scaling factors.

$$x_c = x - (r cos(\theta) cos(\omega) - r sin(\theta) sin(\omega)) s \tag{2.2.5}$$

$$y_c = y - (r cos(\theta) sin(\omega) + r sin(\theta) cos(\omega)) s \tag{2.2.6}$$

$$\omega_{min} \leq \omega \leq \omega_{max} \tag{2.2.7}$$

$$s_{min} \leq s \leq s_{max} \tag{2.2.8}$$

2.2.3.2 Geometric matching (GM)

Geometric matching (GM) refers to those geometry-based methods applied for locating regions in greyscale images that match a determined template of a reference pattern. These methods are capable of finding template matches regardless of lighting variation, blur, noise, occlusion, and geometric transformations such as shifting, rotation, or scaling.

Geometric matching algorithms search for instances of a template in an inspection image and calculate a score for each found match. Score values indicate how closely the template resembles the matches based on the comparison of defined geometric features or properties. Properties can range from low-level features such as basic edges and curves, to high-level features including spatial relations described by the curves in the image [National Instruments, 2015]@.

Based on the type of used geometric features, GM methods are classified in edge-based and feature-based methods. The main difference among both classes is how the curve information is used during matching; while the former class relies on the calculation of gradients and their relative position to a reference point (see GHT → 2.2.3.1), the latter is based on the analysis of spatial relationships described by geometric shapes (e.g., lines, rectangles, and corners).

Edge-based methods perform well for any arbitrary template, whereas feature-based methods are only suited for those templates that can be consistently and reliably represented by defined shapes. As a consequence, edge-based methods generally yield more reliable recognition results than their counterpart, although their memory consumption and execution time may be significantly higher.

2.2.3.3 Shape detection

Typically based on Hough Transform (→ 2.2.3.1), shape detection is a family of methods designed for locating parametric shapes (e.g., lines, ellipses, circles, and rectangles) in grayscale images.

Unlike GM methods (→ 2.2.3.2), shape detection algorithms do not require the definition of templates for matching. Instead, the underlying matching concept is based on the definition of analytical equations for the representation of geometric shapes. Accordingly, this family of methods is highly suitable for localizing parametric objects with non-uniform scaling, for which several templates would have to be defined if geometric matching methods were applied.

2.2.3.4 Morphological operations

Morphology refers to those digital image processing operators based on the notion that images consist of structures which may be unambiguously described by set theory. Morphological operations apply structuring elements to an input image, creating an output image in which the value of each pixel is based on the comparison of the corresponding pixel in the original image and its neighbors.

Typical applications of morphology include noise reduction [JSB08], shape detection [TB02], and characters segmentation [NYK+05].

The most basic morphological operations are those of erosion and dilation. Dilation adds pixels to the boundaries of objects, while erosion removes pixels in such regions. The number of pixels added or removed from an object depends upon the size and shape of the structuring element [Mathworks, 2016]@. The combination of the dilation and erosion operators originates two additional morphological operations namely, opening and closing. Being A an image, B a structuring element, and B_x the translation of B by the spatial vector x [TB02], the aforementioned operators can be mathematically expressed as shown in Table 2.1.

Table 2.1: Morphological operations

Operation	Mathematical Expression
Dilation	$A \oplus B = \delta_B(A) = \{x \mid (B_x \cap A) \neq \emptyset\}$
Erosion	$A \Theta B = \Psi_B(A) = \{x \mid B_x \subseteq A\}$
Opening	$A \circ B = (A \Theta B) \oplus B$
Closing	$A \bullet B = (A \oplus B) \Theta B$

2.2.3.5 Regular expressions

In computer science and formal language theory, a regular expression, usually referred to as *regex* or *regexp*, is a especially encoded string used as pattern for matching sets of text strings [Fit12]. In general, regular expressions can be thought of as wildcards on text content or as statements designed to refer to determined string targets in a concise and flexible fashion. Each character in a regular expression is either defined as a metacharacter with special meaning (e.g., the whitespace "\s"), or a regular character with literal meaning (e.g., the character "s"). A regex processor translates a regular expression into a finite automaton which is then run on the target text string to recognize substrings that match the given regex. Possible uses of regular expressions include web search engines, lexical analysis, and text filtering in Optical Character Recognition (OCR) applications. Table 2.2 illustrates an example of regular expressions in text filtering.

Table 2.2: Example of Regular Expression (RegEx)

Input String	Regular Expression	Resulting String
Name: George Brown ID: Y08P-522m Post: D-22043	[A-Z?]{1}[0-9?]{2}[A-Z-?]{2}[0-9?]{3}[a-z?]{1}	Y08P-522m

Explanation
[A-Z?]{1} match characters in the range between A and Z (case sensitive) Quantifier: {1} Exactly 1 time **[0-9?]{2}** match characters in the range between 0 and 9 Quantifier: {2} Exactly 2 times **[A-Z-?]{2}** match characters in the range between A and Z, "-" included (case sensitive) Quantifier: {2} Exactly 2 times **[0-9?]{2}** match characters in the range between 0 and 9 Quantifier: {3} Exactly 3 times **[a-z?]{1}** match characters in the range between a and z (case sensitive) Quantifier: {1} Exactly 1 time

2.3 State of the art

Current approaches to graphics recognition in legacy documents can be classified in statistical methods (herein referred to as raster-based approaches) and structural/syntactical methods (herein referred to as vector-based approaches) [WZY07]. The former apply statistical techniques in raster images for extracting pixel-level features and derive thereby the underlying structural and connectivity information described by the analyzed document (e.g.,[Yan05, GV09, YWSG95]). The latter, on the other hand, use syntactical methods to recognize structural components (e.g., primitives patterns) describing structural and connectivity artifacts in vector-based documents (e.g.,[YW03, LWJ04, XZB+04]). This section is dedicated to present the state of the art of both raster-based and vector-based approaches to digitization of legacy engineering documents.

2.3.1 Raster-based approaches

State-of-the-art methods for analysis of raster-based plant documentation have largely focused on the conversion of text-based records. Typical examples of such documents include equipment data, work instructions, and operating manuals [RadialSG, 2016]@. The field of computer vision dedicated to the analysis of text is Optical Character Recognition (OCR). OCR methods aim at the conversion of scanned or photographed characters into machine-encoded text. Identified alphanumerical data can be subsequently stored in databases for further use [HLK99] or exported as digital documents which can be electronically edited and searched.

The complexity of text digitization increases when considering the existence of graphics within analyzed documents. A number of authors have reported on extraction of text annotations from mixed text-symbol documents. Fletcher et al. [FK88] and Chowhury et al. [CMDC07] proposed methods for text string separation in images with objects and annotations in multiple alignments. Chai and Dori [CD92] and Gao et al. [GTLT95] introduced approaches for the identification of string boxes in complex engineering diagrams. Wenyin [WD98] and Adam et al. [AOC+00] presented methods for the recognition of text and basic parametric symbols. Several authors addressed the recognition of text identifiers from engineering drawings [Lu98], many of them using this procedure as a previous step for graphics vectorization (raster-to-vector conversion) [HF94, SSC+00].

Other works have focused on the specific analysis of symbols, yielding thereby significant contributions to the field of Optical Symbol Recognition (OSR). Important research on this area has been carried out in mechanical engineering, where several methods for the conversion of design drawings have been proposed. Surveys on existing approaches are presented in [Hen14] and [Tom98]. Most of those works concentrate on the vectorization of graphic forms to generate 2D or 3D graphical models [VT92a]. Yet, little emphasis has been given to information beyond pure geometric features, such as the domain-specific meaning of identified artifacts and their relations.

Ah-Soon and Tombre [AST95], for instance, proposed a method based on the combination of geometric and semantic information for the reconstruction of 3D CAD models from mechanical drawings. The semantic information used in this approach is, however, limited to verify that the symbols on the different views of the reconstructed object reference the same type of entity, and not to describe specific semantic properties of the analyzed objects.

Focusing more on the analysis of engineering diagrams Yu, Samal and Seth [YSS97] presented a system for recognizing a broad range of documents including logic schematics, electrical circuits, and P&IDs. The proposed recognition process is based on edge detection methods and it comprises

two stages: (a) segmentation based on domain-independent rules, and (b) recognition of graphics and error handling through the application of domain specific knowledge. Their approach has two drawbacks: the identification procedure supports neither OCR nor key geometric matching features such as scaling, rotation, and occlusion (i.e., partial overlap) of objects, and as output, the system generates only a plain netlist (text file) containing identified symbols and their connections.

Song, Cai and Cai [SCLC02] proposed a hybrid recognition paradigm combining raster and vector techniques. This method, however, is constrained to graphics identification and does not exploit domain specific knowledge for the interpretation of recognized elements.

Yu [Yu95] presented a method for the automatic understanding of logic circuit diagrams based on a pixel-based attribute graph. The author differentiates between four types of components namely, shape symbols, structure symbols, connecting symbols, and connecting lines. The recognition of shape and structure symbols is done automatically, whereas the identification of connecting symbols is partially manual. Once the aforementioned components have been identified, the remaining components are assumed to be connecting lines, and thus the underlying symbols connectivity is derived. Yet, neither the semantics of identified symbols nor their representation as non-graphic descriptions are addressed in this approach.

Yang [Yan05] introduced a statistical descriptor for the analysis of architectural and electric diagrams. This approach is based on two pixel-based constraints, specifically the minimal length ratio of two vectors (from a pixel to a preassigned reference pixel) and the angle between them. For N pixels, N histograms of each constraint are constructed by using a different pixel as reference point in each iteration. Based on the resulting histograms, two statistical integrated histogram-arrays, one for each constraint, are generated. Integrated histograms are stored in a database and subsequently used for similarity detection. The approach was proven to be suitable for the identification of isolated symbols (i.e., symbols without connections); however, it provides no mechanisms to identify the connectivity among symbols.

Berbar [Ber06] introduced a method for analyzing design drawings, specifically telephone and cable diagrams. The proposed approach is based on both raster and vectorization techniques. No domain-specific knowledge or semantic analysis, which could avoid misidentifications or erroneous associations of text identifiers to symbols, is applied. The contribution focuses entirely on graphics recognition and does not address the representation of the analyzed diagrams as non-graphical digital models.

Gellaboina and Venkoparao [GV09] proposed the use of neural networks for graphics identification. In their approach, graphical objects are binarized, represented as bipolar feature vectors, and subsequently used to train a Hopfield Neural Network which functions as database and recognition engine. For the extraction of the symbols, the authors use overlapping windows, whereas statistical indexes (esp. activity levels, energy, and overlap values) are used for the matching process. This approach was tested for the recognition of symbols in P&IDs. Connectivity detection, however, was not addressed in this work.

In the commercial sector, few companies have developed tools and provide support for the digitization of legacy documentation in process industries. RadialSG [RadialSG, 2016][@], for instance, offers digitization services for engineering documents available on paper. This company focuses on the methodology to scan, store, and display documents but not on the derivation of high-level digital descriptions such as object-oriented models. Their solution –VIEWPORT– implements optical character recognition to extract component lists from P&IDs. Basic optical symbol recognition is

also supported, although the solution is neither capable of determining the connectivity of the plant assets nor deriving structural digital models from the recognized symbols. Other commercial vendors claim to have tools for automatic conversion of legacy P&IDs into computer-interpretable P&IDs. For example, the NextSpace tools (Intergraph PDS P&ID Extractor) [Nextspace, 2015][@] can partially convert AutoCAD drawings into ISO 15926-compliant XML. Likewise, Intergraph [Intergraph, 2016][@] offers a commercial conversion service from AutoCAD drawings to SmartPlant models [Iyu11]. These solutions, however, are not capable of converting engineering documents on scanned media and elementary digital formats such as PDF.

2.3.2 Vector-based approaches

Existing vector-based approaches to graphics recognition in engineering documents have addressed the analysis of both vectorized raster drawings and native vector graphics drawings. The authors of [YSS97], for instance, presented an approach for the vector-based analysis of raster plant schematics. The method starts with a pre-processing vectorization stage, in which lines segments are produced from straight line vectors through piecewise linear approximation. The pre-processing procedure is followed by a phase of segmentation which extracts and differentiates potential symbol segments from connection lines by applying domain-independent rules. Obtained segmentation results are subsequently used for symbol classification based on similarity assessment against domain-dependent libraries containing symbols descriptors as well as descriptors for their components (esp. loops and strokes). The approach was proven to perform well in simple documents, particularly in small logic diagrams. However, the reliability of the method was observed to significantly decrease when analyzing complex documents such as process flow diagrams. Since the approach does not deploy character recognition capabilities, its application requires existing text to be previously removed from analyzed drawings. As ultimate output the system generates a netlist including found symbols, coordinates, and general connections.

Yang et al. [YSL10] presented a method for the semiautomatic detection and learning of symbols in vector graphics architectural drawings. Based on a user-defined library of basic symbols, the proposed method is capable of detecting objects with slight geometric variations. In their approach, symbols are described by geometric features in form of a *Primitive Attribute Information Table* and a *Primitive Relationship Information Table*. Such descriptors are not capable of characterizing symbols containing curves or polylines, fact that hinders their application on the analysis of complex engineering documents where round-shaped elements are abundant. Added to this shortcoming, the approach neither addresses the problems of connectivity detection nor of representation of gathered information as non-graphic descriptions.

In a similar direction, Yan et al. [YW03] proposed a case-based algorithm for the recognition of engineering symbols. In this work the composition of symbols is limited to three types of primitives, i.e., lines, circles and arcs. Considered relations between primitives include intersection, perpendicularity, parallelism, relative size, relative position, start-angle, and end-angle. Derived information is represented by a Spanning Tree (ST), in which vertexes denote primitives, and edges their geometric constraints. The resulting ST is stored in a knowledge database and subsequently used for recognition. Connectivity detection and representation of gathered information are out of the scope of this approach.

Other authors presented methods for the recognition of graphics in miscellaneous text-graphics mixed documents, which in view of their recognition features can be admittedly applied to the

analysis of engineering drawings. Liu et al. [LWJ04], for example, presented an approach for the recognition of graphic objects based on spatial relation trees and intersection-based relations. A similar method was proposed by Xiaogang et al. [XZB+04], who used spatial relation graphs for objects identification. In this approach, five geometrical constraints are used to characterize primitives relations: interconnection, tangency, intersection, parallelism, and concentricity. As most existing recognition methods, these approaches do not address the problem of connectivity detection.

In what concerns character recognition in vector graphics documents, no specific approach has been proposed in the literature. The main reason for this is that VG-documents have the capability of storing characters as text primitives. The recognition of text-characters is thus a simple process that can be accomplished directly by parsing. However, while going through existing engineering documentation, it is possible to find a large number of characters (esp. tag identifiers) stored as combinations of geometric primitives, particularly in PDF documents exported from CAD tools. In these cases, the character recognition process cannot be performed by parsing, which typically results in the impossibility of capturing tags and annotations.

2.4 Drawbacks of the state of the art

Current raster-based and vector-based methods for graphic recognition in engineering documents have several drawbacks which hinder their applicability in industrial practice. Among others:

1. Scarce methods provide a suitable combination of character recognition and symbol recognition techniques, which prevents the automatic association of identified artifacts and their respective text identifiers (tags).
2. A number of existing approaches can recognize basic symbols within design diagrams but are incapable of identifying their specific type of connectivity (e.g., by differentiating interface classes) or even their functional connectivity at all.
3. Only few methods support fundamental geometric recognition features for the identification of mixed text-symbol documents i.e., rotation, occlusion, and scaling.
4. No vector-based character recognition methods exist, which hinders the capture of non-text-characters in vector graphics documents.
5. A small number of approaches exploit domain-specific knowledge and semantic analysis for the enhancement of the identification process and the improvement of expression capabilities for model description.
6. Most off-the-shelf solutions focus on scanning, storing, and displaying documents but not on the derivation of computer-interpretable models.

The methods proposed in the remaining of this chapter are aimed at overcoming the aforementioned shortcomings by: (a) supporting the identification of symbols, text, and connectivity; (b) deploying semantic analysis for the enhancement of the recognition process and the improvement of model expression capabilities; and (c) representing captured information as computer-interpretable formats suited for the derivation of plant digital models.

2.5 Proposed methods for engineering document analysis

2.5.1 Problem overview and practical considerations

2.5.1.1 Problem overview

The problem of automatically extracting content from legacy engineering documentation must be addressed, in first instance, by identifying the media commonly used for information storage in process plants. A general examination of existing engineering documents allows identifying paper, scans, and vector graphics formats as foremost legacy plant and process information carriers. As a second step, the embodiments of identified information sources must be examined, so that suitable capturing and analysis methods can be devised. In the specific case of legacy P&IDs and CLDs, two main types of document embodiments can be pinpointed: raster images (for scanned papers) and PDF files (for vector graphics documents).

Based on such findings, this contribution proposes two methods for automatic information extraction: Method 1: Raster-based document analysis, and Method 2: Vector graphics-based document analysis. The former method implements image processing techniques to capture information contained in raster images (bitmaps), while the latter deploys discovery of patterns in vector graphics code to identify underlying graphic and text content.

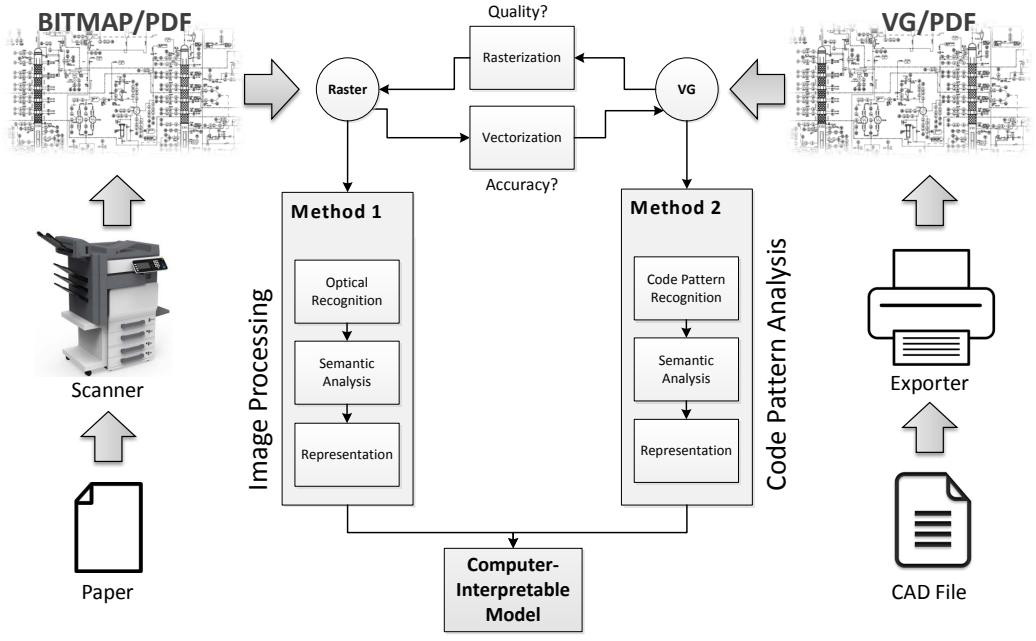


Figure 2.6: Overview of methods for digitization of legacy documentation

As depicted in Figure 2.6, Method 1 allows for the analysis of (scanned) papers and bitmap PDFs, whereas Method 2 is particularly suited for processing digital documents such as vector graphics PDFs or exported CAD files. Both methods have the ultimate aim of generating a computer-interpretable description of the analyzed document and comprise three general steps, respectively: Method 1: Optical Recognition, Semantic Analysis, and Representation; and Method 2: Code Pattern Recognition, Semantic Analysis, and Representation. As shown in the upper part of Figure 2.6, both proposed methods can be potentially interconnected by vectorization and rasterization

techniques, which allows for alternative processing paths as well as for means of cross-validation. For instance, a vector graphics document might be rasterized and subsequently analyzed by image processing techniques (Method 1), and analogously, a raster image might be vectorized and then processed by code analysis (Method 2). Employing such conversion techniques, however, requires the consideration of two key aspects, i.e., (*a*) the minimum image quality/resolution for rasterization, and (*b*) the reliability of the vectorization process based on state-of-the-art methods. These aspects are covered later in this chapter, in Section 2.7.

2.5.1.2 Practical considerations

As pointed out by Yu et al. [YSS97], no automated drawing recognition system is likely to be perfect. Among other factors, noise, poor graphics quality, inconsistent drawing conventions, and unstructured source code affect the reliability of raster-based and vector-based recognition methods. Coping with such difficulties would require considering a never-ending number of document variants and possible inconsistency problems, and accordingly requesting a too large amount of parametrization data from the user. Therefore, this contribution is not intended to present a perfect recognition approach, but to introduce a practical method capable of identifying a large portion of the information contained in analyzed diagrams, and support the user during the identification of eventual inconsistencies by only requesting a little amount of graphic and semantic information.

The proposed recognition methods are especially suited for the digitization of medium to large amounts of documents in projects where a common or similar drawing standard is used. Under these conditions, the time savings yielded by automatic digitization significantly overweight the manual efforts required for the definition or extension of symbol templates and semantic rules at the beginning of the project. In the following, the two proposed methods for analysis of legacy engineering documents are presented in detail.

2.5.2 Method 1: Raster-based document recognition

Method 1, raster-based document recognition, exploits state-of-the-art image processing techniques and semantic analysis for recognizing content in raster engineering diagrams.

As formalized in Figure 2.7, the method relies on the notion that analyzed diagrams are composed of four fundamental artifacts, namely: parametric forms (i.e., simple objects that can be described in terms of parametric expressions such as circles and lines), non-parametric forms (i.e., complex objects composed by multiple forms that cannot be easily described by parametric expressions), text annotations (i.e., characters representing tag identifiers and miscellaneous document annotations), and underlying semantic content (i.e., domain-specific meaning of all the aforementioned graphical components). Accordingly, the approach proposes three main procedures to extract and formalize the content of a diagram:

1. **Low-level recognition** (OSR and OCR) for the identification of both symbols (parametric and non-parametric forms) and text annotations.
2. **High-level recognition** (semantic analysis) for the enhancement of the recognition procedure and the improvement of expression capabilities for model description.
3. **Representation** of the captured information as a non-graphical format aimed at the subsequent derivation of high-level models, specifically object-oriented models.

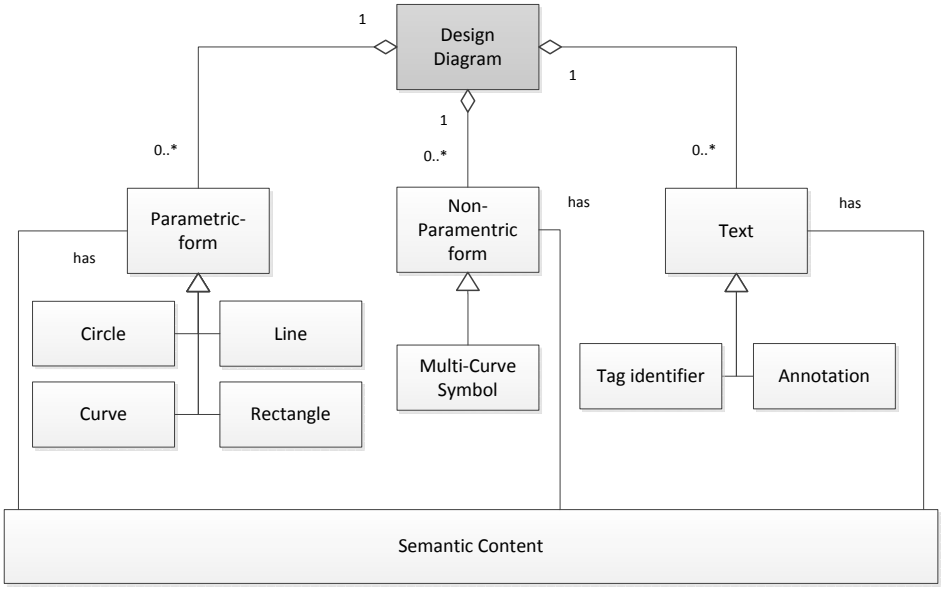


Figure 2.7: Composition of a typical engineering diagram

2.5.2.1 Low-level recognition

In the context of this method, low-level recognition refers to those raster-based techniques used for symbol and character identification. Geometric recognition methods based on Hough Transform and Generalized Hough Transform 2.2.3.1 have been used to implement the required recognition algorithms. The low-level recognition methodology presented in this section is, however, not constraint to a specific image processing technique, and can be, therefore, implemented based on different commercial or proprietary methods according to availability, reliability and performance considerations. In the following, the main steps applied during low-level recognition are presented.

a. Identification of non-parametric symbols

Non-parametric symbols (i.e., multi-curve forms) such as the vessel E100 and the valve V300 shown in Figure 2.8 are localized and matched against pre-defined libraries/databases of templates accounting for typical objects found within analyzed diagrams.

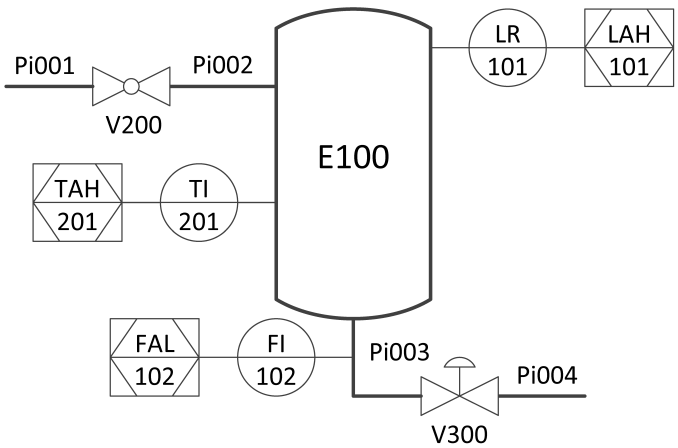


Figure 2.8: Excerpt of a piping and instrumentation diagram (P&ID)

A pre-defined library can represent, for instance, plant asset catalogues or sets of symbols commonly used to embody specific devices, processes, or logical operators. The definition of libraries allow not only for modularity, which in turn results in more efficient search procedures, but also for effective capture and reuse of structural knowledge within different projects.

The creation of library symbols is a simple procedure in which the user must only provide an image containing the symbol to be learnt. From thereon, the algorithm automatically extracts all required matching information and saves this as metadata on the source file. The resulting enriched image is stored in a template library, ready to be used for detection.

The matching criterion employed for non-parametric detection is based on edge properties and resulting geometric features of the examined structural forms. Three key geometric properties (see Figure 2.9) are supported by the geometric matching algorithm (→ 2.2.3.2) to guarantee an exhaustive search of possible matches, namely:

1. **Rotation**: The search algorithm locates possible rotated matches in a range from 0º to 360º.
2. **Scaling**: Possible matches can be larger or smaller than its resembled template as far as geometric proportions are kept. Scaling limits can be suitably adjusted. Typical values are set in the order of 50 % to 200 %.
3. **Occlusion**: Due to overlapping, a given structural form might be partially occluded by other forms. Accordingly, occlusion limits are defined so that matches can be found despite this condition. Typical occlusion values are in the range [0 %, 20 %].

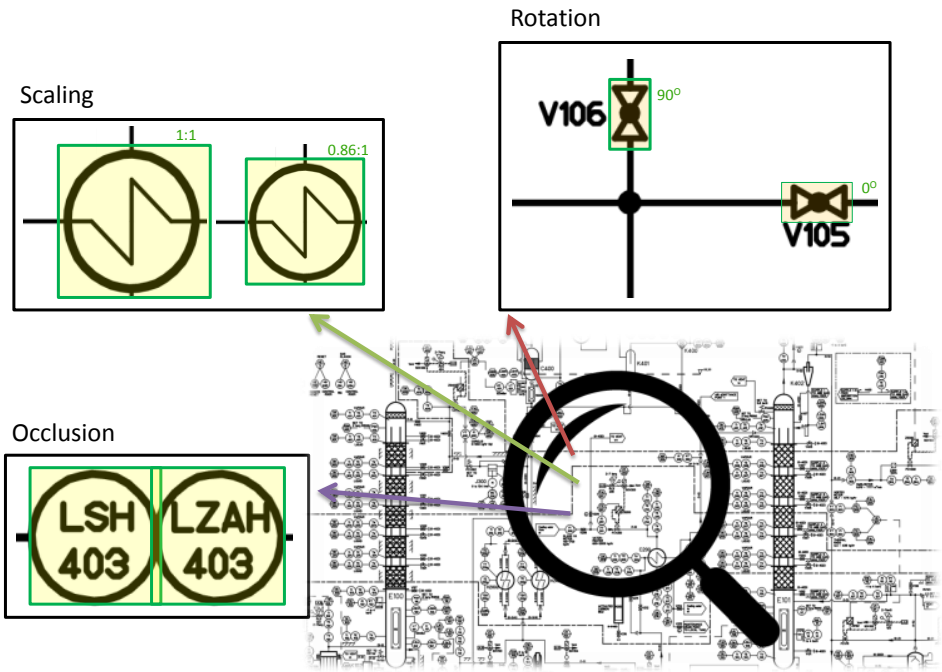


Figure 2.9: Features supported by the geometric matching process

Matching scores quantifying the extent of resemblance between found candidates and respective templates are used to define levels of confidence within the recognition process. Position coordinates of resulting found objects are stored, and their respective curves are suppressed from the image in order to facilitate the recognition of further artifacts.

b. Identification of tag identifiers and assignment to non-parametric symbols

Based on the notion that the tag identifier of an object is located in its surrounding area or within it, the coordinates of identified non-parametric objects are used to generate search windows. The size of a window is typically proportional to the dimensions of the analyzed image or could be conveniently defined by the user depending upon the considered diagram. An OCR algorithm (trained with standard fonts commonly used in engineering diagrams) searches for text within the defined window. The search procedure is enhanced by applying a thinning process based on recursive morphological operations, which reduces noise and allows segmenting touching characters. The process is repeated until one of the conditions described in Table 2.3 is met, in which event the characters corresponding to the current (0) or previous (-1) iteration are accordingly returned. Resulting characters are collected in a string and assigned as the tag of the respective object.

Condition	Stop Criterium	Return
C1	If all detected characters are identified	0
C2	If the number of identified characters is less than in the last iteration	-1
C3	If the number of unknown characters is greater than in the last iteration for the same number of detected characters	-1
C4	If the identification average is less than or equal to the last iteration for an equal or greater number of unknown characters and less or equal detected characters	-1
C5	If all identified characters are unknown after at least 1 iteration has been executed	0

Table 2.3: Enhanced Optical Character Recognition. Iterative thinning process

If more than one tag identifier is found, nomenclature rules are used to discard erroneous strings. For instance, if the analyzed search window surrounds a certain object, possible tag identifiers are to start with a specific alphabetic character X followed by a given number of alphanumerical characters YYY, thus resulting in a tag of type $XYYY$. All strings not satisfying such a pattern are discarded, and, consequently, a unique tag is assigned to the object. In order to facilitate the definition of nomenclature rules, regular expressions ($\rightarrow$ 2.2.3.5) are used here. In case of doubtful identifications, the method triggers a warning notifying the user. As a final step of this process, assigned tags are suppressed from the image to facilitate the identification of remaining artifacts.

c. Parametric recognition

Parametric symbols (i.e., mathematically described forms e.g., circles, lines, and curves) such as the pipeline Pi002 and sensor TI201 illustrated in Figure 2.8 are identified at this stage. Shape detection methods ($\rightarrow$ 2.2.3.3) used within this procedure are capable of finding possible candidates regardless of their size and orientation. Thresholds can be set to define the minimum dimensions of parametric objects and avoid thereby the misinterpretation of characters as symbols (e.g., character “O” as a small sensor, and “I” as a small pipe). Similar to the non-parametric recognition step, the reliability of the recognition process can be adjusted based on calculated matching scores. As a result, position coordinates of found objects are stored and their respective curves are suppressed from the image.

Since parametric recognition does not require templates for object identification, this procedure results highly useful to identify untypical process units represented by blocks. Basically, by combining the identification of blocks (parametric forms) with the semantic analysis of their text identifiers, it is possible to unambiguously recognize the different units without the need to manually define a

template for every of them. This concept is, for instance, exploitable in the analysis of control logic diagrams, where the semantics of certain generic blocks is defined by their tag identifiers (e.g., the AND logical operator is typically represented by a rectangle with the tag "AND" or "&").

d. Identification of tag identifiers and assignment to found parametric symbols

Similar to text identification for non-parametric forms, search windows are generated based on the coordinates of found parametric symbols at this stage. An analogous OCR approach is followed here, i.e., strings are found, rules are applied to ensure nomenclature consistency, and warnings are generated to inform about doubtful identifications. Unlike plant equipment, lines representing pipes are typically not tagged in engineering drawings. In such cases, unique tag identifiers are created and assigned to lines so that these can be subsequently referenced during connectivity detection.

2.5.2.2 High-level recognition

High-level recognition or semantic analysis refers to the incorporation of domain-specific knowledge, particularly functional and structural information, within the recognition of graphical forms and their interrelations. Among other purposes, semantic analysis is exploited for enhancing connectivity detection and model expression capabilities through the definition of specific connection types. The main steps carried along this procedure are:

a. Connectivity detection

Characteristic connections within engineering diagrams are constructed based on polylines arrangements. Hence, connectivity detection requires as a first step to unify adjacent lines so that single linking artifacts can be identified. This step must be applied both for connections formed by continuos and discontinuous (dashed) lines. In the latter case, a maximum distance or gap among line segments (dashes) must be pre-defined to mitigate erroneous detections. During this process, the presence of special symbols (pre-defined templates) such as crossings and intersections is considered in order to determine the correct connection of lines within the diagram. By way of illustration, when a crossing symbol is found, those parallel lines connected to it are unified as single linking artifacts; i.e., the two vertical and two horizontal lines are respectively merged as two independent connections. Once all resulting connections have been identified, the proximity of such artifacts to objects is verified and the underlying connectivity of the diagram recognized. To support the procedure, semantic rules are applied to verify the consistency of found linking artifacts. A semantic rule might state, for instance, that a pipeline can be connected at most to two other elements, one at each termination. For cases violating such a statement, further rules are evaluated so that mistaken detections can be automatically corrected or opportune warnings generated.

The accuracy of the connectivity detection process can be refined by applying further domain-specific knowledge. For instance, knowing that in P&IDs arrows might be incorporated in pipelines to indicate flow directions makes it possible to derive the directional connectivity of the diagram.

Connectivity detection is perhaps the most critical step of the recognition process, as it is sensibly affected by the presence of non previously removed objects and characters. The reason for this is that any object or character can be decomposed in small line segments, which causes the connectivity algorithm to detect undesired lines and derive erroneous connections. In order to reduce the impact of such detractions, line length and width thresholds can be defined, so that specific segments (e.g., short lines forming characters or bold lines forming objects) can be discarded as potential linking

artifacts. Note, however, that such a filtering process is not fully reliable and a verification process may be required, as discussed later in this chapter.

b. Definition of interface types

The generation of detailed computer-interpretable models requires making distinctions between connectivity types. For instance, the semantics (function and physics) of a vessel-sensor link differs from the one of a vessel-pipe connection. The former transports information and is typically implemented by wires, while the latter carries material (i.e., products) and is usually implemented by flow couplings. For several use cases such a distinction is fundamental, e.g., for the derivation of simulation models, where process dynamics are highly dependent on the plant components and their underlying connectivity [YDSC14][AFCH15]*. Accordingly, at this stage, semantic analysis is applied to refine the definition of specific connection classes. Regardless of whether the same type of lines is used to represent different connections within the diagram, the semantics of related symbols (e.g., vessel E100 and sensor LR101 in Figure 2.8) are analyzed so that appropriate interfaces and links are created within the respective derived objects. In case of doubtful connectivity detection, fit-to-purpose warnings are opportunely prompted.

2.5.2.3 Data representation

As a first computer-interpretable representation of the gathered information, the connectivity, positions, and dimensions of found elements are represented in form of an incidence matrix and a coordinates table. As depicted in Table 2.4, the incidence matrix has as first row and column the list of recognized elements, which are referenced to by their type and tag identifier, e.g., vessel E100. The inner entries of the matrix represent the connectivity between elements (row, column). Specific connection types are indicated by distinctive tags, namely “F” for material flow, “I” for information flow and “E” for energy flow. For example, vessel E100 is connected to pipeline Pi002 through a material flow interface and vice versa (see Table 2.4). In cases where the effective flow direction between elements is known, a directional incidence matrix, so called connectivity matrix (CM), can be used to capture directionality information. The CM presented in Table 2.5, for instance, allows not only indicating that the pipeline Pi002 and the vessel E100 are connected, but also that a material flow proceeds from the pipe to the vessel. The coordinates table, in turn, stores the position of found objects as well as their respective dimensions (see Table 2.6).

Table 2.4: Example of incidence matrix

	Vessel E100	Pipeline Pi002	Sensor LR101
Vessel E100		M	I
Pipeline Pi002	M		
Sensor LR101	I		

Table 2.5: Example of connectivity matrix

	Vessel E100	Pipeline Pi002	Sensor LR101
Vessel E100			I
Pipeline Pi002	M		
Sensor LR101			

Table 2.6: Example of coordinates table

Component	Coordinates (x,y)	Dimensions (w,l)
Vessel E100	(150,15)	(90,165)
Pipeline Pi002	(100, 55)	(50,1)
Sensor LR101	(255,15)	(40,40)

Both incidence and connectivity matrices as well as coordinates tables can be codified as spreadsheets (e.g., on MS Excel or C# Windows Forms) and be recursively accessed for the derivation of object-oriented models, as discussed in more detail in Chapter 3. In addition, they might be used for basic tasks such as the generation of part lists or the solution of queries regarding the presence of specifics items or item types.

2.5.3 Method 2: Vector-based document recognition

Method 2, vector-based document recognition[1], exploits a novel approach to the analysis of code patterns for recognizing structural and connectivity content in engineering diagrams stored in vector graphics formats, specifically in Scalable Vector Graphics (SVG). The method is sustained on the notion that characters and symbols as well as their connectivity can be recognized by detecting specific combinations of primitives in the diagram source code. Accordingly, the method proposes three main procedures to extract and formalize the content of a diagram:

1. **Low-level recognition** (code pattern analysis) for the identification of both symbols and text annotations.
2. **High-level recognition** (semantic analysis) for the enhancement of the recognition procedure and the improvement of expression capabilities for model description.
3. **Representation** of the captured information as a non-graphical format aimed at the subsequent derivation of high-level models, specifically object-oriented models.

2.5.3.1 Low-level recognition

As for this method, low-level recognition refers to those vector-based techniques used for symbol and character identification. In the following, the main steps applied during this phase are presented.

a. Preparation

The first step of the method, preparation, is aimed at adjusting and structuring the diagram source code in a form containing only recognition-relevant information. As SVG specifies a broad range of elements (→ 2.2.2.4), graphical symbols in this digital format can be codified by using various primitives or combination of primitives. For instance, a rectangular object might be instantiated either as a rectangle element, four line elements, a polyline element, or a path element. Such variability in the construction of graphical objects hinders the information extraction process, as it implies learning and comparing an unmanageable number of descriptors for object recognition.

Accordingly, the composition of symbols must be constrained by defining a set of primitives complete enough to describe any possible symbol and simple enough to limit the number of object

[1] An extended explanation of this method can be found in [Hoa15]$^{\&}$.

descriptors to a wieldy amount. Thus, the shapes of all objects in the analyzed document must be decomposed or converted into primitives contained in the set. In this approach, the set of primitives is defined as $S_P = \{lines, polylines, circles\}$. These primitives are used to represent respectively straight segments, curves, and circular contours of graphical objects in the diagram.

The next step in the preparation phase is the merging or unification of collinear lines. This aims at reducing the total number of line primitives in cases where straight segments of an object are composed by multiple lines. Two line primitives are merged together if the following criteria are satisfied: (a) they are interconnected[2], (b) they have the same stroke width, and (c) they have the same inclination.

A similar merging step must be taken for polylines, specifically in the case of curves composed by multiple polyline primitives. Two polyline primitives are merged together if they meet the interconnection and width criteria for lines merging (see criteria a and b above), and additionally have the same angle at the interconnection point. For circle primitives no merging step is required.

Note here that although merging criteria of general validity can be defined, the analysis of certain engineering documents might require the definition of further rules based on the consideration of specific recognition cases.

The next preparation step considers the relation between the spatial location of graphical elements within the document and the position of their respective primitives in the SVG code. Such a step is necessary since the structure of the source code does not reflect the spatial distribution of objects, which has a direct impact on the performance of the recognition process. Consider, for instance, a symbol composed by three primitives, the first of which appears at the beginning of the code, and the two others are stored at the end, n code lines below. In this case, the complete code has to be parsed to identify the object, although its primitives are spatially located in the same area and could be accessed faster if their physical location and position in the code would match.

In order to solve such an issue, this approach presents the concept of Primitive Information Matrix (PIM). The PIM is a $M \times N$ matrix, M and N being the maximum y and x integer coordinates of all elements in the document, used to sort and store the information of primitives based on their coordinates. Primitives are stored in the PIM according to the following criteria:

1. Every line and polyline primitive is to be stored in two entries of the matrix: one entry a_{Y0X0} based on the coordinates of its first endpoint, and one entry a_{Y1X1} determined by the coordinates of its second endpoint. In case of containing decimal positions, coordinates must be rounded to the nearest integer in order to determine the respective insertion point. Information stored in both entries is in general identical except for the inverse order of the coordinates (see Table 2.7).

2. Circle primitives are to be stored in one entry of the PIM in accordance to the coordinates of their centers.

3. A PIM entry might contain multiple primitives. Regardless of their type, if two or more primitives are interconnected or in a proximity neighborhood, the respective matrix entry (given by the coordinates of the intersection or proximity point) is to be used to store the respective primitives.

[2]Lines and polylines are interconnected if they have a common endpoint. Circles are said to be interconnected if they share a common center.

Table 2.7: Primitive Information Matrix (PIM)

	x=0	x=1	x=2	x=3
y=0				Line:(PB1, PB2); ID= “B”; Style Information
y=1				
y=2		Line:(PA1, PA2); ID= “A”; Style Information Line:(PB2, PB1); ID= “B”; Style Information		Line:(PA2, PA1); ID= “A”; Style Information

As depicted in Table 2.7, the PIM enables structuring code information in a form reflecting the spatial location of objects in the document. Moreover, it allows retrieving all relevant recognition data without the need to access the SVG code directly.

b. Symbol recognition

This sub-phase is divided in two parts: the first is aimed at the creation of a database of model symbols (templates), while the second corresponds to the actual symbol identification process.

b.1. Symbol database creation

For the creation of the symbol database, model symbols or templates must be, in first instance, extracted from sample documents and subsequently trained. The extraction process can be performed by using a vector graphics editing tool in which the user selects symbols of interest and stores them as single SVG files.

The symbol database is structured in two information levels. The first level stores the Basic Shapes (BSs) of extracted symbols, whereas the second level stores their distinctive features in form of Characteristic Graphs (CGs). Such a storage structure is sustained on the notion that any symbol can be unambiguously described by a BS and a CG.

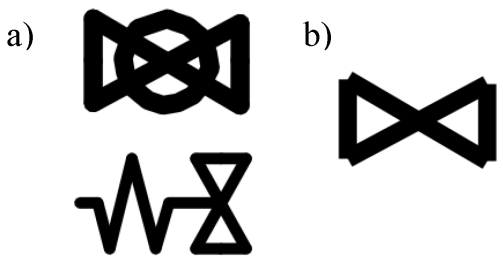


Figure 2.10: (*a*) Two types of valves (*b*) Corresponding BS

Accordingly, the first step for the database creation is the determination of a basic shape for every model symbol. A BS is defined as the largest number of interconnected primitives of a symbol forming a loop. If the symbol has no loops, the open series of primitives with the largest number is regarded as the BS. Figure 2.10a depicts two different types of valves, both characterized by the BS illustrated in Figure 2.10b.

For computation purposes, a BS is described as a set of geometric properties and correspondingly stored in form of a basic shape array (BSA). The quantity of primitives in the BS determines the number of array rows, whereas the amount of geometric properties defines the number of array columns. In this approach, five geometric properties are considered, namely: *Type of Primitive*, *Comparative Length*, *Reference-Angle*, *Start-Angle*, and *End-Angle*.

The row order in the array is determined by the following criteria:

1. Primitives are sorted in ascending row number by length, from the longest to the shortest.
2. If two or more primitives have the same length, the row order is determined by the magnitude of the reference angles α_i, sorting from the largest to the smallest in ascending row number.

Entries in the array columns are defined as follows:

1. Values in the first column are limited to types *line*, *polyline*, and *circle*.
2. Entries in the second column store the comparative length (smaller (<) or equal (=)) of a primitive and the primitive in the previous row of the array. Note here that as rows are organized by descending length, no comparative value larger (>) is possible or required. Notice besides that the first entry of the column is left empty as this primitive cannot be compared with a previous one.
3. Angles α stored in the third column correspond to those described by the projection ($\overrightarrow{R}$) traced from the center of gravity of the BS (COG_{BS}) to the center of gravity of the respective primitive (COG_i), and a vector ($\overrightarrow{L}$) representing the orientation of the primitive element (see Figure 2.11). For line primitives, $\overrightarrow{L}$ corresponds to the vector described by the line itself; for polyline primitives, $\overrightarrow{L}$ is defined as the vector joining the primitive's endpoints; and for circle primitives, $\overrightarrow{L}$ is defined as the vector describing the circle radius in the direction of the axis $x+$. With I being the number of primitives in a BS, equations 2.5.1 to 2.5.4 allow computing the respective angle values.

$$COG_{BS} = \frac{\sum_{i=1}^{I} COG_i}{I} \tag{2.5.1}$$

$$\alpha = arccos\left(\frac{\overrightarrow{L} \cdot \overrightarrow{R}}{\left|\overrightarrow{L}\right|\left|\overrightarrow{R}\right|}\right) \tag{2.5.2}$$

$$\boldsymbol{If} \quad \alpha < 90^{\circ}, \qquad \boldsymbol{Then} \quad \alpha_{new} = 180^{\circ} - \alpha \tag{2.5.3}$$

$$\boldsymbol{If} \quad \left|\overrightarrow{R}\right| = 0, \qquad \boldsymbol{Then} \quad \alpha_{new} = NaN \tag{2.5.4}$$

4. Values in the fourth and fifth columns are exclusively used to describe polyline primitives and contain the *Start-Angle* and *End-Angle* (*Start-Angle* $\geq$ *End-Angle*) of the polyline with respect to $\overrightarrow{L}$. For the rest of primitives (i.e., lines and circles), values in these columns are set to *NaN*.

Figure 2.11 illustrates the BS of the valve shown in Figure 2.10b including reference vectors (represented by green and blue dashed lines) and the COG_{BS} (denoted by the red dot). Table 2.8 shows

the corresponding Basic Shape Array (BSA). Here, the first two rows represent the inclined lines, while the last two denote the vertical lines. Note that the reference angles of the inclined lines are set to *NaN* since the COG_is and the COG_{BS} have the same coordinates (i.e., $\left|\overrightarrow{R}\right| = 0$). In such cases the magnitude of the *NaN* angles cannot be considered as a pattern during symbol identification; instead, the position of *NaN* values in determined cells of the BSA defines the pattern. This allows the method to identify objects with certain geometric variations such as ununiform scaling. By way of illustration, two valves whose inclined lines have different slope (e.g., due to ununiform stretching) have the same BSA, and accordingly can be identified based on the same template.

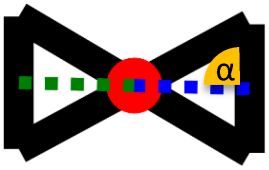


Figure 2.11: Basic shape with reference lines and COG

Table 2.8: Basic shape array of a valve

Primitive Type	Comparative Length	Reference Angle	Start Angle	End Angle
Line		NaN	NaN	NaN
Line	=	NaN	NaN	NaN
Line	<	90°	NaN	NaN
Line	=	90°	NaN	NaN

Once computed, the BSA is classified as *Closed Shape* or *Open Shape* and stored in the first level of the database. Such a classification is aimed at reducing the time required for descriptors comparison during the recognition process, as explained in Section 2.5.3.1.

The next step within the database creation is the derivation of the Characteristics Graphs (CGs) associated to the previously extracted BSs. A CG is a star-shaped graph in which exterior nodes represent the symbol primitives not included in the BS, and the edges denote the geometrical relations between those primitives and a reference (center node) defined as the first primitive of the basic shape array. Figure 2.12b depicts a CG example.

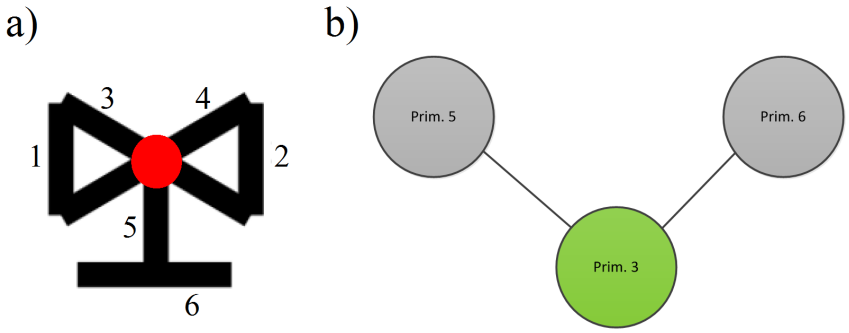


Figure 2.12: (*a*) Valve symbol with COG (*b*) Corresponding CG assuming that Prim. 3 is the first primitive in the BSA

Geometrical relations considered for the definition of CG edges include angles (ϕ_i), relative lengths (l_i), and relative directions (d_i) between the center primitive and the external primitives. An edge is described by the ordered tuple $(\phi_1, l_1, d_1; \phi_2, l_2, d_2)$. Here, ϕ_1 is the angle described by the center

primitive ($\overrightarrow{L}$) and the projection ($\overrightarrow{R_{S1}}$) joining the first endpoint of that primitive and the first endpoint of the respective external primitive; l_1 is the relative length of the projection $\overrightarrow{R_{S1}}$ with respect to the center primitive, i.e., $\frac{|\overrightarrow{R_{S1}}|}{|\overrightarrow{L}|}$; and d_1 is the relative direction of the projection $\overrightarrow{R_{S1}}$ with respect to the center of gravity of the BS (COG_{BS}) with values '1' when the projection is directed towards COG_{BS}, '-1' when the projection is directed away from COG_{BS}, and '0' when the relative direction is undetermined (i.e., when the $\overrightarrow{L}$ vector of the center primitive is in line with the COG_{BS}). The calculation of ϕ_2, l_2, and d_2 is analogously done by considering this time the second endpoints of the center and external primitives.

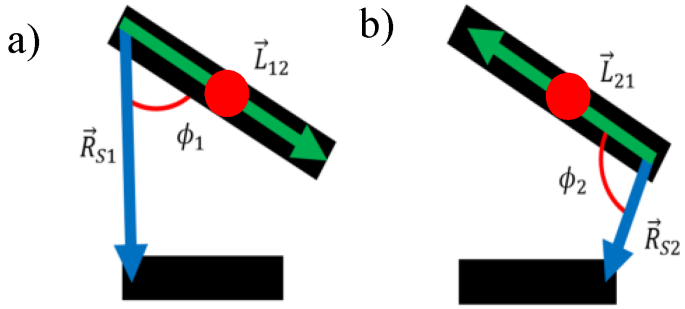


Figure 2.13: (a) Calculation of CG for edge point 1 (b) Calculation of CG for edge point 2

By way of illustration, consider the valve symbol illustrated in Figure 2.12a, and its respective BS depicted in Figure 2.11. Assuming that primitive 3 is the reference primitive of the CG, an edge joining primitive 3 and primitive 6 is constructed. Accordingly, the projection is calculated, as shown in Figure 2.13a. From there, the respective angle and the relative length are computed. In addition, the relative direction is found to be '0' as the center primitive (primitive 3) is in line with the COG_{BS}. Following the same methodology, the respective values for the second endpoints are calculated, as depicted in Figure 2.13b.

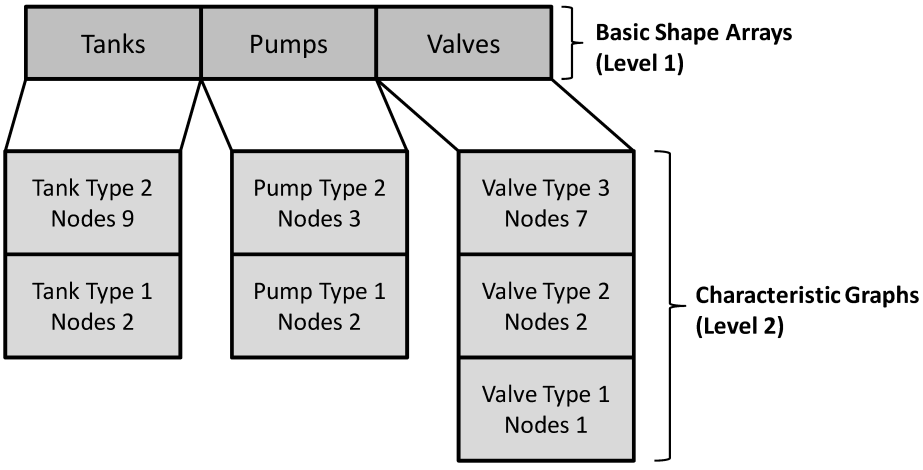


Figure 2.14: General structure of the a symbol database

In case the CG should include polyline or circle primitives, specific geometric features must be added to the respective CG nodes. Concretely, for polylines, the *Relative Length* given by the ratio of the curve length to the distance between its endpoints (magnitude of $\overrightarrow{L}$), as well as the *Start-Angle* and *End-Angle* are stored, whereas for circles, the ratio of the radius to the length of the projection $\overrightarrow{R_S}$ is used to describe the corresponding node.

As last step, the CG is clustered based on its associated Basic Shape Array and stored in the second level of the database. Within a cluster, CGs are sorted by number of nodes in descending order, as illustrated in Figure 2.14.

b.2. Symbol recognition process

The symbol recognition process starts by searching for interconnected primitives in the document and computing potential BSAs. A found candidate is then compared with BSAs in the first layer of the database in search for similarities. If a match is found, the distinctive features of the potential symbol must be examined. Therefore, the information of associated CGs in the second layer of the database is sequentially retrieved and used as pattern for the search of characteristic primitives in the document. Taking as reference primitive the first row of the BSA and considering the edge and node values of the pattern CGs, the coordinates of expected characteristic primitives are calculated and the corresponding cells in the Primitive Information Matrix (PIM) are consulted to verify if the searched primitives exist and have the properties defined in the pattern node. The process evaluates CGs in the database until a complete match is found or all CGs in the cluster have been analyzed.

To enhance the robustness of the identification process and allow for the recognition of symbols with small structural variations, the following tolerance values are defined and considered during the matching process:

1. t_{db}: allowed difference ($\pm$) between the values of a potential BSA and the values of model $\mathrm{BSA_{db}}$s in the database.
2. t_ϕ and t_l: permitted deviations ($\pm$) of the angle and relative length used during the search for potential characteristic primitives.

In the event of a complete match, a green rectangle (bounding box) is drawn around the symbol, and added to the SVG code by using the *<rect>* element. Otherwise, if a partial match is found, the user is warned of a non-fully reliable identification by inserting an orange dotted bounding box, in which case, a recognition score based on the number of matching features is additionally calculated and added to the code as a *<text>* element. As a last step of the process, all primitives corresponding to identified symbols are deleted from the PIM in order to facilitate the recognition of further artifacts, i.e., characters and connection lines.

c. Character recognition

As discussed in Section 2.3.2, VG-characters might be represented in two different ways, i.e., as text primitives or as combinations of geometric primitives. The recognition of the former is straightforward and can be accomplished directly by parsing for text elements in the SVG code. Recognizing the latter, in contrast, requires a detailed analysis of geometric constraints among component primitives and therefore a method for similarity assessment.

Considering the high variability of character shapes, it was determined that the proposed symbol recognition (SR) approach ($\rightarrow$ 2.5.3.1) could not be directly applied for character recognition, as slight font and style variations would require the definition of a very large set of templates for accurate identification. Accordingly, a fit-to-purpose character recognition (CR) method is presented in the following.

As a first step of the proposed method, series of interconnected and connected primitives are searched. In order to differentiate characters from connection lines, the permitted maximal size of searched primitives is constraint to a suitable threshold. Then, a Character Array (CA) is computed for all found primitive series. The CA is similar to the BSA and its number of rows is also determined by the number of primitives in the series. The CA has five columns describing the geometric properties: *Type of Primitive*, *Reference Angle*, *Relative Length*, *Start-Angle*, and *End-Angle*.

The two first and two last geometric properties in the CA are computed in the same way as for the BSA (→ 2.5.3.1). The *Relative Length*, in contrast, is calculated as the ratio of the length of a primitive to the length of the longest primitive in the series.

During matching, each computed CA is compared with trained Character Arrays (CA_{db}s) in the database and accordingly resemblance differences (ν) are calculated. Being, I and J the number of rows and columns in the CA, ${a_{ij}}^{CA}$ and ${a_{ij}}^{CA_{db}}$ the values of the i^{th} row and j^{th} column of the CA and CA_{db} , and K_j a weighting factor defined for every geometric property (column) in the array, equation 2.5.5 allows computing the respective resemblance difference values.

$$\nu = \sum_{i=1}^{I} \sum_{j=1}^{J} \frac{(|{a_{ij}}^{CA} - a_{ij}^{CA_{db}}|)K_j)}{I \cdot J} \tag{2.5.5}$$

The actual character recognition occurs by finding the smallest resemblance differences between CAs and CA_{db}s and retrieving the corresponding character names in the database. The accuracy of the recognition procedure can be adjusted by setting a maximal permitted resemblance difference. Identified characters are subsequently collected into strings based on proximity and nomenclature rules, and ultimately assigned as tag identifiers to previously identified symbols. As a final step of this phase, all primitives corresponding to characters are deleted from the PIM to facilitate the recognition of connection lines.

2.5.3.2 High-level recognition

High-level recognition (semantic analysis) refers to the incorporation of domain-specific knowledge, particularly functional and structural information, within the recognition of graphical forms and their interrelations. Among other purposes, semantic analysis is exploited for enhancing connectivity detection and modeling expression capabilities through the definition of specific connection types. The main steps carried along this procedure are:

a. Connectivity identification

In order to derive the connectivity between diagram symbols, continuous and discontinuous connection lines are to be localized and their respective linkages to objects identified. Presumably, after the symbol and character recognition phases, all remaining line primitives in the PIM represent connection lines. Thus, the coordinates of symbol bounding boxes are used to define areas in the PIM where endpoints of potential connection lines are to be searched. Once an endpoint of a given line has been detected, its second endpoint is checked for interconnected lines by verifying adjacent cells in the PIM (8-adjacency). This process is continued until no other interconnected line is found. As a result, a chain of contiguous lines is obtained and the proximity of the new endpoint to other symbols is analyzed. If a bounding box is found within a defined allowed distance, a connection

between the respective related symbols is established, and a yellow line element is correspondingly inserted in the SVG code to highlight the found connection artifact.

In the event that a line in the searching area has neither an interconnection with other lines nor is in the proximity of other bounding box, it has to be verified whether this is a segment of a discontinuous connection line. For that, the angle of the line is determined and further lines are searched along this orientation. If a line with no interconnections is found in this search, this is to be regarded as a segment of the given discontinuous connection line. The process is repeated until no other segment is found. As a result, a chain of discontinuous lines is obtained. From here on, the process is analogous to the one followed for detecting the connectivity of contiguous line chains.

b. Definition of interface types

Once all connection lines have been identified, their specific types must be unambiguously distinguished. In the particular case of the analyzed diagrams, two differentiated connection types, i.e., information flow and material flow, are considered. Within this step, connection types are derived by analyzing the semantics of the respective connected symbols based on a rule-based approach. By way of illustration, the connection between a pipeline and a valve is of type material flow, whereas the one between a pipe and a sensor or between two control blocks is of type information flow.

2.5.3.3 Data representation

In order to provide a computer-interpretable representation of the gathered information, the connectivity matrix and the coordinates table presented in Section 2.5.2.3 are used respectively to store connectivity and geometrical properties of found elements. As discussed in Chapter 3, such a representation serves as a basis for the derivation of OO plant models exploitable within manifold plant modernization and operational activities.

2.6 Validation

The applicability of the proposed methods[3] has been validated on the examination of a large set of documents stemming both from academic benchmarks and industrial projects. Some obtained experimental results have been previously published and can be found in [AHF15, AHFR16]*. This section is dedicated to present new results obtained within the analysis of existing documentation of the Tennessee Eastman Process (→ 1.4).

2.6.1 Recognition of a piping and instrumentation diagram

As depicted in Appendix A, the TEP P&ID consists of a reactor (R003), a condenser (C115), a separator (S012), a compressor (P100), a stripping column (E005), two pumps (P101, P102), a reboiler (G111), and eleven control valves (V160-V170). In addition, it comprises twelve material flow terminations (e.g., feed "A"), thirty-seven sensors (e.g., TIR100), seventy-two computer functions (e.g., LAH106), and forty-two pipelines. Sections 2.6.1.1 and 2.6.1.2 present the results obtained on the analysis of this schematic with the raster-based and vector-based methods, respectively.

[3]For validation purposes, the raster-based method was implemented on NI LabVIEW by using the IMAQ Vision toolbox (see Appendix D), whereas the vector-based method was deployed on Java by using the Java DOM Parser.

2.6.1.1 Method 1: Raster-based document recognition

The analyzed image of the TEP P&ID was obtained by rasterizing an existing vector graphics version of the schematic with a rasterization resolution of 400 pixels/cm. A template for every diagram symbol was created by using the respective template generation algorithm (→ 2.5.2.1). For the purpose of the test, minimum matching scores were set to 90 % for all parametric, non-parametric, and character recognition algorithms.

Experimental results presented in Figure 2.15 reveal that the raster-based recognition algorithm was capable of identifying the totality of symbols; both plant equipment and flow elements (highlighted with green rectangles) as well as instrumentation devices (depicted with purple rectangles). Likewise, all text identifiers were correctly identified and associated to the respective objects, as depicted by the blue annotations in the figure. All connections between objects with respective type, i.e., information connections (emphasized with discontinuous purple lines) and pipelines (highlighted as yellow lines and tagged with red numbered identifiers), were successfully determined.

During the execution of the recognition process, the algorithm generated one warning, specifically a character recognition warning. As can be observed in the upper part of Figure 2.15, the tag identifier corresponding to the compressor K100 is followed by an exclamation mark ("!"). This is one of the methods deployed by the method to inform the user (or external applications) that partial text was removed from marked tag identifiers as a result of the application of nomenclature rules. A quick

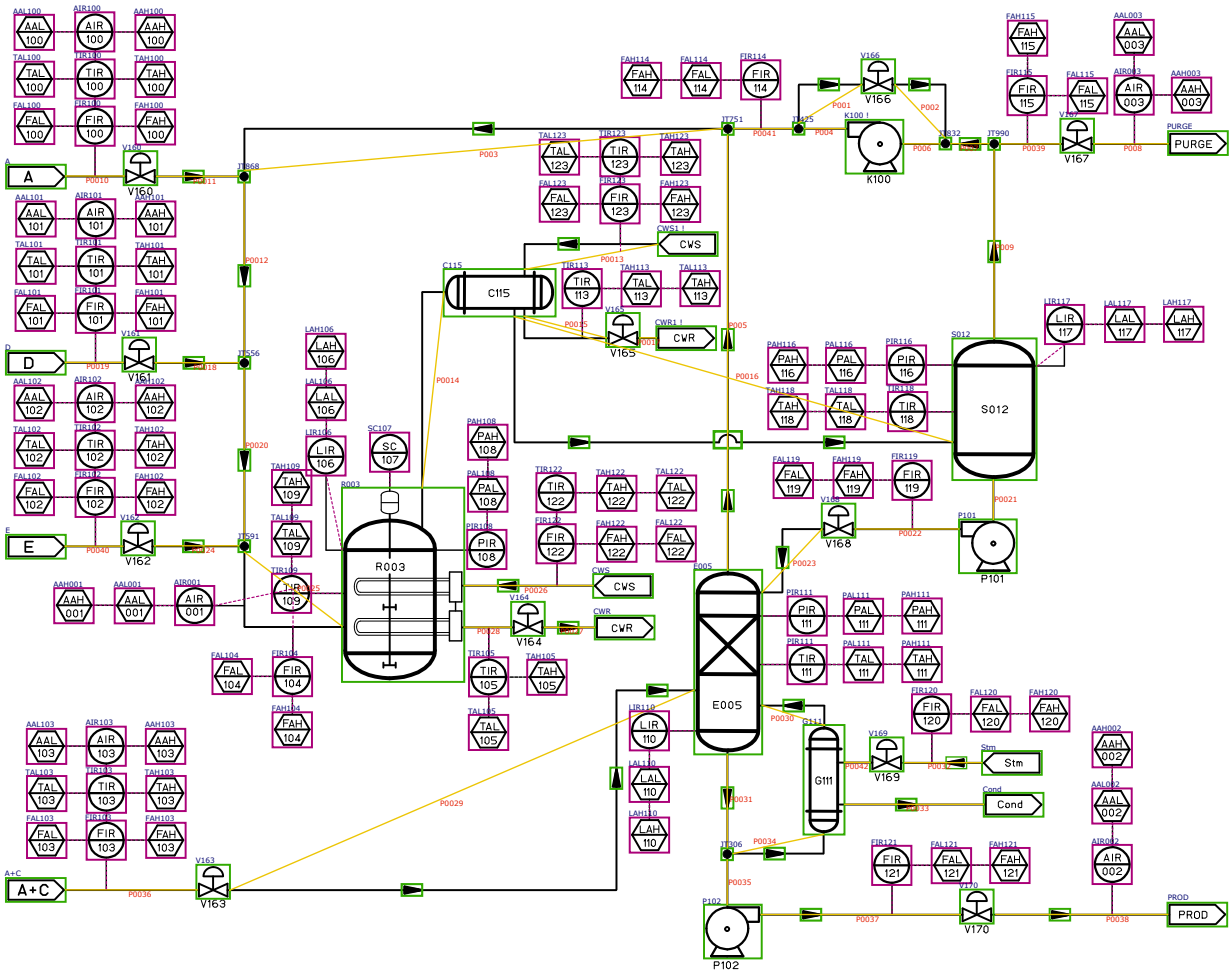


Figure 2.15: P&ID recognized with the proposed raster-based method

inspection of the generated warning allows concluding that the source of the problem is related to an oversize of the compressor text search window (→ 2.5.2.1), which caused that characters of other text identifier (esp. "V166") were initially recognized as a part of the potential object tag. Despite this matter, the algorithm was capable of filtering the erroneous characters and identifying the correct tag, i.e., "K100".

2.6.1.2 Method 2: Vector-based document recognition

The SVG code of the analyzed P&ID was extracted from an existing PDF document. The extraction process was carried out by using the built-in export capability of Inkscape [Bah10]. As for Method 1, a model template for every symbol type was trained and stored in the database following the procedure described in Section 2.5.3.1. In an analogous fashion, character templates were additionally defined. Recognition tolerances t_{db} , t_{ϕ}, t_l and ν were respectively set to 10 %, 1 %, 1 % and 5 %.

Table 2.9: Excerpts of the connectivity matrix (left) and coordinates table (right) obtained in the analysis of a P&ID with the proposed vector-based method

	Reactor R003	Pipeline P0028	Valve V164
Reactor R003		M	
Pipeline P0028			M
Valve V164			

Component	**Coordinates (x,y)**	**Dimensions (w,l)**
Reactor R003	(1899,2653)	(698,1094)
Pipeline P0028	(2597,3443)	(264,0)
Valve V164	(2863,3297)	(186,191)

Experimental results obtained in this test demonstrated that the vector-based algorithm was also capable of recognizing the totality of trained symbols (equipment and instrumentation), as well as all characters (text identifiers) and connections with respective type. Such results are fundamentally the same as those obtained with Method 1 (see Figure 2.15) with the only difference that no warnings were generated during the recognition process. Table 2.9 illustrates excerpts of the respective connectivity matrix and coordinates table obtained in this test.

2.6.2 Recognition of a control logic diagram

The examined CLD is a representation of the TEP regulatory control structure proposed by Luyben [Luy96]. As can be observed in Appendix B, the schematic comprises twelve controllers (e.g., F102), eleven sensors (e.g., FIR101), one calculation block (CALC), and ten actuators (e.g., V161). Sections 2.6.2.1 and 2.6.2.2 summarize the results obtained in the analysis of this diagram.

2.6.2.1 Method 1: Raster-based document recognition

The CLD image was extracted from a rasterized PDF using the same resolution as for the P&ID trial described in Section 2.6.1.1 (400 pixels/cm). A single template was created for every diagram symbol with exception of the calculation block, which as a parametric form does not require a template for identification. Minimum matching scores were set to 90 % for all parametric, non-parametric, and character recognition algorithms.

The results obtained in this trial also demonstrated the robustness and accuracy of the raster-based method, which once again recognized all symbols, tags, and connections within the analyzed schematic; in this case with no warnings.

Table 2.10: Excerpts of the connectivity matrix (left) and coordinates table (right) obtained in the analysis of a CLD with the proposed raster-based method

	ApcFunc CALC	Controller F102	Valve V162
ApcFunc CALC		I	
Controller F102			I
Valve V162			

Component	Coordinates (x,y)	Dimensions (w,l)
ApcFunc CALC	(2866,1469)	(459,604)
Controller F102	(3723,2144)	(1366,1052)
Valve V162	(5320,2733)	(500,457)

A test of particular interest during this experiment was the identification of connection types, specifically those related to actuators (valves). Recalling the recognition of a P&ID in Section 2.6.1, valves, as flow objects, were linked to other components through *MaterialFlow* connections (see Figure 2.15 and Table 2.9). In CLDs, however, valves connect to control elements through logical linking artifacts, and accordingly their connections are to be classified as of type *InformationFlow*. The method successfully passed the test: without explicitly specifying the class of document (i.e., P&ID or CLD), the connectivity detection algorithm was capable of automatically inferring the semantics of *controller-actuator* links and correspondingly creating the required *InformationFlow* connections. These can be observed as "I" entries in the excerpt of the CM presented in Table 2.10.

2.6.2.2 Method 2: Vector-based document recognition

Analogous to the P&ID recognition trial (→ 2.6.1.2), the SVG code of the analyzed CLD was extracted from an existing PDF document. Following the procedure described in Section 2.5.3.1, symbol and character templates were trained and stored in the database. As for recognition tolerances, t_{db} , t_{ϕ}, t_l and ν were respectively set to 10 %, 1 %, 1 % and 5%.

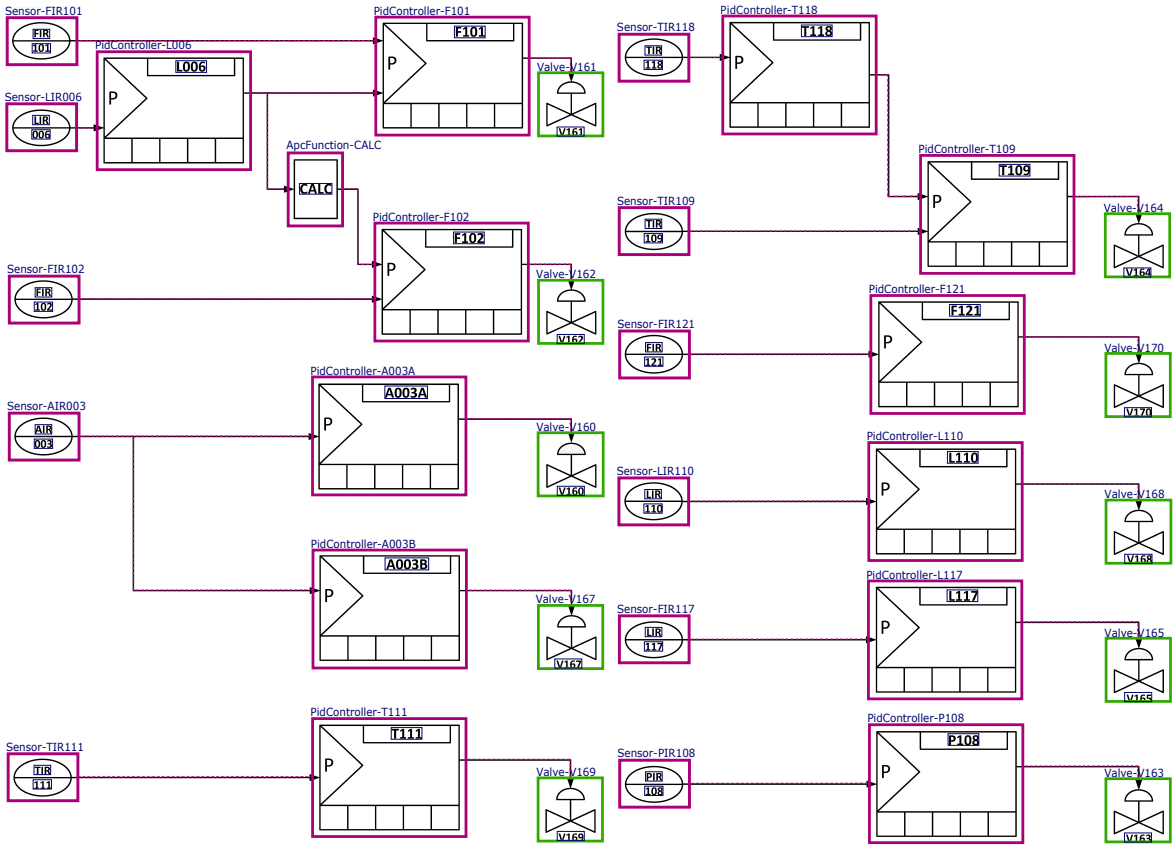


Figure 2.16: CLD recognized with the proposed vector-based method

Figure 2.16 presents the recognition results obtained in this trial. As can be observed in the diagram, all symbols (instrumentation and equipment), text identifiers, and connections were properly recognized. Likewise, the semantics of found linking artifacts were correctly inferred and the respective *InformationFlow* connections identified (see purple doted lines). Note here that unlike pipelines in a P&ID, *InformationFlow* connections are not assigned a numbered tag identifier. The reason for this is that such linking artifacts are regarded as logical connectors and not as physical objects and, accordingly, are not explicitly modeled as topology elements. This can be noticed, for instance, in Table 2.10, where the connection of the ApcFunction CALC to the controller F102 is realized without intermediary elements.

2.7 Assessment

Section 2.6 demonstrated the applicability of the proposed methods on the analysis of existing documentation of the Tennessee Eastman Process. As observed in the four conducted trials (→ 2.6.1.1 to 2.6.2.2), both methods yielded accurate results in the recognition of symbols, text identifiers, and connectivity within the analyzed P&ID and CLD. These results are in line with those obtained in previous experimental tests (see [AHF15, AHFR16]*), which provided evidence of the robust performance of the methods on the examination of several plant schematics, and in addition revealed significant benefits of their use on digitization projects.

As discussed in Section 2.5.1.1, the introduced raster-based and vector-based methods allow the processing of both scanned and digital documents. This enables the possibility of automatically capturing and formalizing data from a large number of legacy engineering drawings, as well as providing means for cross-validation. This has been demonstrated in conducted trials, where the high similarity of the experimental results yielded by both methods on the analysis of different document sources (i.e., raster images and SVG source codes) is proof of a reliable recognition process. In the general case, the cross-validation capabilities allowed by the methods facilitate the identification of erroneous results as well as the consolidation of captured information.

2.7.1 Advantages

Compared to state-of-the-art document recognition approaches, the proposed methods offer the following key advantages:

1. They allow the automatic conversion of legacy engineering diagrams into computer-interpretable models.
2. They deploy both raster-based and vector-based recognition techniques for the identification of graphical elements and text annotations, as well as semantics analysis for consistency check and enhancement of representation capabilities (e.g., definition of specific interface and connection types).
3. They implement matching procedures supporting rotation, occlusion, and scaling which allows for exhaustive searches of structural and connectivity artifacts.
4. They are based on the concept of cumulative libraries which allow for flexibility. Libraries are user-defined and can be easily reused, replaced, or updated so that documents stemming from new projects can be effectively recognized.

5. They deploy mechanisms for the automatic warnings generation, thus providing improved reliability and facilitating consistency checking.

6. They provide means for cross-validation, which enables easy verification of model correctness and completeness.

7. Their use does not require high expertise. The digitalization process can be performed by the user without the need of external parties.

2.7.2 Limitations

The previous sections have discussed the manifold advantages offered by the proposed methods. Until this point, however, no discussion regarding the limitations and required manual efforts for their use has been provided. The remaining of this section is accordingly dedicated to this aim.

2.7.2.1 Diagram complexity

As shown in Sections 2.6.1.1 and 2.6.2.1, the raster-based method (Method 1) allows for reliable automatic extraction and formalization of data in documents of fair image quality and moderate diagram complexity. However, the analysis of schematics exhibiting quality detractions, such as noise and touching characters, might lead to false and missed detections. Likewise, an increase of diagram complexity in terms of ambiguous connections, excessive text legends, and manual annotations has a direct impact on the recognition reliability, especially within connectivity detection.

In an attempt to reduce the impact of the aforementioned factors, Method 1 has been provided with an initial filtering module based on morphological operations. Such a filter allows removing an important amount of image noise, thus facilitating the analysis of the diagram. Remaining noise is dealt with by lowering matching recognition thresholds, which lessens the influence of small pixel regions during parametric and non-parametric recognition. Finally, the impact of touching characters is addressed by performing additional recursive morphological operations during OCR. Note, however, that although such procedures allow for enhanced recognition results, a manual inspection and correction process might be still needed.

Experimental results presented in Sections 2.6.1.2 and 2.6.2.2 also demonstrated that the vector-based method (Method 2) performs well within the analysis of SVG documents containing a fair amount of primitives types and basic transformations. Nevertheless, the processing of complex documents comprising large numbers of primitives transformed in nested nodes might affect the required decomposition of structural objects into basic elements contained in the defined primitive set (→ 2.5.3.1). This, in turn, might lead to missed and false detections and thus to a decrease of the overall recognition reliability. Coping with the aforementioned problem is a cumulative process in which further transformation rules must be defined based on observed new cases. Here again, a manual inspection and correction procedure might be required.

All in all, despite the endeavor required for results inspection, a significant reduction of overall time and effort will be still attained by applying the proposed methods. The concrete quantification of such effort and time savings is subject of variable factors including implementation (code optimality), hardware resources, and execution efficiency of required user manual steps. In view of such considerations, no specific saving estimations are provided here.

2.7.2.2 Required user manual steps

Although the proposed methods are automated to a large extent, certain manual steps are required for their application. Most of them, however, are partially supported by the developed algorithms, as explained in the following.

- *Definition of templates*: in the case of the raster-based method, the manual definition of symbol templates is supported by a template creation algorithm, which automatically extracts matching information upon receipt of a symbol image. In what concerns the vector-based method, symbol code templates are isolated and exported by using the in-built capabilities of a commercial vector graphics tool. Exported templates are subsequently analyzed by the developed algorithm, which extracts relevant patterns and stores them in a symbol database. The vector-based method additionally requires the creation of character templates. These are extracted in the same fashion as the symbol templates. Note here that even though the creation of both raster and code templates must be typically carried out in a project basis, the process is cumulative, i.e., defined templates are collected in object libraries and can be reused in further projects.
- *Manual inspection*: by using the generated graphical representation of recognition results (see e.g., Figure 2.15) in combination with warnings prompted during the identification process, the user can prioritize inspection tasks and quickly localize eventual recognition errors. Alternatively, in some cases results can be cross-validated by conducting the recognition process with both methods. Further inspection mechanisms are discussed in Chapter 3.

2.7.3 Industrial applicability

Section 2.5.1.1 discussed the possibility of interconnecting the proposed methods through rasterization and vectorization techniques as a means for enabling alternative processing paths and results cross-validation. As pointed out, employing such conversion procedures requires considering two key aspects, i.e., (a) the required image quality/resolution for rasterization and (b) the reliability of the vectorization process based on state-of-the-art methods. Experimental tests indicate that rasterization is a robust conversion technique with required quality/resolution values ranging from medium to high (300 pixels/cm or higher). Conversely, state-of-the-art vectorization methods were found to lack robustness when analyzing middle-complexity graphic documents. Potentially, such findings place Method 1 on a preferable spot for current industrial application owing to its capabilities for processing both raster and (rasterized) vector graphics documents. Motivated by these potential benefits, this method has been deployed on commercial engineering and control software products of a world-renowned automation manufacturer [ARF+16]*.

A last aspect worth considering towards eventual industrial application is the maturity level of both methods. While Method 1 is based on well-proven state-of-the-art image processing techniques, Method 2 is built upon newly developed code pattern analysis algorithms. Admittedly, the current lower maturity of Method 2 might be reflected on an eventual decrease of recognition reliability within the analysis of complex engineering documents. In spite of this temporary limitation, Method 2 is expected to capture significant attention in upcoming years, when its potential higher maturity together with the enhancement of vectorization methods and the foreseen broader availability of digital files will enable its wide industrial application.

2.8 Chapter summary

This chapter has presented a new recognition approach for the automatic capture and digitization of information contained in legacy engineering documents, specifically in P&IDs and CLDs. The presented approach comprises two main methods, a raster-based method for processing scanned records and rasterized images, and a vector-based method for the analysis of documents in vector graphics formats. Experimental results obtained during the analysis of sample schematics demonstrated a robust recognition performance of symbols, text, connectivity and underlying semantic content, and a consistent digital representation of captured information. Resulting computer-interpretable descriptions in form of connectivity matrices can support the execution of basic engineering tasks, such as the generation of part lists or the solution of equipment-related queries; but most importantly they embody a key data source for derivation of formal digital plant models. The following chapter discusses a formalization concept aimed precisely at the derivation of object-oriented (OO) plants models based on connectivity matrices and coordinates tables.

3. Derivation of OO plant models

3.1 Object-orientation in process automation

As in most IT-related sectors, object-orientation has come into the mainstream of process automation. Proof of this is that most currently available process control systems, engineering tools, and simulation environments are object-oriented.

In the object-oriented paradigm, the fundamental modeling element, the *object*, is defined as an entity characterized by data, in form of attributes; and behaviors, in form of methods [Wei13]. The type of an object is defined by its *class*, understanding an *instance* of a class as a concrete object with attribute values. Classes are related by *inheritance* and *aggregation* concepts of type "is a" and "consists of" [Sch99]. Thus, the OO paradigm allows the intuitive modeling of production and automation systems, in which every system component can be modeled as an object. Take as an example a simple plant asset, a pump. A pump, as an object of the class *Pump*, can be characterized by e.g., its *technology_type* and *power* (attributes), and by its *pumping* and *suction* functions (methods). A concrete pump, say *P101*, can be modeled as an instance of the class *Pump* with attributes *technology_type=centrifugal* and *power=10 kW*. By the same token, the taxonomy and hierarchical relations of a pump can be effectively represented, for instance: a *Pump* is an *Actuator* (inheritance) and a *Plant* consists of *Pumps* (aggregation).

Despite the wide-spread use of object-orientation in process automation, data exchange and interoperability are still open issues in industrial practice. The heterogeneity of information models used in the current software landscape prevents the seamless exchange of data among tools. As pointed out by Drath and Barth [DB11], *existing automation tools have been developed separately from each other and are therefore not designed to interoperate, to exchange data or to keep common data sets consistent*. Even though certain tools might use the same classes, e.g., *Pump* and *Valve*, the attributes of such classes are likely to differ among the respective information models, which makes the tedious definition of complex schemes necessary for data exchange. This limitation has particular repercussions on the efficiency and costs of engineering processes, where required activities take place within different departments by using an extended range of tools and dissimilar data formats [SDS08].

In an effort to cope with such difficulties, significant research has been conducted in the area of specification and exchange data formats [SGC+99, Dra05, BSF+09, BDSS15]. As a common conclusion drawn from obtained results, most authors agree that as a fundamental requirement for agile data exchange, information should be stored following the OO paradigm and in vendor-independent file formats. Such a provision has been considered within the development and continuous improvement of data exchange standards including ISO 15926 [ISO 15926-1, 2004]#, CAEX according to IEC 62424 [IEC 62424, 2008]#, and AutomationML based on IEC 62714 [IEC 62714, 2014]#.

Aligned with this basic requirement, this chapter presents a method for the OO modeling –based on IEC 62424– of the information captured by the proposed recognition methods (→ 2). The ultimate goal of this formalization step is to enable external algorithms to effectively and unambiguously retrieve captured structural and connectivity information as a basis for different AoA use cases such as automatic generation of simulation models and plant abnormal behavior analysis.

The rest of this chapter is organized as follows: Section 3.2 introduces relevant object-oriented (OO) markup and data exchange formats currently used in the process industry. Section 3.3 reviews previous approaches to automatic generation of OO plant models from existing engineering data sources. Based on the identified drawbacks of the state of the art, a novel method for the automatic derivation of AutomationML plant and process models using basic digital plant representations (connectivity matrices and coordinates tables) is presented in Section 3.4. The applicability of the method is tested in Section 3.5, whereas Section 3.6 analyzes the main obtained results. At last, a brief summary of the chapter is presented in Section 3.7.

3.2 Theoretical framework

3.2.1 Object-oriented markup and data exchange languages

3.2.1.1 XML

Extensible Markup Language (XML) is a simple and flexible text format derived from the Standard Generalized Markup Language [ISO 8879, 1986]#. Originally designed to meet the challenges of large-scale electronic publishing, XML plays nowadays an increasingly important role in the exchange of a wide variety of data on the Web and in other IT-related fields [W3C, 2015a]#.

In the process automation domain, for instance, XML has been used as the core of the following data formats [Bar11]: BatchML [IEC 61512, 1997]#, CAEX [IEC 62424, 2008]#, AutomationML [IEC 62714, 2014]#, Degussa PlantXML, FDT XML, OPC XML, PandIX, PLCOpen XML, eCl@ss [ISO 13584-42, 2010]#[IEC 61360, 2002]#, Simatic ML and XMpLant [ISO 15926-1, 2004]#.

In the frame of this contribution, special emphasis is set on CAEX and AutomationML as the metamodel and data exchange format used for the storage of OO plant topology models. The reason for this stems from the wide use of these standards as the basis for the development of methods for seamless information exchange among engineering tools in the process industry [SSD+15, BHH+16].

3.2.1.2 CAEX

IEC 62424 [IEC 62424, 2008]# defines CAEX (Computer-aided Engineering Exchange) as a metamodel for the storage and exchange of plant description models. Based on XML, CAEX depicts physical or logical plant components in form of data objects organized and described by five fundamental modeling elements (see Figure 3.1):

- *InstanceHierarchy* (*IH*): Description of a specific hierarchy of components from top-level plant down to single components (*InternalElements*, *IEs*) with interfaces (*ExternalInterfaces*, *EIs*) and relations (*InternalLinks*, *ILs*).
- *SystemUnitClass Library* (*SUCL*): Reusable SystemUnitClasses (*SUCs*) defining component types down to their respective technical realizations.
- *RoleClass Library* (*RCL*): Reusable role classes (*RCs*) for abstract descriptions of component requirements and semantic information.
- *InterfaceClass Library* (*ICL*): Reusable InterfaceClasses (*ICs*) for specifying connection points of *RCs*, *SUCs*, and interface types of *EIs*.
- *Attributes*: Properties for characterizing each previously introduced modeling element.

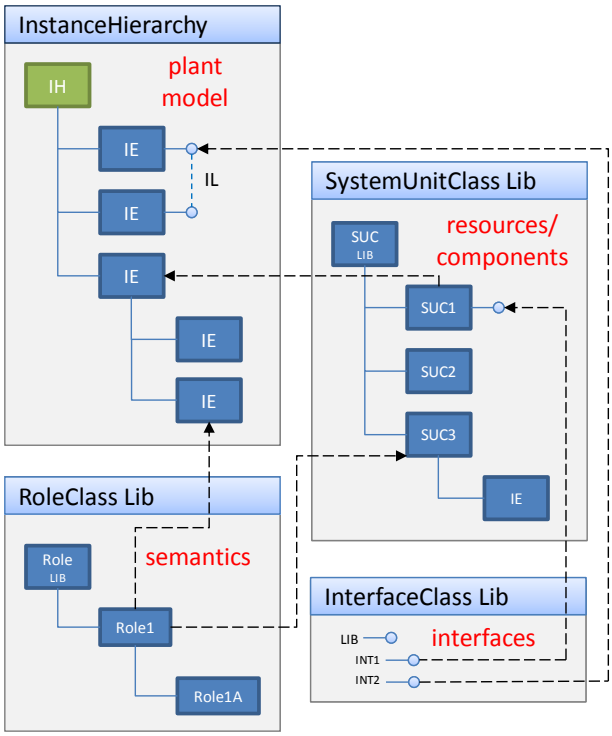


Figure 3.1: CAEX modeling elements and structure

3.2.1.3 AutomationML

AutomationML (AML) is an XML schema-based data format designed for the vendor-independent exchange of plant engineering information [IEC 62714, 2014]#. AML stores engineering data following the OO paradigm and accordingly supports features such as reuse and inheritance. It allows the representation of a wide variety of plant topology-based models encapsulating different functional aspects and offers means for specific data visualization.

The core of AML is the top-level data format CAEX (→ 3.2.1.2), which is used as the backbone structure of plant engineering information. In addition, the schema supports references to other exchange data formats such as COLLADA (geometry and kinematics), and PLCopen XML (programmable logic control).

3.3 State of the art

Within the last decade, the modeling of production systems in data exchange formats has been widely addressed in the literature. From manufacturing cells [LHK10] to process plants [HFB15], electrical systems and process control topologies [MWE+16], virtually every system type has been already modeled. A significant amount of existing modeling approaches have used CAEX/AutomationML as a modeling basis, this due to the semantics flexibility and capabilities for integrating different information sources enabled by this standardized exchange format.

Despite the large number of modeling methods in AML, however, only few authors have explored the automatic generation of models from existing engineering sources. Among them, Barth and Fay [BSF+09] proposed an approach for generating plant models from CAE P&IDs. Hoernicke et al. [HCF14] presented a method for the creation of topology models based on human machine

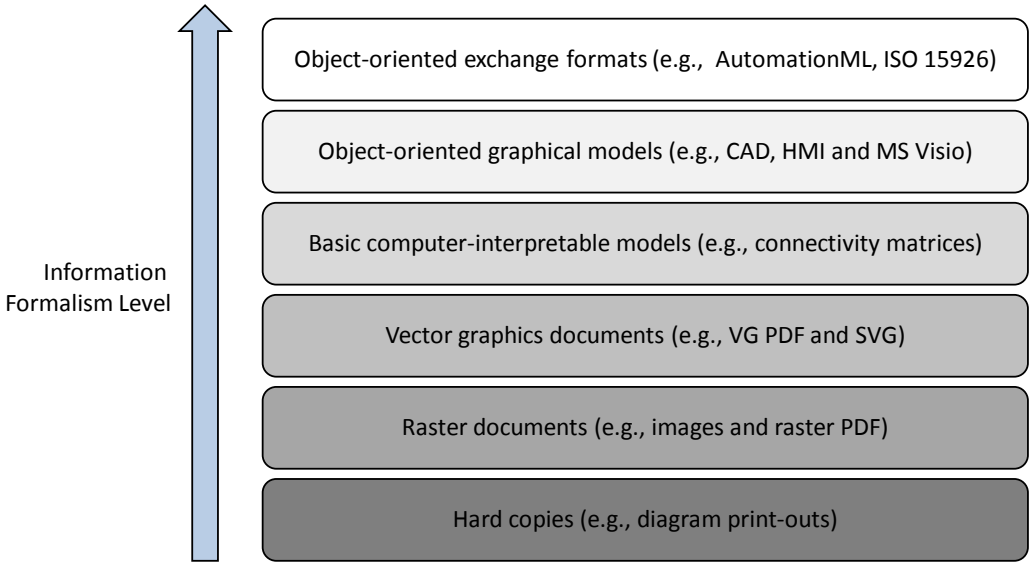


Figure 3.2: Information formalism level of engineering sources

interfaces (HMIs) and Bigvand et al. [BDFR16] investigated means for automatically deriving structural models from .DWG files and other CAE formats. In the commercial sector, Siemens has deployed functions on its CAE tools to export proprietary P&ID formats (COMOS P&ID) to CAEX-compliant representations [Iyu11].

The aforementioned approaches are fundamentally based on the same modeling concept, i.e., mapping information from a source environment to a destination environment. Until now, however, existing methods have only focused on the generation of OO models from OO data sources such as CAE P&IDs and HMIs, i.e., environments with a high level of information formalism (see upper part of Figure 3.2). The derivation of OO models from less formal data sources originally stemming from legacy documentation (see lower part of Figure 3.2) is still a persisting problem, which poses a hurdle for the effective exploitation of existing engineering knowledge in AoA. In an effort to cope with such a limitation, this section presents a modeling approach for the creation of AML plant models from basic computer-interpretable descriptions derived from hard copies and elementary digital engineering documents.

3.4 Proposed method for OO modeling of plant data

3.4.1 OO model generation

Based on the information collected by the document analysis methods (→2.5), OO models (specifically AutomationML models) should be automatically derived from these results. The OO modeling process consists of two main steps namely, (*a*) creating objects accounting for equipment, control, and instrumentation devices, and (*b*) creating interfaces and internal links for depicting the connectivity between objects.

3.4.1.1 Creation of objects

In the first step of the process, an AML object (*InternalElement*) is created for every entry within the first column of the connectivity matrix (CM). Specific object classes (*SystemUnitClasses*) and

roles (*RoleClasses*) are assigned based on a simple lookup table, which maps equipment types in the CM to corresponding classes and roles based on a defined object information model. Consequently, text identifiers (tags) listed in the CM are used to name created instances. By way of illustration, the CM entry "*Vessel E100*" (see Table 2.5) is instantiated as the AML object *Instance=E100:Class=Vessel:Role=Vessel*.

The object instantiation process is enhanced by a rule-based nomenclature approach, which allows increasing the level of detail of object descriptions as well as reclassifying object types based on the etymological analysis of tags. To illustrate, consider the case of an instrument *LR101* which was classified by the recognition method as a sensor. During object-oriented modeling, however, the application of nomenclature rules makes it possible to further specify the component as a *LevelSensor* due to the presence of the character "*L*" in a determined position of its tag.

SystemUnitClasses (SUCs) and *RoleClasses* (RCs) used in this step are based on an extended version[1] of the object information model (OIM) presented in [HMAF16]* and [AHFR16]*. Despite being based on the same OO model, an essential difference between SUCs and RCs is that the latter do not deploy all object attributes defined in the information model, as their function is to depict the roles (and if necessary requirements) of created objects and not to fully characterize them by means of properties.

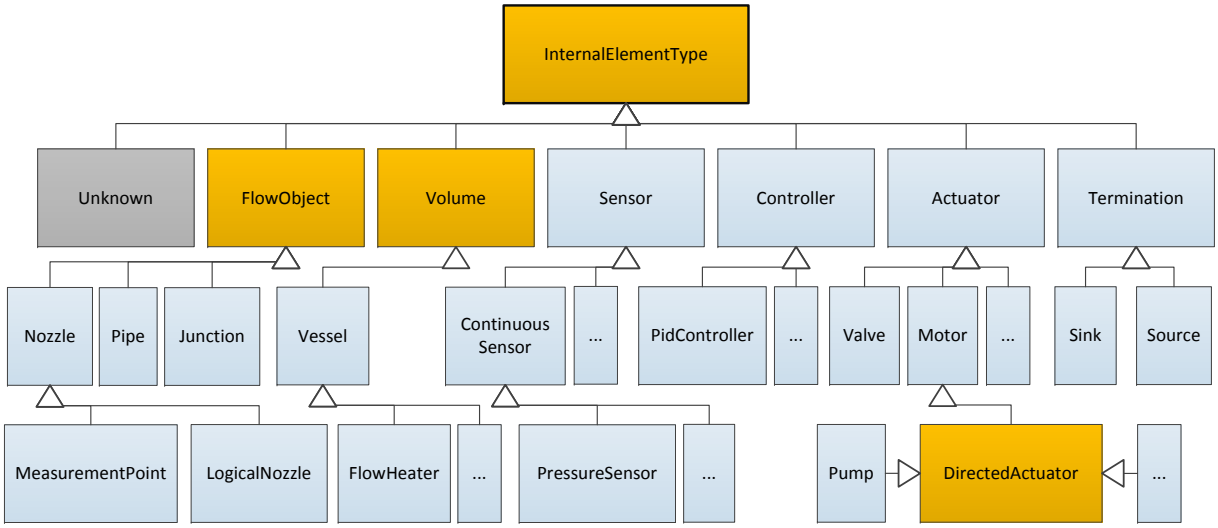


Figure 3.3: Excerpt of the used object information model (OIM). Adapted from [HMAF16]* and [AHFR16]*

The used object information model (see Figure 3.3) has been designed to fulfill two main requirements: (*a*) be generic enough to allow the functional description of any production process in the process industry, and (*b*) be extensible to fit the specificities of particular processes and use cases. Within this model, light blue elements represent common classes of plant components (e.g., *Vessel*, *ContinuousSensor,* and *PidController*), yellow elements indicate abstract classes used for classification purposes (e.g., *FlowObject*), and gray elements designate model leaves, i.e., general classification classes allowing no further derivations (e.g., *Unknown*).

As can be observed in the figure, object types are grouped into seven main classes, namely: *FlowObject*, *Volume*, *Actuator*, *Sensor*, *Controller*, *Termination* and *Unknown*. A brief explanation of each class along with its respective inheritance hierarchy is presented in the following.

[1]The model extension includes new sensor classes as well as additional attributes for some object types.

a. *FlowObject*

The *FlowObject* class comprises those objects involved in the transport of material and information streams along the process. This class include the following concrete types:

- *Pipe*: Objects representing point-to-point connections used to transport material streams between plant equipment. Pipes are typically characterized by geometrical attributes *(length* and *diameter*). Additionally, they contain specific attributes (*FlowDirection* and *MediaType)* for defining the direction and type of transported streams.
- *Junction*: Objects depicting the interconnection of multiple pipes in a given point –typically in form of a T-intersection. Junctions are characterized by a *diameter* attribute.
- *Nozzle*: Nozzles represent internal ports within objects used to store material or information flow interfaces. Their specific function is to characterize the position where a given connection occurs. Except for material connections within *Pipes*, all other connections require nozzles for implementation. Among nozzles, three types can be distinguished:
  - *MeasurementPoint*: Nozzle used to describe the connection of an object to a sensor.
  - *LogicalNozzle*: Nozzle holding the interface connecting information flows among objects. Except for measuring points, all connections involving information flows are described by logical nozzles.
  - *ElectricalNozzle*: Nozzle describing energy connections to producers and consumers.

b. *Volume*

Volume objects are those related to the storage of material in the process. Foremost objects in this class are:

- *Vessel*: Generic volume container (tank) typically characterized by *height, diameter and type* (open or closed) attributes. Among vessels, two specific types are distinguished:
  - *HeatExchanger*: Vessel type used for transferring heat from one fluid to another.
  - *FlowHeater*: Typically a closed vessel with a built-in heating system, which can be parameterized to supply a required nominal temperature.

c. *Actuator*

Actuators are objects that execute control actions and influence thereby the properties of material streams (e.g., pressure and temperature) along the process. Actuator types include:

- *Valve:* Object that regulates, directs or controls the flow of material streams along the process by opening, closing or partially obstructing fluid circulation. Based on its operation mode and connectivity, three types of valves are distinguished here:
  - *BinaryValve*: Valve type with two possible aperture states: open or closed.
  - *ContinuousValve*: Valve type with several aperture states defined in percentage.
  - *ThreewayValve*: Valve type that connects to three pipes instead of two as standard valves.

- *Consumer:* Equipment that consumes a significant amount of energy to perform a control action. Common consumers include:
  - *Motor*: Object type referring to all pieces of equipment that include a motor for operation and are controlled as such. Within the class *Motor,* specific types include:
    - *DirectedActuator*: Motor type used to force a flow into a specific direction. The acting direction is specified by an attribute storing the ID of the nozzle on the pressure side. Directed actuators are classified in:
      - *Pump*: Type of directed actuator forcing a fluid (liquid or two-phase flow) into a given direction.
      - *Compressor*: Type of directed actuator used to force a gas stream into a given direction.
      - *Fan*: Also referred to as a blower, a fan is a directed actuator used to provide a flow of gas (typically air) to the process.
    - *Agitator*: Type of motor typically used to mix liquids in vessels.
  - *Heater*: Consumer type used to heat up fluids.

d. *Sensor*

Sensors are devices used to measure process and equipment variables and transduce observed physical magnitudes into information signals. In this sense, they represent the connection between material and information flows. Sensors are classified into two main types:

- *ContinuousSensor*: Sensor type that captures analog measurements of material or equipment properties. Measured variables are characterized by *unit* attributes, which specify how transduced values must be interpreted. In addition, continuous sensors contain attributes for defining and referencing communication variables as well as for specifying measuring spans and limits. Continuous sensors are classified in:
  - *LevelSensor*
  - *PressureSensor*
  - *TemperatureSensor*
  - *FlowSensor*
  - *SpeedSensor*
  - *ConcentrationSensor*
  - *DensitySensor*
  - *PositionSensor*
- *Switch*: Sensor type used to measure binary variable conditions or states, e.g., open or closed, and high or low.

e. *Controller*

Controllers represent the functions executed by a device to control or regulate a process. Their fundamental aim is to function as intermediary devices between sensors and actuators, providing calculation capabilities within control loops. Two types of controllers are distinguished here:

- *PidController*: Most common type of controller used in closed-loop control. As its name suggests, the proportional-integral-derivative controller performs calculations by applying factoring, integral and derivative operators to a feedback error. The *PidController* class has attributes to reference values for set points (SP), process variables (PV), and outputs (OP). In addition, it contains further attributes to specify controller configuration (single or cascade) and tuning parameters (i.e., gain, reset, and rate values).
- *ApcFunction*: Computer function used to model calculation blocks, alarm functions, and other control elements with calculation functionality.

f. *Termination*

Terminations are objects representing model boundaries. They shall be connected to pipes or other objects to depict external material and information flows. Terminations are characterized by *ModelRef* and *ElementRef* attributes, which are respectively used to store the references to external models and corresponding termination objects where flows originate from or are directed to. Depending upon the direction of the represented external flow, terminations are classified in two types:

- *Source:* Termination type used to model an incoming information or material flow.
- *Sink*: Termination type representing an outcoming information or material flow.

g. *Unknown*

Unknown objects are used to represent those elements for which no other matching type exists in the object information model according to defined mapping tables and nomenclature rules.

3.4.1.2 Creation of nozzles, interfaces, and internal links

Within the second step of the OO modeling procedure, created objects are provided with nozzles (i.e., logical nozzles and measurement points) aimed at characterizing the physical positions where connections occur. Consequently, object interfaces are created within defined nozzles by considering their specific types.

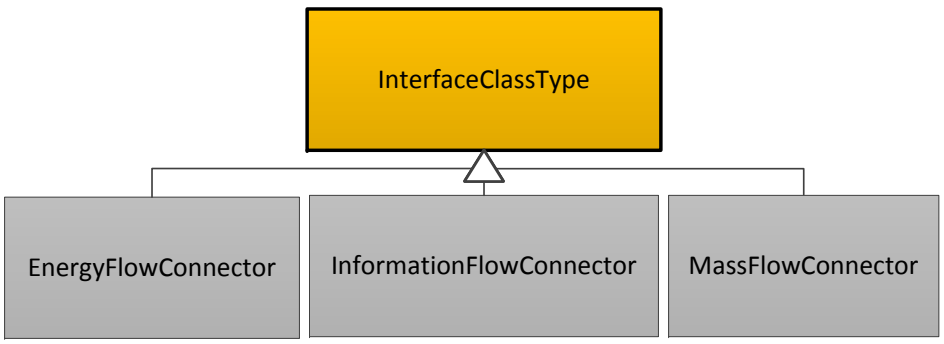


Figure 3.4: Interface information model. Adapted from [HMAF16]* and [AHFR16]*

In order to capture the semantic of the different connection types, three interface classes (*InterfaceClasses*) are defined (see Figure 3.4):

a. ***MassFlowConnector*** **(*M*)**: Interface between two objects which are connected to each other by a material flow. Example: connection between a pipe and a tank.

b. ***InformationFlowConnector*** **(*I*)**: Interface between two objects which are connected to each other by information signals. Example: connections within control loops (sensor, controller, and actuator).

c. ***EnergyFlowConnector*** **(*E*)*:*** Interface between an object and an energy flow. Example: connection of a heater to a source of heat.

The creation of object interfaces considers the number of connections found for every object (i.e., non-empty entries within a row/column of the connectivity matrix). Corresponding interface classes are specified in accordance to the content of the cells. To illustrate, for vessel E100 (see Table 2.5) two interfaces are created, one of type *MaterialFlowConnector* (M) and one of type *InformationFlowConnector* (I).

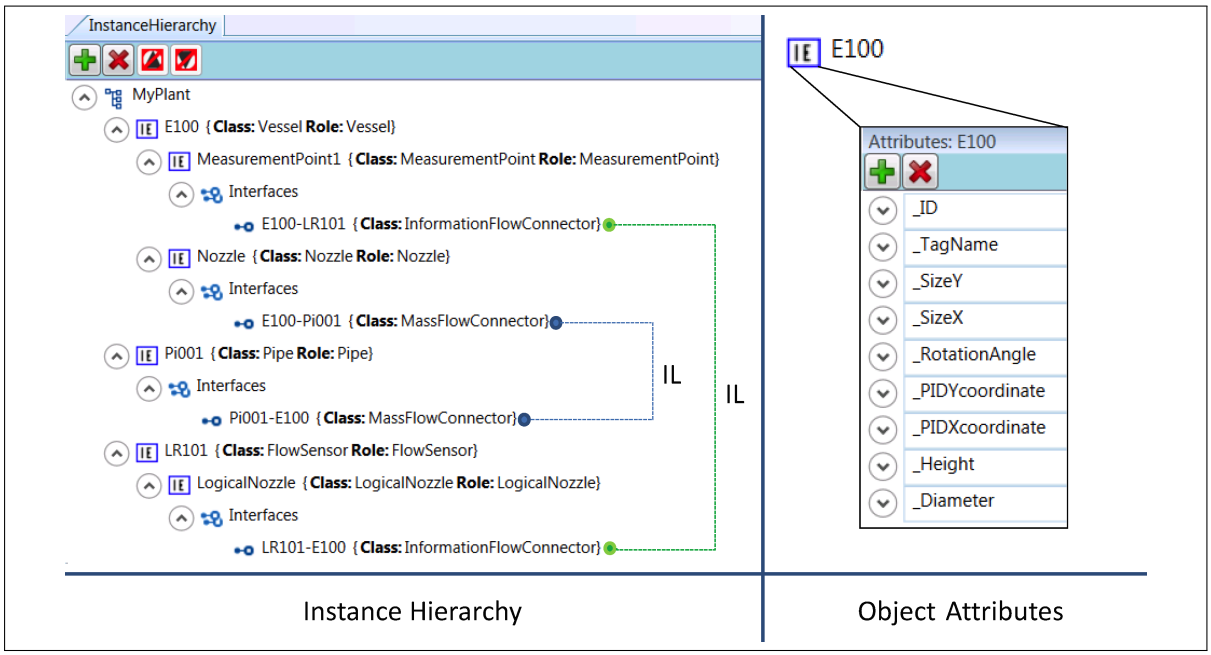


Figure 3.5: Example of OO model in AutomationML

Finally, internal links connecting related objects through consistent interfaces, i.e., interfaces of the same type, are generated (see Figure 3.5 left), and object attributes describing geometrical object properties, i.e., dimensional information collected from the coordinates table, are added (see Figure 3.5 right). As an ultimate result, an OO model describing the connectivity and semantic information of the analyzed document (P&ID or CLD) is obtained.

3.4.2 Visual representation and inspection

In order to allow means of verification and ensure conversion correctness and consistency, a graphical depiction of the model is generated and shown to the user in a topology editor (see Figure 3.6). The graphical representation is overlaid with the original black and white diagram as colored forms

representing the identified objects and their connections. Thus, users can visualize found elements and compare them with the original document, effecting changes where necessary. Corrections carried out in the graphical interface have a direct effect on the OO model. Warnings generated during the recognition process can be used at this stage to prioritize the crosschecking procedure. Moreover, the user can manually add complementary quantitative information to plant assets, such as their type and power, by completing pre-defined object attributes or by creating new ones depending upon the specific information requirements of envisioned use cases.

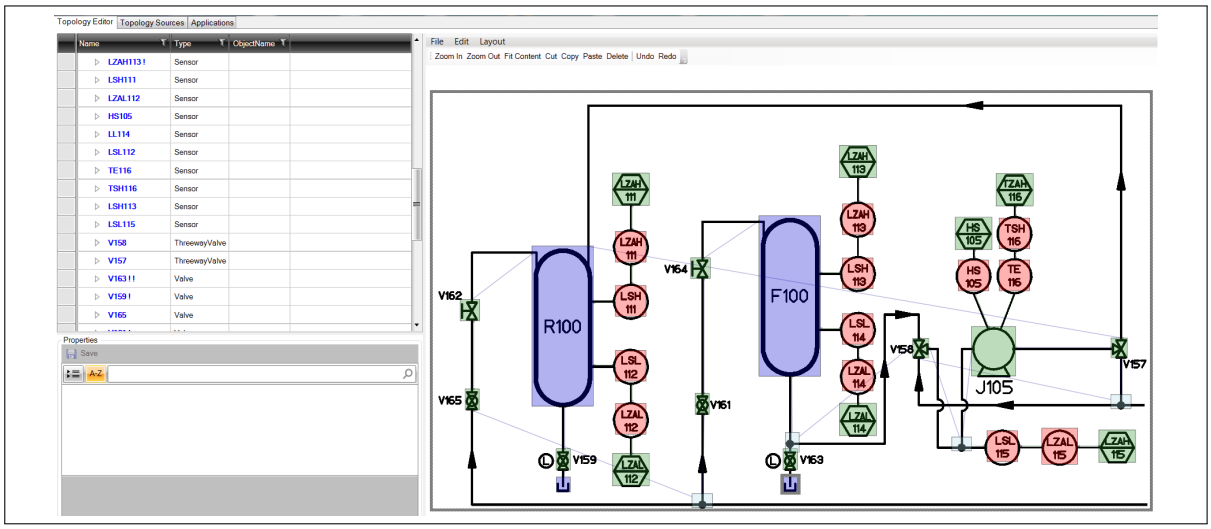


Figure 3.6: Visual inspection process

3.4.3 Inter-model referencing

The method described in the previous sections deals with the generation of OO models from single diagrams, whether they be P&IDs or CLDs. In most projects in the process industry, nevertheless, the complete plant topology and its respective control structures are not depicted as a single schematic, but instead as a series of interrelated diagrams. This implies that in order to create an overall view of the plant, single diagrams must be consistently put together and interpreted as a whole. The same principle applies to single OO models, which in this sense can be regarded as model parts.

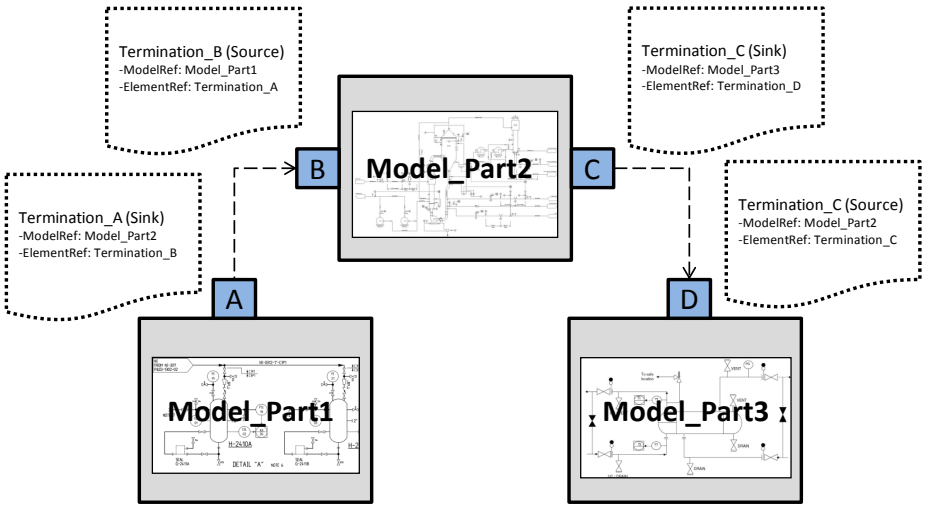


Figure 3.7: Inter-model referencing

With the aim of providing the required means for inter-model referencing, associative properties are created within termination objects representing the boundaries of model parts. Following the approach proposed in [HCF14], the description of such properties is implemented in form of object attributes, which allow specifying the connection of their associated terminations to other terminations (*ElementRef)* in external model parts (*ModelRef*). By way of illustration, consider the schematic presented in Figure 3.7. Here, three model parts (1, 2, and 3) are inter-referenced by source and sink terminations (A, B, C, D) and their respective attributes.

3.5 Validation

For the validation of the proposed modeling approach, the recognition results yielded by the document analysis methods (→ 2.6) were used as data sources. The following sections summarize the experimental results obtained in conducted modeling tests [2].

3.5.1 OO modeling of a piping and instrumentation diagram

Following the procedure presented in Section 3.4.1.1, the OO modeling algorithm parsed the connectivity matrix corresponding to the analyzed P&ID and generated an object, specifically an *InternalElement* (IE), for every found plant component. The mapping scheme used during this step is presented in Table 3.1.

CM Type	**SystemUnitClass**	**RoleClass**
Separator	*Vessel*	*Vessel*
Valve	*Valve*	*Valve*
Reboiler	*HeatExchanger*	*HeatExchanger*
Sensor	*Sensor* (Specific)	*Sensor* (Specific)
Reactor	*Vessel*	*Vessel*
Condenser	*HeatExchanger*	*HeatExchanger*
Pump	*Pump*	*Pump*
Compressor	*Compressor*	*Compressor*
Stripper	*Vessel*	*Vessel*
ApcFunc	*ApcFunction*	*ApcFunction*
Pipe	*Pipe*	*Pipe*
Termination	*Termination*	*Termination*
Intersection	*Junction*	*Junction*

Table 3.1: Modeling look-up table used for P&IDs

As depicted in Figure 3.8, every created IE was assigned a *SystemUnitClass* and a *RoleClass* in accordance to the types originally defined in the object information model. A brief inspection of the created objects indicates that the rule-based nomenclature approach applied during instantiation allowed the successful specification and reclassification of certain object types. By way of illustration, the instruments LIR106 and PIR108 depicted in Figure 3.8 were respectively instantiated as *LevelSensor* and *PressureSensor*, despite the fact of being initially identified as generic sensors by

[2] Due to the large size of the generated OO models, the next subsections present only partial screenshots of the obtained results. The corresponding AML files of the complete TEP P&ID and CLD models can be respectively found at http://aut.hsu-hh.de/TEP_P&ID and http://aut.hsu-hh.de/TEP_CLD.

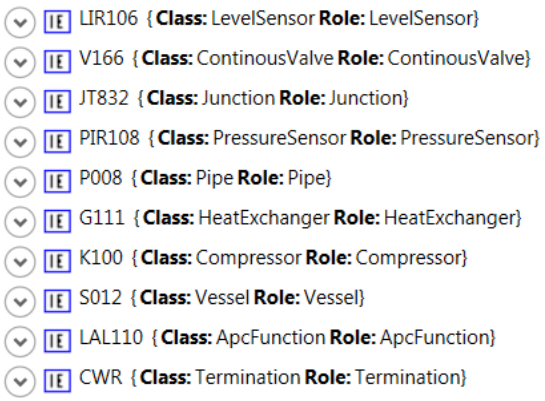


Figure 3.8: Objects within the generated OO P&ID model. AutomationML visualization

the document analysis methods. By the same token, the object K100, preliminarily recognized as a pump, was correctly reclassified as a *Compressor*.

A more detailed observation of the generated model allows confirming that nozzles, interfaces, and internal links were correctly instantiated. This can be observed in Figure 3.9. An important aspect to notice here is the consistency of the established material and information connections. Observe that internal links only connect interfaces of the same type, and that established connections are unambiguously described by the references (esp., universal unique identifiers) of the respective linked connectors (see *RefSideA* and *RefSideB* attributes in the right-hand side of Figure 3.9). Note besides that although internal links do not contain a *direction* attribute, the effective directionality of a given connection can be derived during parsing by interpreting the *RefSideA* and *RefSideB* properties as source and destination connectors, respectively.

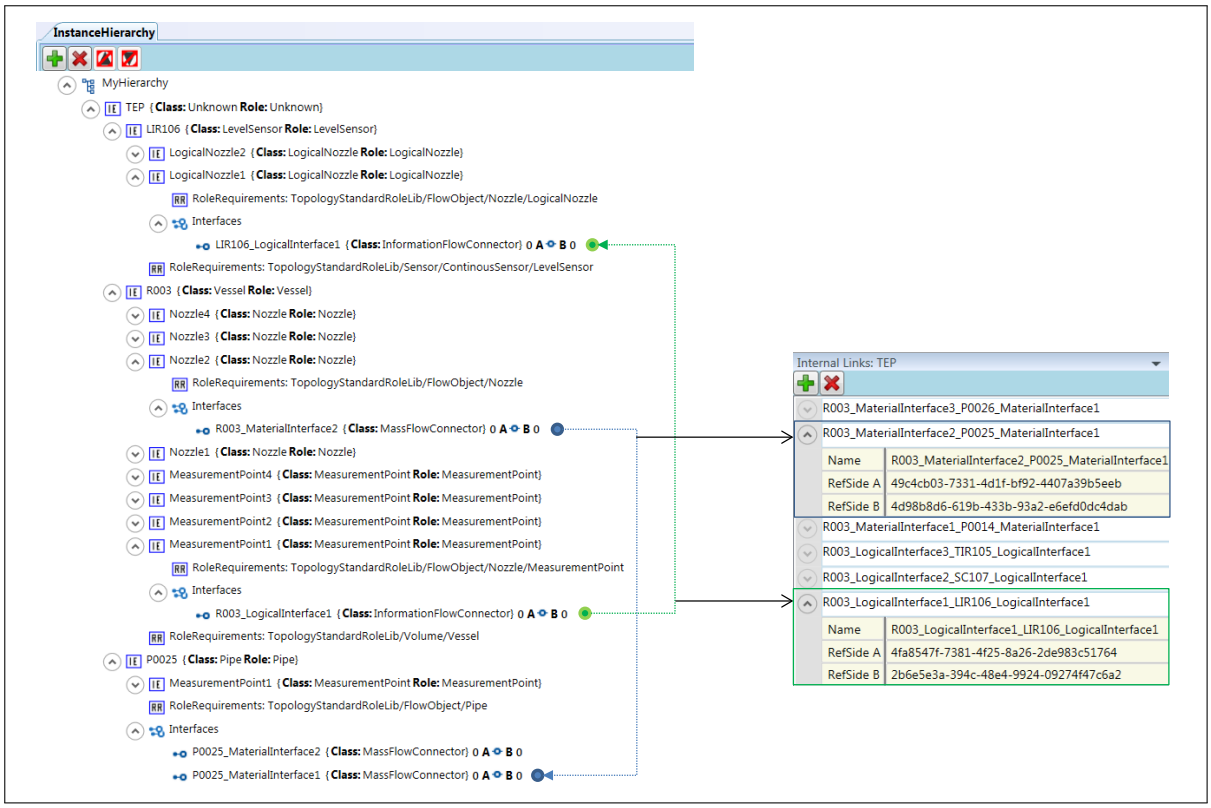


Figure 3.9: Nozzles, interfaces, and internal links within the OO P&ID model. AutomationML visualization with manually added graphical internal links

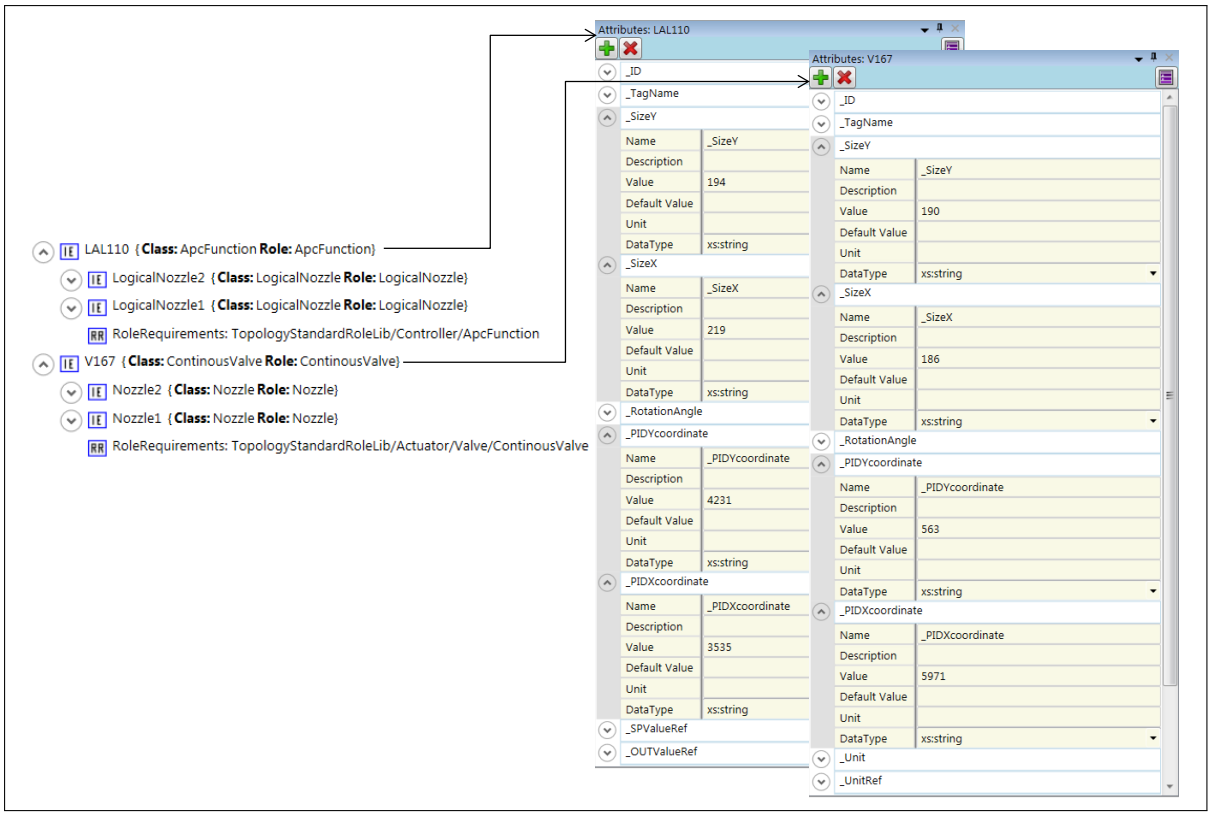


Figure 3.10: Geometric attributes within the OO P&ID model. AutomationML visualization

In what concerns graphical properties of created objects, the algorithm correctly captured and aggregated the dimensionality information contained in the coordinates table to the respective object attributes within the model. Figure 3.10 depicts examples of geometric attributes completed for different objects.

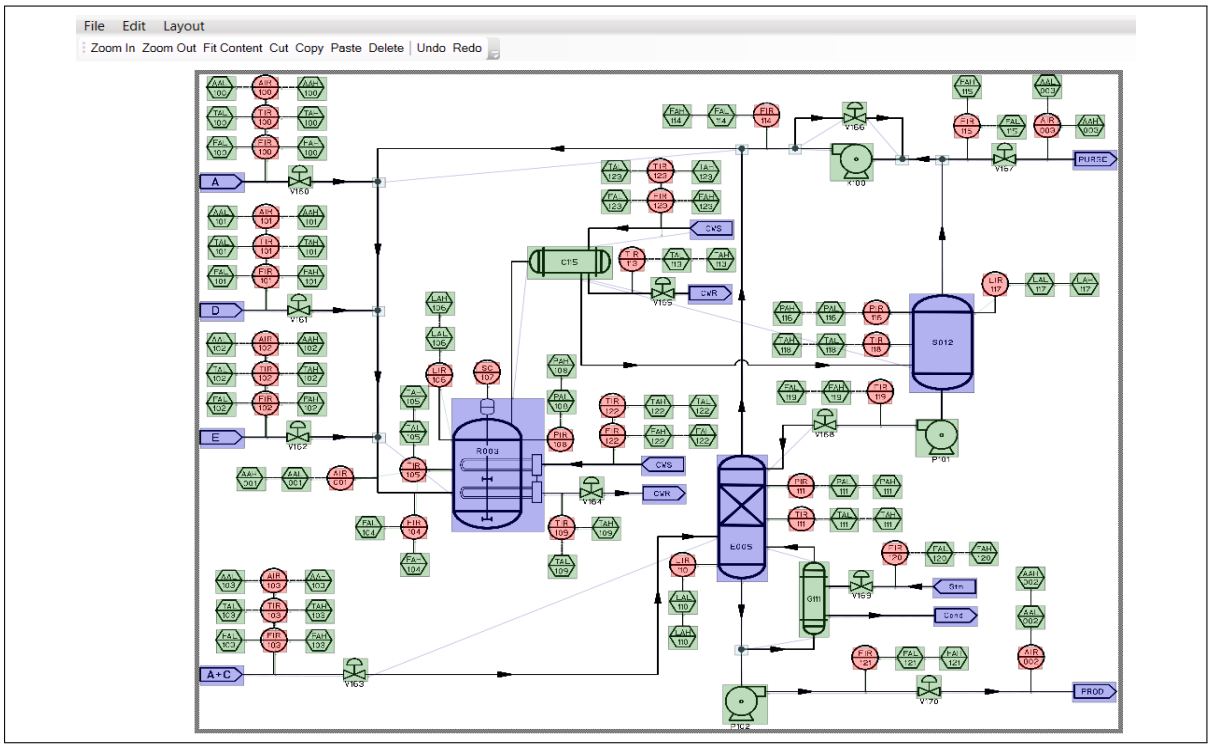


Figure 3.11: Visual inspection of the generated OO P&ID model

This experimental test concluded with a visual verification of the created model. As can be observed in Figure 3.11, the graphical model depiction generated by the algorithm allows the inspection of objects and their connections by direct comparison with the original P&ID image. Although no model changes were necessary in this particular test, if required, the user may effect changes or add missing information at this stage. The complete OO model generated during this test can be found at http://aut.hsu-hh.de/TEP_P&ID.

3.5.2 OO modeling of a control logic diagram

In an analogous fashion, the modeling method was validated on the generation of an OO control logic diagram (CLD). As in the P&ID trial described in the previous section, the corresponding connectivity matrix was used as information source for the generation of objects. Table 3.2 depicts the mapping used in this test.

CM Type	***SystemUnitClass***	***RoleClass***
PidController	*PidController*	*PidController*
ApcFunc	*ApcFunction*	*ApcFunction*
Sensor	*Sensor* (Specific)	*Sensor* (Specific)
Valve	*ContinuousValve*	*ContinuousValve*

Table 3.2: Modeling look-up table used for CLDs

As can be observed in Figure 3.12, the algorithm correctly associated the respective *SystemUnitClasses* and *RoleClasses* to created objects. Note here that the representation shown in this figure is equivalent to the one presented in Figure 3.8, only this time depicted in an XML editor.

InternalElement (34)

	Name	RefBaseSystemUnitPath	ID	RoleRequirements
1	FIR101	TopologyTypeLib/Sensor/ContinousSensor/FlowSensor	eed8a0e0-f435-46e0-bd3d-b774bf1b9d7d	RoleRequirements RefBaseRoleClassPath TopologyStandardRoleLib/Sensor/ContinousSensor/FlowSensor
2	F101	TopologyTypeLib/Controller/PidController	9d5d0a00-2e55-40c3-9adb-c849d2bb9202	RoleRequirements RefBaseRoleClassPath TopologyStandardRoleLib/Controller/PidController
3	V161	TopologyTypeLib/Actuator/Valve/ContinousValve	e750f1d6-d920-411d-9185-b614a99d4afe	RoleRequirements RefBaseRoleClassPath TopologyStandardRoleLib/Actuator/Valve/ContinousValve
4	FIR102	TopologyTypeLib/Sensor/ContinousSensor/FlowSensor	c6923731-5deb-4bd8-91da-09463fc009eb	RoleRequirements RefBaseRoleClassPath TopologyStandardRoleLib/Sensor/ContinousSensor/FlowSensor
5	LIR006	TopologyTypeLib/Sensor/ContinousSensor/LevelSensor	63b123d4-4433-49de-830b-40bdbe5f8eb1	RoleRequirements RefBaseRoleClassPath TopologyStandardRoleLib/Sensor/ContinousSensor/LevelSensor
6	L006	TopologyTypeLib/Controller/PidController	302de03b-93ad-480c-90b9-3f61582aa014	RoleRequirements RefBaseRoleClassPath TopologyStandardRoleLib/Controller/PidController
7	CALC	TopologyTypeLib/Controller/ApcFunction	73758785-1ab2-4eee-9472-e1bef0f1a759	RoleRequirements RefBaseRoleClassPath TopologyStandardRoleLib/Controller/ApcFunction

Figure 3.12: Objects in the generated OO CLD model. XML visualization

Figure 3.13 depicts the XML visualization of the nozzles, interfaces, and internal links created to establish the connectivity of components, specifically the *FlowSensor* FIR101, the *PidController* F101 and the *ContinuousValve* V161, within the CLD model. An aspect worth noticing in this depiction is the existence of only information flow connections, which is explained by the intrinsic nature of the control signals represented in CLDs.

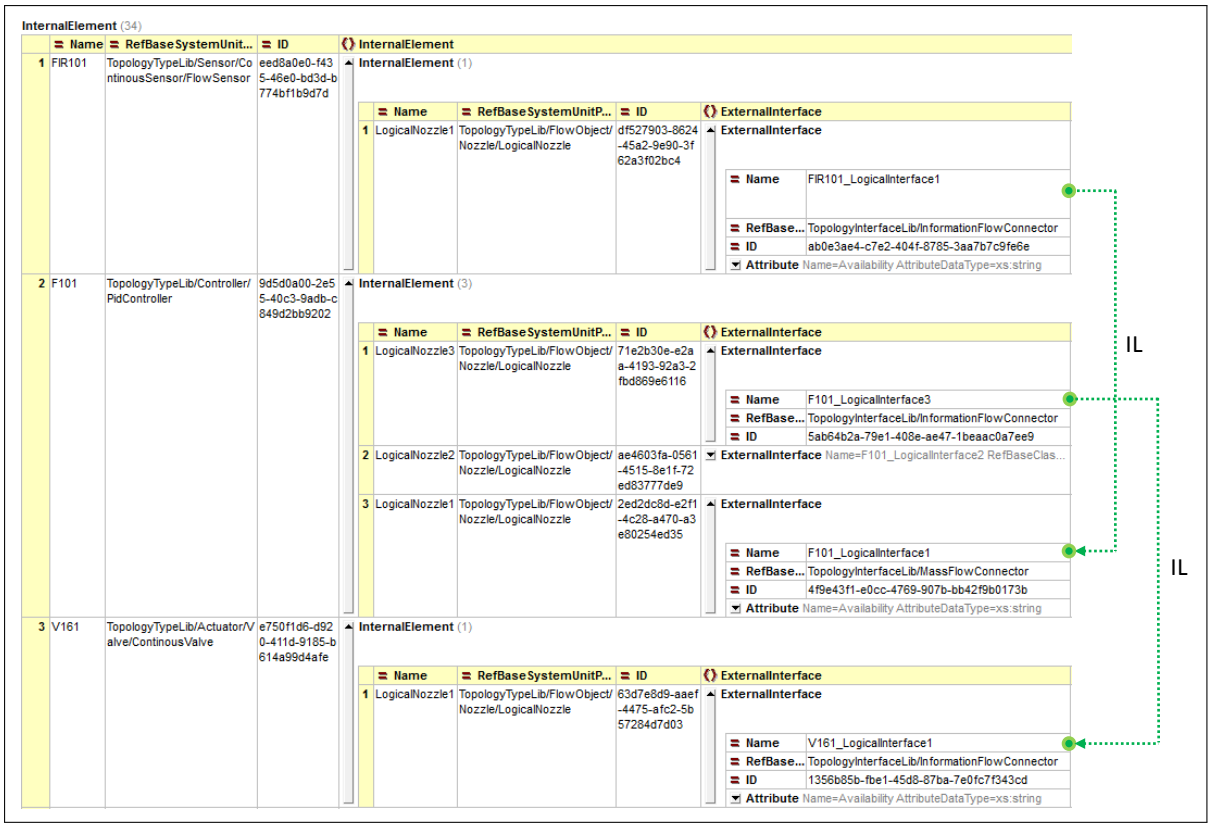


Figure 3.13: Nozzles, interfaces, and internal links in the OO CLD model. XML visualization with manually added graphical internal links

Analogous to the visual inspection of the P&ID in the previous trial, the consistency of the created CLD model was verified by using the graphical representation generated by the algorithm (see Figure 3.14). Once again, no corrections were required in this test. The resulting OO CLD model can be found online at http://aut.hsu-hh.de/TEP_CLD.

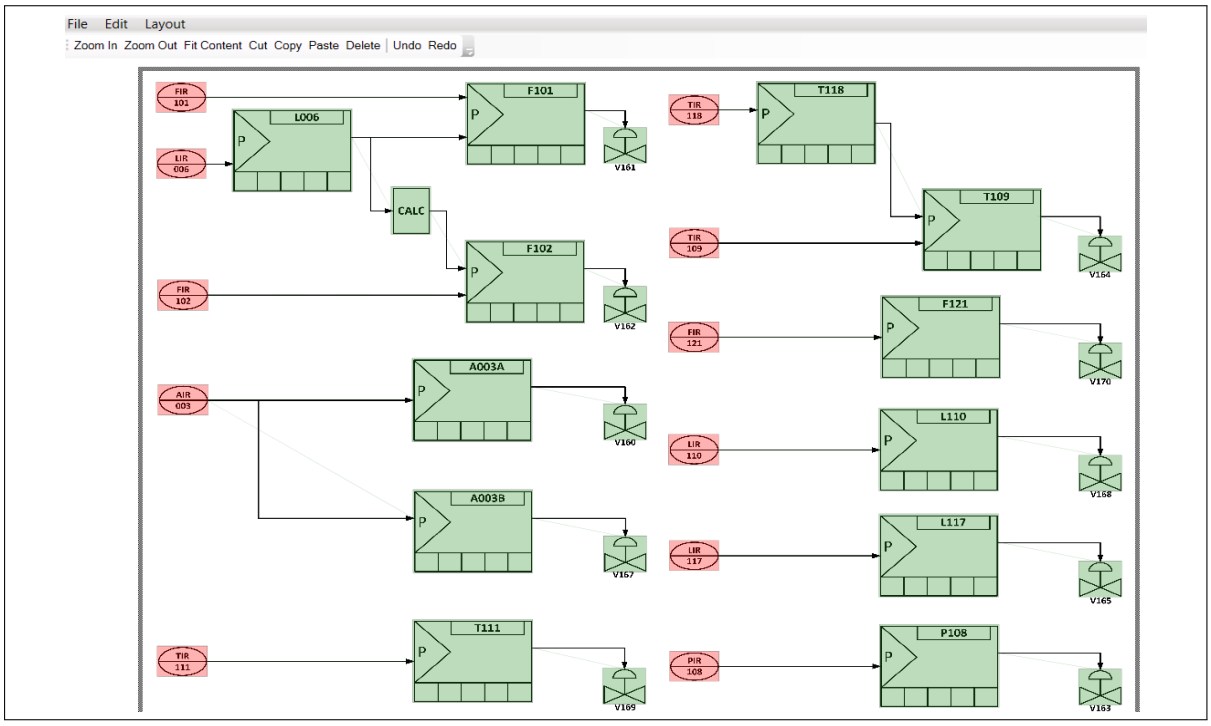


Figure 3.14: Visual inspection of the generated OO P&ID model

3.6 Assessment

Section 3.5 has shown the applicability of the proposed modeling approach on the generation of object-oriented P&ID and CLD descriptions. Based on simple mapping tables, the algorithm was capable of converting plain data captured by document analysis methods (in form of connectivity matrices and tables of coordinates) into AutomationML models with well-defined syntax and semantics. It was demonstrated how the use of nomenclature rules during object instantiation enables the further specification and reclassification of object types according to specific modeling requirements. In addition, it was shown how created OO models can be easily inspected by using the corresponding graphical representations generated by the algorithm.

Compared to existing approaches for automatic generation of OO plant and process models, the method proposed herein enables a larger formalization excursion between source and destination data environments. Concretely, while previous works have concentrated on the generation of AML models from OO sources, the present contribution, instead, focused on the derivation of equivalent formal models from non-object-oriented plant representations originally stemming from legacy data sources. Thus, in combination with the document analysis methods proposed in Chapter 2, this modeling approach enables the effective reuse of legacy engineering information by making data contained in legacy documents directly available for automation of automation [3].

3.7 Chapter summary

This chapter has introduced a modeling method for the generation of object-oriented piping and instrumentation diagrams, and control logic diagrams from plain data sources stemming from legacy documentation. Resulting digital descriptions can be regarded as "smart documents" [Iyu11], which allow not only humans but also computers retrieving plant and process information based on high-level queries. Until this point, however, generated P&ID and CLD models have been regarded as separate information artifacts, despite the fact that they are functionally related depicting together the "controlled plant" concept and must be, therefore, consulted simultaneously during the execution of reengineering and operational activities. The following chapter is accordingly dedicated to introduce concepts for the integration of both model types into a unified topology model as well as for the aggregation and effective retrieval of complementary plant and process data.

[3] For a detailed discussion on the use of AML plant models as an entry point to AoA, see [HMAF16]*.

4. Integration of engineering information sources

4.1 Data integration in the process industry

In the modern process industry, where both products and plants are moving towards higher levels of customization and complexity, the search for new technologies enabling the integration of existing process information, engineering tools, and plant activities has become a key task. The challenge of solving a wide range of design and operational problems involving knowledge from different systems reveals the persistent need of harmonizing and integrating data sources. Under such circumstances, the Integrated Engineering (IE) paradigm [DM07] has emerged as an attractive solution approach, which nowadays captures the attention of the main corporations and research institutes of the process industry in Europe.

In the context of IE, engineering tools and analysis algorithms are called to gain the maximum profit of interdisciplinary knowledge and improve thereby the efficiency and reliability of offered tasks and services [KG13]. Plant diagnosis systems in the process industry, for instance, are called to integrate plant data, historical information, process know-how, and plant structure as a key for enhanced root-cause analysis [IAEA-TECDOC-1252, 2001]#. Likewise, functional diagnostics must be based on models which should preferably stem from engineering tools, be available in object-oriented formats, and provide effective means for information access and exchange.

Aligned with such a vision, different sectors of the process industry have invested significant efforts on the development of solutions for integrating and centralizing plant and process information. A noteworthy example is the NORSOK I-005 standard [NORSOK I-005, 2005]# developed by the Norwegian petroleum industry. This standard introduces the concept of System Control Diagram (SCD), a document combining process descriptions and functional relations implicitly specified by P&IDs with control logic similar to that represented by CLDs (see Figure 4.1).

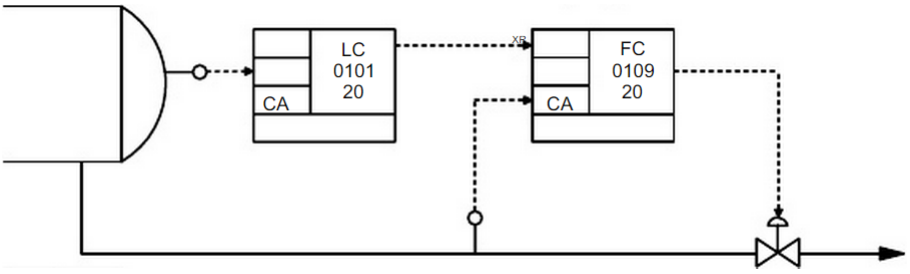


Figure 4.1: Fragment of a System Control Diagram according to [NORSOK I-005, 2005]#

The development of NORSOK I-005 is motivated by existing gaps within the plant engineering workflow, specifically between the process design and the control design phases. As discussed in the standard, process engineers specify the production process through the creation of P&IDs during front end engineering design (FEED). Throughout this phase, they acquire in-depth understanding of the overall plant behavior, particularly on its functionality and operational aspects. However, such understanding is not thoroughly described in the corresponding P&IDs, owing to the fact

that these documents provide limited facilities for documenting such data. Thus, in a subsequent stage of the engineering workflow, automation engineers in charge of designing or revamping the control system must partially reconstruct the process concept in order to develop the required control structures –subsequently documented in CLDs– to fulfill the designed functionality and achieve product specifications. Unfortunately, there is no intuitive link between the control system functions and the process flow itself, except for tags in the respective P&IDs and CLDs. The lack of a explicit interrelation between both information sources hinders the task of engineers to verify that all relevant process aspects have been properly considered during the implementation of the control system. As a consequence, design faults are not identified until the system is commissioned or in operation, which leads to costly modifications in late project stages. SCDs are accordingly aimed at providing the missing information link by allowing the representation of both information sources, process and control, into a single document.

Despite the benefits offered by SCDs, NORSOK I-005 has not being widely adopted in other branches of the process industry. The reason for this is twofold. On the one hand, the lack of a formal computer-interpretable SCD description –beyond the existing SCD Visio drawing library [NORSOK, 2016]$^{@}$– prevents external systems from retrieving data content automatically and hinders thereby the exploitation of these documents within integrated engineering and enterprise solutions. On the other hand, the standard only specifies provisions on functional and drawing related requirements for use of SCDs but not on methods for their generation based on existing data sources, e.g., P&IDs and CLDs. This implies that companies interested in adopting the standard, not only for greenfield projects but also for brownfield and other plant operational activities, must invest strenuous efforts in manually creating new documents for their existing facilities and processes. Evidently, this is an endeavor that not many companies are willing to undertake.

Motivated by both, the benefits and current limitations of SCDs, this chapter presents a method for the automatic integration of P&IDs, CLDs, and complementary plant information into single formal plant topology models allowing the unified description and documentation of process and control information. In addition, concepts for the effective retrieval of integrated data within the resulting digital model are presented, aiming thereby to facilitate the automation of engineering and operational activities across different phases of the plant lifecycle (see AoA → 1.2.1).

The remainder of this chapter is structured as follows: Section 4.2 introduces relevant artifacts for manipulation of XML-based documents together with concepts for effective data access in AML plant models. Section 4.3 gives an overview of existing approaches to integration of heterogeneous engineering information. In Section 4.4, new methods for automatic integration of P&IDs and CLDs, as well as for systematic manual identification and aggregation of complementary process information, and fit-to-purpose data access are presented. The feasibility of the proposed methods for generation of integrated plant topology models is tested in Section 4.5, whereas the respective experimental results are analyzed in Section 4.6. The chapter concludes with a brief summary in Section 4.7.

4.2 Theoretical framework

4.2.1 Document Object Model (DOM)

The Document Object Model (DOM) [W3C, 2016, WHATWG, 2016]$^{@}$ is a W3C specification which defines the logical structure of HTML and XML documents, as well as methods for their access and

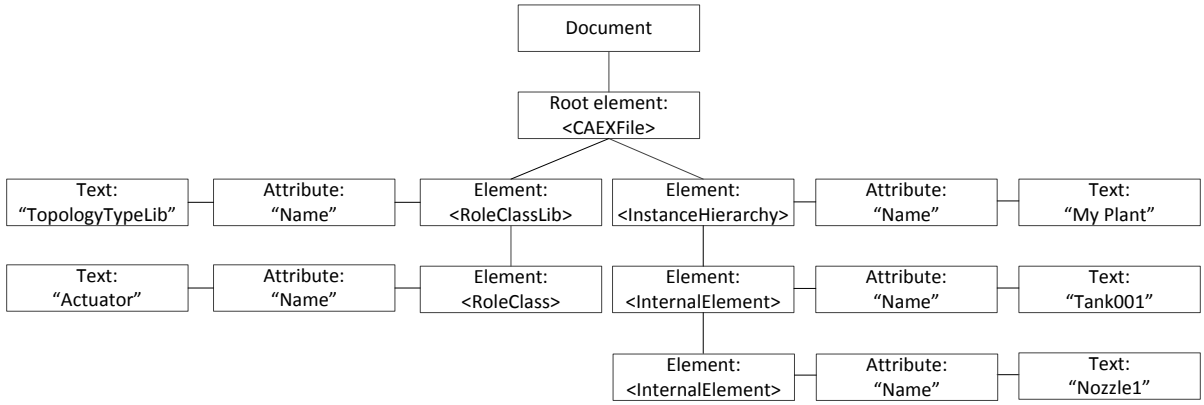


Figure 4.2: Excerpt of a DOM tree structure for a CAEX document

manipulation. The aim of DOM is to provide a standard application programming interface (API) allowing programmers, programs. and scripts to dynamically create, navigate, and modify content in documents within a wide variety of environments, programming languages, and applications.

In the Document Object Model, documents are treated as tree-like structures (node-trees) in which nodes are objects representing document parts, and edges are interfaces depicting their relationships. Such structures allow the effective retrieval of content with little programming effort. Figure 4.2 depicts an example of a DOM node-tree.

In the context of this work, the Document Object Model is used as basis for the navigation and manipulation of CAEX/AutomationML files required during model integration.

4.2.2 Data access concepts in AutomationML

4.2.2.1 AML facet

An AML facet is an object providing a certain view on a subset of attributes or interfaces of a parent AML object. It is a concept designed to allow for effective and automatic access to object information regarding specific engineering or operational aspects.

The syntax and semantics of attributes and interfaces described by a facet are not strictly defined in the CAEX object model. The interpretation of such data is left to external applications (e.g., engineering tools and algorithms), which must have knowledge on the structure and meaning of the content. For provisions on implementation of AML facets see [AutomationML Consortium, 2012a][@].

4.2.2.2 AML group

An AML group is an object allowing the storage of separate views on a subset of AML objects. Specifically, it is a concept for the grouping of *mirror objects*[1] that belong together from a certain engineering or operational perspective.

Groups are conceived for external applications requiring different views on plant data, particularly on its objects. A control performance monitoring algorithm, for instance, may require accessing

[1] In accordance to [IEC 62424, 2008][#], if an instance $A*$ references an instance A, $A*$ and A are respectively a mirror object and a master object. Mirror objects reference master objects and all their data; therefore, they are regarded as pointers to master objects.

information on controllers but not data regarding other components in the plant hierarchy. A group object can provide the required data access by enabling a separate view on the considered objects –controllers in this particular case.

The group and facet concepts can be combined to allow the retrieval of fit-to-purpose information within the plant hierarchy. By using the group attribute *AssociatedFacet*, a group can be pointed to specific facet objects. Thus, algorithms can automatically identify related objects and their corresponding facets in order to collect data of interest (attributes and interfaces). Further provisions on groups and its combined use with facets can be found in [AutomationML Consortium, 2012a][@].

4.3 State of the art

Recent literature on process and manufacturing technologies discusses different approaches to data integration for enhanced engineering and operational activities. Birk et al. [BJC+10], for instance, presented an approach to integration and visualization of process and control data based on signal flow graphs. In such structures, nodes represent signals (esp. process variables), whereas edges embody linear dynamic models approximating the relation between variables. As part of the visual representation, three information layers, i.e., process layer, process models, and process controllers, can be displayed and interrelated either as graphs or as connectivity matrices.

The authors of [LHK10] showed how external AutomationML interfaces (COLLADA and PLCOpen XML) can be exploited to integrate knowledge regarding geometry and control logic of manufacturing units. In the process domain, Christiansen et al. [CJSF11] presented a method to integrate formalized process descriptions [VDI/VDE 3682, 2014][#] into topology models in AutomationML [IEC 62424, 2008][#].

A further data integration approach was presented by Cui et al. [CZZ10]. The authors of this work proposed a method for merging process topology contained in smart P&IDs and HAZOP expert systems. The approach targets the integration of HAZOP as a fundamental part of the process design by enabling the supported execution of required analyses based on available design data.

Dorantes Romero et al. [DGT15, DT14] introduced a method for integrating smart P&IDs, electrical diagrams, and interlock logic in form of cause and effect (C&E) matrices. To that end, the authors proposed the concept of property-graph, a graph network aimed at allowing mainly for visualization, but also for navigation, and solution of queries on plant data.

The aforementioned approaches have presented a variety of methods for integrating disparate plant and process data sources. Most of them, however, require deep process knowledge and significant manual effort for the generation of integrated data models, either for defining the role of the different information pieces within the model [BJC+10] or for establishing the interrelationships between model components [BJC+10, LHK10, CJSF11, FWVHM12]. While other approaches [DGT15, DT14] claim to perform the required data integration by automatic means, they clearly state that the degree of automation depends heavily on the availability of sources in computer-readable forms, and they do not provide details on how the different information pieces are merged together. A last drawback of state-of-the-art methods is the lack of specific means for automatic and fit-to-purpose access to integrated information, which hinders the effective retrieval of content by external systems.

This chapter aims at addressing the aforementioned gaps by presenting methods for the automatic integration of design plant documents, systematic manual identification and aggregation of complementary process data, and effective information access within integrated plant models.

4.4 Proposed method for plant data integration

The proposed data integration method is divided into three main phases: integration of P&IDs and CLDs, identification and aggregation of complementary plant and process information, and data access and management within the integrated model. The following sections provide a detailed description of the main steps carried out within every phase.

4.4.1 Integration of P&IDs and CLDs

Throughout the process of consolidating relevant information during reengineering and operational activities, automation specialists and other experts typically consult existing P&IDs to get an overall picture of the production system as well as to quickly identify existing relations between process units[2]. Once project-relevant assets and plant sections have been identified, experts proceed to collect data regarding existing control structures from available CLDs.

The linking artifacts used for localizing and retrieving data from the right documents are specific P&ID symbols and their associated text identifiers representing instrumentation and control tags. By way of illustration, an automation engineer can retrieve the specification of the temperature control concept used in a given process reactor by firstly identifying the respective instrumentation symbol and its associated tag (e.g., T0023) in the P&ID, and subsequently accessing the corresponding CLD named with this tag, or, in its defect, containing the tag in its underlying content. Note that in view of the large size of industrial sites, tag and document names are commonly based on hierarchical designation systems. The full name of a tag, for instance, may consist of a plant identifier (e.g., LEV1502), an area identifier (e.g., V030), a section identifier (e.g., CA11), and a specific equipment or measurement identifier (e.g., T0023), thus resulting in a string of the type *LEV1502V039CA11T0023*. Common designation systems used in different process industry sectors include *AKZ* [IEC 61346, 2000]#, *KKZ* [Web08], and *DKZ* [DIN EN 61355, 2008]#.

The developed concept for merging process and control engineering diagrams, specifically P&IDs and CLDs, also exploits process tags as the artifacts to localize merging points among the analyzed documents. For computational purposes, however, the data access sequence followed by experts is inverted in the method: i.e., the algorithm accesses first the content of CLDs and then localizes merging points in the P&ID where control structures are to be integrated. The reason for the inverted workflow stems from the typically larger dimensions of P&IDs in relation to CLDs.

The first method steps consist in loading the analyzed CLD and P&ID models into memory and compute their respective tree-graphs T_{CLD} and $T_{P\&ID}$. Thereupon, the algorithm evaluates if the size of T_{CLD} is larger than zero, which indicates whether there are still control objects to integrate or not. In the positive case, it continues the routine; otherwise it concludes execution. For those cases in which T_{CLD} is larger than zero, the algorithm proceeds to get the next object O_i in T_{CLD} and computes the full connectivity chain C_i associated to this object. Subsequently, C_i is stored in

[2]Although a large extent of this information can be retrieved from other data sources, e.g., HMIs, the use of P&IDs is often preferred by experts due to the level of detail offered by these documents.

memory and their elements are removed from T_{CLD}. At this stage, the initial and final objects of C_i, which correspond to sensors and actuators, are determined and stored as O_{C-b} objects.

As next step, the algorithm parses $T_{P\&ID}$ searching for O_{C-b} objects in the P&ID. Recall here that O_{C-b} objects represent linking artifacts (sensors and actuators), and as such they are also contained in the P&ID. Notice, however, that in some cases the tags of corresponding O_{C-b} objects may differ between CLDs and P&IDs owing to document designation rules. Accordingly, the search algorithm incorporates knowledge on used nomenclature conventions by means of user-defined regular expressions (→2.2.3.5). For every found O_{C-b} object, new nozzles and interfaces are created in accordance to the connectivity defined in C_i. The respective IDs of new created interfaces are stored for further use during the definition of connections in a later step of the method.

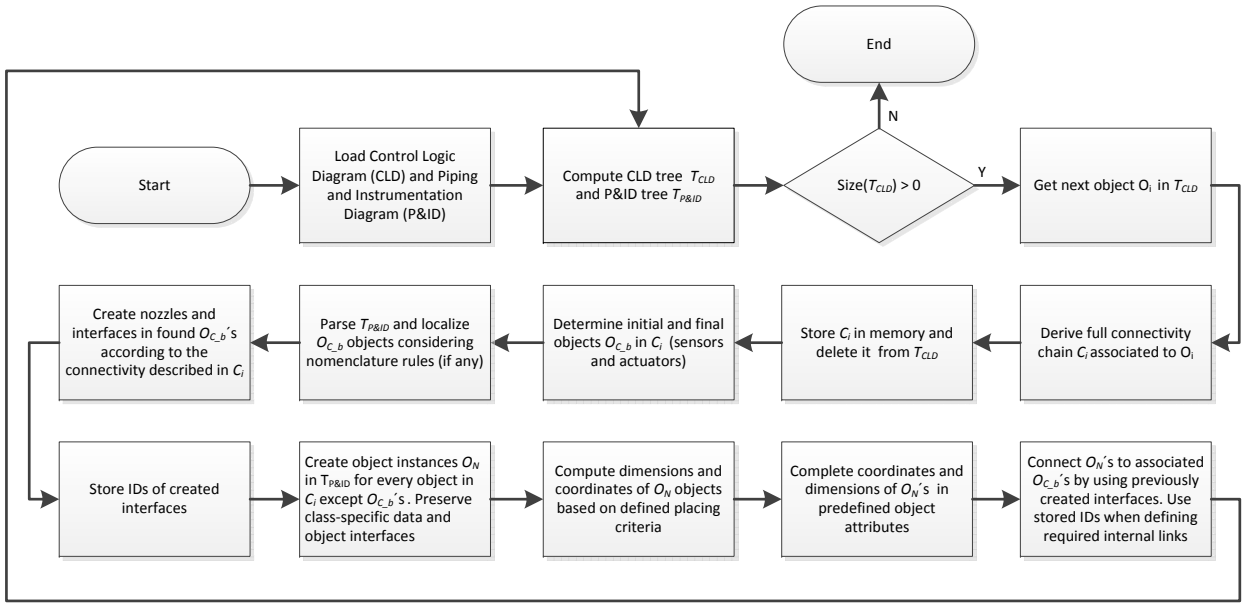


Figure 4.3: Simplified representation of the model integration algorithm

The routine continues by creating instances O_N in $T_{P\&ID}$ for every object in C_i, except O_{C-b} objects. Note that the latter already exist in $T_{P\&ID}$ and accordingly must not be created anew. Should this not be the case and one or more O_{C-b} objects do not exist in $T_{P\&ID}$, the inconsistency will be identified during the creation of object connections in a later step of the method[3]. Within the instantiation process the algorithm preserves class-specific data and object interfaces, but recomputes the values of geometric attributes including diagram coordinates and dimensions. The determination of the coordinates at which O_N objects are to be placed in the integrated model can be addressed by considering different functional and visualization criteria. From a functional perspective, establishing the required connectivity of new and existing objects essentially suffices the requirements of the object placing problem. Thus, O_N objects could be simply placed in the neighborhood of those objects to which they must connect. From a visualization perspective, in contrast, the appearance and readability of objects in the diagram plays a fundamental role in determining insertion coordinates. In this case, optimal locations can be found by minimizing the lengths of new connection lines and the degree of overlap between new and existing objects. As the

[3]In cases in which multiple P&ID and CLD must be integrated, it is recommended to previously merge the respective models into single P&ID and CLD models before executing the algorithm. This step allows mitigating eventual inconsistencies caused by missing tags during the integration process. The required merging functionality is deployed by existing tools such as CAEXMapping [Fraunhofer, 2016][@].

purpose of this contribution is to provide formal models mainly for computer access, the functional definition of coordinates is followed here. In any case, if the user considers it necessary, manual editing can be easily performed later on during visual inspection.

In the last stage of the method, O_N objects are connected to associated O_{C-b} objects by using the previously created interfaces and defining corresponding internal links based on stored interface IDs. Here, a subroutine verifies that all interfaces have been consistently connected, generating opportune warnings and highlighting objects with missing connections otherwise. Finally, the algorithm proceeds to recompute the respective document trees T_{CLD} and $T_{P\&ID}$ and the routine is repeated until all control objects have been integrated or until no correspondence to remaining objects have been found in $T_{P\&ID}$. As ultimate result of the procedure, a new $T_{P\&ID}$ representing the integrated topology model is generated. Figure 4.3 presents a simplified representation of the described model integration algorithm.

4.4.2 Identification and aggregation of additional plant and process data

Previous research has shown that the use of different information sources can improve the efficiency and quality of reengineering and operational activities [TBBT09, BSF+09, CFON11, HFB15]. A noteworthy example is automated fault detection and diagnosis (AFDD), where the consideration of the process characteristics as well as the relationships between plant assets and control structures can enhance the resolution of plant diagnostics. Yet, the behavior of plant disturbances is not simply conditioned by general structural knowledge, but also by further propagation-related factors (PRFs) that shall be identified and retrieved for improved root-cause determination. Up to now, however, the exploitation of such factors has been hindered by (a) the lack of consistent schemas for data identification and classification, (b) the reduced number of methods for integration of heterogeneous data sources, and (c) the difficulties entailed for timely access and exchange of data.

The aforementioned deficiencies do not only represent an obstacle for AFDD but also for several other activities across the entire plant lifecycle. In an effort to cope with such a problematic, this section presents a method for identification and aggregation of relevant plant and process data into formal plant topology models. Aiming at specificity, focus is set on the analysis of data required for fault diagnosis, concretely for plant disturbance propagation analysis.

4.4.2.1 Data identification

The consideration of plant and process data can support the automation of different diagnosis tasks insofar as key information can be timely accessed by condition monitoring systems. General examples of such information include plant connectivity, component specifications, and data transferred between PCS and control field devices. Nonetheless, enhanced diagnostics require a more granular specification of data sources allowing for the identification of explicit propagation-related factors (PRFs). An approach to achieving such degree of specificity relies on the definition of automation use cases (AUCs) covering the different phases of the plant lifecycle. Examples of AUCs include HMI generation, Automated Fault Detection and Diagnosis (AFDD), hazard and operability studies (HAZOP), simulation, Alarm Management (AM), and Plant Asset Management (PAM).

Identification of relevant data within each defined AUC is carried out based on first physical-principles, components and systems specifications, and analysis of process design documents. Such a viewpoint of the information management problem allows for the identification of specific factors,

Table 4.1: AFDD AUC information excerpt

Information Type	Factor	Factor Example	Priority	Source
Process	operation set points	temperature, pressure	High	PCS control code, Design documents
	operation limits	min/max	High	
	product properties	concentration, pH	Moderate	
	flow properties	directionality, phase	Moderate	
Control	structure	components	High	PCS control code, P&IDs, CLDs, PCS screens (Faceplates), C&E matrices
	variables	PV, OP, SP	High	
	interlocks	C&E relations	High	
	parameters	Kc, Ti, Td	High	
Plant Item	specs	type, power	Moderate	P&IDs, Component-lists, PCS screens
	parameters	efficiency	Moderate	
	geometry	shape, length	Low	
	dimension	volume	Moderate	
	location	plant position	High	
Pipeline	role	main, bypass	High	P&ID, PCS screens
	dimension	length, diameter	Moderate	
Alarm System	limits	lolo/hihi	High	PCS control code
	functions	grouping	Moderate	
PCE Measuring Point	type	temp., press.	High	PCS control code, P&IDs
	range	min-max	Moderate	
	location	plant position	High	
	tags	online values	High	
Communication	type	4-20 mA, 0-5V	Low	Control code, CLDs
	structure	topology	High	
	tags	PCS tags	High	

their containing sources and respective availability, as well as individual priorities and requirements. In the particular case of plant abnormal situation analysis, relevant information is collected in the AFDD AUC as shown in the data excerpt of Table 4.1. For further information on automation use cases, the reader is referred to [ACHF14]*.

4.4.2.2 Data aggregation

Some of the information identified in the AFDD AUC is implicitly contained in the integrated topology model. The designed flow directionality and the structure of control loops, for instance, are data items which can be derived by analyzing the connectivity of plant assets and control elements within the model. However, other propagation-related data such as measuring ranges, equipment specifications, and process operation points, are not pieces of information which can be implicitly gathered from P&IDs or CLDs, and accordingly must be aggregated by other means into the integrated topology model.

In this contribution, the aggregation of such data is realized by defining use-case-specific attributes within model objects accounting for plant assets, instrumentation devices, and control equipment. Attributes allow collecting a large set of different information items including annotations, numerical values, logic descriptions, and mathematical expressions. Such a versatility makes them highly suited for the description of heterogeneous plant information as discussed in detail in [ASC+14]*.

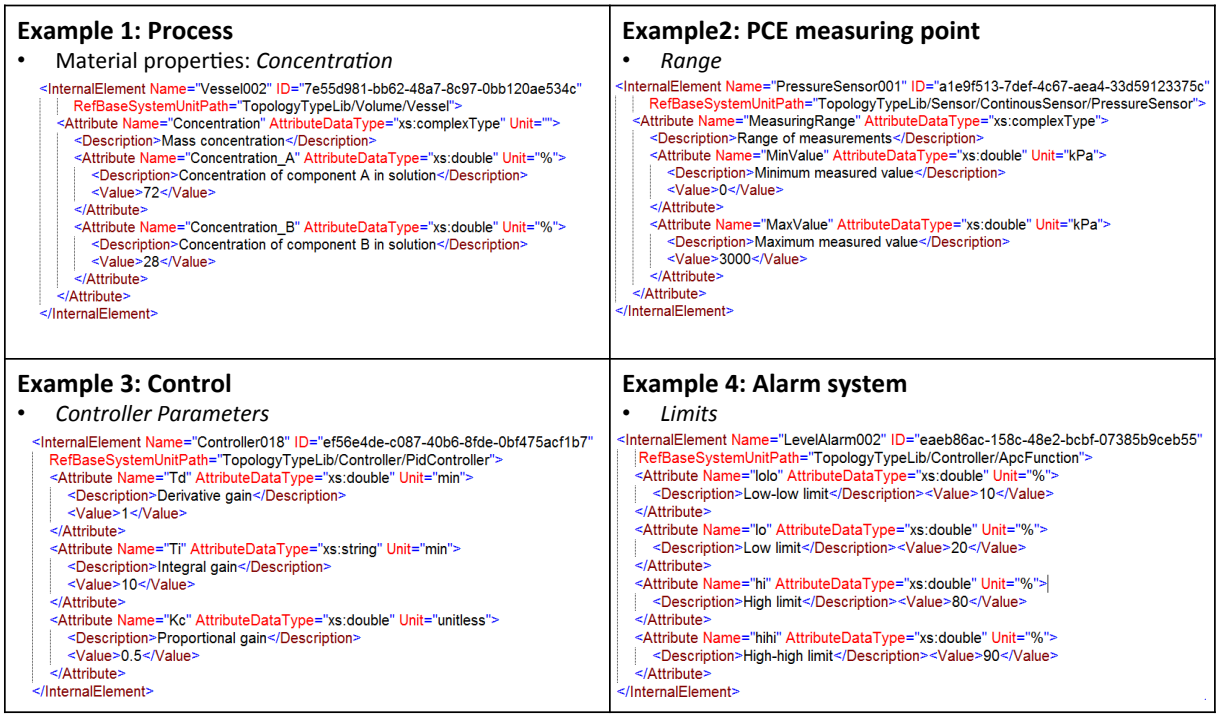


Figure 4.4: Aggregation of different types of AFDD data in model objects

The syntax and semantics of defined attributes are not constrained to a specific implementation. In fact, they can adopt different forms depending upon the particular modeling requirements of the considered use case or target application. The interpretation of the aggregated data is a task of external applications, which should have knowledge on the format and underlying meaning of the retrieved attributes. Observe that attributes considered at this stage are in general not part of the object information model (OIM) presented in Chapter 3. The reason for this is that the OIM aims at providing a general information structure for generating plant models serving as a base for several AoA use-cases but not tailored to any in particular. Defining use-case-specific attributes in the OIM would result in unnecessarily large plant models and reduce modeling flexibility.

To illustrate the aggregation of different types of AFDD AUC data (see Table 4.1), consider the series of examples presented in Figure 4.4. As shown in example 1, process data such as the mass concentration of a substance in a process vessel can be added in form of a *complexType* attribute *Concentration* with numeric sub-attributes *Concentration_A* and *Concentration_B* specifying the percentual abundance of the different components in the solution. In a analogous manner, example 2 illustrates how PCE data, concretely sensor measuring ranges, can be specified by a *complexType* attribute *MeasuringRange* with numeric sub-attributes *MinValue* and *MaxValue* describing the minimum and maximum limit values in the measuring interval. The aggregation of control relevant information, specifically controller parameters, is illustrated in example 3. Here numeric attributes *Kc, Ti, Td* are used to store the proportional, integral and derivative gains of a PID controller. Finally, example 4 shows how alarm-relevant information, particularly alarm limits, can be described as numeric attributes *lolo*, *lo*, *hi*, *hihi* of alarm functions (*ApcFunction* objects). Observe in all cases that the properties of defined attributes (i.e., *Description*, *AttributeDataType*, *Unit,* and *Value*) enable a detailed characterization of the aggregated AFDD data.

A last aspect to be discussed in this section is how to describe the identified priorities of AFDD information items (see column *Priority in* Table 4.1). A possible approach to address this problem

is to define a *priority* property with possible values *low*, *moderate* and *high* for every created AFDD attribute in the model. This solution allows external applications to retrieve data with their respective AFDD-specific priority (relevance) and process it accordingly. An alternative solution consists in defining the respective priority values in the semantics repositories used by external applications accessing model data. In this fashion, priorities can be adjusted and maintained depending on the analysis scope and specific algorithms used by every application. The latter solution is used in this contribution on account of implementation flexibility.

4.4.3 Data access and management within the integrated model

Differences found among the nature of the PRFs suggest that a functional classification may facilitate the access and management of this information in the topology model. Accordingly, this contribution proposes a sorting schema comprising three PRF classes, namely: *Plant Dimension Specific* (PDS), *Plant Component Specific* (PKS), and *Process Specific* (PS). In the context of this classification schema, *Plant Dimension Specific* refers to those factors related to the location, size, and geometry of the plant and its components, e.g., positions, lengths, and diameters. This class of PRFs is particularly relevant for plant diagnostics in oil and gas facilities where the diameters of pipelines may ascend to several meters and their lengths might go up to kilometric distances [Tub16]. *Plant Component Specific*, in turn, makes reference to those parameters related to the function and operation of plant assets. Power, operation mode, and component specifications are some examples of the PKS class. At last, *Process Specific* refers to factors regarding process characteristics derived from operation states or processing conditions. Typical examples of factors in this class include product sequences, control loop configurations, and online process values.

Table 4.2: Classified AFDD AUC data

Class	Factor	Factor Example	Priority	Source
PDS	assets geometry	shape, length	Moderate	Mechanical design documents, P&ID, PCS screens
	assets location	position in plant	High	
	dimensions	length, area, volume	High	
	measuring point location	position in unit/plant	High	
PKS	component specs	type, power	Moderate	Component lists, Assets Specs, P&ID
	component role	main, redundancy	High	
	pipeline role	main, bypass	High	
PS	operation set points	temp.,press.	High	P&ID, CLD, Alarm system, PCS screens, PCS control code
	flow properties	directionality, phase	High	
	control structure	components	Moderate	
	control variables	PV, OP, SP	High	
	control parameters	Kc, Ti, Td	High	
	alarm limits	lolo/hihi	High	
	alarm functions	grouping	Moderate	
	interlocks	C&E relations	High	
	measuring type	temp, press	High	
	measurements	online values	High	
	measuring ranges	min-max	Moderate	
	communication type	4/20 mA, 0-5V	Low	
	communication structure	topology	High	
	communication tags	PCS tags	High	
	product properties	concentration, pH	High	
	process	mixing times	High	

By using the aforementioned data sorting schema, the information of the AFDD AUC (Table 4.1) can be consistently sorted as shown in Table 4.2. This viewpoint of the AFDD information forms the basis of the fit-to-purpose data access concept proposed in the following.

4.4.3.1 Fit-to-purpose data access

During the automatic execution of plant diagnosis, different monitoring algorithms require accessing variable information sources depending upon the scope or resolution of the analysis. In some cases, for instance, process-specific information (such as the mixing speed or the heating time of a reactor) and structural information (such as its input/output connectivity) may be used to generate root-cause hypotheses for a concentration variance in the product, e.g., wrong set points or alteration of the feed. In other situations, plant component specific knowledge may be enough to draw conclusions about the source of the problem, for example, by verifying that the nominal power consumption of the reactor has been exceeded during operation, suggesting thus a possible technical anomaly. Under such circumstances, a significant efficiency improvement of plant diagnoses can be attained by providing monitoring algorithms with specific views of the information contained in the integrated plant topology model. Thereby, only required information for specific diagnosis tasks will be retrieved, which in turn results in shorter processing times and a significant reduction of the data management complexity.

In the particular case of the integrated plant topology model, customized data accessibility and visualization is enabled by exploiting the concepts of AML facet and AML group. Recalling Section 4.2.2, an AML facet is an object providing a certain view or sub-view on attributes and interfaces of a parent object (component-view). Thus, the facet concept can be consistently exploited for providing specific access to different propagation-related factors (PRFs) of a model object representing either a plant asset, an instrumentation device, or a control equipment. To that end, three facets, namely *PDS_FACET*, *PKS_FACET*, and *PS_FACET* are defined within each object in the model.

The upper part of Figure 4.5 illustrates the facet concept applied on a typical plant asset: a pump. By accessing the *PKS_FACET*, for instance, a diagnosis algorithm can effectively visualize and retrieve only the required plant-component-specific factors of the asset (*Attribute B*); in this particular case, e.g., its *nominal capacity*.

The facet notion can be extended to the visualization of subsets of plant objects by exploiting the AML group concept. AML groups allow structuring identical objects in different hierarchies, and combined with the attribute *AssociatedFacet* (AF), it can be related to specific sets of attributes and interfaces within groups of objects. This enables diagnosis algorithms to derive fit-to-purpose information from an entire model.

In this contribution, four groups are defined: a parent group *Group_AFDD* representing the whole set of propagation-related factors, and three child groups with respective associated facets: *Group_PDS* (AF = *PDS_FACET*), *Group_PKS* (AF = *PKS_FACET*), and *Group_PS* (AF = *PS_FACET*) representing the objects encompassing plant dimension specific, plant component specific, and process specific factors of contained mirror objects (→ 4.2.2). Thus, by accessing, for instance, *Group_PDS* with AF= *PDS_FACET*, a diagnosis algorithm will have direct view on and access to particular attributes of the master objects *Pump001* and *Vessel002*, namely *Attribute A* and *Attribute E*, as depicted in Figure 4.5. Typical examples of such attributes might be: *Pump discharge outlet size* and *Tank volume*.

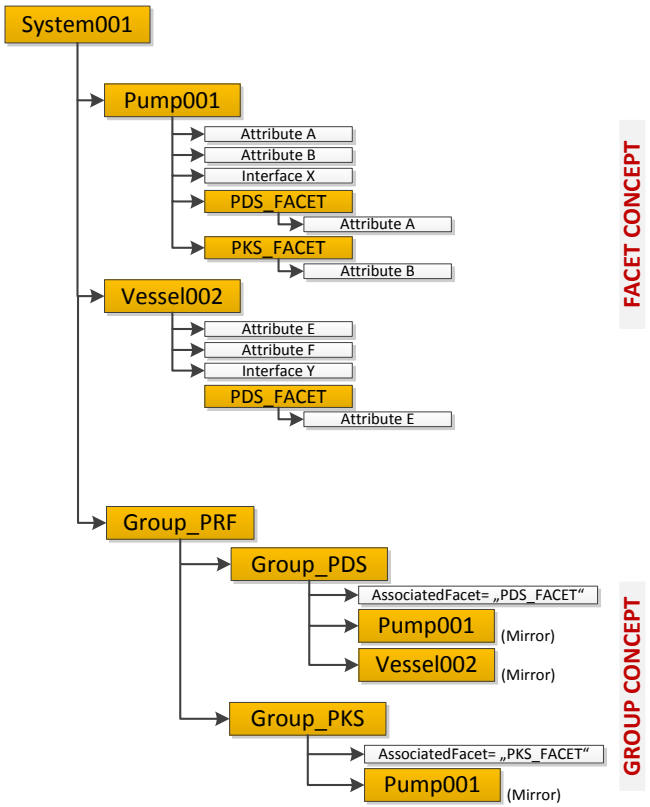


Figure 4.5: Applied AML facet and group concepts

4.4.3.2 Management of static and dynamic information

In order to guarantee the correct operation of diagnosis algorithms, it is fundamental to provide the topology model with means to handle the management of dynamic and static information. Such mechanisms should allow informing external applications about the time behavior of the retrieved data so that this can be processed accordingly. For instance, the topology model must clearly indicate that certain attributes such as the nominal power and volume of a reactor are static factors, whereas level and pressure measurements exhibit dynamic time behaviors and should be accordingly updated during plant monitoring. In this contribution two approaches are proposed for this task:

1. Providing PRFs with a *TimeBehavior* property. Such a property can be implemented in AutomationML by the definition of a sub-attribute *TimeBehavior* and possible values *static* or *dynamic*. Thus, sub-attributes would function as flags on PRFs and can be used by external algorithms to process the data in accordance to its time behavior type.

2. Creating additional data visualizations by the declaration of new facets, groups, and respective associated-facets, namely *PRFDynamic_FACET*, *PRFStatic_FACET*, *Group_PRFDynamic* (*AF=PRFDynamic_FACET*) and *Group_PRFStatic* (*AF= PRFStatic_FACET*). Note that this approach follows the same logic described in Section 4.4.3.1.

As it is possible to infer, the implementation of the first approach results in smaller models, whereas the application of the second yields enhanced means for data access. Hence, the selection of one approach depends on the modeling and execution requirements of the diagnosis solution. Aiming at modeling simplicity, the first approach is chosen for implementation in this work.

4.5 Validation

With the aim of validating the applicability of the proposed integration method, the P&ID and CLD OO models generated in Chapter 3 were used as data inputs. In addition, available process and control descriptions of the TEP [DV93, Luy96] were taken as basis to exemplify the proposed concepts for data aggregation and fit-to-purpose information access. The following sections summarize the main results obtained during the evaluation process.

4.5.1 Integration of P&IDs and CLDs

In accordance to the model integration method presented in Section 4.4.1, the implemented algorithm successfully loaded the analyzed P&ID and CLD AML models into memory and computed their respective node-trees (T_{CLD} and $T_{P\&ID}$) by using DOM (→ 4.2.1). Subsequently, it recursively proceeded to get the next objects in T_{CLD} and calculated their full connectivity chains. Figure 4.6 illustrates two of the chains captured within this process.

Figure 4.6: Example of connectivity chains computed within the analysis of a CLD

Later on, connectivity chains were stored in memory and their respective objects (e.g., *LIR117*, *L117*, and *V165*) were iteratively removed from the respective T_{CLD}. Initial and final objects within the chains (see sensors and actuators highlighted in purple color in Figure 4.6) were correctly determined and assigned to the set of O_{C-b} objects.

Following the workflow, the algorithm parsed the $T_{P\&ID}$ searching for instances of O_{C-b} objects in the P&ID. For the purpose of this test, no nomenclature mapping rules (i.e., user-defined regular expressions) were defined as sensors and actuators had the same names within the analyzed CLD and P&ID models. During this step, the algorithm prompted one warning reporting that no instance of the O_{C-b} object *LIR006* was found in the P&ID. Despite this problem, the algorithm correctly continued the execution, creating the required nozzles and interfaces for every found O_{C-b} instance in $T_{P\&ID}$ based on the connectivity described by their specific connectivity chains.

In the last stage of the process, the algorithm created instances of the control objects still remaining in the connectivity chains (see objects in black color in Figure 4.6) and assigned insertion coordinates by following the functional object location approach discussed in Section 4.4.1. Finally, it correctly established the connectivity of new and existing objects by creating internal links and assigning respective interface IDs. During this step, however, a second warning was generated, this time informing that not all created interfaces were successfully connected, concretely one or more interfaces in the created controller *L006*. Apart of the respective prompted execution message, this warning can be visualized in the resulting integrated topology model as a "*" symbol concatenated to the name of the object (see *L006** in Figure 4.7).

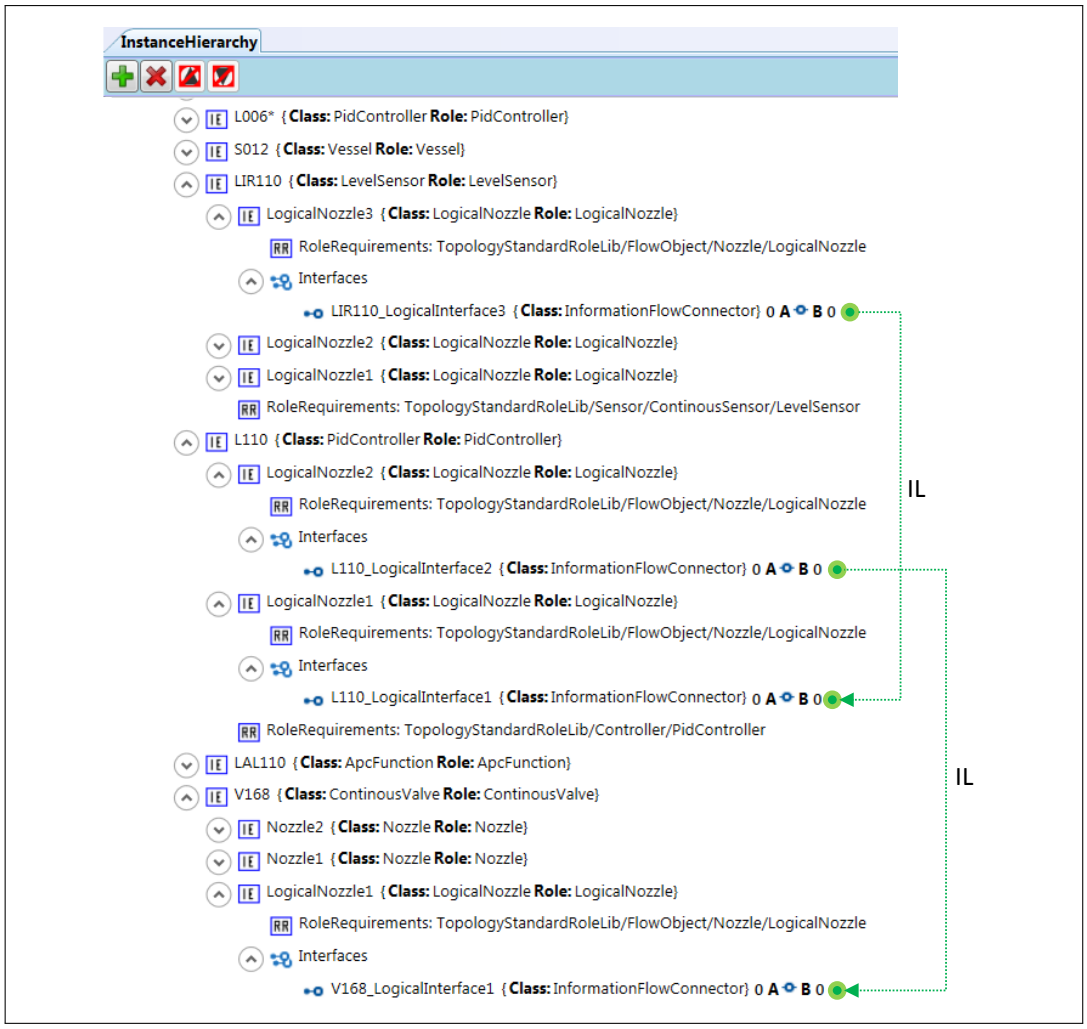


Figure 4.7: Excerpt of the resulting integrated topology model

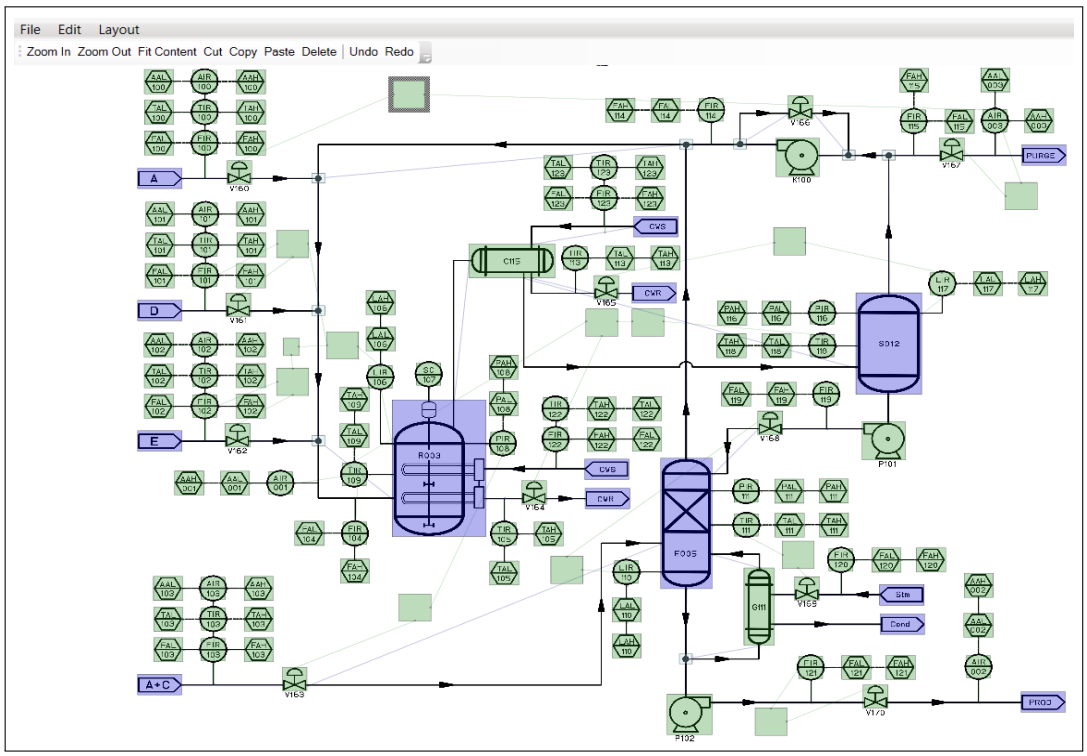


Figure 4.8: Graphical depiction of the resulting integrated topology model

A brief analysis of the two generated warnings suggested that the source of the problem could be related to a nomenclature mismatch between objects within the analyzed models. This type of inconsistency typically results from transcription mistakes made by humans during the creation of documents or from character recognition errors occurring throughout the document digitization process. By inspecting the analyzed P&ID and CLD documents (→Appendix A and Appendix B), it was confirmed that a mismatch between the tags *LIR106* (P&ID) and *LIR006* (CLD) caused by a transcription mistake ("1" replaced by "0") was the root-cause of the problem. The error was manually amended by adding the respective missing connections in the model. At last, the resulting integrated topology model was visually inspected in the generated graphical representation, which now shows both process and control objects in a unified diagram[4] (see Figure 4.8).

4.5.2 Aggregation of AUC AFDD data

Based on the AUC approach to data identification and collection (→ Section 4.4.2), AFDD-relevant information on the analyzed process and its regulatory control structure was manually gathered and organized as partially shown in Appendix E.

Subsequently, sorted information was manually aggregated in the topology model in form of attributes following the general provisions discussed in Section 4.4.2.2. Figure 4.9 depicts an excerpt of the resulting integrated model with aggregated AFDD data (i.e., propagation-related factors). Observe here that created attributes are associated to model objects representing plant assets, instrumentation devices, and control equipment. Notice besides that added information is characterized by metadata in form of descriptive properties (i.e., names, descriptions, units, values, and data types), which enable external applications to retrieve and interpret content unambiguously.

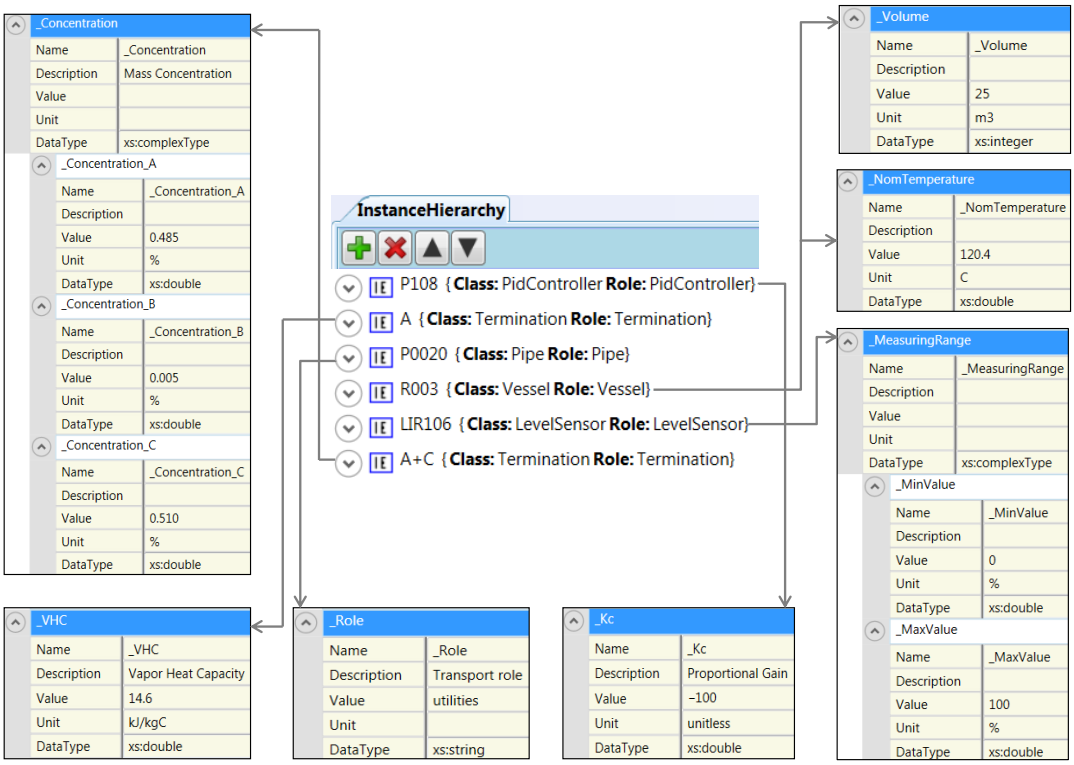


Figure 4.9: Integrated topology model with AFDD-relevant data

[4]For visualization purposes, the coordinates of control devices (see empty green boxes in Figure 4.8) were manually adjusted in this representation.

4.5.3 Data access and management in the integrated topology model

On account of enabling means for fit-to-purpose data access, AFDD information was sorted in accordance to the classification schema proposed in Section 4.4.3. Based on the resulting classified data (see Appendix F), facets and groups were systematically created within the instance hierarchy allowing thereby access to specific data across the model. Finally, time behavior properties described in Section 4.4.3.2 were added to created attributes to enable the distinctive management of static and dynamic information. As an ultimate result, an integrated topology model enriched with AFDD-relevant data and effective means for data access was obtained.

Figure 4.10 illustrates a fragment of the resulting enriched topology model. In this illustration, it should be noticed that:

- Facets are defined as child objects (InternalElements) of objects embodying plant components (e.g., R003). Groups, in turn, are defined in a separate structure with a use-case-specific group object (*Group_AFDD*) as parent. Both facets and groups are assigned standard roles derived from the *AutomationMLBaseRoleClass* library but no specific class.
- The depicted facets (*PDS_FACET*, P*KS_FACET* and *PS_FACET*) allow filtering the attributes of its parent object (R003) and provide specific data views containing only AFDD-related information.
- By means of the associated facet (*PDS_FACET*), the highlighted group (*Group_PDS*) provides specific access to all PDS information stored in the master objects referenced by its contained mirror objects, i.e., R003, S012 and E005.
- Time behavior properties implemented by means of sub-attributes enable the distinction of dynamic and static data as shown for the attribute *_Volume* of the object R003.

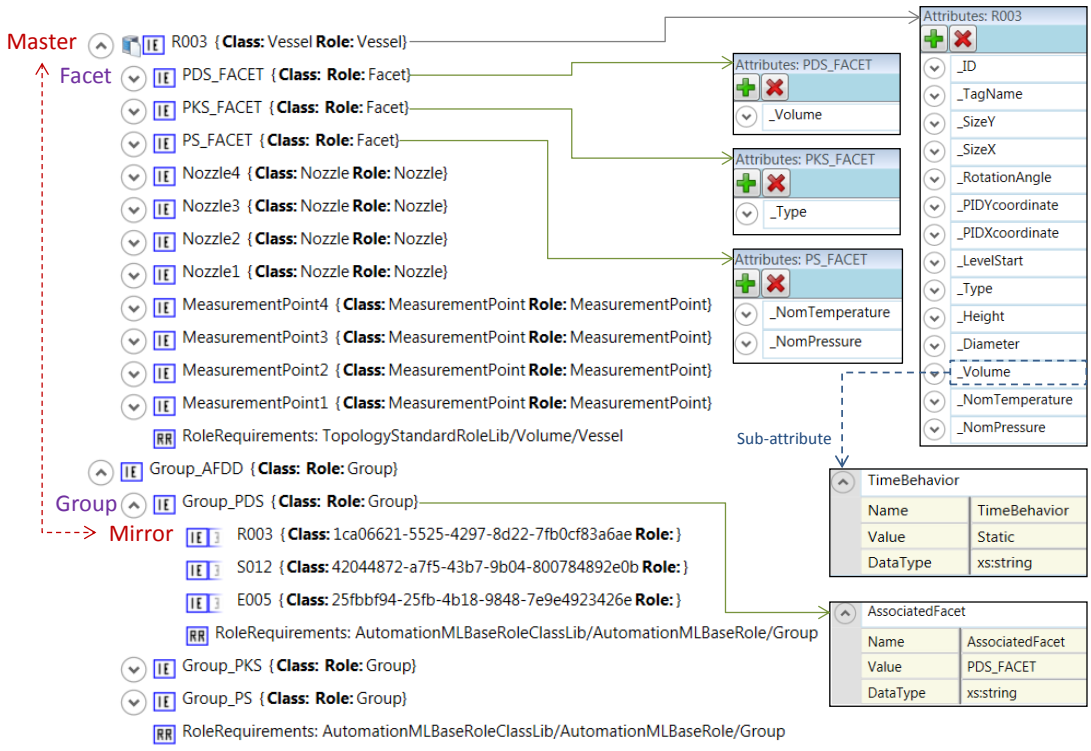


Figure 4.10: Fragment of the enriched plant topology model

4.6 Assessment

The previous section has illustrated the applicability of the proposed data integration concept within the analysis of available information on the Tennessee Eastman Process. In first instance, it was shown how the process P&ID and CLD were seamlessly merged into a single topology model, thus combining process and control information. Here, the developed algorithm correctly performed the recognition of linking artifacts among the analyzed models, merging, removing, and updating the associated XML-tree structures as required. Furthermore, it accurately generated warnings when inconsistencies were found, allowing thereby the user to opportunely inspect and correct anomalies.

In a second step, use-case-specific plant and process data, particularly AFDD-relevant information, was collected, sorted, and aggregated into the model in accordance to the identification and classification approach proposed in Sections 4.4.2.2 and 4.4.3. Although this process required manual effort, being this proportional to the number of plant components and the amount of available data, the methodology proposed herein alleviates the task by presenting a repository of relevant data sources that should be enquired when pursing enhanced plant diagnostics, as well as by presenting provisions on how collected information can be effectively integrated in an OO plant model.

An aspect worth noticing at this point is the little amount of PDS and PKS data –in comparison to PS information– available for the case study (see Appendix F). The main reason for this is the black-box nature of the TEP model. The reader should observe, however, that for real physical systems the amount of available PDS and PKS data is significantly larger.

In the last stage of the process, facets, groups, and time behavior properties were systematically aggregated in the model as a means to provide fit-to-purpose data access to specific information across the model. Analogously to data collection and aggregation, this step required manual work particularly within data classification and the subsequent association of attributes to specific facets. Yet, note that this process may be fully automated if the types, naming sets, and syntax of the aggregated attributes are strictly defined in the used information model. For the purpose of this contribution, however, the syntax and semantics of use-case-specific attributes were not constrained to a specific implementation, and accordingly were not specified in the OIM ($\rightarrow$ 3.4). As discussed in Section 4.4.2.2, this modeling decision aimed at keeping the OIM as general and versatile as possible, as well as at favoring modeling flexibility.

4.7 Chapter summary

This chapter has proposed a method for integration, access, and management of plant models and complementary process information as a basis for effective data retrieval within AoA applications. To that end, a concept for merging P&IDs and CLDs together with a data collection approach, a data classification schema, and provisions for fit-to-purpose information access have been proposed. Thereby, heterogeneous plant and process information was seamlessly integrated into a single plant topology model. In the remaining of this contribution, the exploitation of the resulting model is illustrated within two AoA use cases, specifically automatic derivation of low-fidelity simulation models and plant abnormal behavior analysis.

5. Automatic derivation of low-fidelity simulation models

5.1 Simulation models within plant modernization projects

Plant modernization plays a fundamental role in maintaining the competitiveness and increasing the resource efficiency of process industries by enabling the cost-effective enhancement of operations and the lifetime extension of existing production facilities. Over the last years, digital plant models have been increasingly used in the process industry as a basis to support modernization projects by means of simulation [HFB15]. Simulation-based engineering allows assessing new solutions early during the project, enabling thereby a smoother, faster, and safer deployment of changes in real production sites [PSF15a].

During a modernization project, a large set of technical and design steps are prone to mistakes and may lead to safety hazards and economic consequences [SRFF11]. Concerning the automation domain, errors in the control code, wrongly set variable limits, and missing interlock logic are examples of typical missteps that can cause severe damages to human operators, plant assets, and the environment. Although many of these flaws can be corrected during plant commissioning, the costs of debugging on-site are significantly higher than in previous project phases [OWU15, PHK16], and the pressure caused by upcoming project deadlines may increase the risk of overseeing critical details. Under such circumstances, the use of simulation-based engineering allows overcoming existing technical difficulties by enabling the following key-features:

- Simulation of dangerous process states without endangering personnel and plant assets.
- Execution of process simulation in different modes (stepwise vs. continuous) and speeds (real-time, slower, and faster).
- Simplified testing engineering workflows by allowing the interruption, storage, and restart of simulation sessions.
- Easy reproduction of required process states, enabling engineers to test solutions repeatedly and with reduced effort.

Despite these and other benefits offered by simulation [Ban98], the manual effort and costs required for the creation of plant simulation models are still significantly high [BF13]. At present, this drawback poses one of the main obstacles towards a wider adoption of simulation in industrial practice [OBWU15].

As discussed by Barth and Fay [BF13], one of the most time-consuming and error-prone steps during the generation of digital plant models is the collection and consolidation of necessary plant and process data. Although a significant part of the information required for this task can be found in engineering documents, most current computer-aided methods are incapable of exploiting this resource, as available data sources are typically not computer-interpretable [BSF+09] or reside in vendor-specific formats. Certainly, if the data contained in such documents could be automatically retrieved and made available to simulation tools, an important reduction of required modeling efforts would be attained.

This chapter is accordingly dedicated to illustrate how open-standard topology models (→ 3 and 4) originally stemming from legacy documentation can be used as a basis for the automatic creation of dynamic plant simulation models. Resulting models are not intended to simulate every detail of a chemical process but to represent qualitative behaviors of plant components and their influence on processes and sub-processes. Specific targeted applications include engineering tasks requiring low-fidelity simulation such as the validation of base control functions and the assessment of control-related design aspects.

The rest of this chapter is structured as follows: Section 5.2 presents fundamental concepts on simulation types, simulation languages and environments, and industrial communication protocols. Section 5.3 summarizes related work on automatic derivation of simulation models from existing engineering sources. In Section 5.4, an extended version of an existing method for generation of low-fidelity plant simulation models is presented. The extended method is evaluated in Section 5.5 within the automatic creation of a simulation model of the TEP, showing thereby the applicability of the resulting model in different automation-related tasks. Experimental results and provisions on further applications of generated models are discussed in Section 5.6, and at last, Section 5.7 provides a brief summary of the chapter.

5.2 Theoretical framework

5.2.1 Types of simulation

Simulation can be classified based on manifold criteria concerning aspects such as the nature of the used models and their accuracy. For the purpose of this work, two main categorizations are considered relevant, namely: *static vs. dynamic,* and *high-fidelity vs. low-fidelity.*

5.2.1.1 Static vs. dynamic

Static simulation is that in which the time is not considered in the estimation of simulation values. It often involves drawing random samples to generate an outcome, for which it is sometimes referred to as Monte Carlo simulation [Sen13]. Static simulation is typically used in engineering projects for activities not requiring system dynamics but only estimations of specific conditions such as maximum and minimum operation points. Examples of common applications in process industry include plant equipment dimensioning and material requirements planning [AAWU15].

Dynamic simulation, in turn, is based on the description of the time-varying behavior of a system. To that end, a digital clock mechanism keeps record of the time as state variables are updated. Dynamic models can be executed in real time to reproduce a virtual response close to the actual system, which makes them well-suited for process control applications such as tuning of automation devices and operator training [PAR14].

5.2.1.2 High-fidelity vs. low-fidelity

In the context of simulation, the term fidelity refers to the extent to which the behavior of a simulation matches the behavior of the simulated system [MG03]. Based on this definition, simulation models can be classified in two main groups: high-fidelity and low-fidelity.

High-fidelity simulation uses detailed numeric models to reproduce accurate system behaviors both qualitatively and quantitatively. A typical application of this type of simulation are the models used within operator training simulators (OTS) in the chemical and petrochemical industries.

Low-fidelity simulation, in turn, aims at depicting approximate system responses focusing on qualitative behavior and not on quantitative accuracy. Models used within cold commissioning in automation projects [VDI/VDE 3693, 2015]# are examples of this type of simulation.

In general, the required fidelity and level of detail of a simulation model depends upon its targeted application [PSF15b]. Economic costs, modeling efforts, and computational resources are key factors to be considered when defining simulation fidelity requirements.

5.2.2 Simulation languages and environments

5.2.2.1 Modelica

Modelica is a non-proprietary, object-oriented, equation-based language and open standard for the modeling of complex physical systems [Modelica Association, 2016]@. Originally introduced in 1996 by the Modelica Association, the Modelica language and its libraries have been continuously extended and increasingly adopted both in academia and industry for the dynamic simulation of mechanical, electrical, electronic, hydraulic, thermal, control, power, and process systems.

Modelica-based simulation environments are available both commercially and as open source software. Examples of current simulators include CATIA Systems, CyModelica, LMS AMESim, JModelica.org, MapleSim, OpenModelica, SCICOS, SimulationX, Vertex Wolfram SystemModeler, and Dymola. For the purpose of this contribution, Modelica and Dymola have been respectively chosen as modeling language and simulation environment.

5.2.2.2 Dymola

Dymola is a commercial Modelica-based modeling and simulation environment developed by Dassault Systemes [Dassault Systemes, 2016]@. It is an object-oriented, multi-domain simulation tool providing a graphical interface for model development along with open access to the underlying Modelica code. This feature allows users to create complex simulation models both by *drag and drop* or by coding, as well as to define their own object catalogues based on existing Modelica-standard and proprietary libraries.

5.2.3 Communication standards

5.2.3.1 OPC

The Open Platform Communication (OPC), originally known as OLE for Process Control, is an interoperability standard for data exchange in process control and other automation-related domains. OPC comprises a series of specifications developed by industry vendors, end-users, and software developers, which define provisions on access to real-time data, monitoring of alarms and events, access to historical data, and other applications [OPC Foundation, 2016]@.

Within the last years, the scope of OPC has grown beyond its original OLE implementation and restriction to Windows-based systems (OPC Classic) towards an extensible platform-independent

architecture (OPC Unified Architecture [IEC 62541, 2015]#) supporting an extended range of technologies such as .NET Framework, XML, binary-encoded TCP and SOAP-HTTPS.

In the context of this work, OPC is used as communication bridge to enable the interaction between the simulation environment and external control systems.

5.3 State of the art

In modern process industry, simulation is applied in most phases of the plant lifecycle. Among others, process design [BD06], FAT [BF13], virtual commissioning [OU14], and operator training [PAR14] are examples of the broad range of activities in which digital plant models are used as a basis to support engineering and operational processes.

Unfortunately, due to the large modeling efforts as well as the rigorous know-how required for the derivation of functional simulation models, up to now the use of simulation has not been adopted as a standardized industrial practice. In an attempt to cope with such obstacles, industry and academia have explored methods for the automatic and semiautomatic derivation of digital plant models. Barth [Bar11], for instance, presented a method for the derivation of simulation models based on P&ID data stored on CAE systems. His approach uses CAEX/AutomationML as the data format for the exchange and subsequent mapping of CAE objects into simulable instances on Modelica. Resulting simulation models are connected to external controllers via OPC to form a testbed for control code testing. The applicability of this approach was conceived for the factory acceptance test (FAT) in greenfield projects.

In a similar direction, Hoernicke et al. [HFB15] introduced an approach to automatic derivation of virtual plant models for existing facilities. This method exploits available process HMIs –transformed into CAEX representations– for generating low-fidelity models of the plant, which serve as a basis for validating control functions required during modernization activities in brownfield projects. Since HMIs are typically updated more often than P&IDs, the proposed use of HMIs offers a key advantage concerning the actuality of plant and process information used for model generation. However, the price to pay for such a benefit is a reduced applicability scope, as the approach is limited for use within the upgrade of control systems stemming from a determined vendor but not within migration projects in general. The reason for this is that HMIs are developed in proprietary environments and data formats, whose source code cannot be accessed and interpreted by third parties for automatic model derivation.

Another approach for automatic derivation of simulation models was presented by Oppelt et al. [ODLW13]. The authors of this work proposed the use of CAE design data for the generation of simple process simulation models which are interconnected with control models derived from the PCS and other domain-specific models in a co-simulation framework. Compared to the approaches presented in [Bar11] and [HFB15], this method does not use an intermediary format for the exchange of data between plant design and simulation environments. Instead, it deploys direct data exchange based on vendor-specific tools. Although such a difference allows for enhanced efficiency during model derivation, the high dependency on vendor specific solutions may constrain the applicability of the approach.

For the purpose of this contribution, the approach originally introduced by Barth [Bar11] is used as a basis for the derivation of simulation models. However, unlike Barth's and other works [ODLW13], data sources used in this contribution as a basis for model derivation do not stem from digital P&IDs

in a CAE system, but from an integrated topology model derived from legacy P&IDs and CLDs in elementary digital formats or in paper form. Another significant difference is that previous approaches, particularly [Bar11, HFB15], exclude the modeling of control equipment within the simulation environment, as its fundamental aim is to allow for testing external controllers in the process control system. The approach presented herein proposes in turn to include control devices as a part of the simulation model in addition to an alternative connection to the real controllers in the PCS. Such a difference is aimed at enabling not only for control code testing during FAT, but also for the assessment of qualitative closed-loop process responses during process control design without the compulsory need of interfacing the simulation model to external controllers.

In combination with the methods for document digitization and data integration discussed in the previous chapters, the proposed approach can serve as a basis to support the modernization of automation systems in brownfield projects (where legacy documents are only available in paper format due to the inexistence of or restricted access to digital files) as well as the development of new automation solutions in greenfield projects (for which the plant owners might have computer-interpretable plant models but would not provide them to the automation specialist owing to confidentiality reasons). Potential application benefits are expected for both automation providers and plant owners, specifically by decreasing the manual effort required for the generation of engineering models, thus allowing early verification and higher quality of project deliverables, and ultimately reducing project execution times.

5.4 Proposed method for automatic creation of simulation models

Based on the integrated topology model (→ Chapter 4), a dynamic low-fidelity plant simulation model is automatically generated. Such a model can be regarded as a cold-commissioning model [VDI/VDE 3693, 2015]#, which, by depicting qualitative process behaviors and equipment dependencies, is suitable for testing and validating basic control functions within an automation system. Note, however, that models focused in this section are in principle not suited for high-fidelity simulation, as certain required component- and process-specific information (e.g., dimensions of plant equipment and properties of processed materials) may be not available in the topology model.

In general, the overall concept for automatic derivation of simulation models presented herein[1] is valid for any object-oriented simulation environment. Yet, aiming at specificity and clarity, the main steps of the method are explained in the following by using Modelica and Dymola as modeling language and simulation environment, respectively.

5.4.1 General simulation aspects

In this approach, Modelica, specifically the *Modelica.Fluid* library [COP+06], is used as a base for the creation of flow networks and thereby for modeling analyzed production systems. Appendix G presents a list of common modeling object types predefined in *Modelica.Fluid*.

Within Modelica, two language constructs are relevant for simulation, namely (*a*) the instantiation of object instances from defined types, and (*b*) the connection of ports among object instances. The first construct is used when creating new objects. For that, relevant attributes must be defined in

[1] Partial content presented in this section has been previously published in [AHFR16]*. The author acknowledges and appreciates the contribution of Mario Hoernicke in the description of the presented method, especially in the elaboration of Figures 5.1, 5.2 and 5.3, and Table 5.1.

the respective instance constructors, as depicted in the upper part of Figure 5.1. In this way, all parameters required for dynamic simulation are set during the object instantiation process. The second construct, in turn, is used for connecting objects and thus enabling flow dynamics within the network. As shown in the lower part of Figure 5.1, connections among objects are implemented by linking consistent object ports through so-called connectivity equations.

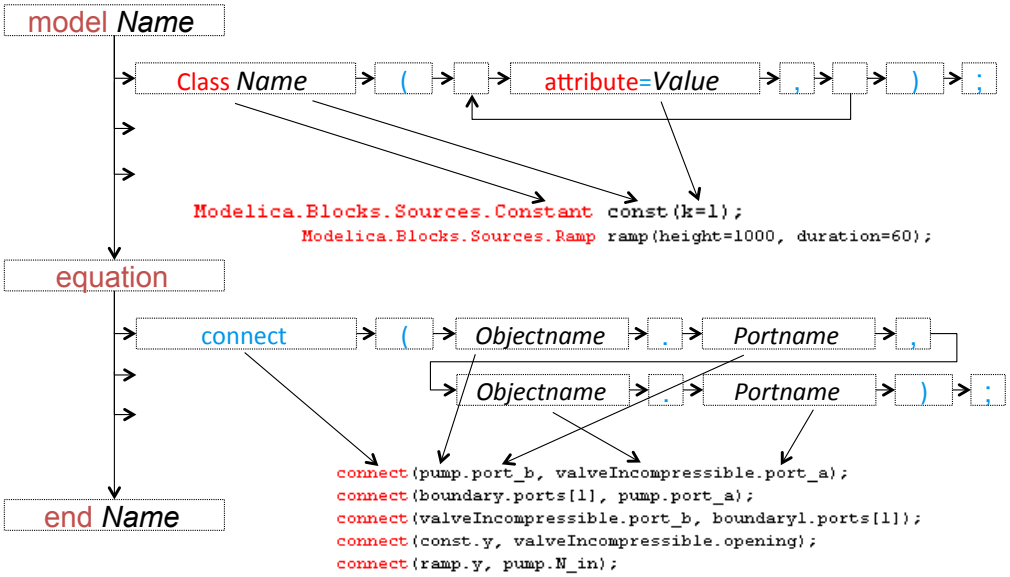


Figure 5.1: Objects and connections in Modelica

5.4.2 Assumptions and prerequisites for simulation

For the automatic creation of simulation models, some assumptions regarding object parametrization and fluid dynamics have to be made (see Table 5.1). Such assumptions are particularly necessary in view of possibly missing physical dimensioning of plant equipment. In cases, however, in which plant dimensions are a priori known by the user, these should be specified either in the topology model or directly in the simulation environment.

Table 5.1: Necessary assumptions for the automatic creation of a simulation model

Consideration	Description	Property	
		Object	**Attribute**
Simulation parameters	Default values for objects with unknown parameters	Pipe	$l = 0.2m$ $d = 0.1m$
		Vessel	$A = 0.05m^2$ $h = 10m$
		Volume	$V = 0.5m^3$
		Pump	$S_{nom} = 30000rpm$ $\triangle Pressure = 1.8bar$
		Compressor	$S_{nom} = 30000rpm$ $\triangle Pressure = 3bar$
		Controller	$K_c = 1$ $T_i = empty$ $T_d = empty$
Pressure model	Flow calculation model	Pressure differences produce a flow	
Medium	Product stream	Water	
Optimization	Model optimized for speed	Fast flow, short pipes, small vessels	

5.4.3 Algorithm for model generation

The proposed algorithm for model generation is based on a template-driven approach. For each simulation object type, a generation class containing a template of the Modelica code for the respective object instance is defined. During model generation, templates are completed according to the parameters of found object instances in the topology model, and the resulting simulation code is ultimately written into a simulation Modelica file. By way of illustration, the Modelica code template of a flow heater (modeled here as a vessel connected to a heater) and its Dymola graphical representation are respectively depicted in Figure 5.2 and Figure 5.3. In addition, each generation class provides a method to discover unused ports within object instances and thereby implement consistent connections via connectivity equations.

```
Modelica.Fluid.Vessels.ClosedVolume _" + ModelicaConstants.NAME +
"\n(use_HeatTransfer=true, " + "\n" +
"redeclare model HeatTransfer = Modelica.Fluid.Vessels.BaseClasses.HeatTransfer.IdealHeatTransfer, " +
"\n" +
"use_portsData=true," + "\n" +
" V=" + ModelicaConstants.VOLUME + @", " + "\n" +
"p_start=100000, " + "\n" +
@"portsData={" + ModelicaConstants.PORTS + @"}," + "\n" +
"nPorts=" + ModelicaConstants.NPORTS + @", " + "\n" +
"redeclare package Medium =" + ModelicaConstants.MEDIUM + ", " +
"\n" +
"use_T_start=true, " + "\n" +
"T_start=300/*," +
ModelicaConstants.ALL_PORTS_M_FLOW_TANK + "*/)" + "\n" +
"annotation (Placement(transformation(extent={{" +
ModelicaConstants.X_MIN + "," + ModelicaConstants.Y_MIN + "},{" +
ModelicaConstants.X_MAX + "," +
ModelicaConstants.Y_MAX + "}})));" + "\n" +
"Modelica.Thermal.HeatTransfer.Sources.FixedHeatFlow _"+ ModelicaConstants.NAME +
"fixedTemperature(alpha=-1, Q_flow=100000, T_ref=" + ModelicaConstants.T_START + ");\n";
```

Figure 5.2: Modelica code template of a flow heater

As a first step for the generation of a simulation model, the algorithm examines the topology model identifying object instances that can be simulated based on existing (predefined and user-defined) Modelica simulation objects. Every identified simulable object is subsequently converted into a simulation instance, where the respective instance-specific attributes are completed in the corresponding code template. In this fashion, every created simulation object implicitly generates its own Modelica code and stores it internally. It is important to observe that Modelica specific naming conventions must be followed at this stage. For example, instance names are neither allowed to include the characters "-" and "/" nor to start with a numerical character. Therefore, the algorithm replaces forbidden characters with empty strings and places an underscore in front of each instance name to ensure that no name starts with a number (see e.g., *_Tank* in Figure 5.3).

As a next step, material connections between created simulation objects must be established. Initially, a consistency check method verifies that every pipe object has exactly two material connection points, one in each termination. Cases violating this rule are prompted to the user by warnings. Once a given pipe has been verified, the related objects are asked to provide a free material port for every connection. By using its own port discovery method, every object is aware of its number of ports and their availability. Thus, objects decide which specific ports are to be connected in accordance the connectivity information stored in the topology model. The respective object connections are implemented through connectivity equations linking the corresponding object and pipe ports. The process is repeated for every pipe object until all material connections have been generated.

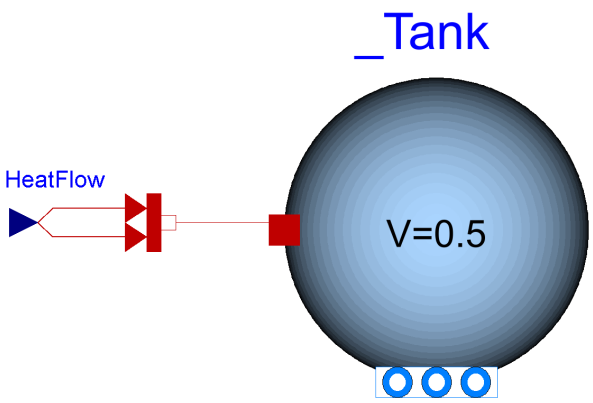


Figure 5.3: Modelica graphical representation of the template code presented in Figure 5.2

After the creation of material links, the next method step deals with the generation of information connections, specifically sensing and actuation points. Such connections are implemented as numeric blocks linked to sensor and actuator ports by connectivity equations. By exposing their value, numeric blocks are read and written by the communication interface, allowing thereby the required closed-loop interaction between the control system (PCS or external controller) and the process model. Likewise, aiming at verifying that control signals and corresponding actions sent by the controllers to the process are correctly received and executed, feedback points are created by following a similar approach. The opening of a valve, for instance, is verified by the feedback values fetched from the respective control signal and the final position of the valve (see *OP feedback* and *Aperture feedback* in Figure 5.4). Finally, information connections depicting links within internal control loops are established. To that end, sensor, actuator, and controller objects use their port discovery methods to define the required connectivity equations by analyzing the directionality and semantics of the connected signals. By way of illustration, a controller object would respectively connect sensor, actuator and master controller signals to its PV, OP and remote SP ports. Figure 5.4 illustrates the different information connections comprised by the control communication structure used in this approach. A further analysis of this topology is presented in Section 5.4.4.

As a last generation step, the thermal dynamics of the system are defined. Accordingly, a heat source is created for and connected to every heating object, i.e., an object aimed at heating or cooling the medium (e.g., a boiler). Connections between heat sources and heating objects are implemented through connectivity equations by using energy ports. Figure 5.3 shows an example of a heat source connected to a closed volume used to heat up a fluid.

After the execution of the aforementioned steps, the generated simulation code is transferred to a Modelica text file, which can be imported and executed on different simulation environments including Dymola (→ 5.2.2.2) and Open Modelica [OSMC, 2016]@. The resulting simulation model constitutes a low-fidelity representation of the controlled process suited for the assessment of qualitative responses, and which can be alternatively interconnected to an external PCS for the verification of basic control functions during FAT.

The following section discusses technical and application aspects on the different control execution alternatives, i.e., internal controllers and external PCS, enabled by the approach.

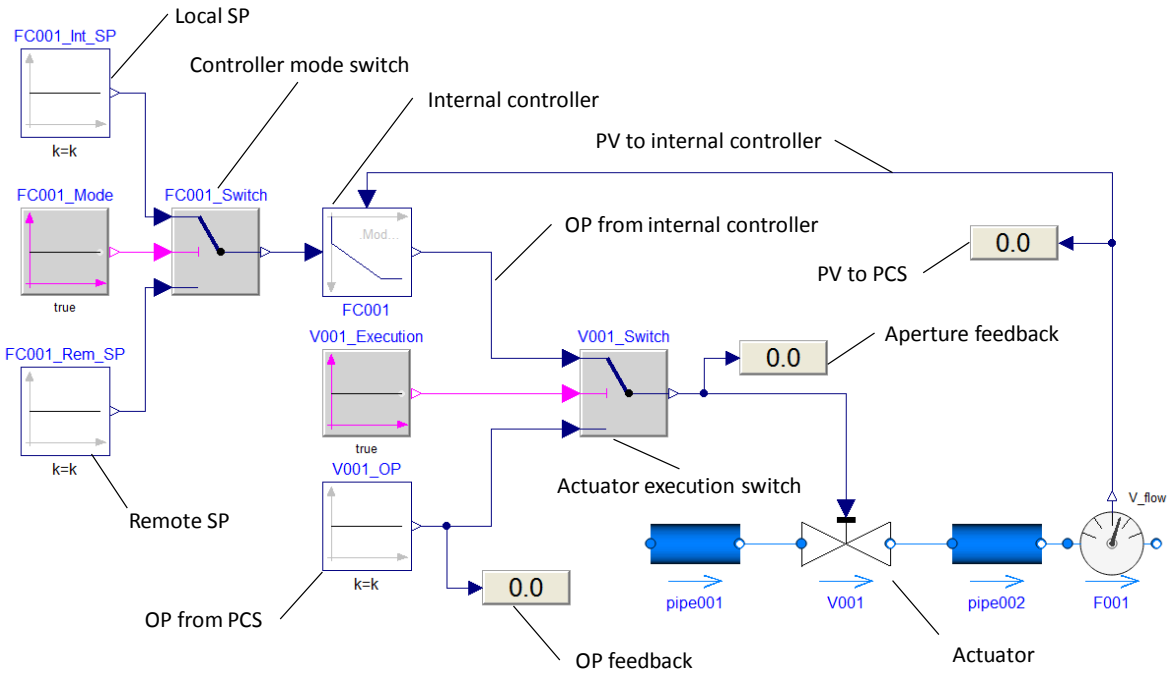


Figure 5.4: Communication structure used for process control in the model

5.4.4 Control execution

As previously discussed in this chapter, this approach incorporates internal controllers as built-in model components in addition to structures for the connection of external control systems. This allows the execution of control actions both in internal and external modes.

In order to enable the selection of the control execution mode, each controlled actuator in the model is provided with an execution switching mechanism (see *V001_Switch* in Figure 5.4). In this construct, a binary signal (*see V001_Execution*) allows changing the state of the switch, causing the respective actuator to take the OP value either from a built-in model controller (internal mode) or an external module in a PCS (external mode).

5.4.4.1 Built-in model controllers (internal execution mode)

As depicted in Figure 5.4, a control block is formed by a PID object connected to a switching mechanism (see *FC001* and *FC001_Switch*). In this configuration, the PID instance deploys the proportional-integral-derivative algorithm used for control, while the switch instance allows defining the mode (automatic or cascade) in which the controller operates. By means of a mode selection variable (see *FC001_Mode*), a controller object can be configured to use a local set point in automatic mode (see *FC001_Int_SP*) or a remote set point in cascade mode (see *FC001_Rem_SP*).

Internal controllers enable the simulation of qualitative closed-loop process behaviors sufficing the fidelity requirements of different automation engineering tasks without the need of interfacing the real control system. Examples of typical applications include the verification of control action directions (i.e., inverse vs direct) as well as the qualitative assessment of control algorithms.

In addition, internal controllers allow driving the process to desired operation points (e.g., half or full load) and facilitate matching the states of the simulation and the real control system if an external connection is required. This is particularly useful for ensuring model convergence in

complex open-loop unstable processes which cannot be easily driven by manual means due to their large set of actuators as well as the high degree of causal dependencies.

5.4.4.2 External PCS (external execution mode)

One of the fundamental aims of the generated simulation model is to allow the validation of PCS control functions. For that, a copy of the real external PCS must be interconnected to the simulation environment through a suitable communication interface or bridge. Such interface must enable a bidirectional data exchange procedure in which process variables are read from the simulation environment (sensing and feedback points) and send to the PCS, and manipulated variables are read from the PCS and send to the simulation environment (actuation points).

As proposed in [BF13], OPC (→ 5.2.3.1) is used to implement the required interface in this approach. The use of this standard as communication technology is justified not only by its wide use in current PCSs but also by the built-in OPC capabilities provided by several simulation environments, including Dymola.

As a fundamental requirement for the data exchange between the PCS and the simulation model via OPC, a mapping between simulation and control system variables must be defined. In many cases such a mapping can be automatically generated provided that a naming convention based on P&ID tag identifiers is used to name control modules in the PCS. Such tags are considerably similar to those used for naming objects in the simulation model, as the latter are automatically captured from a P&ID-based topology model and only eventually modified by prefixes or suffixes. Accordingly, if the used naming convention is known, different methods –e.g., regular expressions (→ 2.2.3.5)– can be used to automatically create the required variable associations within the mapping.

```
<Mapping>
    <cycleTime>10</cycleTime>
    <direction>ReadLeft</direction>
    <leftVariable>
        <Name>_PIR108</Name>
        <OPCPath>ModelVariables._PIR108.p</OPCPath>
        <Type>Real</Type>
        <ScaleMin>0</ScaleMin>
        <ScaleMax>0</ScaleMax>
    </leftVariable>
    <rightVariable>
        <Name>PIR_108</Name>
        <OPCPath>Applications.P_PCS_16.AppCode1.TEPTopology.PIR_108.pIO.IOValue</OPCPath>
        <ScaleMin>0</ScaleMin>
        <ScaleMax>0</ScaleMax>
    </rightVariable>
</Mapping>
```

Figure 5.5: Example of mapping between PCS and simulation variables

For implementation purposes, in this approach the required mapping is defined as an XML file. Within it, corresponding variables are characterized by relevant attributes and grouped pair-wise as *leftVariable*(*simulation*)-*rightVariable*(*PCS*). In addition, the directionality of the information flow is described in the form [*Action* (*Read/Write*)][*Variable* (*Left/Right*)]. Figure 5.5 depicts an example of the XML mapping required to read a pressure value from the simulation model and send it to the PCS. Note that during runtime, the mapping file is read by the OPC bridge, which executes the event-based data exchange.

5.5 Validation

The approach is validated within the automatic generation of a low-fidelity simulation model for the TEP (→ 1.4). As no external control system is publicly available for this process, the applicability of the resulting simulation model is illustrated only for the assessment of qualitative process responses in internal control execution mode. For results obtained during the factory acceptance test of an external PCS in a real industrial site, the reader is referred to [AHFR16]*.

5.5.1 Generated low-fidelity model

Based on the TEP integrated topology model presented in Chapter 4 and by using the algorithm described in Section 5.4, the process simulation model shown in Figure 5.6 was automatically generated[2]. Here, connection blocks (i.e., sensing, actuation, and feedback points) as well as information links and switches have been omitted for the sake of readability. These objects, however, are part of the model (i.e., they are instantiated in the actual Modelica code, but without visualization parameters) and are accordingly used during simulation. The depicted model forms the testbed used in the sequel for several applications including the analysis of open-loop and closed-loop process behaviors, and the verification of controller actions.

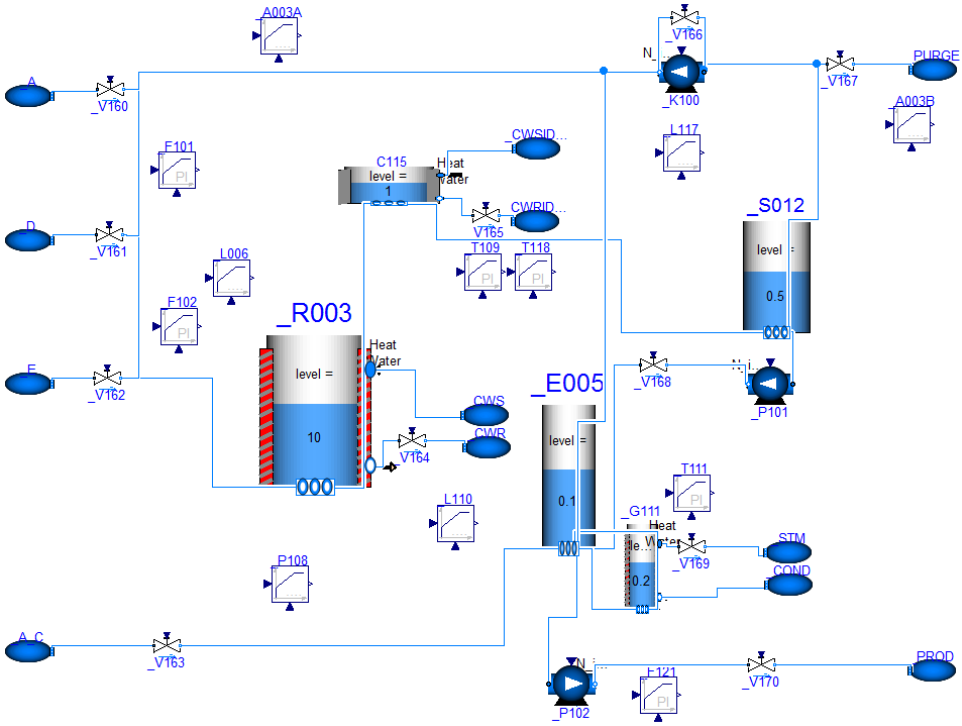


Figure 5.6: Simplified low-fidelity simulation model of the TEP

5.5.2 Application examples

5.5.2.1 Analysis of open-loop process behavior

The trends obtained during a series of open-loop flow dynamics tests are presented in Figure 5.7. In the conducted trials, a determined amount of material was fed from the separator S012 into

[2]Pipes have been partially rearranged in the figure for enhanced visualization.

the stripper E005 (see Figure 5.6, right hand side). The valves V163, V169 and V170 were fixed closed, while the aperture of the control valve V168 was varied from 0 % to 10 %, 50 % and 90 % in three different trials. For all tests, the initial separator and stripper levels were respectively set to 8 m and 1 m, the pump P101 was operated at nominal speed (30000 rpm), and the pump P102 was out of operation. Obtained results show that the separator and stripper levels reach faster the equilibrium value (approx. 4 m) for larger valve apertures.

A more detailed analysis of Figure 5.7 allows observing a flatter curvature in the trends corresponding to the stripper level (see brown, purple, and blue curves). This behavior reveals a concurrent outflow leaving the stripper base and accumulating in the holdup of reboiler G111 as the stripper is being filled.

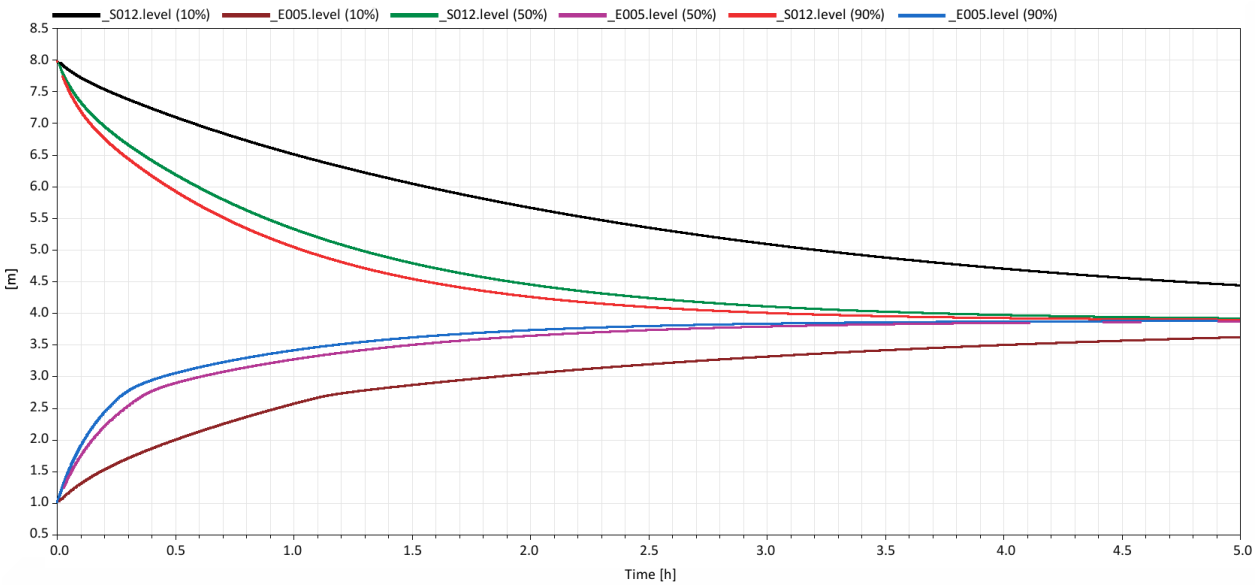


Figure 5.7: Levels of the separator S012 and stripping column E005 for different valve apertures. 10 % (black and brown), 50 % (green and purple), 90 % (red and blue)

Open-loop tests, as the ones shown above, provide a fair first approximation of the dynamics (e.g., time constants) of modeled systems. Such information might be used as an initial basis for controller tuning as well as for the definition of benchmarks required for control performance monitoring (e.g., settling times). Evidently, the accuracy of obtained results depends upon the equipment dimensioning and the specification of material properties.

5.5.2.2 Assessment of closed-loop process behavior

With the aim of assessing and comparing the qualitative closed-loop response of the considered regulatory control structure, some tests with different controllers, namely PI and P-only controllers, were conducted. For the purpose of the experiment, the response of the stripper temperature control loop (TIR111-T111-V169) was considered for evaluation. As preliminary step, the system was brought into steady state by setting the following process conditions and parameters:

- The initial stripper E005 and reboiler G111 temperatures were both 26.85 ºC.
- A liquid flow of material A+C (2 bar, 18 ºC) was fed into the stripper E005 by fixing the aperture of valve V163 to 40 % (manual mode).

- The aperture of valve V168 was manually fixed to 0 % to avoid a material flow from the separator S012 into the stripper.
- The flow controller F121 was set in automatic mode to establish a product outflow equal to the inflow of material A+C by adjusting the aperture of V170.
- As the inflow and outflow to the stripper were equal, the level of the stripper was kept constant (6.3 m) during the test.
- The heating medium used in the reboiler tube was water with a temperature of 99 ºC and a pressure of 3 bar.
- The initial position of the control valve V169 was 1.2 %.
- Pump P101 was out of operation, while pump P102 was operated at nominal speed.
- Both tested controllers had a initial set point of 26.85 ºC and the following parameters:
  - PI: Kc = 1.5; Ti = 1800 s; span 0-100 %.
  - P-only: Kc = 2.1; span 0-100 %.

During the experiment, a step set point change of 10 ºC (from 26.85 ºC to 36.85 ºC) was conducted in both controllers. Figures 5.8 and 5.9 depict the controller responses as well as the respective changes in the process temperature observed in the conducted trials. As can be observed in Figure 5.8, the PI controller drove the control valve V169 (red trend) to approximately 18 % and thereupon to 5.2 % with a small transient. The resulting temperature response exhibited an initial overshoot of approximately 2.45 ºC (see temperature peak of 39.3 ºC in blue curve) followed by a small undershoot of 0.35 ºC, and finally reached the desired temperature set point (36.85 ºC) without steady state error.

Figure 5.9 revealed that the P-only controller, on the other hand, drove the control valve to nearly 22.2 % aperture and subsequently to 4 % with no oscillations (see purple curve). However, the temperature set point (36.85 ºC) was never reached due to a permanent error of 1.35 ºC, as can be observed in the green trend of the figure.

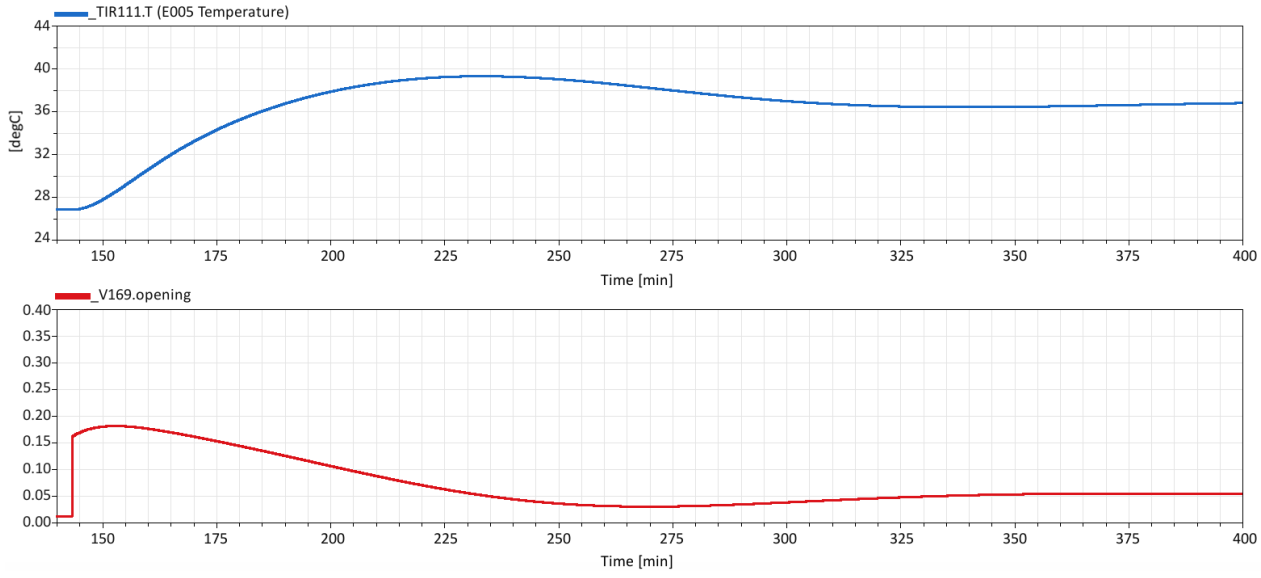


Figure 5.8: Closed-loop process responses. PI temperature control.

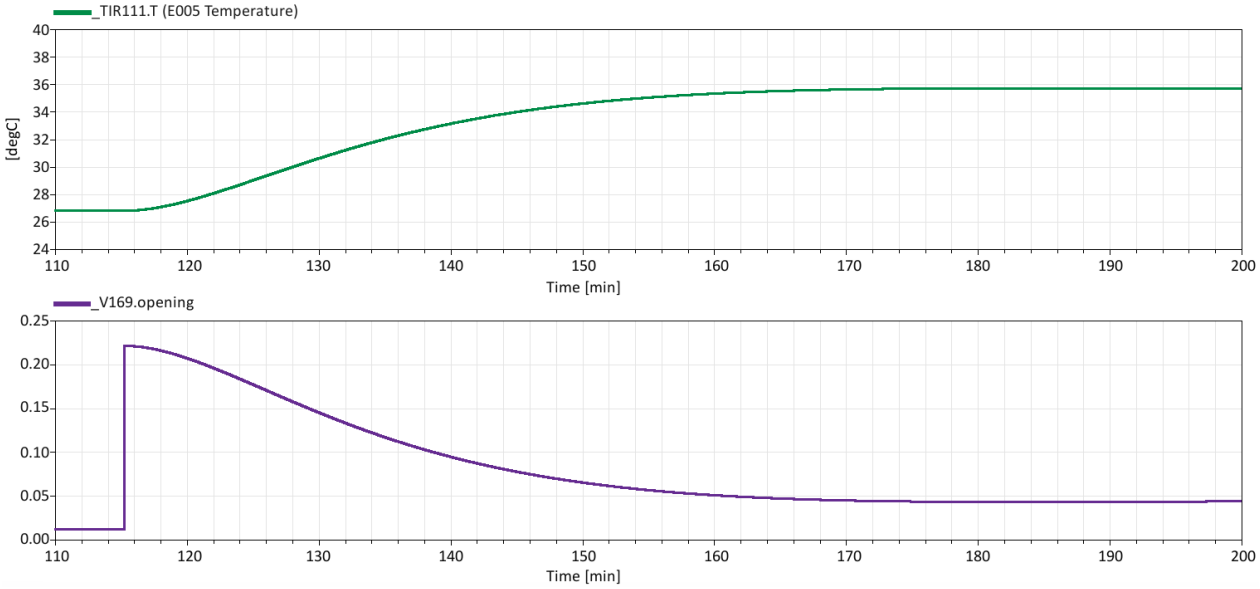


Figure 5.9: Closed-loop process responses. P-only temperature control.

Obviating the different observed response times, which can be influenced by adjusting the control parameters, a quick inspection of the results reveals that the main difference between both controller responses is the existence of a permanent error or steady state offset in the response of the P-only controller. Yet, the small magnitude of this error supports the claim of Luyben [Luy96], who stated that for the control of the TEP, PI controllers do not offer significant advantages over P-only controllers owing to the intrinsic integrating nature of the process.

While the above low-fidelity closed-loop tests are not suited for fine controller tuning, automation engineers and process experts can benefit from the results gathered in conducted simulations by getting insight into the dynamics of the controlled process. The identification of process-control interactions as well as the study of the behavior of different control algorithms are examples of the specific applications allowed by this type of tests.

5.5.2.3 Verification of controller action directions

Determining the required controller direction (i.e., direct-acting vs inverse-acting [Wad04]) might be in many cases an error-prone procedure as control signals and actuator failure modes (e.g., fail-closed vs fail-open) may differ from application to application. In order to evaluate the capabilities enabled by the approach to identify wrongly set controller directions, the response of a multi-level temperature control structure, specifically the control cascade formed by the master controller T118 and the slave controller T109, was studied. This control loop regulates the temperature of the separator (S012) by setting the temperature of the reactor (R003), which in turn is controlled by manipulating the aperture of the cooling water valve (V164).

For the purpose of the trial, a given material flow between the reactor, the condenser (C115), and the separator was established under the following process conditions:

- Flow controllers F101 and F102 were set in automatic mode with local set points 1 kg/s and 0.85 kg/s, respectively.
- Valves V160, V163 and V170 were manually closed to prevent flows of A, A+C and PROD materials entering and leaving the system.

- The initial reactor and separator levels were respectively 5 m and 0.5 m.
- Initial temperature of the process fluid material was 26.85 ºC.
- Valve V166 was fixed closed and compressor K100 was taken out of operation to avoid material recirculation entering the reactor.
- Cooling water valve V164 was initially closed.
- Parameters of both master and slave controllers were over-dimensioned aiming at fast, almost instant controller responses. The following parameters were accordingly used:
  - T118: Inverse-acting, Kc=10; Ti=3000; span: 0-100 ºC.
  - T119:Inverse-acting, Kc=0.1; Ti=1000; span 0-100 %.

With the system in steady state, the cascade loop was activated with a local set point of 36.85 ºC, i.e., 10 ºC higher than the initial value of the controlled process variable. Figure 5.10 illustrates the output of both controllers as well as the respective responses of the separator and reactor temperatures. A quick inspection of the obtained trends allows observing a decrease of both temperatures (blue and magenta curves) instead of the aimed increase. This behavior indicates that the acting direction of the control loop has been mistakenly configured, which implies that the proportional constant of one of the controllers has the wrong sign. But which one? the master or the slave?

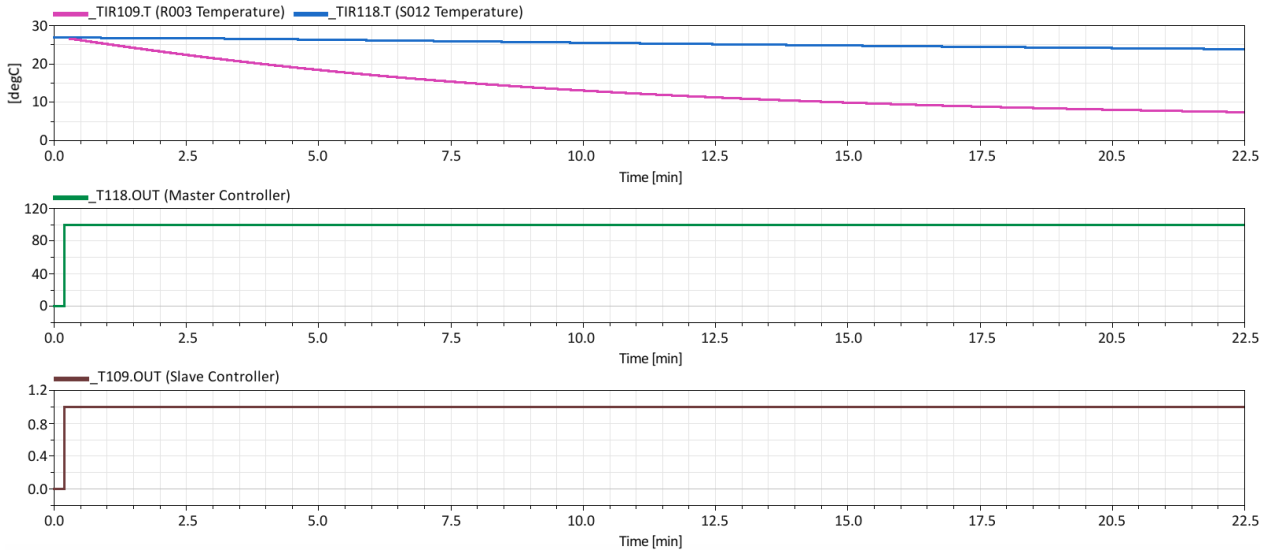


Figure 5.10: Simulated controller actions. Detection of a wrong cascade configuration

Certainly, the best way to solve this inquiry is by comparing *what should have happened* with *what actually happened* in response to the intended temperature change. As synthesized in expression 5.5.1, in order to achieve the desired increase in the separator temperature, it is required that the reactor temperature also increases, which in turn demands reducing the aperture of the cooling water valve.

$$(+)S012Temperature \leftarrow (+)R003Temperature \leftarrow (-)V164Aperture \tag{5.5.1}$$

As depicted in the green curve of Figure 5.10, the master controller (T118) reacted to the intended temperature change by driving its output to the span limit (100 ºC), which caused the slave controller (T109) to increase its output (brown curve) also to the span limit (100 %), and as consequence to increase the aperture of the cooling water valve (V164). While the action of the master controller matches the expected behavior (i.e., increasing the separator temperature requires demanding more reactor temperature), the action of the slave controller goes against the required response (increasing the reactor temperature requires less valve aperture and not more as observed). This brief analysis allows pinpointing the slave controller as the source of the inverted cascade control action, as well as concluding that the sign of its proportional constant (Kc) must be inverted (i.e., negative gain, direct-acting direction). Note that even with the correct cascade control action, the process cannot reach the desired temperature set point under the predefined system conditions. The reason for this is that the cooling water valve cannot be further closed (initially at 0 %) to produce the required increase of the reactor and separator temperatures. For the purpose of this test, however, this limitation is not relevant.

Simple trials as the one shown in the above example allow control engineers to mitigate the impact of wrongly configured control directions on the process stability before actual commissioning.

5.6 Assessment

5.6.1 Considerations on the approach scope and limitations

Section 5.5 has demonstrated the applicability of the proposed approach as well as some potential benefits of its use within basic automation engineering tasks. Until this point, however, no discussion regarding the approach scope and limitations has been provided. This subsection is accordingly dedicated to this aim.

5.6.1.1 Required manual steps

Despite being automated to a large extent, the proposed method still requires certain manual steps. Among them:

- *Variables mapping.* Based on a naming convention, the mapping between variables in the simulation model and the PCS can be automated. However, the definition of a naming convention requires some manual steps in which the user must specify the segments of the topology tags to be used as pre- and suffixes in the PCS tags as well as the delimiters used between characters.
- *Manual inspection.* Despite stemming from an inspected plant topology representation, the generated simulation model might contain inconsistencies. Aspects of particular relevance that must be verified and eventually readjusted include the elevation of equipment and respective ports, as well as the correct connection of these to material flows and process utilities (e.g., steam and cooling water).

5.6.1.2 Actuality of captured information

As previously discussed in this chapter, the proposed approach can be applied in both greenfield and brownfield projects. Yet, a relevant aspect to be considered for use in brownfield is the actuality

of the plant and process data captured from available legacy documents. As pointed out by Hoernicke et al. [HFB15], existing plants change their physical structure as sensors and actuators are exchanged, pipes added or removed, plants modules replaced, and automation or communication systems upgraded. In theory, all those changes should be reflected in a back-documentation process, where every change is registered as an adjustment in existing plant documents, such as P&IDs and CLDs. In industrial practice, however, that might not always be the case and accordingly the actuality of plant and process data must be considered.

In the context of this work, it is assumed that available P&IDs and CLDs used for the derivation of plant topology models represent the latest status of the plant, or at least, a state that can be easily updated to the last version by following a management of change (MOC) list or applying other methods for change management [GF13, SFTF11]. The required effort for such an update is assumed significantly lower than the effort required for a complete manual generation of a simulation model. In cases in which the actuality of source documents cannot be confirmed, process data, specifically process signals, might be used to assess the validity of resulting models by comparing the qualitative behavior of real and simulated process signals. Note, however, that for large systems it might be difficult to find the source of discrepancy by following this approach.

5.6.2 Levels of detail and additional use cases

The required level of detail of an adequate simulation model depends directly on the targeted application. For the purpose of the use cases addressed by the approach –validation of basic control functions, and assessment of qualitative process responses–, low-fidelity plant simulation models depicting qualitative plant behavior suffice the required level of detail. Nevertheless, with few modifications and the manual addition of relevant data, automatically generated models might be an adequate basis for more detailed use cases including fault detection and diagnosis [DV00a] and operator training simulators [CSD06, PAR14].

In general, the system structures of low-fidelity and high-fidelity simulation models are identical, i.e., the plant components and their connectivity are basically the same in both cases, as they are derived from the same plant sources, e.g., from the same P&IDs. The main difference among the models is the parameterization of the equipment, the processes, and the products being processed. While a high-fidelity simulation model contains complete component- and process-specific parameters, a low-fidelity simulation model uses standard values for all those components and materials for which no data is available or known (→ 5.4.2). Therefore, assuming that the mathematical component models in the simulation environment are detailed and accurate enough, adapting automatic generated models for high-fidelity simulation is mainly a matter of parameterization.

A possible alternative to systemize the collection of required plant and process data, supporting thereby the model parameterization process, relies on the use of formalized process descriptions [VDI/VDE 3682, 2014]#. As shown in [ACHF14]* and [ASC+14]*, relevant data can be consistently integrated into formalized knowledge models. Based on such representations and only with minor modifications in the proposed algorithm for model generation, it would be possible to automatically create simulation models fulfilling the required level of detail of further use cases.

5.6.3 Applicability in industrial practice

As revealed by recent market analysis reports [Mor16], the installed base of legacy automation systems is highly valuable for automation players, given that industrial process plants typically have a useful lifecycle of over 20 years. It is, therefore, critical for automation companies to continue to support and modernize their installed bases, which not only represent a major revenue stream, but also a means to defend market share.

Knowing that plants older than 20 years represent a substantial portion of the global installed automation base, and conservatively assuming that 50 % of those plants have engineering documents in hard copies or in elementary digital formats, the amount of automated process facilities that can benefit from the combined use of methods for information capturing, formalization, and integration ($\rightarrow$ 2, 3, and 4) and the proposed approach for model derivation is more than significant. In addition, a considerable number of younger and new plants, in which confidentiality restrictions hinder the access to high-level digital records, are also expected to benefit from this approach.

5.7 Chapter summary

This chapter has presented a method for the automatic generation of low-fidelity plant simulation models based on formalized plant topology descriptions stemming from legacy piping and instrumentation diagrams and control logic diagrams. It has been discussed and shown how obtained simulation models can be applied for the verification of control functions within FAT as well as for the assessment of qualitative process responses required during basic automation engineering tasks. In addition, provisions on how to adapt the level of detail and extend the applicability of the approach towards further applications have been analyzed.

Combined with the methods for information capturing, formalization, and integration presented along Chapters 2, 3, and 4, the method proposed herein is the first one allowing for the exploitation of legacy plant and process data sources as a basis for automatic model derivation. Such a fundamental difference enables a significant reduction of time and modeling efforts during (re)engineering tasks both in greenfield and brownfield projects within a number of industries including chemical, pharmaceutical, oil and gas, and power generation.

The next chapter discusses a further AoA use case which can benefit from the exploitation of integrated plant topology models, i.e., plant abnormal behavior analysis.

6. Plant abnormal behavior analysis

6.1 The complexity of plant monitoring in process plants

Automated analysis of plant anomalies has become a central requirement of new generation control systems conceived to fulfill the operation demands of large-scale process facilities. The reason for this is twofold: first, in such production scenarios the fault of a single component might propagate and cause the malfunction of the whole plant, leading to severe safety and economic consequences [LJJ16]; and second, due to the multiple interactions between plant assets and control structures, manual fault diagnosis turns out to be a highly complex task [BJC+10].

Only to provide an insight on the complexity of disturbance propagation analysis in process systems, consider a fault detected in the Tennessee Eastman Process (→ 1.4), specifically a loss of the cooling water supply in the condenser C115 [Bro15]&. Assume for the purpose of this example that the root-cause of the problem is known beforehand and, based on it, an analysis of the observed plant symptoms must be conducted. A detailed account of the phenomena occurred during this anomaly is presented below. Note that ordered pairs (x, y) mentioned in the following refer to the position of measurements in Figure 6.1.

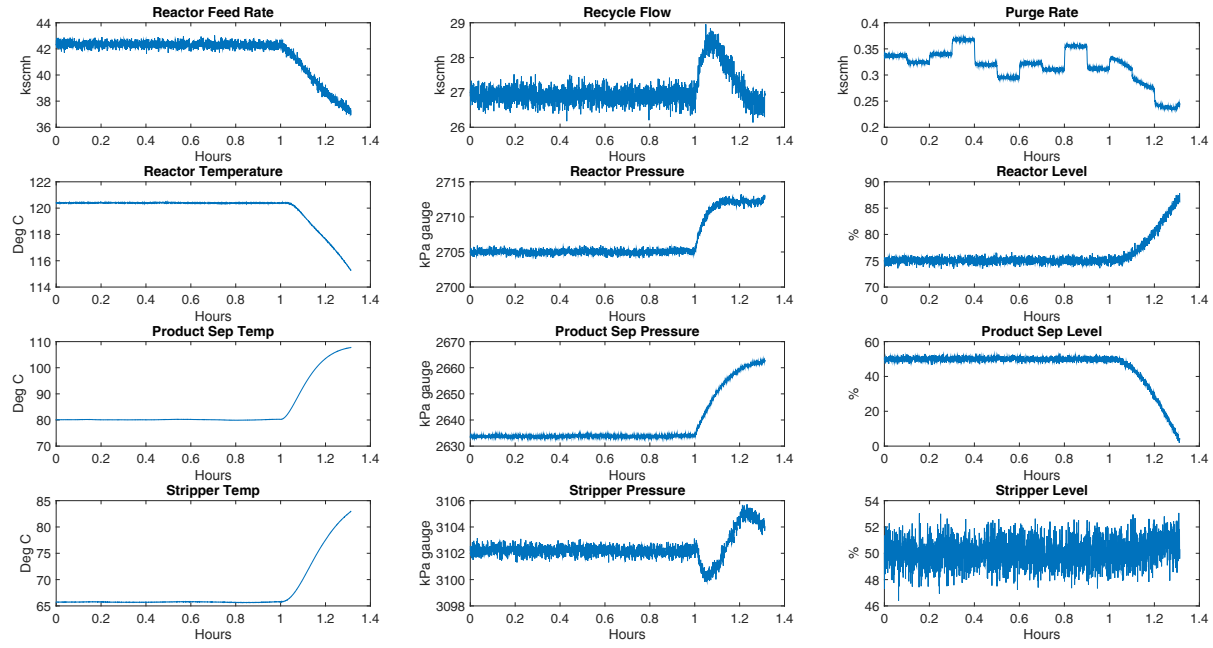


Figure 6.1: Fault propagation analysis in the Tennessee Eastman Process. Signal measurements captured during a loss of cooling water supply in condenser C115 at t=1h

After the loss of cooling water supply in condenser C115 at t=1h, the stream entering the separator S012 increases sharply in temperature. As a consequence, both separator temperature (3,1) and pressure (3,2) increase in accordance with the ideal gas law (IGL). The differential pressure driving the flow between the reactor R003 and the separator S012 (unmeasured) decreases as the separator pressure increases. This reduces the outflow from R003 without altering its inflow, leading to a reactor pressure increase occurring in parallel (2,2). Temperature in the reactor decreases (2,1) due to the action of a control loop which increases reactor cooling water to counteract the separator temperature increase. This effect outweighs the IGL effect of the reactor pressure increment. The

temperature decrease and pressure increase together shift the Vapor-Liquid Equilibrium (VLE), favoring the condensation of vapor and as a consequence the increase of the level in the reactor (2,3). This effect is reinforced by the decrease in reactor outflow, which causes a composition change favoring heavy components. Although the observed temperature decrease causes a reduction in the reaction rate, which should tend to decrease the reactor level as the production of heavy components slows down, this effect is outweighed by the vapor condensation and the composition change.

Increased temperature in S012 drives VLE towards favoring vapor species and causes the separator level (3,3) to lower. The increase of vapor species in S012 results in a decrease of the concentration of the inert B in the vapor leaving the head of the separator. This in turn causes a reduction of the purge rate (1,3) due to the closing action of the concentration control loop driving the purge valve. Meanwhile, the pressure of the stripper E005 exhibits an initial negative response followed by a counter-response (4,2). The initial decrease is related to the control loop which reacts to the high reactor pressure and consequently reduces the flow of the A+C feed into S012. While the A+C feed eventually stabilizes as the reactor pressure steadies, the temperature of E005 continues to rise (4,1) due to the hot stream from S012. An IGL effect eventually overcomes the initial stripper pressure decrease, leading to a higher steady state pressure.

Even though the temperature shift in E005 is just as strong as it is in S012, there is virtually no effect on the stripper level (4,3). As liquid flow rates in and out of the stripper are maintained roughly constant, it can be inferred that the stripper VLE has not been significantly altered. The reason for this can be ascribed to the composition difference between the two vessels. Due to the separator pressure increase, the recycle flow also increases (1,2). This recycle is only one of five streams conforming the reactor feed rate (1,1), which has a net decrease due to the reduction of the dominant streams, D and E, caused by the control loop response counteracting the increase in reactor level.

The above example evidentiates the complex interactions occurring between plant components and control structures during a plant disturbance, as well as the significant amount of process knowledge required for a detailed analysis of the observed plant behavior. During plant operation, operators and process experts are often confronted with similar faulty scenarios in which anomalies must be timely diagnosed and corrected to avoid dangerous operation states and costly plant shut-downs [WYCS15]. The understanding of disturbance propagations across plant units and pipelines based on observed qualitative behaviors, process alarms, and previous experiences is the main resource used by plant personnel to analyze anomalies and formulate causal hypotheses.

While in many cases installed monitoring and alarm systems allow operators pinpointing the origin of plant disturbances with high resolution, the propagation of faults through the process might trigger multiple alarms and result in cases difficult to diagnose. The order by which alarms appear on plant alarm systems does not necessarily correspond to the physical propagation of faults, as abnormalities can propagate through equipment without immediately triggering some alarms [RCHH16]. In certain scenarios, particularly in processes where large amounts of alarms are configured without a proper alarm philosophy [ISO 18.2, 2009]#, operators might be overwhelmed by unmanageable amounts of notifications within short time intervals (see *alarm flood* or *alarm shower* [NAMUR NA102, 2003]#), which might prevent them from isolating the fault root-cause and taking the required counteractions to keep the plant within safe operation boundaries [SCTF13, WYCS15]. A thorough analysis of industrial incidents, such as the Texaco refinery explosion in Milford Haven (1994), shows that alarm floods are closely related to critical accidents [HB07, WYCS15].

In view of these and other related safety problems, different methods for supporting operators and process experts during plant fault diagnosis have been proposed. Among them, a large amount of approaches are based on data-driven methods (e.g., [BTM05, Zha08, TBBT09, JFVH12]). However, the dimensions of process plants and their vast amount of process signals have limited their usability in large-scale processes, as calculations required for disturbance detection and root-cause analysis become time- and resource-intensive. In this regard, Vorst et al. [VHG+14] noted that collecting production data in process plants can amount to millions of signals and events in a single month, and that the analysis of this amount of data requires complex diagnosis algorithms which up to now cannot satisfy real-time requirements. Other existing approaches have used expert systems [RW93, RW95, CRB01, Chr15], and analytical methods [MSSdC05, SFP02] to diagnose process anomalies based on observed plant symptoms. Such approaches typically perform well in plants for which deep know-how and accurate models can be obtained. Yet, due to high process dependency, their use in different facilities implies tedious and expensive engineering processes.

All in all, the state of the art leaves room for improvement, particularly concerning temporal requirements, efficient use of computational resources, and transferability to new processes. This chapter is dedicated to address these and other issues by proposing a new diagnosis method for discovery of plant-wide disturbance propagation paths based on the combined use of plant topology, process knowledge, and alarm data.

The remainder of this chapter is organized as follows: Section 6.2 presents an overview of plant diagnosis methods, relevant definitions on causality and connectivity, and theoretic fundamentals of signed directed graphs. Section 6.3 discusses existing approaches to fault detection and diagnosis, and alarm management. The main drawbacks of the state of the art are discussed in Section 6.4, and based on the identified knowledge gap, a novel method for plant disturbance analysis is introduced in Section 6.5. The feasibility of the proposed method as a basis to support alarm management and root-cause determination in a chemical process is demonstrated in Section 6.6. Experimental results are analyzed in Section 6.7, and thereupon the main advantages and limitations of the method are identified. The chapter concludes with a brief summary in Section 6.8.

6.2 Theoretical framework

6.2.1 Classification of plant diagnosis methods

The essential knowledge required for fault diagnosis is the set of process failures as well as the relationships between system observations and failures. This knowledge can be expressed explicitly or inferred from some source of process information. For instance, it may be built based on fundamental understanding of the process using engineering principles, structural models of the system, or behavioral descriptions of its components. Such knowledge is referred to as deep, causal or model-based. Alternatively, it may be gathered from past experiences, process observations, and heuristics. This knowledge is referred to as shallow, compiled, evidential or process history-based. Thus, based on the nature of the used knowledge, plant diagnostic approaches can be classified as model-based or process history-based methods [VRYK03a, VRK03, VRYK03b].

6.2.1.1 Model-based methods

In a broad sense, model-based methods can be categorized as quantitative or qualitative. In quantitative models, the required process understanding is expressed in terms of mathematical relationships between the inputs and outputs of the system. In contrast, in qualitative models these relations are expressed as qualitative functions centered around different units in the process [VRYK03a].

Quantitative model-based methods are often referred to as analytical methods [CRB01]. In analytical modeling, detailed knowledge of the physical relationships and characteristics of all components in the system are represented as a set of equations based on mass, energy, momentum, and stoichiometric balances. By comparing process measurements with analytically computed values, process deviations signifying the occurrence of a fault can be identified. The main strength of this type of methods is their accurate diagnosis capability [Iyu11]. On the down side, their implementation in plant-wide fault diagnosis is costly and time consuming [NAMUR NA96, 2017]#. Examples of quantitative model-based methods include observers, parity space, and Kalman filters.

Qualitative model-based methods, also known as knowledge-based methods [CRB01], in turn, do not involve detailed mathematics to describe the physics of the system. Instead, they express the fundamental understanding of the process in terms of symbolic representations similar to those used in human reasoning. Key advantages of this class of methods include easy model understanding, reduced modeling efforts, and low computational burden. The price to pay for those benefits, however, is a reduced diagnostic resolution when compared to their quantitative counterpart. Typical examples of this type of methods are causal graphs and knowledge-based expert systems [Wat86].

6.2.1.2 Process history-based methods

Also referred to as data-driven methods [CRB01], process history-based methods extract knowledge from the analysis of process observations. In contrast to model-based approaches, where explicit knowledge about the process is needed, in this class of methods only the availability of large amount of historical process data is required [VRYK03b].

In data-driven analysis, available data is filtered and transformed into knowledge that can be used for diagnosis in a process known as feature extraction. The fundamental benefit achieved with this technique is the independence on complex first principle modeling processes at the expense of requiring high computational resources and losing on diagnosis accuracy. Examples of qualitative and quantitative process-history based approaches include expert systems, neural networks, principal component analysis (PCA), transfer entropy, cross-correlation, and pattern recognition.

6.2.2 Connectivity and causality

"Large-scale complex systems, such as modern industrial processes [...] *are interconnected by different units or elements; the system behavior is determined by the interrelationship between every pair of elements as well as the local dynamics within each element. It is essential to identify such interrelationships, namely connectivity and causality, in order to analyze influence mechanisms, structural properties, and the overall dynamic behavior"* [YDSC14].

In the process automation domain, connectivity and causality play a fundamental role in modeling and analysis, particularly within HAZOP, fault detection, and alarm management. Especially in the context of Industry 4.0, where an increasing number of interactions between plant components and

information systems is foreseen, the importance of this role is expected to persist or even acquire higher relevance [Chr15]. The reason for this is that in complex production systems, disturbances can propagate within and between plant assets due to material, energy, and information flows transferred across the process [YY15, DAIT11]. Thus, the problem of plant abnormal behavior analysis is essentially concerned with the determination of root-causes based on the analysis of connectivity and causality relations between process variables and system components.

In a general sense, connectivity relates to the existence of physical paths in a process whether they are direct or indirect, while causality delves into the analysis of flow and temporal directions between related variables. Specifically in industrial processes, connectivity refers to material, energy, and information flow connections between or within process units, sensors, actuators, and controllers. Causality, in turn, describes the cause-effect relationship between variables by analyzing causal properties such as empirical association (correlation), connectivity, and responsiveness.

Connectivity and causality, as key features of a process description, can be captured from process knowledge (e.g., plant topology models and first-principles) and from process data (e.g., by cross-correlation, transfer entropy, and Granger causality methods) [YDSC14].

The motivation for using plant topology models as a front-end for capturing connectivity and causality can be ascribed to the wide availability of plant and process design diagrams in industrial plants [DAIT11]. Examples of approaches exploiting process diagrams, specifically P&IDs, can be found in [YAB$^+$06, TBBT09, Iyu11, LKJJ14]. Although such documents embody a valuable source of causal information, relationships inferred from these sources are limited to plant components and their interconnections. As a consequence, resulting causal models cannot capture associations between the state variables describing the physics and chemistry of a process. In order to cope with such a limitation, process understanding based on first-principles [MRV03, DAIT11] and experiential knowledge (heuristics) [CRB01] can be merged with connectivity information described by process diagrams. Yet, without a systematic data integration concept, the reconciliation of these data sources may be a time-intensive and error-prone task.

Methods capturing causality information from process data are presented in [BTM05, YSX12, LKJJ14]. While the derivation of causal models from this data source avoids the time-intensive process required by first-principles modeling, a major problem related to the application of these approaches is the inference of redundant or irrelevant causalities. Consider, for instance, the case in which the actual causality of three process variables is given by the ordered sequence X→Y→Z. A data-driven method is likely to discover only the single causalities X→Y, Y→Z and X→Z. Accordingly, three causal relations are constructed in the learned causal model. Yet, note that the real causality X→Z is only established through Y. Therefore, unless there is a sound reason to justify the need for such an additional relationship, the latter causality is to be regarded as redundant and should be removed to avoid spurious results [YDSC14]. Detecting redundant or irrelevant causalities based on process data is not an easy procedure and requires in many cases the analysis of process knowledge. Added to this limitation, data-driven methods are often computationally intensive.

Aiming at gaining the benefits of both type of methods alongside coping with their main drawbacks, different approaches to causal modeling have explored combinations of methods. A general approach consists in defining, in first instance, the structure of the model based on process diagrams and first-principles, and subsequently validating resulting causal relations based on process data. This solution, however, requires a significant amount of modeling and data pre-processing effort, as well as high computational resources, specially when applied to large-scale systems.

For an extended discussion on concepts, descriptions, and methods for capturing connectivity and causality in industrial processes, see [YDSC14].

6.2.3 Signed directed graphs

A signed directed graph or signed digraph (SDG) is a qualitative model approach to fault diagnosis that incorporates causal analysis to represent process behavior. It is a map reflecting causal associations between variables by means of nodes and directed arcs. In general, nodes can represent process parameters such as physical magnitudes, sensors, system faults, component failures, or subsystem failures [CRB01], whereas directed arcs represent causal relationships between nodes. A node can take values normal "0", high "+", or low "-" representing its qualitative state. In turn, directed arcs are characterized by values "-" and "+", which represent direct or inverse relationships (resp. reinforcement and suppression) between cause and effect nodes. Figure 6.2 depicts the basic topology of a SDG.

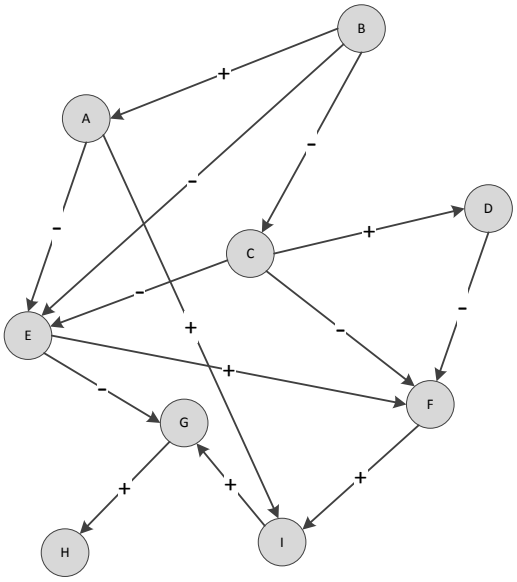


Figure 6.2: Possible topology of a signed directed graph

The fundamental premise of digraph techniques is that cause and effect linkages must connect the fault origin to the observed symptoms of the underlying fault [KP87]. Accordingly, during diagnosis measured deviations are propagated from the effect node (symptom) to the cause nodes through consistent arcs until the root-node is identified [CRB01].

6.2.3.1 SDG formal definition

As a causal graph, a SDG can be formalized as follows: let $G = (V, E, \lambda, \Delta)$ be a graph defined by two non-empty sets namely, the vertex set V (nodes) and the edge set E (directed arcs), with $E \subseteq VxV$, such that $\lambda : E \rightarrow +, -$ represent the forward influences of the edges, and Δ: $V \rightarrow +, 0, -$ the qualitative status of a vertex given by:

$$x_{vi} - \tilde{x}_{vi} > \varepsilon_{vi} \; then \; \Delta(vi) = \text{“} + \text{”} \tag{6.2.1}$$

$$|x_{vi} - \tilde{x}_{vi}| < \varepsilon_{vi} \; then \; \Delta(vi) = \text{“0”} \tag{6.2.2}$$

$$\tilde{x}_{vi} - x_{vi} > \varepsilon_{vi} \; then \; \Delta(vi) = \text{“} - \text{”} \tag{6.2.3}$$

where $\tilde{x}_{vi}$ is the set point (steady-state value), and ε is a given threshold.

6.2.3.2 Arc consistency and search strategies in SDGs

An arc of a SDG is said to be *consistent* if the product of the cause node, the directed arc, and the effect node is positive, i.e., $\Delta(v_{+n}) \cdot \lambda(v_n) \cdot \Delta(v_{-n}) = \text{“} + \text{”}$. In general, only consistent arcs can propagate disturbances; however, special arc consistency rules apply for controlled variables, where faults might propagate under the condition $\Delta(v_{n_i}) = \text{“0”}$ [ASC+14]*.

Arc consistency rules are applied when back-tracking and forward-tracking directed graphs in the search for root-causes and consequences. Common search strategies used for these purposes are the depth-first and width-first traversal algorithms [Rav08, Iyu11, PP12].

6.3 State of the art

Based on the nature of the used a priori knowledge, qualitative and quantitative approaches to fault detection and diagnosis, and alarm management can be categorized as model-based and process history-based methods (→ 6.2.1). This section presents a review on previous works[1] in this classes, which are functionally related to the proposed diagnosis method.

6.3.1 Model-based methods

6.3.1.1 Ruled-based and expert systems

Rule-based and expert systems are knowledge-based methods, which are close in style to human problem solving. In such approaches, knowledge is structured in an easy-to-store and easy-to-manipulate form, usually by means of rules or cases. Expert systems are capable of representing expert knowledge, accommodating existing databases, accumulating new knowledge, and making logical inferences and decisions with reasoning [CRB01].

Reifman and Wei [RW93, RW95] presented a diagnosis expert system based on rules derived from the observation of imbalances in conservation equations of general validity (e.g., mass, energy, and momentum). Balance inequalities allow observing certain behavioral patterns among key-process variables, which can be associated to malfunctions of specific plant components. In addition, the approach utilizes process P&IDs to enhance diagnosis resolution, specifically to generate plant asset lists and detect the presence of components which might explain observed behaviors.

Christiansen [Chr15] presented a rule-based approach to plant-wide diagnosis combining formalized process knowledge [VDI/VDE 3682, 2014]# and plant structural information. His approach

[1] Several approaches discussed in this section combine features of different method classes. Therefore, some of them may be referenced in one class or another, or in occasions may be listed in more than one class.

demonstrated that the incorporation of process-specific information can improve the resolution of diagnostic results by enlarging the amount of considered possible root-causes. In a similar direction, Abele [Abe14] proposed a decentralized monitoring system based on the combination of ontologies and rule-based representations of heterogeneous plant knowledge including component specifications, topology information, and process descriptions.

In the alarm management domain, Schleburg et al. [SCTF13] introduced an approach to alarm grouping based on the combination of alarm logs, plant topology models, and a set of rules describing interrelations between observed alarms. By using different criteria such as time intervals, process connectivity, and allowed in-between components, the method is capable of discarding or verifying pair-wise alarm causal relations, which ultimately allows grouping alarms stemming from presumable common root-causes.

6.3.1.2 Causal graphs

Graph-based approaches to abnormal behavior analysis are characterized by their suitability to describe process connectivity and qualitative behavior. However, some shortcomings such as difficulties in dealing with multiple fault diagnosis and transient responses have limited their applicability in complex industrial processes.

In an effort to overcome such drawbacks, significant research has been conducted in the field of signed directed graphs (SDGs) in combination with alternative methods. Chang and Yu [CY90], for instance, proposed the use of truth tables and modular SDGs to describe the transition states of the system, and addressed the non-single-transition problem (i.e., arcs changing sign during transients) by introducing a method based on the description of variables in velocity-forms. Mosterman and Biwas [MB99] combined bond graphs and temporal information for the derivation of temporal causal graphs. The authors differentiated between instantaneous and delayed causal effects between process variables resulting from the influence of multiplicative or integrative components in the system, namely resistances, capacitances, and inductances. Faragasan et al. [FPG04] combined a set-membership method with quantitative causal models incorporating transfer functions, and Ligęza and Kościelny [LK08] explored the integration of expert rules and diagnostic matrices to derive AND/OR causal graphs and address the detection of multiple faults.

Montmain and Gentil [MG00] incorporated quantitative information to the structure of causal digraphs by means of differential equations. The proposed causal structure allows the generation of residuals, which are used for fault isolation and posteriorly for fault identification based on fuzzy reasoning. Gao et al. [GZMW09] presented a monitoring algorithm in which SGDs are combined with real-time bidirectional inference and fuzzy logic. In the same direction, Zhang et al. [ZCWC05] developed a diagnosis approach using fuzzy-SDGs for qualitative/quantitative patterns identification and diagnosis resolution enhancement, while Arroyo et al. [ASC$^+$14]* proposed a fuzzy-based approach incorporating formalized process knowledge to refine causal relationships in SDGs.

Maurya et al. [MRV03, MRV04] and Di Geronimo Gil et al [DAIT11] explored the creation of modular causal maps by integrating plant component models. The latter proposed the creation of a model library containing basic operation units such as heat exchangers, reactors and separation columns. Such models relate model equations to process variables and are stored as biadjacency matrices. By using the connectivity information described in a digital plant topology model, unit models can be connected to generate a plant-wide biadjacency matrix, which based on a perfect

matching procedure [MRV03], is converted into a perfect matched matrix. Finally, both biadjacency and perfect matched matrices are combined to generate a variable relationship connectivity matrix representing the plant-wide structural model. This causal process representation is used in [DAIT11] to determine degrees of freedom as a front-end to support plant control design.

Chiang and Braatz [CB03] incorporated the concept of causal map and multivariate statistics for fault identification. Proposed causal maps depict directed relationships between measured signals in a system, and are built based on expert knowledge and validated with a sample covariance matrix from normal operation data. During fault identification, the dependencies depicted by the system map are broken based on the computation of causal distances upon analysis of the measurement's frequency distribution captured during the plant anomaly. As causal relations progressively break when the fault propagates across the system, the number of interconnected variables reduces its size until eventually only a single causal relation or a small group of them remains, thus revealing the root-cause(s) of the anomaly. In a similar direction, Bauer, Thornhill, and Meaburn [BTM05] combined cross-correlation methods with topology information to build causal models for chemical processes, and Lu et al. [LWC+13] presented a PCA-SDG-based method integrating statistical and causal analysis.

On the field of alarm management, Hamaguchi et al. [HTNK11] presented a method for the design of plant alarm systems based on a hierarchical SDG-based model, so-called *Cause-Effect* (CE) *model*. The CE model comprises three information layers: the lowest layer, *primitive variable layer*, is a directed graph relating process variables in accordance to material and energy balances and plant topology. The intermediate layer, *strongly-connected component layer*, is derived from the analysis of connections and loops in the primitive variable layer, and consists of a set of plant segments or modules which allow the variable selection of alarm sources. Finally, the top layer, *group unit layer*, is a set of plant segments grouping strongly-connected components based on the location of existing sensors in such a manner that up to one fault origin can be assigned in each group unit. By exploiting the information contained in the different layers of the CE model, the method allows the identification of potential fault origins and accordingly select alarm variables for fault identification. An extension of Hamaguchi's work was presented by Takeda et al. [THKN13], who delved into the concept of modules and their relation to fault origins as a means to explicitly explain the design rationale of alarm systems.

6.3.2 Process history-based methods

Most process history-based approaches tackle the fault diagnosis problem by following a two-step approach. The first step, fault detection, is based on the assumption that measurements whose spectra are similar –according to a given metric– are subject to the same plant disturbance. A plant-wide spectral analysis aims, therefore, at detecting measurements having similar features and clustering them accordingly. In the second step, fault identification, correlation methods are applied on identified signal groups to infer associations and determine possible fault root-causes [TCP03, BTM05].

Previous research has shown that the use of plant topology information can improve the resolution and reliability of process history-based methods. Yim et al. [YAB+06] and Thambirajah et al. [TBBT09], for instance, used CAEX plant models (→ 3.2.1.2) to capture plant connectivity and flow directionality. This information was combined with the results of data-driven analysis methods in order to establish physical paths among measurements exhibiting similar spectra. An extension

of these approaches was presented by Iyun [Iyu11], who combined connectivity and directionality information extracted from smart P&IDs [ISO 15926-1, 2004]# and basic process knowledge to validate data-driven causality results.

Beyond the common use of connectivity and directionality information, a further commonalty among the aforementioned works is the use of the PDA® signal analysis tool as main engine for data-driven analysis. Among other methods, the tool deploys principal component analysis (PCA) for signal clustering (due to Thornhill et al. [TSHV02]), nonlinearity analysis for the diagnosis of oscillating disturbances (due to Thornhill, Cox and Paulonis [TCP03]; and Bauer et al. [TCP03, BTM05]), and transfer entropy and time delay analysis for the diagnosis of non-periodic faults (due to Bauer et al. [BCC+07], and Bauer and Thornhill [BT08]).

In a similar direction, Landman, Kortela and Jämsä-Jounela [LKJJ14, LJJ16] proposed a causal search algorithm combining transfer entropy and topology information captured from AutoCAD and ISO 15926 P&IDs. Other approaches on this area include the contributions of Yang, Shang, and Xiao [YSX12], who applied both cross-correlation and transfer entropy techniques to validate causal relations in directed graphs, and Duan, Yang, Chen and Shah[DYCS13], who proposed the use of direct transfer entropy to detect and discriminate between direct and indirect causality relationships between process variables in both linear and nonlinear multivariate systems.

In the field of alarm management, Yu and Yang [YY15] presented an approach to causality detection based on the analysis of alarm data. The authors proposed the application of discrete transfer entropy in binary alarm series to derive causal relations among process variables during plant abnormal operation. Likewise, Rodrigo et al. [RCHH16] used a combination of alarm logs, process data, and connectivity analysis to isolate consequence alarms and to provide causal alarm suggestions. Their approach extends the alarm grouping method originally proposed by Schleburg et al. [SCTF13], by incorporating causal analysis based on transfer entropy to establish cause-effect relationships among alarms groups. With a similar aim, Folmer and Vogel-Heuser [JFVH12] and Folmer et al. [JFVH14] used sequential pattern matching to cluster frequent occurring subsequences in alarm logs and identify alarms with causal relations.

6.4 Drawbacks of the state of the art

Despite the significant advances made by previous contributions in the field of plant abnormal behavior analysis, the state of the art has several drawbacks particularly in aspects concerning performance and applicability. This section lists the main shortcomings identified within the conducted literature review.

1. Approaches to derivation of causal models based on mathematical representations [MRV03, MRV04, DAIT11] require complex optimization methods and additional non-causal redundant equations for the generation of consistent structural models.

2. The application of classic graph-based methods in large-scale systems involves a time-intensive collection of highly detailed information of the plant [CB03] alongside significant efforts for model maintenance [YAB+06].

3. Although a number of existing causal approaches have exploited plant topology information, namely connectictivity and directionality, most of them do not incorporate component-specific knowledge, and are limited to static and not always updated process causalities.

4. Only few causal approaches have separated the source of knowledge (e.g., causal relations based on first physical principles or heuristics) from the model instance. The lack of a source of knowledge independent of the actual process model makes model maintenance a costly and time-intensive procedure as knowledge modifications and extensions require the manual amendment of large and complex models (e.g., plant-wide causal graphs).

5. Quantitative extensions of causal graphs such as those proposed in [LGFB94],[MG00], [FPG04] and [ABF+07] require numerous and relatively complex mathematical models for the description of causalities, which hinders their implementation in large-scale processes.

6. Fault diagnosis based on purely analytical models is a challenge in process industry because accurate and calibrated model are expensive to create and maintain [MG00, YAB+06]. The implementation of analytical methods demands a complex and error-prone development phase, especially due to extensive efforts required during the derivation of models and their respective validation for each plant.

7. Most knowledge-based expert systems are designed to diagnose faults in specific processes (e.g., [Chr15]). This shortcoming implies error-prone and time-consuming adaptation steps for the utilization of these systems in other processes.

8. Process history-based methods require the availability of large amounts of process historical data [VRYK03b]. In many cases, however, available data might not be enough to extract significant features and derive meaningful conclusions on possible root-causes [YDSC14].

9. Pure data-driven methods are sensibly affected by the influence of control loops in the process as signal spectra are notably altered while propagating across information connections. The existence of such structures might result in high missed detection rates (MDR) and spurious root-cause hypotheses [TBBT09]. This shortcoming supposes an obstacle for implementation in modern and heavily automated facilities with up to thousands of control loops.

10. Several process history-driven methods (e.g., nonlinear time-series analysis) require high computational effort and their execution might be time-intensive [LJJ16], particularly when applied to a large number of process signals. Other data-drive approaches such as those based on transfer entropy require a time-consuming and error-prone estimation of probability density functions, models parameters, and statistical thresholds [LJJ16].

11. A number of approaches have used data analysis techniques for the derivation of process causal models. Such techniques, however, are significantly affected by the inference of redundant or irrelevant process causalities, which might lead to inaccurate root-cause results [YDSC14]. Other approaches, in turn, have derived causal models from process diagrams such as PFDs and P&IDs. Yet, diagram-based models cannot capture relationships between the state variables that describe the physics and chemistry of the process leading to poor diagnostic resolution.

These and other limitations of the state of the art reveal the need of complementary approaches aimed at enhancing the reliability and applicability of automated plant diagnostics alongside reducing the required effort for their deployment. In an effort to contribute towards the solution of the current shortcomings, this chapter presents a new hybrid method for the monitoring of plant

disturbance propagations based on the dynamic derivation of simple plant causal models, and the systematic evaluation of plant and process information.

The proposed approach is intended for the analysis of disturbance propagation paths, the fast discovery of alarm groups, and ultimately the isolation of faulty plant zones. Such information is aimed at two key tasks: (*a*) supporting process experts during the handling of process abnormal situations by grouping alarms and suggesting potential causal alarms, as well as by providing insight into the behavior of disturbance propagations along the plant, and (*b*) determining specific plant sections and signals for which other dedicated analysis algorithms should be applied for detailed fault diagnosis. Note that this contribution is not intended to present rigorous causal models, but to introduce a new concept for the dynamic generation of modular causal graphs aimed at the discovery of disturbance propagation paths. Such models can be applied for abnormal behavior analysis in different process systems and its accuracy may be adjusted based on available information.

6.5 Proposed method for disturbance propagation analysis

6.5.1 General diagnosis concept

The proposed method for plant abnormal behavior analysis is based on a graph-scheme which compartmentalizes plant topology and process knowledge into modular components associated with plant equipment. From a functional perspective [BDN16], this method falls into the class *root-cause diagnosis approaches*, whereas based on the used a priori knowledge, it can be classified as a *hybrid approach* [Iyu11], since it incorporates characteristics of both *quantitative* and *qualitative model-based methods* [VRYK03a, VRK03, VRYK03b].

The following three key-notions sustain the underlying diagnosis concept:

1. Disturbances are transported along the plant by a defined set of material and information propagation carriers.
2. Certain equipment and process characteristics –static and dynamic propagation-related factors– influence the behavior of disturbances propagating across the plant.
3. The occurrence of multiple process alarms may be an indicator of disturbance propagation. Analyzing the causality between alarms can, therefore, allow supporting the determination of potential root-causes.

Based on such notions, the method performs a systematic evaluation of process causalities by combining topology and process information for the discovery of disturbance propagation paths linking observed alarms. Such a procedure comprises two main parallel tasks: (*a*) the event-triggered generation of *dynamic causal digraphs* (DCDGs), i.e., modular causal models interconnected based on plant connectivity information, and (*b*) the recursive consultation of a *propagation look-up table* (PLUT) containing general and process-specific information for establishing updated causalities within and among plant components.

In the sequel, the main conceptual artifacts of the proposed approach, i.e., the dynamic causal digraph and the propagation look-up table, are presented.

6.5.2 Conceptual artifacts

6.5.2.1 Dynamic causal digraph

A DCDG is a modular causal model depicting relationships between process variables by capturing the first-principle structure of the underlying system without providing detailed equations for the modeled causal relations. DCDGs are essentially based on the concept of signed directed graphs (SDGs); however, unlike typical SDGs, dynamic causal digraphs extend and modify certain aspects concerning causal descriptions, graph structure, and model generation. The following subsections discuss in detail the main differences between SDGs and DCDGs.

a. Causality and temporal description capabilities

Although SDGs are a suitable approach to causal modeling, their capabilities for detailed process descriptions may result insufficient in some cases. Consider for instance the SDG segment illustrated in Figure 6.3, which depicts some of the causal relations occurring in a typical reactor. As it is known from first physical principles, the cooling water (CW) flow causes a suppression effect (-) on the process temperature, whereas the temperatures of the reactant feeds produce a reinforcement influence (+) on the same variable. In the event of a disturbance, the behavior of the process temperature will largely depend on whether the cooling water flow or the reactants temperatures have the stronger influence, as well as on the time constants of the underlying causalities. Such relations, however, cannot be represented through SDGs, as these neither quantify the extent to which a node affects its successor nor describe temporal causalities.

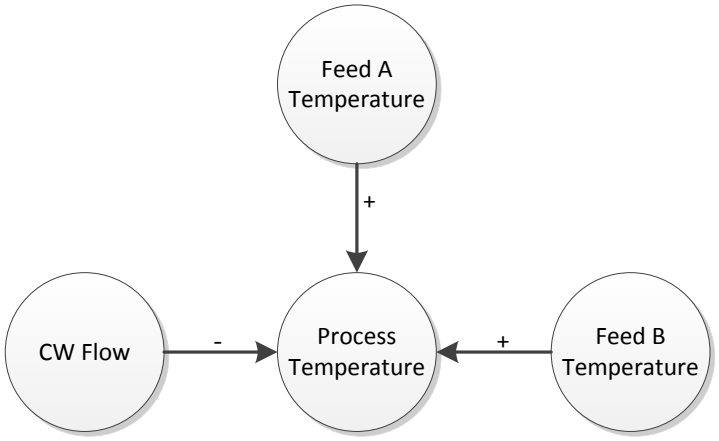


Figure 6.3: SDG segment of a chemical reaction process

Another limitation of SDGs is their incapability to model qualitative relationships between process variables that cannot be simply described as “suppression” or “reinforcement” effects. Take as an example the case in which the magnitude “feed composition” is modeled as a node in a SDG. One may naturally expect that the composition has a direct influence, among other variables, on the process temperature, implying thereby the existence of a direct arc in the causal graph. Nevertheless, due to the vagueness of the term “feed composition” –understanding this as the concentrations of different chemical components in the feed– it is not possible to clearly determine wether the relation is of type “-” or “+”. While the increase of the concentration of a certain component may imply an increase of process temperature and thus a “+” relation, the same increase involves the reduction of the concentration of another component, which in turn can be seen as the cause of the observed temperature increase and therefore suggests a “-” relation. Although one could argue that this

problem can be solved by modeling the concentration of every feed component as a separate node in the SDG, this may result in an undesired explosion of the graph size. For several purposes, it might be of interest to keep certain causal relations at a "vague" or "ambiguous" level. It follows that a new causal state to characterize such process relations is required.

In view of the aforementioned shortcomings, the dynamic causal digraph extends the description of directed arcs by incorporating the quantitative magnitudes *influence strength* or *intensity* (Φ) and *expected time delay or lag* (τ), as well as by adding a new qualitative causal state (*0*) to characterize ambiguous causal relationships between nodes. As discussed in more detail later on in this chapter, influence strengths allow providing a simple quantitative approximation of the influence of a node on its successor, while *expected time delays* enable the description of temporal relations between nodes. The addition of these new descriptive features is formalized in the following definition:

Let $G = (V, E, \Phi, \tau, \lambda, \Delta)$ be a graph defined by two non-empty sets namely, the vertex set V (nodes) and the edge set E (directed arcs), with $E \subseteq VxV$, such that $\Phi : E \rightarrow a \in [0, 1]$, $\tau : E \rightarrow t \in \mathbb{R}$, and $\lambda : E \rightarrow \{+, 0, -\}$ represent the strength, expected time delay (lag), and forward influence of an edge, and Δ: $V \rightarrow \{+, 0, -\}$ describes the qualitative status of a vertex as given by:

$$x_{vi} - \tilde{x}_{vi} > \varepsilon_{vi} \; then \; \Delta(vi) = \text{“} + \text{”} \tag{6.5.1}$$

$$|x_{vi} - \tilde{x}_{vi}| < \varepsilon_{vi} \; then \; \Delta(vi) = \text{“0”} \tag{6.5.2}$$

$$\tilde{x}_{vi} - x_{vi} > \varepsilon_{vi} \; then \; \Delta(vi) = \text{“}-\text{”} \tag{6.5.3}$$

where $\tilde{x}_{vi}$ is the set point (steady-state value), and ε is a given threshold.

Based on the above definition, the arc consistency criteria of the DCDG can be formulated as follows: An arc is said to be *consistent* if the product of the cause node, the directed arc, and the effect node is positive, i.e., $\Delta(v_{+n}) \cdot \lambda(v_n) \cdot \Delta(v_{-n}) = \text{“} + \text{”}$, whereas an arc is regarded as *ambiguous* if its value is zero, i.e., $\lambda(v_n) = \text{“0”}$. In general, disturbances may propagate both through consistent and ambiguous arcs, although penalization clauses may be applied to the latter due to the associated propagation uncertainty (→ 6.5.3.3).

The proposed model extensions can significantly increase the expression capabilities of the causal graph while keeping its structure still simple and independent of complex mathematical models. This is one of the main advantages of dynamic causal digraphs compared to previous approaches incorporating e.g., transfer functions [LGFB94], differential equations [MG00], and further mathematical models [FPG04, ABF+07] into bond graphs, fault trees, and other causal graphs.

b. Defined set of model variables

Unlike SDGs, which only specify the structure of the causal graph but not its variables, dynamic causal digraphs define a specific set of model variables, so-called propagation carriers, to describe the process causalities occurring within and among components in a process. As depicted in Figure 6.4, considered propagation carriers are divided into two main classes, i.e., *Material* (subdivided in *Pressure*, *Flow*, *Level*, *Composition,* and *Temperature*), and *Information* (subdivided in *SensorSignal*, *ControllerSignal*, and *ActuatorSignal*).

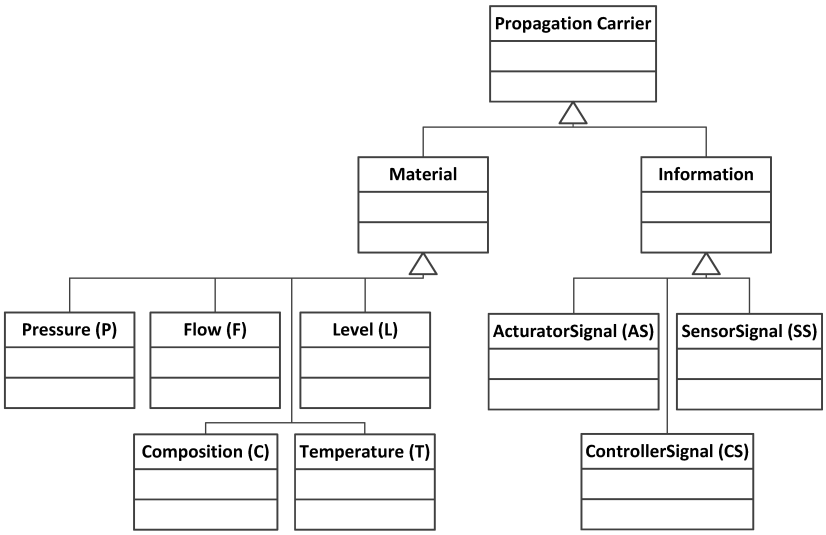


Figure 6.4: Considered propagation carriers in process systems

The consideration of a specific set of variables pursues uniforming the modeling of process systems at a level of detail suited for the analysis of disturbance propagation. While taking into account a larger number of process variables would increase the capabilities of the DCDG for depicting more detailed causalities, this would result in larger models –harder to maintain and interpret– as well as in the need of more computational resources and longer execution times. Accordingly, the defined set of propagation carriers has been chosen to be specific enough to enable the causal modeling of most chemical processes in industrial applications, and compact enough to allow for fast discovery of propagation paths.

c. Graph-tabular structure

The consolidation of process relevant data through sensible visualization can significantly facilitate process diagnostics [NAMUR NA96, 2017]#. With the aim of allowing for a structured visualization of process causalities and potential propagation paths, dynamic causal digraphs in contrast to typical SDGs, are organized in rows and columns (see Figure 6.5). In this structure, rows represent plant assets organized by topological connectivity, whereas columns symbolize considered propagation carriers, respectively: *Composition* (C), *Temperature* (T), *Pressure* (P), *Flow* (F), *Level* (L), *SensorSignal* (SS), *ActuatorSignal* (AS), and *ControllerSignal* (CS).

A DCDG node embodies the state of a given variable (column) within a certain plant asset (row), indicating whether the variable is in normal or upset condition (e.g., too low or too high). DCDG edges, in turn, describe the qualitative and quantitative causal influence of a node on its successor as well as the expected lag of the underlying cause-effect relation. As discussed later on in Section 6.5.2.2, the estimation of causal and temporal values considers process information, such as component and process properties, and experiential knowledge or heuristics.

Columns corresponding to propagation carriers of type *material* (T, P, F, Q, L) are populated with nodes embodying plant assets in contact with the material flow (e.g., pumps, pipes, and tanks), whereas columns representing propagation carriers of type *information* (SS, AS, CS) are populated with nodes accounting for devices within control loops (i.e., sensors, actuators, and controllers). Note here that only those components aimed at storing liquid volumes, such as vessels, contain nodes in the *Level* (L) column.

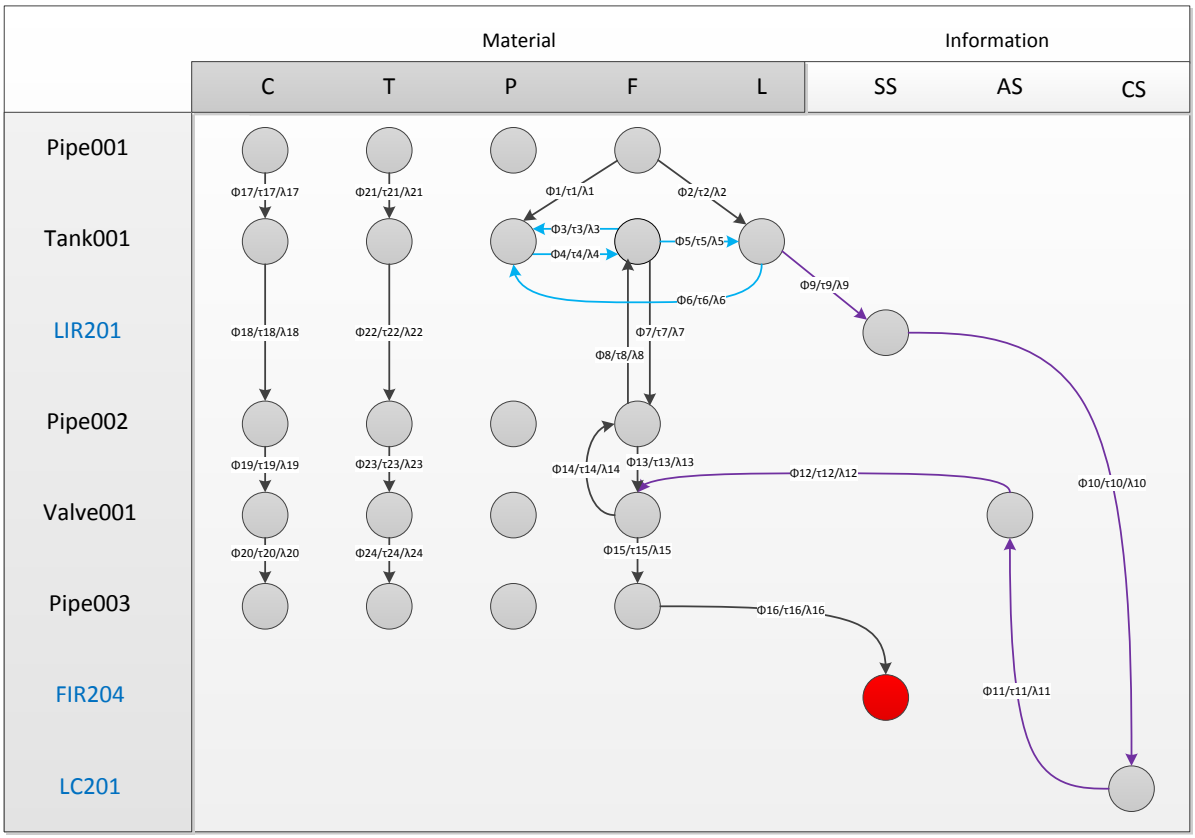


Figure 6.5: Example of dynamic causal digraph. DCDG corresponding to the single-phase open tank system with level control depicted in Figure 6.6

For the connectivity of the nodes within rows and columns, three types of links are distinguished, namely: *Directed Link* (DL), *Component Inherent Link* (CIL), and *Information Link* (IL). As depicted in Figure 6.5, a DL (black arrow) relates nodes in different rows and represents the existent physical connection among plant assets. A CIL (light blue arrow), in turn, connects nodes within a given row and symbolizes existing bonds between variables in a determined piece of equipment due to a physical principle; for instance, the relation temperature-pressure in a two-phase closed tank (due to the ideal gas law). Finally, an IL (purple arrow) symbolizes the communication path between elements in a control loop, and can connect different rows and columns. Unlike the previously defined DCDG causal properties –i.e., strengths, lags, and qualitative causalities– link types are not modeled as edge attributes, as they are implicitly defined by the coordinates (rows and columns) of the related nodes in the graph.

The consideration of link types allows capturing the semantics of the different connections occurring in the process, which enables filtering searches and prioritizing diagnosis results based on detailed data queries (→ 6.5.3.3). During the generation of DCDG models, required connectivity links are created by consulting the connectivity information contained in the plant topology model and a propagation look-up table (→6.5.2.2).

d. DCDG model generation

In contrast to SDGs, which typically are static models built once for a modeled process, dynamic causal digraphs are generated in an event-triggered basis, i.e., an individual DCDG is dynamically constructed when a process disturbance (signalized by alarms) is detected or alternatively upon request of the user. The particular DCDG for that event considers up-to-date information of

the process and its assets, thus allowing an accurate description of the plant abnormal situation. Topology information (e.g., connectivity, flow directionality, and static characteristics of plant assets) required during the generation of the DCDG is extracted from a plant topology model as that presented in Chapter 4, whereas dynamic process data can be collected directly from a control system (e.g., DCS or SCADA), an intermediate data server, or a simulation environment.

The linkage between topology information and causal process knowledge (i.e., first principles, heuristics, and static and dynamic process data) is accomplished by storing the causal relations of plant components such as heat exchangers, pumps, and reactors as templates in a knowledge repository or library (see propagation look-up table → 6.5.2.2). When one of these components is identified during model generation, the corresponding causal relations of the asset are consulted in the library and instantiated in the DCDG. Should causal relations contain expressions or functions in terms of plant or process data (e.g., types, dimensions, or states), the algorithm retrieves this information from the topology model by exploiting fit-to-purpose data access methods, specifically AFDD PDS, PKS, and PS model facets (→ 4.4.3.1). Once all causal relations have been evaluated and instantiated, the resulting component causal models (modules) are interconnected based on the connectivity information described in the topology model. In order to illustrate the dynamic generation of the DCDG, consider the following brief example.

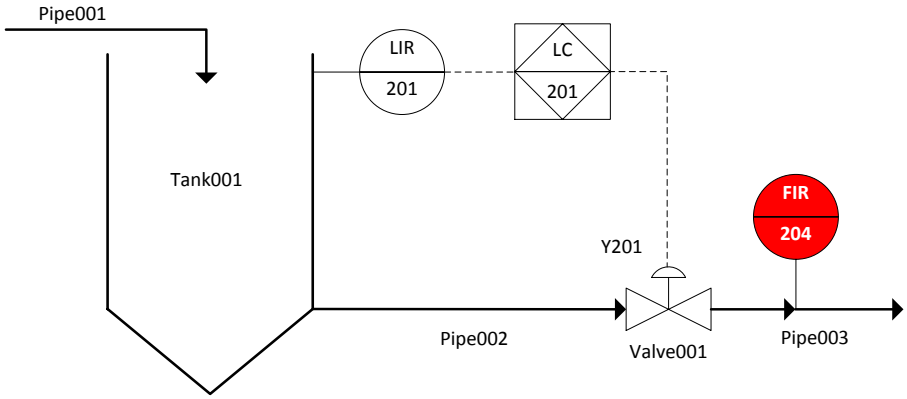


Figure 6.6: Plant section. Flow anomaly detected by sensor FIR204. Single-phase liquid open tank with level control

In the plant section depicted in Figure 6.6, a flow disturbance, concretely an unusual flow increase, is detected in Pipe002 by the flow sensor F204 (highlighted in red color). The corresponding flow alarm with status *hihi* is activated and shown to the operator. An automatic diagnosis is started and accordingly a DCDG for the specific event is generated by instantiating the causal models of every component in the process and linking them according to the system connectivity. The corresponding Dynamic Causal Digraph is depicted in Figure 6.5. From it, two possible *consistent propagation paths* are discovered, namely:

1. FIR204(SS)←Pipe003(F)←Valve001(F)←Pipe002(F)←Tank001(P)←Pipe001
2. FIR204(SS)←Pipe003(F)←Valve001(F)←LC201(CS)←LIR201(SS)←Tank001(L)←Pipe001

The first found propagation path indicates the disturbance could have been produced due to an increase in the outflow of Pipe001 causing a rise of Tank00 pressure and ultimately an increase of the outflow propagating along the subsequent plant components (Pipe002, Valve001, and Pipe003) through the flow carrier. The second propagation path also presumes the disturbance was originated

due to an alteration in the outflow of Pipe001, which produced an increase of the Tank001 level and consequently a raise of the outflow propagating to Pipe003 caused by the control loop action (LIR201, LC201, V001). Despite suggesting two different propagation paths, one containing only material flow carriers and the other both material and information flow carriers, both discovered propagation hypotheses pinpoint an alteration in the flow of Pipe001 as the potential root-cause of the anomaly.

For the ranking or prioritization of discovered propagation paths, the *overall strengths* of every path are computed. This procedure is systematically executed by multiplying the strength values of every edge (Φ_i) in a given path. Resulting overall strengths are sorted in order of magnitude, which allows prioritizing and filtering paths based on defined strength thresholds.

Given the existence of multiple alarms caused by plant anomalies, the same disturbance propagation concept can be applied to group related alarms and classify observed alarms as *causal* and *consequential*. In general, attending causal alarms requires specific operator counteractions, whereas the clearance of consequential alarms does not. In virtue of the underlying causal relation, consequential alarms will be automatically suppressed after the root of their causal alarm has been found and attended. Therefore, the proposed concept may significantly facilitate the required actions of plant operators when facing plant abnormalities.

6.5.2.2 Propagation look-up table

The propagation look-up table (PLUT) is a reusable and cumulative data repository or library containing component-specific connectivity links and expressions for the estimation of their respective causal and temporal properties. Aligned with the modular vision of causal models described by Di Geronimo et al. [DAIT11] and Yang, Shah, and Xiao [YSX12], the PLUT embodies a collection of templates or modules representing the causal behavior of plant assets and process units, which are instantiated and connected during the generation of dynamic causal digraphs based on connectivity and process data.

Table 6.1: General structure of a propagation look-up table (PLUT)

Component	***Coordinates***	***Causal properties* ($\Phi/\tau/\lambda$)**	***Rationale***
Component_1	X_a, Y_b	$IF: Condition_1$ ($\Phi_1/\tau_1/\lambda_1$)	Explanation_1
...	...	...	...
Component_n	X_a, Y_b	$IF: Condition_n$ ($\Phi_n/\tau_n/\lambda_n$)	Explanation_n

As depicted in Table 6.1, the PLUT comprises four columns, namely *Component*, *Coordinates*, *Causal properties* ($\Phi/\tau/\lambda$), and *Rationale.*

- The column *Component* contains the list of plant assets or process units for which connectivity templates are available (*Component_1*,...,*Component_n*). The granularity of stored templates is variable and can go from single components, e.g., pumps, tanks, and valves, to complex equipment combining multiple components such as washing and separation units.
- The column *Coordinates* specifies the external and internal causal relations or connections of every stored component. Coordinates have the form (X_a, Y_b), where X and Y refer to the

source and destination carriers (i,e., T, P, F, C, L, SS, AS, CS) linked by a given connection, and subindices a and b make reference to the inlet and outlet ports of the component associated by the causal relation. By way of illustration, the ordered pair (F_{i-1}, P_i) describe the connection between the inlet flow of a component and its outlet pressure. Note that in this structure, the outlet of a component corresponds to the inlet of the succeeding component.

- The column *Causal properties* ($\Phi/\tau/\lambda$) describes the quantitative and qualitative causal and temporal characteristics of a connection (strength, lag, and causal effect) by providing values and expressions such as constants and simple functions for their estimation. Expressions might contain IF-statements constraining the existence and properties of causal relations to specific component and process characteristics, such as the device type or operation state (e.g., "on", "off", "open", and "close"). The definition of $\Phi/\tau/\lambda$ values is not constrained to a specific methodology. In fact, it can be carried out by following different approaches such as abstraction of expert know-how, heuristics, or process data analysis [YDSC14]. Values may be estimated, for instance, based on plant component characteristics, empirical observations, or statistical analysis of historical process data [BTM05].

- The column *Rationale* presents a description of the first-physical-principle, heuristic rule, or notion justifying the existence of a given connection (edge) and optionally a comment regarding the values or expressions assigned to the causal and temporal characteristics of a given edge. Its fundamental aim is to facilitate knowledge management within the PLUT.

Table 6.2 illustrates a fragment of a propagation look-up table describing one of the causal relations a volume object may have under specific conditions. Here, the graph coordinates, the respective causal and temporal expressions, and the rationale for the modeled causal relation can be retrieved by look-up. When a volume object (e.g., a reactor) is identified during model generation, the algorithm systematically consults the causal relations of this type of device in the PLUT, verifies that the current causal relation applies to the particular object by evaluating the specified IF statements, and instantiates it in the DCDG with the respective causal and temporal values if the verification result is positive.

Table 6.2: Fragment of a propagation look-up table

Component	***Coordinates***	***Causal properties*** ($\Phi/\tau/\lambda$)	***Rationale***
Volume	(F_{i-1}, L_i)	IF: Two-phase, open volume, liquid flow inlet $(.99/.1 * d/+)$	Volume level is dictated by accumulation of liquid. A flow disturbance has a direct and significant effect in the level carrier. This effect is not instant. The underlying lag is determined, among other factors, by the dimensions of the volume object.
	...	...	...

d: Volume diameter in meters. Time lag τ in hours.

From the knowledge management perspective, the PLUT can be regarded as an extensible data repository based on the concept of aggregation, i.e., new and refined knowledge can be added in accordance to the current information requirements or data availability. Such a knowledge base can be defined for a set of standard plant assets, i.e., a vendor-independent catalogue of plant devices commonly found in process facilities (e.g., generic valve, generic tank, and generic centrifugal pump). This property allows the methodology to be system-independent, and guarantees its effective and intuitive applicability in projects of different nature. Notice, however, that the PLUT also allows the addition of vendor-specific catalogues or plant-specific assets. This fact, nevertheless, does not go against the philosophy of non-system dependency but rather constitutes a resource to refine the precision and accuracy of the evaluation functions, if required.

The knowledge modularity offered by the propagation look-up table alongside the dynamic nature of the DCDG allows overcoming one of the main drawbacks of SDG-based fault diagnosis, i.e., the high effort required for information collection and model maintenance [YAB+06, CB03]. As modules can be easily extended, updated, or exchanged in complete independence of the causal graph itself, the tedious inspection process required for the update of large models can be avoided. In turn, updated knowledge can be automatically and systematically transferred from the PLUT to those instances in the causal graph affected by the underlying changes.

6.5.3 Integrated diagnosis concept

Previous research has shown that the use of causality information between process alarms allows identifying propagation paths and thus supporting operators during root-cause analysis [HB07, YY15]. Aligned with this notion, the proposed diagnosis method uses alarms as key disturbance indicators for the discovery of propagation paths based on the concepts of dynamic causal digraphs and propagation look-up tables. However, as an initial requisite for involving alarms in causal disturbance analysis, specifically in propagation paths searching and filtering, a direct association or mapping between alarms, measurements, and plant components is needed. The following subsections discuss how this mapping can be automatically generated based on available information sources, and explain how integrated alarm data, plant information, and process knowledge are exploited within the proposed path searching and filtering strategies.

6.5.3.1 Mapping process alarms to plant assets

In order to relate alarm-based root-cause hypotheses to process signals and ultimately to plant assets, a mapping between related instances is required. In this contribution, this mapping is created by analyzing the connectivity between sensors, alarm functions, and components in the topology model. By way of illustration, consider the schematic of Figure 6.7a in which a process vessel (S001), a pressure sensor (PIR116), and two alarm functions (PAL116 and PAH116) are depicted.

The topology model corresponding to this schematic contains the connections S001→PIR116, PIR116→PAL116, and PIR116→PAH116. The analysis of these links allows assigning the described alarm functionalities to the pressure measurement as well as relating these alarms to the process vessel, as shown in the left-hand side of Figure 6.7b. Such a representation can be further transformed into a structure matching the status convention used in alarm logs. Supposing that in the used convention, the statuses of low and high level alarms are indicated as *LO_ALM*, *LO_LO_ALM* and *HI_ALM*, *HI_HI_ALM*, Algorithm 6.1 can be used to generate the required associations. Figure 6.7b illustrates the mapping generated for alarms with one and two status levels.

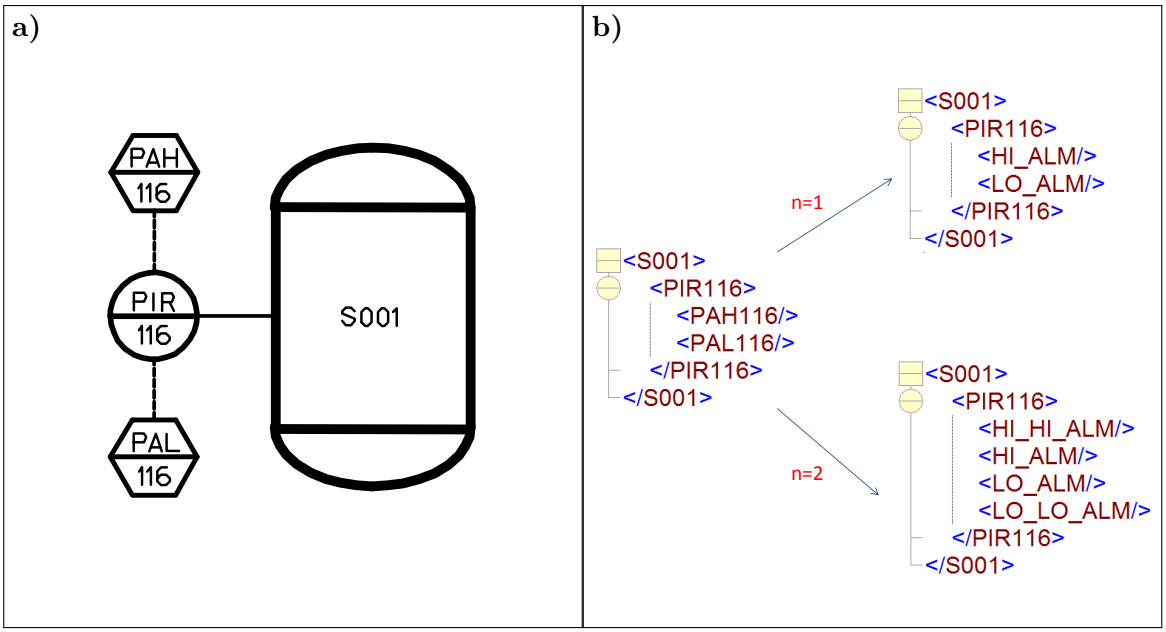


Figure 6.7: a) Process vessel with pressure measurement and associated alarm functionalities b) Possible mappings for one (n=1) and two (n=2) alarm status levels

In many occasions, however, no explicit graphic symbols are used in the process schematic to depict alarm functions. Instead, these are described by additional characters in the tags of the measurements to which they are associated. The tag *"LICA+-"*, for instance, indicates a level controller with value indication (*LIC*) and high/low alarm functions (*A+-*). In these cases, the mapping is derived from the syntactical analysis of the tags –based on regular expressions– alongside the consideration of the connectivity between plant assets and instruments. Once the alarm functionalities have been identified, Algorithm 6.1 can be applied to generate the required mappings.

As for other common alarms and warnings not specified in process schematics (e.g., valve and pump malfunctions), default alarm functionalities might be associated to actuation devices in the topology model with the respective failure statuses used in the alarm convention. By way of illustration, all controlled valves in the process might be assigned a standard alarm functionality "*failure*".

Algorithm 6.1 Mapping process alarms to plant assets

```
For all alarm functions:

    If (alarm function is of type "L")
        Create 1...n low alarm functions such that:
        n=1: LO_ALM, n=2: LO_LO_ALM...

    If (alarm function is "H")
        Create 1...n high alarm functions such that:
        n=1: HI_ALM, n=2: HI_HI_ALM...

Where n ∈ ℕ is a user-defined parameter indicating the number of low and
high alarm status levels in the log.
```

6.5.3.2 Path search algorithm

Based on the alarm mapping information, alarms can be associated to specific elements in the DCDG; hence, traversing the graph allows the discovery of propagation paths linking observed alarms during a plant disturbance. The path finding algorithm can be configured to search for paths beginning at an element, ending at an element, or between two elements in the graph. This enables operators and process experts to query information on three types of causal results, namely: possible consequential alarms derived from a potential causal alarm, potential causal alarms for a chosen consequential alarm, and possible propagation paths between two given alarms.

Provided an initial element, propagation paths are computed by iteratively adding new elements until a stop criterion is met. For a new element to be added in the current path, the following criteria must be fulfilled:

1. *Maximum path length*: The length of a path cannot exceed a maximum number of allowed elements. Empirical observations suggest that long propagation paths tend to be spurious and are less likely to reveal the actual source of an anomaly. Limiting the maximum path length to a moderate value allows increasing diagnosis accuracy and reducing computation time.

2. *Minimum overall strength threshold*: Because each edge in a path has a baseline strength, it is possible to keep track of the overall strength of a growing path after each new edge is added. If a given path is deemed implausible early on during computation, it is neither pragmatic nor computationally efficient to keep adding further elements, since the path only becomes less plausible as new elements are added.

3. *Number of times a given node can be used in a path*: Due to recycles and control loops in the process, a given DCDG element might appear more than once in a propagation path. Limiting this value to a low threshold allows preventing undesired path loops.

4. *Number of times a particular edge can be used in a path*: Similar to the previous criterion, limiting this value to a given maximum prevents the generation of spurious paths with unwanted inner loops.

The generation of propagation paths between two selected elements provides the basis for alarm causal analysis and alarm grouping. This process is conducted by testing each pair of alarms bidirectionally, i.e., each tested as potential causal and consequential alarm. However, in order to prevent unnecessary computations, it is recommendable to pre-process analyzed alarm logs by removing chattering alarms [ISO 18.2, 2009]#. As discussed by Kabir et al. [KIC+13] and Rodrigo et al. [RCHH16], such alarms can be simply filtered by setting a threshold for the minimum allowed time between two occurrences of the same alarm, i.e., two alarm instances with the same alarm tag and status (see alarm log in Figure 6.8). Thus, if the elapsed time between two alarm occurrences is shorter than the given threshold, the second alarm is to be removed.

Time In	Alarm Tag	Area	Description	Status	Priority
08-Nov-16 22:30:15:10	L7061	CA38	Level Wash Column	HI_ALM	Critical
08-Nov-16 22:31:13:32	P7063	CA38	Pressure Wash Column	HI_HI_ALM	Critical
...	...	...	...	...	...
08-Nov-16 22:39:16:40	L7061	CA38	Level Wash Column	HI_ALM	Critical

Figure 6.8: Fragment of a common alarm log

a)

	A1	A2	A3
A1	0	1	1
A2	0	0	1
A3	0	0	0

b)

	A1	A2	A3
A1	0	1	0
A2	0	0	0
A3	0	0	0

c)

	A1	A2	A3
A1	0	0	0
A2	0	0	0
A3	0	0	0

d)

	A1	A2	A3
A1	0	1	0
A2	0	0	1
A3	1	0	0

Figure 6.9: Example ACMs for an alarm log with three entries. Colors illustrate the alarms belonging to the same alarm group

Once the analyzed log has been initially filtered, a search algorithm computes potential paths between every pair of alarms by considering graph connectivity and predefined stop criteria, as well as by applying penalization and filtering strategies (→ 6.5.3.3). Search results are stored in form of an alarm connectivity matrix (ACM) describing the directional causality among inspected alarms. If the algorithm finds a feasible propagation path connecting one alarm to another, this result is stored as a "1" in a specific row and column of the ACM, meaning the alarm associated with that row was found to cause the alarm associated with the respective column.

The analysis of alarm connectivity matrices allows identifying alarm groups as well as classifying observed alarms as causal and consequential. Sequences of interconnected alarms in the matrix reveal alarm groups, in which any alarm without a found cause is regarded as causal, and, by the same token, any alarm caused by others is regarded as consequential. Figure 6.9 presents four abstract examples of ACMs. Here, the following alarm groups and alarm types are inferred:

- **ACM a)** → 1 alarm group $AG_1 = \{A1, A2, A3\}$; causal alarms $\{A1\}$; consequential alarms $\{A2, A3\}$.
- **ACM b)** → 2 alarm groups $AG_1 = \{A1, A2\}$, $AG_2 = \{A3\}$; causal alarms $\{A1\}$, $\{A3\}$; consequential alarms $\{A2\}$.
- **ACM c)** → 3 alarm groups $AG_1 = \{A1\}$, $AG_2 = \{A2\}$, $AG_3 = \{A3\}$; causal alarms $\{A1\}$, $\{A2\}$, $\{A3\}$; consequential alarms $\{\emptyset\}$.
- **ACM d)** → 1 alarm group $AG_1 = \{A1, A2, A3\}$; causal alarms $\{\emptyset\}$; consequential alarms $\{A1\}, \{A2\}, \{A3\}$.

The causal results derived from the above ACM examples account for a large amount of common causalities found in typical plant alarm logs. The first example, for instance, illustrates the best-case scenario in which all current alarms belong to the same alarm group and can be explained by a single causal alarm. The second example, in turn, shows different alarm groups with different causal alarms. The case in which all current alarms are unrelated and belong to different groups is illustrated in the third example. Finally, the fourth example presents the scenario in which despite the existence of alarm causalities, no causal alarm can be inferred due to reciprocal causal relations among certain process variables.

Scenarios as the one illustrated in the last example are in general difficult to diagnose. However, in certain cases their occurrence may be mitigated by adjusting the defined search stop criteria, particularly the number of times given nodes and edges can be used in a path. While this strategy proves useful in dealing with the effect caused by control loops and process recycles, it cannot mitigate the effect of reciprocal relations caused by inherent links in a component. An example for this is the relation between temperature and pressure in a closed vapor tank. By the above strategy, neither variable could be identified as causal alarm if both alarms were triggered, for there will always be an alarm that can be inferred as a cause of the other. In such cases, an exception must classify both alarms as causal, provided no other alarm is found to cause them.

It is worth noticing that causal analysis as described above can only pinpoint causal alarms, but not specific fault root-causes. However, based on identified alarms groups and respective causal alarms, the task of operators and process experts in finding fault root-causes is substantially simplified owing to the achieved reduction of the possible solution space. Alternatively, obtained causal results can be further processed by rule-based and data-driven approaches to draw automatic inferences relating causal alarms and root-causes. For example, if the flow in and out of an open tank are measured, observing *tank level* to be the causal alarm of the observed abnormal situation would lead to automatically identifying a tank leak as fault root-cause based on the analysis of flow behavior.

6.5.3.3 Path penalization and filtering strategies

Aiming at reducing the number of potential causal hypotheses resulting from the DCDG analysis, penalization and filtering clauses are applied to discovered propagation paths before diagnostic results are shown to the operator. Proposed strategies combine both qualitative and quantitative data on aspects regarding process deviations and temporal behavior, as well as types of variables and propagations paths. On account of flexibility, strategies are designed to be independent of each other, which allows users to select the penalization clauses and the respective order in which they should be applied depending upon the monitored system and the particular monitoring objectives.

a. Penalization based on deviation sign and magnitude

Dynamic process data is incorporated both qualitatively and quantitatively to penalize potential propagation paths based on the sign and magnitude of observed process deviations. As the signs of influence of each edge (λ_i) along a path are known, a given path can be ruled out when the observed deviation of any variable along the path does not match the predicted sign, i.e., when the path is found to be *qualitatively inconsistent*. Exceptions to this criterion apply in paths containing ambiguous edges and unmeasured nodes, cases in which a different penalization strategy aimed at dealing with uncertainties is used ($\rightarrow$ Strategy c).

In addition, it is desirable to remove paths including variables that are *qualitative consistent*, but do not deviate significantly enough from steady-state to propagate a disturbance. The magnitude of deviation should be accordingly considered to introduce further penalties, which may push unlikely paths below the threshold for consideration. Yet, there is a challenge to maintain general applicability at this point, because determining when a signal has deviated sufficiently to impact other signals is an arbitrary decision typically based on an expert notion. In developing an arbitrary penalization strategy that works for a specific process, there is no guarantee for similar effectiveness in other applications. To deal with such a obstacle, the penalization strategy must incorporate process-specific knowledge.

Process metrics such as absolute or relative deviations are dependent on several arbitrary factors like units. However, there is a source of information containing process-specific expert judgments about deviations: the alarm limits. Here, it is presumed that limits are set by process experts and control engineers in a reasonable way, such that they are triggered when process variables deviate "significantly". If the deviation from steady-state values is normalized by the low-to-high alarm range, then a ± 100 % deviation will almost certainly be significant, while a ± 5 % will likely not be responsible for disturbance propagations –except in the case of controlled variables. Accordingly, penalization factors, based on e.g., heuristics and statistical data, are defined for different normalized deviation magnitudes. Table 6.3 presents an example of a deviation-based penalization scheme.

Table 6.3: Quantitative penalization of propagation paths based on deviation magnitude. Example of possible deviations ranges and associated penalty factors ($p1 \leq p2 \leq p3 \leq p4 \leq 1$)

Normalized absolute deviation	Penalty factor
0 % – 10 %	$p1$
10 % – 25 %	$p2$
25 % – 50%	$p3$
$\geq$ 50 %	$p4$

While the above penalization concept may filter a large number of spurious propagation paths, caution is recommended when applying this strategy, since wrongly set alarm limits have been reported among the most common causes of alarm floods in process industries [WYCS15].

b. Penalization based on time

As expressed in equation 6.5.4, the time delay $|T_A - T_B|$ between a given pair of related alarms A and B can be estimated as the sum of the lags T_{dyn}, T_{thr}, and T_{cs} resulting respectively from the system dynamics, the defined alarm thresholds, and the alarm-related parameters of the control system (e.g., filter time constants and sampling period). Time delays modeled in the DCDG, however, only provide a first approximation of the lags related to system dynamics (T_{dyn}) and do not consider those delays introduced by other factors. As a consequence, the accumulated time delay of a propagation path linking two given alarms cannot be used to judge whether these are related by comparing observed and expected time delays in absolute terms. Nevertheless, the consideration of time lags predicted by the DCDG provides an intuitive basis to penalize those causal hypotheses with time delays differing from expected values in relative terms.

$$|T_A - T_B| \approx T_{dyn} + T_{thr} + T_{cs} \tag{6.5.4}$$

Provided $T_{dyn} \leq |T_A - T_B|$, the proposed penalization concept divides the time frame given by the magnitude of the delay between two alarms $|T_A - T_B|$ in n discrete time bins, and locates both the observed time delay T_{obs} and the expected time delay T_{exp} in the resulting bins based on their magnitude. Naturally, the observed time delay T_{obs} will always lie in the n_{th} bin, as its magnitude is equivalent to the size of the considered frame (i.e., $T_{obs} = |T_A - T_B|$). However, the expected time delay T_{exp}, which is equivalent to the lag caused by the system dynamics and predicted by the DCDG (i.e., $T_{exp} = T_{dyn}$), may lie in any of the n bins. The distance between observed and expected time bins is penalized by defined factors p_1, $p_2 \ldots p_{n-1}$, setting stronger penalties to those bins lying far away from each other, i.e., $p_1 \leq p_2 \leq p_{n-1} \leq 1$. In this way, those alarms

with time delays differing significantly –in relative terms– from those predicted by the DCDG are penalized. If the accumulated strength of potential propagation paths was already low, the time penalization may force those paths to be rejected or ranked as less likely. The number of time bins and the magnitudes of the respective penalization factors may be defined based on the statistic analysis of previous process alarms or by considering heuristic rules. The user can decide whether time penalization is to be applied or not based on the properties of the system and the particular monitoring objectives. Figure 6.10 illustrates the proposed time penalization concept.

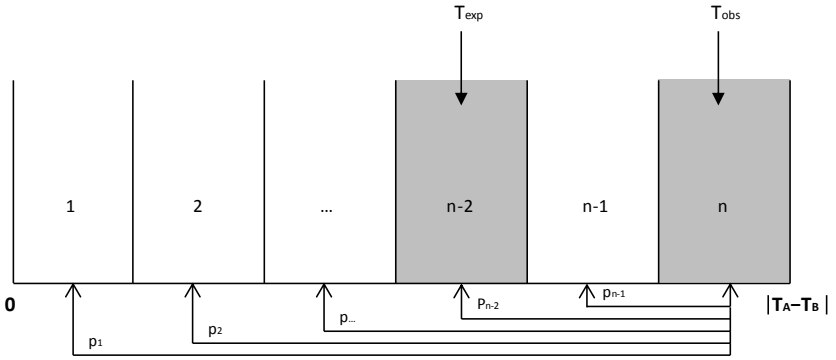


Figure 6.10: Graphical representation of the time penalization concept

c. Penalization of ambiguous edges and unmeasured variables

As previously explained in this section, the deviation-based strategy (→ Strategy a) penalizes those paths found to be qualitatively inconsistent. Yet, due to the existence of ambiguous arcs and unmeasured nodes in the DCDG, in some cases paths cannot be simply regarded as consistent or inconsistent, but instead must be deemed *ambiguous.* While disturbances may propagate through ambiguous paths, the dynamics of such propagations cannot be verified through process data as certain required measurements are not available. In view of the underlying propagation uncertainty, a given penalty factor p is applied to each ambiguous edge and unmeasured node in the path. Thus, the overall strength of paths containing multiple ambiguities may be ranked as less likely or forced below the defined acceptance threshold.

d. Filtering based on type of propagation carrier

The description of different connection links allowed by the DCDG (i.e., direct, component inherent, and information links) enables the application of a further filtering strategy based on semantics. By applying high level queries of the type "*search for propagation paths including/excluding x type of connection links*", for instance, this strategy makes it possible to analyze the individual effect of information and material connections on process disturbance propagations. A possible application of this strategy is foreseeing during detailed troubleshooting; for example, to analyze if a given control loop was responsible for propagating a certain fault.

6.5.3.4 Visualization of diagnostic results

After all plausible paths have been identified and filtered, diagnostic results are presented to the operator for visualization. To that end, the algorithm generates four different representations of inferred causalities: (*i*) a colored DCDG highlighting those nodes which triggered alarms as well as the edges found to link these nodes, (*ii*) a graph-based representation of the process schematic highlighting those components found to be affected by the disturbance propagation, (*iii*) a colored

connectivity matrix depicting the connectivity between alarms and the resulting alarm groups, and (*iv*) a list of ranked propagation paths with respective causal alarms. The user (i.e., operator or process expert) may choose which of these representations is more convenient to visualize the information required for his current task.

6.6 Validation

The proposed method was validated on the analysis of alarms and disturbance propagations in the TEP model controlled with Luyben's plant-wide regulatory control scheme [Luy96]. The TEP specifies 28 process faults (see *built-in disturbances* in Appendix H), from which 7 were found to trigger multiple alarms and accordingly were used for testing. In addition, 11 anomalies caused by valve failures were considered during experimental trials (see *additional disturbances* in Appendix H). The resulting 18 different plant and process disturbances were used as base ensemble for the generation of test scenarios including single and multiple faults. During conducted tests, the monitoring focus was set on the analysis of disturbances propagating along main process streams, omitting explicitly process utilities. The goal of the diagnostic procedure was to group related alarms triggered during fault scenarios, identifying in each case the respective causal alarm. The following sections describe the used experimental setup as well as the obtained diagnostic results.

6.6.1 Experimental setup

6.6.1.1 Information sources

As previously discussed in this chapter, the presented diagnosis concept combines plant topology, static and dynamic process information, and first physical principles for the discovery of disturbance propagation paths linking observed process alarms. During validation, this information was extracted from three main sources: a formalized plant topology model, a process simulation model, and a propagation look-up table.

a. Plant topology model

The TEP integrated topology model presented in Chapter 4 was used as connectivity and directionality information source, as well as data repository containing dimension-specific, component-specific, and process-specific information applied for the evaluation of system causalities.

During the generation of dynamic causal digraphs, relevant information in form of attributes in the topology model was retrieved through the proposed fit-to-purpose data access methods, specifically the AFDD PDS, PKS, and PS facets (→ 4.4.3). By consulting the respective *Timebehavior* property of every retrieved attribute, the algorithm correctly differentiated between static and dynamic information. While the content of static attributes was directly collected from the topology model, dynamic attributes were only interpreted as references to the actual data content found in another data source, in this case in the simulation model.

b. Simulation model

The MATLAB TEP simulation model originally proposed by Downs and Vogel [DV93], and recently revised by Bathlet et al. [BRJ15] was used to reproduce the system dynamics occurring during plant and process anomalies. Process signals referenced by attributes with dynamic *Timebehavior* properties in the topology model were retrieved from the simulation environment and used during the search and penalization of potential disturbance propagation paths.

c. Propagation look-up table

The propagation look-up table presented in Appendix I was used as fundamental repository of process causalities. As causal connections and qualitative relationships among variables and components in the PLUT stem from sound first physical principles, they are virtually valid for any chemical process. However, strength values and time delays (lags) presented in the particular PLUT of Appendix I have been tailored specifically for the TEP based on previous observations and heuristics. On account of simplicity, these values were defined as constants and not as general mathematical expressions. The reason for this relies in the fact that the ultimate aim of the presented method is not to delve into the definition of detailed process causal equations, but to present a new modular causal analysis approach whose level of detail can be adapted based on available causal information.

6.6.1.2 Configuration of process alarms

Since no standard alarms have been reported in the literature for the Tennessee Eastman Process, custom alarms were configured in the simulation model by considering normal and limit process operation values reported in [DV93]. In the interest of counting with high monitoring resolution, two alarm status levels, i.e., *ALM_LO_LO, ALM_LO*, *ALM_HI*, and *ALM_HI_HI*, were defined for every measured signal in the process main streams. In the particular case of composition measurements collected at the same process location, measurements were lumped together into a single concentration alarm functionality, which is triggered when one of the single thresholds is exceeded. For the purpose of the trials, no actuator failure alarms were configured.

6.6.1.3 Definition of path search and penalization thresholds

a. Path search thresholds

As discussed in Section 6.5.3.2, four numeric criteria are iteratively evaluated during the search for propagation paths. The following values were respectively used during conducted tests: maximum path length = 10, minimum overall strength threshold = 0.2, number of times a given node can be used in a path = 1, number of times a particular edge can be used in a path = 1.

b. Penalization settings

During validation, three penalizations strategies were applied for filtering potential paths. Table 6.4 lists those strategies and summarizes their specific parameterization.

Table 6.4: Penalization strategies used during conducted experimental tests

Penalization strategy	Criteria	Penalty factor
Deviation magnitude	**Normalized absolute deviation**	
	0 % – 10 %	$p_1 = 0.1$
	10 % – 25 %	$p_2 = 0.6$
	25 % – 50 %	$p_3 = 0.8$
	$\geq$ 50 %	$p_4 = 1.0$
Time delays	**Number of bins**	
	5	$p_1 = 0.1$
		$p_2 = 0.3$
		$p_3 = 0.6$
		$p_4 = 0.8$
Uncertain paths	**Penalized items**	
	Ambiguous arcs and unmeasured nodes	$p_1 = 0.95$

6.6.2 Diagnosis results

By using the described experimental setup, the proposed diagnosis method was tested on the analysis of 30 fault scenarios including single, double, and triple disturbances. Table 6.5 reports the obtained diagnosis results. As can be observed, the report shows the fault number as listed in Appendix H, the type of fault (i.e., single faults vs. multiple faults triggered simultaneously or in time intervals), the immediate process variable affected by the anomaly, the number of alarms triggered during the abnormal situation, and the position of the causal alarm in the analyzed log. In addition, it presents the result (success, partial success, or failure) of the diagnosis procedure, as well as the type of error (i.e., false positive (P) vs false negative (N)) for those cases in which the method failed in detecting true causalities or assigned false alarm relations.

Table 6.5: Experimental diagnosis results. In this report successful detections (✓) refer to those in which the correct alarm groups were identified and the right causal alarms diagnosed, partially successful detections (⊙) make reference to those in which more than one causal alarm hypothesis was found for a given alarm group, and failure detections (✗) are those in which alarms were mis-grouped or causal alarms were not detected

IDV	Fault type	Disturbed variable	Alarms triggered	Causal alarm position in log	Result	Error
1	Single	Comp. A+C feed	5	1	✓	–
2	Single	Comp. A+C feed	3	1	✗	N
4	Single	Temp. CW R003	51	3	✓	–
5	Single	Temp. CW C115	12	5	⊙	–
6	Single	A feed loss	24	1	✓	–
8	Single	Comp. A+C feed	8	1	✓	–
13	Single	Reaction kinetic	10	1	✓	–
1, 2	Simultaneous	See above	6	1, 2	✓	–
1, 4	Simultaneous	See above	50	2,18	✓	–
1, 5	Simultaneous	See above	18	1,5	⊙	–
1, 6	Simultaneous	See above	41	1,2	✓	–
1, 13	Simultaneous	See above	14	1,3	✗	P
4, 5	Simultaneous	See above	43	4,8	✗	P
4, 6	Simultaneous	See above	74	1,3	⊙	–
5, 6	Simultaneous	See above	16	1,6	✓	–
XMV	**Fault type**	**Disturbed variable**	**Alarms triggered**	**Causal alarm position in log**	**Result**	**Error**
1	Single	Flow D feed	52	1	⊙	–
2	Single	Flow E feed	60	1	✓	–
3	Single	Flow A feed	24	1	✓	–
4	Single	Flow A+C feed	44	1	✗	N
5	Single	Flow K100 recycle	28	1	✓	–
6	Single	Flow Purge	5	1	✓	–
7	Single	Flow S012 bottom	13	1	✓	–
8	Single	Flow E005 bottom	17	1	✓	–
10	Single	Flow R003 CW	32	6	⊙	–
11	Single	Flow C115 CW	22	4	⊙	–
7,8	Simultaneous	See above	12	1,2	✓	–
1,2,3	Simultaneous	See above	44	1,2,5	✓	–
1,2,3	15 min interval	See above	38	1,7,14	✗	P
1,3,2	15 min interval	See above	45	1,3,6	✓	–
3,6,8	Simultaneous	See above	24	1,2,6	✓	–

Experimental results presented in Table 6.5 revealed 19 successful detections, 6 partial detections, and 5 mis-detections. As summarized in Figure 6.11, these absolute figures evidentiate that the algorithm successfully or partially successfully diagnosed 83 % of the analyzed cases, while failed at finding the causal alarm for 17 % of the fault scenarios. The causes of partial and failed detections are various: detailed accounts on these cases are presented in the next subsection.

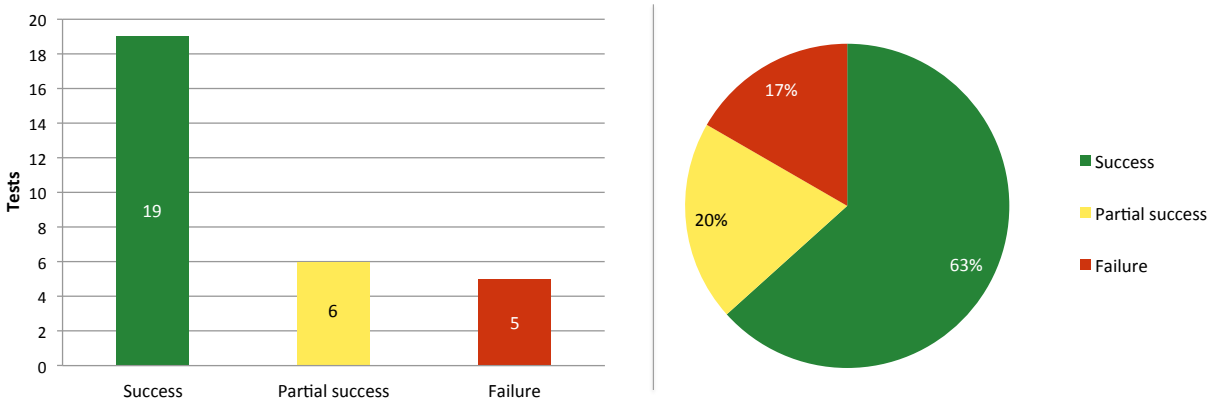


Figure 6.11: Summary of obtained diagnosis results

6.6.3 Analysis of partial and failed detections

6.6.3.1 Partial detections

As previously mentioned, partial detections are those in which more than one causal alarm hypothesis is generated for a given alarm group. During experimentation, the analysis of the following six fault scenarios resulted in partial detections.

1. **IDV 5**: Due to existence of reciprocal alarm pairs, four potential causal alarms were identified for this scenario.
2. **IDV 1,5*:*** IDV1 was correctly diagnosed, however, three possible causal alarms for IDV5 were suggested due to the existence of reciprocal alarm pairs. In both cases, the alarm groups were correctly identified.
3. **IDV 4,6**: Root IDV6 was correctly diagnosed, but six candidates for a potential second causal alarm were proposed due to the combined effect of control loops and process recycles.
4. **XMV 1**: With the D feed stream completely cut off, the pressure in the reactor feed pipeline (mixing zone) decreased significantly, thus causing an important increase of the flow exiting the stripper headspace. As no pressure sensor exists in the mixing zone, the causal algorithm diagnosed both the D feed flow and the stripper outflow as potential causal alarms.
5. **XMV 10**: Another scenario in which the lack of a pressure sensor in the mixing zone caused the detection of more than one hypothesis, specifically three potential causal alarms.
6. **XMV 11**: Due to the existence of reciprocal alarm pairs, four causal alarm were diagnosed for this scenario. A detailed description of the system behavior during this fault was previously presented in Section 6.1.

The analysis of the aforementioned accounts allows concluding that the main causes for partial detections are the existence of reciprocal alarm pairs (e.g., temperature and pressure) and the lack of certain process measurements. The first cause can be approached by using complementary process knowledge, for instance in form or expert rules, whereas addressing the second requires additional instrumentation, or, alternatively, the application of soft sensors and data-driven analysis algorithms. Due to the involved costs of acquiring additional instrumentation, to date the use of complementary methods seems to be a more suitable approach; however, the continuous growth of installed sensors in process plants suggests that the lack of measurements will be no longer an issue in near future, specially in the frame of Industry 4.0 [WKCB15].

6.6.3.2 Failed detections

Failed detections refer to those cases in which the algorithm did not succeed at determining the correct alarm group(s) or at finding the respective causal alarm(s). Among failed detections, two specific types are distinguished: false positives (P) and false negatives (N). The former type designates cases in which unrelated alarms were grouped, while the latter refers to scenarios in which related alarms were not associated. From the 5 failed detection cases observed during conducted experimental tests and reported below, 3 were false positives and 2 false negatives.

1. **IDV 2**: False negative. Due to alarm thresholds, only two sensors triggered alarms, i.e., FIR103 (A+C feed flow) and FIR115 (Purge flow). These measurements are located on opposite sides of the process. The connection among alarms can be detected, but the maximum allowed path length must be extended from 10 to 20, and there must be no penalty for unmeasured nodes.

2. **IDV 1,13**: False positive. IDV1 significantly alters the compositions throughout the entire process, thus the slow drift in reaction kinetics could not be clearly observed. As a consequence, only IDV1 was identified as a causal alarm.

3. **IDV 4,5**: False positive. IDV4 was correctly identified, but IDV5 was interpreted as an effect of IDV4. The short pipelines of the TEP and the resulting lack of significant transport delays make the simultaneous temperature spikes in the reactor (R003) and the separator (S012) consistent both with a disturbance only caused by IDV4, and with a fault caused simultaneously by IDV4 and IDV5.

4. **XMV 4**: False negative. The algorithm was unable to find the connection between the reduction of A+C feed flow and the increase in stripper (E005) temperature. The actual cause of the observed temperature increase is unclear, as the stripper pressure dropped while inlet feed temperature decreased. A potential explanation for this phenomenon is a drastic composition change in the stripper leading to a shift in VLE, which in turned caused rapid condensation and heat release. The observed slight increase of the stripper level supports this hypothesis.

5. **XMV 1,2,3*:*** False positive. When the D feed fails before the A and E feeds (15 min interval), the latter disturbances cannot be diagnosed because the former fault alters the magnitudes of the A and E feeds making the respective flow alarms look as consequential.

The accounts above evidentiate that the main cause of failed detections is related to the propagation of faults across the process, which makes the algorithm interpret causal alarms as consequences

of other causal alarms due to process connectivity and consistent qualitative behavior. This effect is particularly observable in the TEP as no transport delays between process units are modeled, preventing the algorithm from penalizing certain causalities based on observed time lags. In order to address this drawback, data-driven methods (e.g., non-linear series analysis) can be applied on identified signals to verify the causality or independence between specific alarms.

6.7 Assessment

Section 6.6 has demonstrated the applicability of the proposed diagnosis method on the determination of alarm groups and the diagnosis of causal alarms. As revealed by the obtained experimental results, the method demonstrated a good performance recognizing or partially recognizing 83 % of the analyzed fault scenarios and failing at diagnosing only 17 % of the cases. Considering the simplicity of the used expressions for causal properties (i.e., influence strengths and time lags), the little manual effort required for the implementation of the concept in the analyzed system, and the overall high complexity of the Tennessee Eastman Process, obtained diagnosis results are regarded as highly positive.

Detailed accounts on cases in which the algorithm did not succeed at fully diagnosing plant abnormal situations alongside provisions on how to address current drawbacks were provided. Overall, it was concluded that with additional instrumentation and/or complementary use of rule-based and data-driven methods, the diagnosis reliability of the proposed concept could be further enhanced. In the remaining of this section, the main advantages and limitations of the approach are summarized.

6.7.1 Advantages

Compared to existing approaches in academia and current industrial practice (→ 6.4), the diagnosis method proposed herein offers the following key advantages:

- It is based on a modular causal concept, which clearly separates the knowledge-base from the model instance. This allows for effective and largely automated implementation in different processes as well as for efficient knowledge and model maintenance.
- Unlike most knowledge-based methods, the proposed approach is process-generic as it is built on topology-based relations and process causalities of general validity. Accordingly, it can be applied with little manual effort in a large number of systems within sectors ranging from chemical, nuclear, to oil and gas industries.
- It integrates connectivity, directionality, first physical principles, as well as component-specific and process-specific data for model generation. Therefore, causal models derived from the application of the approach are capable of representing not only the structure of the system but also the internal relationship between process variables describing its physics and chemistry. This in turn allows for granularity and accuracy during plant abnormal situation analysis.
- Due to its dynamic and event-triggered nature, it allows depicting up-to-date process causalities in contrast to other approaches based on static causal models.

- The diagnosis accuracy of the method is adjustable. Both standard plant asset catalogues of general validity as well as vendor-specific catalogues of higher precision can be stored in the PLUT and used according to the required diagnosis resolution and information availability. In addition, penalization strategies can be selected and configured by the user depending upon the particular system.
- It does not require large amounts of process data to make inference on causal alarms and consequently on potential fault root-causes.
- The approach is not sensibly affected by the existence of control loops and process recycles, as it does not delve into spectral analysis but only into the observation of qualitative signal behaviors. This, in turn, results in a further advantage: its execution does not demand high computational resources.
- It might be applied both for fault propagation analysis during fault detection and diagnosis, and within alarm management.
- In contrast to previous approaches to alarm management, it generates not only alarm groups but also identifies potential causal alarms. According to industry recommendations on process diagnostics [NAMUR NA96, 2017]#, this feature can significantly facilitate the task of plant operators when facing process anomalies.
- Different to existing modular causal approaches, it neither requires complex mathematical models to describe the relations between process variables, nor optimization methods and additional non-causal equations for the generation of consistent process causal models.
- It can be applied in conjunction with other methods (e.g., data-driven approaches) with overall low computation resources. Although complementary methods might demand extensive computations, applying the proposed approach as front-end during diagnosis allows filtering the number of process signals that should be analyzed, thus leading to a significant reduction of required resources and execution times.

6.7.2 Limitations

Despite offering several advantages, the proposed method has also limitations and requires some manual steps. Among them:

- It requires the definition of multiple expressions or constants for describing the causal properties (i.e., influence strengths and time lags) between process variables among and within components in the propagation look-up table. Likewise, it demands the specification of penalization factors for the different proposed path filtering strategies.
- It relies on dense process instrumentation for detailed diagnosis. The lack of relevant process measurements may prevent the method from finding single causal alarms in certain fault scenarios.
- It has difficulties for dealing with the detection of multiple faults which appear to be related based on process connectivity and qualitative process behaviors. The solution of this limitation requires the application of complementary approaches such as data-driven techniques.

- It is affected by the existence of reciprocal alarm pairs stemming from internal relations among variables in certain plant components. In such cases, the method does not diagnose a single causal alarm but several instead (partial detection). Overcoming this drawback requires the application of rule-based approaches incorporating deep process know-how.

6.8 Chapter summary

This chapter has proposed a new modular causal methodology for the analysis of plant abnormal situations. The proposed diagnosis concept combines connectivity, directionality, and component-specific information extracted from a plant topology model with physical principles and dynamic process data for the discovery of disturbance propagation paths. In virtue of its generic and dynamic nature, this approach can be applied in fault detection and diagnosis, and alarm management in different production systems across chemical, nuclear, and oil and gas sectors. A case study of a simulated chemical production system, specifically the Tennessee Eastman Process, has been used to demonstrate the applicability of the concept on the analysis of alarms and disturbance propagations during different fault scenarios. Finally, the main advantages and current limitations of the method alongside provisions on how to overcome existing shortcomings have been discussed.

The next and final chapter of this thesis summarizes the different concepts and methods presented along the entire contribution, and discusses open issues and potential new applications to be explored in future works.

7. Summary and outlook

7.1 Summary

Legacy engineering documentation embodies a fundamental source of plant and process information which can be exploited within different modernization and operational activities in process facilities. The effective use of existing design documents, such as piping and instrumentation diagrams and control logic diagrams, can yield important economic savings by reducing the effort and time required for the completion of several tasks during projects execution. In addition, engineering diagrams form the basis for the derivation of digital plant models, fundamental artifacts within the digital production thread brought by Industry 4.0. Unfortunately, the heterogeneity and non-computer-interpretable nature of legacy documents hinder the automatic access of plant software systems (e.g., engineering, control, and enterprise tools) to the underlying data content, thus confronting engineers and plant personnel with the need to retrieve information manually when conducting required activities.

Although a number of approaches to this problem have been proposed in the academic and industrial sectors, existing methods and commercial solutions have mainly focused on scanning, indexing, and storing documents in centralized data management systems, as well as on generating digital models from CAx schematics. Despite these advances, up to now, state-of-the-art approaches are incapable of supporting the reliable recognition and formalization of legacy engineering diagrams on paper and elementary digital formats, particularly technical drawings composed by characters, symbols, connectivity, and underlying semantic content.

In view of such circumstances, this work was aimed at devising a novel methodology to enable the automatic capture, formalization, integration, and exploitation of legacy engineering documents within plant modernization and operational activities. This purpose was essentially motivated by the hypothesis that such a methodology may be an important driver for the effective application of automation of automation (AoA) in different industrial use cases including process simulation and plant abnormal situation analysis.

The first phase of the methodology, *information capture*, was addressed in Chapter 2. Here, a recognition approach for the automatic extraction and digitization of data contained in legacy engineering documents, specifically P&IDs and CLDs on paper and elementary digital files, was proposed. The presented approach comprises two main methods, a raster-based method for processing scanned records and rasterized images, and a vector-based method for the analysis of documents in vector graphics formats. Experimental results obtained during the analysis of sample schematics demonstrated a robust recognition performance of symbols, text, connectivity and underlying semantic content, and a consistent representation of captured information in form of connectivity matrices. It was discussed, how resulting digital representations can support the execution of basic engineering tasks such as the generation of part lists or the solution of equipment-related queries, as well as serve as a basis for the derivation of formal object-oriented models. Motivated by the potential benefits enabled by this approach, the introduced recognition concept has been deployed on commercial engineering and control software products of a world-renowned automation manufacturer.

Chapter 3 addressed the second phase of the methodology, *information formalization*, by introducing a method for the generation of object-oriented models based on the analysis of connectiv-

ity matrices. Aiming at seamless data exchange, the standardized format CAEX/AutomationML [IEC 62424, 2008][IEC 62714, 2014]# was chosen as metamodel for the storage of generated models. Accordingly, the object and interface information models originally presented in [HMAF16]* and [AHFR16]* were extended to allow for the representation of plant components, instrumentation, and control devices typically found in process facilities, as well as to enable the modeling of different interfaces and connection types. Resulting P&ID and CLD formal models can be regarded as "smart engineering documents", which allow not only humans but also computers retrieving plant and process information based on high-level queries.

The fourth phase of the methodology, *data integration*, aims at centralizing plant and control information contained in formal P&ID and CLD models as well as in other sources of process knowledge. Accordingly, Chapter 4 proposed a method for integration, access, and management of plant models and complementary process information as a basis for effective information retrieval. The method consists of a concept for merging P&IDs and CLDs, a data collection and classification schema, and a series of concepts for fit-to-purpose data access and information handling. The resulting integrated topology model constitutes a seamless source of plant and process information ready to be exploited within different AoA use cases.

With the purpose of illustrating the applicability of formalized and integrated plant models within AoA applications, Chapters 5 and 6 approached the last phase of the methodology, *information exploitation*. Chapter 5 presented an extended version of the method originally proposed in [Bar11] for automatic generation of low-fidelity plant simulation models. The presented extension proposes the deployment of control devices as a part of the simulation model in addition to the respective connections to real controllers in the PCS. It was shown that this extension allows not only for control code testing during FAT, but also for the assessment of qualitative closed-loop process responses supporting the execution of basic automation engineering tasks. In addition, provisions on how to adapt the level of detail and widen the applicability of the approach towards further applications, such as fault detection and diagnosis and operator training simulators, were discussed. Overall, combined with the methods for analysis, formalization and integration of engineering documents presented in Chapters 2, 3, and 4, the proposed method is the first one allowing for the exploitation of legacy plant and process data sources as a basis for automatic derivation of simulation models. Such feature can enable a significant reduction of time and modeling efforts during (re)engineering tasks both in brownfield and greenfield projects within a number of industries including chemical, pharmaceutical, oil and gas, and power generation.

Finally, Chapter 6 illustrated a further AoA application of integrated plant topology models. With that aim, a new modular causal methodology for the analysis of plant abnormal situations was proposed. The underlying diagnosis concept combines connectivity, directionality, and component-specific information extracted from a plant topology model with first physical principles and dynamic process data for the discovery of disturbance propagation paths. The method comprises two main parallel tasks: *(a)* the event-triggered generation of dynamic causal digraphs, i.e., modular causal models interconnected based on plant connectivity information, and *(b)* the recursive consultation of a propagation look-up table containing general and process-specific information for establishing updated process causalities within and among plant components. In virtue of its generic and dynamic nature, this approach can be applied in fault detection and diagnosis, and alarm management in different production systems. Experimental results demonstrated the applicability of the concept on the analysis of alarms and disturbance propagations during different fault scenarios in a complex chemical process model.

In conclusion, the main contributions of this work can be summarized as follows:

- A novel document recognition approach for the automatic extraction and digitization of information contained in legacy engineering documents, specifically in piping and instrumentation diagrams, and control logic diagrams.
- A new method for the generation of object-oriented models based on basic digital document descriptions in form of connectivity matrices.
- New concepts for merging P&ID and CLD formal descriptions into integrated plant topology models, as well as for aggregation, access, and management of different types of plant and process information.
- The extension of an existing method for automatic derivation of low-fidelity simulation models, which widens the applicability of the original approach beyond verification of control functions during FAT towards the evaluation of process closed-loop responses required during basic automation engineering.
- A novel causal method based on modular and dynamic structures for plant abnormal situation analysis which can be applied not only for disturbance propagation analysis but also for alarm grouping in different process facilities.

7.2 Outlook

This work has presented a functional methodology for capturing and exploiting plant and process data as a basis to support modernization and operational activities in process plants. Based on the main results and conclusions, several action items and possible topics for future research are identified in this section. In general, such items aim at a higher reliability and a wider application of the different methods proposed in the contribution. In the following, identified topics and action items are listed by respective methodology phase.

7.2.1 Information extraction

Engineering diagrams contain a large amount of different plant and process data. Yet, not all that data is relevant for a specific application, and in many cases its consideration may lead to spurious recognition results. To address this problematic, the proposed recognition approach was provided with filtering and consistency-check methods, which generate warnings when defined connectivity or naming rules are violated. Nonetheless, such methods cannot deal with certain ambiguities found during the analysis of complex diagrams, thus leading in some cases to failed detections. Accordingly, future research should concentrate on the development of further methods for filtering irrelevant information and guaranteeing recognition consistency. An interesting approach to be explored consists in incorporating statistical methods to predict or verify the existence of determined symbols in given sections of a diagram. Based on the occurrence frequency of certain symbols and the analysis connectivity patterns in the analyzed schematic, it would be feasible to define areas in which, for example, valves are likely to be found given an a priori identified tank or vessel. Such information can be used for prioritizing search regions or as a means for statistical consistency-check.

A further action item for future work is the extension of the scope of the recognition approach towards the processing of other graphic engineering documents, such as interlock schematics and electric diagrams. Supported by the proposed raster-based and vector-based recognition methods, this extension can be achieved by adapting the definition of nomenclature and connectivity rules based on the detailed analysis of the graphic and semantic features of the new document types.

7.2.2 Information formalization

On account of allowing for seamless data exchange, the presented data formalization approach uses CAEX/AutomationML [IEC 62424, 2008, IEC 62714, 2014] as metamodel for the storage of generated object-oriented plant models. Although this standard data format has been widely used in academia and recent industrial practice [SSD+15, BHH+16], many existing approaches and commercial tools in the market have used other standards (e.g., ISO 15926 [ISO 15926-1, 2004]) to represent and exchange plant topology data. Therefore, a useful extension of the proposed formalization concept is the automatic generation of topology models in other standard formats, which would allow for the exploitation of digitized legacy information within a larger spectrum of tools and applications.

7.2.3 Data integration

As a possible extension of the proposed data collection and aggregation concept, *eCl@ss* descriptive elements and lists of properties [IEC 61360, 2002] can be used to characterize components within the plant topology model in a standardized manner. As discussed in [RAF15]*, such descriptive structures can be seamlessly integrated into CAEX/AutomationML models and used as a semantic repository to specify both abstract components and specific devices. The use of standard properties with common semantics will allow external algorithms to retrieve model data without the need of data mappings and additional semantic repositories, thus facilitating the overall exploitation of plant topology models within AoA use cases.

A further research topic is the integration of formalized process descriptions [VDI/VDE 3682, 2014] and topology models as a basis for the generation of modular causal graphs. As preliminary shown in [ASC+14]* and [Chr15], integrated plant and process models allow depicting not only the structure of the plant and the characteristic of its components, but also the process sequence and the properties of the products generated in each process phase. Using these models in combination with the proposed modular causal approach would expand diagnostic capabilities towards the analysis of batch and semi-batch processes.

7.2.4 Information exploitation

Future research on information exploitation should focus on the application of integrated topology models and modular causal analysis in further industrial use cases.

Concerning plant topology models, one possible new application is the automatic generation of Human Machine Interfaces (HMIs) during plant modernization projects, specifically within control system migrations. In current practice, new HMIs must be manually created during the installation of the new control system, as the source code of the existing HMIs is proprietary and cannot be

reused by the new automation vendor. Therefore, automating the derivation of HMIs based on plant topology models stemming from legacy engineering documentation may leverage significant economic benefits by reducing engineering efforts.

With regard to modular causal analysis, an interesting topic for future research is the adaption of the proposed causal approach as a basis to support hazard and operability analysis (HAZOP). Here, dynamic causal digraphs and propagation look-up tables can be adapted to solve cause-effect queries by analyzing the connectivity paths described by material and information flows along the process. In this context, a further relevant topic is the exploration of graph-based big data approaches [BCD16] to accelerate the dynamic generation of modular causal models as well as to speed-up search procedures.

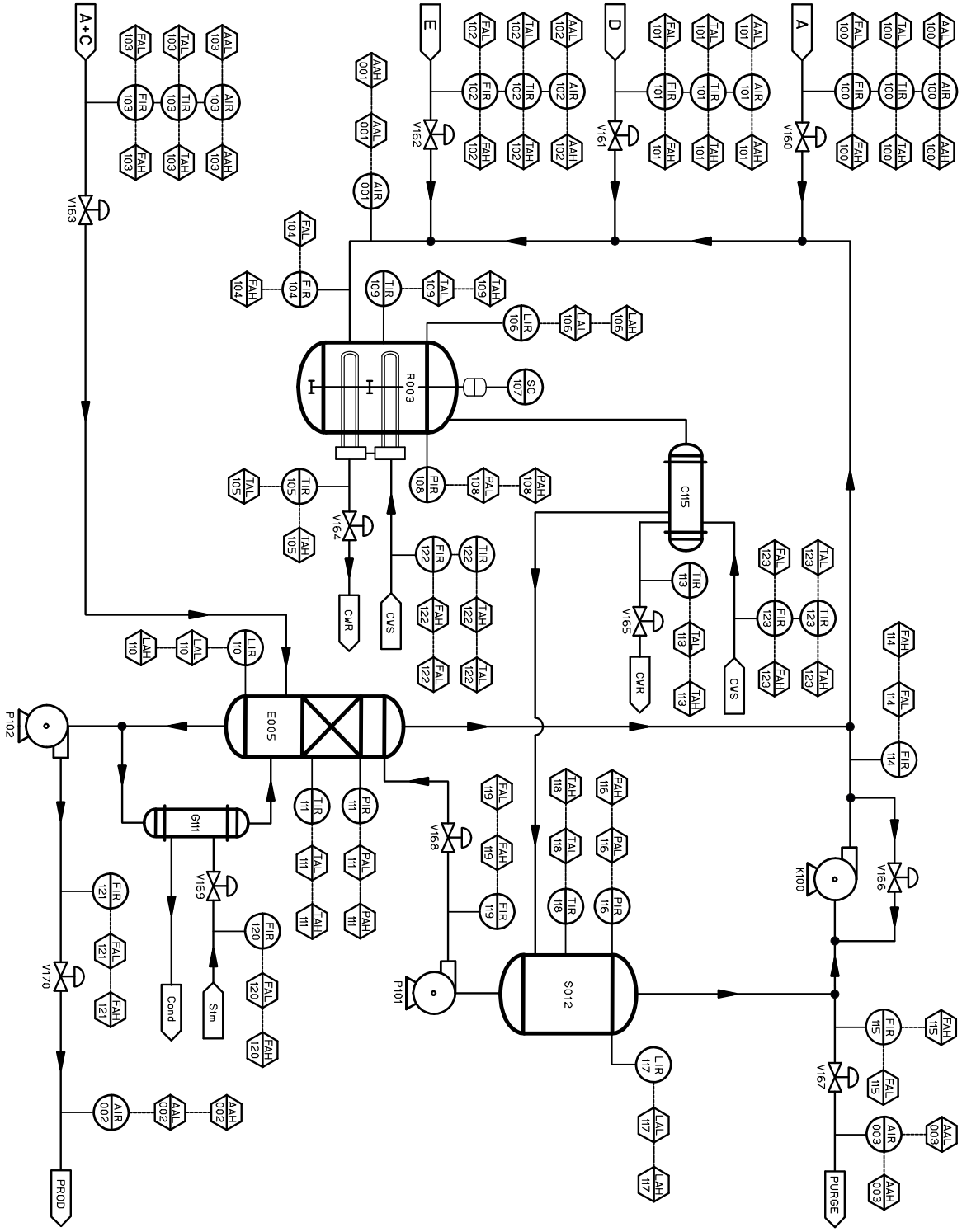
7.3 Chapter summary

Confronted with the competitive challenges brought by the ongoing digitization of the industrial sector –Industry 4.0, plant owners and contractors in the process industry are called to create and exploit digital plant models enabling the efficient execution and integration of activities across the plant lifecycle. In the case of existing process facilities, an important part of the information required for this task exists already in form of legacy engineering documents, such as scanned diagrams and schematics in elementary digital formats. Nevertheless, current engineering and enterprise tools cannot fully exploit this information due to the heterogeneity and non-computer-interpretable nature of the underlying data sources. As a consequence, process experts and engineers must often retrieve and consolidate information manually when conducting daily activities; practice that is not only error-prone, but also time-intensive, and costly.

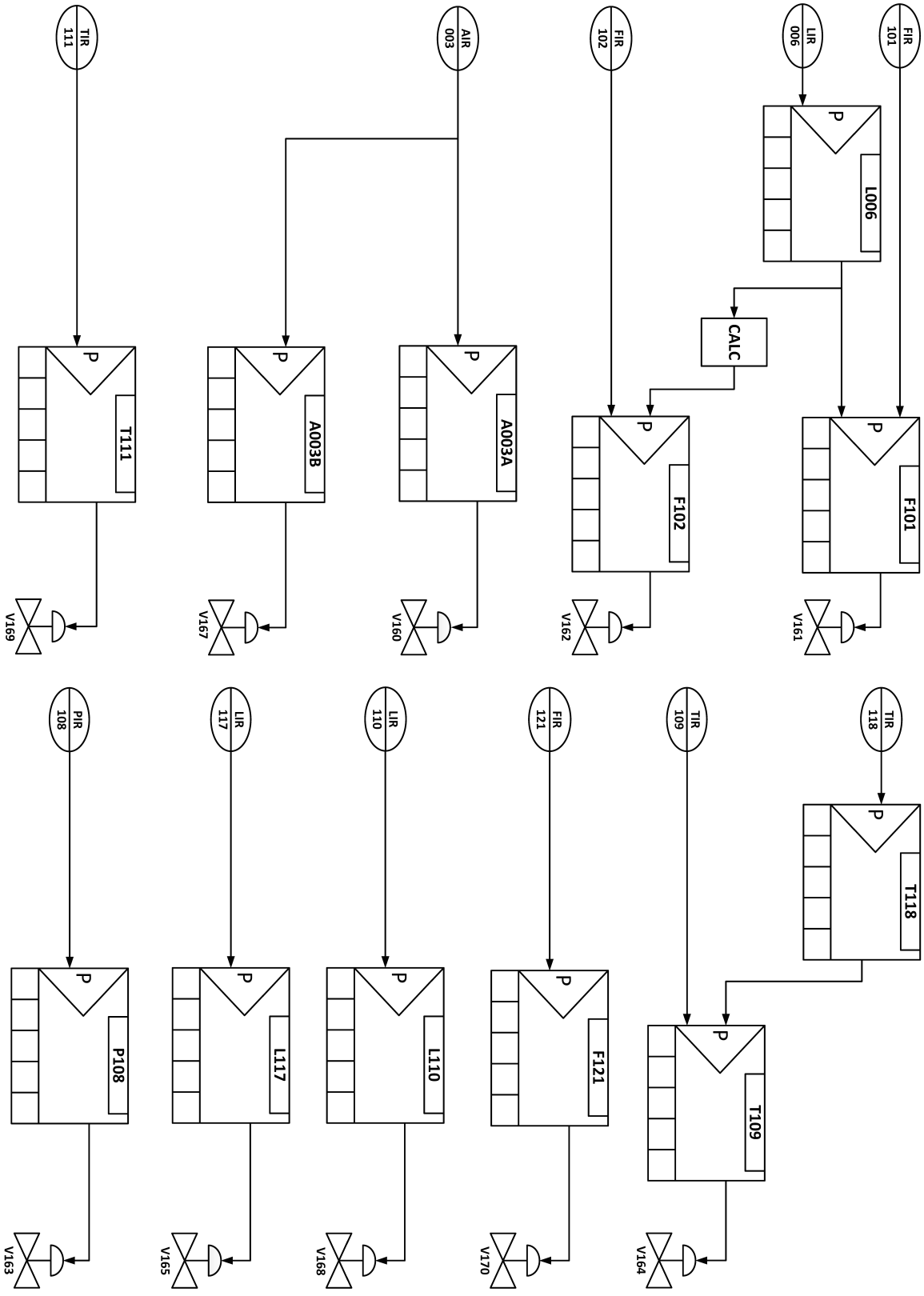
In an effort to cope this problem, the work reported in this thesis has created methods and tools for enabling the capture, formalization, integration, and subsequent exploitation of legacy design documents, specifically Piping and Instrumentation Diagrams, and Control Logic Diagrams. These developments have been demonstrated to serve as a basis for the automatic execution of required steps within automation-related plant modernization and operational tasks. Experimental results provide evidence of the effective applicability of the approach in two use cases of industrial relevance: (a) automatic generation of plant simulation models for the validation of basic control functions during plant modernization projects, and (b) fault propagation analysis for supporting alarm management and fault diagnosis during plant operation. This work has thus extended the field of automatic engineering information reuse in a rigorous and also practical way that has been designed for industrial impact in the frame of Automation of Automation.

Appendices

Appendix A: P&ID of the TEP



Appendix B: CLD of Luyben's regulatory control structure

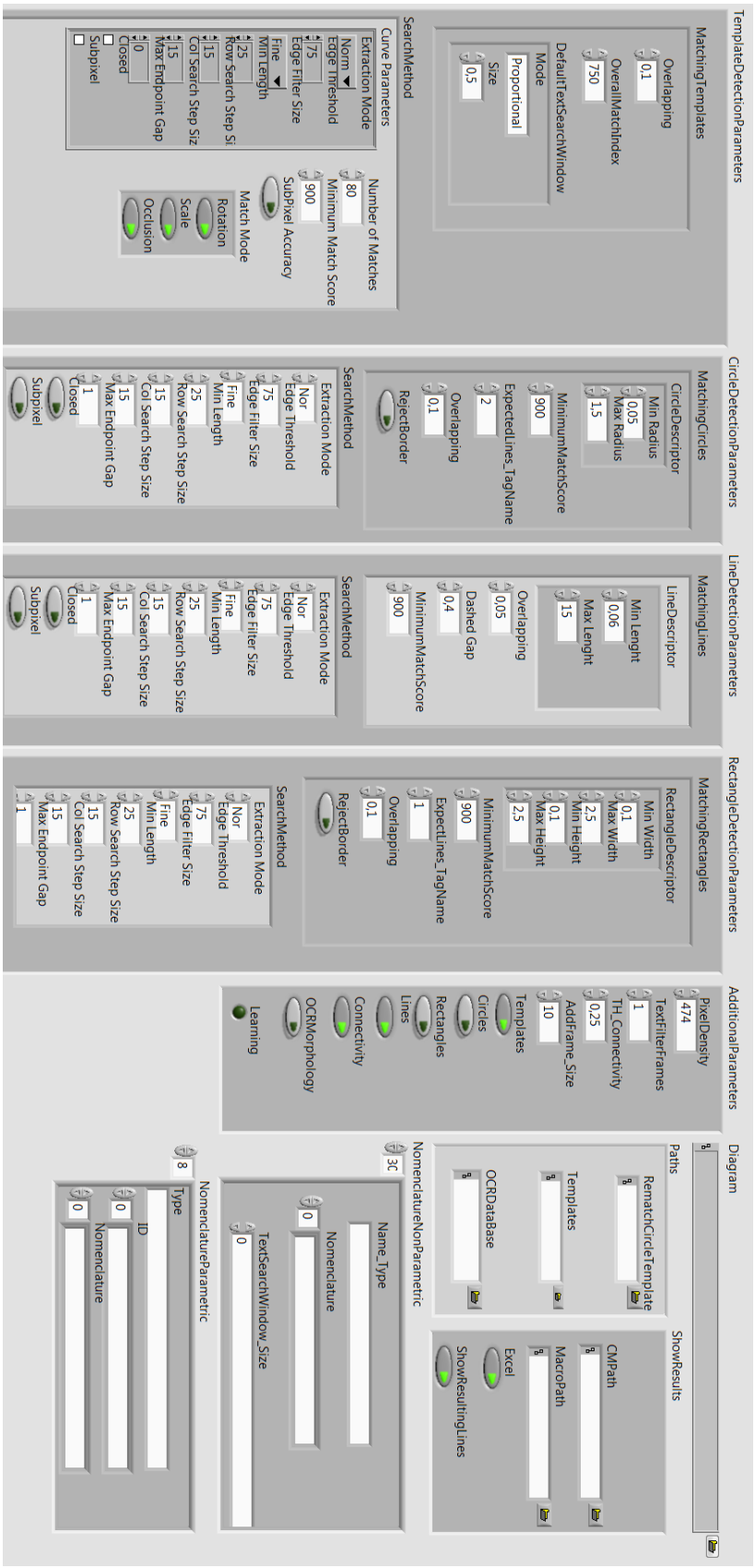


Appendix C: Reference designation for sensors and actuators according to ISO 14617

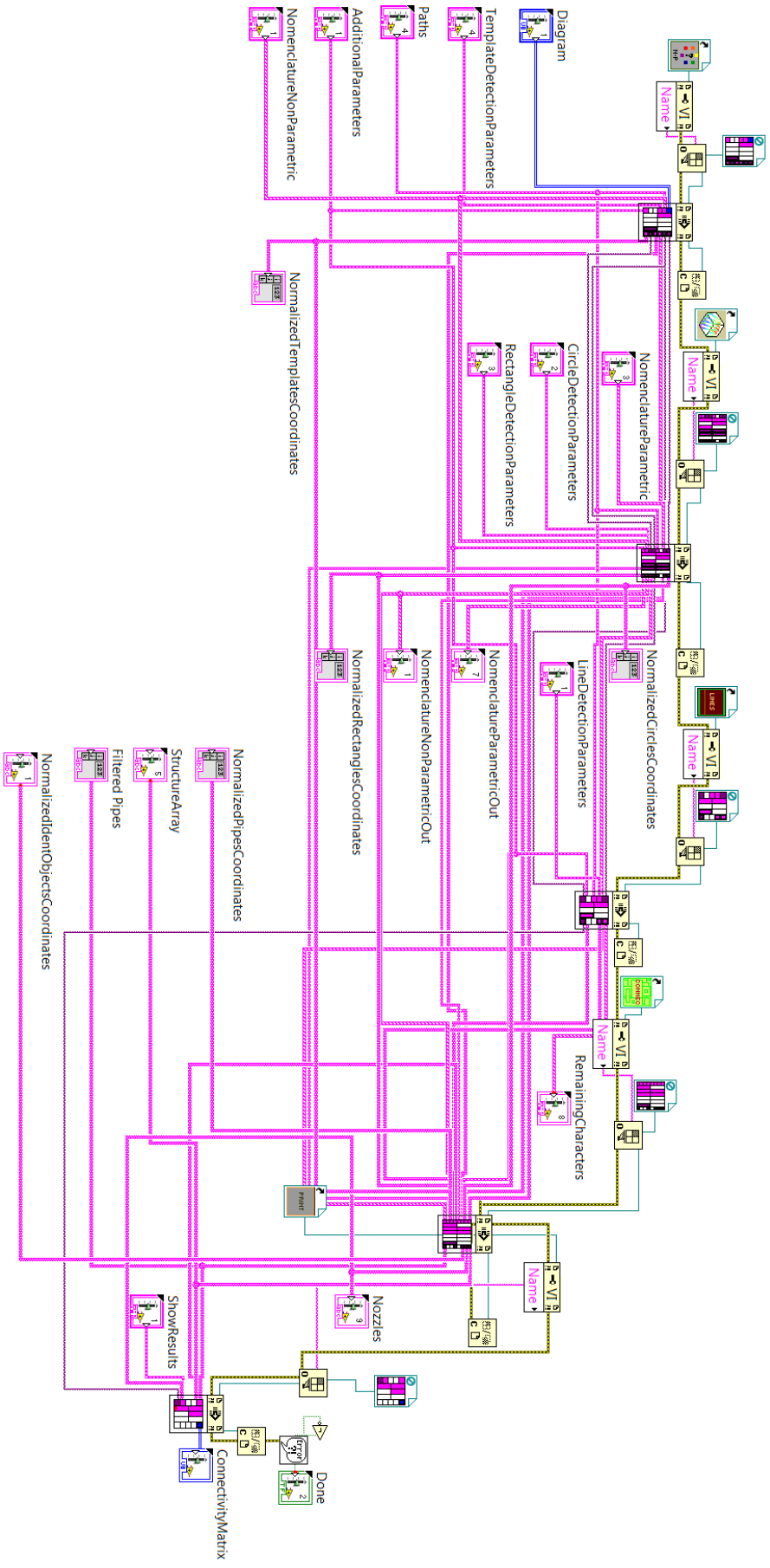
Designation	Magnitude	Designation	Magnitude
A	Analysis	N	UD (usually Torque)
B	Burner	O	UD
C	UD (usually Conductivity)	P	Pressure
D	UD (usually Density)	Q	Quantity
E	Voltage	R	Radiation
F	Flow	S	Speed
G	UD	T	Temperature
H	Hand	U	UD
I	Current	V	Vibration
J	Power	W	Weight
K	Time	X	UD
L	Level	Y	Event
M	UD	Z	Position

UD: User Defined

Appendix D: LabVIEW implementation of the raster-based method



D.1: Fragment of the parameterization interface (front panel)



D2: Block diagram depicting the main classes, inputs, and outputs of the method

Appendix E: Excerpt of collected AFDD-relevant data on the TEP

Information Type	Factor	Object	Value	Priority
Process	operation set points	R003	$T = 120.4\,^{o}C$	High
			$P = 2705\,kPa$	
		S012	$T = 80.1\,^{o}C$	
			$P = 3102.2\,kPa$	
	product properties	G (PROD)	$HV = 523\,kJ/kg$	High
		A	$VHC = 14.6\,kJ/kg^{o}C$	
		A+C	$0.485\,A$	
			$0.005\,B$	
			$0.510\,C$	
	operation limits	SC107	$S = 150 - 250\,rpm$	Moderate
		S012	$L = 30 - 100\,\%$	
Control	parameters	L006	$Kc = -2$	High
		P108	$Kc = -100$	
		T118	$Kc = -0.15$	
		A003A	$Kc = -10$	
Plant Item	dimensions	R003	$V = 25\,m^3$	Moderate
		S012	$V = 12\,m^3$	
		E005	$V = 8\,m^3$	
	types	R003	$Type = closed$	
Pipeline	role	P0020	$main$	High
		P0026	$utility$	
Alarm System	limits	LAH106	$hihi = 90\,\%$	High
		TAL105	$lolo = 100\,^{o}C$	
		PAL116	$lo = 2620\,kPa$	
PCE Measuring Point	ranges	LIR106	$0 - 100\,\%$	Moderate
		PIR108	$0 - 3000\,kPa$	
		TIR118	$0 - 100\,^{o}C$	
		AIR003	$0 - 100\,mol\%$	

F: Flow HC: Heat of Vaporization P: Pressure S: Speed T: Temperature V: Volume VHC: Vapor Heat Capacity

Appendix F: Excerpt of classified AFDD-relevant data on the TEP

Class	Factor	Object	Value	Priority
PDS	component dimensions	R003	$V = 25\,m^3$	Moderate
		S012	$V = 12\,m^3$	
		E005	$V = 8\,m^3$	
PKS	pipeline roles	P0020	*main*	High
		P0026	*utility*	
	component specs	R003	$Type = closed$	
PS	operation set points	R003	$T = 120.4\,^oC$	High
			$P = 2705\,kPa$	
		S012	$T = 80.1\,^oC$	
			$P = 3102.2\,kPa$	
	control parameters	L006	$Kc = -2$	Moderate
		P108	$Kc = -100$	
		T118	$Kc = -0.15$	
		A003A	$Kc = -10$	
	alarm limits	LAH106	$hihi = 90\,\%$	High
		TAL105	$lolo = 100\,^oC$	
		PAL116	$lo = 2620\,kPa$	
	operation limits	SC107	$S = 150 - 250\,rpm$	Moderate
		S012	$L = 30 - 100\,\%$	
	PCE measuring range	LIR106	$0 - 100\,\%$	Moderate
		PIR108	$0 - 3000\,kPa$	
		TIR118	$0 - 100\,^oC$	
		AIR003	$0 - 100\,mol\%$	
	product properties	G (PROD)	$HV = 523\,kJ/kg$	High
		A	$VHC = 14.6\,kJ/kg^oC$	
		A+C	$0.485\,A$	
			$0.005\,B$	
			$0.510\,C$	

F: Flow HC: Heat of Vaporization P: Pressure S: Speed T: Temperature V: Volume VHC: Vapor Heat Capacity

Appendix G: Modelica.Fluid object types

Modelica.Fluid type	Description
ClosedVolumen	Closed vessel typically used for mass holdup and pressure compensation
OpenTank	Standard vessel
StaticPipe	Ideal pipe with no roughness, mass and energy storage, or heat transfer
PrescribedPump	Ideal pump with controlled speed
ValveLinear	Continuous valve with controlled aperture and adaptable opening characteristics
ValveDiscrete	Binary valve (open/closed)
TeeJunctionIdeal	Junction/joint for three pipe endings or material flows
FixedBoundary	Boundary representing a material sink or material source. For pressure models, the nominal pressure of the boundary defines whether it is a source or a sink
Pressure	Pressure sensor
Density	Density sensor
Temperature	Temperature sensor
MassFlowRate	Mass flow sensor

Appendix H: TEP disturbances

Built-in disturbances

Variable number	Process variable	Type
IDV (1)	A/A+C feed ratio, B composition constant	Step
IDV (2)	B composition, A/A+C ratio constant	Step
IDV (3)	D feed temperature	Step
IDV (4)	Reactor cooling water inlet temperature	Step
IDV (5)	Condenser cooling water inlet temperature	Step
IDV (6)	A feed loss	Step
IDV (7)	C header pressure loss –reduced availability	Step
IDV (8)	A, B, C feed composition	Random variation
IDV (9)	D feed temperature	Random variation
IDV (10)	Reactor cooling water inlet temperature	Random variation
IDV (11)	Condenser cooling water inlet temperature	Random variation
IDV (12)	Reaction kinetics	Slow drift
IDV (13)	Reactor cooling water valve	Sticking
IDV (14)	Condenser cooling water valve	Sticking
IDV (15)	Unknown	Unknown
IDV (16)	Unknown	Unknown
IDV (17)	Unknown	Unknown
IDV (18)	Unknown	Unknown
IDV (19)	Unknown	Unknown
IDV (20)	Unknown	Unknown
IDV (21)	A feed temperature	Random variation
IDV (22)	E feed temperature	Random variation
IDV (23)	A feed pressure	Random variation
IDV (24)	D feed pressure	Random variation
IDV (25)	E feed pressure	Random variation
IDV (26)	A and C feed pressure	Random variation
IDV (27)	Pressure fluctuation in reactor cooling water	Random variation
IDV (28)	Pressure fluctuation in condenser cooling water	Random variation

Additional disturbances

Variable number	Process variable	Type
XMV (1)	Valve V161 fail closed	Step
XMV (2)	Valve V162 fail closed	Step
XMV (3)	Valve V160 fail closed	Step
XMV (4)	Valve V163 fail closed	Step
XMV (5)	Valve V166 fail closed	Step
XMV (6)	Valve V167 fail closed	Step
XMV (7)	Valve V168 fail closed	Step
XMV (8)	Valve V170 fail closed	Step
XMV (9)	Valve V169 fail closed	Step
XMV (10)	Valve V164 fail closed	Step
XMV (11)	Valve V165 fail closed	Step

Appendix I: Propagation look-up table for the TEP

Component	*Coordinates*	$(\Phi/\tau/\lambda)$	*Rationale*
Pipe	(F_{i-1}, F_i)	(.99/0/+)	Pipes serve only to propagate flow. They have no pressure gradient in the TEP model. The relation between pressure and flow is governed by volume object causalities. The likelihood to propagate a disturbance is near certain and instant due to their negligible length.
	(F_i, F_{i-1})	(.99/0/+)	A flow disturbance in pipes can propagate in both directions.
	(T_{i-1}, T_i)	(.99/0/+)	Temperature disturbances are propagated by flow. If transport delays were modeled in the TEP, then the propagation time would be estimated by dividing pipe length by flow velocity.
	(C_{i-1}, C_i)	(.99/0/+)	Same observation as for temperature propagation in pipes. For all paths that incorporate composition elements, the sign of propagation is forced to be zero or ambiguous.
Valve	(F_{i-1}, F_i)	IF: Aperture >0 (.99/0/+) Else (0/0/+)	Valves serve to regulate flow. The propagation of flow depends upon the aperture of the valve. The specific relation flow vs aperture is given by the type of valve, e.g., linear or equal percentage, and the type of fluid. For the purpose of this approach, however, the propagation of flow disturbances is simply modeled as a binary event: if the valve is open, the disturbance propagation is nearly certain and instant; otherwise, the propagation won't occur.
	(F_i, F_{i-1})	IF: Aperture >0 (.99/0/+) Else (0/0/+)	A flow disturbance in valves can propagate in both directions. The same argument presented above applies to this relation.
	(T_{i-1}, T_i)	IF: Aperture >0 (.99/0/+) Else (0.1/0/+)	Temperature disturbances are mainly propagated by flow. However, in some cases, even if the valve is closed, there might exist heat transfer by conduction.
	(C_{i-1}, C_i)	(.99/0/+)	Same observation as for temperature propagation. For all paths that incorporate composition elements, the sign of propagation is forced to be zero or ambiguous.
Pump, Compressor	(P_{i-1}, T_i)	(.1/0/−)	The thermal increase associated with vapor compression in the TEP recycle loop is modeled, although this effect is almost undetectable. An increase in inlet pressure decreases the pressure increase, thereby lowering outlet temperature.
	(P_i, T_i)	(.1/0/+)	Same observation as above. As outlet pressure increases, the larger pressure jump across the compressor produces a larger temperature increase.
	(F_{i-1}, F_i)	(.99/0/+)	Pumps and compressors serve to propagate flow. They have no pressure gradient in the TEP model. The relation between pressure and flow is governed in volume object functions. The ability to propagate a disturbance is near certain and instant.
	(F_i, F_{i-1})	(.99/0/+)	A flow disturbance in pumps and compressors can propagate in both directions.

	(T_{i-1}, T_i)	(.99/0/+)	Temperature disturbances are propagated by flow. If transport delays were modeled in the TEP, then the propagation time would be estimated by dividing pipe length by flow velocity.
Volume	(T_{i-1}, T_i)	(.99/.1/+)	An inlet temperature disturbance propagates with near certainty throughout a volume. It is not an instant effect.
	(C_{i-1}, C_i)	(.99/.1/0)	Inlet composition disturbance propagates through volumes.
	(F_{i-1}, P_i)	IF: single-phase, or two-phase with vapor inlet (.99/.05/+)	Pressure increases in a volume when inlet flows exceed outlet flows. This effect is not instant.
		IF: Two-phase volume, inlet liquid stream (.9/.05/+)	Upon entering a two-phase volume, liquids will near-instantly vaporize in accordance with VLE and cause a pressure increase. Slightly lower effect than a pure vapor stream's impact.
	(C_{i-1}, P_i)	IF: Two-phase, inlet liquid stream (.4/.05/0)	The composition of a liquid inlet stream can be strong in light components or strong in heavy components. If the inlet stream is typically full of heavy components, but is full of light components during a disturbance, then these light components are instantly vaporized upon entering the volume directly influencing volume pressure.
	(P_i, P_{i-1})	(.99/0/+)	Pressure disturbances in volume objects are propagated to the inlets.
	(P_i, F_{i-1})	(.9/0/−)	A pressure increase in volume objects decreases the flow of material into the volume.
	(F_{i-1}, C_i)	IF: Volume has multiple inlets (.9/.2/0)	If volume has multiple inlets, each feeding it a different composition, then altering the feed flow rates alters the volume composition.
	(F_{i-1}, L_i)	IF: Two-phase or open volume, inlet is liquid (.99/.1/+)	Volume level is dictated by accumulation of liquid.
	(P_i, T_i)	IF: Two-phase or vapor-only volume (.8/0/+)	Ideal gas law. The causality is bidirectional although the influence strength tends to be stronger from temperature to pressures, than vice-versa.
	(T_i, P_i)	IF: Two-phase or vapor-only volume (.99/0/+)	Ideal gas law. See above.
		IF: Liquid-only volume (.3/0/+)	Thermal expansion of liquid.

	(F_i, P_i)	IF: Liquid-only volume or Vapor-only volume $(.99/.2/-)$	Flow out of volume has a negative effect on volume pressure, which is controlled by accumulation.
	(L_i, P_i)	IF: Two-phase volume $(.3/0/+)$	Level change alters headspace volume, which has an ideal gas law effect on pressure. Higher level increases pressure.
	(C_i, P_i)	IF: Two-phase volume & no reaction $(.5.1, 0)$	VLE effect. If fewer light species make up the volume composition, pressure drops.
		IF: Two-phase volume & reaction $(.6/.05/0)$	The above VLE effect and reaction effect. If the characteristics of a reaction to consume or produce gaseous species were accessible in an easy logical form, then that would be incorporated into the IF condition. As without this information, it is uncertain whether the reaction consumes or produces gaseous species, a very cautious increase in probability is given.
	(T_i, C_i)	IF: Reaction $(.8/.15/0)$	Reaction rate increase due to temperature.
	(P_i, C_i)	IF: Vapor or two-phase volume & reaction $(.3/.15/0)$	Reaction rate increase due to pressure effect.
	(F_i, L_i)	IF: Open volume $(1/.1/-)$	Volume level dictated by accumulation.
	(T_i, L_i)	IF: Two-phase & no reaction $(.7/.1/-)$ IF: Two-phase & reaction $(.8/.4/0)$	If no reaction, then temperature acting through VLE effects alters the volume level. With reaction, the sign of increased temperature effect becomes ambiguous (unless it is known ahead of time details of all reactions in the volume), as it can increase reaction rate and produce more liquid mols to outweigh the VLE effect.
	(P_i, L_i)	IF: Two-phase & reaction $(.5/.1/0)$ IF: Two-phase & no reaction $(.4, .1, +)$	Same as above.

	(C_i, L_i)	IF: two-phase & reaction (.8/.4/0) IF: Two-phase & no reaction (.7/.3/0)	Same as above.
	(F_{i+1}, P_i)	IF: two-phase, outlet is vapor (.5/.25/−)	Outlet flow decreases pressure.
	(F_{i+1}, L_i)	IF: two-phase, outlet is liquid (.9/.1/−)	Outlet flow decreases level.
	(P_i, F_{i+1})	IF: two-phase reactor (.9/.1/−)	The pressure in the headspace drives both vapor out of the top of a two phase reactor and increases the pressure (in addition to hydrostatic pressure) that drives fluid out the bottom.
	(F_{i+1}, C_i)	IF: multiple outlets (.9/.1/−)	If each outlet removes different species from the reactor, their respective flow rates affect tank concentration.
Sensor	(SS, CS)	(.99/0/+)	High measured variable leads to high sensor signal. Topology model dictates which element is sensed and which controller is delivered the information.
Controller	(CS, AS)	IF : $Kc < 0$ (.99/0/+) IF : $Kc \geq 0$ (.99/0/−)	Controller response depends on the action direction, i.e., "direct" vs "inverse". Based on the output computation *out=Kc (SP-PV)*, it is said that if the controller output follows the PV, its action is "direct", and "inverse" on the contrary case. Thus if the proportional constant Kc is negative, the controller action is "direct" (+), and inverse (-), otherwise.
	(CS, CS)	IF : $Kc < 0$ (.99/0/+) IF : $Kc \geq 0$ (.99/0/−)	Cascade (master-slave) configuration. Same as above.
Actuator	$(AS, Actuator)$	(.99/0/+)	Valve actuators controls valve flow. Volume actuator represent the function of the cooling water system in jacketed reactors. The modeled volume actuator acts directly on the reactor temperature. Pipe actuators model the function of the cooling water system in total condensers. The modeled pipe actuator acts directly on the temperature of the stream being transported.

Bibliography

General literature

[AAWU15] R. Al-Aomar, E. Williams, and O. Ulgen, *Process Simulation using WITNESS*. Wiley, 2015.

[Abe14] L. Abele, "Resource monitoring in industrial manufacturing using knowledge-based technologies," Ph.D. dissertation, TU München, Fakultät für Elektrotechnik und Informationstechnik, 2014.

[ABF+07] J. Aslund, J. Biteus, E. Frisk, M. Kryssander, and L. Nielsen, "Safety analysis of autonomous systems by extended fault tree analysis," *International Journal of Adaptive Control and Signal Processing*, vol. 21, no. 2-3, pp. 287–298, 2007.

[Ahr01] W. Ahrens, "Computer Aided Engineering -eine Disziplin an der Nahtstelle zwischen Engineering und Informatik," *Automatisierungstechnische Praxis -Sonderausgabe "Perspektive Automatisierungstechnik"*, 2001.

[AOC+00] S. Adam, J. Ogier, C. Cariou, R. Mullot, J. Labiche, and J. Gardes, "Symbol and character recognition: application to engineering drawings," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 3, no. 2, pp. 89–101, 2000.

[AST95] C. Ah-Soon and K. Tombre, "A step towards reconstruction of 3D CAD models from engineering drawings," in *International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1, 1995, pp. 331–334.

[Bah10] T. Bah, *Inkscape: Guide to a Vector Drawing Program*. Pearson Education, 2010.

[Bal81] D. Ballard, "Generalizing the Hough Transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, no. 2, pp. 111–122, 1981.

[Ban98] J. Banks, *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*. Jonh Wiley & Sons, 1998.

[Bar11] M. Barth, "Automatisch generierte Simulationsmodelle verfahrenstechnischer Anlagen für den Steuerungstest," Ph.D. dissertation, Institut für Automatisierungstechnik der Helmut Schmidt Universität, 2011.

[BCC+07] M. Bauer, J. Cox, M. H. Caveness, J. Downs, and N. F. Thornhill, "Finding the direction of disturbance propagation in a chemical process using transfer entropy," *IEEE Transactions on Control Systems Technology*, vol. 5, pp. 12–21, 2007.

[BCD16] R. Buyya, R. Calheiros, and A. Dastjerdi, *Big Data: Principles and Paradigms*. Morgan Kaufmann, 2016.

[BD06] D. Bausa and G. Dünnebier, "Lifecycle modelling in the chemical industries: Is there any reuse of model in automation and control?" in *European Symposium on Computer Aided Process Engineering (ESCAPE)*, 2006, pp. 3–8.

[BDFR16] P. Bigvand, R. Drath, A. Fay, and P. Rodriguez, "Concept and development of a semantic based data hub between process design and automation system engineering tools: An approach to achieve iterative data exchange without the need of standardized semantics," in *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2016, pp. 1–8.

[BDN16] A. Bunte, A. Diedrich, and O. Niggemann, "Integrating semantics for diagnosis of manufacturing systems," in *IEEE International Conference on Emerging Technology and Factory Automation (ETFA)*, 2016.

[BDSS15] P. Bigvand, R. Drath, A. Scholz, and A. Schüller, "Agile standardization by means of PCE requests," in *IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, 2015, pp. 1–8.

[Ber06] M. Berbar, "Automatic diagrams analysis," in *IEEE Geometric Modeling and Imaging -New Trends (GMAI)*, 2006, pp. 160–170.

[BF13] M. Barth and A. Fay, "Automated generation of simulation models for control code tests," *Control Engineering Practice*, vol. 21, pp. 218–230, 2013.

[BFRR01] T. Bräunl, S. Feyrer, W. Rapf, and M. Reinhardt, *Parallel Image Processing.* Springer-Verlag Berlin Heidelberg, 2001.

[BHH+16] J. Bernshausen, A. Haller, T. Holm, M. Hoernicke, M. Obst, and J. Ladiges, "Namur Modul Type Package – Definition: Beschreibungsmittel für die Automation modularer Anlagen," *atp-edition / Automatisierungstechnische Praxis*, vol. 1-2, pp. 72–81, 2016.

[BJC+10] W. Birk, A. Johansson, M. Castano, S. Rönnbäck, T. Nordin, and N. Ekholm, "Interactive modeling and visualization of complex processes in pulp and paper making," in *Control Systems*, 2010.

[Bou09] A. Bouridane, *Imaging for Forensics and Security: From Theory to Practice.* Springer, 2009.

[BRJ15] A. Bathelt, N. Ricker, and M. Jelali, "Revision of the Tennessee Eastman Model," in *IFAC Symposium on Advanced Control of Chemical Processes*, 2015, pp. 309–314.

[BS10] M. Bellgran and K. Säfsten, *Production Development.* Springer New York Heidelberg Dordrechr London, 2010.

[BSF+09] M. Barth, M. Strube, A. Fay, P. Weber, and J. Greifeneder, "Object-oriented engineering data exchange as a base for automatic generation of simulation models," in *Annual Conference of IEEE Industrial Electronics (IECON)*, 2009, pp. 2465–2470.

[BT08] M. Bauer and N. Thornhill, "A practical method for identifying the propagation path of plant-wide disturbances," *Journal of Process Control*, vol. 18, pp. 707–719, 2008.

[BTM05] M. Bauer, N. Thornhill, and A. Meaburn, "Cause and effect analysis of chemical processes analysis of a plant-wide disturbance," in *IEEE Seminar on Control Loop Assessment and Diagnosis*, 2005, pp. 23–30.

[Bur04] A. Burdorf, "Extended Equipment-Modelling für die rechnergestützte Aufstellungsplanung von Chemieanlagen," Ph.D. dissertation, TU Dortmund, Fachbereich Bio- und Chemieingenieurwesen, 2004.

[CB03] L. Chiang and R. Braatz, "Process monitoring using causal map and multivariate statistics: fault detection and identification," *Chemometrics and Intelligent Laboratory Systems*, vol. 65, pp. 159–178, 2003.

[CD92] I. Chai and D. Dori., "Extraction of text boxes from engineering drawings," in *Conference on Character Recognition and Digitizer Technologies*, 1992, pp. 38–49.

[CFON11] L. Christiansen, A. Fay, B. Opgenoorth, and J. Neidig, "Improved diagnosis by combining structural and process knowledge," in *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2011, pp. 1–8.

[Chr15] L. Christiansen, "Wissensgestütztes Diagnosekonzept durch Kombination von Anlagestruktur- und Prozessmodell," Ph.D. dissertation, Institut für Automatisierungstechnik der Helmut Schmidt Universität, 2015.

[CJSF11] L. Christiansen, T. Jäger, M. Strube, and A. Fay, "Integration of a formalized process description into MS Visio with regard to an integrated engineering process," in *Workshop on Industrial Automation Tool Integration for Engineering Project Automation (iATPA)*, 2011.

[CMDC07] S. Chowdhury, S. Mandal, A. Das, and B. Chanda, "Segmentation of text and graphics from document images," in *International Conference on Document Analysis and Recognition (ICDAR)*, vol. 2, 2007, pp. 619–623.

[COP+06] F. Casella, M. Otter, K. Proelss, C. Richter, and H. Tummescheit, "The Modelica Fluid and Media Library for modeling of incompressible and compressible thermo-fluid pipe networks," in *Conference Modelica*, 2006, pp. 621–640.

[CRB01] L. Chiang, E. Russel, and R. Braatz, *Fault Detection and Diagnosis in Industrial Systems*. Springer, 2001.

[CSD06] R. Cox, J. Smith, and Y. Dimitratos, "Can simulation technology enable aparadigm shift in process control? Modeling for the rest of us," *Computers & Chemical Engineering*, vol. 30, no. 1542–1552, 2006.

[CY90] C. Chang and C. Yu, "On-line fault diagnosis using the signed directed graph," *Industrial & Engineering Chemistry Research*, vol. 29, pp. 1290–1299, 1990.

[CZZ10] L. Cui, J. Zhao, and R. Zhang, "The integration of HAZOP expert system and piping and instrumentation diagrams." *Process Safety and Environmental Protection*, vol. 88, pp. 327–334, 2010.

[DAIT11] G. Di Geronimo Gil, D. Alabi, O. Iyun, and N. F. Thornhill, "Merging process models and plant topology," in *International Symposium on Advanced Control of Industrial Processes (ADCONIP)*, 2011, pp. 15–21.

[DB11] R. Drath and M. Barth, "Concept for interoperability between independent engineering tools of heterogeneous disciplines," in *IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, 2011, pp. 1–8.

[DGT15] D. Dorantes Romero, T. Graven, and N. F. Thornhill, "Linking process, electrical and logical connectivity for supported fault diagnosis," in *European Symposium on Computer Aided Process Engineering (ESCAPE)*, 2015.

[DM07] D. Dobson and R. Martinez, "Integrated engineering: Engineering tool integration for process industries„" *ABB Review*, vol. 2, pp. 48–52, 2007.

[Dra05] R. Drath, "XML-basiertes Datenaustauschformat ermöglicht effizientes Leittechnikengineering," *CAV Chemie Anlagen Verfahren*, no. 5, p. 82, 2005.

[DT14] D. Dorantes Romero and N. F. Thornhill, "Integration, navigation and exploration of plant topology networks using the property-graph model," in *IEEE International Symposium on System Integration (SICE)*, 2014, pp. 743–748.

[DV93] J. Downs and E. Vogel, "A plant-wide industrial process control problem," *Computers & Chemical Engineering*, vol. 17, no. 3, pp. 245–255, 1993.

[DV00a] S. Dash and V. Venkatasubramanian, "Challenges in the industrial application of fault diagnostic systems," *Computers & Chemical Engineering*, vol. 24, pp. 785–791, 2000.

[DYCS13] P. Duan, F. Yang, T. Chen, and S. Shah, "Direct causality detection via the transfer entropy approach," *IEEE Transactions on Control Systems Technology*, vol. 21, pp. 2052–2066, 2013.

[Fit12] M. Fitzgerald, *Introducing Regular Expressions*. O'Reilly, 2012.

[FK88] L. Fletcher and R. Kasturi, "A robust algorithm for text string separation from mixed text/graphics images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 910–918, 1988.

[FPG04] I. Faragasan, S. Ploix, and S. Gentil, "Causal fault detection and isolation based on a set-membership approach," *Automatica*, vol. 40, no. 12, 2004.

[FSS09] A. Fay, T. Schmidberger, and T. Scherf, "Knowledge-based support of HAZOP studies using a CAEX plant model," *Functional Safety*, vol. 2, pp. 5–15, 2009.

[FWVHM12] J. Folmer, B. Weisenberger, B. Vogel-Heuser, and H. Meyer, "Diagnosis of automation devices based on engineering and historical data," in *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2012, pp. 1–4.

[GF13] M. Göring and A. Fay, "Method for the analysis of temporal change of physical structure in the instrumentation and control life-cycle," *Nuclear Engineering and Technology*, vol. 45, no. 5, pp. 653–664, 2013.

[GTLT95] J. Gao, L. Tang, W. Liu, and Z. Tang, "Segmentation and recognition of dimension texts in engineering drawings," in *International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1, 1995, pp. 528–531.

[GV09] M. Gellaboina and V. Venkoparao, "Graphic symbol recognition using auto associative neural network model," in *International Conference on Advances in Pattern Recognition*, 2009, pp. 297–301.

[GZMW09] D. Gao, B. Zhang, X. Ma, and C. Wu, "SDG multiple fault diagnosis by fuzzy logic and real-time bidirectional inference," in *International Conference on Information Engineering and Computer Science (ICIECS)*, 2009, pp. 1–8.

[HB07] M. Hollender and C. Beuthel, "Intelligent alarming," *ABB Review*, pp. 20–23, 2007.

[HCF14] M. Hoernicke, L. Christiansen, and A. Fay, "Anlagetopologien automatisch erstellen. Modelle aus der Mensch-Maschine-Schnittstelle erzeugen," *atp-edition / Automatisierungstechnische Praxis*, vol. 56, no. 4, pp. 28–40, 2014.

[Hen14] T. Henderson, *Analysis of Engineering Drawings and Raster Map Images*. New York Heidelberg Dordrecht London: Springer, 2014.

[HF94] C. Han and K. Fan, "Skeleton generation of engineering drawings via contour matching," vol. 27, no. 2, pp. 261–275, 1994.

[HFB15] M. Hoernicke, A. Fay, and M. Barth, "Virtual plants for Brown-Field projects: Automated generation of simulation models based on existing engineering data," in *IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, 2015, pp. 1–8.

[HH10] B. Hollifield and E. Habibi, *The Alarm Management Handbook*. PAS, 2010.

[HLK99] S.-H. Hahn, J. H. Lee, and J.-H. Kim, "A study on utilizing OCR technology in building text database," in *International Workshop on Database and Expert Systems Applications*, 1999, pp. 582–586.

[Hoe15] M. Hoernicke, "Topology engineering: Virtual plants for brownfield projects," *ABB Research Center Germany –Annual Report 2015*, pp. 60–64, 2015.

[HTNK11] T. Hamaguchi, K. Takaeda, M. Noda, and N. Kimura, "A method of designing plant alarm systems with hierarchical cause-effect model," in *International Symposium on Process Systems Engineering (PSE)*, 2011, pp. 265–269.

[Iyu11] O. Iyun, "Plant-wide diagnosis: Cause-and-effect analysis using process connectivity and directionality information," Ph.D. dissertation, Centre for Process Systems Engineering, Department of Chemical Engineering and Chemical Technology, Imperial College of Science, Technology and Medicine London, 2011.

[JFVH12] J, Folmer, and B. Vogel-Heuser, "Computing dependent industrial alarms for alarm flood reduction," in *International Multi-Conference on Systems, Signals and Devices (SSD)*, 2012, pp. 1–6.

[JFVH14] F. S. J. Folmer and B. Vogel-Heuser, "Detection of temporal dependencies in alarm time series of industrial plants," in *IFAC World Congress*, 2014, pp. 1802–1807.

[JSB08] N. Jamil, T. M. T. Sembok, and Z. Bakar, "Noise removal and enhancement of binary images using morphological operations," in *International Symposium on Information Technology (ITSim)*, vol. 4, 2008, pp. 1–6.

[JSL01] B. Juricek, D. Seborg, and W. Larimore, "Identification of the Tennessee Eastman Challenge Process with subspace methods," *Control Engineering Practice*, vol. 9, pp. 1337–1351, 2001.

[KCC98] N. Krim, B. Courtois, and B. Conq, "CADnet framework: An open platform for multidisplinary and collaborative design," in *Technologies for the Information Society: Developments and Opportunities*, 1998, pp. 827–834.

[KG13] I. Kaiser and P. Geiss, "Integrated engineering on the rise," *Process Worldwide*, vol. 1, no. 26-28, 2013.

[KIC+13] A. Kabir, I. Iyadi, T. Chen, D. Joe, and T. Burton, "Similarity analysis of industrial alarm flood data," *IEEE Transactions on Automation Science and Engineering*, vol. 10, no. 2, pp. 452–457, 2013.

[KP87] M. Kramer and B. Palowitch, "A rule-based approach to fault diagnosis using the signed directed graph," *AIChE Journal*, pp. 233–243, 1987.

[LG95] P. Lyman and C. Georgakis, "Plantwide control of the tennessee eastman process," *Computers & Chemical Engineering*, vol. 19, no. 3, pp. 321–331, 1995.

[LGFB94] L. Leyval, S. Gentil, and S. Ferray-Beaumont, "Model-based causal reasoning for process supervision," *Automatica*, vol. 30, no. 8, pp. 1295–1306, 1994.

[LHK10] A. Lüder, L. Hundt, and A. Keibel, "Description of manufacturing processes using AutomationML," in *IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, 2010, pp. 1 –8.

[LJJ16] R. Landman and S.-L. Jämsä-Jounela, "Hybrid approach to casual analysis on a complex industrial system based on transfer entropy in conjunction with process connectivity information," *Control Engineering Practice*, vol. 53, pp. 14–23, 2016.

[LK08] A. Ligęza and J. Kościelny, "A new approach to multiple fault diagnosis: A combination of diagnostic matrices, graphs, algebraic and rule-based models. The case of two layer models," *International Journal of Applied Mathematics and Computer Science*, vol. 18, no. 4, pp. 465–476, 2008.

[LKJJ14] R. Landman, J. Kortela, and S. Jämsä-Jounela, "Fault propagation analysis by combining data-driven causal analysis and plant connectivity," in *IEEE International Conference on Emerging Technology and Factory Automation*, 2014, pp. 1–4.

[Lu98] Z. Lu, "Detection of text regions from digital engineering drawings," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 4, pp. 431–439, Apr 1998.

[Luy96] W. Luyben, "Simple regulatory control of the eastman process," *Industrial & Engineering Chemistry Research*, vol. 35, pp. 3280–3289, 1996.

[LWC+13] Y. Lu, F. Wang, Y. Chang, M. Jia, and M. Zhu, "PCA-SDG based fault diagnosis on CAPL furnace temperature system," in *Chinese Control and Decision Conference (CCDC)*, 2013, pp. 3550–3554.

[LWJ04] Y. Liu, L. Wenyin, and C. Jiang, "A structural approach to recognizing incomplete graphic objects," in *International Conference on Document Analysis and Recognition (ICDAR)*, 2004, pp. 371–375.

[MB99] P. Mosterman and G. Biswas, "Diagnosis of continuous valued systems in transient operating regions," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 29, no. 6, pp. 554–565, 1999.

[MBS+11] H. Mersch, D. Behnen, D. Schmitz, U. Epple, C. Brecher, and M. Jarke, "Gemeinsamkeiten und Unterschiede der Prozess- und Fertigungstechnik," *Automatisierungstechnik*, vol. 59, no. 1, pp. 7–17, 2011.

[MG00] J. Montmain and S. Gentil, "Dynamic causal model diagnostic reasoning for online technical process supervision," *Automatica*, vol. 36, no. 8, 2000.

[MG03] N. Maran and R. Glavin, "Low- to high-fidelity simulation – a continuum of medical education?" *Medical Education*, vol. 37, pp. 22–28, 2003.

[Mor16] Morgan Stanley, "Insight: cloud control –the future of industrial automation," *Global Capital Goods*, vol. 47, 2016.

[MRV03] M. Maurya, R. Rengaswamy, and V. Venkatasubramanian, "A systematic framework for the development and analysis of signed digraphs for chemical processes. Control loops and flowsheet analysis," *Industrial and Engineering Chemistry Research*, vol. 42, no. 20, pp. 4811–4827, 2003.

[MRV04] M. Maurya, R. Rengaswamy, and V. Venkatasubramanian, "Application of signed digraphs-based analysis for fault diagnosis of chemical process flowsheets," *Engineering Applications of Artificial Intelligence*, vol. 17, no. 5, pp. 501–518, 2004.

[MSSdC05] L. Mendonsa, J. Sousa, and J. Sa da Costa, "Fault detection and isolation of industrial processes using optimized fuzzy models," in *IEEE International Conference on Fuzzy Systems (FUZZ)*, 2005, pp. 851–856.

[MWE+16] R. Mordinyi, D. Winkler, F. Ekaputra, M. Wimmer, and S. Biffl, "Investigating model slicing capabilities on integrated plant models with AutomationML," in *IEEE International Conference on Emerging Technology and Factory Automation (ETFA)*, 2016, p. Accepted for publication.

[MY94] T. McAvoy and N. Ye, "Base control for the Tennessee Eastman problem," *Computers & Chemical Engineering*, vol. 17, no. 5, pp. 383–413, 1994.

[NYK+05] S. Nomura, K. Yamanaka, O. Katai, H. Kawakami, and T. Shiose, "A novel adaptive morphological approach for degraded character image segmentation," *Pattern Recognition*, vol. 38, no. 11, pp. 1961–1975, 2005.

[OBWU15] M. Oppelt, M. Barth, G. Wolf, and L. Urbas, "Simulation im Lebenszyklus einer prozessanlage. teil 1: Ergibnisse einer globalen Umfrage und eine Roadmap," *atp-edition / Automatisierungstechnische Praxis*, vol. 9, 2015.

[ODLW13] M. Oppelt, O. Drumm, B. Lutz, and G. Wolf, "Approach for integrated simulation based on plant engineering data," in *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2013, pp. 1–4.

[OU14] M. Oppelt and L. Urbas, "Integrated virtual commissioning an essential activity in the automation engineering process: From virtual commissioning to simulation supported engineering," in *Annual Conference of the IEEE Industrial Electronics Society (IECON)*, 2014, pp. 2564–2570.

[OWU15] M. Oppelt, G. Wolf, and L. Urbas, "Towards an integrated use of simulation within the life-cycle of a process plant," in *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2015, pp. 1–8.

[Oxf14] Oxford Dictionaries, *The British and World English dictionary.* Oxford University Press, 2014.

[PAR14] D. Patle, Z. Ahmad, and G. Rangaiah, "Operator training simulators in the chemical industry: review, issues, and future directions," *Reviews in Chemical Engineering*, vol. 30, no. 2, pp. 199–216, 2014.

[PHK16] N. Paganus, K. Honkoila, and T. Karhela, "ntegrating dynamic process simulation into detailed automation engineering," in *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2016, pp. 1–8.

[PP12] L. Pronzato and A. Pázman, *Design of Experiments in Nonlinear Models: Asymptotic Normality, Optimality Criteria and Small-Sample Properties.* Springer New York Heidelberg Dordrechr London, 2012.

[PSF15a] P. Puntel-Schmidt and A. Fay, "Applying the domain-mapping-matrix to identify the appropriate level of detail of simulation models for virtual commissioning," in *IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control Schedule (CESCIT)*, 2015.

[PSF15b] P. Puntel-Schmidt and A. Fay, "Levels of detail and appropriate model types for virtual commissioning in manufacturing engineering," in *Conference on Mathematical Modelling*, 2015, pp. 17–20.

[RAM88] *Reliability, Availability and Maintainability Dictionary.* ASQC Quality Press, Milwaukee, 1988.

[Rav08] A. Ravi Ravindran, Ed., *Operations Research and Management Science Handbook.* CRC Press Taylor & Francis Group, 2008.

[RCHH16] V. Rodrigo, M. Chioua, T. Hagglund, and M. Hollender, "Causal analysis for alarm flood reduction," *IFAC Symposium on Dynamics and Control of Process*, vol. 49, no. 7, pp. 723–728, 2016.

[Ric95] L. Ricker, "Optimal steady-state operation of the Tennessee Eastman Challenge Process," *Computers & amp; Chemical Engineering*, vol. 19, no. 9, pp. 949–959, 1995.

[Rot09] D. Rothenberg, *Alarm Management for Process Control.* Momentum Press, 2009.

[RW93] J. Reifman and T. Wei, "System diagnostics using qualitative analysis and component functional classification," Patent US 5 265 035, November 23, 1993.

[RW95] J. Reifman and T. Wei, "Systematic construction of qualitative physics based rules for process diagnostics," *Progress in Artificial Intelligence, Lecture Notes in Computer Science*, vol. 990, pp. 311–322, 1995.

[Sch99] E. Schnieder, *Methoden der Automatisierung: Beschreibungsmittel, Modellkonzepte und Werkzeuge für Automatisierungssysteme.* Vieweg, 1999.

[SCLC02] J. Song, M. Cai, M. Lyu, and S. Cai, "Graphics recognition from binary images: one step or two steps," in *International Conference on Pattern Recognition*, vol. 3, 2002, pp. 135–138.

[SCTF13] M. Schleburg, L. Christiansen, N. F. Thornhill, and A. Fay, "A combined analysis of plant connectivity and alarm logs to reduce the number of alerts in an automation system," *Journal of Process Control*, vol. 23, no. 6, pp. 839–851, 2013.

[SDS08] M. Schleipen, R. Drath, and O. Sauer, "The system-independent data exchange format CAEX for supporting an automatic configuration of a production monitoring and control system," in *IEEE International Symposium on Industrial Electronics (ISIE)*, 2008, pp. 1786–1791.

[Sen13] S. Sengupta, *System Simulation and Modeling.* Pearson Education, 2013.

[SFJ15] S. Schröck, A. Fay, and T. Jäger, "Systematic interdisciplinary reuse within the engineering of automated plants," in *IEEE International Systems Conference (SysCon)*, 2015.

[SFP02] S. Simani, C. Fantuzi, and R. Patton, *Model-based Fault Diagnosis in Dynamic Systems Using Identification Techniques.* Springer, 2002.

[SFTF11] M. Strube, A. Fay, S. Truchat, and H. Figalist, "Modellgestützte Modernisierungsplanung," *atp-edition / Automatisierungstechnische Praxis*, no. 7-8, pp. 889–895, 2011.

[SGC+99] C. Schlenoff, M. Gruninger, T. Creek, M. Ciocoiu, and J. Lee, "The essence of the Process Specification Language," *Transactions of the Society for Computer Simulation*, vol. 16, pp. 204–216, 1999.

[SRFF11] M. Strube, S. Runde, H. Figalist, and A. Fay, "Risk minimization in modernization projects of plant automation: A knowledge-based approach by means of semantic web technologies," in *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2011, pp. 1–8.

[SSC+00] J. Song, F. Su, J. Chen, C. Tai, and S. Cai, "Line net global vectorization: an algorithm and its performance evaluation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2000, pp. 383–388.

[SSD+15] A. Schüller, A. Scholz, D. Drath, T. Tauchnitz, and T. Scherwietes, "Speed-Standardisierung am Beispiel der PLT-Stelle: Datenaustausch mit dem Namur-Datencontainer," *atp-edition / Automatisierungstechnische Praxis*, vol. 1-2, pp. 36–46, 2015.

[SSE09] S. Schmitz, M. Schluetter, and M. Epple, "Automation of Automation -Definition, components and challenges," in *IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, 2009, pp. 1–7.

[TB02] H. Talbot and R. Beare, Eds., *Mathematical Morphology: Proceedings of the International Symposium (ISMM)*, 2002.

[TBBT09] J. Thambirajah, L. Benabbas, M. Bauer, and N. F. Thornhill, "Cause-and-effect analysis in chemical processes utilizing xml, plant connectivity and quantitative process history," *Computers & Chemical Engineering*, vol. 33, no. 2, pp. 503 – 512, 2009.

[TBS99] R. Trigg, J. Blomberg, and L. Suchman, "Moving documents collections online: The evolution of a share repository," in *European Conference on Computer Supported Cooperative Work*, 1999.

[TCP03] N. F. Thornhill, J. Cox, and M. Paulonis, "Diagnosis of plant-wide oscillation through data-driven analysis and process understanding," *Control Engineering Practice*, vol. 11, pp. 1481–1490, 2003.

[THKN13] K. Takeda, T. Hamaguchi, N. Kimura, and M. Noda, "A method of designing plant alarm system based on first alarm alternative signals for each assummed plant malfunction," in *International Conference on Porcess Systems Engineering*, 2013.

[Tom98] K. Tombre, "Analysis of engineering drawings: State of the art and challenges," *Graphics Recognition: Algorithms and Systems*, vol. 1389, pp. 257–264, 1998.

[TSHV02] N. Thornhill, S. Shah, B. Huang, and A. Vishnubhotla, "Spectral principal component analysis of dynamic process data," *Control Engineering Practice*, vol. 10, pp. 833–846, 2002.

[Tub16] R. Tubb, "P&GJ's 2016 worldwide construction report," *Pipeline & Gas Journal*, vol. 243, no. 1, pp. 23–27, 2016.

[VHG+14] P. Vorst, M. Hollender, T. Goldschmidt, B. Schlich, C. Messinger, J. Schmitt, and M. Dix, "Cloud-based enterprise alarm management with analytic services," *ABB Annual Report*, pp. 30–35, 2014.

[VRK03] V. Venkatasubramanian, R. Rengaswamy, and S. Kavuri, "A review of process fault detection and diagnosis part ii: Qualitative model and search strategies," *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 313–326, 2003.

[VRYK03a] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. Kavuri, "A review of process fault detection and diagnosis part i: Quantitative model-based methods," *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 293–311, 2003.

[VRYK03b] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. Kavuri, "A review of process fault detection and diagnosis: Part iii: Process history based methods," *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 327–346, 2003.

[VT92a] P. Vaxiere and K. Tombre, "CAD conversion of mechanical drawings," *IEEE Computer*, 1992.

[Wad04] H. Wade, *Basic and Advanced Regulatory Control: System Design and Application*, 2nd ed. ISA–The Instrumentation, Systems, and Automation Society, 2004.

[Wat86] D. Waterman, *A guide to expert systems.*, 1st ed. Addison-Wesley, 1986.

[WD98] L. Wenyin and D. D, "Genericity in graphics recognition algorithms. Graphics recognition: Algorithms and systems," *Lecture notes in Computer Science*, 1998.

[Web08] K. Weber, *Dokumentation verfahrenstechnischer Anlagen: Praxisbuch mit Checklisten und Beispielen.* Springer, 2008.

[Wei13] M. Weisfeld, *The Object-Oriented Thought Process*, 4, Ed. Developer's Library, 2013.

[WKCB15] D. Wee, R. Kelly, J. Cattell, and M. Breunig, "Industry 4.0 how to navigate digitization of the manufacturing sector," McKinsey&Company, Tech. Rep., 2015.

[WYCS15] J. Wang, F. Yang, T. Chen, and S. Shah, "An overview of industrial alarm systems: Main causes for alarm overloading, research status, and open problems," *IEEE Transactions on Automation Science and Engineering*, no. 99, pp. 1–17, 2015.

[WZY07] L. Wenyin, W. Zhang, and L. Yan, "An interactive example-driven approach to graphics recognition in engineering drawings," *Internation Journal on Document Analysis and Recognition (IJDAR)*, 2007.

[XZB+04] X. Xiaogang, S. Zhengxing, P. Binbin, J. Xiangyu, and L. Wenyin, "An online composite graphics recognition approach based on matching of spatial relation graphs," *Internation Journal on Document Analysis and Recognition (IJDAR)*, vol. 7, pp. 44–55, 2004.

[YAB+06] S. Yim, H. Ananthakumar, L. Benabbas, A. Horch, R. Drath, and N. F. Thornhill, "Using process topology in plant-wide control loop performance assessment," *Computers & Chemical Engineering*, vol. 31, no. 2, pp. 86 – 99, 2006.

[Yan05] S. Yang, "Symbol recognition via statistical integration of pixel-level constraint histograms: A new descriptor new descriptor," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, pp. 278–281, 2005.

[YDH+12] S. Yin, S. Ding, A. Haghani, H. Hao, and P. Zhang, "A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark Tennessee Eastman Process," *Journal of Process Control*, vol. 22, no. 9, pp. 1567–1581, 2012.

[YDSC14] F. Yang, P. Duan, S. Shah, and T. Chen, *Capturing Connectivity and Causality in Complex Industrial Processes.* Springer Briefs in Applied Sciences and Technology, 2014.

[YSL10] R. Yang, F. Su, and T. Lu, "Research of the structural-learning-based symbol recognition mechanism for engineering drawings," in *International Conference on Digital Content, Multimedia Technology and its Applications (IDC)*, 2010, pp. 346–349.

[YSS97] Y. Yu, A. Samal, and S. Seth, "A system for recognizing a large class of engineering drawings," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 8, pp. 868–890, 1997.

[YSX12] F. Yang, S. Shah, and D. Xiao, "Signed directed graph based modeling and its validation from process knowledge and process data," *International Journal of Applied Math and Computational Science*, vol. 22, pp. 41–53, 2012.

[Yu95] B. Yu, "Automatic understanding of symbol-connected diagrams," in *International Conference on Document Analysis and Recognition (ICDAR)*, 1995, pp. 803–806.

[YW03] L. Yan and L. Wenyin, "Engineering drawings recognition using a case-based approach," in *International Conference on Document Analysis and Recognition (ICDAR)*, 2003, pp. 190–194.

[YWSG95] D. Yu, D. Williams, D. Shields, and J. Gomm, "A parity space method of fault detection for bilinear systems," in *American Control Conference*, vol. 2, 1995, pp. 1132–1133.

[YY15] W. Yu and F. Yang, "Detection of causality between process variables based on industrial alarm data using transfer entropy," *Entropy*, vol. 17, pp. 5868–5887, 2015.

[ZCWC05] J. Zhang, W. Cao, B. Wang, and N. Cui, "Fault location algorithm based on the qualitative and quantitative knowledge of signed directed graph," in *IEEE International Conference on Industrial Technology*, 2005, pp. 1231–1234.

[Zha08] Y. Zhang, "Fault detection and diagnosis of nonlinear processes using improved kernel independent component analysis (kica) and support vector machine (SVM)," *Industrial & Engineering Chemistry Research*, vol. 47, no. 18, pp. 6964–6971, 2008.

Publications of the author

[ACHF14] E. Arroyo, M. Chioua, M. Hoernicke, and A. Fay, "Integrating plant and process information as a basis for automated plant diagnosis tasks," in *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2014, pp. 1–8.

[AFCH15] E. Arroyo, A. Fay, M. Chioua, and M. Hoernicke, "Supporting plant disturbance analysis by dynamic causal digraphs and propagation look-up tables," in *IEEE International Conference on Intelligent Engineering Systems (INES)*, 2015, pp. 283–289.

[AFH16] E. Arroyo, A. Fay, and M. Hoernicke, "A method of digitalizing engineering documents," Patent Application EP3104302, 2016.

[AFHC16] E. Arroyo, A. Fay, M. Hoernicke, and M. Chioua, "Method for analysis of plant disturbance propagations," Patent Application EP3048613, 2016.

[AFHR15] E. Arroyo, A. Fay, M. Hoernicke, and P. Rodriguez, "Digitalisierung grafischer Engineering-Dokumente mit Hilfe optischer Erkennung und semantischer Analyse als Grundlage für die Modernisierung bestehender Anlagen," in *Kongress Automation*, 2015.

[AHF15] E. Arroyo, L. Hoang, and A. Fay, "Automatic extraction of structural and connectivity information from SVG-based engineering documents," in *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2015, pp. 1–8.

[AHFR16] E. Arroyo, M. Hoernicke, A. Fay, and P. Rodriguez, "Automatic derivation of qualitative simulation models from legacy piping and instrumentation diagrams," *Computers & Chemical Engineering*, vol. 92, pp. 112–132, 2016.

[ALL13] E. Arroyo, J. Lima, and P. Leitao, "Adaptive image pre-processing for quality control in production lines," in *IEEE International Conference on Industrial Technology (ICIT)*, 2013, pp. 1044–1050.

[ARF+16] E. Arroyo, R. Royston, A. Fay, M. Hoernicke, and P. Rodriguez, "From paper to digital," *ABB Review*, vol. 1, pp. 65–69, 2016.

[ASC+14] E. Arroyo, D. Schulze, L. Christiansen, A. Fay, and N. Thornhill, "Derivation of diagnostic models based on formalized process knowledge," in *World Congress of the International Federation of Automatic Control (IFAC)*, vol. 47, no. 3, 2014, pp. 3456–3464.

[HAF16] L. Hoang, E. Arroyo, and A. Fay, "Automatische Analyse und Erkennung graphischer Inhalte von SVG-basierten Engineering-Dokumenten," *at-Automatisierungstechnik*, vol. 64, no. 2, pp. 133–146, 2016.

[HMAF16] M. Hoernicke, C. Messinger, E. Arroyo, and A. Fay, "Topologiemodelle in AutomationML: Grundlage für die Automatisierung der Automatisierung," *atp-edition / Automatisierungstechnische Praxis*, vol. 58, no. 5, pp. 28–41, 2016.

[LFA+15] J. Ladiges, A. Fülber, E. Arroyo, A. Fay, and C. Haubeck, "Learning material flow models for manufacturing plants from data traces," in *IEEE International Conference on Industrial Informatics (INDIN)*, 2015, pp. 294–301.

[LHW+13] J. Ladiges, C. Haubeck, I. Wior, E. Arroyo, A. Fay, and W. Lamersdorf, "Evolution of production facilities and its impact on non-functional requirements," in *IEEE International Conference on Industrial Informatics (INDIN)*, 2013.

[RAF15] M. Riedel, E. Arroyo, and A. Fay, "Knowledge-based specification of principle solutions for sensors and actuators based on standardized plant description and semantic concepts," in *IEEE International Conference on Emerging Technologies Factory Automation (ETFA)*, 2015, pp. 1–8.

[WLAF13] I. Wior, J. Ladiges, E. Arroyo, and A. Fay, "First steps from a traffic node to traffic networks; modeling and stability," in *IEEE International Conference on Emerging Technologies Factory Automation (ETFA)*, 2013, pp. 1–4.

Standards, norms and guidelines

[DIN EN 61355, 2008] DIN EN 61355 (2008). Classification and designation of documents for plants, systems and equipment.

[DIN EN ISO 10628, 2001] DIN EN ISO 10628 (2001). Diagrams for the chemical and petrochemical industry.

[IAEA-TECDOC-1252, 2001] IAEA-TECDOC-1252 (2001). Information integration in control rooms and technical offices in nuclear power plants.

[IEC 61346, 2000] IEC 61346 (2000). Industrial systems, installations and equipment and industrial products.

[IEC 61360, 2002] IEC 61360 (2002). Standard data element types with associated classification scheme for electric components.

[IEC 61512, 1997] IEC 61512 (1997). Batch control -Part 1: Models and terminology.

[IEC 62023, 2011] IEC 62023 (2011). Structuring of technical information and documentation.

[IEC 62424, 2008] IEC 62424 (2008). Representation of process control engineering requests in P&I diagrams and data exchange between PI&D tools and PCE-CAE tools.

[IEC 62541, 2015] IEC 62541 (2015). OPC unified architecture.

[IEC 62714, 2014] IEC 62714 (2014). Engineering data exchange format for use in industrial automation systems engineering - Automation Markup Language (AutomationML) - Part 1: Architecture and general requirements.

[ISA 5.2, 1992] ISA 5.2 (1976 (R1992)). Binary logic diagrams for process operations publications.

[ISO 13584-42, 2010] ISO 13584-42 (2010). Industrial automation systems and integration - Parts library - Part 42: Methodology for structuring parts families.

[ISO 14617, 2005] ISO 14617 (2005). Graphical symbols for diagrams.

[ISO 15926-1, 2004] ISO 15926-1 (2004). Industrial automation systems and integration - Integration of life-cycle data for process plants including oil and gas production facilities - Part 1: Overview and fundamental principles.

[ISO 18.2, 2009] ISO 18.2 (2009). Management of alarm systems for the process industries.

[ISO 3511, 1984] ISO 3511 (1984). Process measurement control functions and instrumentation -symbolic representation.

[ISO 8879, 1986] ISO 8879 (1986). Information processing - text and office systems - Standard Generalized Markup Language (SGML).

[LANL, 2006] LANL (2006). Engineering Standards Manual ISD 341-2 -Apendix G: Control Logic Diagrams.

[NAMUR NA102, 2003] NAMUR NA102 (2003). Alarm Management. Namur Recommendation.

[NAMUR NA35, 2003] NAMUR NA35 (2003). Handling PCT Projects. Namur Recommendation.

[NAMUR NA96, 2017] NAMUR NA96 (2017). Process Diagnostics –Status Report. Namur Recommendation.

[NAMUR NE152, 2014] NAMUR NE152 (2014). Controller performance management: Monitoring and optimisation of regulatory control in production plants. Namur Recommendation.

[NORSOK I-005, 2005] NORSOK I-005 (2005). System Control Diagram.

[SAMA PMC 22-1, 1981] SAMA PMC 22-1 (1981). Functional diagramming of instruments and control systems.

[VDI/VDE 3682, 2014] VDI/VDE 3682 (2014). Formalized Process Description guideline.

[VDI/VDE 3693, 2015] VDI/VDE 3693 (2015). Virtual commissioning –Model types and glossary.

Internet sources and software

[Adobe, 2008] Adobe (2008). PDF Reference. v.1.7.

[Autodesk, 2016] Autodesk (2016). Autodesk P&ID. http://www.autodesk.com/products/autocad-p-id/overview. Last accessed on Jun. 17, 2016.

[AutomationML Consortium, 2012a] AutomationML Consortium (2012a). Whitepaper AutomationML Part 1: AutomationML Architecture.

[Aveva, 2016] Aveva (2016). AvevaP&ID. http://www.aveva.com/en/Products_and_Services/Product_Finder.aspx. Last accessed on Jun. 17, 2016.

[Dassault Systemes, 2016] Dassault Systemes (2016). Viewport. http://www.3ds.com/products-services/catia/products/dymola. Last accessed on Aug. 16, 2016.

[Fraunhofer, 2016] Fraunhofer (2016). CAEXMapping. http://www.iosb.fraunhofer.de/servlet/is/18823/. Last accessed on Jan. 26, 2017.

[Intergraph, 2016] Intergraph (2016). XMpLant PDS P&IDSmartPlant. http://www.intergraph.com/products/ppm/smartplant. Last accessed on Jun. 18, 2016.

[Mathworks, 2016] Mathworks (2016). Morphology fundamentals: Dilation and erosion. http://de.mathworks.com/help/images/morphological-dilation-and-erosion.html. Last accessed on Jul. 17, 2016.

[Modelica Association, 2016] Modelica Association (2016). Modelica and the Modelica Association. https://www.modelica.org. Last accessed on Aug. 14, 2016.

[National Instruments, 2015] National Instruments (2015). NI Vision Concepts. Manual No. 372916R- 01.

[Nextspace, 2015] Nextspace (2015). XMpLant PDS P&ID. http://www.nextspace.co.nz/products-and-services/solutions/collect. Last accessed on Sep. 12, 2015.

[NORSOK, 2016] NORSOK (2016). SCD Visio Toolbox. http://scd-diagram.no. Last accessed on Jul. 10, 2016.

[OPC Foundation, 2016] OPC Foundation (2016). OPC. https://opcfoundation.org. Last accessed on Aug. 19, 2016.

[OSMC, 2016] OSMC (2016). Open Modelica. https://www.openmodelica.org. Last accessed on Sep. 7, 2016.

[Pagin, 2016] Pagin, S. (2016). Raster vs. vector: The easy-to-understand guide. http://www.fastprint.co.uk/blog/raster-vs-vector-the-easy-to-understand-guide.html. Last accessed on Dec. 04, 2016.

[RadialSG, 2016] RadialSG (2016). Viewport. http://www.radialsg.com/viewport/viewport-engineering. Last accessed on Jul. 17, 2016.

[W3C, 2015a] W3C (2015a). Extensible Markup Language (XML). http://www.w3.org/XML/. Last accessed on Jul. 08, 2015.

[W3C, 2015b] W3C (2015b). Scalable Vector Graphics (SVG) 1.0 Specification - W3C Recommendation. http://www.w3.org/TR/SVG10. Last accessed on Jul. 10, 2015.

[W3C, 2016] W3C (2016). Document Object Model (DOM). http://www.w3.org/DOM. Last accessed on Jul. 10, 2016.

[WHATWG, 2016] WHATWG (2016). Document Object Model (DOM) -Living Standard. https://dom.spec.whatwg.org. Last accessed on Jul. 10, 2016.

[Wikipedia Foundation, 2017] Wikipedia Foundation (2017). Piping and instrumentation diagram. https://en.wikipedia.org/wiki/Piping_and_instrumentation_diagram. Last accessed on Jan. 28, 2017.

Supervised student works

[Bro15] K. Brown, "Analysis of fault propagation and alarm behavior by using dynamic causal digraphs and propagation look up tables. A Case Study of the Tennessee Eastman Process," Internship, Helmut Schmidt University, Institute of Automation Technology, 2015.

[Hoa15] X. L. Hoang, "Derivation of content from graphical documents based on the analysis of Scalable Vector Graphics," Master's thesis, Helmut Schmidt University, Institute of Automation Technology, 2015.

[Kie14a] C. F. Kiefer, "Development and implementation of a process mining agent for alarm management and plant asset management in process facilities," Master's thesis, Helmut Schmidt University, Institute of Automation Technology, 2014.

[Kie14b] C. F. Kiefer, "From alarm management to plant asset management," Seminar Work, Helmut Schmidt University, Institute of Automation Technology, 2014.

[Sch13] D. Schulze, "Development and implementation of a concept for an efficient diagnostic process based on process knowledge," Master's thesis, Helmut Schmidt University, Institute of Automation Technology, 2013.

About the author

Curriculum Vitae

Personal Information

Name	Esteban Arroyo Esquivel
Birth	16.01.1988 in Alajuela, Costa Rica

Professional Experience

01/2016 - Present	**Covestro AG**, Leverkusen Advanced Process Control Project Manager
10/2012 - 12/2015	**Helmut Schmidt University**, Hamburg Researcher Assistant
07/2011 - 04/2012	**Instituto Politécnico de Bragança**, Portugal Researcher Assistant
03/2010 - 09/2010	**Cabletica SA**, Costa Rica Electrical Engineer

Academic Formation

09/2010 - 07/2012	**Instituto Politécnico de Bragança**, Portugal Master of Science in Industrial Engineering
02/2006 - 02/2010	**Universidad de Costa Rica**, Costa Rica Honor Bachelor of Electrical Engineering
02/2001 - 12/2005	**Bilingual Exp. High School of Naranjo**, Costa Rica Secondary School
02/1995 - 12/2000	**Republic of Uruguay Primary School**, Costa Rica Primary School