 Technische Universität München

Department of Mechanical Engineering

Institute of Automation and Information Systems

Prof. Dr.-Ing. Birgit Vogel-Heuser

Bachelorarbeit

Dynamic Generation of Modular Industrial Plant Visualizations on a Manufacturing Execution System (MES) Interface



|  |  |
| --- | --- |
| Author: | Miguel Romero Karam |
| Advisor: | Dr.-Ing. Daniel Schütz |
| Supervisor: | Emanuel Trunzer, M.Sc. |
| Begin Date: | 15 April 2018 |
| Submission Date: | 15 Oktober 2018 |

Statutory Declaration

I hereby confirm to have written the present dissertation independently and only with the use of the sources and resources I have indicated. Both content and literal content were identified as such. The work has not been available in this or similar form to any other panel of examiners.



Date: Signature:

Abstract

…

Kurzzusammenfassung

…

Index

[Statutory Declaration III](#_Toc519946402)

[Abstract V](#_Toc519946403)

[Index VII](#_Toc519946404)

[1 Introduction 1](#_Toc519946405)

[1.1 Overview and Motivation 1](#_Toc519946406)

[1.2 Problem Definition 1](#_Toc519946407)

[1.3 Initial Situation 2](#_Toc519946408)

[1.4 Aim of the Bachelor Thesis 2](#_Toc519946409)

[1.5 Project Requirements 3](#_Toc519946410)

[1.6 Composition of the Bachelor Thesis 3](#_Toc519946413)

[1.6.1 GIST Project Management 3](#_Toc519946414)

[1.6.2 Agile Development Methodology 3](#_Toc519946415)

[2 Technological Standpoint 2](#_Toc519946416)

[2.1 Industrial Control in General 2](#_Toc519946417)

[2.1.1 Definitions 2](#_Toc519946418)

[2.1.2 Historical Industrial Context 2](#_Toc519946419)

[2.1.3 Current Trends 2](#_Toc519946420)

[2.1.4 Areas of Application 3](#_Toc519946421)

[2.2 Industrial Process Control 3](#_Toc519946422)

[2.2.1 Classification of Industrial Processes 3](#_Toc519946423)

[2.2.2 Process Control System (PCS) 3](#_Toc519946424)

[2.2.3 Plant Hierarchy Model (ISA-95) 4](#_Toc519946425)

[2.2.4 Process Visualizations 4](#_Toc519946426)

[2.2.5 Piping and Instrumentation Diagram (P&ID) 5](#_Toc519946427)

[2.2.5.1 Definition 5](#_Toc519946428)

[2.2.5.2 Functions 5](#_Toc519946429)

[2.2.5.3 Advantages 5](#_Toc519946430)

[2.2.5.4 Disadvantages 5](#_Toc519946431)

[2.3 Manufacturing Execution Systems 5](#_Toc519946432)

[2.3.1 Overview (Automatisierungspyramid -> decentral Network) 5](#_Toc519946433)

[2.3.2 Functions 5](#_Toc519946434)

[2.3.3 MES in Context of the 4th Industrial Revolution 5](#_Toc519946435)

[2.3.4 Overview of Legato Sapient® 5](#_Toc519946436)

[2.3.4.1 Design 5](#_Toc519946437)

[2.3.4.2 Features 6](#_Toc519946438)

[2.3.4.3 Software Architecture 6](#_Toc519946439)

[3 P&ID Shapes Library 8](#_Toc519946440)

[3.1 Overview 8](#_Toc519946441)

[3.2 Graph Theory 8](#_Toc519946442)

[3.2.1.1 What is a graph? 8](#_Toc519946443)

[3.2.1.2 Graph Visualizations 8](#_Toc519946444)

[3.2.1.3 Graph Interaction 8](#_Toc519946445)

[3.2.1.4 Graph Layouts 8](#_Toc519946446)

[3.2.1.5 Graph Analysis 8](#_Toc519946447)

[3.3 mxGraph API 8](#_Toc519946448)

[3.3.1 Core Architecture 8](#_Toc519946449)

[3.3.1.1 mxGraph Model 8](#_Toc519946450)

[3.3.1.2 Transaction Model 8](#_Toc519946451)

[3.3.1.3 mxCell 9](#_Toc519946452)

[3.3.1.4 Styles 9](#_Toc519946453)

[3.3.1.5 Geometry 9](#_Toc519946454)

[3.3.1.6 User Objects 9](#_Toc519946455)

[3.3.1.7 Cell Types 9](#_Toc519946456)

[3.3.1.8 Group Structure 9](#_Toc519946457)

[3.3.2 Technologies 9](#_Toc519946458)

[3.3.2.1 Deployment 9](#_Toc519946459)

[3.3.2.2 mxGraph Technologies 9](#_Toc519946460)

[3.3.2.3 9](#_Toc519946461)

[3.3.3 Schema 9](#_Toc519946462)

[4 Legato Sapient® Boardlet 12](#_Toc519946463)

[4.1 Overview of Software Architecture 12](#_Toc519946464)

[4.2 Design 12](#_Toc519946465)

[4.2.1 User Interface (UI) 12](#_Toc519946466)

[4.2.2 Presentation Logic 12](#_Toc519946467)

[4.3 Business Logic 13](#_Toc519946468)

[4.3.1 Overview 13](#_Toc519946469)

[4.3.2 Database Queries 13](#_Toc519946470)

[4.3.2.1 PostgreSQL Queries 13](#_Toc519946471)

[4.3.2.2 Get Data Generic Function via getRecords() 13](#_Toc519946472)

[4.3.2.3 Waiting for Asynchronous Requests to Complete 13](#_Toc519946473)

[4.3.3 Object Relational Data Mapping 13](#_Toc519946474)

[4.3.3.1 Nodes to Vertex Shapes (E, I, A, G) 13](#_Toc519946475)

[4.3.3.2 Connections to Edge Shapes (L) 14](#_Toc519946476)

[4.3.4 Graph Layout Algorithm 14](#_Toc519946477)

[SEE NEXT CHAPTER 14](#_Toc519946478)

[4.3.5 Generation of the XML File 14](#_Toc519946479)

[4.3.5.1 Structure of XML File 14](#_Toc519946480)

[4.3.5.2 Recursive Instantiation 15](#_Toc519946481)

[4.3.5.3 15](#_Toc519946482)

[5 P&ID Graphing Algorithm 17](#_Toc519946483)

[5.1 Build Hierarchy 17](#_Toc519946484)

[5.2 Hierarchy Traversal 17](#_Toc519946485)

[5.3 Simplification of Edges 17](#_Toc519946486)

[5.4 Vertex Placement 17](#_Toc519946487)

[5.4.1 Simplifying connections in model 18](#_Toc519946488)

[6 Testing, Verification and Validation 24](#_Toc519946489)

[6.1 Prototypical Implementation in an Industrial Context 24](#_Toc519946490)

[7 Synopsis XXVI](#_Toc519946491)

[List of Figures XXVIII](#_Toc519946492)

[List of Tables XXX](#_Toc519946493)

[Abbreviations 32](#_Toc519946494)

1. Introduction
   1. Overview and Motivation

The ProcAppCom (Process Application Composer) research project behind this bachelor thesis represents a cooperation between multiple industrial partners, namely 3S-Smart Software Solutions GmbH, Gefasoft GmbH, Johann Albrecht Brautechnik GmbH and APE Engineering GmbH with the Technical University of Munich. The main objective of the ProcAppCom research project is the automated configuration and generation of control code and visualizations for production plants in the field of process engineering.

Gefasoft GmbH is a leading and innovative provider of production-related software solutions. With the product Legato Sapient® Gefasoft offers a completely web-based production control system (MES) for the cross-plant evaluation of messages, alarms, process variables ​​and key figures, amongst other industry key functionalities.

Today, the development of control software and visualization interfaces for the operation of smaller process engineering systems is a major cost driver in process automation projects. Apart from the implementation: connection and configuration, costs rise rapidly during the MES product life cycle. Because of this, creating and later modifying plant-specific visualization interfaces represents a significant technical effort, which translates to continued increments in costs.

Motivation of this bachelor thesis is the development of system for the automated generation of dynamic Piping and Instrumentation Diagram (P&ID) visualizations for industrial plants with the goal of reducing implementation and operation expenditure of a MES, so that any enterprises can profit from these software solutions.

* 1. Problem Definition

The present trends in automation technology lead to a permanent increase in the complexity of industrial production facilities and to permanent technical changes. These changes propagate through the documentation, maintenance and operation of mentioned facilities, which represent a major engineering challenge. This leads to the frequent manual reconfiguration and adjustment of such systems during it’s life cycle. The creation and modification of plant-specific graphical user interfaces (GUI) contributes significantly to this.

* 1. Initial Situation

The foundations of this project had been laid by various other projects at Gefasoft in the context of the ProcAppCom research project. A general description model for process engineering plants was initially developed. Before the start of this project, it was also possible for plant models to be read and transcribed directly to database tables of the MES Legato Sapient® thus enabling factory edge gateways for the data-related connection of the control of systems to the MES to be generated dynamically. A system for the automated generation of the modelled process engineering plants represents the last part of this research project.

<INSERT GLOBAL ProcAppCom PROJECT ARCHITECTURE/CONCEPT SHOWING ALL PUZZLE PIECES, ASK DANIEL FOR PHOTO>

<INSERT INFO MAS ESPECIFICA QUE ES PARA PIDs ETC)

* 1. Goals of the Bachelor Thesis
* Reduce technical effort and accelerate development and modification of Piping and Instrumentation Diagrams (P&IDs) visualization generation of industrial processes.
* Modular design of visualization components according to the P&ID Standards for generating unopinionated visualization interfaces at the process control level for real time process monitoring.
* Prototypal implementation of the software solution in the MES Legato Sapient® in the form of an easy to use boardlet for the generation and update of visualizations.

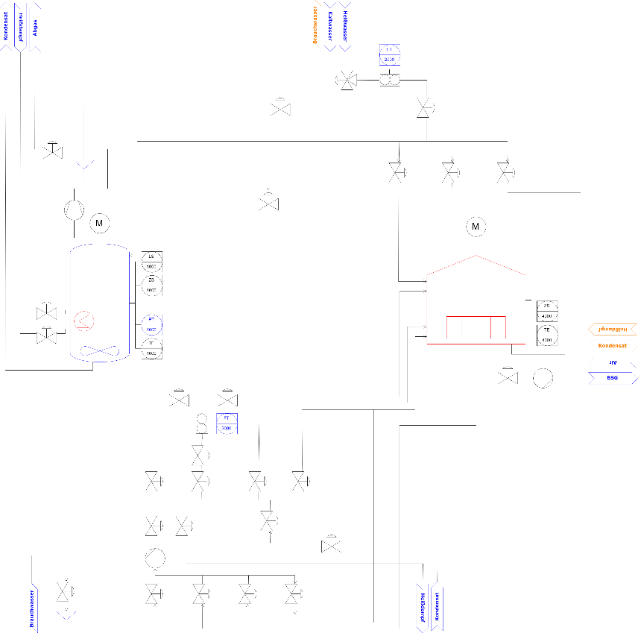


Figure 1 Manually generated preliminary goal of the P&ID Visualization of the Aida Brewery.

* 1. Project Requirements

As part of this bachelor thesis, specific requirements where set for the technical and conceptual aspects of the project. It was intended that all requirements where met by the end of the project. Although realizations during the project’s development cycle brought slight variations to some requirements, they remained for the most part same throughout the project.

* + 1. Technical Requirements

The technical requirements define the intended outcome of the practical implementation of the project at Gefasoft® Gmbh.

TR1: Library of modular P&ID Visualization Components According to Industry Standards

* Object oriented abstraction of industrial process engineering elements into P&ID classes and categories and representation of data model as UML 2.0 class diagram.
* Conception of a library of modular and composable P&ID visualization components in SVG format according to the current industry standards.
* Static geometrical definition of visualization components in a P&ID shapes library implemented as a JSON file for ease of personalization and maintainability.
* Automatic propagation of inherited and composed styles throughout library for ease of user personalization.
* Shapes should display real-time plant process values.

TR2: User Friendly Graphical User Interface (GUI) Boardlet for Creation of P&ID Visualizations

* User friendly boardlet with a simple interface for generating new or updating previously generated P&ID visualizations.
* No configurations needed and abstraction of inner workings for the user.
* File input for the selection of the desired version of the P&ID shapes library.
* Buttons for generating the XML of the visualization, for downloading the visualization in both JSON and XML format, and to upload the XML visualization file to Sapient Engine® file system for the Legato Graphic Designer® boardlet to import and render.
* Visual feedback: progress bar for script run and viewer for the generated XML as text.
* Dashboard with boardlets: P&ID Creator, Node Tree Selector, P&ID Viewer.

TR3: Client-Side Script for the Automated P&ID Visualization Generation as XML Files

* Required from user is just the selection of the desired P&ID shapes library version for the visualization and no additional configurations.
* Script encapsulates all required business logic in a single modular and composable, well-documented JavaScript file.
* Separation of primary concerns: presentation logic, database queries, data mappings, graphing algorithm and xml generation are all separate and inter-independent from each other.

TR4: Prototypal Implementation in the Infrastructure of a MES (Legato Sapient®) and Documentation

* Component-driven, moduar design of boardlet implemented with the Ember.js framework used in Legato Sapient®.
* Evaluation of the system: functionality, …
* Clear documentation of code.
* Document with next steps in case of interest on continuing development.
  + 1. Conceptual Requirements

The conceptual requirements define the core concepts which the proposed technical solution addresses and on which the functionality is based. Furthermore, the subject of these concepts represents the research part of the project and the development of new methodologies to achieve the project goals outlined IN SECTION 1.4.

CR1: Object-oriented library of statically defined visualization components

* Geometrical definition of shapes as JSON objects with a defined set of properties to take in a predefined value, or a default value if left blank.
* Shape instances inherit object properties from their parent P&ID class and category based on the data model defined as a UML 2.0 class diagram.
* Semi-colon separated string of styles destructured into styles object for targeted configuration of individual styles. Styles object then concatenated back to string on XML generation.
* Library implemented as a single JSON File for ease of translation to table format for editing of predefined values.

CR2: Mapping of Plant Instances to Corresponding Visualized Component

* Mapping to work with minimal changes to the original data model for an unobtrusive implementation of the automated P&ID visualization generator and to avoid the need for new tables and fields in database.
* Vertices will at least require a *shapeName* property to be set in the model and thus in the database.
* Connections don’t require changes in model. The shapeName attribute for each line shape is determined via logic.

CR3: Automatic Type Detection and Simplification of Connections

* Logic for setting the corresponding *shapeName* property to all connections: differentiate between data, process, connection and signal lines. Because of this, no need to specify a *shapeName* for connections in the data model.
* Connections defined in plant model, and thus also in database, in a logical instance to instance way, but suboptimal manner for the application of P&ID line shapes.
* Connections with multiple waypoints simplified by skipping intermediate ports, until a shape to shape connection (from start source to end target) is reached.
* Orthogonal line shapes optimized for minimal crossings and shortest routing between source and target.

CR4: Declarative specification of Graphing Constraints in Form of Tags

* Declarative approach of tags which allow targeting specific shapes to be positioned according to specific set of positioning rules.
* Tags are loosely coupled so they don’t intervene with the algorithm, rather define the algorithm to be run.
* Separates vertex placement logic for shapes to be positioned with distinct positioning rules.
* Vertex placement algorithm can be easily progressively enhanced through the addition of more and more tags.

CR4: P&ID Graphing Algorithm

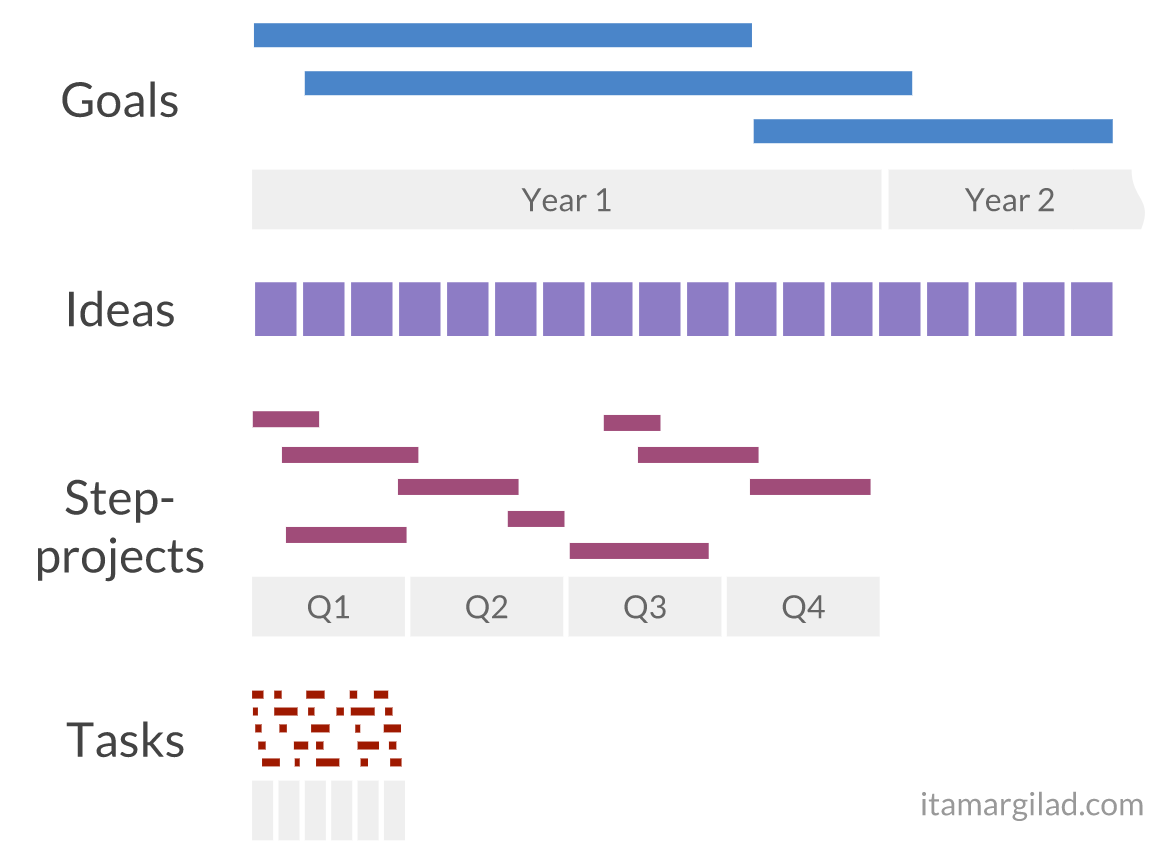
* Research and analysis of state of the art graphing algorithms.
* Simplicity over efficiency of the algorithm as to allow later improvements and since the creation of P&ID visualizations is not time critical.
* Depth-first post-order instance hierarchy traversal to get nodes in graphing order.
* Block-packing algorithm for the positioning of groups in groups to minimize the area.
* Algorithm concept for P&ID visualizations works no matter the complexity of the modelled process engineering plant.
* Ability of progressively enhancing the algorithm for creation of better and more complex visualizations without change in concept.
* Implementation of the algorithm for the example Aida Brewery plant.

CR5: Dynamic Real-Time Display of Process Variables in the P&ID Visualization

* P&ID visualization with real-time updating shapes and shape labels.
* Different types of display of process values depending on data type of process variable and on shape category (for example: boolean values set fill color for valves, but not for tanks).
* Components encapsulate a uniform set of data bindings to the actual process values and display values in real time.
* Default settings to override labels for shapes with data bindings to empty values.
* One-way data bindings that update automatically on the client-side instead of on the server-side for optimizing performance.
* Data bindings implemented with the sapient-bind property of the shape’s XML user-defined object which uses the mxGraph API already (placeholders).
  1. Composition of the Bachelor Thesis
     1. GIST Project Management

[1] https://www.linkedin.com/pulse/why-i-stopped-using-product-roadmaps-switched-itamar-gilad/

In favor of lightweight and flexible project management, the GIST methodology was preferred over more traditional agile methods. GIST is called after its main building blocks: Goals, Ideas, Sprints and Tasks, each with distinct planning perspective and frequency of change [1]. Instead of initially declaring tasks, goals where set, which enabled a plan to be defined in terms of the desired project outcomes. The goals stated ABOVE/IN THE PREVIOUS SECTION 1.4 lead the decisions from beginning to end of the project and where maintained for the most part. Ideas where tracked during the entire project’s life cycled and reconsidered for implementation or discarded at the beginning of each sprint. Sprints where executed until all tasks where completed, though tasks of previous and future sprints where sometimes worked upon outside of the corresponding sprint. Tasks themselves where reconsidered weekly for the current sprint and tracked with a KANBAN board.



Sprints

Figure 2 GIST methodology life cycles. From itamargilad.com

* + 1. Project SprintS TIMELINE

S1: Research and Choice of Tools and Technologies

Before any work was done, the tools and technologies with which the goals would be achieved had to be at least preliminary decided. Though many diagramming frameworks and libraries exist, not all where optimal for the task at hand, therefore comparisons where done between the available technologies after meticulous analysis of the projects technical and conceptual requirements. The open-source mxGraph JavaScript diagramming library was chosen due to its lightweightedness, robustness between distinct web browsers and compatibility with the diagramming tool draw.io, built with mxGraph. Furthermore, the Legato Graphic Designer Boardlet in which the visualization is to be view is implemented with the mxGraph library.

S2: Creation of a P&ID Shapes Library

The second project sprint was the conception of an object-oriented library of modular shapes conforming to the industry standards for P&ID visualizations. This task was further divided into subtasks. first of which was the analysis of the mxGraph application programming interface (API), with which the visualizations were to be implemented in the browser. These consisted of the manual creation and analysis of example visualizations as well as a thorough reading to the API’s documentation. mxGraph is a fully client-side JavaScript diagramming library that uses SVG and HTML for rendering. The predefined process engineering shape library was used as a base for the next step: the creation of a statically defined, modular and composable object-oriented shapes library for P&ID elements. It was decided that this library was to be defined in JSON format, to facilitate user modification and tuning of the geometries, rather than in XML format like the visualization file itself.

S3: Requirements and Design of Software Architecture

After the creation of the statically defined P&ID shapes library file in JSON format came the conceptual elaboration of a preliminary software architecture for the project’s technical implementation in the MES Legato Sapient®. This task preceded the commencement of the agile development life cycle.

S4: Boardlet Design

Aligning to the Legato Sapient® design and coding principles implemented in the component based Ember.js framework, the start of the development phase consisted in setting up boilerplate code for the P&ID Creator Boardlet. After the creation of both a JavaScript (.js) and handlebars (.hbs) templating file, the preliminary wireframe design for the boardlet was made and coded. Attaining to principles of component- based design, the handlebars template was designed and developed modularly with both new and reused ember components. After having the boardlet up and running on the Sapient Engine® it became possible to start the progressive development of the business logic for the automated P&ID visualization generation.

S5: Generation of the XML file of the P&ID Visualization

The first development sprint for the generation of the XML file of the visualization where made before establishing a database connection for fetching of the plant instances on a separate testing boardlet. This testing boardlet allowed for constant modification and experimentation of the algorithms with pure JavaScript, HTML5 and CSS. These allowed for rapid coding without needing to be connected to the full sapient architecture. The plant hierarchy was first modelled statically in form of JSON files in place of the database queries which return equivalent JSON responses. The file input component for the uploading of the P&ID shapes library was recycled to directly load the needed files in the client, thus enabling faster trials and testing of the script in development until XML of unplaced, overlapping vertices was correctly generated.

S6: Connection and Fetching of Plant Instances from Database

After the script successfully and automatically generated an XML file of the P&ID from static JSON files of plant instances, the connection to the database and registering of database tables in the Legato Configuration Center® (LC2) followed. This allowed to test the XML generation script now with actual plant data queried from the database. This required a global data map of all required tables and fields. With the data map, name mappings where done and a function to fetch the data with custom filters implemented via the available Legato getRecords() function. Modification of the database queries could now be done only by modification of the passed parameters for the query. The result of the XML generation algorithm with the actual plant data corresponded now with what was originally modelled for the plant. By now, all vertices and edges where correctly instantiated in the diagram, but vertices where placed one on top of the other. This lead to the start of the graphing algorithm for the placement of these vertices.

S7: P&ID Graphing Algorithm

The main task during this sprint was the development of a vertex placement algorithm to set the x and y properties of each vertex according to a defined set of positioning rules. Both a declarative, rule-based approach and an algorithmic approach for the optimization of area where used. First part of the algorithm consisted in the declaration of constraints in the form of tags based on shape attributes. This way, the positioning logic could be later better targeted at the distinct tags individually, since distinct shapes are to be positioned based on a distinct set of rules. The loosely-coupled tags where specified first and apart from the positioning logic, as this part was based on algorithmic optimization rather than classification. Afterwards, distinct sets of positioning rules where defined for each of the tags. The shapes would thus be iteratively positioned by the algorithm in a distinct and independent way. Furthermore, a block-packing algorithm was to be implemented for the positioning of groups in groups to minimize the area. Though much progress was made in a short time, a standpoint was later reached. Though the algorithm could still be made better, the time invested was too much compared to the progress, and because of the lack of time, the algorithm was left as is in order to continue with the last sprint.

S8: Dynamic Real-Time Display of Process Variables in the P&ID Visualization

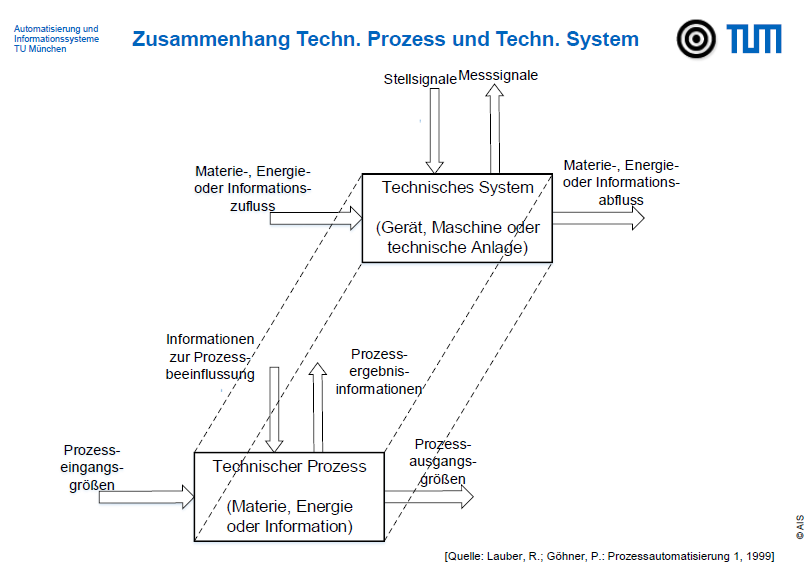
Although the output of the P&ID Creator boardlet and the input for the P&ID Viewer is a static XML file of the P&ID Visualization, it must contain the data bindings for the dynamic display of the process variables. The data binding should be independently set based on the data type of the process variable via the sapient-bind attribute in each XML object. This logic had to be set before the string generation so that the distinct data types of the process values result in distinct labels or colors for each shape instance. The Legato Graphic Designer requires the ID (primary key) of the value in the database and automatically fetches the value in the background whenever it changes. These functionality is implemented as a mxGraph placeholders and allows for the data bindings to be also included in the static XML file of the P&ID visualization.

S9: Prototypal Implementation and Evaluation in the Infrastructure of MES Legato Sapient®

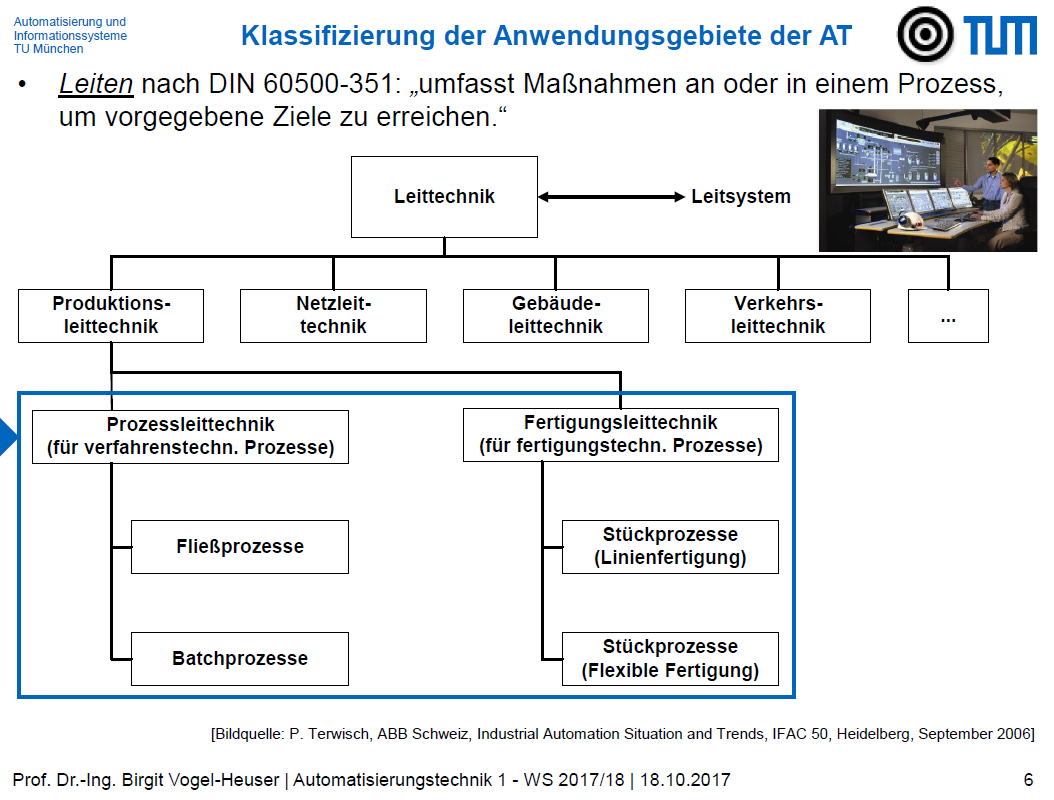
Although plant instances where already modelled and available from the database during the project, no connection to the gateway of the example Aida Brewery plant existed. This

1. Technological Standpoint
   1. Industrial Control in General
      1. Definitions

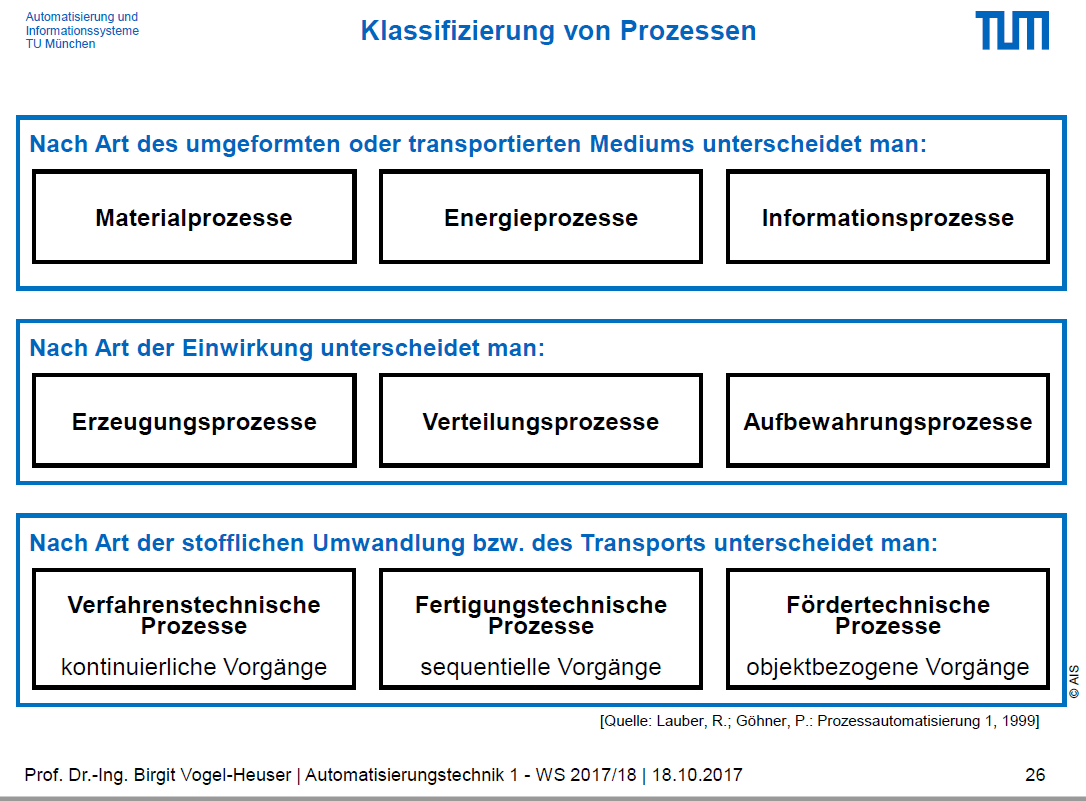
* Model
* Technical Process
* Tech. System AT23,
* Relationship Diagram AT24



* + 1. Historical Industrial Context
    2. Current Trends
    3. Areas of Application

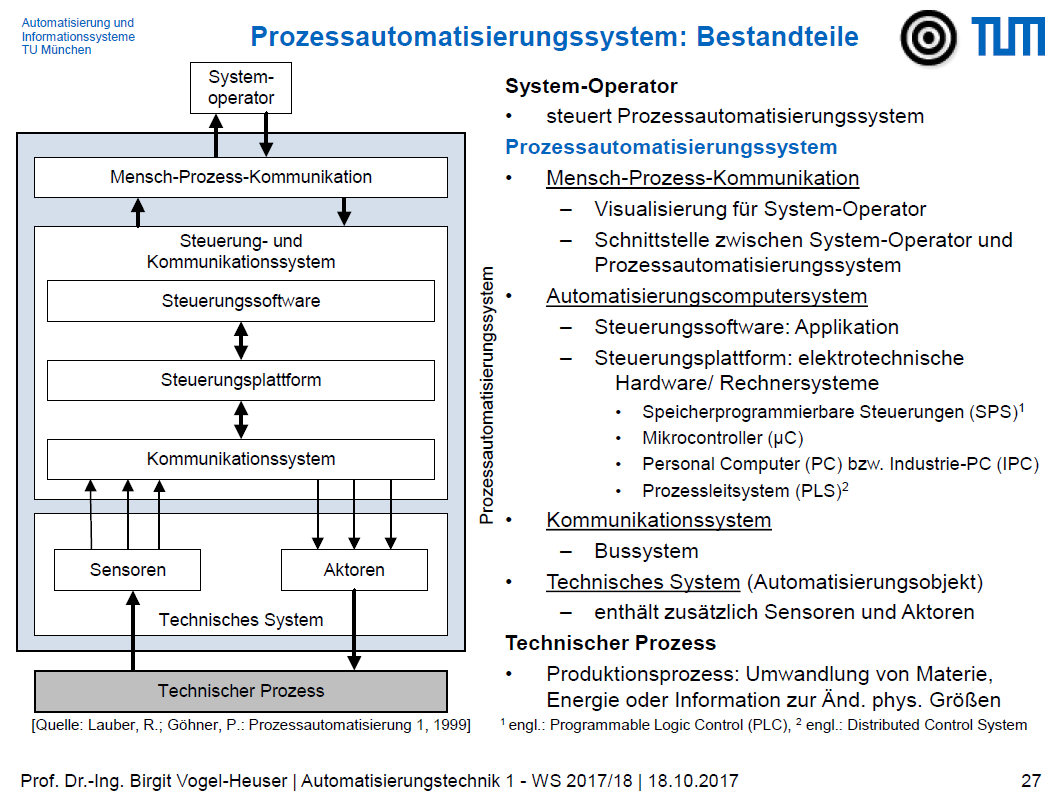


* 1. Industrial Process Control
     1. Classification of Industrial Processes



* + 1. Process Control System (PCS)

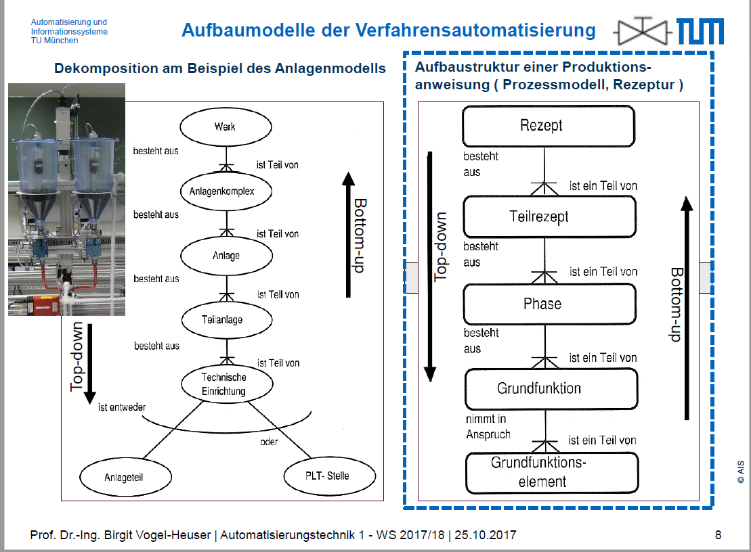
Definiton



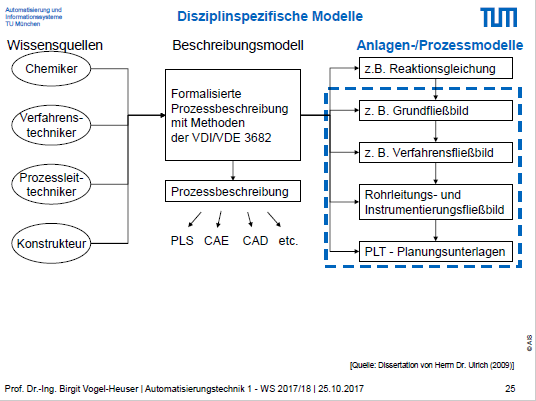
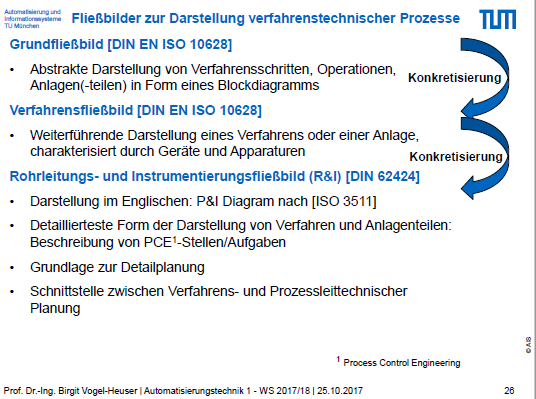
* + 1. Plant Hierarchy Model (ISA-95)

[ISA-95](https://en.wikipedia.org/wiki/ISA-95) as it is more commonly referred, is an international standard for developing an automated interface between enterprise and control systems.

(<https://en.wikipedia.org/wiki/Enterprise_control#ISA95_.E2.80.9Clevels.E2.80.9D_for_enterprise_integration>)



* + 1. Process Visualizations

* + 1. Piping and Instrumentation Diagram (P&ID)
* Definition
* Functions
* Advantages
* Disadvantages
  1. Manufacturing Execution Systems
     1. Overview (Automatisierungspyramid -> decentral Network)
     2. Functions
     3. MES in Context of the 4th Industrial Revolution
     4. Overview of Legato Sapient®

Entirely Web-based architecture and modular and customizable to the core to keep of with requirements.

* Design

Component based, modular design of dashboards (easy creation by adding boardlets)

Dashboard > Boardlet > Ember Components

* Features
* Software Architecture
  1. Related Works
     1. Overview of Related Works

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Work** |  | **CR1** | **CR6** | **CR7** |
| W1 | Prat, 2017 |  |  |  |
| W2 | Oppelt, Wolf, Drumm, Lutz, 2014 |  |  |  |
| W3 | Yang, Prasad, Xie, 2013 |  |  |  |
| W4 |  |  |  |  |
| W5 |  |  |  |  |
| W6 |  |  |  |  |
| W7 | Romero Karam, 2018 |  |  |  |
|  |  |  |  |  |

Figure 3 Overview of Related Works

|  |  |
| --- | --- |
| **Key** | Title |
| W1 | An Automated Generation Approach of Simulation Models for Checking Control/Monitoring System [1] |
| W2 | Automatic Model Generation for Virtual Commissioning based on Plant Engineering Data [2] |
| W3 | A Grey-Box Approach for Automated GUI-Model Generation of Mobile Applications [3] |
| W4 |  |
| W5 |  |
| W6 |  |
|  |  |

* + 1. Comparison of Related Works

The conceptual requirements to be addressed for this project compare as follows to the previously listed set of related works:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Work** | **R1** | **R2** | **R3** | **R4** | **R5** | **R6** | **R7** |
| W1 | ✓ |  |  |  |  |  |  |
| W2 | ✗ |  |  |  |  |  |  |
| W3 | ○ |  |  |  |  |  |  |
| W4 |  |  |  |  |  |  |  |
| W5 |  |  |  |  |  |  |  |
| W6 |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

Figure 4 Related works summary based on the adressed conceptual requirements.

✓✓ – Addressed and implemented

✓ – Addressed and partially implemented

○ – Not addressed

✗ – Not implemented

|  |  |
| --- | --- |
| CR1 | Library of modular P&ID Visualization Components According to Industry Standards |
| CR2 | User Friendly Graphical User Interface (GUI) Boardlet for Creation of P&ID Visualizations |
| CR3 | Client-Side Script for the Automated P&ID Visualization Generation as XML Files |
| CR4 | Graphing Algorithm Tailored for P&ID Visualizations |
| CR5 | Display Real-Time Process Values for Plant Monitoring |
| CR6 | Prototypal Implementation in the Infrastructure of a MES and Documentation |

1. P&ID Shapes Library
   1. Overview
   2. Graph Theory

* What is a graph?
* Graph Visualizations
* Graph Interaction
* Graph Layouts
* Graph Analysis
  1. mxGraph API

<INSERT Overview FROM 1.1 and Basic Licenscing Info FROM 1.5 and javascript client side working principles FROM 2.2 FROM mxGraph Javascript User Manual>

* + 1. Core Architecture

<INSERT mxGraph API Diagrams globales (class diagram)>

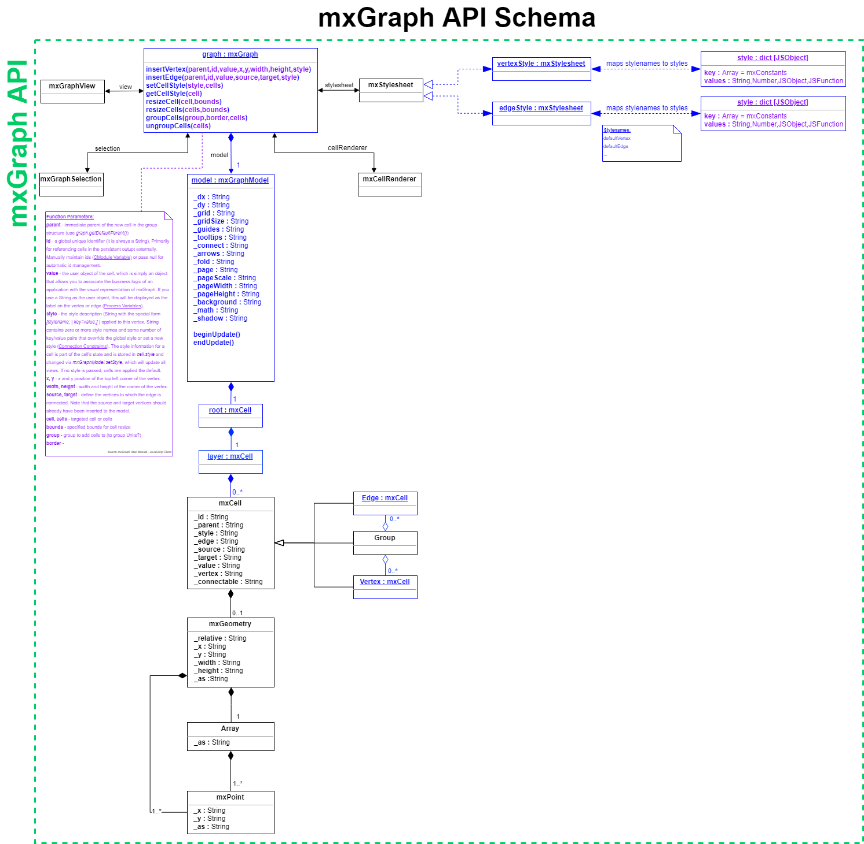
* mxGraph Model
* Transaction Model
* mxCell
* Styles
* Geometry
* User Objects
* Cell Types
* Group Structure
  + 1. Technologies
* Deployment

<INSERT HIGHLIGHTS FROM 1.3 mxGraph Javascript User Manual>

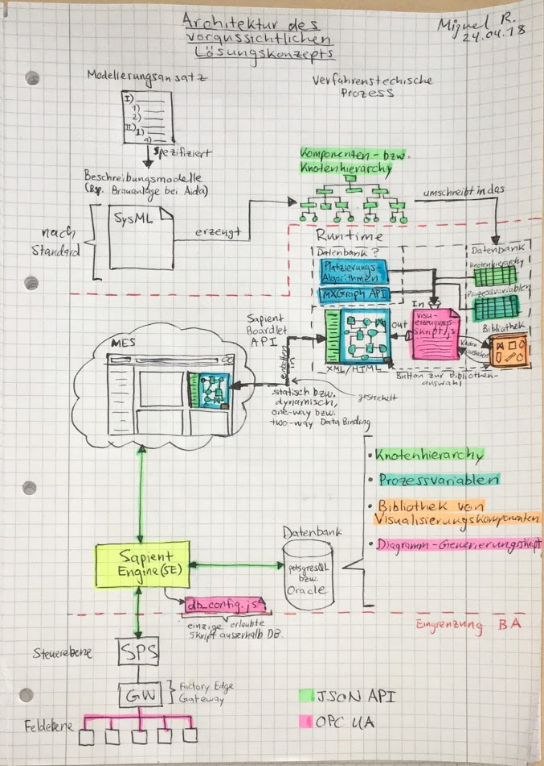
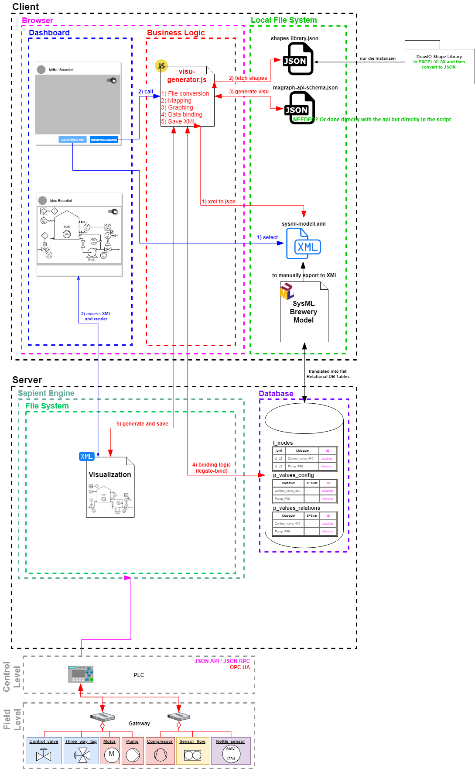
* mxGraph Technologies

<INSERT HIGHLIGHTS FROM 1.4 mxGraph Javascript User Manual>

* + 1. Schema



1. Legato Sapient® Boardlet
   1. Overview of Software Architecture



Agregar datenbank y SE y conectar modelierungsansatz y knotenbaum y reestrucRequirements

* 1. Design

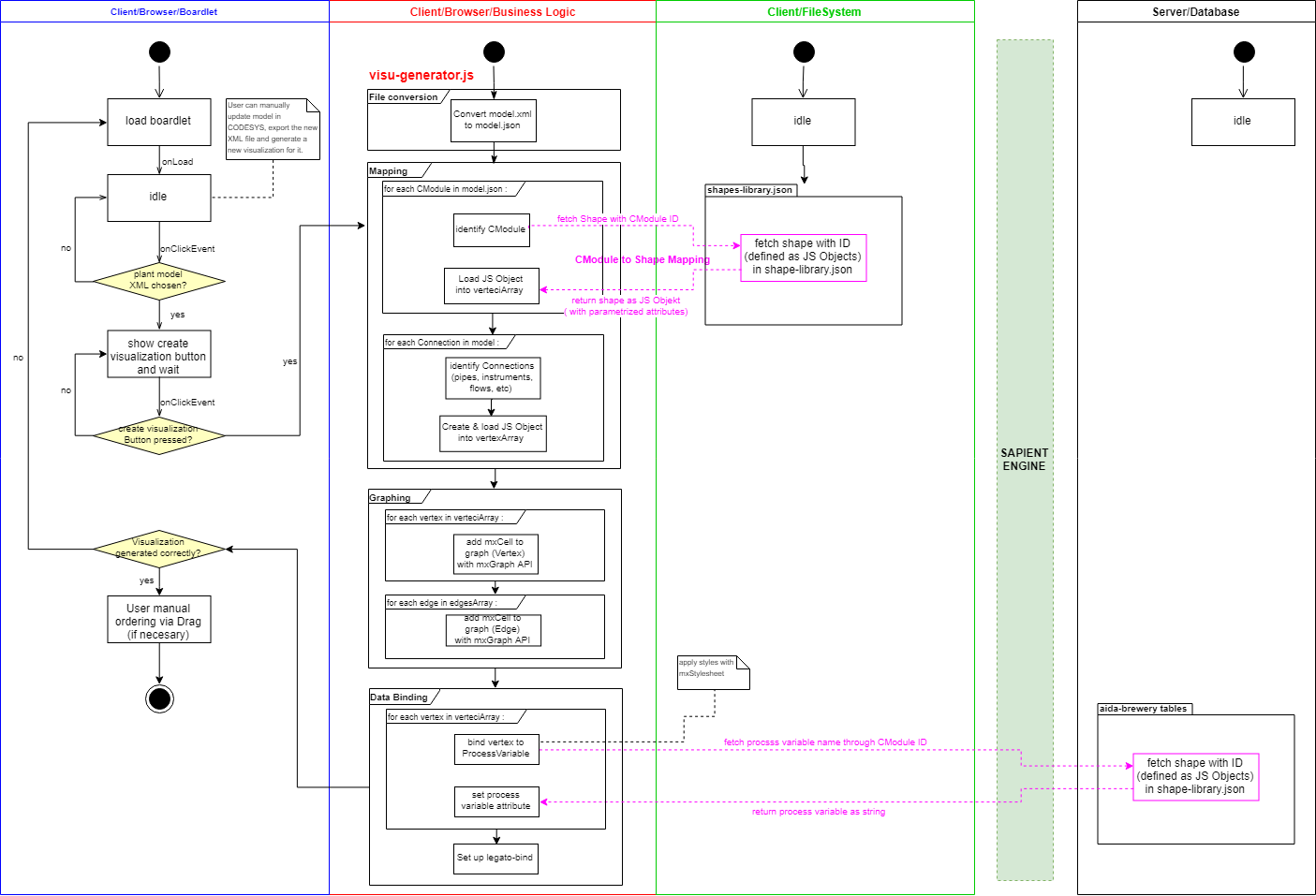
<INSERT PHOT OF P&ID VISUALIZER DASHBOARD WITH ALL BOARDLETS, SQUARE ON THE P&ID CREATOR BOARDLET AND SQUARE ON INDIVIDUAL EMBER COMPONENTS? OR TOO UNUBERSICHTLICH >

Modular Component-based solution. All included, up and ready boardlet.

* + 1. User Interface (UI)
    2. Presentation Logic
  1. Business Logic

<INSERT Table with ALL FUNCTIONS AND VARIABLES LIKE API DOCUMENTATION>

* + 1. Overview



* + 1. Database Queries
* PostgreSQL Queries
* Get Data Generic Function via getRecords()
* Waiting for Asynchronous Requests to Complete

Asynchronous

* + 1. Object Relational Data Mapping
* Nodes to Vertex Shapes (E, I, A, G)
* Connections to Edge Shapes (L)

Only process\_flows are modelled in model, so business logic to determine the line shapes accordingly.

ROBUST SET OF RULES

P&ID Line Shapes:

P - pipe\_line

C – connection\_line

S – signal\_line

D – data\_line

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Source\Target** | **Equipment** | **Instrument** | **Group** | **Arrow** | **Line** |
| **Equipment** | P | P (\*) | P (\*\*) |  | - |
| **Instrument** | P (\*) | D | D (\*\*) |  | (\*) |
| **Group** | P (\*\*) | D (\*\*) | P (\*\*\*) |  | - |
| **Arrow** |  |  |  |  |  |
| **Line** | - | (\*) | - |  | - |

Table 1[QUITAR POR QUE EL LINE TIPE EN REALIDAD SE DEBE ESPECIFICAR EN EL MODEL, NO TENGO QUE HABLAR DE ESTE FALLBACK POR QUE YA VA MAS ALLA DE MI BA]

Special Cases:

\* if Equipment to Instrument to Equipment (Instrument between 2 equipment, short circuit Equipment to Equipment with one single pipe\_line and connect instrument to that pipe\_line with a connection\_line.

\*\* if group to anything or anything to group, connect to group border, but if outermost group, then create a new arrow and connect to this arrow (attention to arrow direction).

\*\*\* use ports so that lines are continuous and don’t appear to break on group borders

* + 1. Graph Layout Algorithm

SEE NEXT CHAPTER

* + 1. Generation of the XML File
* Structure of XML File

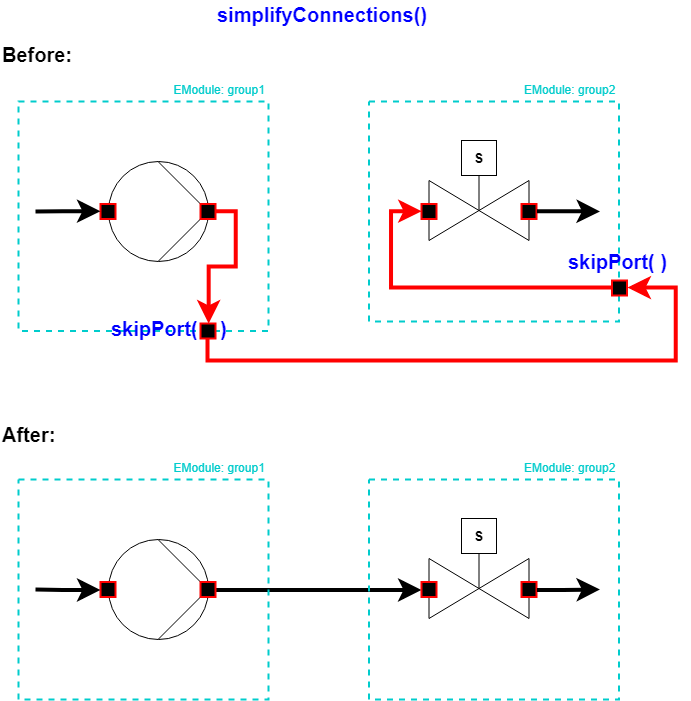
<INSERT CLASS DIAGRAM WITH VERERBUNGEN HIER THAT MODELLS THE STRUCTURE (FOR EXAMPLE MXGRAPH <|-----MXMODEL…

* Recursive Instantiation

1. P&ID Graphing Algorithm
   1. Build Hierarchy
   2. Hierarchy Traversal

Pathfinder in form of post-order depth-first search to find ordered path of node visited while traversing hierarchy.

* 1. Simplification of Edges



* 1. Vertex Placement

Algorithm can be PROGRESSIVELY ENHANCED: allows for incremental

* Overview
* Settings implemented as parameters allow for fine tuning of the algorithm.
* Specification of constraints as tags (loosely coupled to positioning logic)
* Vertex positioning based on constraints/rules
  + 1. Simplifying connections in model

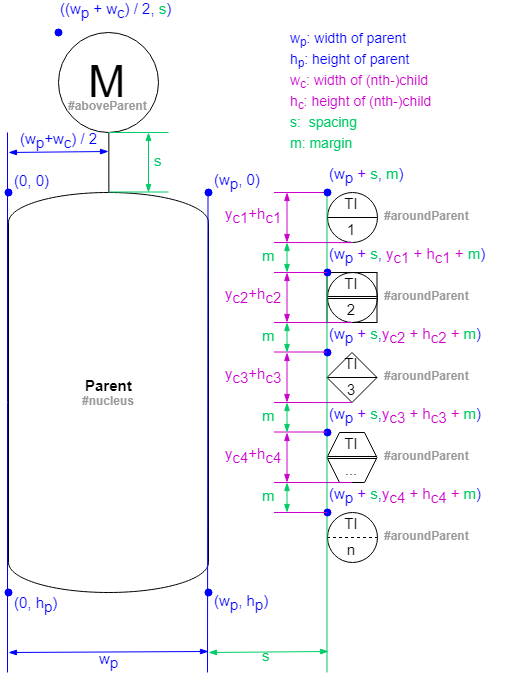


Figure 5 [INSERT CAPTION HERE]

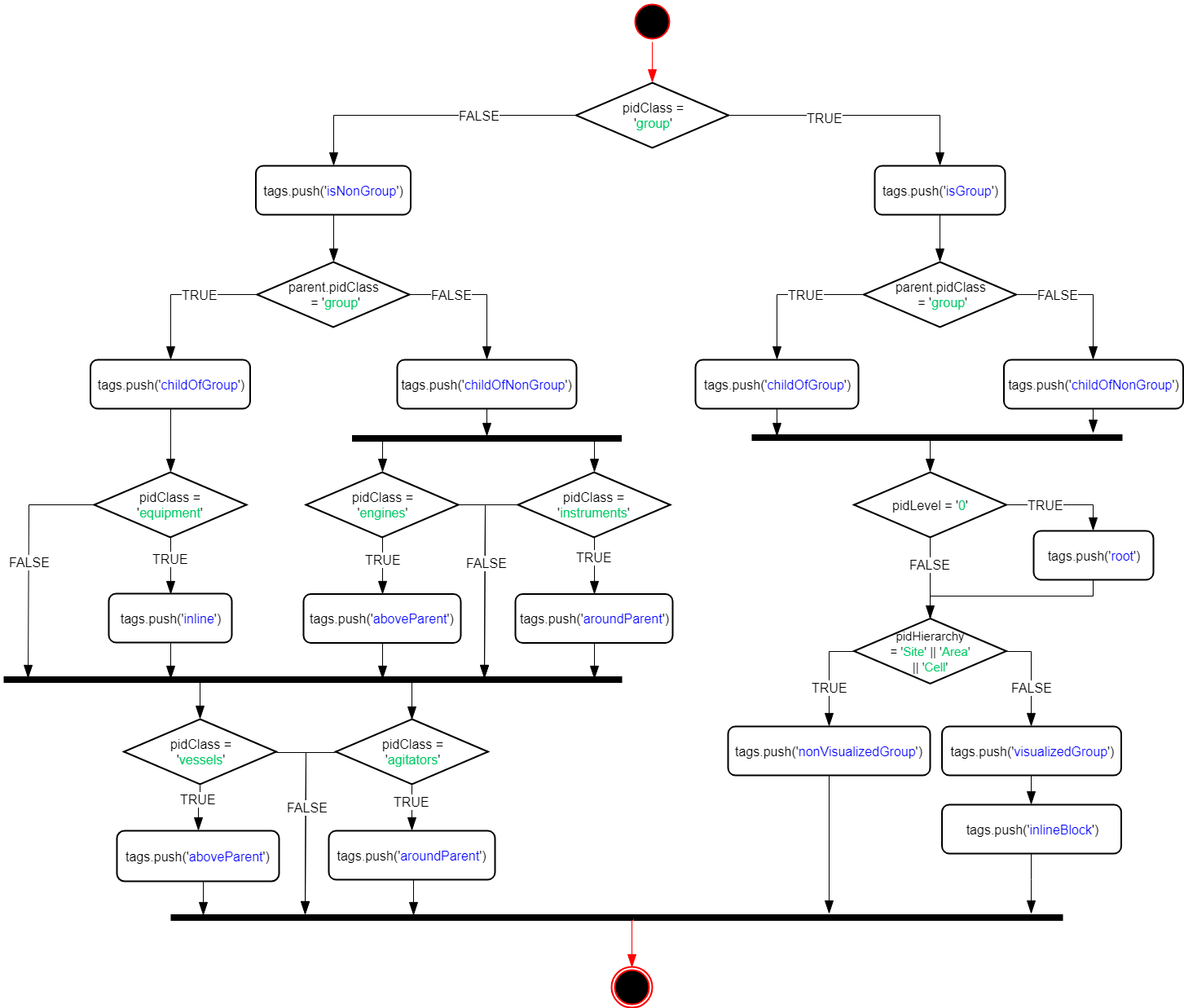


Figure 6 [ACTIVITY DIAGRAM OF SPECIFICATION OF CONSTRAINTS]

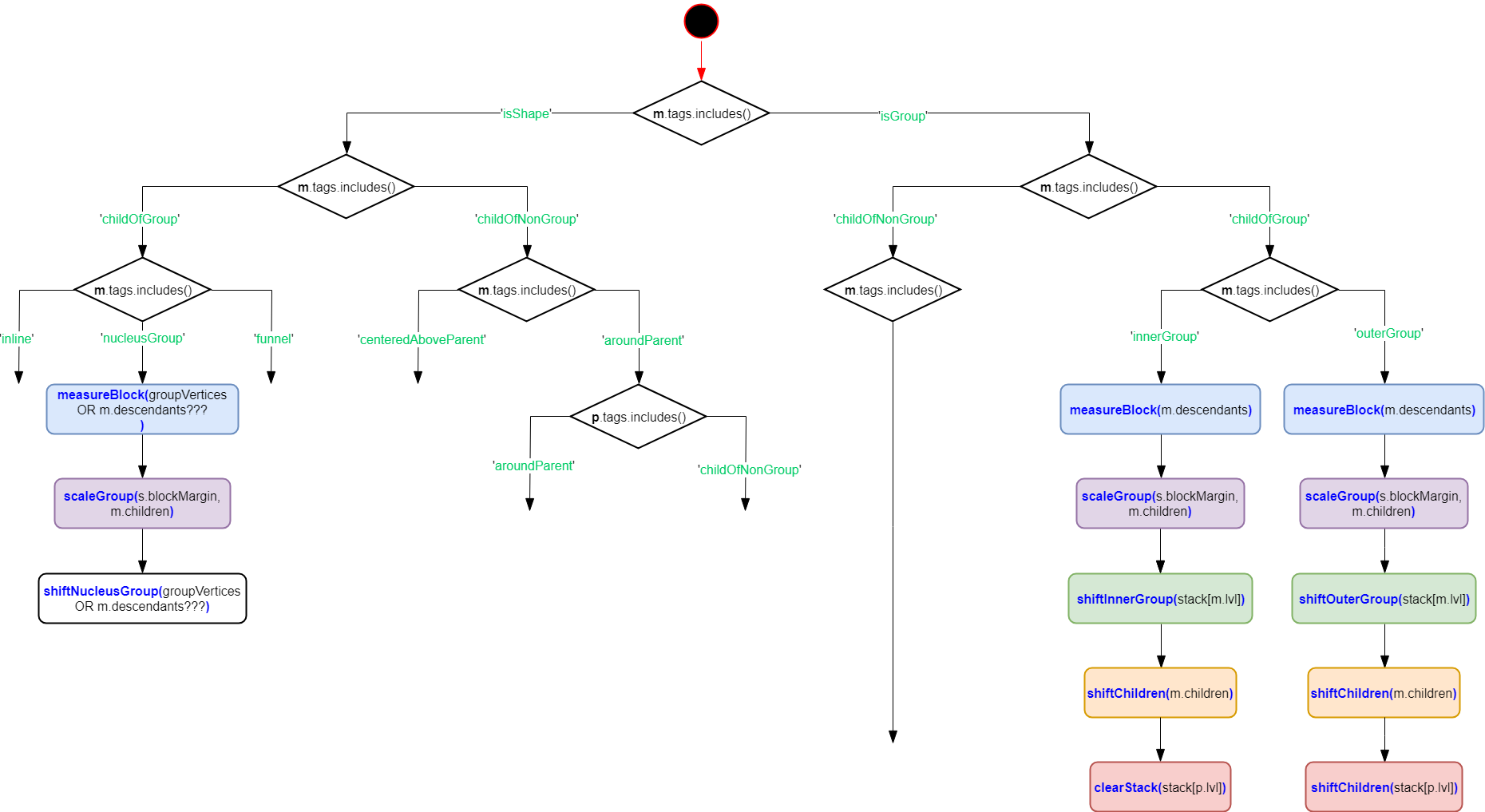
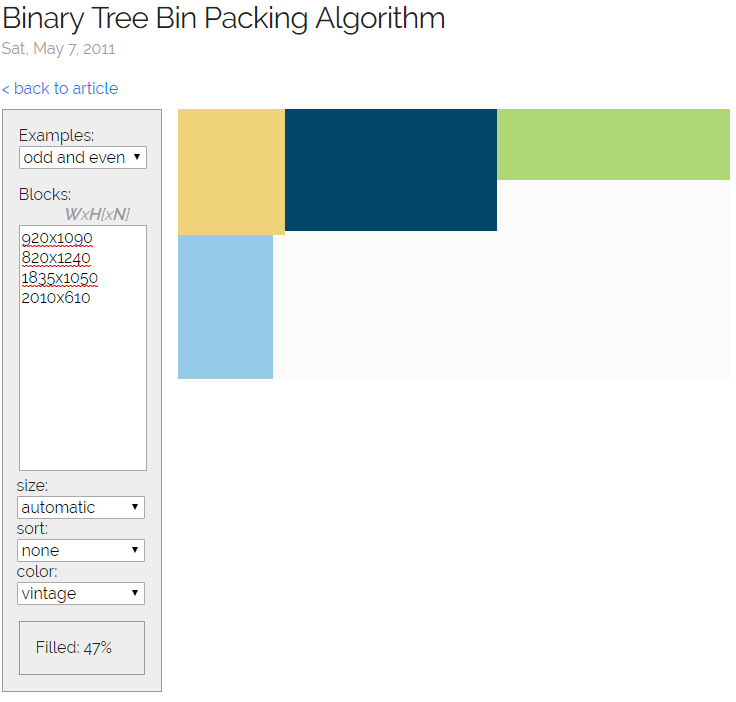
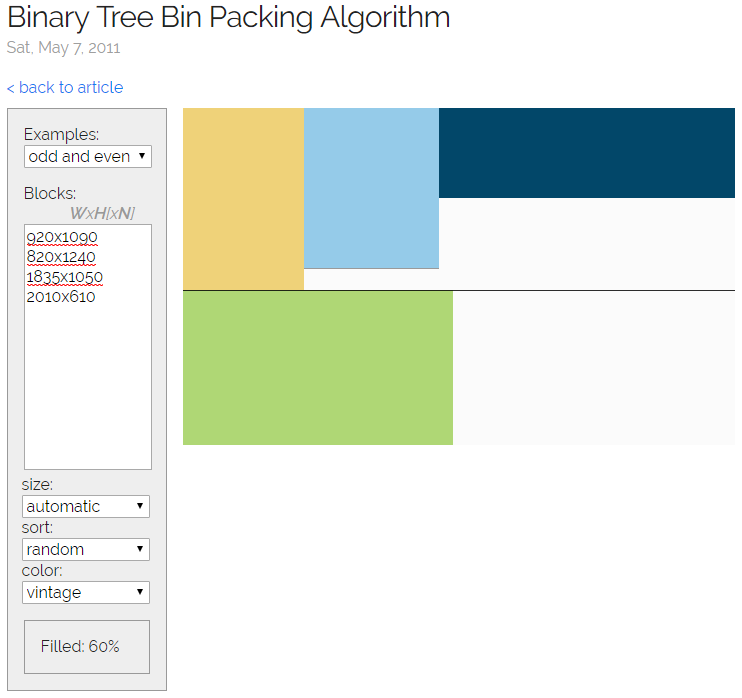
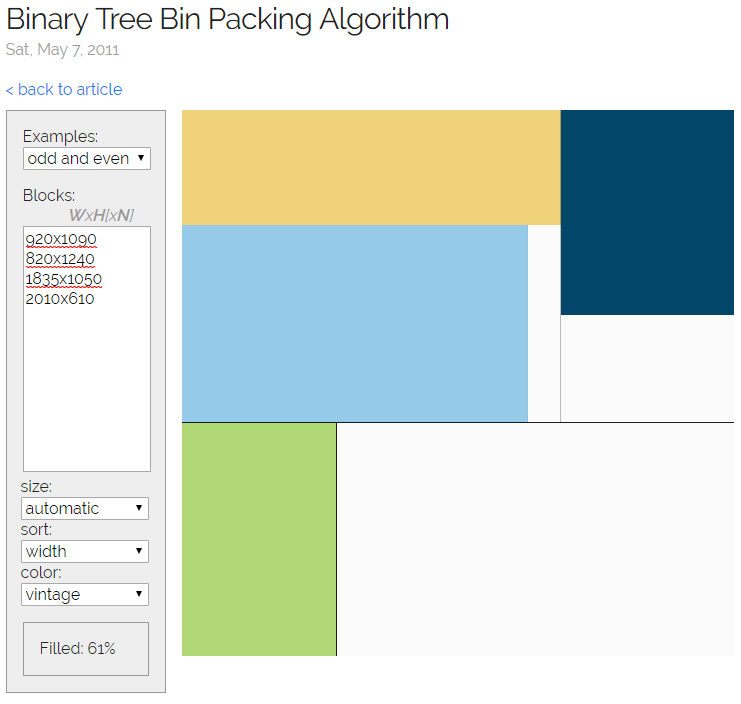
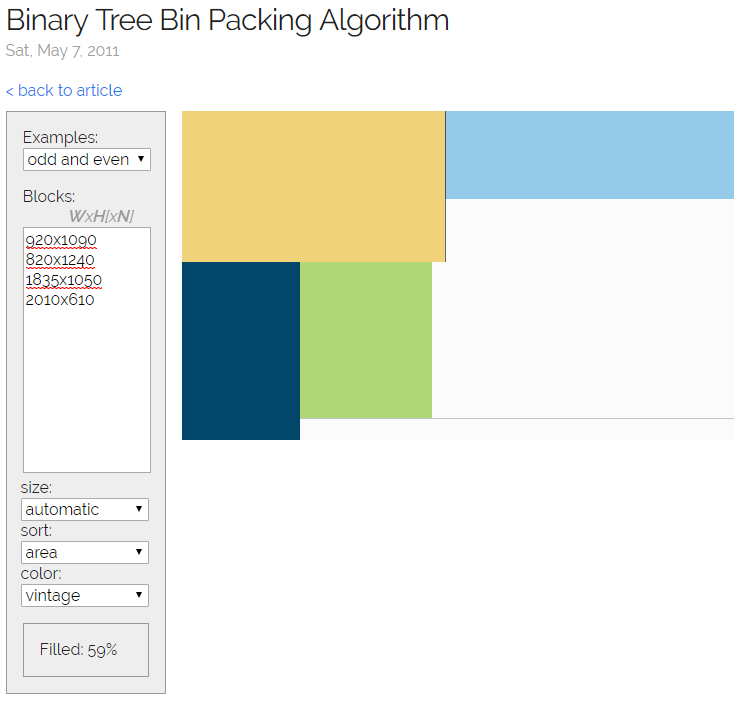
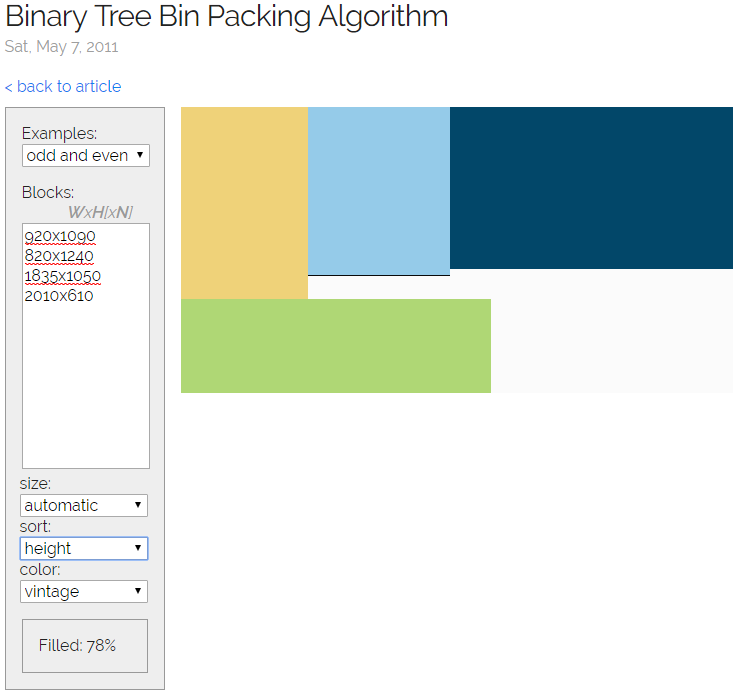
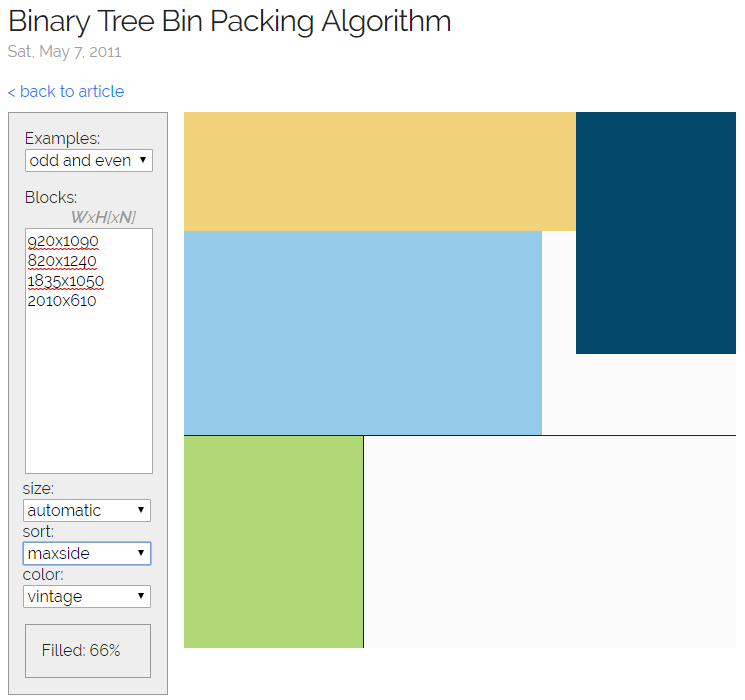


Figure 7[ACTIVITY DIAGRAM OF VERTEX PLACEMENT ALGORITHM WITH CONSTRAINTS]

* Fine Tuning of Parameters
* Group Placemente (#innerGroup, #OuterGroup) algorithm: <https://codeincomplete.com/posts/bin-packing/>
* Orthogonal packing of rectangles in auto-scaling containing group.
* TRIALS WITH THE DEMO WITH ALL SORT SETTINGS: (best results with maxside) RECREATE THE GRAPHICS FROM BELLOW IN DRAW IO



1. Verification, Validation and Evaluation
   1. Prototypical Implementation in an Industrial Context

For Comercial Deployment and Industrial Application

1. Synopsis

List of Figures

[Figure 1 [VORLAUFIGE ZIEL DER VISUALISIERUNG] 2](#_Toc519595442)

[Figure 2 [INSERT CAPTION HERE] 18](#_Toc519595443)

[Figure 3 [ACTIVITY DIAGRAM OF SPECIFICATION OF CONSTRAINTS] 19](#_Toc519595444)

[Figure 4[ACTIVITY DIAGRAM OF VERTEX PLACEMENT ALGORITHM WITH CONSTRAINTS] 19](#_Toc519595445)

List of Tables

[Table 1[QUITAR POR QUE EL LINE TIPE EN REALIDAD SE DEBE ESPECIFICAR EN EL MODEL, NO TENGO QUE HABLAR DE ESTE FALLBACK POR QUE YA VA MAS ALLA DE MI BA] 14](#_Toc519595446)

Abbreviations

|  |  |
| --- | --- |
| Abbreviation | Description |
|  |  |
| API | Application Programming Interface |
| GUI | Graphical User Interface |
|  |  |
| MES | Manufacturing Execution System |
| ProcAppCom | Process Application Composer |
| PCS | Process Control System |
|  |  |

1. Bibliography

[1]

[2]

[3]<https://ac.els-cdn.com/S2405896317314921/1-s2.0-S2405896317314921-main.pdf?_tid=6de94709-7dc6-4269-beb0-fac1338b7c82&acdnat=1532269678_c4b89c7b4e187d7595218b0a1fe60b22>

[4]<https://ac.els-cdn.com/S1474667016434671/1-s2.0-S1474667016434671-main.pdf?_tid=f5714d82-fd23-4d7d-901b-89b13e70dd5b&acdnat=1532269602_0022f42b4de6b59dd533fb07b008d28e>

[5]<https://patentimages.storage.googleapis.com/d2/51/b7/d2fff3397660d9/US20040230328A1.pdf>

[6] <https://pubs.acs.org/doi/pdfplus/10.1021/ie020644a>

[7]

[8]<https://ac.els-cdn.com/S0951832013002433/1-s2.0-S0951832013002433-main.pdf?_tid=86339899-bf4d-4070-b990-5b5512119412&acdnat=1532289234_12ee529d873799f27b7de9a90910e9fb>

[9] <https://link.springer.com/content/pdf/10.1007%2F978-3-642-37057-1_19.pdf>

[10]<https://patentimages.storage.googleapis.com/a9/d3/e4/754ea4055b3871/US20070150821A1.pdf>