05

2018

BUSINESS SIMULATION I.T. TOOL

TECHNICAL REPORT

Action guideline 3:

Development of an innovative Business Simulation Information Technology Tool

Dr. Kosmas Kosmidis, Scientific Coordinator
Dr. Vasileios Mardiris
Dr. Vasileios Chatzis
Dr. Konstantinos Terzidis
Charilaos Mizas
Aleksandra Poltermann
Gabriel Dobrescu
Carmen–Elena Marica
Maricica Herghelegiu

Impressum

Editor

TEI of Eastern Macedonia and Thrace (EMaTTech)

Ag.Loucas

65404 Kavala

Greece

www,teiemt.gr

Authors

Dr. Kosmas Kosmidis, Scientific Coordinator

Dr. Vasileios Mardiris

Dr. Vasileios Chatzis

Dr. Konstantinos Terzidis

Charilaos Mizas

Aleksandra Poltermann

Gabriel Dobrescu

Carmen–Elena Marica

Maricica Herghelegiu

Funding

Co-funded by the Erasmus + Programme of the European Union

Year of publication

2018

Online available

http://innovet.teiemt.gr

1. Introduction

Globalization, economic pressures, and the changing nature of work are a number of emerging challenges which have combined to create a changing business environment that demands innovative and flexible training solutions. Business simulation Information Technology Tools are promising for creating more realistic, experiential learning environments in order to help organizations meet these emerging training challenges.

Modern educational systems use information technologies to increase their immediacy and their efficiency. Business simulation Information Technology tools as a form of experiential learning are focused on improving business decision making skills by using student's natural capacity for technology. Business simulations provide a space in which learning is an outcome of tasks stimulated and executed by the content of the simulation, while the knowledge is developed through the content of the simulation, and skills are developed as a result of playing the simulation game.

A modular tool for production planning and control processes simulation, provides a significant overcome to the proposed in-company dual system. The computer-aided planning and processes simulation makes the trainees face the main problems of control in industrial production.

The VET (Vocational Education and Training) IT tool serves a simplified model of a factory, in which decisions regarding: purchase orders, production orders and production capacities have to be made. The production output of the factory model has to be adjusted to the market requirements by the trainees through selected variables such as: throughput time, delivery capability, loading, inventories, manufacturing costs etc. Modularity concerns the ability given to the system's administrator to plan and simulate any manufacturing flow he needs best for his VET. Simulation results will be presented through graphs, lists or tables of several objects.

2. Requirements

The first of the development processes of each System is the Analysis of Requirements, which is usually described as the definition of the work to be performed by the system and the software, as well as the limitations and assumptions that will apply to this. Determining system requirements is a task similar to defining software requirements, but apart from the software, it also concerns the other components of a system, like people and machines.

The software requirements are either a function that this should perform or a condition that should satisfy when its development is completed. The requirements usually distinguished in functional and non-functional. The functional requirements describe the tasks (functions) that the software should perform and fully define the behavior of the system. Non-functional requirements describe features that software must have which do not involve the tasks should be performed from it. Non-functional requirements define idioms, operating environment, performance, etc., which generally characterize the software, but can not be seen as software's functions.

For the VET tool, the requirements defined, the roles and licenses of the users and the basic entities (and the relationships between them) are presented in the following.

2.1 Functional Requirements

The VET tool must be modular and have the following main functions:

1. Model design mode
    a) Importing Model Elements
    b) Delete Model Elements
    c) Move Model Elements
    d) Linking Elements / Defining Flows

e) Specifying data

2. Simulation mode
   a) Define simulation parameters
   b) Run Simulation
   c) Save Simulation
3. Results mode
   a) Present Simulation Results
4. Educational Functionality
   a) Defining Parameters
   b) Decision making
   c) Graphic Design

The system must have also and the following supporting functions, regarding the users who will use the system and their permissions in managing the models.

1. Add users
2. User Login
3. Delete Users
4. Assign permissions
5. Modify permissions

2.2 Non-functional requirements

1. The Information System will be web based.
2. The software will be installed on the Server consisting of a) Database Support System and b) a set of source code files that will implement all the functions of the System and will be developed using the Javascript, PHP and HTML.
3. All users will use the system using a browser without requiring any other installation
4. All modifications made by users will be stored in the database in real time

and will be immediately visible to all System users. For this reason needed, continuous access to the Web and the unbreakable Web clients communication with the Server System

5. The graphical user interface should be easy to use

3. Development

The application produced combines a multitude of technologies to achieve the desired result and meet the requirements.

3.1 Front end

The front end of the web application was coded using HTML 5 and CSS [1]. JavaScript was used to add extra functionality, and in particular the JavaScript framework PatternFly [2] was used. There are two parts to PatternFly that helps the efficiently design and develop enterprise web applications. First, the widget library implements enterprise-optimized visual styling for common web UI widgets. Second, the pattern library includes a set of interaction patterns that offer consistent solutions to common user interaction problems. Using PatternFly ensures the user's functionality and usability because users are familiar with the interactions. Visual consistency establishes a look and feel that users will recognize. CSS was used to handle the styling of the application.

jQuery [3] also was used to add DOM (Document Object Model) manipulation and handle AJAX requests. The using of AJAX improve the user experience by removing the need for page reloads after changing the data.

The implemented application gives the ability to user to design a production process from structural modules like machines and warehouses. To design the production process used Scalable Vector Graphics (SVG) [4]. For the creation of SVG elements d3.js [5] library was used. D3 also lets to create a lot of different types of charts and graphs to represent data in efficient ways, characteristics that were necessary for

the presentation of results of the simulation process.

3.2 Back end

The back-end is all of the technology required to process the incoming request and generate and send the response to the client. For the implementation of the application's back end php scripting language was used. As database is required to store application data, for the purpose of the project we used the MySQL relational database managing system.

A good database design contributes to software development and improve performance. It is always very important to keep in mind the data retrieval, storage and modification efficiency. Good data modeling will provide easier and more intuitive access than others. In Appendix A the database structure is presented in detail.

All modifications made by users are stored in the database in real time and are immediately visible to all System users (if they have the right permissions). For this reason the use of Server-Sent Events (SSE) is necessary [6].

The information system was installed on an Ubuntu-based server with Apache and MySQL working for 24 hours a day, 7 days a week.

3.3 Technologies

3.3.1 Paternfly

PatternFly is an open source project that promotes design commonality and improved user experience. Open source developers and designers can collaborate, create, and share user interface (UI) components and widgets. PatternFly is based on Bootstrap, a mobile-first front-end framework for creating web sites and applications.

The PatternFly library is a collection of UI design patterns. Design patterns are recurring solutions that solve common design problems and provide a common language for people who create user interfaces.

PatternFly is under MIT license and it can be downloaded from the PatternFly home page [2].

3.3.2 SVG

Scalable Vector Graphics (SVG) is a graphical standard maintained and endorsed by the World Wide Web Consortium (W3C). SVG is a markup language for describing two-dimensional graphics applications and images, and a set of related graphics script interfaces. SVG is supported by all modern browsers for desktops and mobiles. Some features, such as SMIL animation and SVG Fonts are not as widely supported. There are many SVG authoring tools, and export to SVG is supported by all major vector graphics authoring tools [7].

3.3.3 D3.js

D3 is one of the most powerful and flexible data visualization frameworks [5]. **D3** (**Data-Driven Documents** or **D3.js**) is a JavaScript library for visualizing data using web standards. D3 helps developers bring data to life using SVG, Canvas and HTML. D3 combines powerful visualization and interaction techniques with a data-driven approach to DOM manipulation, giving the ability to exploit all the capabilities of modern browsers and the freedom to design the right visual interface for your data. Some popular uses include creating interactive graphics for online news websites, information dashboards for viewing data, and producing maps from GIS map making data [8][9]. D3.js is under BSD license and it can be downloaded from the D3.js home page [10].

3.3.4 jQuery

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers [3].

The core features of jQuery assist us in accomplishing the following tasks [11]:

- Access elements in a document
- Modify the appearance of a web page
- Alter the content of a document
- Respond to a user's interaction
- Animate changes being made to a document
- Retrieve information from a server without refreshing a page
- Simplify common JavaScript tasks

jQuery is under MIT license and you can downloading from the D3.js home page [3].

3.3.5 JSON

JSON (JavaScript Object Notation) is a lightweight text-based code to create objects to transfer data over the Internet. It is a data exchange format that is human-readable (like XML, but without the markup around your actual payload) and its syntax is a subset of the JavaScript language that was standardized in 1999 [12]. JSON is built on two structures:

- A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
- An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

These are universal data structures. Virtually all modern programming languages support them in one form or another. It makes sense that a data format that is

interchangeable with programming languages also be based on these structures [13].

3.3.6 SSE (Server Server-Sent Events)

Server-sent events (SSE) is a technology where a browser receives automatic updates from a server via HTTP connection. Server-Sent Events enables efficient server-to-client streaming of text-based event data—e.g., real-time notifications or updates generated on the server. To meet this goal, SSE introduces two components: a new EventSource interface in the browser, which allows the client to receive push notifications from the server as DOM events, and the "event stream" data format, which is used to deliver the individual updates [14].

Server Sent Events are supported in the major browsers (Chrome, Firefox, Safari), but not yet in MS Edge where this feature is currently “under consideration” for implementation [15].

The Server-Sent Events EventSource API is standardized as part of HTML5 by the W3C [6].

3.4 Maintaining Security

During development, the necessary attention was given to increasing application security. The following list provides some of the security guidelines that applied to information system:

1. Access to the application requires username and password
2. The users passwords are encrypted
3. A logout function for the application, and an idle session timeout implemented.
4. Only the authorized users be allowed to view/edit designs and simulations information

5. Session management is widely applied
6. Finally, to increase security, https protocol was used instead of http.

3.5 Subsystems of INNOVET Business Simulator

For the InnoVET project "A Innovative Modular Double System Based on Modeling and Simulation of Business Processes for Vocational Education and Training with Business", a modular tool for production planning and control processes simulation (INNOVET Business Simulator) was developed.

Figure 1: The four subsystems of the INNOVET Business Simulator

INNOVET Business Simulator consists of four subsystems (Figure 1):

1. **Administration**: Through this subsystem, the administrator manages user accounts and application data such as designs and simulations. Figure 2 shows all available functions in the Administration subsystem. As we can see administrator can create a new user, modify the existing users data and reset users password. Also from this subsystem administrator can erase from database the data of specific designs and simulations.

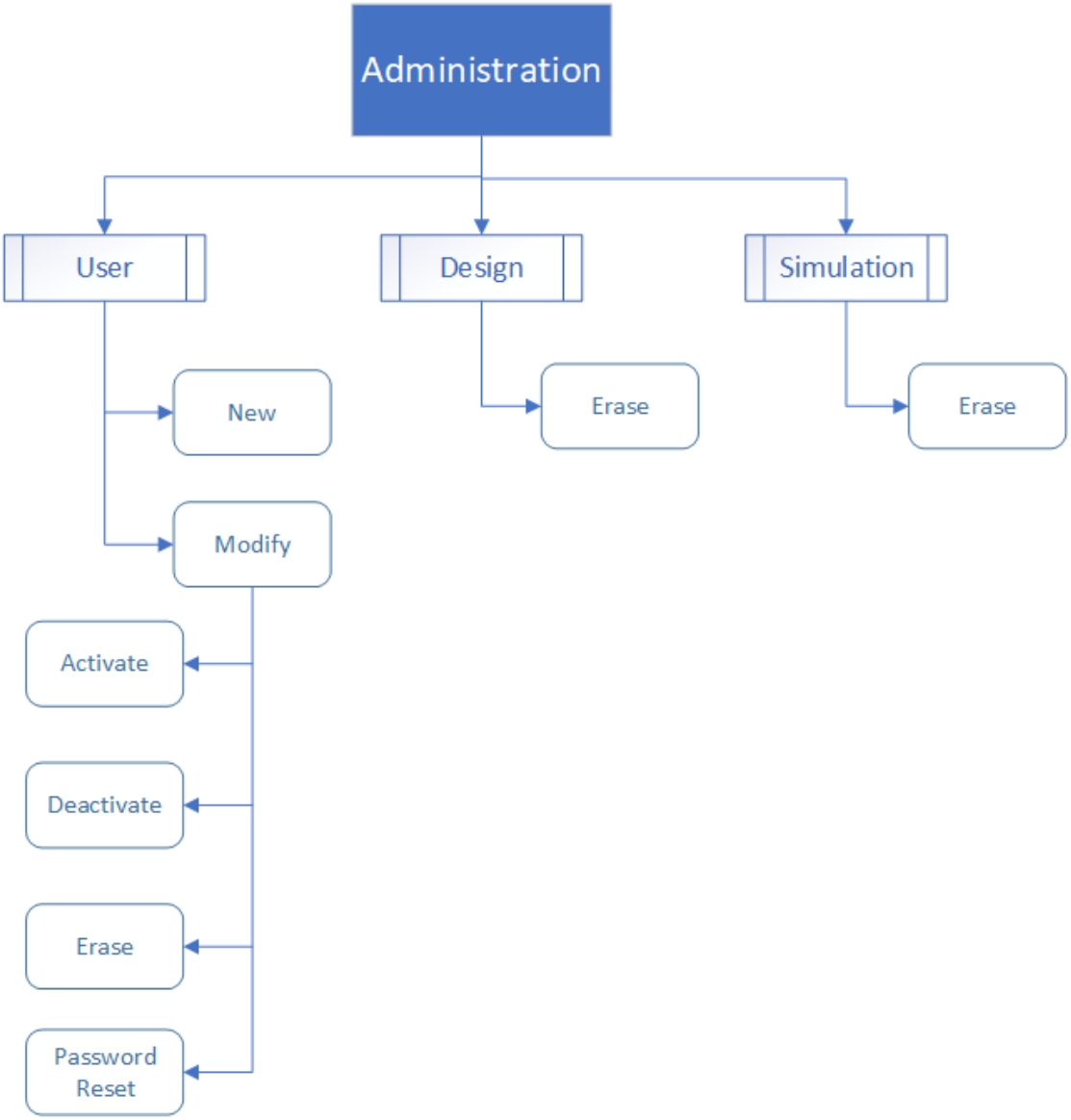

Figure 2: Administration Subsystem

2. **Design Mode**: Through Design Mode, the user of INNOVET Business Simulator can model any production or service process by using the provided structural modules (machines, warehouses, products and carriers). Specifically, the user has available the following functions (Figure 3):

- Create a new design
- Open and edit an existing design

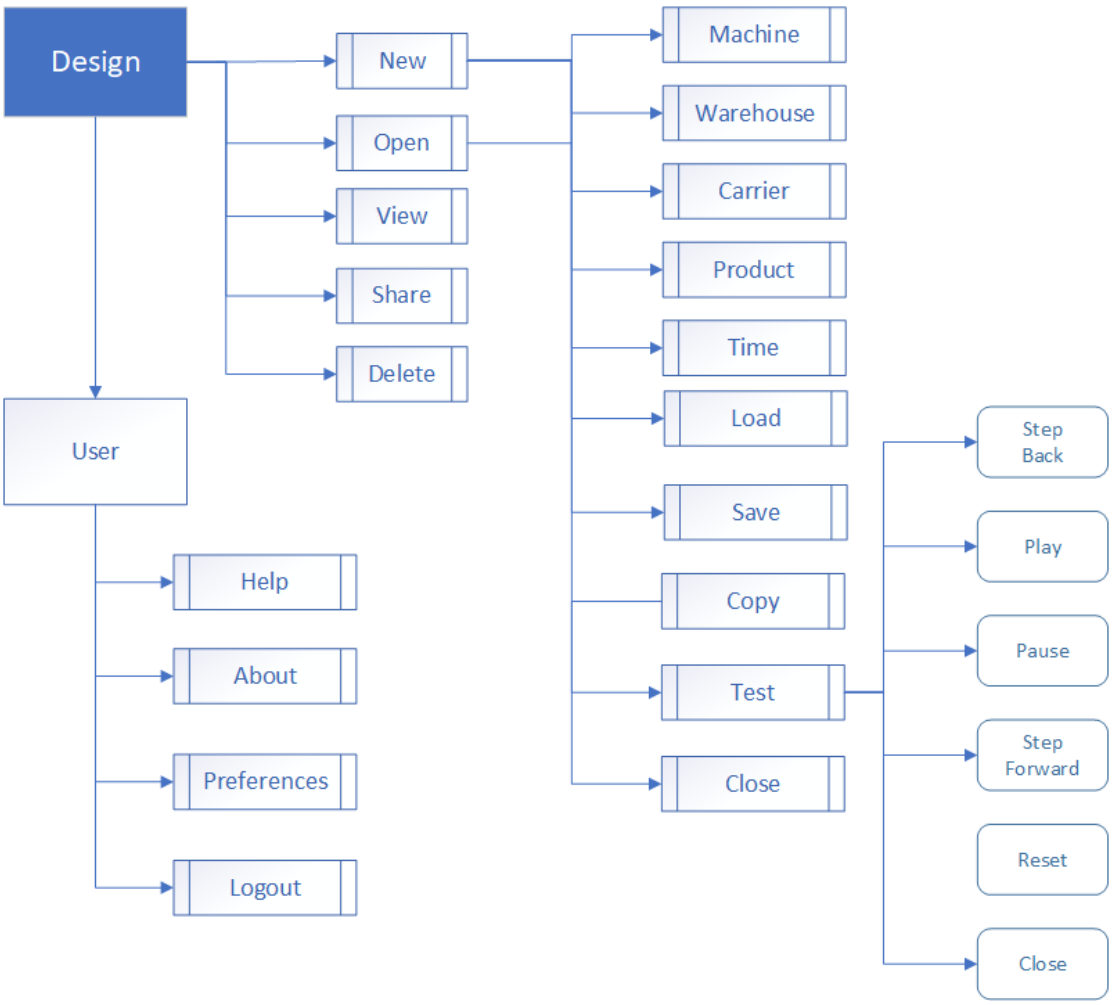

Figure 3: Design Mode

- When create a new design or edit an existing one, user can add, modify, connect and delete structural modules (machines, carriers, warehouses and products) or set design's time unit.
- View an existing design
- Share a design with other users and grant specific permissions for it

(Table 1)

- Save a new version of the design
- Load an existing design version
- Test the design (logical test) step by step or continuously
- Delete a design. After the deletion the design status is inactive but the design's data remains in the database. Only Administrator can erase design's data from database

Permission	Description
Full	User have full access to a design (Design owner)
Copy	User can make a copy of the design
Test	User can test the design
View	User can only view a design and has no right to change anything
	No permission on the design

Table 1: Design Permissions

3. **Simulation Mode:** The user can simulate the production process of a factory model (designed in the Design Mode subsystem) by defining various parameters such as purchase orders, production orders, human schedule etc. Specifically, the user has available the following functions (Figure 4):
   - Create a new simulator
   - Open and edit an existing simulator
   - View an existing simulator
   - Share a simulator with other users and grant specific permissions for it (Table 2)

Permission	Description
Full	User have full access to a simulator (Simulator's owner)
Copy	User can make a copy of the simulator
Order	Run permissions + set or change orders (Machine and Human Schedule, Purchase and Sale Orders
Run	View permissions + can run simulator
View	User can only view a simulator and has no right to change anything
	No permission on the simulator

Table 2: Simulator Permissions

- When create a new simulator or edit an existing one, user can set simulation parameters: Simulation time, Machine and Human Schedule, Purchase and Sale Orders
- Save a new run of the simulator
- Load an existing run of a simulator
- Run a simulator
- Delete a simulator. After the deletion, the simulator status is inactive but the simulator's data remains in the database. Only Administrator can erase simulator's data from database

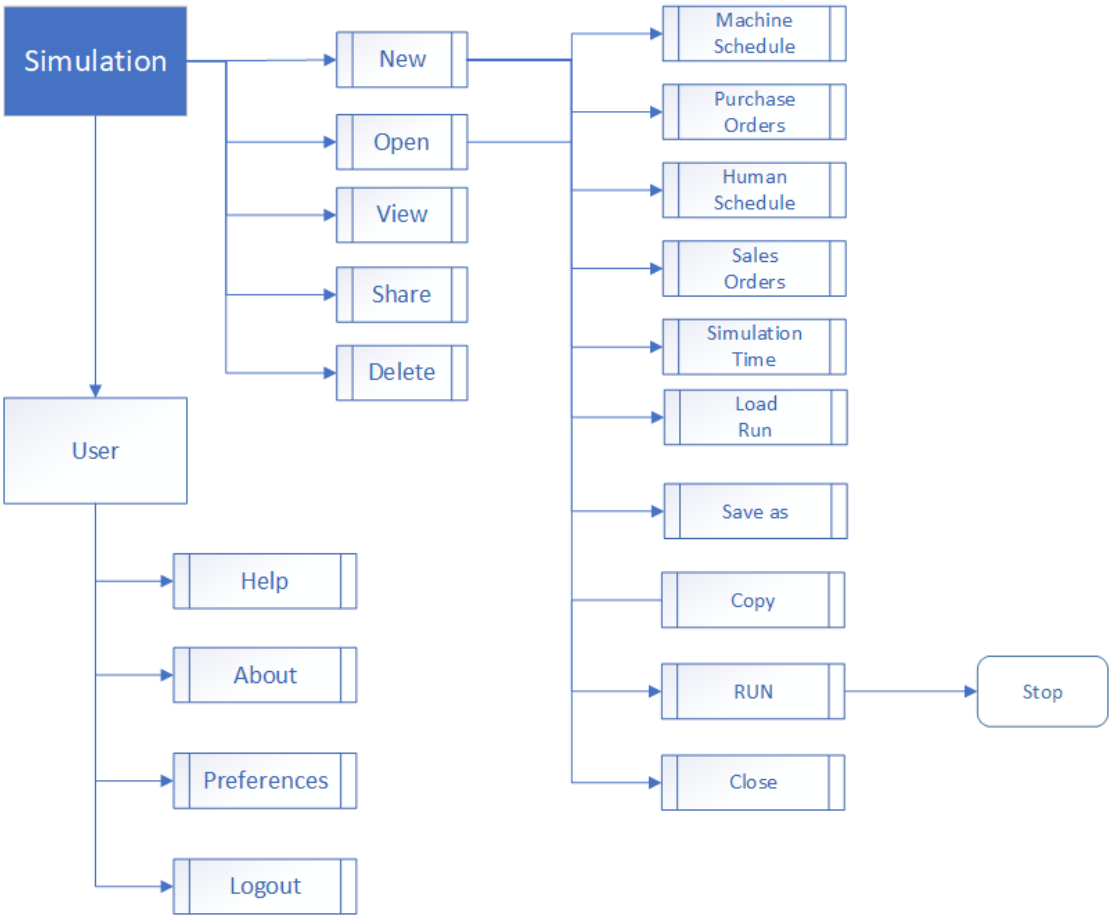

Figure 4: Simulation Mode

4. **Results**: Through this subsystem, the simulation results of the production process are presented through graphs or tables of several objects, as shown in Figure 5.

Co-funded by the Erasmus+ Programme of the European Union

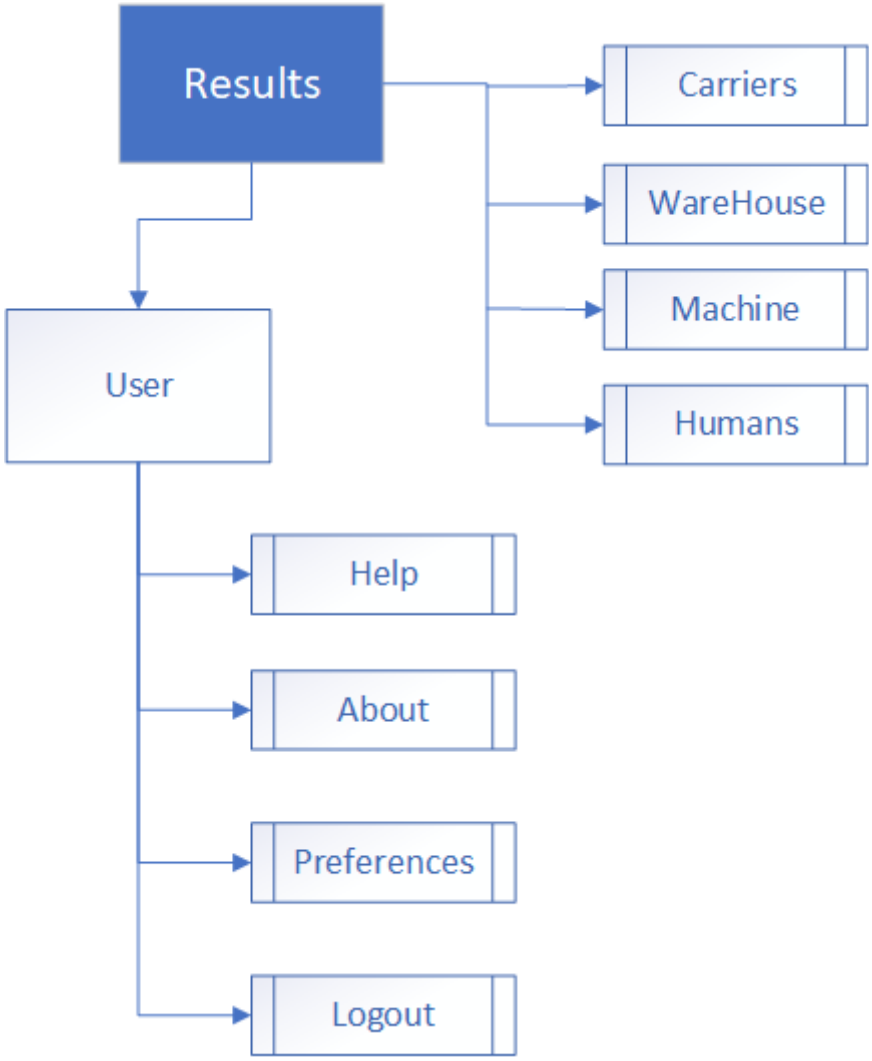

Figure 5: Results Subsystem

Also the user have the following abilities from all subsystems:

- Help: Access to Help
- Logout: Close the application
- My Account: modify personal information and reset password
- About: Access to information about the application

3.6 Structural modules

Modularity concerns the ability given to the application's user to plan and simulate

any manufacturing flow he needs best for his VET. Next, the structural modules that are available to the user for designing a production line, will be presented.

3.6.1 Product

It includes the concepts of end product/ service, intermediate and raw materials. The main parameters of a product are:

- Product id
- Product name
- Metric unit of a product

The parameters of a product are stored in the database table *product* (Appendix A). Every design can have many products. Each design has its own products.

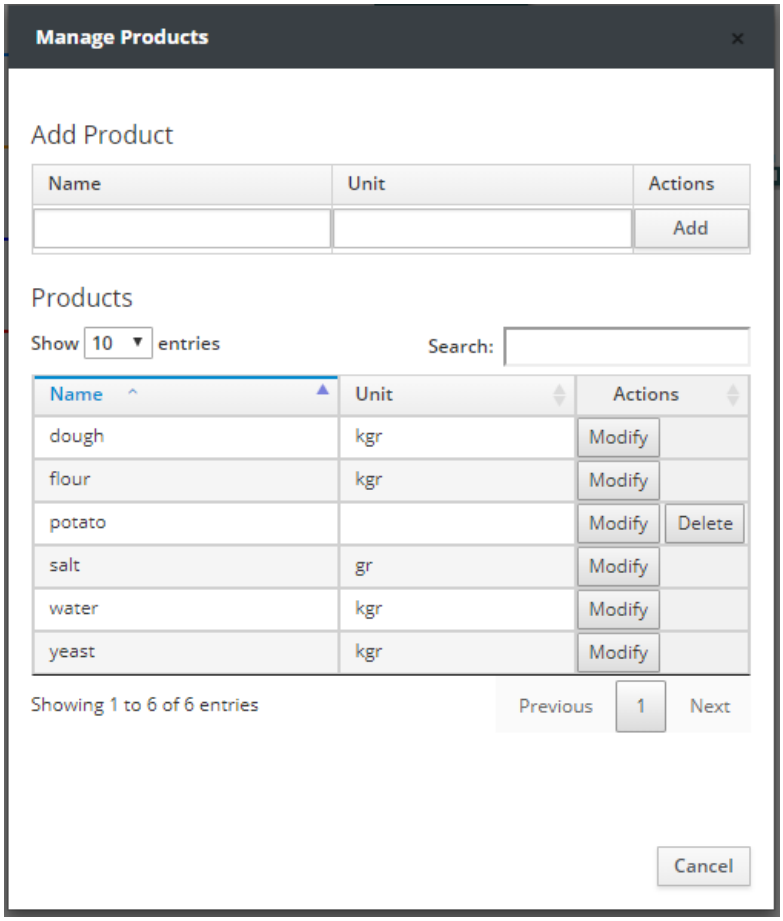

Figure 6: Manage Products at Design Mode

As seen in Figure 6 the user can add a new product, modify or delete an existing product at the Design Mode.

3.6.2 Machine

It includes the concept of machine/processing that accepts as input products and gives output products.

The main parameters of a machine are:

- Machine id
- Machine name
- The minimum input time units for two successive loadings at the machine inputs
- Minimum number of humans for the machine to operate
- Number of inputs
- Number of outputs
- The necessary input product/products (name and amount of product)
- The output product/ products
- Production time
- Machine state

The parameters of a machine is stored in the database tables *machine* and *machine_io* (Appendix A). Every design can have many machines and every machine's input or output must be connected to a carrier.

3.6.3 Warehouse

It includes the concept of a warehouse that accepts as input products and gives output products

The main parameters of a warehouse are:

- Warehouse id

- Input Products
- Maximum amount for every product
- Minimum number of humans for the warehouse to operate

The parameters of a warehouse is stored in the database table *warehouse* (Appendix A)

3.6.4 Carrier

It includes the concept of product transmission line.

The main parameters of a carrier are:

- Carrier id
- Name of transportable product
- Maximum amount of a product to the transmission line

The parameters of a carrier is stored in the database table storage (Appendix A). Carriers must be connected to every machine's input or output.

3.7 Design

Using the structural modules mentioned in previous paragraph, the application's user can model any production or service process at the Design Mode subsystem of Innovet Business Simulator.

User saves the model in a design. The main parameters of a design are:

- Design id
- Design name
- Creation Date of the design
- Design Time unit that is necessary for the machines operation
- Design status (active or inactive)
- Creator – Owner
- Machines, carriers, warehouses and products that belongs to design.

The parameters of a design is stored in the database table *design* (Appendix A) after the creation.

The user can save a new version of the design by clicking the *Save Version* button (Figure 18). The data of the design's new version stored in the database table *version* (Appendix A).

The creator of a design is the owner and have full access to it. The owner can share the design with other application's users and can give specific permissions to them as demonstrated in Figure 7.

The owner of a design, as mention it, has full access and can delete the design by clicking the *Delete* button (Figure 16). After the deletion of a design the design status is inactive but the design's data remains in the database. Only the application administrator can remove it from the database if it was not used in a simulation. In Figure 1 we see a warning message that we can not delete specific designs.

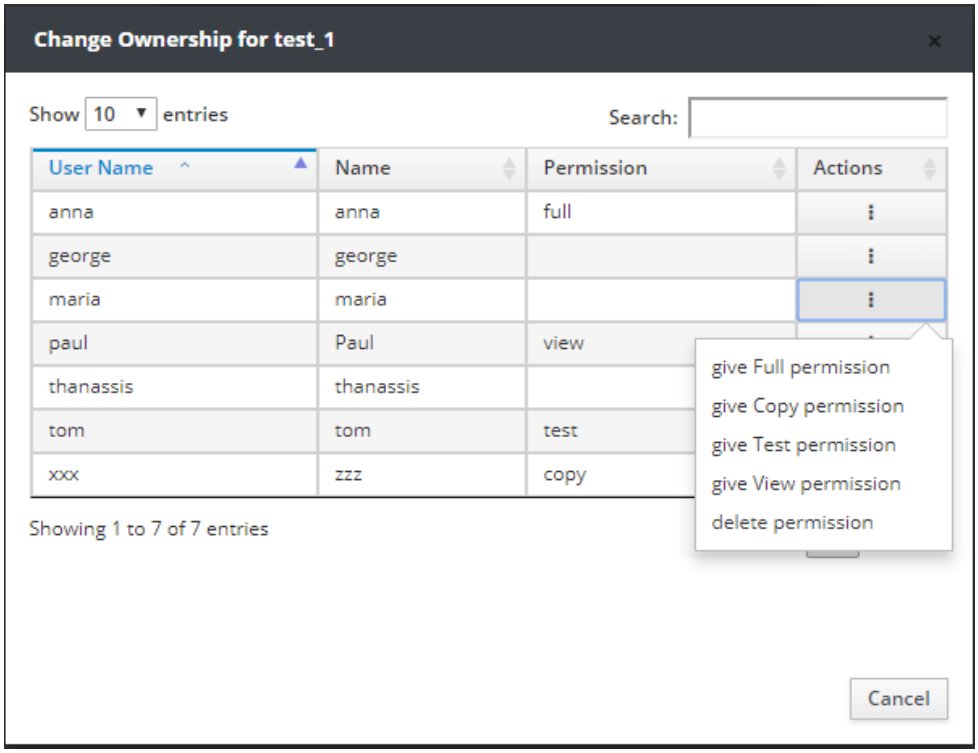

Figure 7: Changing design's users permissions

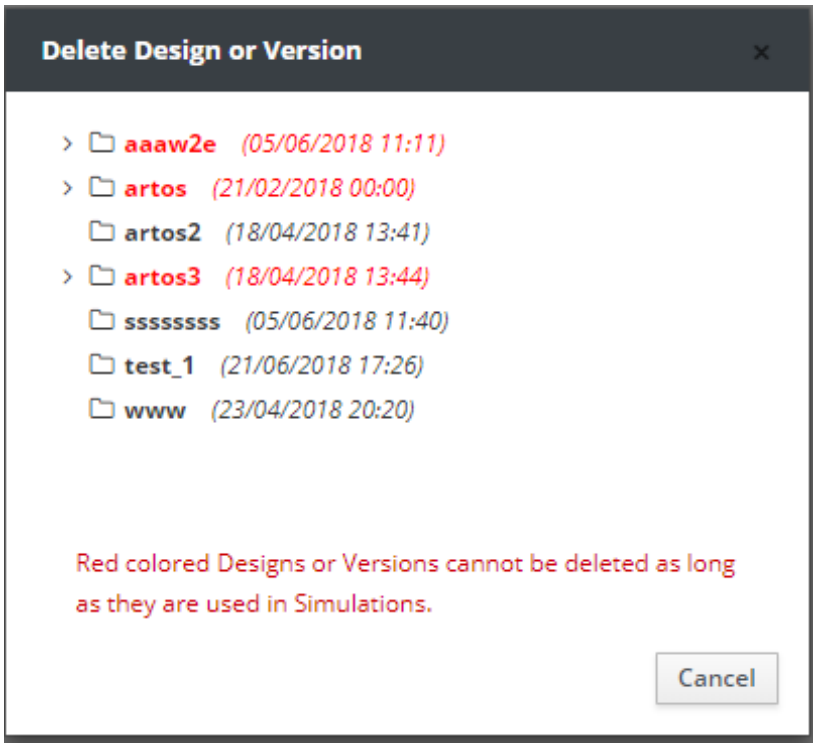

Figure 8: Design or Version Deletion

3.8 Machines and Carriers Logical Operation

As mentioned in previous section, a design consists of connected structural components such as machines and carriers to model any production or service process. Each machine has multiple inputs through which load products and after some time units (Production time) the produced product or products are ready on machine's outputs. Every machine's input or output must be connected to a carrier (product transmission line). In Figure 9 a flowchart regarding the logical operation of machines and carriers is presented. Figure 10 presents a flowchart to describe how a machine produces products at outputs and connected carriers. Figure 11 presents a flowchart to describe how a machine loads products from the carriers into the inputs.

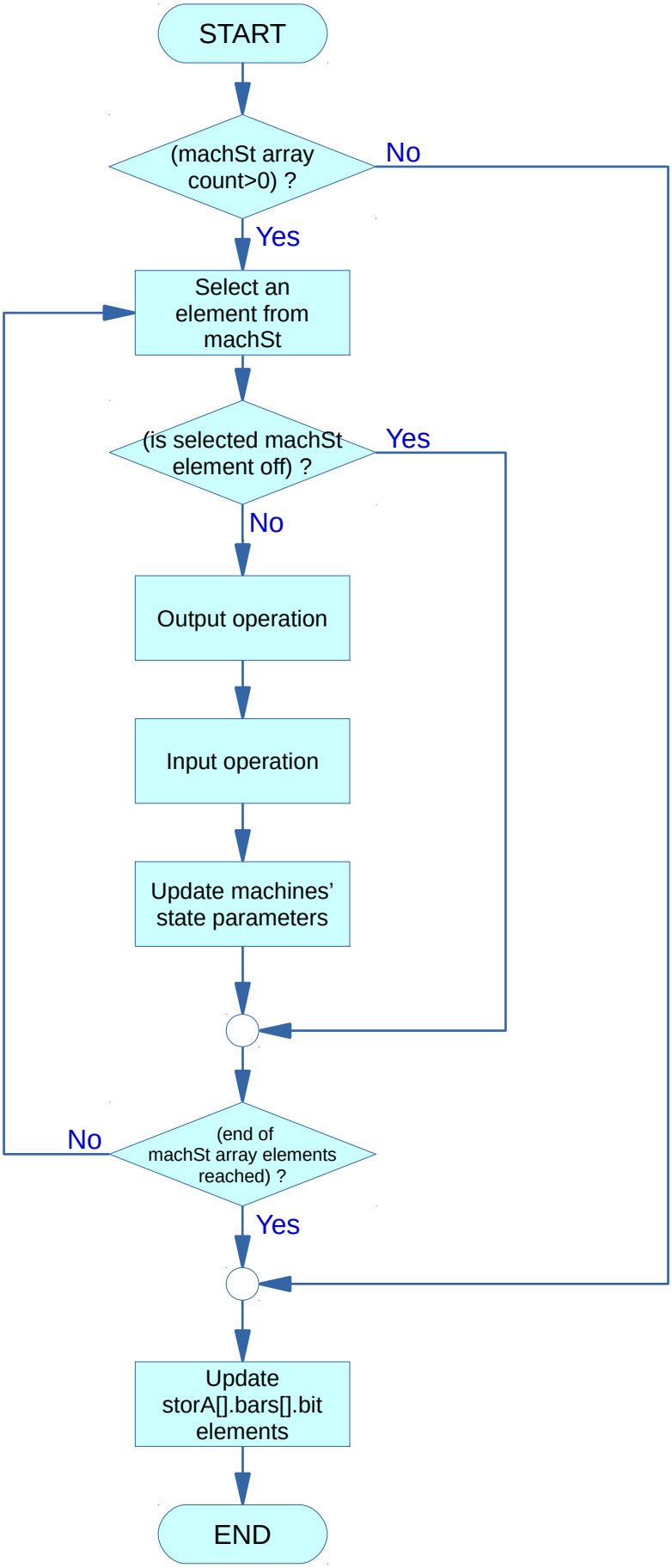

Figure 9: Modules Operation Flowchart

Co-funded by the Erasmus+ Programme of the European Union

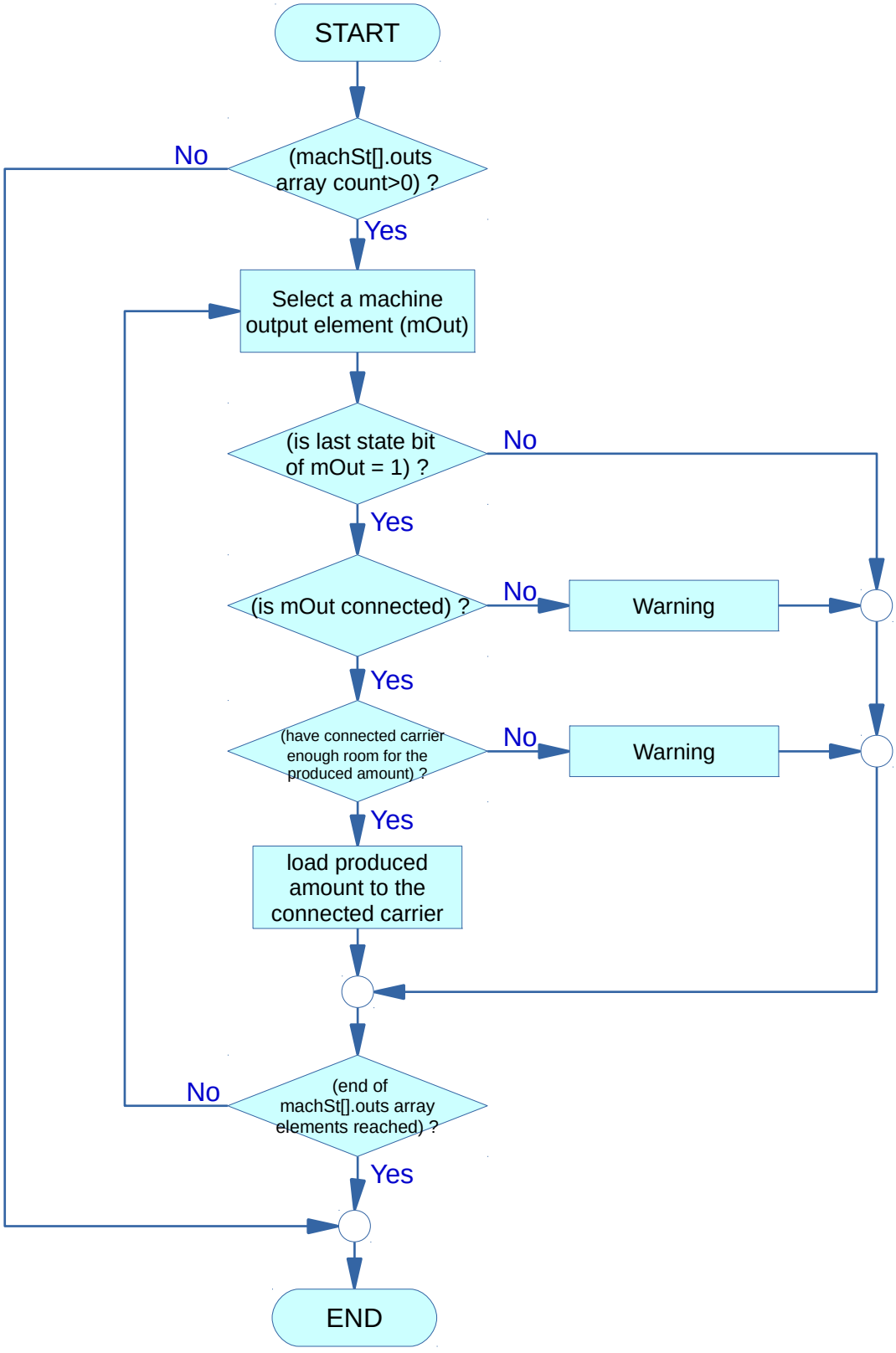

Figure 10: Output operation flowchart

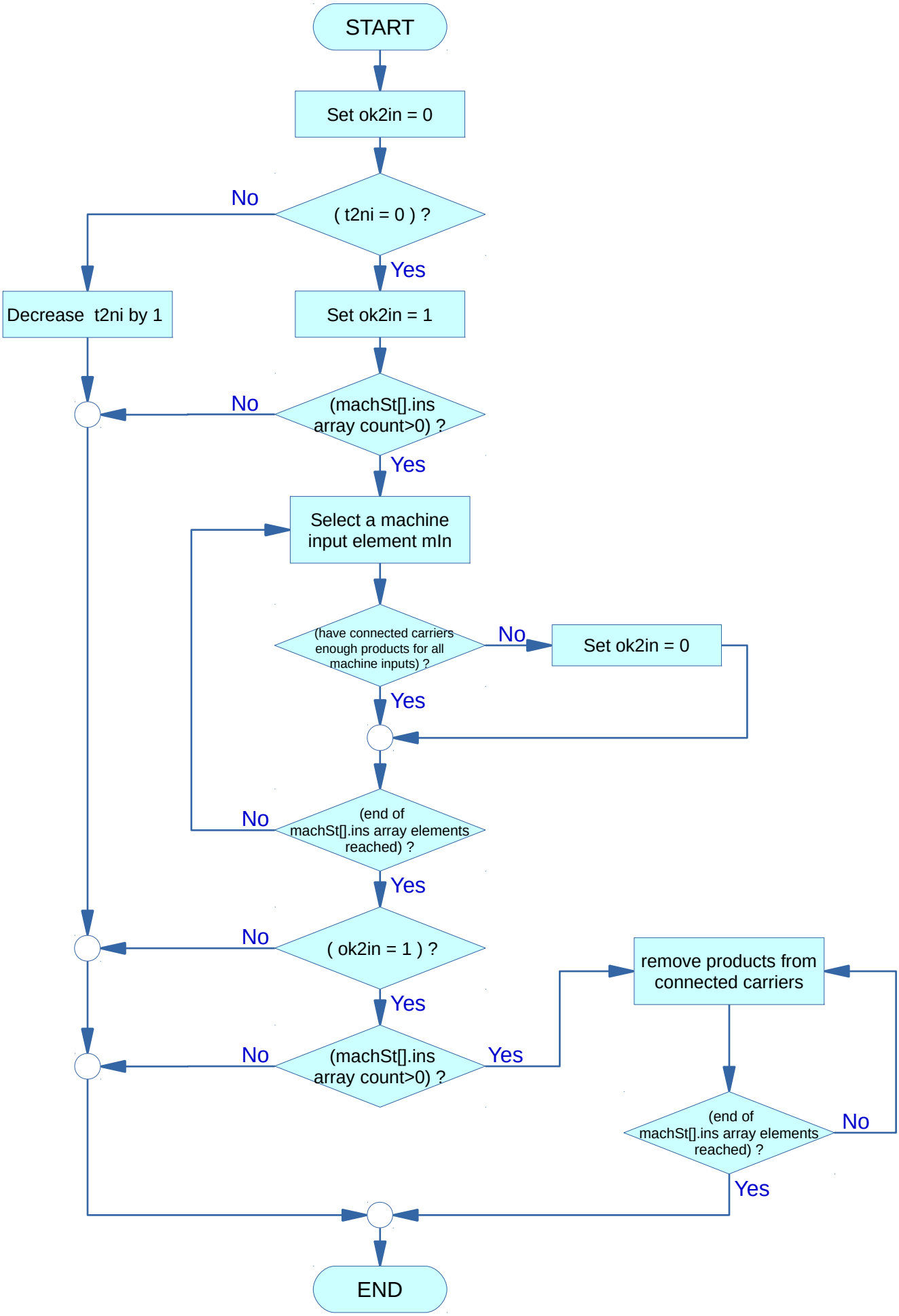

Figure 11: Input operation flowchart

3.9 Global Variables

Variable name: *currentMode*

Description: The location of the current run.

Values of the variable: am, dm, dm-design, dm-test, sm, rm

Variable name: *allowSSE*

Description: Enables or disables the SSE (Server Sent Events) capability

Default Value: YES

Variable name: *designId*

Description: Working Design id

Variable name: *machineX, machineY*

Description: Arrays for x, y coordinates of a machine module

Variable name: *storageX, storageY*

Description: Arrays for x, y coordinates of carrier/ storage module

Variable name: *MACHINE_INS_OUTS_MAX*

Description: Maximum number of machine inputs and outputs

Default Value: 5

Variable name: *machStorEdit*

Description: Enables or disables right click on machines and carries/ storages

Default Value: YES

Variable name: *TIME_STEPS*

Description: Time steps values for the simulation

Values of the variable: 1: second, 60: minute, 3600: hour, 86400: day, 604800: week

Variable name: *TIME_STEPS_LIMIT*

Description: The maximum number of simulation time steps

Default Value: 10.000

Variable name: *designTimeResolution*

Description: Time resolution for test purposes of the design.

Variable name: *testDesignEnv*

Description: Test operation environment selected

Default Value: NO

Variable name: *testTime*

Description: Design Test time

Default Value: 0

Variable name: *sse_source*

Description: sse_source object definition

Variable name: *WORKING_HOURS*

Description: *Human* Working hours for simulation purposes

Default Value: 4, 6, 8, 10, 12

Variable name: *simId*

Description: Working Simulation Id

Variable name: *resultsId*

Description: Working Results id

Variable name: *ok2in*

Description: Determines whether the products should be loaded into the inputs of the machine.

Variable name: *jsonDesign*

Description: Array containing information for the structural modules of a design. The table elements are JSON objects machines and storages.

Variable name: *machSt*

Description: Machine state data JSON object

Variable name: *storA*

Description: Carrier state data JSON object

3.10 JSON Objects

Machines: Object for every machine module on the design

humans_min: Minimum number of humans for the machine to operate

Id: Machine id

input_time: The minimum input time units for two successive loadings at the machine inputs (next input time)

ins: Array of machine's inputs

a: Amount of a product in the input

color: Color of the input on the design view

id: Input id

row_num: The order of the input on the design view

storage_id: The carrier id is connected with the input

name: Name of the machine

outs: Array of machine's outputs

a: Amount of a product in the output

color: Color of the output on the design view

id: Output id

prod_time: Production time. How many time units needed for the production of a product

row_num: The order of the output on the design view

storage_id: The carrier id is connected with the output

x: x coordinates that determine the position of the machine on the design view

y: y coordinates that determine the position of the machine on the design view

machSt: Machine state data JSON object

InputTime: The minimum input time units for two successive loadings at the machine inputs (next input time)

Ins: Array of JSON Objects for every machine's input

a: Amount of a product in the input

storageId: The carrier id is connected with the input

outs: Array of JSON Objects for every machine's output

a: Amount of a product in the output

color: Color of the output on the design view

id: Output id

prod_time: Production time. How many time units needed for the production of a product

state: Array of JSON Objects (number of rectangles) for machine's status bar. The status bar consists of a number of rectangles that show the time units for the production of a product

bit: if the "bit" value is "1" then the rectangle color is solid

obj: the rectangle object

stateN: The decimal number showing the state of production (how many rectangle is solid)

storage_id: The carrier id is connected with the output

t2ni: time to next input

storages: Object for every storage of a carrier module on the design

a: Current amount of the product in the carrier

aMax: The maximum amount of the product in the carrier

color: Color of the line that connect the carrier with the other modules

id: Carrier id

name: Carrier name

x: x coordinates that determine the position of the carrier on the design view

y: y coordinates that determine the position of the carrier on the design view

storA: Carrier state data JSON object

a: Current amount of the product in the carrier

aMax: The maximum amount of the product in the carrier

bars: Array of JSON Objects (number of rectangles) for carrier's status bar. The status bar consists of a number of rectangles indicating the percent of carrier's fullness

bit: if the "bit" value is "1" then the rectangle color is solid

obj: the rectangle object

txtobj: The carrier's text object

3.11 Example of Design's JSON Objects

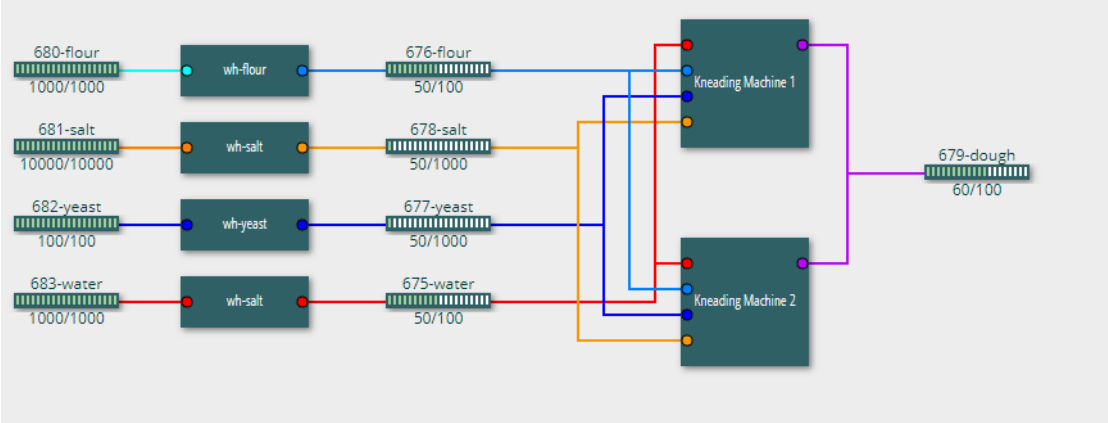

Figure 12. A Design Example

The design in Figure 12 consists from six machines and nine carriers connected to the inputs and outputs of the machines. The JSON objects of the current design is presented below:

```
jsonDesign:
machines:Array(6)
0:
        humans_min:"0.00"
        id:"324"
        input_time:"20"
        ins:Array(4)
                0:
                        a:"100"
                        color:"#f89807"
                        id:"1242"
                        row_num:"4"
                        storage_id:"678"
```

```
            1:
                    a:"10"
                    color:"#ff0000"
                    id:"1244"
                    row_num:"1"
                    storage_id:"675"
            2:
                    a:"10"
                    color:"#0080ff"
                    id:"1245"
                    row_num:"2"
                    storage_id:"676"
            3:
                    a:"1"
                    color:"#0000ff"
                    id:"1246"
                    row_num:"3"
                    storage_id:"677"
                    length:4
                    name:"Kneading Machine 1"
        outs:Array(1)
            0:
                    a:"12"
                    color:"#b509f7"
                    id:"1243"
                    out_state:"0"
                    prod_time:"20"
                    row_num:"1"
                    storage_id:"679"
                    length:1
            __proto__:Array(0)
        x:"620"
        y:"20"

1:
        humans_min:"0.00"
        id:"325"
        input_time:"3"
        ins:Array(4)
            0:
```

```
            a:"10"
            color:"#ff0000"
            id:"1247"
            row_num:"1"
            storage_id:"675"

        1:
            a:"10"
            color:"#0080ff"
            id:"1248"
            row_num:"2"
            storage_id:"676"

        2:
            a:"10"
            color:"#0000ff"
            id:"1250"
            row_num:"3"
            storage_id:"677"

        3:
            a:"10"
            color:"#f89807"
            id:"1251"
            row_num:"4"
            storage_id:"678"

        length:4

name:"Kneading Machine 2"
outs:Array(1)
        0:
        a:"12"
        color:"#b509f7"
        id:"1249"
        out_state:"512"
        prod_time:"10"
        row_num:"1"
        storage_id:"679"
```

```
            length:1

        x:"620"
        y:"190"

2:
        humans_min:"0.00"
        id:"326"
        input_time:"1"
        ins:Array(1)
                0:
                a:"20"
                color:"#00ffff"
                id:"1252"
                row_num:"1"
                storage_id:"680"
                length:1
        __proto__:Array(0)
        name:"wh-flour"
        outs:Array(1)
                0:
                a:"20"
                color:"#0080ff"
                id:"1253"
                out_state:"1"
                prod_time:"1"
                row_num:"1"
                storage_id:"676"
                length:1
        __proto__:Array(0)
        x:"230"
        y:"40"

3:
        humans_min:"0.00"
        id:"327"
        input_time:"1"
        ins:Array(1)
                0:
                a:"200"
```

```
            color:"#ff8000"
            id:"1255"
            row_num:"1"
            storage_id:"681"

            length:1
            __proto__:Array(0)
        name:"wh-salt"
        outs:Array(1)
            0:
            a:"200"
            color:"#f89807"
            id:"1254"
            out_state:"1"
            prod_time:"1"
            row_num:"1"
            storage_id:"678"

            length:1
        x:"230"
        y:"100"

4:
        humans_min:"0.00"
        id:"328"
        input_time:"1"
        ins:Array(1)
            0:
            a:"1"
            color:"#0000ff"
            id:"1256"
            row_num:"1"
            storage_id:"682"

        length:1
        __proto__:Array(0)
        name:"wh-yeast"
        outs:Array(1)
            0:
            a:"1"
```

```
            color:"#0000ff"
            id:"1257"
            out_state:"1"
            prod_time:"1"
            row_num:"1"
            storage_id:"677"

            length:1
            __proto__:Array(0)
        x:"230"
        y:"160"

5:
        humans_min:"0.00"
        id:"329"
        input_time:"1"
        ins:Array(1)
            0:
            a:"20"
            color:"#ff0000"
            id:"1258"
            row_num:"1"
            storage_id:"683"
            length:1
            __proto__:Array(0)

        length:1
        __proto__:Array(0)
        name:"wh-salt"
        outs:Array(1)
            0:
            a:"20"
            color:"#ff0000"
            id:"1259"
            out_state:"1"
            prod_time:"1"
            row_num:"1"
            storage_id:"675"

        length:1
```

```
        x:"230"
        y:"220"

        length:6
storages:Array(9)

0:
        a:"50"
        aMax:"100"
        color:"#ff0000"
        id:"675"
        name:"water"
        x:"390"
        y:"230"
        __proto__:Object
1:
        a:"50"
        aMax:"100"
        color:"#0080ff"
        id:"676"
        name:"flour"
        x:"390"
        y:"50"
        __proto__:Object
2:
        a:"50"
        aMax:"1000"
        color:"#0000ff"
        id:"677"
        name:"yeast"
        x:"390"
        y:"170"
        __proto__:Object
3:
        a:"50"
        aMax:"1000"
        color:"#f89807"
        id:"678"
        name:"salt"
        x:"390"
```

```
        y:"110"
        __proto__:Object
4:
        a:"60"
        aMax:"100"
        color:"#b509f7"
        id:"679"
        name:"dough"
        x:"810"
        y:"130"
        __proto__:Object
5:
        a:"1000"
        aMax:"1000"
        color:"#00ffff"
        id:"680"
        name:"flour"
        x:"100"
        y:"50"
        __proto__:Object
6:
        a:"10000"
        aMax:"10000"
        color:"#ff8000"
        id:"681"
        name:"salt"
        x:"100"
        y:"110"
        __proto__:Object
7:
        a:"100"
        aMax:"100"
        color:"#0000ff"
        id:"682"
        name:"yeast"
        x:"100"
        y:"170"
        __proto__:Object
8:
        a:"1000"
```

```
        aMax:"1000"
        color:"#ff0000"
        id:"683"
        name:"water"
        x:"100"
        y:"230"
        __proto__:Object
length:9



machSt:

        324:
                inputTime:"20"
                ins:
                1:
                a:"10"
                storageId:"675"
                __proto__:Object
                2:
                a:"10"
                storageId:"676"
                __proto__:Object
                3:
                a:"1"
                storageId:"677"
                __proto__:Object
                4:
                a:"100"
                storageId:"678"
                __proto__:Object
                outs:
                        1:
                        a:"12"
                        color:"#b509f7"
                        id:"1243"
                        prodTime:"20"
                        state:
```

```
                    0:{bit: ""}
                    1:{bit: "0", obj: pt}
                    2:{bit: "0", obj: pt}
                    3:{bit: "0", obj: pt}
                    4:{bit: "0", obj: pt}
                    5:{bit: "0", obj: pt}
                    6:{bit: "0", obj: pt}
                    7:{bit: "0", obj: pt}
                    8:{bit: "0", obj: pt}
                    9:{bit: "0", obj: pt}
                    10:{bit: "0", obj: pt}
                    11:{bit: "0", obj: pt}
                    12:{bit: "0", obj: pt}
                    13:{bit: "0", obj: pt}
                    14:{bit: "0", obj: pt}
                    15:{bit: "0", obj: pt}
                    16:{bit: "0", obj: pt}
                    17:{bit: "0", obj: pt}
                    18:{bit: "0", obj: pt}
                    19:{bit: "0", obj: pt}
                    20:{bit: "0", obj: pt}
            stateN:"0"
            storageId:"679"
            t2ni:"20"

    325:

            inputTime:"3"
            ins:
                    1:{storageId: "675", a: "10"}
                    2:{storageId: "676", a: "10"}
                    3:{storageId: "677", a: "10"}
                    4:{storageId: "678", a: "10"}
            outs:
                    1:
                            a:"12"
                            color:"#b509f7"
                            id:"1249"
                            prodTime:"10"
                            state:
                                    0:{bit: ""}
```

```
                        1:{bit: "1", obj: pt}
                        2:{bit: "0", obj: pt}
                        3:{bit: "0", obj: pt}
                        4:{bit: "0", obj: pt}
                        5:{bit: "0", obj: pt}
                        6:{bit: "0", obj: pt}
                        7:{bit: "0", obj: pt}
                        8:{bit: "0", obj: pt}
                        9:{bit: "0", obj: pt}
                        10:{bit: "0", obj: pt}
                stateN:"512"
                storageId:"679"
        t2ni:"3"

326:

        inputTime:"1"

        ins:
                1:{storageId: "680", a: "20"}
        outs:
                1:
                a:"20"
                color:"#0080ff"
                id:"1253"
                prodTime:"1"
                state:{0: {…}, 1: {…}}
                stateN:"1"
                storageId:"676"
        t2ni:"1"

327:

        inputTime:"1"
        ins:
                1:{storageId: "681", a: "200"}
        outs:
                1:
                a:"200"
                color:"#f89807"
                id:"1254"
```

```
                prodTime:"1"
                state:{0: {…}, 1: {…}}
                stateN:"1"
                storageId:"678"
        t2ni:"1"

328:

        inputTime:"1"
        ins:
                1:{storageId: "682", a: "1"}
        outs:
                1:
                a:"1"
                color:"#0000ff"
                id:"1257"
                prodTime:"1"
                state:{0: {…}, 1: {…}}
                stateN:"1"
                storageId:"677"
        t2ni:"1"

329:

        inputTime:"1"
        ins:
                1:{storageId: "683", a: "20"}
        outs:
                1:
                a:"20"
                color:"#ff0000"
                id:"1259"
                prodTime:"1"
                state:
                0:{bit: ""}
                1:{bit: "1", obj: pt}
                __proto__:Object
                stateN:"1"
                storageId:"675"
        t2ni:"1"
machStorEdit:"no"
```

```
storA:
        675:
                a:"50"
                aMax:"100"
                bars:
                        1:
                                bit:"1"
                                obj:pt
                                        _groups:[Array(1)]
                                        _parents:[html.layout-pf.layout-pf-
                                        fixed.transitions]
                                        __proto__:Object
                                __proto__:Object
                        2:{obj: pt, bit: "1"}
                        3:{obj: pt, bit: "1"}
                        4:{obj: pt, bit: "1"}
                        5:{obj: pt, bit: "1"}
                        6:{obj: pt, bit: "1"}
                        7:{obj: pt, bit: "1"}
                        8:{obj: pt, bit: "1"}
                        9:{obj: pt, bit: "1"}
                        10:{obj: pt, bit: "1"}
                        11:{obj: pt, bit: "0"}
                        12:{obj: pt, bit: "0"}
                        13:{obj: pt, bit: "0"}
                        14:{obj: pt, bit: "0"}
                        15:{obj: pt, bit: "0"}
                        16:{obj: pt, bit: "0"}
                        17:{obj: pt, bit: "0"}
                        18:{obj: pt, bit: "0"}
                        19:{obj: pt, bit: "0"}
                        20:{obj: pt, bit: "0"}
                txtobj:pt {_groups: Array(1), _parents: Array(1)}

        676:
                a:"50"
                aMax:"100"
                bars:
                        1:{obj: pt, bit: "1"}
```

```
            2:{obj: pt, bit: "1"}
            3:{obj: pt, bit: "1"}
            4:{obj: pt, bit: "1"}
            5:{obj: pt, bit: "1"}
            6:{obj: pt, bit: "1"}
            7:{obj: pt, bit: "1"}
            8:{obj: pt, bit: "1"}
            9:{obj: pt, bit: "1"}
            10:{obj: pt, bit: "1"}
            11:{obj: pt, bit: "0"}
            12:{obj: pt, bit: "0"}
            13:{obj: pt, bit: "0"}
            14:{obj: pt, bit: "0"}
            15:{obj: pt, bit: "0"}
            16:{obj: pt, bit: "0"}
            17:{obj: pt, bit: "0"}
            18:{obj: pt, bit: "0"}
            19:{obj: pt, bit: "0"}
            20:{obj: pt, bit: "0"}
            __proto__:Object
        txtobj:pt {_groups: Array(1), _parents: Array(1)}

677:
        a:"50"
        aMax:"1000"
        bars:
            1:{obj: pt, bit: "1"}
            2:{obj: pt, bit: "0"}
            3:{obj: pt, bit: "0"}
            4:{obj: pt, bit: "0"}
            5:{obj: pt, bit: "0"}
            6:{obj: pt, bit: "0"}
            7:{obj: pt, bit: "0"}
            8:{obj: pt, bit: "0"}
            9:{obj: pt, bit: "0"}
            10:{obj: pt, bit: "0"}
            11:{obj: pt, bit: "0"}
            12:{obj: pt, bit: "0"}
            13:{obj: pt, bit: "0"}
            14:{obj: pt, bit: "0"}
```

```
            15:{obj: pt, bit: "0"}
            16:{obj: pt, bit: "0"}
            17:{obj: pt, bit: "0"}
            18:{obj: pt, bit: "0"}
            19:{obj: pt, bit: "0"}
            20:{obj: pt, bit: "0"}
            __proto__:Object
        txtobj:pt {_groups: Array(1), _parents: Array(1)}

678:
        a:"50"
        aMax:"1000"
        bars:
            1:{obj: pt, bit: "1"}
            2:{obj: pt, bit: "0"}
            3:{obj: pt, bit: "0"}
            4:{obj: pt, bit: "0"}
            5:{obj: pt, bit: "0"}
            6:{obj: pt, bit: "0"}
            7:{obj: pt, bit: "0"}
            8:{obj: pt, bit: "0"}
            9:{obj: pt, bit: "0"}
            10:{obj: pt, bit: "0"}
            11:{obj: pt, bit: "0"}
            12:{obj: pt, bit: "0"}
            13:{obj: pt, bit: "0"}
            14:{obj: pt, bit: "0"}
            15:{obj: pt, bit: "0"}
            16:{obj: pt, bit: "0"}
            17:{obj: pt, bit: "0"}
            18:{obj: pt, bit: "0"}
            19:{obj: pt, bit: "0"}
            20:{obj: pt, bit: "0"}
            __proto__:Object
        txtobj:pt {_groups: Array(1), _parents: Array(1)}

679:
        a:"60"
        aMax:"100"
        bars:
            1:{obj: pt, bit: "1"}
```

```
            2:{obj: pt, bit: "1"}
            3:{obj: pt, bit: "1"}
            4:{obj: pt, bit: "1"}
            5:{obj: pt, bit: "1"}
            6:{obj: pt, bit: "1"}
            7:{obj: pt, bit: "1"}
            8:{obj: pt, bit: "1"}
            9:{obj: pt, bit: "1"}
            10:{obj: pt, bit: "1"}
            11:{obj: pt, bit: "1"}
            12:{obj: pt, bit: "1"}
            13:{obj: pt, bit: "0"}
            14:{obj: pt, bit: "0"}
            15:{obj: pt, bit: "0"}
            16:{obj: pt, bit: "0"}
            17:{obj: pt, bit: "0"}
            18:{obj: pt, bit: "0"}
            19:{obj: pt, bit: "0"}
            20:{obj: pt, bit: "0"}
            __proto__:Object

        txtobj:pt {_groups: Array(1), _parents: Array(1)}
680:
        a:"1000"
        aMax:"1000"
        bars:
            1:{obj: pt, bit: "1"}
            2:{obj: pt, bit: "1"}
            3:{obj: pt, bit: "1"}
            4:{obj: pt, bit: "1"}
            5:{obj: pt, bit: "1"}
            6:{obj: pt, bit: "1"}
            7:{obj: pt, bit: "1"}
            8:{obj: pt, bit: "1"}
            9:{obj: pt, bit: "1"}
            10:{obj: pt, bit: "1"}
            11:{obj: pt, bit: "1"}
            12:{obj: pt, bit: "1"}
            13:{obj: pt, bit: "1"}
            14:{obj: pt, bit: "1"}
```

```
            15:{obj: pt, bit: "1"}
            16:{obj: pt, bit: "1"}
            17:{obj: pt, bit: "1"}
            18:{obj: pt, bit: "1"}
            19:{obj: pt, bit: "1"}
            20:{obj: pt, bit: "1"}
            __proto__:Object
        txtobj:pt {_groups: Array(1), _parents: Array(1)}

681:
        a:"10000"
        aMax:"10000"
        bars:
            1:{obj: pt, bit: "1"}
            2:{obj: pt, bit: "1"}
            3:{obj: pt, bit: "1"}
            4:{obj: pt, bit: "1"}
            5:{obj: pt, bit: "1"}
            6:{obj: pt, bit: "1"}
            7:{obj: pt, bit: "1"}
            8:{obj: pt, bit: "1"}
            9:{obj: pt, bit: "1"}
            10:{obj: pt, bit: "1"}
            11:{obj: pt, bit: "1"}
            12:{obj: pt, bit: "1"}
            13:{obj: pt, bit: "1"}
            14:{obj: pt, bit: "1"}
            15:{obj: pt, bit: "1"}
            16:{obj: pt, bit: "1"}
            17:{obj: pt, bit: "1"}
            18:{obj: pt, bit: "1"}
            19:{obj: pt, bit: "1"}
            20:{obj: pt, bit: "1"}
            __proto__:Object
        txtobj:pt {_groups: Array(1), _parents: Array(1)}

682:
        a:"100"
        aMax:"100"
        bars:
            1:{obj: pt, bit: "1"}
```

```
            2:{obj: pt, bit: "1"}
            3:{obj: pt, bit: "1"}
            4:{obj: pt, bit: "1"}
            5:{obj: pt, bit: "1"}
            6:{obj: pt, bit: "1"}
            7:{obj: pt, bit: "1"}
            8:{obj: pt, bit: "1"}
            9:{obj: pt, bit: "1"}
            10:{obj: pt, bit: "1"}
            11:{obj: pt, bit: "1"}
            12:{obj: pt, bit: "1"}
            13:{obj: pt, bit: "1"}
            14:{obj: pt, bit: "1"}
            15:{obj: pt, bit: "1"}
            16:{obj: pt, bit: "1"}
            17:{obj: pt, bit: "1"}
            18:{obj: pt, bit: "1"}
            19:{obj: pt, bit: "1"}
            20:{obj: pt, bit: "1"}
            __proto__:Object
        txtobj:pt {_groups: Array(1), _parents: Array(1)}

683:
        a:"1000"
        aMax:"1000"
        bars:
            1:{obj: pt, bit: "1"}
            2:{obj: pt, bit: "1"}
            3:{obj: pt, bit: "1"}
            4:{obj: pt, bit: "1"}
            5:{obj: pt, bit: "1"}
            6:{obj: pt, bit: "1"}
            7:{obj: pt, bit: "1"}
            8:{obj: pt, bit: "1"}
            9:{obj: pt, bit: "1"}
            10:{obj: pt, bit: "1"}
            11:{obj: pt, bit: "1"}
            12:{obj: pt, bit: "1"}
            13:{obj: pt, bit: "1"}
            14:{obj: pt, bit: "1"}
```

```
        15:{obj: pt, bit: "1"}
        16:{obj: pt, bit: "1"}
        17:{obj: pt, bit: "1"}
        18:{obj: pt, bit: "1"}
        19:{obj: pt, bit: "1"}
        20:{obj: pt, bit: "1"}
        __proto__:Object
txtobj:pt {_groups: Array(1), _parents: Array(1)}
```

4. User Interface

The INNOVET Business Simulation User Interface has been designed with the intention of functionality. The platform's User Interface (basic screens) is presented below.

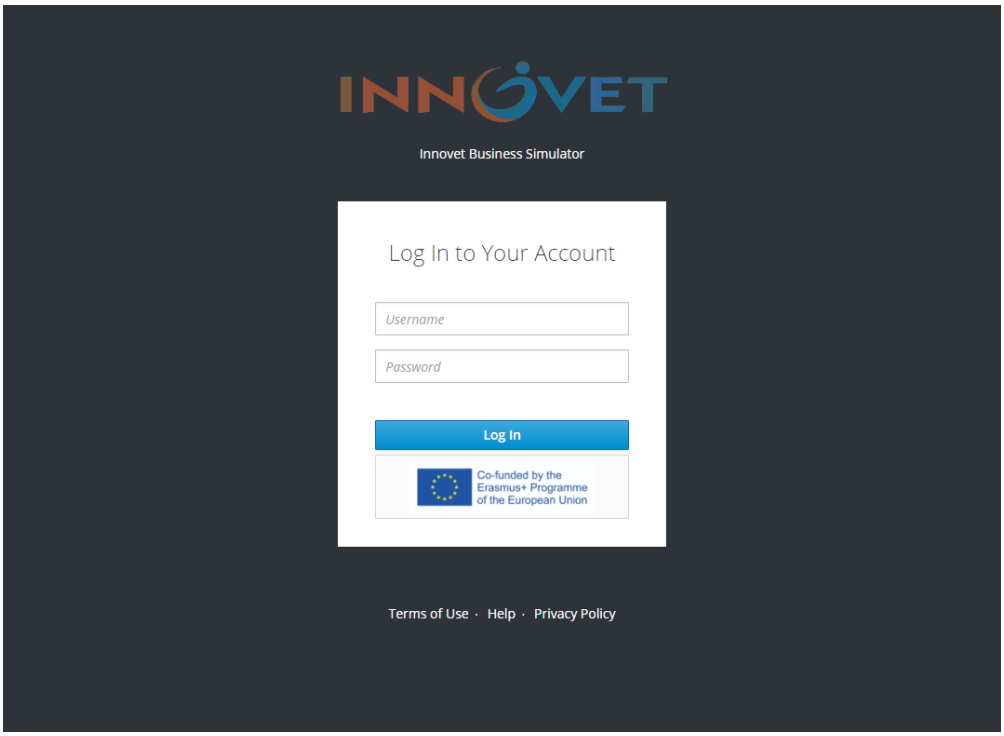

Figure 13: Login Page

Figure 13 presents the login page of INNOVET Business Simulator. Figure 14 presents the administrator's home page with a link to four subsystems. Users have access only to three subsystems as shown in Figure 15. In the following selected Figures of the basic functions implemented in the User Interface will be presented. The Figures regard the Design, Simulation and Results modes of the application.

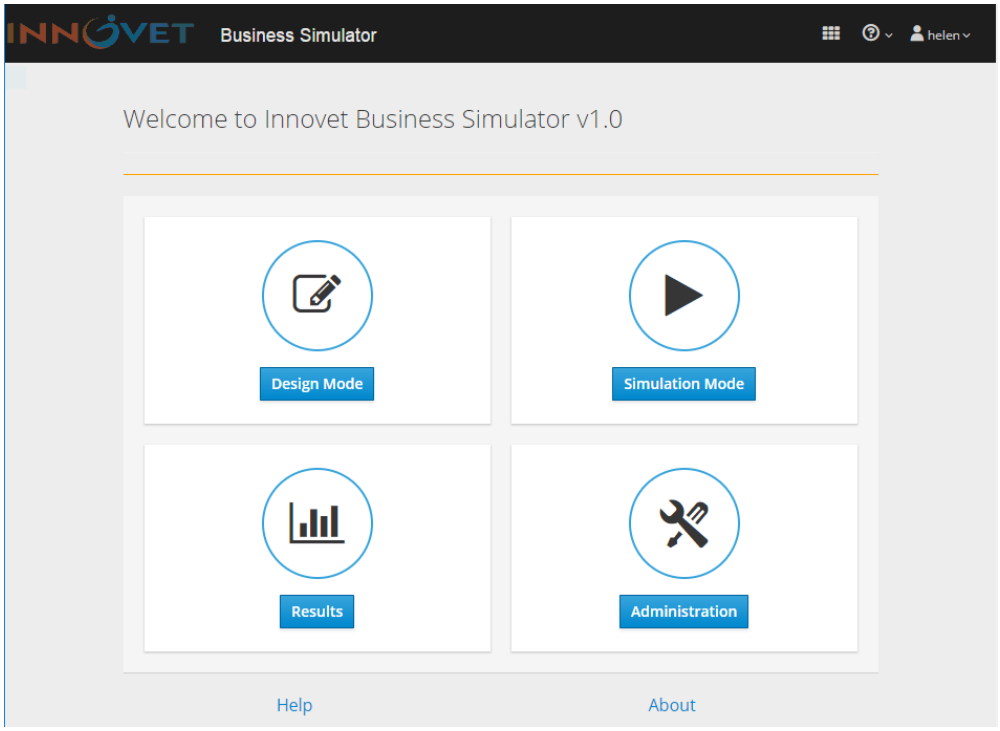

Figure 14: Administrators Home page

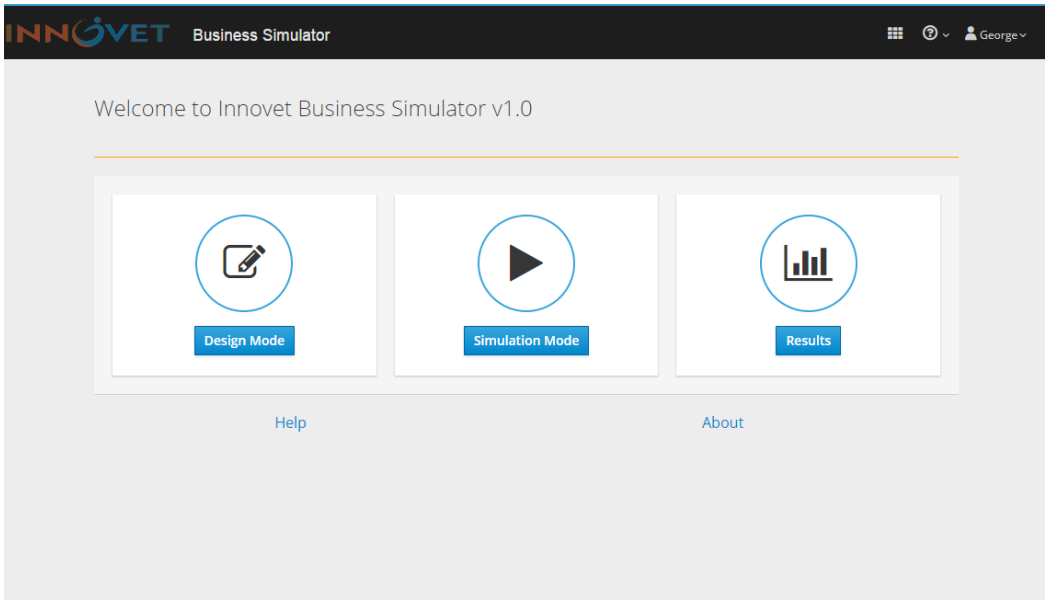

Figure 15: User Home Page

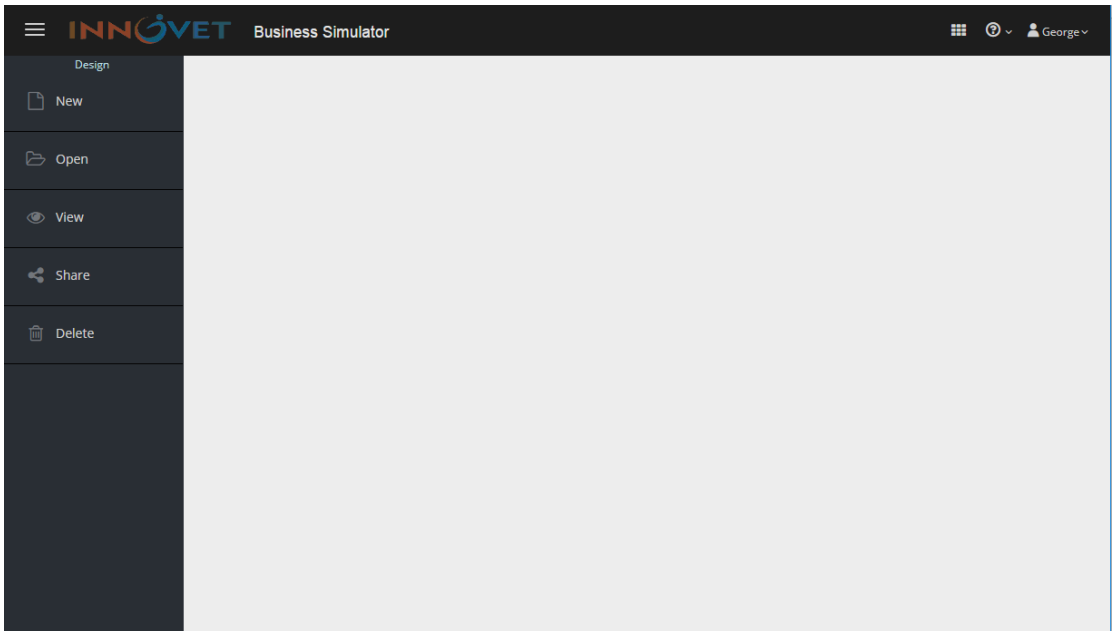

Figure 16: Design Mode User interface

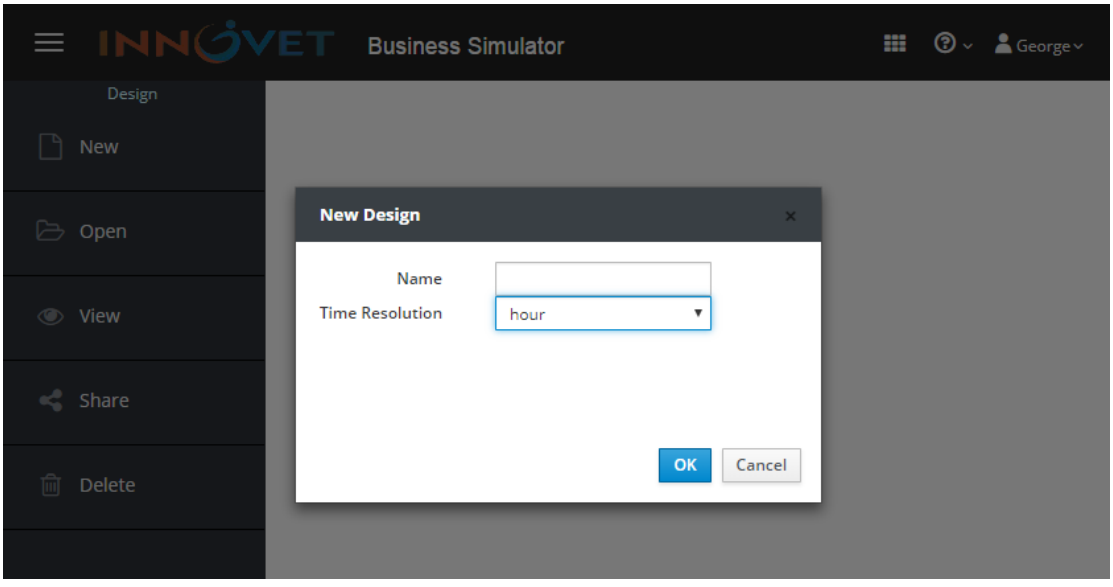

Figure 17: New design creation

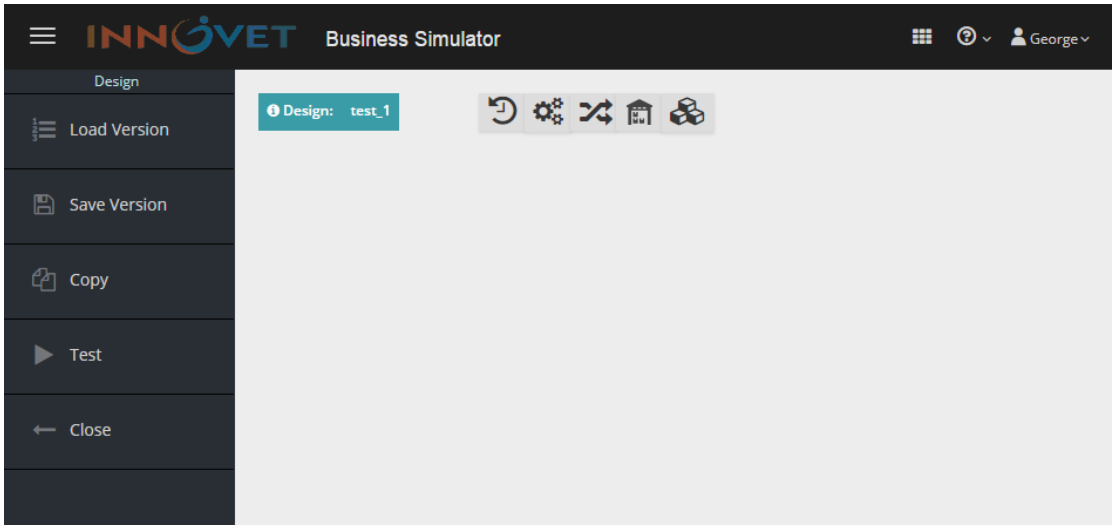

Figure 18: User interface after new design creation

Icon Toolbar	Name	Description
	Time resolution	Change the design's time unit
	Machine	Add a new machine
	Carrier	Add a new carrier
	Warehouse	Add a new warehouse
	Product	Add a new product

Table 3: Design Mode Toolbar

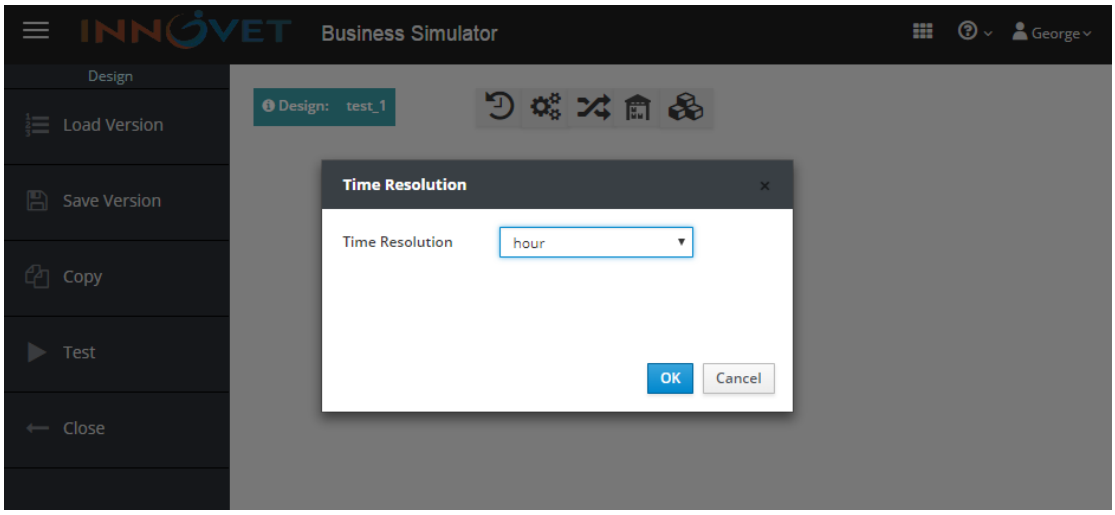

Figure 19: Setting design's time resolution

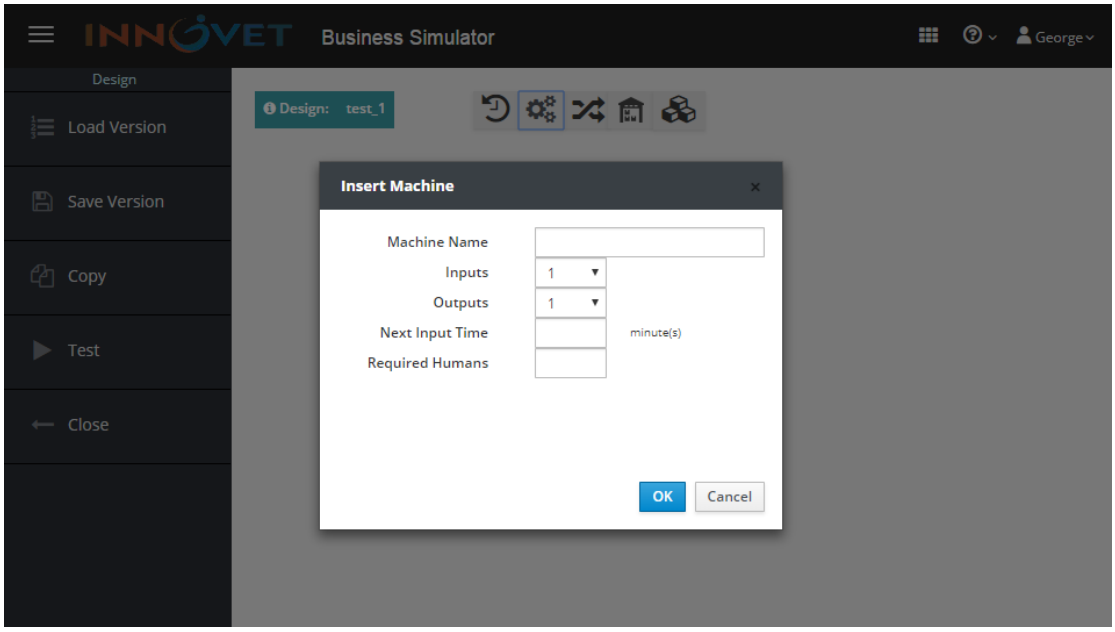

Figure 20: Insert new machine and set its parameters

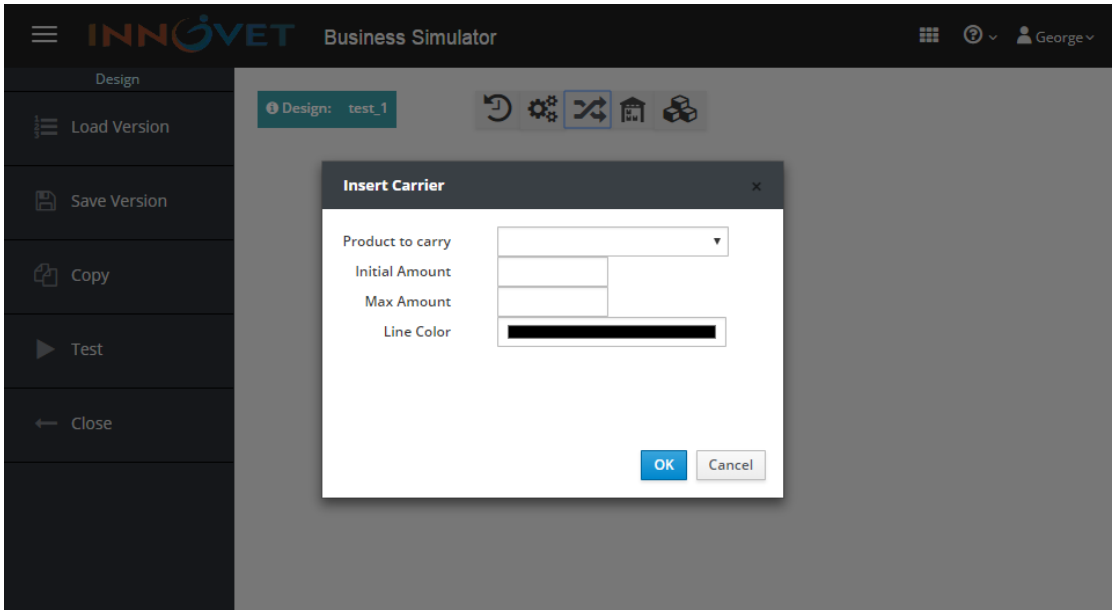

Figure 21: Create a new carrier and set its parameters

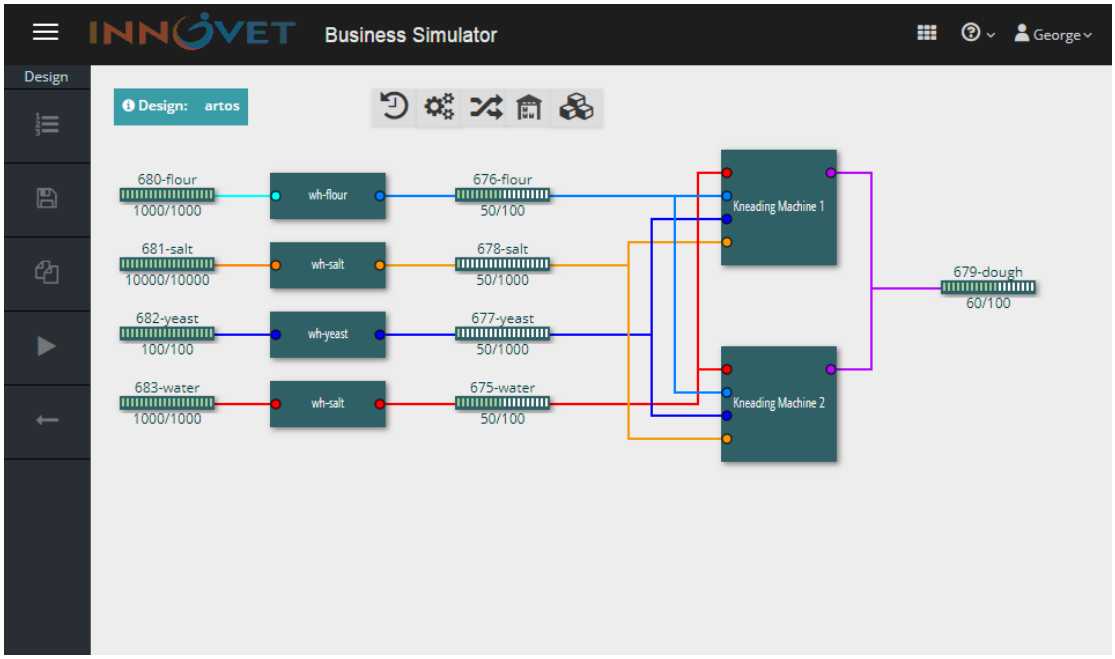

Figure 22: An example of a Design in Design Mode

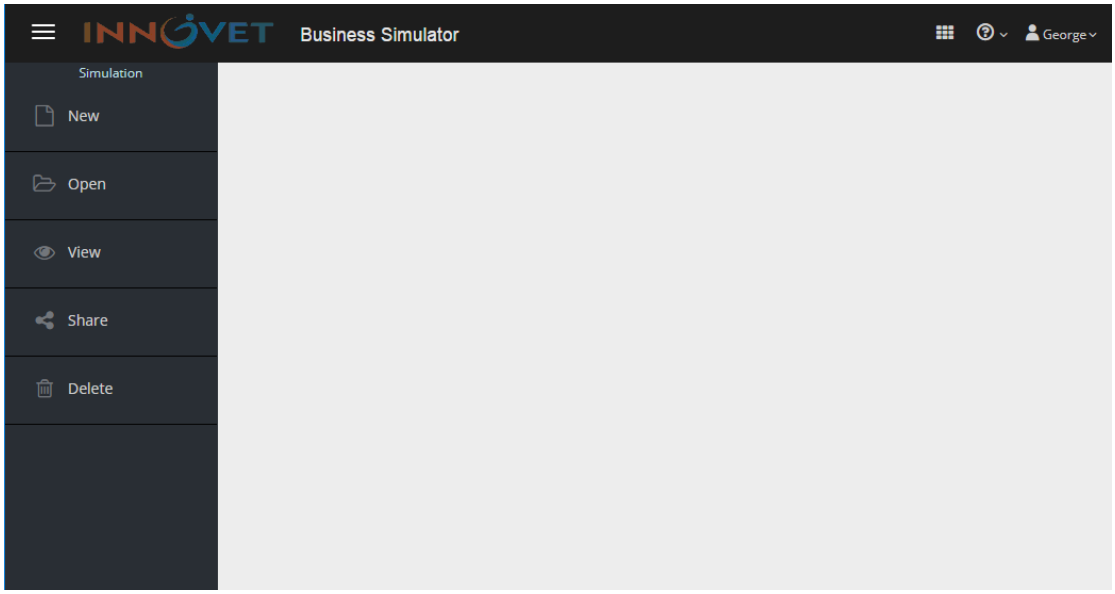

Figure 23: Simulation Mode

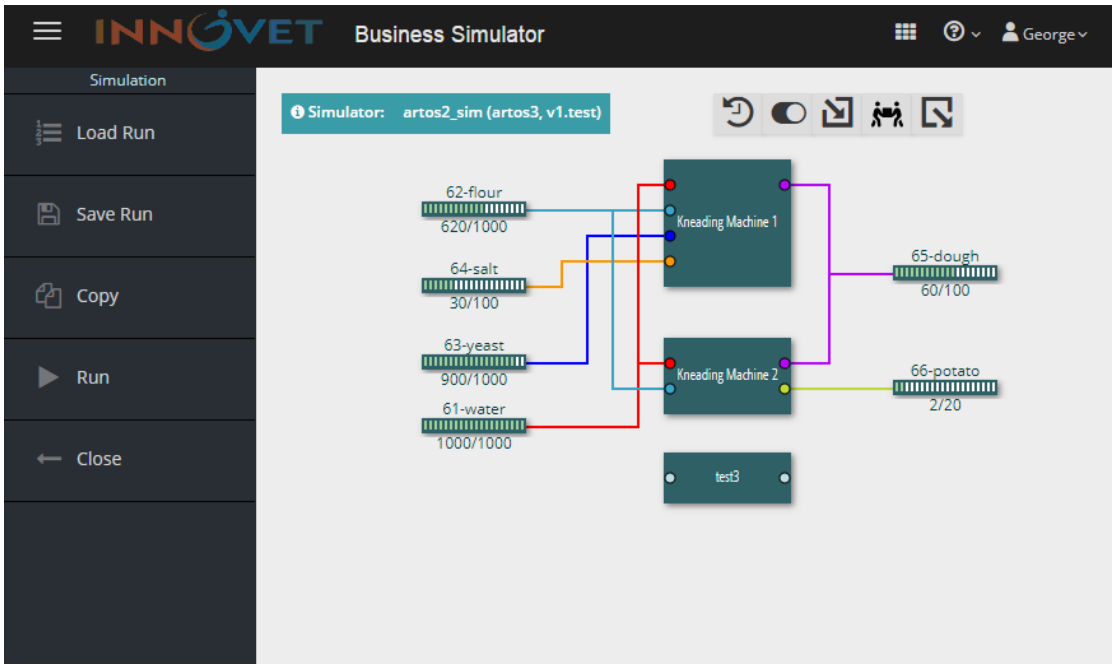

Figure 24: New Simulator for a selected design

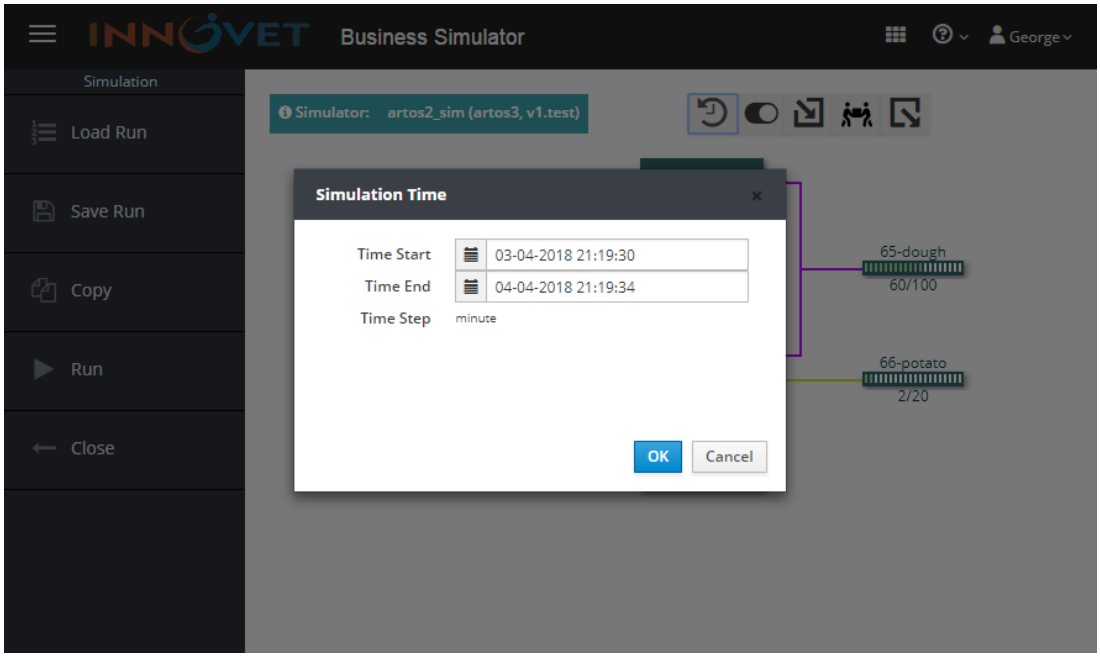

Figure 25: Set simulation time

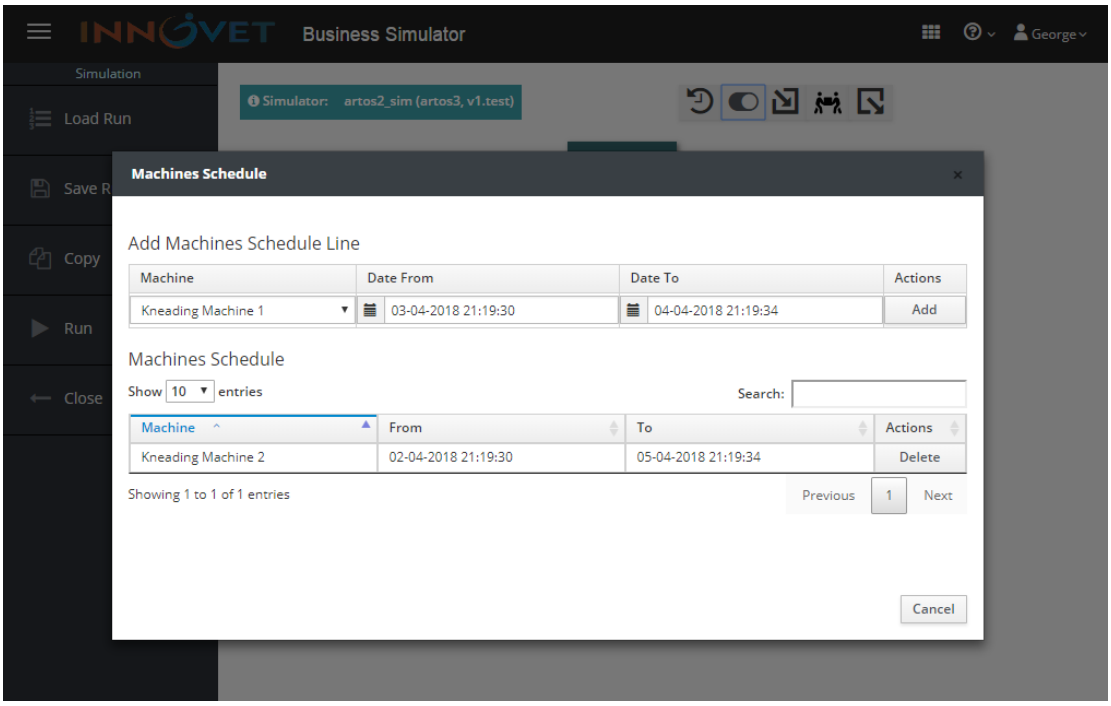

Figure 26: Set machines Schedule

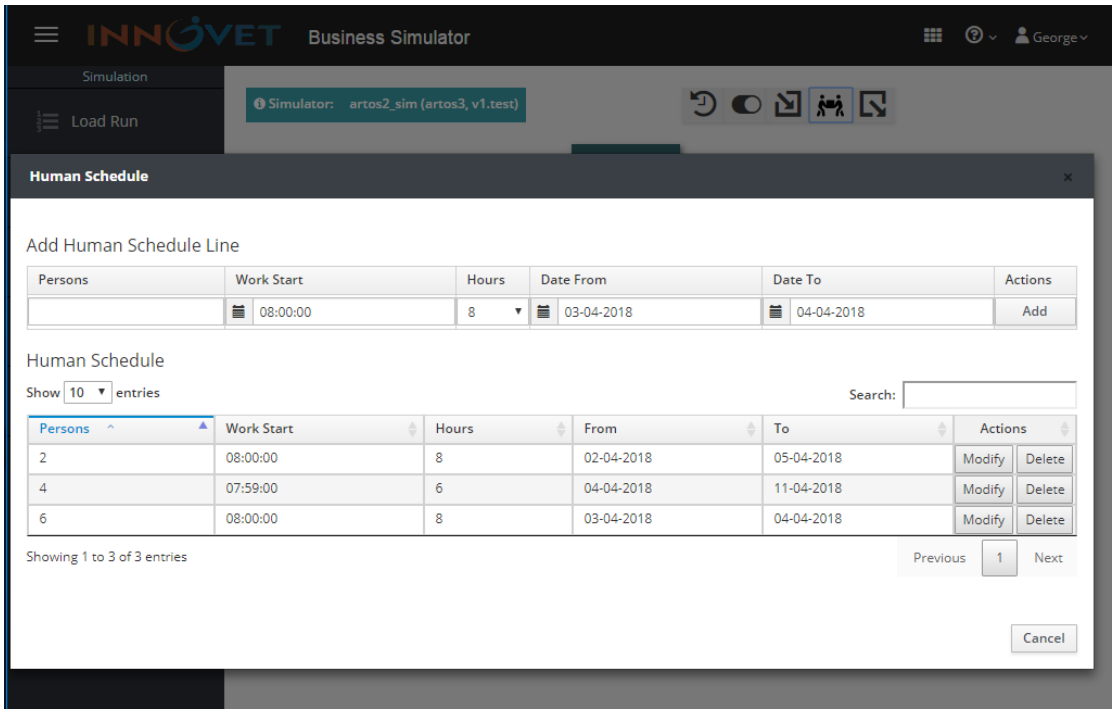

Figure 27: Set Human Schedule

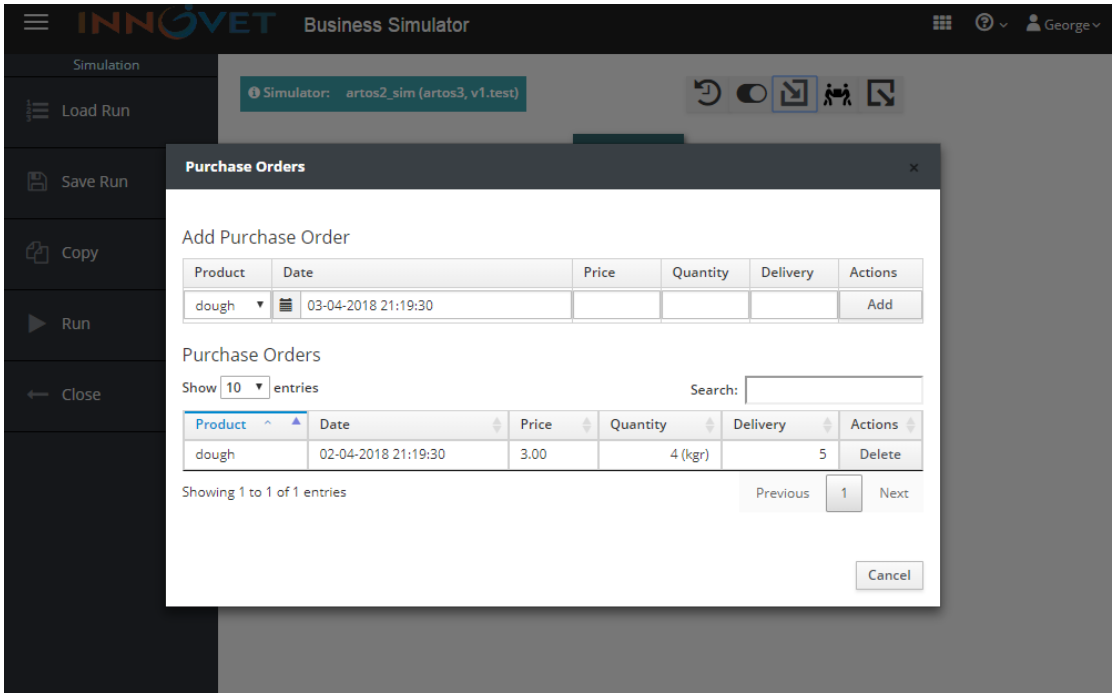

Figure 28: Set Purchase orders

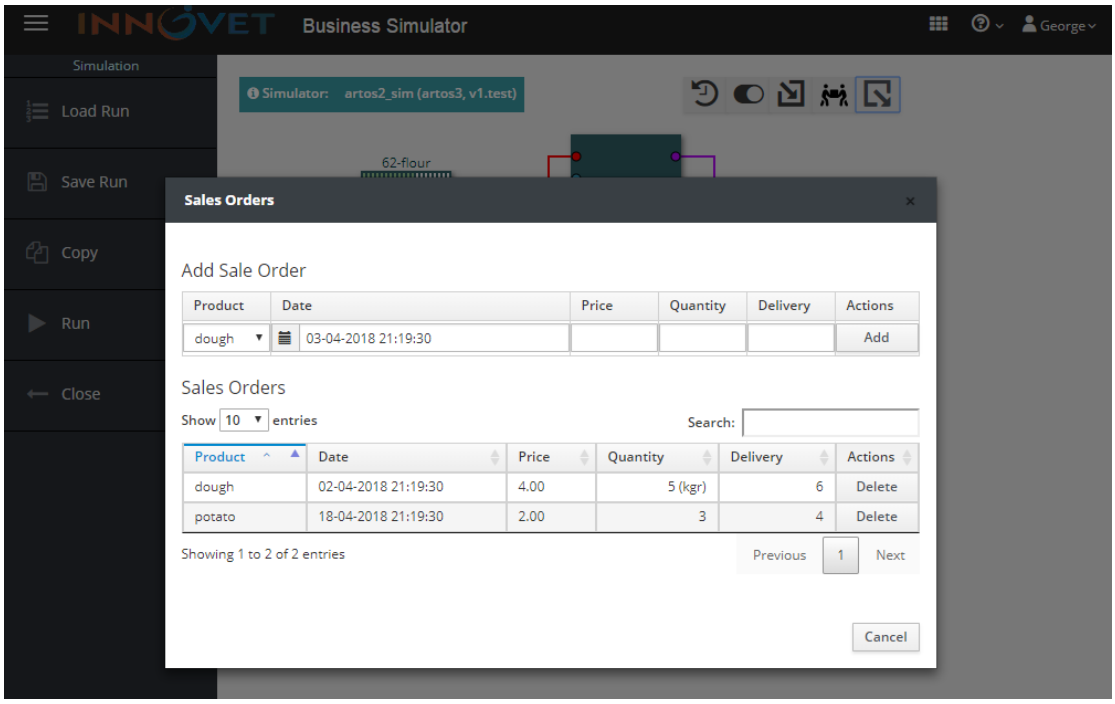

Figure 29: Set Sales Orders

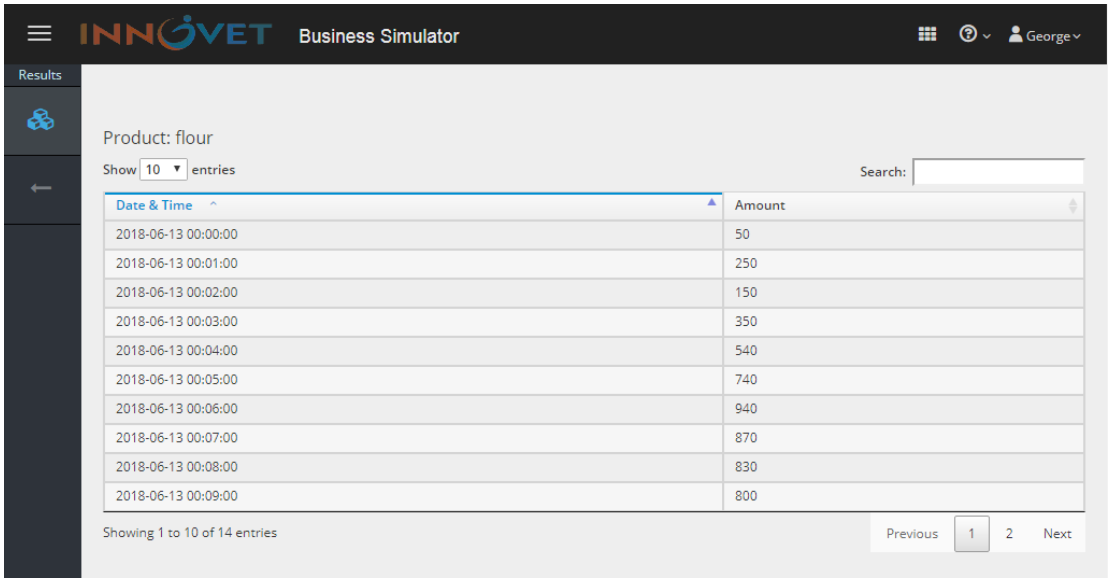

Figure 30: Example of Results in Table format

Co-funded by the Erasmus+ Programme of the European Union

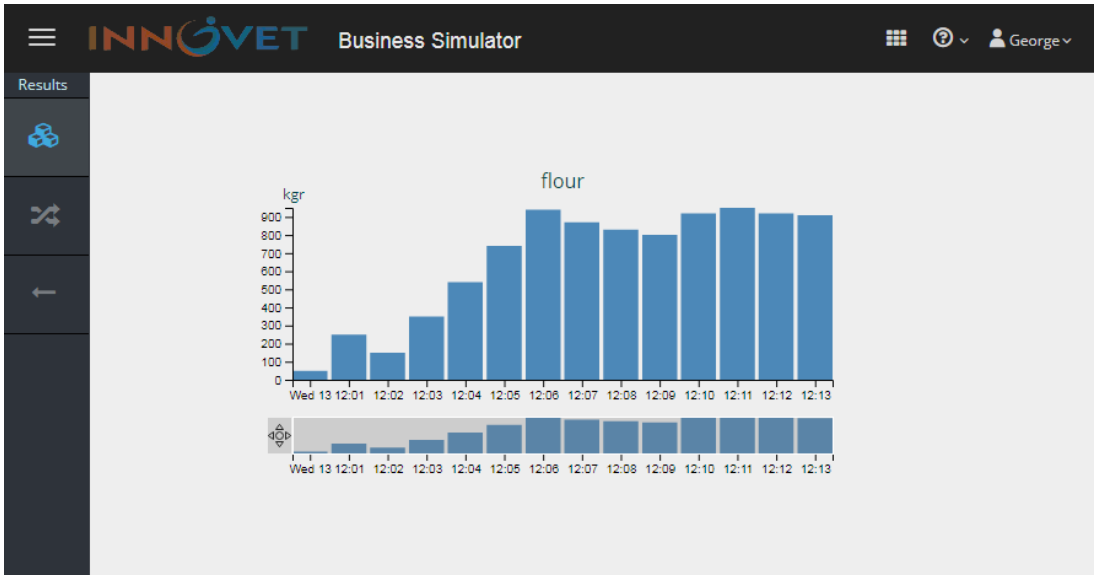

Figure 31: Example of Results in Graph format

5. Appendix A: Database structure

Table structure for table user

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
uname	varchar(30)	No		Index	
passwd	varchar(30)	No			
name	varchar(100)	Yes	NULL		
cat	tinyint(3)	No			

Columns description

- id: User id
- uname: Username
- passwd: User Password
- name: User Full Name
- cat: User category
  - 0: application administrator
  - 1: application user

Table structure for table user_permissions

Column	Type	Null	Default	Key	Extra
des_id	int(10)	No		Primary	
user_id	int(10)	No		Primary	
permission	enum('view', 'test', 'copy', 'full', 'owner')	No			

Columns description

- des_id: Design id from table design
- user_id: User id from table user
- permission: User permission for a design
  - view: User can only see the design
  - test: The user can perform a functional test in the design
  - copy: user can create a copy of the design
  - full: full access to the design. The user can make changes to the design but can not delete the design
  - owner: Creator and owner of a design, have full access to the design and he can delete it.

Table structure for table design

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
name	varchar(100)	No			
date_start	datetime	No	CURRENT_TIMESTAMP		
time_resolution	int(10)	No			
active	tinyint(1)	No	1		

Columns description

- id: Design id
- name: Design name (user defined)
- date_start: Creation Date of the design
- time_resolution: Design Time unit (user defined)
- active: Current status of design
  - 0: The design is inactive. The design has been deleted by its owner but it

exists in the database. Only the application administrator can remove it from the database.

- 1: The design is active. The default status of a design, the owner has full access.

Table structure for table version

Column	Type	Null	Default	Key	Extra
id	int(11)	No		Primary	AUTO_INCREMENT
number	int(10)	No			
description	varchar(100)	Yes	NULL		
date_start	datetime	No	CURRENT_TIMESTAMP		
time_resolution	int(10)	No			
active	tinyint(1)	No	1		
des_id	int(10)	No			

Columns description

- id: Design version id
- number: Design version number
- description: Short description of design version (user defined)
- date_start: Design version creation date
- time_resolution: Design version time unit (user defined)
- active: Current status of design version
  - 0: The design version is inactive. The design version has been deleted by its owner but it exists in the database. Only the application administrator can remove it from the database.
  - 1: The design version is active. The default status of a design version
- des_id: Design id from table design

Table structure for table machine

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
des_id	int(10)	No		Index	
x	smallint(6)	No			
y	smallint(6)	No			
name	varchar(50)	No			
input_time	smallint(6)	No	1		
humans_min	decimal(10,2)	No	0.00		

Columns description

- id: Machine id
- des_id: The design id in which the machine belongs
- x: x coordinates that determine the position of the machine on the design view
- y: y coordinates that determine the position of the machine on the design view
- name: Machine name (user defined)
- input time: the minimum input time units for two successive loadings at the machine inputs
- humans_min: Minimum number of humans for the machine to operate

Table structure for table machine_io

It contains one record for every input and output of each machine

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
mach_id	int(10)	No		index	
row_num	tinyint(3)	No			
in_out	enum('in',	No			

	'out')				
storage_id	int(10)	No		index	
a	int(10)	No			
prod_time	smallint(5)	No			
out_state	bigint(20)	No			

Columns description

- id: table machine_io id
- mach_id: The machine id in which the input or the output belongs
- row_num: Order of input or output (only for the design view)
- in_out: Input or Output
  - in: Input
  - out: Output
- storage_id: The carrier id is connected with the input or the output
- a: amount of products in the input or output
- prod_time: Production time. How many time units needed for the production of a product (Only for output)
- out_state: State of a machine. Related to production of a product

Table structure for table machine_io_versions

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
mach_id	int(10)	No		index	
row_num	tinyint(3)	No			
in_out	enum('in', 'out')	No			
storage_id	int(10)	No		index	
a	int(10)	No			
prod_time	smallint(5)	No			
out_state	bigint(20)	No			

Columns description

- id: Table machine_io_version id (for the design version)
- mach_id: The machine id in which the input or the output belongs
- row_num: Order of input or output (only for the design view)
- in_out: Input or Output
  - in: Input
  - out: Output
- storage_id: The carrier id is connected with the input or the output
- a: Amount of products in the input or output
- prod_time: Production time. How many time units needed for the production of a product (Only for output)
- out_state: State of a machine. Related to production of a product

Table structure for table machine_versions

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	
ver_id	int(10)	No		index	
x	smallint(6)	No			
y	smallint(6)	No			
name	varchar(50)	No			
input_time	smallint(6)	No	1		
humans_min	decimal(10,2)	No	0.00		

Columns description

- id: Machine id for the design version
- ver_id: The design version id in which the machine belongs
- x: x coordinates that determine the position of the machine on the design
- y: y coordinates that determine the position of the machine on the design

- name: Machine name (user defined)
- input time: the minimum input time units for two successive loadings at the machine inputs
- humans_min: Minimum number of humans for the machine to operate

Table structure for table product

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
name	varchar(50)	No		index	
unit	varchar(30)	No			
des_id	int(10)	No			

Columns description

- id: Product id
- name: Name of the product
- unit: Metric unit of a product
- des_id: Design id in which the product belongs

Table structure for table product_versions

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
name	varchar(50)	No			
unit	varchar(30)	No			
ver_id	int(10)	No			

Columns description

- id: Product id
- name: Name of the product

- unit: Metric unit of a product
- ver_id: Design version id in which the product belongs

Table structure for table simulation

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
name	varchar(100)	No			
date_create	datetime	No	CURRENT_TIMESTAMP		
active	tinyint(1)	No	1		
ver_id	int(10)	No			
timestart	timestamp	No	0000-00-00 00:00:00		
timeend	timestamp	No	0000-00-00 00:00:00		

Columns description

- id: Simulation id
- name: Simulation name
- date_create: date of simulation creation
- active: Current simulation status
  - 0: The simulation is inactive. The simulation has been deleted by its owner but it exists in the database. Only the application administrator can remove it from the database.
  - 1: The simulation is active. The default status of a simulation, the owner has full access.
- ver_id: The design version id being simulated
- timestart: Start time and date of the simulation
- timeend: End time and date of the simulation

Table structure for table sim_human_schedule

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
sim_id	int(10)	No			
persons	smallint(5)	No			
time_start	time	No			
hours	smallint(5)	No			
date_from	date	No			
date_to	date	No			

Columns description

- id: table sim_human_schedule id
- sim_id: Simulation id in witch the human schedule belongs
- persons: The number of people in the human schedule
- time_start: Working starting time (start of working shift)
- hours: Working hours
- date_from: Start date of the working schedule
- date_to: End date of the working schedule

Table structure for table sim_machines_schedule

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
sim_id	int(10)	No			
mach_ver_id	int(10)	No			
date_from	timestamp	No	0000-00-00 00:00:00		
date_to	timestamp	No	0000-00-00 00:00:00		

Columns description

- id: Product id
- sim_id: Simulation id
- mach_ver_id: Machine id for table machine_versions
- date_from: Start date of the machine
- date_to: End date of machine

Table structure for table sim_machine_log

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
sim_id	int(10)	No		Index	
timeslot	int(10)	No		Index	
mach_io_id	int(10)	No			
mach_state	bigint(20)	No			
t2ni	int(10)	No			

Columns description

- id: test_machine_log id
- sim_id: Simulation id
- timeslot: Timer value
- mach_io_id: Machine input/output id
- mach_state: Machine state
- t2ni: Time to next input

Table structure for table sim_prices

Column	Type	Null	Default	Key	Extra
id	int(11)	No		Primary	AUTO_INCREMENT
sim_id	int(11)	No			
product_id	int(11)	No			
date_from	timestamp	No	0000-00-00		

			00:00:00		
date_to	timestamp	No	0000-00-00 00:00:00		
price	decimal(10,2)	No			
quantity	int(11)	No			
delivery	int(11)	No			
type	enum('sell','buy')	No			

Columns description

- id: Id for table sim_prices
- sim_id: Simulation id
- product_id: Product id
- date_from: starting date when the order is valid
- date_to: ending date when the order is valid
- price: the price of a raw material or a product to be purchased or selled
- quantity: Purchase/ sell quantity
- delivery: Delivery time (timesteps)
- type: Type of transaction (purchase or sale)

Table structure for table sim_ps_orders

Column	Type	Null	Default	Key	Extra
id	int(11)	No		Primary	AUTO_INCREMENT
sim_id	int(11)	No			
product_id	int(11)	No			
date	timestamp	No	0000-00-00 00:00:00		
price	decimal(10,2)	No			
quantity	int(11)	No			

delivery	int(11)	No			
type	enum('purchase', 'sale')	No			

Columns description

- id: Purchase/ Sale order id
- sim_id: Simulation id
- product_id: Product id
- date: Purchase date of raw material or sale date of a product
- price: Purchase/ Sale price
- quantity: Quantity of raw material or product
- delivery: Delivery time (timesteps)
- type: Type of transaction (purchase or sale)

Table structure for table sim_user_permissions

Column	Type	Null	Default	Key	Extra
sim_id	int(10)	No		Primary	
user_id	int(10)	No		Primary	
permission	enum('view', 'run', 'order', 'copy', 'full', 'owner')	No			

Columns description

- id: Simulation id
- user_id: user id
- permission: User permission for a simulation
  - view: User can only see the simulation
  - run: The user can perform a simulation for the specific design
  - order: User has permission to add or change the purchase/ sale orders,

the human schedule and the machine schedule

- copy: user can create a copy of the simulation
- full: full access to the simulation. The user can make changes to the simulation but can not delete it
- owner: Creator and owner of a simulation, have full access to the simulation and he can delete it

Table structure for table sim_version

Column	Type	Null	Default	Key	Extra
id	int(11)	No		Primary	AUTO_INCREMENT
number	int(10)	No			
description	varchar(100)	Yes	NULL		
date_create	datetime	No	CURRENT_TIMESTAMP		
active	tinyint(1)	No	1		
sim_id	int(10)	No			
timestart	timestamp	No	0000-00-00 00:00:00		
timeend	timestamp	No	0000-00-00 00:00:00		

Columns description

- id: Simulation version id
- number: Simulation version number
- description: Simulation version description
- date_create: date of simulation creation
- active: Current simulation status
  - 0: The simulation version is inactive. The simulation has been deleted by its owner but it exists in the database. Only the application administrator can remove it from the database.

- 1: The simulation version is active. The default status of a simulation, the owner has full access.

- sim_id: Simulation id (table simulation)
- timestart: Start time and date of the simulation version
- timeend: End time and date of the simulation version

Table structure for table storage

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
product_id	int(10)	No		Index	
a	int(10)	No			
aMax	int(10)	No			
x	smallint(6)	No			
y	smallint(6)	No			
color	varchar(40)	No			
des_id	int(10)	No			

Columns description

- id: Storage of a carrier id
- product_id: Product id in the storage
- a: current amount of the product in the storage
- aMax: the maximum amount of the product in the storage
- x: x coordinates that determine the position of the storage on the design view
- y: y coordinates that determine the position of the storage on the design view
- color: color of the connecting line on the design view thats connect the storage with the other elements on the design
- des_id: Design id in which the storage of the carrier belongs

Table structure for table storage_versions

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
product_id	int(10)	No		Index	
a	int(10)	No			
aMax	int(10)	No			
x	smallint(6)	No			
y	smallint(6)	No			
color	varchar(40)	No			
ver_id	int(10)	No			

Columns description

- id: Storage of a carrier id
- product_id: Product id in the storage
- a: current amount of the product in the storage
- aMax: the maximum amount of the product in the storage
- x: x coordinates that determine the position of the storage on the design view
- y: y coordinates that determine the position of the storage on the design view
- color: color of the connecting line on the design view thats connect the storage with the other elements on the design
- ver_id: Design version id in which the storage of the carrier belongs

Table structure for table test_machine_log

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
des_id	int(10)	No		Index	
timeslot	int(10)	No		Index	

mach_io_id	int(10)	No			
mach_state	bigint(20)	No			
t2ni	int(10)	No			

Columns description

- id: test_machine_log id
- des_id: design id
- timeslot: the timer value
- mach_io_id: Machine input/output id
- mach_state: Machine state
- t2ni: Time to next input

Table structure for table test_sse

Column	Type	Null	Default	Key	Extra
design_id	int(10)	No		Primary	AUTO_INCREMENT
sess_id	varchar(40)	No			
test_time	int(10)	No			

Columns description

- design_id: Design id
- sess id: Session id. The session number
- test_time: the timers value

Table structure for table test_storage_log

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
des_id	int(10)	No		Index	
timeslot	int(10)	No		Index	
storage_id	int(10)	No			
a	int(11)	No			

Columns description

- id: Table test_storage_log id
- des_id: Design id
- timeslot: the timer value
- storage_id: Carrier id
- a: amount

Table structure for table warehouse

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
product_id	int(10)	No		Index	
a	int(10)	No			
aMax	int(10)	No			
x	smallint(6)	No			
y	smallint(6)	No			
color	varchar(40)	No			
humans_min	decimal(10,2)	No	0.00		
des_id	int(10)	No			

Columns description

- id: warehouse id
- product_id: Product id in the storage
- a: current amount of the product in the warehouse
- aMax: the maximum amount of the product in the warehouse
- x: x coordinates that determine the position of the warehouse on the design view
- y: y coordinates that determine the position of the warehouse on the design view
- color: color of the connecting line on the design view that's connect the

warehouse with the other elements on the design

- humans_min: Minimum number of humans for the warehouse to operate
- des_id: Design id in which the warehouse of the carrier belongs

Table structure for table warehouse_versions

Column	Type	Null	Default	Key	Extra
id	int(10)	No		Primary	AUTO_INCREMENT
product_id	int(10)	No		Index	
a	int(10)	No			
aMax	int(10)	No			
x	smallint(6)	No			
y	smallint(6)	No			
color	varchar(40)	No			
humans_min	decimal(10,2)	No	0.00		
ver_id	int(10)	No			

Columns description

- id: warehouse r id
- product_id: Product id in the warehouse
- a: current amount of the product in the warehouse
- aMax: the maximum amount of the product in the warehouse
- x: x coordinates that determine the position of the warehouse on the design view
- y: y coordinates that determine the position of the warehouse on the design view
- color: color of the connecting line on the design view that's connect the warehouse with the other elements on the design
- ver_id: Design version id in which the warehouse belongs

Appendix B: Functions

Function: closeDesign

Description: close a design

```
function closeDesign(user_id) {
        hide_design();
        design_v1_menu_show();
        machStorEdit="yes";
}
```

Function: copyDesign

Description: copy a design

```
function copyDesign(user_id, design_id){
  var obj = {"user_id":user_id};
  d3.request("./design_mode/get_designs_name.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {
    jsonDesignes=JSON.parse(phpResponseObj.response);
console.log(jsonDesignes);
    $('#des_copy_menu').removeClass("active");
    $("#copyModal").modal();
              $('#copy-modal-body').html('<form class="form-
horizontal"><div class="form-group"><label class="col-sm-2
control-label" for="textInput-markup2">Name</label><div
class="col-sm-10"><input type="text" id="textInput-markup2"
class="form-control"></div></div></form>');
    $('#copy-modal-msg').html('');
        $("#copy-btn_ok_1").off('click').on('click', function
```

```
(event, data) {
      var found,i;
      found=false;
      i=0;
      while ((found==false)&&(i<jsonDesignes.length)){
                  if   (jsonDesignes[i].name==$("#textInput-
markup2").val()){
          found=true;
        }
        i++;
      }
           if  ((found==false)&&($("#textInput-markup2").val()!
=='')&&(($("#textInput-markup2").val().match(/^\s+$/)        ===
null))) {
        $("#copyModal").modal("hide");
        hide_design();
         var obj2 = {"user_id":user_id, "design_id":design_id,
"design_name":($("#textInput-markup2").val()).trim(),
"time_resolution":designTimeResolution};
        d3.request("./design_mode/copy_design.php")
        .mimeType("application/json")
                .header("Content-Type","application/x-www-form-
urlencoded")
        .response(function(phpResponseObj2) {
          if ($.isNumeric(phpResponseObj2.response)) {
                $('#selected_design_name').text(($("#textInput-
markup2").val()).trim());
            designId=phpResponseObj2.response;
            $('#design_info').show();
            $('#design_buttons').show();
            design_v2_menu_show();
            sse_source.close();
            machStorEdit="yes";
            design_rd(phpResponseObj2.response);
```

```
            }
          })
          .send("POST","q="+JSON.stringify(obj2));
        }
        else{
                if (found==true) $('#copy-modal-msg').html('<div class="form-group has-error"><div class="col-sm-10"><span class="help-block">Name in use. Please select another name.</span></div></div>');
          else $('#copy-modal-msg').html('<div class="form-group has-error"><div class="col-sm-10"><span class="help-block">Please select a name.</span></div></div>');
        }
      })
    })
    .send("POST","q="+JSON.stringify(obj));
}
```

Function: design_rd

Description: Read data of a design

```
function design_rd(design_id, undrag) {
 currentMode = "dm-design";

 //Set the value of global var
 designId = design_id;

 //acquire objects
 var obj = {"design_id":design_id};
 d3.request("./design_mode/design_rd.php")
    .mimeType("application/json")
    .header("Content-Type","application/x-www-form-urlencoded")
    .response(function(phpResponseObj) {
```

```
    jsonDesign=JSON.parse(phpResponseObj.response);
    console.log('---------');
    console.log(jsonDesign);
    console.log('---------');
    d3.select("body").append("svg")
        .attr("height","100%")
        .attr("width","100%")
        .attr("id","SVG_ID")
        .on("contextmenu", function (data, i) {
          d3.event.preventDefault();
        });
    d3.select("svg").append("defs");
    defineShadow();
    createLines();
    createMachines(undrag);
    createStorages(undrag);
  }).send("POST","q="+JSON.stringify(obj));

 //SSE
 if(typeof(sse_source) != "undefined" && sse_source !== null) sse_source.close();
        sse_source = new EventSource("./design_mode/design_rd_sse.php?
q="+JSON.stringify(obj));
 sse_source.onmessage = function(event) {
  if (allowSSE == "yes") {
   jsonDesign=JSON.parse(event.data);
   d3.selectAll("polyline").remove();
   d3.selectAll(".machine").remove();
   d3.selectAll(".storage").remove();
   createLines();
   createMachines(undrag);
   createStorages(undrag);
  }
```

```
 };
 //sse_source.close();

 //start testSSE
        if(typeof(testSSE_source) != "undefined" && testSSE_source !== null)
testSSE_source.close();
 var obj = {"design_id":design_id};
          testSSE_source = new EventSource("./design_mode/test_db_rd_sse.php?
q="+JSON.stringify(obj));
 testSSE_source.onmessage = function(event) {
  var jsonResponse=JSON.parse(event.data);
  console.log(currentMode);
  console.log(jsonResponse.test_owner);
  console.log(jsonResponse.test_time);
  console.log(jsonResponse.des_update);
   var currTime = jsonResponse.test_time;
  //when somebody else enters to test mode
  if ((currentMode == "dm-design")&&(jsonResponse.test_owner == "not owner")) {
   //test_design function without the db interaction
   currentMode = "dm-test";
  console.log("a");
   //buttons mamagement
   $('#des_test_menu').removeClass("active");
   design_test_info_buttons_hide();
   $('#design_buttons').hide();
   $('#design_list').hide();
   $('#DesignArea').show();
   $('#test_buttons').show();
   $('#time_info').show();
   $('#test_step_back_button').hide();
   $('#test_play_button').hide();
   $('#test_step_for_button').hide();
```

```
$('#test_stop_button').hide();
$('#test_pause_button').hide();
$('#test_close_button').hide();
$('#test_reset_button').hide();

//read test data for current testTime from db
var obj = {"des_id":designId, "time":currTime};
d3.request("./design_mode/test_db_rd.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
  testObj = JSON.parse(phpResponseObj.response);
  //entering to design test mode
  testDesignEnv = "yes";
  machStorEdit = "no";
  d3.selectAll(".machineTitle")
      .attr("fill","#000")
      .attr("y","-7");
  //set test time
  $('#time_design').text(currTime);
  //Create machine state bars and create MachSt json object
  var machineG1 = "";
  machSt = {};
  var jsonMachines = jsonDesign["machines"];
  jsonMachines.forEach((dMach,iMach) => {
   machineG1 = d3.select("#machineG1"+dMach.id);
   machSt[dMach.id]={};
   machSt[dMach.id].inputTime=dMach.input_time;
   machSt[dMach.id].t2ni=dMach.input_time;
   machSt[dMach.id].ins={};
   dMach.ins.forEach((dMIn,iIn) => {
    machSt[dMach.id].ins[dMIn.row_num]={};
```

```
        machSt[dMach.id].ins[dMIn.row_num].storageId=dMIn.storage_id;
        machSt[dMach.id].ins[dMIn.row_num].a=dMIn.a;
       });
       machSt[dMach.id].outs={};
       dMach.outs.forEach((dMOut,iOut) => {
        machSt[dMach.id].outs[dMOut.row_num]={};
        machSt[dMach.id].outs[dMOut.row_num].id=dMOut.id;
        machSt[dMach.id].outs[dMOut.row_num].storageId=dMOut.storage_id;
        machSt[dMach.id].outs[dMOut.row_num].a=dMOut.a;
        machSt[dMach.id].outs[dMOut.row_num].prodTime=dMOut.prod_time;
         machSt[dMach.id].outs[dMOut.row_num].color=(dMOut.color==""?"#bedee1":dM
Out.color);
        machSt[dMach.id].outs[dMOut.row_num].stateN=dMOut.out_state;
        machSt[dMach.id].outs[dMOut.row_num].state={};
        machSt[dMach.id].outs[dMOut.row_num].state[0]={};
        machSt[dMach.id].outs[dMOut.row_num].state[0].bit="";
        for (var i=1;i<=dMOut.prod_time;i++) {
         machSt[dMach.id].outs[dMOut.row_num].state[i]={};
          machSt[dMach.id].outs[dMOut.row_num].state[i].bit=stateBit(dMOut.out_state,d
MOut.prod_time,i);
         machSt[dMach.id].outs[dMOut.row_num].state[i].obj = machineG1.append("rect")
          .attr("x",Math.round(15+(70*(i-1)/dMOut.prod_time)))
          .attr("y",Math.round(18+(iOut*20)))
            .attr("width",(Math.round(15+(70*i/dMOut.prod_time))-Math.round(15+(70*(i-
1)/dMOut.prod_time))))
          .attr("height","4")
          .attr("opacity","0.2")
          .attr("filter","url(#shadow)")
          .attr("fill","#fff");
        };
       });
      });
```

```
        //create storA array
        storA = {};
        var jsonStorages = jsonDesign["storages"];
        jsonStorages.forEach((dStor,iStor) => {
          storA[dStor.id] = {};
          storA[dStor.id].a = dStor.a;
          storA[dStor.id].aMax = dStor.aMax;
          storA[dStor.id].bars = {};
          storA[dStor.id].txtobj = d3.select("#storVals"+dStor.id);
          for (var p=0; p<20; p++) {
            storA[dStor.id].bars[+p+1]={};
            storA[dStor.id].bars[+p+1].obj=d3.select("#storBar"+dStor.id+"-"+p);
            storA[dStor.id].bars[+p+1].bit=(dStor.a>=(p+1)*(dStor.aMax/20)?"1":"0");
          }
        });
        //update machine state colors
        for (var mId in machSt) {
          for (var mOut in machSt[mId].outs) {
            for (var mOutBit in machSt[mId].outs[mOut].state) {
              if (mOutBit!=0)
                machSt[mId].outs[mOut].state[mOutBit].obj
                                                              .attr("opacity",
(stateBit(testObj.machTest[machSt[mId].outs[mOut].id].mach_state,
machSt[mId].outs[mOut].prodTime, mOutBit)=="1"?"1":"0.2"))
                        .attr("fill",(stateBit(testObj.machTest[machSt[mId].outs[mOut].id].mach_state,
machSt[mId].outs[mOut].prodTime, mOutBit)=="1"?machSt[mId].outs[mOut].color:"#fff"));
            };
          };
        };
        //Update storage bar data
        for (var sId in storA) {
          for (var p=0; p<20; p++) {
```

```
        storA[sId].bars[p+1].obj
                                                    .attr("fill",((testObj.storTest[sId].a
>=((p+1)*(storA[sId].aMax/20)))?"#9ecf99":"#fff"));
        };
        storA[sId].txtobj
         .text(testObj.storTest[sId].a+"/"+storA[sId].aMax);
       };

     })
     .send("POST","q="+JSON.stringify(obj));
  }

  //when the test owner changes time
  if ((currentMode == "dm-test")&&(jsonResponse.test_owner == "not owner")) {
   $('#test_step_back_button').hide();
   $('#test_play_button').hide();
   $('#test_pause_button').hide();
   $('#test_step_for_button').hide();
   $('#test_reset_button').hide();
   $('#test_close_button').hide();

   //read test data for testTime from db
   var obj = {"des_id":designId, "time":currTime};
   d3.request("./design_mode/test_db_rd.php")
    .mimeType("application/json")
    .header("Content-Type","application/x-www-form-urlencoded")
    .response(function(phpResponseObj) {
     testObj = JSON.parse(phpResponseObj.response);

     //set test time
     $('#time_design').text(currTime);
```

```
        //update machine state colors
        for (var mId in machSt) {
         for (var mOut in machSt[mId].outs) {
          for (var mOutBit in machSt[mId].outs[mOut].state) {
           if (mOutBit!=0)
            machSt[mId].outs[mOut].state[mOutBit].obj
                                                              .attr("opacity",
(stateBit(testObj.machTest[machSt[mId].outs[mOut].id].mach_state,
machSt[mId].outs[mOut].prodTime, mOutBit)=="1"?"1":"0.2"))
                  .attr("fill",(stateBit(testObj.machTest[machSt[mId].outs[mOut].id].mach_state,
machSt[mId].outs[mOut].prodTime, mOutBit)=="1"?machSt[mId].outs[mOut].color:"#fff"));
          };
         };
        };

        //Update storage bar data
        for (var sId in storA) {
         for (var p=0; p<20; p++) {
          storA[sId].bars[p+1].obj
                                                          .attr("fill",((testObj.storTest[sId].a
>=((p+1)*(storA[sId].aMax/20)))?"#9ecf99":"#fff"));
         };
         storA[sId].txtobj
          .text(testObj.storTest[sId].a+"/"+storA[sId].aMax);
        };

      })
      .send("POST","q="+JSON.stringify(obj));
    }

    //when the test owner leaves test mode
    if ((currentMode == "dm-test")&&(jsonResponse.test_owner == "no test")) {
```

```
    sse_source.close();
    d3.selectAll("polyline").remove();
    d3.selectAll(".machine").remove();
    d3.selectAll(".storage").remove();
    design_rd(designId);
    testDesignEnv = "no";
    machStorEdit = "yes";
    //design_v2_menu_show();

    $('#test_buttons').hide();
    $('#time_info').hide();
    $('#design_buttons').show();
    $('#design_list').show();
  }
 };
}
```

Function: newDesign

Description: Create a new Design

```
function newDesign(user_id) {
 var obj = {"user_id":user_id};
 d3.request("./design_mode/get_designs_name.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {

//   read Designes from db to check if name exists...
   jsonDesignes=JSON.parse(phpResponseObj.response);

   var options="";
```

```
  for (x in TIME_STEPS) {
        if (x=='3600') options+='<option value="'+x+'" selected="selected">'+TIME_STEPS[x]
+'</option>';
     else options+='<option value="'+x+'">'+TIME_STEPS[x]+'</option>';
  }

 //call new modal
  $('#des_new_menu').removeClass("active");
  $("#newModal").modal();
  $('#new-modal-body').html('\
   <div class="container-fluid">\
    <form class="form-horizontal">\
    <div class="form-group">\
     <div class="row">\
      <label class="col-sm-4 control-label" for="textInput-markup">Name</label>\
      <div class="col-sm-6">\
       <input type="text" id="textInput-markup" class="form-control">\
      </div>\
     </div>\
     <div class="row" >\
      <label class="col-sm-4 control-label" for="textInput-time">Time Resolution </label>\
      <div class="col-sm-6">\
       <select id="textInput-time" class="form-control">'+options+'\
       </select>\
      </div>\
     </div>\
    </div>\
    </form>\
   </div>');
  $('#new-modal-msg').html('');
  $("#btn_ok_1").off('click').on('click', function (event, data) {
   var found,i;
```

```
    found=false;
    i=0;
    while ((found==false)&&(i<jsonDesignes.length)){
     if (jsonDesignes[i].name==$("#textInput-markup").val()){
      found=true;
     }
     i++;
    }
                    if    ((found==false)&&($("#textInput-markup").val()!='')&&(($("#textInput-
markup").val().match(/^\s+$/) === null))) {
     $("#newModal").modal("hide");
     sse_source.close();

     // hide the old design, the buttons and info
     hide_design();

     //send sql to insert new design
         var obj2 = {"user_id":user_id, "design_name":($("#textInput-markup").val()).trim(),
"time_resolution":$("#textInput-time").val()};
     d3.request("./design_mode/insert_new_design.php")
     .mimeType("application/json")
     .header("Content-Type","application/x-www-form-urlencoded")
     .response(function(phpResponseObj2) {
      if ($.isNumeric(phpResponseObj2.response)) {
       //set global vars
       designTimeResolution=$("#textInput-time").val();
       //designId=phpResponseObj2.response;
       //show buttons and info
       $('#selected_design_name').text(($("#textInput-markup").val()).trim());
       designId=phpResponseObj2.response;
       $('#design_info').show();
       $('#design_buttons').show();
```

```
        design_v2_menu_show();
      }
     })
     .send("POST","q="+JSON.stringify(obj2));
   }
   else{
       if (found==true) $('#new-modal-msg').html('<div class="form-group has-error"><div class="col-sm-10"><span class="help-block">Name in use. Please select another name.</span></div></div>');
       else $('#new-modal-msg').html('<div class="form-group has-error"><div class="col-sm-10"><span class="help-block">Please select a name.</span></div></div>');
   }
  })
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: `openDesign`

Description: User opens an existing design

```
function deleteDesign(user_id) {
 var obj = {"user_id":user_id};
 d3.request("./design_mode/get_own_designs_and_versions.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
//   read Designes from db to select for opening ...
  jsonDesignes=JSON.parse(phpResponseObj.response);

  //call delete modal
```

```
$('#des_delete_menu').removeClass("active");
$("#deleteModal").modal();
    $('#delete-modal-msg').html('<div class="form-group has-error"><div class="col-sm-12"><span class="help-block">Red colored Designs or Versions cannot be deleted as long as they are used in Simulations.</span></div></div>');
$('#delete-modal-body').treeview({
collapseIcon: "fa fa-angle-down",
data: jsonDesignes,
expandIcon: "fa fa-angle-right",
nodeIcon: "fa fa-folder",
onNodeSelected: function(event, data) {

  //code to make inactive design and all its versions...
  if (data.ver_id=='design_selected'){
   $("#warningModal").css("z-index", "8000").modal();
     $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon pficon-warning-triangle-o"></span><strong>Design \''+data.name+'\' and all its versions will be deleted! Are you sure you want to continue?</strong></div>');
    $("#warning_ok").off('click').on('click', function() {

    $("#warningModal").modal("hide");
    $("#deleteModal").modal("hide");
    var obj3 = {"design_id":data.des_id};
    d3.request("./design_mode/design_inactive.php")
     .mimeType("application/json")
     .header("Content-Type","application/x-www-form-urlencoded")
     .response(function(phpResponseObj3) {
      var jsonDesign3=JSON.parse(phpResponseObj3.response);
     }).send("POST","q="+JSON.stringify(obj3));
    })
  }
  //code to make inactive version ...
```

```
      else {
       $("#warningModal").css("z-index", "8000").modal();
         $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon pficon-warning-triangle-o"></span><strong>Version '+data.number+' of design \''+data.name+'\' will be deleted! Are you sure you want to continue? </strong></div>');
       $("#warning_ok").off('click').on('click', function() {

       $("#warningModal").modal("hide");
       $("#deleteModal").modal("hide");
       var obj2 = {"version_id":data.ver_id};
       d3.request("./design_mode/version_inactive.php")
        .mimeType("application/json")
        .header("Content-Type","application/x-www-form-urlencoded")
        .response(function(phpResponseObj2) {
         var jsonDesign4=JSON.parse(phpResponseObj2.response);
        }).send("POST","q="+JSON.stringify(obj2));
       })
      }
     //}
     },
   levels:1,
   showBorder: false
   });
  })
  .send("POST","q="+JSON.stringify(obj));
}
```

Function: `openDesign`

Description: User opens an existing design

```
function openDesign(user_id) {
  var obj = {"user_id":user_id};
  d3.request("./design_mode/get_designs.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {
    jsonDesignes=JSON.parse(phpResponseObj.response);
    $('#des_open_menu').removeClass("active");
    $("#openModal").modal();
    $('#open-modal-body').treeview({
    collapseIcon: "fa fa-angle-down",
    data: jsonDesignes,
    expandIcon: "fa fa-angle-right",
    nodeIcon: "fa fa-folder",
    onNodeSelected: function(event, data) {
      $("#openModal").modal("hide");
      hide_design();
      $('#selected_design_name').text(data.name);
      designId=data.id;
      $('#design_info').show();
      machStorEdit="no";
                  if   (data.permission   ==   'test'   )   {$
('#des_test_menu').show();machStorEdit="no";}
                  if   (data.permission   ==   'copy'   )   {$
('#des_copy_menu').show();$
('#des_test_menu').show();machStorEdit="no";}
                  if   (data.permission   ==   'full'   )   {$
('#design_buttons').show();$('#des_load_version_menu').show();
$('#des_save_version_menu').show();$('#des_copy_menu').show();
$('#des_test_menu').show();machStorEdit="yes";}
                 if   (data.permission   ==   'owner'   )   {$
('#design_buttons').show();$('#des_load_version_menu').show();
$('#des_save_version_menu').show();$('#des_copy_menu').show();
$('#des_test_menu').show();machStorEdit="yes";}
```

```
      $('#des_close_menu').show();
      designTimeResolution=data.time_resolution;
      sse_source.close();
      if (machStorEdit=="no") design_rd(data.id,1);
      else if (machStorEdit=="yes") design_rd(data.id);
    },
    levels:1,
    showBorder: false
    });
  })
  .send("POST","q="+JSON.stringify(obj));
}
```

Function: shareDesign

Description: Sharing a design

```
function shareDesign(user_id) {
  var obj = {"user_id":user_id};
  d3.request("./design_mode/get_own_designs.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {
    jsonDesignes=JSON.parse(phpResponseObj.response);

    $('#des_share_menu').removeClass("active");
    $("#shareModal").modal();
    $('#share-modal-msg').html('');
    $('#share-modal-body').treeview({
    collapseIcon: "fa fa-angle-down",
    data: jsonDesignes,
    expandIcon: "fa fa-angle-right",
    nodeIcon: "fa fa-folder",
    onNodeSelected: function(event, data) {
```

```
        var design_id=data.id;
        var obj3 = {"user_id":user_id,"design_id":design_id};
        d3.request("./design_mode/perm_design.php")
          .mimeType("application/json")
                .header("Content-Type","application/x-www-form-
urlencoded")
          .response(function(phpResponseObj3) {
                                                          var
jsonDesign3=JSON.parse(phpResponseObj3.response);
            $("#chownModal").modal();
             $('#chown-modal-head').html('<button type="button"
class="close"          data-dismiss="modal">&times;</button><h4
class="modal-title"        >Change        Ownership        for
'+data.name+'</h4>');
            $('#chown-modal-msg').html('');
            create_table(user_id, design_id);
          }).send("POST","q="+JSON.stringify(obj3));
      },
    levels:1,
    showBorder: false
    });
  })
  .send("POST","q="+JSON.stringify(obj));
}
```

Function: give_permission

Description: Gives permissions to a design

```
function give_permission(user_id,user_id_to_change,design_id,permission){
 var obj = {"user_id":user_id_to_change, "design_id":design_id, "permission":permission};
 d3.request("./design_mode/give_permission.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
```

```
 .response(function(phpResponseObj) {
   JSON.parse(phpResponseObj.response);
   create_table(user_id, design_id);
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: create_table

Description: create permissions table for a design

```
function create_table(user_id, design_id) {
 var obj3 = {"user_id":user_id,"design_id":design_id};
 d3.request("./design_mode/perm_design.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj3) {
   var jsonDesign3=JSON.parse(phpResponseObj3.response);
   $('#chown-modal-body').html('');
        $('#chown-modal-body').html('<table style="width:100%" class="table table-striped
table-bordered table-hover" id="table1">\
    <thead><tr>\
     <th>User Name   </th>\
     <th>Name   </th>\
     <th>Permission   </th>\
     <th>Actions         </th>\
    </tr></thead></table>');
    $('#table1').DataTable({
     data: jsonDesign3,
     columns: [
      { data: "uname" },
      { data: "name" },
      { data: "permission" },
```

```
        { data: null,
          searchable: false ,
          className: "table-view-pf-actions",
          render: function (data, type, full, meta) {
          return '<div class="dropdown dropdown-kebab-pf">' +
                        '<button class="btn btn-default dropdown-toggle" type="button" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="true">' +
           '<span class="fa fa-ellipsis-v"></span></button>' +
                                    '<ul class="dropdown-menu dropdown-menu-right" aria-
labelledby="dropdownKebabRight">' +
                                '<li><a href="#" onclick=give_permission('+user_id+','+data['id']
+','+design_id+',"full");>give Full permission</a></li>' +
                                '<li><a href="#" onclick=give_permission('+user_id+','+data['id']
+','+design_id+',"copy");>give Copy permission</a></li>' +
                                '<li><a href="#" onclick=give_permission('+user_id+','+data['id']
+','+design_id+',"test");>give Test permission</a></li>' +
                                '<li><a href="#" onclick=give_permission('+user_id+','+data['id']
+','+design_id+',"view");>give View permission</a></li>' +
                                '<li><a href="#" onclick=give_permission('+user_id+','+data['id']
+','+design_id+',"");>delete permission</a></li></ul></div>';
           }
         }
       ]
     });
  }).send("POST","q="+JSON.stringify(obj3));
}
```

Function: `testDesign`

Description: User can perform a logical test to a design

```
function testDesign(user_id) {
```

```
if (testDesignEnv == "no") {
  currentMode = "dm-test";
  $('#des_test_menu').removeClass("active");
  design_test_info_buttons_hide();
  $('#design_buttons').hide();
  $('#design_list').hide();
  $('#DesignArea').show();
  $('#test_buttons').show();
  $('#time_info').show();
  $('#test_play_button').show();
  $('#test_step_for_button').show();
  $('#test_reset_button').show();
  $('#test_close_button').show();
  testDesignEnv = "yes";
  machStorEdit = "no";
  d3.selectAll(".machineTitle")
        .attr("fill","#000")
        .attr("y","-7");
  testTime = 0;
  $('#time_design').text(testTime);
  var machineG1 = "";
  machSt = {};
  var jsonMachines = jsonDesign["machines"];
  jsonMachines.forEach((dMach,iMach) => {
    machineG1 = d3.select("#machineG1"+dMach.id);
    machSt[dMach.id]={};
    machSt[dMach.id].inputTime=dMach.input_time;
    machSt[dMach.id].t2ni=0;
    machSt[dMach.id].ins={};
    dMach.ins.forEach((dMIn,iIn) => {
      machSt[dMach.id].ins[dMIn.row_num]={};
      machSt[dMach.id].ins[dMIn.row_num].storageId=dMIn.storage_id;
```

```
      machSt[dMach.id].ins[dMIn.row_num].a=dMIn.a;
    });
    machSt[dMach.id].outs={};
    dMach.outs.forEach((dMOut,iOut) => {
      machSt[dMach.id].outs[dMOut.row_num]={};
      machSt[dMach.id].outs[dMOut.row_num].id=dMOut.id;
      machSt[dMach.id].outs[dMOut.row_num].storageId=dMOut.storage_id;
      machSt[dMach.id].outs[dMOut.row_num].a=dMOut.a;
      machSt[dMach.id].outs[dMOut.row_num].prodTime=dMOut.prod_time;
       machSt[dMach.id].outs[dMOut.row_num].color=(dMOut.color==""?"#bedee1":dMOut.
color);
      machSt[dMach.id].outs[dMOut.row_num].stateN=dMOut.out_state;
                machSt[dMach.id].t2ni=Math.max(machSt[dMach.id].t2ni,(dMach.input_time-
(dMOut.prod_time-Math.floor(Math.log2(dMOut.out_state)))));
      machSt[dMach.id].outs[dMOut.row_num].state={};
      machSt[dMach.id].outs[dMOut.row_num].state[0]={};
      machSt[dMach.id].outs[dMOut.row_num].state[0].bit="";
      for (var i=1;i<=dMOut.prod_time;i++) {
       machSt[dMach.id].outs[dMOut.row_num].state[i]={};
        machSt[dMach.id].outs[dMOut.row_num].state[i].bit=stateBit(dMOut.out_state,dMO
ut.prod_time,i);
       machSt[dMach.id].outs[dMOut.row_num].state[i].obj = machineG1.append("rect")
        .attr("x",Math.round(15+(70*(i-1)/dMOut.prod_time)))
        .attr("y",Math.round(18+(iOut*20)))
              .attr("width",(Math.round(15+(70*i/dMOut.prod_time))-Math.round(15+(70*(i-
1)/dMOut.prod_time))))
        .attr("height","4")
        .attr("opacity","0.2")
        .attr("filter","url(#shadow)")
        .attr("fill","#fff");
      };
    });
```

```
  });
  storA = {};
  var jsonStorages = jsonDesign["storages"];
  jsonStorages.forEach((dStor,iStor) => {
   storA[dStor.id] = {};
   storA[dStor.id].a = dStor.a;
   storA[dStor.id].aMax = dStor.aMax;
   storA[dStor.id].bars = {};
   storA[dStor.id].txtobj = d3.select("#storVals"+dStor.id);
   for (var p=0; p<20; p++) {
    storA[dStor.id].bars[+p+1]={};
    storA[dStor.id].bars[+p+1].obj=d3.select("#storBar"+dStor.id+"-"+p);
    storA[dStor.id].bars[+p+1].bit=(dStor.a>=(p+1)*(dStor.aMax/20)?"1":"0");
   }
  });

//reset data in db tables: test_machine_log, test_storage_log
  var obj = {"des_id":designId, "machine_states":machSt, "storages_a":storA};
  d3.request("./design_mode/test_db_reset.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {})
  .send("POST","q="+JSON.stringify(obj));

  machUpdateStateColors();
 }
}

//stateBit returns a bit of a state (number, length, bit#)
function stateBit(num, len, b) {
 var binNum = parseInt(num, 10).toString(2);
 var zero = len - binNum.length + 1;
```

```
 var binNum = Array(+(zero > 0 && zero)).join("0") + binNum;
 return binNum.substring(b-1,b);
}

//binArr2Num converts the binary state array to number
//binary input value must be sent through JSON.stringify function
function binArr2Num(strArr) {
 var bitArr = JSON.parse(strArr);
 var num = "";
 for (var d in bitArr) {
  if (d > 0) num = num+bitArr[d].bit;
 }
 return parseInt(num, 2);
}

//Update machine state bar colors
function machUpdateStateColors() {
 for (var mId in machSt) {
  for (var mOut in machSt[mId].outs) {
   for (var mOutBit in machSt[mId].outs[mOut].state) {
    if (mOutBit!=0)
     machSt[mId].outs[mOut].state[mOutBit].obj
       attr("opacity",(machSt[mId].outs[mOut].state[mOutBit].bit=="1"?"1":"0.2"))
       .attr("fill",(machSt[mId].outs[mOut].state[mOutBit].bit=="1"?
       machSt[mId].outs[mOut].color:"#fff"));
   };
  };
 };
}

//Update storage bar data
function storUpdateBarData() {
```

```
 for (var sId in storA) {
  for (var p=0; p<20; p++) {
   storA[sId].bars[p+1].obj
    .attr("fill",((storA[sId].bars[+p+1].bit=="1")?"#9ecf99":"#fff"));
  };
  storA[sId].txtobj
   .text(storA[sId].a+"/"+storA[sId].aMax);
 };
}
```

Function: viewDesign

Description: View an existing design

```
function viewDesign(user_id) {
 var obj = {"user_id":user_id};
 d3.request("./design_mode/get_designs_and_versions.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
//   read Designes from db to select for opening ...
  jsonDesignes=JSON.parse(phpResponseObj.response);
  //call open modal
  $('#des_view_menu').removeClass("active");
  $("#viewModal").modal();
  $('#view-modal-body').treeview({
  collapseIcon: "fa fa-angle-down",
  data: jsonDesignes,
  expandIcon: "fa fa-angle-right",
  nodeIcon: "fa fa-folder",
  onNodeSelected: function(event, data) {
   $("#viewModal").modal("hide");
```

```
// hide the old design, the buttons and info
    hide_design();

 // show menu
    $('#selected_design_name').text(data.name);
    designId=data.des_id;
    $('#design_info').show();
    $('#des_close_menu').show();
    machStorEdit="no";

 //code to read and display design ...
 //acquire objects in case of design selected
    if (data.ver_id=='design_selected'){
      sse_source.close();
      design_rd(data.des_id,true);
    }
//code to read and display version ...
//acquire objects in case of version selected
    else {
      sse_source.close();
      version_rd(data.ver_id,true);
    }
   },
   levels:1,
   showBorder: false
   });
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: insertProduct

Description: insert products list to a design

```
function insertProduct(user_id,design_id,table_add_mod) {
  var obj = {"user_id":user_id,"design_id":design_id};
 d3.request("./design_mode/get_products_name.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
//   read Products from db to check if name exists ...
  jsonProducts=JSON.parse(phpResponseObj.response);
   //console.log(jsonProducts);
  var html_table_add='\
   <h3>Add Product</h3>\
        <table style="width:100%" class="table table-striped table-bordered table-hover"
id="add_product_table">\
   <thead><tr>\
    <th>Name</th>\
    <th>Unit</th>\
    <th>Actions</th>\
   </tr></thead><tr>\
   <td style="padding-left:0px;padding-right:0px;">\
   <div style="padding-left:0px;">\
    <input id="textInput-product-name" class="form-control " ></input>\
   </div>\
   </td>\
   <td style="padding-left:0px;padding-right:0px;">\
   <div style="padding-left:0px;">\
    <input id="textInput-product-unit" class="form-control " ></input>\
   </div>\
   </td>\
```

```
    <td style="padding-left:0px;padding-right:0px;width:75px;">\
            <div><button class="btn btn-default dropdown-toggle" type="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="true" style="padding-left:0px;padding-right:0px;width:75px"\
          onclick=add_product('+user_id+','+design_id+',$("#textInput-product-name").val(),$("#textInput-product-unit").val(),jsonProducts);>Add</button></div>\
    </td>\
    </tr>\
    </table>';
   var html_table='\
     <h3>Products</h3><table style="width:100%" class="table table-striped table-bordered table-hover" id="product_table">\
    <thead><tr>\
      <th>Name   </th>\
      <th>Unit   </th>\
                                                <th style="padding-left:24px;padding-right:25px;">Actions      </th>\
    </tr></thead></table>';


   //call modal
   $("#insertProductModal").css("z-index", "8000").modal();
   if ((table_add_mod==undefined)||(table_add_mod==''))
    $('#insert-product-modal-body').html(html_table_add+html_table);
   else $('#insert-product-modal-body').html(table_add_mod+html_table);

   $('#product_table').DataTable({
      data: jsonProducts,
      columns: [
       { data: "name" },
       { data: "unit" },
       { data: null,
```

```
        searchable: false ,
        className: "table-view-pf-actions",
        render: function (data, type, full, meta) {
        // Inline action kebab renderer
        var html_return;
                        if (data['delete']==true) html_return='<div><button type="button"
onclick=modify_product('+user_id+','+design_id+','+data['id']+',"'+data['name']
+'","'+data['unit']+'");>' +
          'Modify</button>\
           <button type="button" onclick=delete_product('+user_id+','+design_id+','+data['id']
+');>' +
          'Delete</button></div>';
                                    else html_return='<div><button type="button"
onclick=modify_product('+user_id+','+design_id+','+data['id']+',"'+data['name']
+'","'+data['unit']+'");>' +
          'Modify</button></div>';
        return html_return;
        }
       }
      ]
   });
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: add_product

Description: Add a product to products list of a design

```
function add_product(user_id, design_id, name, unit, jsonProducts){
 var found=false;
 for (x in jsonProducts) {
```

```
  if (jsonProducts[x].name==name) found=true;
 }
 if (found==false){
  var obj = {"design_id":design_id, "name":name, "unit":unit};
  d3.request("./design_mode/add_product.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {
   insertProduct(user_id,design_id,'');
  })
  .send("POST","q="+JSON.stringify(obj));
 }
 else {
  $("#warningModal").css("z-index", "10000").modal();
      $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon pficon-warning-triangle-o"></span><strong>Name in use. Please select another name.</strong></div>');
  $("#warning_ok").off('click').on('click', function() {
   $("#warningModal").modal("hide");
  })
 }
}
```

Function: delete_product

Description: Delete a product from products list

```
function delete_product(user_id,design_id,id){
 var obj = {"id":id};
 d3.request("./design_mode/delete_product.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
```

```
 .response(function(phpResponseObj) {
   insertProduct(user_id,design_id);
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: modify_produc

Description: modify an existing product from products list

```
function modify_product(user_id, design_id, id, name, unit){
   var html_table_modify='\
    <h3>Modify Products</h3>\
        <table style="width:100%" class="table table-striped table-bordered table-hover" id="product_table_mod">\
    <thead><tr>\
     <th>Name</th>\
     <th>Unit</th>\
     <th>Actions</th>\
    </tr></thead><tr>\
    <td style="padding-left:0px;padding-right:0px;">\
    <div style="padding-left:0px;">\
     <input id="modProduct-name" class="form-control" value="'+name+'"></input>\
    </div>\
    </td>\
    <td style="padding-left:0px;padding-right:0px;">\
    <div style="padding-left:0px;">\
     <input id="modProduct-unit" class="form-control" value="'+unit+'"></input>\
    </div>\
    </td>\
    <td style="padding-left:0px;padding-right:0px;width:75px;">\
            <div><button class="btn btn-default dropdown-toggle" type="button" data-
```

```
toggle="dropdown" aria-haspopup="true" aria-expanded="true" style="padding-left:0px;padding-right:0px;width:75px"\
        onclick=save_product('+user_id+','+design_id+','+id+',$("#modProduct-name").val(),$("#modProduct-unit").val());>Save</button></div>\
    </td>\
    </tr>\
    </table>';
    insertProduct(user_id, design_id, html_table_modify);
}
```

Function: save_product

Description: save the modifications of a product of an existing

```
function save_product(user_id, design_id, id, name, unit){
 if (1){
  var obj = {"design_id":design_id, "id":id, "name":name, "unit":unit};
  d3.request("./design_mode/save_product.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {
   insertProduct(user_id,design_id);
  })
  .send("POST","q="+JSON.stringify(obj));
 }
 else {
  $("#warningModal").css("z-index", "10000").modal();
      $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon pficon-warning-triangle-o"></span><strong>Not added due to invalid values.</strong></div>');
  $("#warning_ok").off('click').on('click', function() {
   $("#warningModal").modal("hide");
```

```
  })
 }
}
```

Function: insertStorage

Description: Insert a carrier to a design

```
function insertStorage(user_id,design_id) {
 var obj = {"user_id":user_id,"design_id":design_id};
 d3.request("./design_mode/get_products.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
//   read Products from db to display list ...
  var options=phpResponseObj.response;

  var html_inbody = '\
  <div class="container-fluid">\
   <form class="form-horizontal">\
    <div class="form-group">\
     <div class="row">\
              <label class="col-sm-4 control-label" for="textInput-product_id">Product to carry</label>\
      <div class="col-sm-3" id="product_list">\
                  <select id="textInput-product_id" class="form-control" style="width: 200px;">'+options+'</select>\
      </div>\
     </div>\
     <div class="row">\
      <label class="col-sm-4 control-label" for="textInput-amount">Initial Amount</label>\
            <div class="col-sm-4"><input type="text" id="textInput-amount" class="form-
```

```
control"></div>\
        </div>\
        <div class="row">\
                        <label  class="col-sm-4  control-label"  for="textInput-amountMax">Max
Amount</label>\
            <div class="col-sm-4"><input type="text" id="textInput-amountMax" class="form-
control"></div>\
        </div>\
        <div class="row">\
         <label class="col-sm-4 control-label" for="textInput-color">Line Color</label>\
                  <div  class="col-sm-7"><input  type="color"  id="textInput-color"  class="form-
control"></div>\
        </div>\
      </div>\
    </form>\
  </div>';

  //call new modal
  $("#insertStorageModal").modal();
  $('#insert-storage-modal-body').html(html_inbody);
  $('#insert-storage-modal-msg').html('');
  $("#insert-storage-btn_ok").off('click').on('click', function (event, data) {

   if ((Number($("#textInput-amountMax").val())>=Number($("#textInput-amount").val()))
     &&($.isNumeric($("#textInput-amount").val()))
     &&($.isNumeric($("#textInput-amountMax").val()))
     &&(Number($("#textInput-amount").val())>=0)
     &&(Number($("#textInput-amountMax").val())>0)) {
     $("#insertStorageModal").modal("hide");

     //send sql to insert new storage
            var  obj2  =  {"product_id":$("#textInput-product_id").val(),  "a":$("#textInput-
```

```
amount").val(),  "aMax":$("#textInput-amountMax").val(),  "color":$("#textInput-color").val(), "design_id":design_id};

      d3.request("./design_mode/insert_storage.php")
      .mimeType("application/json")
      .header("Content-Type","application/x-www-form-urlencoded")
      .response(function(phpResponseObj2) {
//       alert(phpResponseObj2.response);
        sse_source.close();
        design_rd(design_id);
      })
      .send("POST","q="+JSON.stringify(obj2));
    }
    else{
     $('#insert-storage-modal-msg').html('<div class="form-group has-error"><div class="col-sm-10"><span class="help-block">Please select numeric values for amounts (initial >= 0, max > 0 and max >= initial).</span></div></div>');
    }
   })
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: createLines

Description: Create a new line that connect the structural modules of a design

```
function createLines () {
 var jsonStorages = jsonDesign["storages"];
 var x1,x2,x5,x6,y1,y3,y6,machLineDistance;
 //loop for each storage
 jsonStorages.forEach((d,i) => {
```

```
  storageX[d.id]=d.x;
  storageY[d.id]=d.y;
 })
 var jsonMachines = jsonDesign["machines"];
 var svg = d3.select("svg");
 //loop for each machine
 jsonMachines.forEach((d,i) => {
          var  machHeight  =  20*(1+(d.ins.length<d.outs.length?  +d.outs.length:
+d.ins.length));
  var machBottom = +d.y+machHeight;
  var machMiddle = +d.y+machHeight/2;
  //loop for each machine input
  d.ins.forEach((dd,ii) => {
   if (dd.storage_id!=0){
    machLineDistance = 20*dd.row_num;
    y1 = +d.y+machLineDistance;
    x6 = +storageX[dd.storage_id]+82;
    y6 = +storageY[dd.storage_id]+10;
    x2 = (+d.x-machLineDistance);
    x5 = +x6+20;
         y3  =  (x5<x2?  +y6:(y6>machMiddle?+machBottom+machLineDistance:+d.y-
machLineDistance));
    svg.append("polyline")
      .attr("id","li-"+d.id+"-"+dd.storage_id)
      .attr("class","line")
      .attr("stroke",(dd.color==""?"black":dd.color))
      .attr("points",d.x+","+y1+" "+
           x2+","+y1+" "+
           x2+","+y3+" "+
```

```
            x5+","+y3+" "+
            x5+","+y6+" "+
            x6+","+y6)
        .attr("style","fill:none;stroke-width:2;");
    }
  })
  //loop for each machine output
  d.outs.forEach((dd,ii) => {
    if (dd.storage_id!=0){
      machLineDistance = 20*dd.row_num+10;
      x1 = +d.x+100;
      y1 = +d.y+20*dd.row_num;
      x6 = +storageX[dd.storage_id];
      y6 = +storageY[dd.storage_id]+10;
      x2 = (+x1+machLineDistance);
      x5 = +x6-20;
          y3  =  (x5>x2?  +y6:(y6>machMiddle?+machBottom+machLineDistance:+d.y-
machLineDistance));
      svg.append("polyline")
        .attr("id","lo-"+d.id+"-"+dd.storage_id)
        .attr("class","line")
        .attr("stroke",(dd.color==""?"black":dd.color))
        .attr("points",x1+","+y1+" "+
            x2+","+y1+" "+
            x2+","+y3+" "+
            x5+","+y3+" "+
            x5+","+y6+" "+
            x6+","+y6)
        .attr("style","fill:none;stroke-width:2;");
```

```
    }
   })
  })
}
```

Function: updateLinesMach

Description: updates the lines that are connected to a machine

```
function updateLinesMach (mach, machX, machY) {
 var x1,x2,x5,x6,y1,y3,y6,machLineDistance;
  var  machHeight  =  20*(1+(mach.ins.length<mach.outs.length?  +mach.outs.length:
+mach.ins.length));
 var machBottom = +machY+machHeight;
 var machMiddle = +machY+machHeight/2;
 //loop for each machine input
 mach.ins.forEach((dd,ii) => {
  if (dd.storage_id!=0){
   machLineDistance = 20*dd.row_num;
   y1 = +machY+machLineDistance;
   x6 = +storageX[dd.storage_id]+82;
   y6 = +storageY[dd.storage_id]+10;
   x2 = (+machX-machLineDistance);
   x5 = +x6+20;
      y3  =  (x5<x2?  +y6:(y6>machMiddle?+machBottom+machLineDistance:+machY-
machLineDistance));
   d3.select("#li-"+mach.id+"-"+dd.storage_id)
         .attr("points",machX+","+y1+"  "+x2+","+y1+"  "+x2+","+y3+"  "+x5+","+y3+"
"+x5+","+y6+" "+x6+","+y6);
  }
```

```
 })
 //loop for each machine output
 mach.outs.forEach((dd,ii) => {
  if (dd.storage_id!=0){
    machLineDistance = 20*dd.row_num+10;
    x1 = +machX+100;
    y1 = +machY+20*dd.row_num;
    x6 = +storageX[dd.storage_id];
    y6 = +storageY[dd.storage_id]+10;
    x2 = (+x1+machLineDistance);
    x5 = +x6-20;
       y3  =  (x5>x2?  +y6:(y6>machMiddle?+machBottom+machLineDistance:+machY-
machLineDistance));
    d3.select("#lo-"+mach.id+"-"+dd.storage_id)
              .attr("points",x1+","+y1+"  "+x2+","+y1+"  "+x2+","+y3+"  "+x5+","+y3+"
"+x5+","+y6+" "+x6+","+y6);
   }
 })
}
```

Function: updateLinesStor

Description: updates the lines that are connected to a carrier

```
function updateLinesStor (stor, storX, storY) {
 //loop for each machine
 jsonDesign["machines"].forEach((d,i) => {
  var x1,x2,x5,x6,y1,y3,y6,machLineDistance;
          var   machHeight   =   20*(1+(d.ins.length<d.outs.length?   +d.outs.length:
+d.ins.length));
```

```
  var machBottom = +d.y+machHeight;
  var machMiddle = +d.y+machHeight/2;
  //loop for each machine input
  d.ins.forEach((dd,ii) => {
   if ( dd.storage_id == stor.id ) {
    machLineDistance = 20*dd.row_num;
    y1 = +d.y+machLineDistance;
    x6 = +storX+82;
    y6 = +storY+10;
    x2 = (+d.x-machLineDistance);
    x5 = +x6+20;
        y3  =  (x5<x2?  +y6:(y6>machMiddle?+machBottom+machLineDistance:+d.y-
machLineDistance));
    d3.select("#li-"+d.id+"-"+dd.storage_id)
          .attr("points",d.x+","+y1+"  "+x2+","+y1+"  "+x2+","+y3+"  "+x5+","+y3+"
"+x5+","+y6+" "+x6+","+y6);
   }
  })
  //loop for each machine output
  d.outs.forEach((dd,ii) => {
   if ( dd.storage_id == stor.id ) {
    machLineDistance = 20*dd.row_num+10;
    x1 = +d.x+100;
    y1 = +d.y+20*dd.row_num;
    x6 = +storX;
    y6 = +storY+10;
    x2 = (+x1+machLineDistance);
    x5 = +x6-20;
        y3  =  (x5>x2?  +y6:(y6>machMiddle?+machBottom+machLineDistance:+d.y-
```

```
machLineDistance));
      d3.select("#lo-"+d.id+"-"+dd.storage_id)
                .attr("points",x1+","+y1+"  "+x2+","+y1+"  "+x2+","+y3+"  "+x5+","+y3+"
"+x5+","+y6+" "+x6+","+y6);
    }
   })
 })
}
```

Function: loadVersion

Description: Load a design version

```
function loadVersion(design_id) {
 //call warning modal
 $("#warningModal").modal();
    $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon
pficon-warning-triangle-o"></span><strong>Any unsaved changes in current version will be
lost! Are you sure you want to continue?</strong></div>');
 $("#warning_ok").off('click').on('click', function() {
  $('#des_load_version_menu').removeClass("active");
  $("#warningModal").modal("hide");
//alert(design_id);
  var obj = {"design_id":design_id};
  d3.request("./design_mode/get_versions.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {
   jsonVersions=JSON.parse(phpResponseObj.response);
 //   alert ("info:"+phpResponseObj.response);

 //   call open modal
```

```
    $("#loadVersionModal").modal();
    $('#load-version-modal-body').treeview({
    collapseIcon: "fa fa-angle-down",
    data: jsonVersions,
    expandIcon: "fa fa-angle-right",
    nodeIcon: "fa fa-folder",
    onNodeSelected: function(event, data) {
//    alert (data.id);
    $("#loadVersionModal").modal("hide");

//    hide design
    d3.select("#SVG_ID").remove();

//    delete design objects...
//    copy version to design
//    code to read and display version ...
//    acquire objects
     var obj2 = {"design_id":design_id, "version_id":data.id};
     d3.request("./design_mode/load_version.php")
        .mimeType("application/json")
        .header("Content-Type","application/x-www-form-urlencoded")
        .response(function(phpResponseObj2) {
         //send sql to read new design and show
         //$('#test_buttons').hide();
         sse_source.close();
         design_rd(design_id);
        }).send("POST","q="+JSON.stringify(obj2));
    },
    levels:1,
    showBorder: false
    });
  })
```

```
  .send("POST","q="+JSON.stringify(obj));
 });
 $("#warning_cancel").off('click').on('click', function() {
    $('#des_load_version_menu').removeClass("active");
 });
}
```

Function: createMachines

Description: Create a new machine

```
function createMachines (undrag) {
 var jsonMachines = jsonDesign["machines"];
 var svg = d3.select("svg");

 //loop for each machine
 jsonMachines.forEach((d,i) => {
  //load data in machine coordinate arrays
  machineX[d.id]=d.x;
  machineY[d.id]=d.y;

  //G (machine's up level group - includes everything)
  //creates all machines according to jsonStorages objects
  //-------------------------------------------------------
  var machineG = svg.append("g")
        .attr("id","machine"+d.id)
        .attr("class","machine")
        .attr("cursor","pointer")
        .attr("transform","translate("+d.x+","+d.y+")");

  //G1 (includes the main box of the machine)
  //-------------------------------------------------------------------------
```

```
  if (undrag==true) {
   var machineG1 = machineG.append("g")
          .attr("id","machineG1"+d.id)
          .attr("transform","translate(0,0)")
          .on("contextmenu", function () {
// if (machStorEdit=="yes") updateMachine(designId,d.id,d.name,d.ins.length,d.outs.length,d.input_time,d.humans_min);
            updateMachine(designId,d.id,d.name,d.ins.length,d.outs.length,d.input_time,d.humans_min);
          });
  } else {
   var machineG1 = machineG.append("g")
          .attr("id","machineG1"+d.id)
          .attr("transform","translate(0,0)")
          .call(d3.drag()
           .on("start", dragStartFunc)
           .on("drag", dragFunc)
           .on("end", dragEndFunc))
          .on("contextmenu", function () {
// if (machStorEdit=="yes") updateMachine(designId,d.id,d.name,d.ins.length,d.outs.length,d.input_time,d.humans_min);
            updateMachine(designId,d.id,d.name,d.ins.length,d.outs.length,d.input_time,d.humans_min);
          });

   function dragStartFunc() {
    if (testDesignEnv=='no'){
     allowSSE = "no";
     d3.select(this.parentNode).raise();
                                        xMouseDiff = event.clientX -
```

```
d3.select(this.parentNode).node().transform.baseVal[0].matrix.e;
                                                    yMouseDiff = event.clientY -
d3.select(this.parentNode).node().transform.baseVal[0].matrix.f;
    }
  }
  function dragFunc() {
   if (testDesignEnv=='no'){
    var dragX = ((event.clientX - xMouseDiff)-((event.clientX - xMouseDiff)%10));
    var dragY = ((event.clientY - yMouseDiff)-((event.clientY - yMouseDiff)%10));
    d3.select(this.parentNode).attr("transform", "translate("+dragX+","+dragY+")");
    updateLinesMach(d,dragX,dragY);
   }
  }
  function dragEndFunc() {
   if (testDesignEnv=='no'){
    var currentMachine={};
    currentMachine.id = d.id;
    currentMachine.x = ((event.clientX - xMouseDiff)-((event.clientX - xMouseDiff)%10));
    currentMachine.y = ((event.clientY - yMouseDiff)-((event.clientY - yMouseDiff)%10));
    d3.request("./design_mode/machine_wr.php")
     .mimeType("application/json")
     .header("Content-Type","application/x-www-form-urlencoded")
     .response(function(phpResponseObj) {
      allowSSE = "yes";
      var obj2 = {"design_id":designId};
      d3.request("./design_mode/design_rd.php")
       .mimeType("application/json")
       .header("Content-Type","application/x-www-form-urlencoded")
       .response(function(phpResponseObj) {
        jsonDesign=JSON.parse(phpResponseObj.response);
        jsonDesign["storages"].forEach((d,i) => {
         storageX[d.id]=d.x;
```

```
            storageY[d.id]=d.y;
           });
           jsonDesign["machines"].forEach((d,i) => {
            machineX[d.id]=d.x;
            machineY[d.id]=d.y;
           });
          })
          .send("POST","q="+JSON.stringify(obj2));
        })
        .send("POST","x="+JSON.stringify(currentMachine));
    allowSSE = "yes";
   }
  }
}
//G1-background frame
machineG1.append("rect")
    .attr("x","0")
    .attr("y","0")
    .attr("width","100")
    .attr("height",(20+20*(d.ins.length<d.outs.length?d.outs.length:d.ins.length)))
    .attr("opacity","0.8")
    .attr("fill","#003d44")
    .attr("filter","url(#shadow)");

//G1-name text
machineG1.append("text")
    .attr("x","50")
    .attr("y",((20+20*(d.ins.length<d.outs.length?d.outs.length:d.ins.length))/2))
    .attr("class","machineTitle")
    .attr("font-family","Open Sans Condensed")
    .attr("font-size","12px")
    .attr("fill","#fff")
```

```
        .attr("text-anchor","middle")
        .attr("alignment-baseline","middle")
        .text(d.name);
    //input pads
    d.ins.forEach((dd,ii) => {
     machineG1.append("circle")
        .attr("id","pad"+dd.id)
        .attr("cx","5")
        .attr("cy",(20*dd.row_num))
        .attr("r","4")
        .attr("stroke","black")
        .attr("stroke-width","1")
        .attr("fill",(dd.color==""?"#bedee1":dd.color))
        .on("contextmenu", function () {
         d3.event.preventDefault();
         d3.event.stopPropagation();
//                                                    if    (machStorEdit=="yes")
updateMachinePad(designId,dd.storage_id,'in',dd.a,dd.id,'','');
         updateMachinePad(designId,dd.storage_id,'in',dd.a,dd.id,'','');
        });
    });
    //output pads
    d.outs.forEach((dd,ii) => {
     machineG1.append("circle")
        .attr("id","pad"+dd.id)
        .attr("cx","95")
        .attr("cy",(20*dd.row_num))
        .attr("r","4")
        .attr("stroke","black")
        .attr("stroke-width","1")
        .attr("fill",(dd.color==""?"#bedee1":dd.color))
        .on("contextmenu", function () {
```

```
        d3.event.preventDefault();
        d3.event.stopPropagation();
// if (machStorEdit=="yes") updateMachinePad(designId,dd.storage_id,'out',dd.a,dd.id,dd.prod_time,dd.out_state);
        updateMachinePad(designId,dd.storage_id,'out',dd.a,dd.id,dd.prod_time,dd.out_state);
      });
   });
 });
}
```

Function: operate

Description: Logical operation of a design

```
function operate() {
 var n = 0;
 var ok2in = '0';
 for (var mId in machSt) {
  //output operation
  for (var mOut in machSt[mId].outs) {
   //n - last state bit
   n = machSt[mId].outs[mOut].prodTime;
   if (machSt[mId].outs[mOut].state[n].bit=="1") {
    if (machSt[mId].outs[mOut].storageId!="0") {
      if ((+storA[machSt[mId].outs[mOut].storageId].a + (+machSt[mId].outs[mOut].a)) <= (+storA[machSt[mId].outs[mOut].storageId].aMax)) {
                storA[machSt[mId].outs[mOut].storageId].a = (+storA[machSt[mId].outs[mOut].storageId].a)+(+machSt[mId].outs[mOut].a);
      } else console.log('Warning storage '+machSt[mId].outs[mOut].storageId+' overflow');
    } else  console.log('Warning machine '+mId+' output '+mOut+' not connected');
   };
```

```
  };
  //input operation
  ok2in = '0';
  if (machSt[mId].t2ni == 0) {
   ok2in = '1';
   for (var mIn in machSt[mId].ins) {
    if (machSt[mId].ins[mIn].storageId != "0") {
     if (+storA[machSt[mId].ins[mIn].storageId].a < +machSt[mId].ins[mIn].a) ok2in = '0';
    } else ok2in = '0';
   }
   if (ok2in == '1')
    for (var mIn in machSt[mId].ins) {
     if (machSt[mId].ins[mIn].storageId != "0") {
                                                    storA[machSt[mId].ins[mIn].storageId].a      =
(+storA[machSt[mId].ins[mIn].storageId].a) - (+machSt[mId].ins[mIn].a);
      } else console.log('Warning machine '+mId+' input '+mIn+' not connected');
     }
  }
  //change states operation
  for (var mOut in machSt[mId].outs) {
   //n - last state bit
   n = machSt[mId].outs[mOut].prodTime;
   machSt[mId].outs[mOut].state[0].bit = ok2in;
   for (var i=(+n); i>0; i--)
    machSt[mId].outs[mOut].state[i].bit = machSt[mId].outs[mOut].state[i-1].bit;
   console.log("t2ni"+machSt[mId].t2ni);
   if (ok2in == '1')
    machSt[mId].t2ni = +machSt[mId].inputTime-1;
   else if (machSt[mId].t2ni != 0)
    machSt[mId].t2ni--;
                                                    machSt[mId].outs[mOut].stateN          =
binArr2Num(JSON.stringify(machSt[mId].outs[mOut].state));
```

```
  }
 };
 for (var sId in storA) {
  for (var p=0; p<20; p++) {
   storA[sId].bars[+p+1].bit=(storA[sId].a>=(p+1)*(storA[sId].aMax/20)?"1":"0");
  }
 }
}
```

Function: saveVersion

Description: Save a design version

```
function saveVersion(user_id, design_id) {
 //call description modal
 $("#descriptionModal").modal();
 $('#description-modal-body').html('Write a short Description for the New Version');
      $('#description-modal-body2').html('<form class="form-vertical"><div class="form-
group"><div class="col-sm-12"><input type="text" id="textInput-markup1" class="form-
control"></div></div>');
 $("#description_ok").off('click').on('click', function() {
    $('#des_save_version_menu').removeClass("active");
    $("#descriptionModal").modal("hide");
//   console.log(design_id);
    $('#test_buttons').hide();
    var obj = {"design_id":design_id, "description":($("#textInput-markup1").val()).trim()};
    d3.request("./design_mode/save_version.php")
    .mimeType("application/json")
    .header("Content-Type","application/x-www-form-urlencoded")
    .send("POST","q="+JSON.stringify(obj));
 });
 $("#description_cancel").off('click').on('click', function() {
```

```
    $('#des_save_version_menu').removeClass("active");
 });
}
```

Function: setTimeResolution

Description: Setting design time resolution

```
function setTimeResolution(user_id,design_id) {
 var obj = {"design_id":design_id};
 d3.request("./design_mode/get_time_resolution.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {

  jsonTimeRes=JSON.parse(phpResponseObj.response);
//     console.log(jsonSimsTimes['timestart']);

  var options="";
  for (x in TIME_STEPS) {
                                  if     (x==jsonTimeRes)     options+='<option     value="'+x+'"
selected="selected">'+TIME_STEPS[x]+'</option>';
     else options+='<option value="'+x+'">'+TIME_STEPS[x]+'</option>';
  }

  var html_inbody = '\
    <div class="container-fluid">\
     <form class="form-horizontal">\
     <div class="form-group">\
      <div class="row">\
                        <label   class="col-sm-4   control-label"   for="textUpd-timestep">Time
Resolution</label>\
```

```
        <div class="col-sm-6">\
         <select id="textUpd-timestep" class="form-control">'+options+'\
         </select>\
        </div>\
       </div>\
      </div>\
      </form>\
     </div>';

    //call design time modal
    $("#timeModal").css("z-index", "8000").modal();
    $('#time-modal-body').html(html_inbody);
    $('#time-modal-msg').html('');
    $("#time-btn_ok").off('click').on('click', function (event, data) {

     var timestep = $("#textUpd-timestep").val();

     $("#timeModal").modal("hide");
     designTimeResolution=timestep;

     //send sql to update times
     var obj = {"design_id":design_id, "timestep":timestep};
     d3.request("./design_mode/update_time_resolution.php")
     .mimeType("application/json")
     .header("Content-Type","application/x-www-form-urlencoded")
     .response(function(phpResponseObj2) {
        $("#warningModal").modal();
         $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon
pficon-warning-triangle-o"></span><strong>Changes in Time Resolution affect the
parameters Next Input Time in machine and Production Time in machine out pad. Please
revise them accordingly.</strong></div>');
        $("#warning_ok").off('click').on('click', function() {
```

```
        $("#warningModal").modal("hide");//sse_source.close();
        })

    })
    .send("POST","q="+JSON.stringify(obj));

  });
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: createStorages

Description: Create a new carrier

```
function createStorages (undrag) {
 var jsonStorages = jsonDesign["storages"];

 var svg = d3.select("svg");

 //loop for each machine
 jsonStorages.forEach((d,i) => {
  //G (storage's up level group - includes everything)
  //creates all storages according to jsonStorages objects
  //-------------------------------------------------
  var storageG = svg.append("g")
        .attr("id","storage"+d.id)
        .attr("class","storage")
        .attr("cursor","pointer")
        .attr("transform","translate("+d.x+","+d.y+")");

  //G1 (includes the main box of the storage not the popup button and the popup frame)
```

```
//-------------------------------------------------------------------------------
if (undrag==true) {
 var storageG1 = storageG.append("g")
        .attr("transform","translate(0,3)")
        .on("contextmenu", function (data, i) {
          // react on right-clicking
           updateStorage(designId,d.id,d.name,d.a,d.aMax,d.color);
        });
}
else {
 var storageG1 = storageG.append("g")
        .attr("transform","translate(0,3)")
        .call(d3.drag()
         .on("start", dragStartFunc)
         .on("drag", dragFunc)
         .on("end", dragEndFunc))
        .on("contextmenu", function (data, i) {
          // react on right-clicking
           updateStorage(designId,d.id,d.name,d.a,d.aMax,d.color);
        });
  function dragStartFunc() {
   if (testDesignEnv=='no'){
    allowSSE = "no";
    d3.select(this.parentNode).raise();
                                        xMouseDiff = event.clientX -
d3.select(this.parentNode).node().transform.baseVal[0].matrix.e;
                                        yMouseDiff = event.clientY -
d3.select(this.parentNode).node().transform.baseVal[0].matrix.f;
   }
  }
  function dragFunc() {
   if (testDesignEnv=='no'){
```

```
    var dragX = ((event.clientX - xMouseDiff)-((event.clientX - xMouseDiff)%10));
    var dragY = ((event.clientY - yMouseDiff)-((event.clientY - yMouseDiff)%10));
    d3.select(this.parentNode).attr("transform", "translate("+dragX+","+dragY+")");
    updateLinesStor(d,dragX,dragY);
   }
  }
  function dragEndFunc() {
   if (testDesignEnv=='no'){
    var currentStorage={};
    currentStorage.id = d.id;
    currentStorage.x = ((event.clientX - xMouseDiff)-((event.clientX - xMouseDiff)%10));
    currentStorage.y = ((event.clientY - yMouseDiff)-((event.clientY - yMouseDiff)%10));
    d3.request("./design_mode/storage_wr.php")
     .mimeType("application/json")
     .header("Content-Type","application/x-www-form-urlencoded")
     .response(function(phpResponseObj) {
      allowSSE = "yes";
      var obj2 = {"design_id":designId};
      d3.request("./design_mode/design_rd.php")
      .mimeType("application/json")
      .header("Content-Type","application/x-www-form-urlencoded")
      .response(function(phpResponseObj) {
       jsonDesign=JSON.parse(phpResponseObj.response);
       jsonDesign["storages"].forEach((d,i) => {
        storageX[d.id]=d.x;
        storageY[d.id]=d.y;
       });
       jsonDesign["machines"].forEach((d,i) => {
        machineX[d.id]=d.x;
        machineY[d.id]=d.y;
       });
      })
```

```
        .send("POST","q="+JSON.stringify(obj2));
      })
      .send("POST","x="+JSON.stringify(currentStorage));
   }
  }
}
//G1-background frame
storageG1.append("rect")
    .attr("x","0")
    .attr("y","0")
    .attr("width","82")
    .attr("height","12")
    .attr("opacity","0.8")
    .attr("fill","#003d44")
    .attr("filter","url(#shadow)");
//G1-fullness bar
for (var p=0; p<20; p++) {
 storageG1.append("rect")
      .attr("id","storBar"+d.id+"-"+p)
      .attr("x",2+p*4)
      .attr("y","2")
      .attr("width","2")
      .attr("height","7")
      .attr("fill",((Math.round(d.a/d.aMax*100)>=((p+1)*5))?"#9ecf99":"#fff"));
}
//G1-name text
storageG1.append("text")
    .attr("x","41")
    .attr("y","-6")
    .attr("font-size","12px")
    .attr("fill","#003d44")
    .attr("text-anchor","middle")
```

```
        .attr("alignment-baseline","middle")
        .text(d.id+"-"+d.name);
   //G1-amounts text
   storageG1.append("text")
        .attr("id","storVals"+d.id)
        .attr("x","41")
        .attr("y","21")
        .attr("font-size","12px")
        .attr("fill","#003d44")
        .attr("text-anchor","middle")
        .attr("alignment-baseline","middle")
        .text(d.a+"/"+d.aMax);
  })
}
```

Function: testPlay

Description: Start the logical test at the Design Mode

```
function testPlay(user_id,design_id) {
  $('#test_step_back_button').hide();
  $('#test_play_button').hide();
  $('#test_step_for_button').hide();
  $('#test_stop_button').hide();
  $('#test_pause_button').show();
  $('#test_reset_button').hide();
  $('#test_close_button').hide();

  runTest = setInterval(function () {testStepForward(user_id,design_id);},2000);
}
```

Function: testPause

Description: Pause the logical test at the Design Mode

```
function testPause(user_id,design_id) {
 $('#test_step_back_button').show();
 $('#test_play_button').show();
 $('#test_step_for_button').show();
 $('#test_stop_button').hide();
 $('#test_pause_button').hide();
 $('#test_reset_button').show();
 $('#test_close_button').show();

 clearInterval(runTest);
 runTest = "";
}
```

Function: testReset

Description: Reset the logical test at the Design Mode

```
function testReset(user_id,design_id) {
 $('#test_step_back_button').hide();
 $('#test_play_button').show();
 $('#test_step_for_button').show();
 $('#test_stop_button').hide();
 $('#test_pause_button').hide();

 //reset test time
 testTime = 0;
 $('#time_design').text(testTime);

 //read test data for testTime from db
 var obj = {"des_id":designId, "time":testTime};
 d3.request("./design_mode/test_db_rd.php")
```

```
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {
   testObj = JSON.parse(phpResponseObj.response);

   //update machine state colors
   for (var mId in machSt) {
    for (var mOut in machSt[mId].outs) {
     for (var mOutBit in machSt[mId].outs[mOut].state) {
      if (mOutBit!=0) {
       machSt[mId].outs[mOut].state[mOutBit].obj
                                                    .attr("opacity",
(stateBit(testObj.machTest[machSt[mId].outs[mOut].id].mach_state,
machSt[mId].outs[mOut].prodTime, mOutBit)=="1"?"1":"0.2"))
                .attr("fill",(stateBit(testObj.machTest[machSt[mId].outs[mOut].id].mach_state,
machSt[mId].outs[mOut].prodTime, mOutBit)=="1"?machSt[mId].outs[mOut].color:"#fff"));
        //reset state bits on machSt object
                                        machSt[mId].outs[mOut].state[mOutBit].bit    =
stateBit(testObj.machTest[machSt[mId].outs[mOut].id].mach_state,
machSt[mId].outs[mOut].prodTime, mOutBit);
       }
      };
      //reset stateN value on machSt object
                                            machSt[mId].outs[mOut].stateN       =
testObj.machTest[machSt[mId].outs[mOut].id].mach_state;
      //reset t2ni value on machSt object
      machSt[mId].t2ni = testObj.machTest[machSt[mId].outs[mOut].id].t2ni;
     };
    };

   //Update storage bar data
   for (var sId in storA) {
```

```
      for (var p=0; p<20; p++) {
        storA[sId].bars[p+1].obj
                                                    .attr("fill",((testObj.storTest[sId].a
>=((p+1)*(storA[sId].aMax/20)))?"#9ecf99":"#fff"));
        //reset bar bits on storA object
                                          storA[sId].bars[p+1].bit    =    ((testObj.storTest[sId].a
>=((p+1)*(storA[sId].aMax/20)))?"1":"0");
      };
      storA[sId].txtobj
        .text(testObj.storTest[sId].a+"/"+storA[sId].aMax);
      //reset a value on storA object
      storA[sId].a = testObj.storTest[sId].a;
    };

    //reset data in db tables: test_machine_log, test_storage_log
    var obj = {"des_id":designId, "machine_states":machSt, "storages_a":storA};
    d3.request("./design_mode/test_db_reset.php")
    .mimeType("application/json")
    .header("Content-Type","application/x-www-form-urlencoded")
    .response(function(phpResponseObj) {})
    .send("POST","q="+JSON.stringify(obj));
  })
  .send("POST","q="+JSON.stringify(obj));

}
```

Function: testStepBackward

Description: Step backward at the logical test at the Design Mode

```
function testStepBackward(user_id,design_id) {
 // if at start hide back and stop
```

```
if (testTime == 1) $('#test_step_back_button').hide();

//set test time
testTime = +testTime-1;
$('#time_design').text(testTime);

//read test data for testTime from db
var obj = {"des_id":designId, "time":testTime};
d3.request("./design_mode/test_db_rd.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
  testObj = JSON.parse(phpResponseObj.response);

  //update machine state colors
  for (var mId in machSt) {
   for (var mOut in machSt[mId].outs) {
    for (var mOutBit in machSt[mId].outs[mOut].state) {
     if (mOutBit!=0)
      machSt[mId].outs[mOut].state[mOutBit].obj
                                                    .attr("opacity",
(stateBit(testObj.machTest[machSt[mId].outs[mOut].id].mach_state,
machSt[mId].outs[mOut].prodTime, mOutBit)=="1"?"1":"0.2"))
                .attr("fill",(stateBit(testObj.machTest[machSt[mId].outs[mOut].id].mach_state,
machSt[mId].outs[mOut].prodTime, mOutBit)=="1"?machSt[mId].outs[mOut].color:"#fff"));
    };
   };
  };
  //Update storage bar data
  for (var sId in storA) {
   for (var p=0; p<20; p++) {
    storA[sId].bars[p+1].obj
```

```
                                                    .attr("fill",((testObj.storTest[sId].a
>=((p+1)*(storA[sId].aMax/20)))?"#9ecf99":"#fff"));
      };
      storA[sId].txtobj
        .text(testObj.storTest[sId].a+"/"+storA[sId].aMax);
    };

  })
  .send("POST","q="+JSON.stringify(obj));
}
```

Function: testStepForward

Description: Step forward at the logical test at the Design Mode

```
function testStepForward(user_id,design_id) {
 $('#test_step_back_button').show();
 if (runTest == "") {
  $('#test_reset_button').show();
  $('#test_step_back_button').show();
  $('#test_close_button').show();
 } else {
  $('#test_reset_button').hide();
  $('#test_step_back_button').hide();
  $('#test_close_button').hide();
 }

 //get max test time from db
 var obj = {"des_id":designId};
 d3.request("./design_mode/test_get_max_time.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
```

```
  .response(function(phpResponseObj) {
    var maxTime=JSON.parse(phpResponseObj.response);

    //set test time
    testTime = +testTime+1;
    $('#time_design').text(testTime);

    if ((+maxTime+1) == testTime) {
      operate();
      machUpdateStateColors();
      storUpdateBarData();

      //save test result to db
                var obj = {"des_id":designId, "time":testTime, "machine_states":machSt,
"storages_a":storA};
      d3.request("./design_mode/test_db_wr.php")
        .mimeType("application/json")
        .header("Content-Type","application/x-www-form-urlencoded")
        .response(function(phpResponseObj) {})
        .send("POST","q="+JSON.stringify(obj));
    } else {
      //read test data for testTime from db
      var obj = {"des_id":designId, "time":testTime};
      d3.request("./design_mode/test_db_rd.php")
        .mimeType("application/json")
        .header("Content-Type","application/x-www-form-urlencoded")
        .response(function(phpResponseObj) {
          testObj = JSON.parse(phpResponseObj.response);

          //update machine state colors
          for (var mId in machSt) {
            for (var mOut in machSt[mId].outs) {
```

```
            for (var mOutBit in machSt[mId].outs[mOut].state) {
              if (mOutBit!=0)
                machSt[mId].outs[mOut].state[mOutBit].obj
                                                              .attr("opacity",
(stateBit(testObj.machTest[machSt[mId].outs[mOut].id].mach_state,
machSt[mId].outs[mOut].prodTime, mOutBit)=="1"?"1":"0.2"))
                    .attr("fill",(stateBit(testObj.machTest[machSt[mId].outs[mOut].id].mach_state,
machSt[mId].outs[mOut].prodTime, mOutBit)=="1"?machSt[mId].outs[mOut].color:"#fff"));
            };
          };
        };
        //Update storage bar data
        for (var sId in storA) {
          for (var p=0; p<20; p++) {
            storA[sId].bars[p+1].obj
                                                              .attr("fill",((testObj.storTest[sId].a
>=((p+1)*(storA[sId].aMax/20)))?"#9ecf99":"#fff"));
            };
            storA[sId].txtobj
              .text(testObj.storTest[sId].a+"/"+storA[sId].aMax);
          };
        })
        .send("POST","q="+JSON.stringify(obj));
    }
  })
  .send("POST","q="+JSON.stringify(obj));
}
```

Function: testClose

Description: Close the logical test at the Design Mode

```
function testClose(user_id,design_id) {
 sse_source.close();
 d3.selectAll("polyline").remove();
 d3.selectAll(".machine").remove();
 d3.selectAll(".storage").remove();
 testDesignEnv = "no";
 $('#des_test_menu').removeClass("active");
 $('#design_list').show();
 $('#DesignArea').show();
 $('#design_info').show();
 $('#design_buttons').show();
 $('#test_buttons').hide();

 // set machStorEdit based on permission
 var obj = {"des_id":designId, "user_id":user_id};
 d3.request("./design_mode/get_permission.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
  perm = JSON.parse(phpResponseObj.response);
  if ((perm=="full") || (perm=="owner")) machStorEdit = "yes";
  else machStorEdit = "no";
  if (machStorEdit=="no") design_rd(designId,1);
  else if (machStorEdit=="yes") design_rd(designId);

  //remove test owner session from test_sse table in db
  var obj = {"des_id":designId};
  d3.request("./design_mode/test_db_sess_del.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {})
  .send("POST","q="+JSON.stringify(obj));
```

```
  //-----
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: testDesign

Description: Design Test

```
function testDesign(user_id) {
 if (testDesignEnv == "no") {
  currentMode = "dm-test";

  //buttons mamagement
  $('#des_test_menu').removeClass("active");
  design_test_info_buttons_hide();
  $('#design_buttons').hide();
  $('#design_list').hide();
  $('#DesignArea').show();
  $('#test_buttons').show();
  $('#time_info').show();
  $('#test_play_button').show();
  $('#test_step_for_button').show();
  $('#test_reset_button').show();
  $('#test_close_button').show();
  //entering to design test mode
  testDesignEnv = "yes";
  machStorEdit = "no";
  d3.selectAll(".machineTitle")
        .attr("fill","#000")
        .attr("y","-7");
 //reset test time
```

```
  testTime = 0;
  $('#time_design').text(testTime);

  var machineG1 = "";
  //Reset machine state bars and create MachSt json object
  machSt = {};
  var jsonMachines = jsonDesign["machines"];
  jsonMachines.forEach((dMach,iMach) => {
   machineG1 = d3.select("#machineG1"+dMach.id);
   machSt[dMach.id]={};
   machSt[dMach.id].inputTime=dMach.input_time;
   machSt[dMach.id].t2ni=0;
   machSt[dMach.id].ins={};
   dMach.ins.forEach((dMIn,iIn) => {
    machSt[dMach.id].ins[dMIn.row_num]={};
    machSt[dMach.id].ins[dMIn.row_num].storageId=dMIn.storage_id;
    machSt[dMach.id].ins[dMIn.row_num].a=dMIn.a;
   });
   machSt[dMach.id].outs={};
   dMach.outs.forEach((dMOut,iOut) => {
    machSt[dMach.id].outs[dMOut.row_num]={};
    machSt[dMach.id].outs[dMOut.row_num].id=dMOut.id;
    machSt[dMach.id].outs[dMOut.row_num].storageId=dMOut.storage_id;
    machSt[dMach.id].outs[dMOut.row_num].a=dMOut.a;
    machSt[dMach.id].outs[dMOut.row_num].prodTime=dMOut.prod_time;
     machSt[dMach.id].outs[dMOut.row_num].color=(dMOut.color==""?"#bedee1":dMOut.
color);
    machSt[dMach.id].outs[dMOut.row_num].stateN=dMOut.out_state;
          machSt[dMach.id].t2ni=Math.max(machSt[dMach.id].t2ni,(dMach.input_time-
(dMOut.prod_time-Math.floor(Math.log2(dMOut.out_state)))));
    machSt[dMach.id].outs[dMOut.row_num].state={};
    machSt[dMach.id].outs[dMOut.row_num].state[0]={};
```

```
      machSt[dMach.id].outs[dMOut.row_num].state[0].bit="";
      for (var i=1;i<=dMOut.prod_time;i++) {
       machSt[dMach.id].outs[dMOut.row_num].state[i]={};
        machSt[dMach.id].outs[dMOut.row_num].state[i].bit=stateBit(dMOut.out_state,dMOut.prod_time,i);
       machSt[dMach.id].outs[dMOut.row_num].state[i].obj = machineG1.append("rect")
        .attr("x",Math.round(15+(70*(i-1)/dMOut.prod_time)))
        .attr("y",Math.round(18+(iOut*20)))
              .attr("width",(Math.round(15+(70*i/dMOut.prod_time))-Math.round(15+(70*(i-1)/dMOut.prod_time))))
        .attr("height","4")
        .attr("opacity","0.2")
        .attr("filter","url(#shadow)")
        .attr("fill","#fff");
      };
    });
  });
  //create storA array
  storA = {};
  var jsonStorages = jsonDesign["storages"];
  jsonStorages.forEach((dStor,iStor) => {
    storA[dStor.id] = {};
    storA[dStor.id].a = dStor.a;
    storA[dStor.id].aMax = dStor.aMax;
    storA[dStor.id].bars = {};
    storA[dStor.id].txtobj = d3.select("#storVals"+dStor.id);
    for (var p=0; p<20; p++) {
     storA[dStor.id].bars[+p+1]={};
     storA[dStor.id].bars[+p+1].obj=d3.select("#storBar"+dStor.id+"-"+p);
     storA[dStor.id].bars[+p+1].bit=(dStor.a>=(p+1)*(dStor.aMax/20)?"1":"0");
    }
  });
```

```
  //reset data in db tables: test_machine_log, test_storage_log
  var obj = {"des_id":designId, "machine_states":machSt, "storages_a":storA};
  d3.request("./design_mode/test_db_reset.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {})
  .send("POST","q="+JSON.stringify(obj));

  machUpdateStateColors();
 }
}
```

Function: stateBit

Description: stateBit returns a bit of a state (number, length, bit#) (Design Test)

```
function stateBit(num, len, b) {
 var binNum = parseInt(num, 10).toString(2);
 var zero = len - binNum.length + 1;
 var binNum = Array(+(zero > 0 && zero)).join("0") + binNum;
 return binNum.substring(b-1,b);
}
```

Function: binArr2Num

Description: sbinArr2Num converts the binary state array to number. Binary input value must be sent through JSON.stringify function (Design Test)

```
function binArr2Num(strArr) {
 var bitArr = JSON.parse(strArr);
 var num = "";
 for (var d in bitArr) {
  if (d > 0) num = num+bitArr[d].bit;
```

```
 }
 return parseInt(num, 2);
}
```

Function: machUpdateStateColors

Description: Update machine state bar colors (Design Test)

```
function machUpdateStateColors() {
 for (var mId in machSt) {
  for (var mOut in machSt[mId].outs) {
   for (var mOutBit in machSt[mId].outs[mOut].state) {
    if (mOutBit!=0)
     machSt[mId].outs[mOut].state[mOutBit].obj
      .attr("opacity",(machSt[mId].outs[mOut].state[mOutBit].bit=="1"?"1":"0.2"))
                                  .attr("fill",(machSt[mId].outs[mOut].state[mOutBit].bit=="1"?
machSt[mId].outs[mOut].color:"#fff"));
   };
  };
 };
}
```

Function: machUpdateStateColors

Description: Update carrier's state bar colors (Design Test)

```
function storUpdateBarData() {
 for (var sId in storA) {
  for (var p=0; p<20; p++) {
   storA[sId].bars[p+1].obj
    .attr("fill",((storA[sId].bars[+p+1].bit=="1")?"#9ecf99":"#fff"));
  };
  storA[sId].txtobj
```

```
    .text(storA[sId].a+"/"+storA[sId].aMax);
 };
}
```

Function: updateMachine

Description: Updates machine dta

```
function updateMachine(design_id,machine_id,name,ins,outs,input_time,humans_min) {

        var time_step=TIME_STEPS[designTimeResolution];
        var options_ins="";
        for (i=0;i<=MACHINE_INS_OUTS_MAX;i++) {
                if (i==ins) options_ins+='<option value="'+i+'" selected="selected">'+i+'</option>';
                else options_ins+='<option value="'+i+'">'+i+'</option>';
        }
        var options_outs="";
        for (i=0;i<=MACHINE_INS_OUTS_MAX;i++) {
                if (i==outs)
                        options_outs+='<option value="'+i+'" selected="selected">'+i+'</option>';
                else
                        options_outs+='<option value="'+i+'">'+i+'</option>';
        }
        var html_inbody = '\
         <div class="container-fluid">\
                <form class="form-horizontal">\
                <div class="form-group">\
                        <div class="row">\
                                <label class="col-sm-5 control-label" for="textUpd-markup">Machine Name</label>\
```

```
                    <div class="col-sm-7"><input type="text" id="textUpd-
markup" class="form-control"  value="'+name+'"></div>\
                </div>\
                <div class="row">\
                    <label  class="col-sm-5  control-label"  for="textUpd-
markup-in">Inputs</label>\
                    <div class="col-sm-3">\
                        <select id="textUpd-markup-in" class="form-
control">'+options_ins+'\
                        </select>\
                    </div>\
                </div>\
                <div class="row">\
                    <label  class="col-sm-5  control-label"  for="textUpd-
markup-out">Outputs</label>\
                    <div class="col-sm-3">\
                        <select              id="textUpd-markup-out"
class="form-control">'+options_outs+'\
                        </select>\
                    </div>\
                </div>\
                <div class="row">\
                    <label  class="col-sm-5  control-label"  for="textUpd-
markup-input-time">Next Input Time</label>\
                    <div class="col-sm-3"><input type="text" id="textUpd-
markup-input-time" class="form-control" value="'+input_time+'"></div>\
                    <div><span              style="font-size:10px;vertical-
align:bottom;">'+time_step+'(s)</span></div>\
                </div>\
                <div class="row">\
                    <label  class="col-sm-5  control-label"  for="textUpd-
markup-humans-min">Required Humans</label>\
```

```
                                        <div class="col-sm-3"><input type="text" id="textUpd-
markup-humans-min" class="form-control" value="'+humans_min+'"></div>\
                                    </div>\
                                </div>\
                                </form>\
                            </div>';

                        //call update modal
                        $("#updateMachineModal").css("z-index", "8000").modal();
                        $('#update-machine-modal-body').html(html_inbody);
                        $('#update-machine-modal-msg').html('');
                        if (machStorEdit=="no") {
                                $("#update-machine-btn_ok").hide();
                                $("#delete-machine-btn_ok").hide();
                                $('#update-machine-modal-header').html('<button type="button"
class="close" data-dismiss="modal">&times;</button><h4 class="modal-title">View
Machine</h4>');
                        }
                        else if (machStorEdit=="yes"){
                                $("#update-machine-btn_ok").off('click').on('click', function (event, data)
{

                                        var name = ($("#textUpd-markup").val()).trim();
                                        var inputs = $("#textUpd-markup-in").val();
                                        var outputs = $("#textUpd-markup-out").val();
                                        var input_time = $("#textUpd-markup-input-time").val();
                                        var humans_min = $("#textUpd-markup-humans-min").val();
                                        if ((name!='')&&((name.match(/^\s+$/) === null))
                                                &&($.isNumeric(input_time))
                                                &&(Number(input_time)>0)
                                                &&($.isNumeric(humans_min))
                                                &&(Number(humans_min)>=0)
```

```
                    ) {
                                        $("#updateMachineModal").modal("hide");
                                        //send sql to update machine
                                        var obj = {"design_id":design_id,
"machine_id":machine_id, "name":name, "inputs":inputs, "outputs":outputs,
"input_time":input_time, "humans_min":humans_min};

            d3.request("./design_mode/update_machine.php")
                                        .mimeType("application/json")
                                        .header("Content-Type","application/x-
www-form-urlencoded")
                                        .response(function(phpResponseObj2) {
                                                sse_source.close();
                                                d3.selectAll("polyline").remove();
                                                d3.selectAll(".machine").remove();
                                                d3.selectAll(".storage").remove();
                                                design_rd(design_id);

                                        })
                                        .send("POST","q="+JSON.stringify(obj));

                    } else {
                                        $('#update-machine-modal-msg').html('<div
class="form-group has-error"><div class="col-sm-12"><span class="help-block">Please
select numeric values for Time ( >0 ) and Humans ( >=0).</span></div></div>');
                    }
            });
            $("#delete-machine-btn_ok").off('click').on('click', function (event, data)
{
                    $("#warningModal").css("z-index", "10000").modal();
                    $('#warning-modal-body').html('<div class="alert alert-
warning"><span class="pficon pficon-warning-triangle-o"></span><strong>A Machine will
```

```
be deleted! Are you sure you want to continue?</strong></div>');
                        $("#warning_ok").off('click').on('click', function() {
                                $("#warningModal").modal("hide");
                                $("#updateMachineModal").modal("hide");
                                var obj4 = {"machine_id":machine_id};
                                d3.request("./design_mode/delete_machine.php")
                                .mimeType("application/json")
                                .header("Content-Type","application/x-www-form-
urlencoded")
                                .response(function(phpResponseObj2) {
                                        sse_source.close();
                                        design_rd(design_id);
                                }).send("POST","q="+JSON.stringify(obj4));
                        })
                });
        }
}
```

Function: updateStorage

Description: Updates carrier's data

```
function
updateStorage(design_id,storage_id,product_name,storage_a,storage_aMax,storage_color) {

 var obj = {"design_id":design_id,"product_name":product_name};
 d3.request("./design_mode/get_products.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {

//   read Products from db to display list ...
```

```
var options=phpResponseObj.response;

var html_inbody = '\
<div class="container-fluid">\
 <form class="form-horizontal">\
  <div class="form-group">\
   <div class="row">\
            <label class="col-sm-4 control-label" for="textUpd-product_id">Product to carry</label>\
    <div class="col-sm-3" id="product_list">\
             <select id="textUpd-product_id" class="form-control" style="width: 200px;">'+options+'</select>\
    </div>\
   </div>\
   <div class="row">\
             <label class="col-sm-4 control-label" for="textUpd-amount">Current Amount</label>\
     <div class="col-sm-4"><input type="text" id="textUpd-amount" class="form-control" value="'+storage_a+'"></div>\
   </div>\
   <div class="row">\
             <label class="col-sm-4 control-label" for="textUpd-amountMax">Max Amount</label>\
       <div class="col-sm-4"><input type="text" id="textUpd-amountMax" class="form-control" value="'+storage_aMax+'"></div>\
   </div>\
   <div class="row">\
    <label class="col-sm-4 control-label" for="textUpd-color">Line Color</label>\
      <div class="col-sm-7"><input type="color" id="textUpd-color" class="form-control" value="'+storage_color+'"></div>\
   </div>\
  </div>\
```

```
  </form>\
  </div>';

  //call new modal
  $("#updateStorageModal").css("z-index", "8000").modal();
  $('#update-storage-modal-body').html(html_inbody);
  $('#update-storage-modal-msg').html('');
  if (machStorEdit=="no") {
   $("#update-storage-btn_ok").hide();
   $("#delete-storage-btn_ok").hide();
       $('#update-storage-modal-header').html('<button type="button" class="close" data-dismiss="modal">&times;</button><h4 class="modal-title">View Storage</h4>');
  }
  else if (machStorEdit=="yes"){
   $("#update-storage-btn_ok").off('click').on('click', function (event, data) {

    if ((Number($("#textUpd-amountMax").val())>=Number($("#textUpd-amount").val()))
     &&($.isNumeric($("#textUpd-amount").val()))
     &&($.isNumeric($("#textUpd-amountMax").val()))
     &&(Number($("#textUpd-amount").val())>=0)
     &&(Number($("#textUpd-amountMax").val())>0)) {
     $("#updateStorageModal").modal("hide");

     //send sql to update new storage
              var obj2 = {"product_id":$("#textUpd-product_id").val(), "a":$("#textUpd-amount").val(), "aMax":$("#textUpd-amountMax").val(), "color":$("#textUpd-color").val(), "storage_id":storage_id};

     d3.request("./design_mode/update_storage.php")
     .mimeType("application/json")
     .header("Content-Type","application/x-www-form-urlencoded")
     .response(function(phpResponseObj2) {
```

```
        sse_source.close();
        d3.selectAll("polyline").remove();
        d3.selectAll(".machine").remove();
        d3.selectAll(".storage").remove();
        design_rd(design_id);
      })
      .send("POST","q="+JSON.stringify(obj2));
    }
    else{
            $('#update-storage-modal-msg').html('<div class="form-group has-error"><div class="col-sm-10"><span class="help-block">Please select numeric values for amounts (initial >= 0, max > 0 and max >= initial).</span></div></div>');
    }
   });
   $("#delete-storage-btn_ok").off('click').on('click', function (event, data) {
    $("#warningModal").css("z-index", "10000").modal();
      $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon pficon-warning-triangle-o"></span><strong>A Carrier will be deleted! Are you sure you want to continue?</strong></div>');
    $("#warning_ok").off('click').on('click', function() {
     $("#warningModal").modal("hide");
     $("#updateStorageModal").modal("hide");
     var obj4 = {"storage_id":storage_id};
     d3.request("./design_mode/delete_storage.php")
     .mimeType("application/json")
     .header("Content-Type","application/x-www-form-urlencoded")
     .response(function(phpResponseObj2) {
      sse_source.close();
      design_rd(design_id);
     }).send("POST","q="+JSON.stringify(obj4));
    })
```

```
    });
   }
  })
  .send("POST","q="+JSON.stringify(obj));
}
```

Function: version_rd

Description: Read version data

```
function version_rd(version_id, undrag) {
    machStorEdit="no";
    var obj = {"version_id":version_id};
    d3.request("./design_mode/version_rd.php")
      .mimeType("application/json")
      .header("Content-Type","application/x-www-form-urlencoded")
      .response(function(phpResponseObj) {
       jsonDesign=JSON.parse(phpResponseObj.response);
       d3.select("body").append("svg")
           .attr("height","100%")
           .attr("width","100%")
           .attr("id","SVG_ID")
           .on("contextmenu", function (data, i) {
            d3.event.preventDefault();
           });
       d3.select("svg").append("defs");
       defineShadow();
       createLines();
       createMachines(undrag);
       createStorages(undrag);
      }).send("POST","q="+JSON.stringify(obj));
}
```

Function: viewDesign

Description: view a design

```
function viewDesign(user_id) {
 var obj = {"user_id":user_id};
 d3.request("./design_mode/get_designs_and_versions.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
//   read Designes from db to select for opening ...
   jsonDesignes=JSON.parse(phpResponseObj.response);
   //call open modal
   $('#des_view_menu').removeClass("active");
   $("#viewModal").modal();
   $('#view-modal-body').treeview({
   collapseIcon: "fa fa-angle-down",
   data: jsonDesignes,
   expandIcon: "fa fa-angle-right",
   nodeIcon: "fa fa-folder",
   onNodeSelected: function(event, data) {
    $("#viewModal").modal("hide");

// hide the old design, the buttons and info
    hide_design();

 // show menu
    $('#selected_design_name').text(data.name);
    designId=data.des_id;
    $('#design_info').show();
    $('#des_close_menu').show();
```

```
    machStorEdit="no";

 //code to read and display design ...
 //acquire objects in case of design selected
    if (data.ver_id=='design_selected'){
      sse_source.close();
      design_rd(data.des_id,true);
    }
//code to read and display version ...
//acquire objects in case of version selected
    else {
      sse_source.close();
      version_rd(data.ver_id,true);
    }
   },
   levels:1,
   showBorder: false
   });
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: closeSim

Description: Simulation close

```
function closeSim(user_id) {
 //hide design, buttons and info
 d3.select("#SVG_ID").remove();
 $('#selected_sim_name').text('');
 simId='';
 $('#sim_info').hide();
```

```
 $('#sim_buttons').hide();
 $('#run_buttons').hide();
 sim_v1_menu_hide();
 sim_v2_menu_hide();
 // show menu
 sim_v1_menu_show();
}
```

Function: deleteSim

Description: Delete a simulator

```
function deleteSim(user_id) {
 var obj = {"user_id":user_id};
 d3.request("./sim_mode/get_own_sims_and_versions.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {

//   read Sims from db to select for delete ...
   jsonSims=JSON.parse(phpResponseObj.response);

   //call delete sim modal
   $('#sim_delete_menu').removeClass("active");
   $("#simDeleteModal").modal();
   $('#sim-delete-modal-msg').html();
   $('#sim-delete-modal-body').treeview({
   collapseIcon: "fa fa-angle-down",
   data: jsonSims,
   expandIcon: "fa fa-angle-right",
   nodeIcon: "fa fa-folder",
   onNodeSelected: function(event, data) {
```

```
    //code to make inactive sim and all its versions...
     if (data.ver_id=='sim_selected'){
      $("#warningModal").css("z-index", "8000").modal();
        $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon pficon-warning-triangle-o"></span><strong>Simulator \''+data.name+'\' and all its runs will be deleted! Are you sure you want to continue?</strong></div>');
      $("#warning_ok").off('click').on('click', function() {

      $("#warningModal").modal("hide");
      $("#simDeleteModal").modal("hide");
      var obj3 = {"sim_id":data.sim_id};
      d3.request("./sim_mode/sim_inactive.php")
       .mimeType("application/json")
       .header("Content-Type","application/x-www-form-urlencoded")
       .response(function(phpResponseObj3) {
        var jsonDesign3=JSON.parse(phpResponseObj3.response);
       }).send("POST","q="+JSON.stringify(obj3));
      })
     }
     //code to make inactive version ...
     else {
      $("#warningModal").css("z-index", "8000").modal();
        $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon pficon-warning-triangle-o"></span><strong>Run '+data.number+' of simulator \''+data.name+'\' will be deleted! Are you sure you want to continue? </strong></div>');
      $("#warning_ok").off('click').on('click', function() {

      $("#warningModal").modal("hide");
      $("#simDeleteModal").modal("hide");
      var obj2 = {"version_id":data.ver_id};
```

```
        d3.request("./sim_mode/sim_version_inactive.php")
         .mimeType("application/json")
         .header("Content-Type","application/x-www-form-urlencoded")
         .response(function(phpResponseObj2) {
          var jsonDesign4=JSON.parse(phpResponseObj2.response);
         }).send("POST","q="+JSON.stringify(obj2));
        })
       }

     },
   levels:1,
   showBorder: false
   });
  })
  .send("POST","q="+JSON.stringify(obj));
}
```

Function: humanSchedule

Description: Create a new human schedule list

```
function humanSchedule(user_id,sim_id,table_add_mod) {

 var obj = {"sim_id":sim_id};
 d3.request("./sim_mode/human_schedule.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {

  jsonOrders=JSON.parse(phpResponseObj.response);
//    console.log(jsonOrders);
    jsonOrdersData=jsonOrders['orders'];
```

```
    jsonOrdersSim=jsonOrders['sim'];

  var options="";
  for (x in WORKING_HOURS) {
        if (WORKING_HOURS[x]=='8') options+='<option value="'+WORKING_HOURS[x]+'" selected="selected">'+WORKING_HOURS[x]+'</option>';
         else options+='<option value="'+WORKING_HOURS[x]+'">'+WORKING_HOURS[x]+'</option>';
  }

  $(function () {
      $('#datepicker_huo1').datetimepicker({date:jsonOrdersSim[0].timestart, format: 'DD-MM-YYYY'});
       $('#datepicker_huo2').datetimepicker({date:jsonOrdersSim[0].timeend, format: 'DD-MM-YYYY'});
            $('#datepicker_huo3').datetimepicker({date:'1900-01-01 08:00:00', format: 'HH:mm:ss'});
    $('#datepicker_huo4').datetimepicker({format: 'DD-MM-YYYY'});
    $('#datepicker_huo5').datetimepicker({format: 'DD-MM-YYYY'});
    $('#datepicker_huo6').datetimepicker({format: 'HH:mm:ss'});
  });

  var html_table_add='\
    <h3>Add Human Schedule Line</h3>\
        <table style="width:100%" class="table table-striped table-bordered table-hover" id="human_schedule_table2">\
    <thead><tr>\
     <th>Persons</th>\
     <th>Work Start</th>\
     <th>Hours</th>\
     <th>Date From</th>\
     <th>Date To</th>\
```

```
 <th>Actions</th>\
</tr></thead><tr>\
<td style="padding-left:0px;padding-right:0px;">\
<div style="padding-left:0px;">\
 <input id="textHuman-persons" class="form-control " ></input>\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;">\
<div class="input-group date" id="datepicker_huo3">\
 <div class="input-group-addon">\
  <span class="glyphicon glyphicon-calendar"></span>\
 </div>\
 <input type="text" class="form-control" id="textHuman-start" value="">\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;">\
<div style="padding-left:0px;">\
 <select id="textHuman-hours" class="form-control " >'+options+'\
 </select>\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;">\
<div class="input-group date" id="datepicker_huo1">\
 <div class="input-group-addon">\
  <span class="glyphicon glyphicon-calendar"></span>\
 </div>\
 <input type="text" class="form-control" id="textHuman-datefrom" value="'+''+'">\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;">\
<div class="input-group date" id="datepicker_huo2">\
 <div class="input-group-addon">\
```

```
      <span class="glyphicon glyphicon-calendar"></span>\
     </div>\
     <input type="text" class="form-control" id="textHuman-dateto" value="'+''+'">\
    </div>\
    </td>\
    <td style="padding-left:0px;padding-right:0px;width:75px;">\
            <div><button class="btn btn-default dropdown-toggle" type="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="true" style="padding-left:0px;padding-right:0px;width:75px"\
        onclick=add_human_schedule('+user_id+','+sim_id+',$("#textHuman-persons").val(),$("#textHuman-start").val(),$("#textHuman-hours").val(),$("#textHuman-datefrom").val(),$("#textHuman-dateto").val());>Add</button></div>\
    </td>\
    </tr>\
    </table>';
  var html_table='\
     <h3>Human Schedule</h3><table style="width:100%" class="table table-striped table-bordered table-hover" id="human_schedule_table">\
    <thead><tr>\
     <th>Persons   </th>\
     <th>Work Start   </th>\
     <th>Hours   </th>\
     <th>From   </th>\
     <th>To   </th>\
                                        <th style="padding-left:24px;padding-right:25px;">Actions      </th>\
    </tr></thead></table>';

    //call sim human schedule modal
    $("#simHumanScheduleModal").css("z-index", "8000").modal();

    if ((table_add_mod==undefined)||(table_add_mod==''))
```

```
      $('#sim-human-schedule-modal-body').html(html_table_add+html_table);
    else $('#sim-human-schedule-modal-body').html(table_add_mod+html_table);
//$('#sim-human-schedule-modal-body').html(html_table_add+html_table);
    $('#human_schedule_table').DataTable({
     data: jsonOrdersData,
     columns: [
      { data: "persons" },
      { data: "time_start" },
      { data: "hours" },
      { data: "date_from" },
      { data: "date_to" },
      { data: null,
       searchable: false ,
       className: "table-view-pf-actions",
       render: function (data, type, full, meta) {
       // Inline action kebab renderer
                                                return '<div><button type="button"
onclick=modify_human_schedule('+user_id+','+sim_id+','+data['id']+','+data['persons']
+',"'+data['time_start']+'",'+data['hours']+',"'+data['df']+'","'+data['dt']+'");>' +
         'Modify</button>\
                                                    <button type="button"
onclick=delete_human_schedule('+user_id+','+sim_id+','+data['id']+');>' +
         'Delete</button></div>';
        }
       }
     ]
    });
  })
  .send("POST","q="+JSON.stringify(obj));
}
```

Function: humanSchedule

Description: Add a human schedule to a human schedule list

```
function add_human_schedule(user_id, sim_id, persons, time_start, hours, date_from, date_to){
        date_from=moment(date_from, "DD/MM/YYYY HH:mm:ss").format('YYYY-MM-DDTHH:mm:ss');
 date_to=moment(date_to, "DD/MM/YYYY HH:mm:ss").format('YYYY-MM-DDTHH:mm:ss');
 if ((date_from<=date_to)&&(persons>0)){
    var obj = {"sim_id":sim_id, "persons":persons, "time_start":time_start, "hours":hours, "date_from":date_from, "date_to":date_to};
  d3.request("./sim_mode/add_human_schedule.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {
   humanSchedule(user_id,sim_id,'');
  })
  .send("POST","q="+JSON.stringify(obj));
 }
 else {
  $("#warningModal").css("z-index", "10000").modal();
      $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon pficon-warning-triangle-o"></span><strong>Not added due to invalid values.</strong></div>');
  $("#warning_ok").off('click').on('click', function() {
   $("#warningModal").modal("hide");
  })
 }
}
```

Function: delete_human_schedule

Description: Delete a human schedule from a human schedule list

```
function delete_human_schedule(user_id,sim_id,sim_human_order_id){
 $("#warningModal").css("z-index", "10000").modal();
    $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon pficon-warning-triangle-o"></span><strong>A Human Schedule Line will be deleted! Are you sure you want to continue?</strong></div>');
 $("#warning_ok").off('click').on('click', function() {
  $("#warningModal").modal("hide");
  var obj = {"sim_human_order_id":sim_human_order_id};
  d3.request("./sim_mode/delete_human_schedule.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {
   humanSchedule(user_id,sim_id);
  })
  .send("POST","q="+JSON.stringify(obj));
 })
}
```

Function: modify_human_schedule

Description: Modify (update) a human schedule to a human schedule list

```
function modify_human_schedule(user_id, sim_id, id, persons, time_start, hours, date_from, date_to){
  var options="";
  for (x in WORKING_HOURS) {
      if (WORKING_HOURS[x]==hours) options+='<option value="'+WORKING_HOURS[x]+'" selected="selected">'+WORKING_HOURS[x]+'</option>';
          else options+='<option value="'+WORKING_HOURS[x]+'">'+WORKING_HOURS[x]+'</option>';
  }
```

```
//var options='<option value="4">4</option><option value="6">6</option><option value="8" selected="selected">8</option><option value="10">10</option><option value="12">12</option>';
var html_table_modify='\
<h3>Modify Human Schedule Line</h3>\
<table style="width:100%" class="table table-striped table-bordered table-hover" id="human_schedule_table_mod">\
<thead><tr>\
<th>Persons</th>\
<th>Work Start</th>\
<th>Hours</th>\
<th>Date From</th>\
<th>Date To</th>\
<th>Actions</th>\
</tr></thead><tr>\
<td style="padding-left:0px;padding-right:0px;">\
<div style="padding-left:0px;">\
<input id="modHuman-persons" class="form-control" value="'+persons+'"></input>\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;">\
<div class="input-group date" id="datepicker_huo6">\
<div class="input-group-addon">\
<span class="glyphicon glyphicon-calendar"></span>\
</div>\
<input type="text" class="form-control" id="modHuman-start" value="'+time_start+'">\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;">\
<div style="padding-left:0px;">\
<select id="modHuman-hours" class="form-control " >'+options+'\
</select>\
```

```
    </div>\
    </td>\
    <td style="padding-left:0px;padding-right:0px;">\
    <div class="input-group date" id="datepicker_huo4">\
     <div class="input-group-addon">\
      <span class="glyphicon glyphicon-calendar"></span>\
     </div>\
                        <input type="text" class="form-control" id="modHuman-datefrom"
value="'+date_from+'">\
    </div>\
    </td>\
    <td style="padding-left:0px;padding-right:0px;">\
    <div class="input-group date" id="datepicker_huo5">\
     <div class="input-group-addon">\
      <span class="glyphicon glyphicon-calendar"></span>\
     </div>\
     <input type="text" class="form-control" id="modHuman-dateto" value="'+date_to+'">\
    </div>\
    </td>\
    <td style="padding-left:0px;padding-right:0px;width:75px;">\
            <div><button class="btn btn-default dropdown-toggle" type="button" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="true" style="padding-
left:0px;padding-right:0px;width:75px"\
                    onclick=save_human_schedule('+user_id+','+sim_id+','+id+',$("#modHuman-
persons").val(),$("#modHuman-start").val(),$("#modHuman-hours").val(),$("#modHuman-
datefrom").val(),$("#modHuman-dateto").val());>Save</button></div>\
    </td>\
    </tr>\
    </table>';
    humanSchedule(user_id, sim_id, html_table_modify);
}
```

Function: save_human_schedule

Description: Save changes of a human schedule to a human schedule list

```
function save_human_schedule(user_id, sim_id, id, persons, time_start, hours, date_from, date_to){
        date_from=moment(date_from, "DD/MM/YYYY HH:mm:ss").format('YYYY-MM-DDTHH:mm:ss');
 date_to=moment(date_to, "DD/MM/YYYY HH:mm:ss").format('YYYY-MM-DDTHH:mm:ss');
 if ((date_from<=date_to)&&(persons>0)){
        var obj = {"sim_id":sim_id, "persons":persons, "id":id, "time_start":time_start, "hours":hours, "date_from":date_from, "date_to":date_to};
   d3.request("./sim_mode/save_human_schedule.php")
   .mimeType("application/json")
   .header("Content-Type","application/x-www-form-urlencoded")
   .response(function(phpResponseObj) {
    humanSchedule(user_id,sim_id);
   })
   .send("POST","q="+JSON.stringify(obj));
 }
 else {
   $("#warningModal").css("z-index", "10000").modal();
       $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon pficon-warning-triangle-o"></span><strong>Not added due to invalid values.</strong></div>');
   $("#warning_ok").off('click').on('click', function() {
    $("#warningModal").modal("hide");
   })
 }
}
```

Function: machinesSchedule

Description: Create a new machines schedule list

```
function machinesSchedule(user_id,sim_id) {

 var obj = {"sim_id":sim_id};
 d3.request("./sim_mode/machines_schedule.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {

  jsonOrders=JSON.parse(phpResponseObj.response);
   console.log(jsonOrders);
   jsonOrdersData=jsonOrders['orders'];
   jsonOrdersMachines=jsonOrders['machines'];
  jsonOrdersSim=jsonOrders['sim'];

  var options="";
  for (x in jsonOrdersMachines) {
                                                    options+='<option
value="'+jsonOrdersMachines[x].mach_id+'">'+jsonOrdersMachines[x].name+'</option>';
  }
  $(function () {
          $('#datetimepicker_machsch1').datetimepicker({date:jsonOrdersSim[0].timestart,
format: 'DD-MM-YYYY HH:mm:ss'});
   $('#datetimepicker_machsch2').datetimepicker({date:jsonOrdersSim[0].timeend, format:
'DD-MM-YYYY HH:mm:ss'});
  });

  var html_table='\
   <h3>Add Machines Schedule Line</h3>\
       <table style="width:100%" class="table table-striped table-bordered table-hover"
id="machines_schedule_table2">\
```

```
<thead><tr>\
 <th>Machine</th>\
 <th>Date From</th>\
 <th>Date To</th>\
 <th>Actions</th>\
</tr></thead><tr>\
<td style="padding-left:0px;padding-right:0px;">\
<div style="padding-left:0px;">\
 <select id="textMachSchedule-machine" class="form-control " >'+options+'\
 </select>\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;">\
<div class="input-group date" id="datetimepicker_machsch1">\
 <div class="input-group-addon">\
  <span class="glyphicon glyphicon-calendar"></span>\
 </div>\
        <input type="text" class="form-control" id="textMachSchedule-datefrom" value="'+''+'">\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;">\
<div class="input-group date" id="datetimepicker_machsch2">\
 <div class="input-group-addon">\
  <span class="glyphicon glyphicon-calendar"></span>\
 </div>\
 <input type="text" class="form-control" id="textMachSchedule-dateto" value="'+''+'">\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;width:75px;">\
      <div><button class="btn btn-default dropdown-toggle" type="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="true" style="padding-
```

```
left:0px;padding-right:0px;width:75px"\
                onclick=add_machines_schedule('+user_id+','+sim_id+',$("#textMachSchedule-machine").val(),$("#textMachSchedule-datefrom").val(),$("#textMachSchedule-dateto").val());>Add</button></div>\
    </td>\
    </tr>\
    </table>\
    <h3>Machines Schedule</h3><table style="width:100%" class="table table-striped table-bordered table-hover" id="machines_schedule_table">\
    <thead><tr>\
     <th>Machine   </th>\
     <th>From   </th>\
     <th>To   </th>\
     <th>Actions      </th>\
    </tr></thead></table>';

    //call sim macines schedule modal
    $("#simMachinesScheduleModal").css("z-index", "8000").modal();
    $('#sim-machines-schedule-modal-body').html(html_table);
    $('#machines_schedule_table').DataTable({
     data: jsonOrdersData,
     columns: [
      { data: "name" },
      { data: "date_from" },
      { data: "date_to" },
      { data: null,
       searchable: false ,
       className: "table-view-pf-actions",
       render: function (data, type, full, meta) {
       // Inline action kebab renderer
         return '<div><button class="btn btn-default dropdown-toggle" type="button" data-toggle="dropdown"                aria-haspopup="true"                aria-expanded="true"
```

```
onclick=delete_machines_schedule('+user_id+','+sim_id+','+data['id']+');>' +
          'Delete</button></div>';
        }
       }
      ]
    });
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: add_machines_schedule

Description: Add a new machines schedule to an existing machines schedule list

```
function add_machines_schedule(user_id, sim_id, mach_id, date_from, date_to){
        date_from=moment(date_from, "DD/MM/YYYY HH:mm:ss").format('YYYY-MM-
DDTHH:mm:ss');
 date_to=moment(date_to, "DD/MM/YYYY HH:mm:ss").format('YYYY-MM-DDTHH:mm:ss');
 if (date_from<=date_to){
          var obj = {"sim_id":sim_id, "mach_id":mach_id, "date_from":date_from,
"date_to":date_to};
  d3.request("./sim_mode/add_machines_schedule.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {
   machinesSchedule(user_id,sim_id);
  })
  .send("POST","q="+JSON.stringify(obj));
 }
 else {
  $("#warningModal").css("z-index", "10000").modal();
      $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon
```

```
pficon-warning-triangle-o"></span><strong>Not added due to invalid values.</strong></div>');
   $("#warning_ok").off('click').on('click', function() {
    $("#warningModal").modal("hide");
   })
 }
}
```

Function: delete_machines_schedule

Description: Delete a machines schedule from an existing machines schedule list

```
function delete_machines_schedule(user_id,sim_id,sim_prod_order_id){
 var obj = {"sim_prod_order_id":sim_prod_order_id};
 d3.request("./sim_mode/delete_machines_schedule.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
  machinesSchedule(user_id,sim_id);
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: newSim

Description: Create a new simulator

```
function newSim(user_id) {
 var obj = {"user_id":user_id};
 d3.request("./sim_mode/get_sims_name.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
```

```
// read Designes from db to check if name exists...
  jsonResponse=JSON.parse(phpResponseObj.response);
  jsonSims=jsonResponse["names"];
  jsonDesignes=jsonResponse["des_ver"];

  //call new modal
  $("#simNewModal").modal();
      $('#sim-new-modal-body').html('<span class="col-sm-12" style="font-style:italic;font-size:8pt">Write the Name for the New Simulator and select the Version of the Design to use.</span><br>');
       $('#sim-new-modal-body2').html('<form class="form-horizontal"><div class="form-group"><label class="col-sm-3 control-label" for="textInput-sim-name">Name</label><div class="col-sm-8"><input type="text" id="textInput-sim-name" class="form-control"></div></div></form>');
  $('#sim-new-modal-msg').html('');
  $('#sim-new-modal-body3').treeview({
  collapseIcon: "fa fa-angle-down",
  data: jsonDesignes,
  expandIcon: "fa fa-angle-right",
  nodeIcon: "fa fa-folder",
  onNodeSelected: function(event, data) {
   var found,i;
   found=false;
   i=0;
   while ((found==false)&&(i<jsonSims.length)){
    if (jsonSims[i].name==$("#textInput-sim-name").val()){
     found=true;
    }
    i++;
   }
            if  ((found==false)&&($("#textInput-sim-name").val()!='')&&(($("#textInput-sim-
```

```
name").val().match(/^\s+$/) === null))) {
      $("#simNewModal").modal("hide");
      sse_source.close();

      //hide sim, buttons and info
      d3.select("#SVG_ID").remove();
      $('#selected_sim_name').text('');
      simId='';
      $('#sim_info').hide();
      $('#sim_buttons').hide();
      sim_v1_menu_hide();
      sim_v2_menu_hide();

      //send sql to insert new sim
                        var obj2 = {"user_id":user_id, "sim_name":($("#textInput-sim-
name").val()).trim(),"ver_id":data.ver_id,"time_resolution":data.time_resolution};
      d3.request("./sim_mode/insert_new_sim.php")
      .mimeType("application/json")
      .header("Content-Type","application/x-www-form-urlencoded")
      .response(function(phpResponseObj2) {
       if ($.isNumeric(phpResponseObj2.response)) {
        //show buttons and info
            $('#selected_sim_name').html("<abbr style='border: none;text-decoration: none;'
title='based on version "+data.number+"."+data.description+" of design "+data.name+"'>"+
($("#textInput-sim-name").val()).trim()+"</abbr>");
        simId=phpResponseObj2.response;
        $('#sim_info').show();
        $('#sim_buttons').show();
        sim_v2_menu_show();
        sse_source.close();
        version_rd(data.ver_id,true);
       }
```

```
      })
      .send("POST","q="+JSON.stringify(obj2));
    }
    else{
     if (found==true) $('#sim-new-modal-msg').html('<div class="form-group has-error"><div class="col-sm-10"><span class="help-block">Name in use. Please select another name.</span></div></div>');
      else $('#sim-new-modal-msg').html('<div class="form-group has-error"><div class="col-sm-10"><span class="help-block">Please select a name.</span></div></div>');
    }
   },
   levels:1,
   showBorder: false
   });
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: openSim

Description: Open an existing Simulator

```
function openSim(user_id) {
 var obj = {"user_id":user_id};
 d3.request("./sim_mode/get_sims.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {

//   read Designes from db to select for opening ...
   jsonSims=JSON.parse(phpResponseObj.response);
```

```
//call open modal
$('#sim_open_menu').removeClass("active");
$("#simOpenModal").modal();
$('#sim-open-modal-body').treeview({
collapseIcon: "fa fa-angle-down",
data: jsonSims,
expandIcon: "fa fa-angle-right",
nodeIcon: "fa fa-folder",
onNodeSelected: function(event, data) {
 $("#simOpenModal").modal("hide");

 //hide design, buttons and info
 d3.select("#SVG_ID").remove();
 $('#selected_sim_name').text('');
 simId='';
 $('#sim_info').hide();
 $('#sim_buttons').hide();
 sim_v1_menu_hide();
 sim_v2_menu_hide();

 // show menu based on permissions
      $('#selected_sim_name').html("<abbr style='border: none;text-decoration: none;'
title='based on version "+data.number+"."+data.description+" of design
"+data.des_name+"'>"+data.name+" ("+data.des_name+",
v"+data.number+"."+data.description+")"+"</abbr>");

 simId=data.id;
 $('#sim_info').show();
 if (data.permission == 'test' ) {$('#sim_run_menu').show();}
                if (data.permission == 'copy' ) {$('#sim_copy_menu').show();$
('#sim_run_menu').show();}
                if (data.permission == 'full' ) {$('#sim_buttons').show();$
```

```
('#sim_load_version_menu').show();$('#sim_save_version_menu').show();$
('#sim_copy_menu').show();$('#sim_run_menu').show();}
                        if (data.permission == 'owner' ) {$('#sim_buttons').show();$
('#sim_load_version_menu').show();$('#sim_save_version_menu').show();$
('#sim_copy_menu').show();$('#sim_run_menu').show();}
    $('#sim_close_menu').show();

    //code to read and display design ...
    sse_source.close();
    version_rd(data.ver_id,true);

  },
  levels:1,
  showBorder: false
  });
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: purchaseOrders

Description: Create a new purchase orders list

```
function purchaseOrders(user_id,sim_id) {

 var obj = {"sim_id":sim_id, "purchase_sale":"purchase"};
 d3.request("./sim_mode/ps_orders.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {

  jsonOrders=JSON.parse(phpResponseObj.response);
```

```
    jsonOrdersData=jsonOrders['orders'];
    jsonOrdersProducts=jsonOrders['products'];
   jsonOrdersSim=jsonOrders['sim'];

   var options="";
   for (x in jsonOrdersProducts) {
                                                                    options+='<option
value="'+jsonOrdersProducts[x].prod_id+'">'+jsonOrdersProducts[x].name+'</option>';
   }

   $(function () {
        $('#datetimepicker_puo1').datetimepicker({date:jsonOrdersSim[0].timestart, format:
'DD-MM-YYYY HH:mm:ss'});
   });
   var html_table='\
    <h3>Add Purchase Order</h3>\
        <table style="width:100%" class="table table-striped table-bordered table-hover"
id="purchase_table2">\
    <thead><tr>\
     <th>Product</th>\
     <th>Date</th>\
     <th>Price</th>\
     <th>Quantity</th>\
     <th>Delivery</th>\
     <th>Actions</th>\
    </tr></thead><tr>\
    <td style="padding-left:0px;padding-right:0px;width:80px;">\
    <div style="padding-left:0px;">\
     <select id="textPurchase-product" class="form-control " >'+options+'\
     </select>\
    </div>\
    </td>\
```

```
<td style="padding-left:0px;padding-right:0px;">\
<div class="input-group date" id="datetimepicker_puo1">\
 <div class="input-group-addon">\
  <span class="glyphicon glyphicon-calendar"></span>\
 </div>\
 <input type="text" class="form-control" id="textPurchase-date" value="'+''+'">\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;width:80px;">\
<div style="padding-left:0px;">\
 <input id="textPurchase-price" class="form-control " ></input>\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;width:80px;">\
<div style="padding-left:0px;">\
 <input id="textPurchase-quantity" class="form-control " ></input>\
</div>\
</td>\
<td style="padding-left:0px; padding-right:0px;width:80px;">\
    <input id="textPurchase-delivery" class="form-control " style="margin:0px;padding-left:0px;"></input>\
</td>\
<td style="padding-left:0px;padding-right:0px;width:75px;">\
        <div><button class="btn btn-default dropdown-toggle" type="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="true" style="padding-left:0px;padding-right:0px;width:75px"\
    onclick=add_purchase_order('+user_id+','+sim_id+',$("#textPurchase-product").val(),$("#textPurchase-date").val(),$("#textPurchase-price").val(),$("#textPurchase-quantity").val(),$("#textPurchase-delivery").val(),"purchase");>Add</button></div>\
</td>\
</tr>\
</table>\
```

```
<h3>Purchase Orders</h3><table style="width:100%" class="table table-striped table-bordered table-hover" id="purchase_table1">\
<thead><tr>\
<th>Product   </th>\
<th>Date    </th>\
<th>Price   </th>\
<th>Quantity   </th>\
<th>Delivery   </th>\
<th>Actions   </th>\
</tr></thead></table>';

//call sim purchase order modal
$("#purchaseOrdersModal").css("z-index", "8000").modal();
$('#purchase-orders-modal-body').html(html_table);
$('#purchase_table1').DataTable({
 data: jsonOrdersData,
 columns: [
  { data: "name" },
  { data: "date" },
  { data: "price" },
  { data: "quantity" },
  { data: "delivery" },
  { data: null,
   searchable: false ,
   className: "table-view-pf-actions",
   render: function (data, type, full, meta) {
   // Inline action kebab renderer
     return '<div><button class="btn btn-default dropdown-toggle" type="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="true" onclick=delete_purchase_order('+user_id+','+sim_id+','+data['so_id']+');>' +
      'Delete</button></div>';
   }
```

```
      }
     ],
     "columnDefs": [
      { className: "dt-body-right", "targets": [ 3,4,5 ] }
     ]
    });
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: add_purchase_order

Description: Add a purchase order to an existing purchase orders list

```
function add_purchase_order(user_id, sim_id, prod_id, date, price, quantity, delivery, purchase_sales){
 date=moment(date, "DD/MM/YYYY HH:mm:ss").format('YYYY-MM-DDTHH:mm:ss');
 if ((price>=0)&&(quantity>0)&&delivery>=0){
          var obj = {"sim_id":sim_id, "prod_id":prod_id, "date":date, "price":price, "quantity":quantity, "delivery":delivery, "purchase_sales":purchase_sales};
  d3.request("./sim_mode/add_ps_order.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {
   purchaseOrders(user_id,sim_id);
  })
  .send("POST","q="+JSON.stringify(obj));
 }
 else {
  $("#warningModal").css("z-index", "10000").modal();
      $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon pficon-warning-triangle-o"></span><strong>Not added due to invalid
```

```
values.</strong></div>');
   $("#warning_ok").off('click').on('click', function() {
    $("#warningModal").modal("hide");
   })
 }
}
```

Function: delete_purchase_order

Description: Delete a purchase order from an existing purchase orders list

```
function delete_purchase_order(user_id,sim_id,sim_order_id){
 var obj = {"sim_order_id":sim_order_id};
 d3.request("./sim_mode/delete_ps_order.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
  purchaseOrders(user_id,sim_id);
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: runSim

Description: Simulation run

```
function runSim(user_id,sim_id) {

 var obj = {"sim_id":sim_id};
 d3.request("./sim_mode/get_sim_time.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
```

```
jsonSimsTimes=JSON.parse(phpResponseObj.response);

                    var vts=moment(jsonSimsTimes['timestart'], "YYYY-MM-DDTHH:mm:ss").format('DD/MM/YYYY HH:mm:ss');
                    var vte=moment(jsonSimsTimes['timeend'], "YYYY-MM-DDTHH:mm:ss").format('DD/MM/YYYY HH:mm:ss');
  var html_inbody = '\
   <div class="card-pf">\
    <div class="card-pf-heading">\
    <h2 class="card-pf-title">\
     Top Utilized Clusters\
    </h2>\
    </div>\
    <div class="card-pf-body">\
    <div class="progress-description">\
     RHOS6-Controller\
    </div>\
    <div class="progress progress-label-top-right">\
     <div class="progress-bar progress-bar-danger" role="progressbar" aria-valuenow="95" aria-valuemin="0" aria-valuemax="100" style="width: 95%;" data-toggle="tooltip" title="95% Used">\
     <span><strong>190.0 of 200.0 GB</strong> Used</span>\
     </div>\
          <div class="progress-bar progress-bar-remaining" role="progressbar" aria-valuenow="5" aria-valuemin="0" aria-valuemax="100" style="width: 5%;" data-toggle="tooltip" title="5% Available">\
     <span class="sr-only">5% Available</span>\
     </div>\
    </div>\
    </div>\
   </div>';
```

```
    //call sim time modal
    $("#simRunModal").css("z-index", "8000").modal();
    $('#sim-run-modal-body').html(html_inbody);
    $('#sim-run-modal-msg').html('');

   //Load machines' data to machSt
   machSt = {};
   var jsonMachines = jsonDesign["machines"];
   jsonMachines.forEach((dMach,iMach) => {
    machSt[dMach.id]={};
    machSt[dMach.id].inputTime=dMach.input_time;
    machSt[dMach.id].t2ni=0;
    machSt[dMach.id].ins={};
    dMach.ins.forEach((dMIn,iIn) => {
     machSt[dMach.id].ins[dMIn.row_num]={};
     machSt[dMach.id].ins[dMIn.row_num].storageId=dMIn.storage_id;
     machSt[dMach.id].ins[dMIn.row_num].a=dMIn.a;
    });
    machSt[dMach.id].outs={};
    dMach.outs.forEach((dMOut,iOut) => {
     machSt[dMach.id].outs[dMOut.row_num]={};
     machSt[dMach.id].outs[dMOut.row_num].id=dMOut.id;
     machSt[dMach.id].outs[dMOut.row_num].storageId=dMOut.storage_id;
     machSt[dMach.id].outs[dMOut.row_num].a=dMOut.a;
     machSt[dMach.id].outs[dMOut.row_num].prodTime=dMOut.prod_time;
      machSt[dMach.id].outs[dMOut.row_num].color=(dMOut.color==""?"#bedee1":dMOut.
color);
     machSt[dMach.id].outs[dMOut.row_num].stateN=dMOut.out_state;
              machSt[dMach.id].t2ni=Math.max(machSt[dMach.id].t2ni,(dMach.input_time-
(dMOut.prod_time-Math.floor(Math.log2(dMOut.out_state)))));
```

```
      machSt[dMach.id].outs[dMOut.row_num].state={};
      machSt[dMach.id].outs[dMOut.row_num].state[0]={};
      machSt[dMach.id].outs[dMOut.row_num].state[0].bit="";
      for (var i=1;i<=dMOut.prod_time;i++) {
       machSt[dMach.id].outs[dMOut.row_num].state[i]={};
        machSt[dMach.id].outs[dMOut.row_num].state[i].bit=stateBit(dMOut.out_state,dMOut.prod_time,i);
      };
     });
   });

   //Load storages' data to storA
   storA = {};
   var jsonStorages = jsonDesign["storages"];
   jsonStorages.forEach((dStor,iStor) => {
    storA[dStor.id] = {};
    storA[dStor.id].a = dStor.a;
    storA[dStor.id].aMax = dStor.aMax;
    storA[dStor.id].bars = {};
    for (var p=0; p<20; p++) {
     storA[dStor.id].bars[+p+1]={};
     storA[dStor.id].bars[+p+1].bit=(dStor.a>=(p+1)*(dStor.aMax/20)?"1":"0");
    }
   });
console.log(machSt);
console.log(storA);
   //Write initial run data to db : sim_machine_log, sim_storage_log
   timeslot = 0;
            var obj = {"sim_id":simId, "timeslot":timeslot, "machine_state":machSt, "storage_a":storA};
   d3.request("./sim_mode/sim_db_wr.php")
   .mimeType("application/json")
```

```
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {})
  .send("POST","q="+JSON.stringify(obj));

  //Run one timestep
  simOperate();
  timeslot = +timeslot+1;

  //Write current run data to db : sim_machine_log, sim_storage_log
        var obj = {"sim_id":simId, "timeslot":timeslot, "machine_state":machSt,
"storage_a":storA};
  d3.request("./sim_mode/sim_db_wr.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {})
  .send("POST","q="+JSON.stringify(obj));

 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: salesOrders

Description: Create a new sales orders list

```
function salesOrders(user_id,sim_id) {

 var obj = {"sim_id":sim_id, "purchase_sale":"sale"};
 d3.request("./sim_mode/ps_orders.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
```

```
  jsonOrders=JSON.parse(phpResponseObj.response);
   jsonOrdersData=jsonOrders['orders'];
   jsonOrdersProducts=jsonOrders['products'];
  jsonOrdersSim=jsonOrders['sim'];

  var options="";
  for (x in jsonOrdersProducts) {
                                                    options+='<option
value="'+jsonOrdersProducts[x].prod_id+'">'+jsonOrdersProducts[x].name+'</option>';
  }

  $(function () {
        $('#datetimepicker_sao1').datetimepicker({date:jsonOrdersSim[0].timestart, format:
'DD-MM-YYYY HH:mm:ss'});
  });
  var html_table='\
   <h3>Add Sale Order</h3>\
       <table style="width:100%" class="table table-striped table-bordered table-hover"
id="sales_table2">\
    <thead><tr>\
     <th>Product</th>\
     <th>Date</th>\
     <th>Price</th>\
     <th>Quantity</th>\
     <th>Delivery</th>\
     <th>Actions</th>\
    </tr></thead><tr>\
    <td style="padding-left:0px;padding-right:0px;width:80px;">\
    <div style="padding-left:0px;">\
     <select id="textSales-product" class="form-control " >'+options+'\
     </select>\
```

```
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;">\
<div class="input-group date" id="datetimepicker_sao1">\
 <div class="input-group-addon">\
  <span class="glyphicon glyphicon-calendar"></span>\
 </div>\
 <input type="text" class="form-control" id="textSales-date" value="'+''+'">\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;width:80px;">\
<div style="padding-left:0px;">\
 <input id="textSales-price" class="form-control " ></input>\
</div>\
</td>\
<td style="padding-left:0px;padding-right:0px;width:80px;">\
<div style="padding-left:0px;">\
 <input id="textSales-quantity" class="form-control " ></input>\
</div>\
</td>\
<td style="padding-left:0px; padding-right:0px;width:80px;">\
        <input id="textSales-delivery" class="form-control " style="margin:0px;padding-left:0px;"></input>\
</td>\
<td style="padding-left:0px;padding-right:0px;width:75px;">\
         <div><button class="btn btn-default dropdown-toggle" type="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="true" style="padding-left:0px;padding-right:0px;width:75px"\
              onclick=add_sales_order('+user_id+','+sim_id+',$("#textSales-product").val(),$("#textSales-date").val(),$("#textSales-price").val(),$("#textSales-quantity").val(),$("#textSales-delivery").val(),"sale");>Add</button></div>\
</td>\
```

```
    </tr>\
    </table>\
        <h3>Sales Orders</h3><table style="width:100%" class="table table-striped table-
bordered table-hover" id="sales_table1">\
    <thead><tr>\
     <th>Product   </th>\
     <th>Date   </th>\
     <th>Price   </th>\
     <th>Quantity   </th>\
     <th>Delivery   </th>\
     <th>Actions   </th>\
    </tr></thead></table>';

    //call sim supplies order modal
    $("#salesOrdersModal").css("z-index", "8000").modal();
    $('#sales-orders-modal-body').html(html_table);
    $('#sales_table1').DataTable({
     data: jsonOrdersData,
     columns: [
      { data: "name" },
      { data: "date" },
      { data: "price" },
      { data: "quantity" },
      { data: "delivery" },
      { data: null,
       searchable: false ,
       className: "table-view-pf-actions",
       render: function (data, type, full, meta) {
       // Inline action kebab renderer
         return '<div><button class="btn btn-default dropdown-toggle" type="button" data-
toggle="dropdown"          aria-haspopup="true"          aria-expanded="true"
onclick=delete_sales_order('+user_id+','+sim_id+','+data['so_id']+');>' +
```

```
          'Delete</button></div>';
        }
       }
      ],
      "columnDefs": [
       { className: "dt-body-right", "targets": [ 3,4,5 ] }
      ]
     });
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: add_sales_order

Description: Add a sale order to an existing sales orders list

```
function add_sales_order(user_id, sim_id, prod_id, date, price, quantity, delivery, purchase_sales){
 date=moment(date, "DD/MM/YYYY HH:mm:ss").format('YYYY-MM-DDTHH:mm:ss');
 if ((price>=0)&&(quantity>0)&&delivery>=0){
          var obj = {"sim_id":sim_id, "prod_id":prod_id, "date":date, "price":price, "quantity":quantity, "delivery":delivery, "purchase_sales":purchase_sales};
   d3.request("./sim_mode/add_ps_order.php")
   .mimeType("application/json")
   .header("Content-Type","application/x-www-form-urlencoded")
   .response(function(phpResponseObj) {
    salesOrders(user_id,sim_id);
   })
   .send("POST","q="+JSON.stringify(obj));
 }
 else {
   $("#warningModal").css("z-index", "10000").modal();
```

```
        $('#warning-modal-body').html('<div class="alert alert-warning"><span class="pficon pficon-warning-triangle-o"></span><strong>Not added due to invalid values.</strong></div>');
    $("#warning_ok").off('click').on('click', function() {
      $("#warningModal").modal("hide");
    })
  }
}
```

Function: delete_sales_order

Description: Delete a sale order from an existing sales orders list

```
function delete_sales_order(user_id,sim_id,sim_order_id){
  var obj = {"sim_order_id":sim_order_id};
  d3.request("./sim_mode/delete_ps_order.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj) {
    salesOrders(user_id,sim_id);
  })
  .send("POST","q="+JSON.stringify(obj));
}
```

Function: setSimTime

Description: Set simulation time (start and end)

```
function setSimTime(user_id,sim_id) {

  var obj = {"sim_id":sim_id};
  d3.request("./sim_mode/get_sim_time.php")
  .mimeType("application/json")
```

```
.header("Content-Type","application/x-www-form-urlencoded")
.response(function(phpResponseObj) {

  jsonSimsTimes=JSON.parse(phpResponseObj.response);
  var time_res=TIME_STEPS[jsonSimsTimes['timestep']];

                              var vts=moment(jsonSimsTimes['timestart'], "YYYY-MM-
DDTHH:mm:ss").format('DD/MM/YYYY HH:mm:ss');
                              var vte=moment(jsonSimsTimes['timeend'], "YYYY-MM-
DDTHH:mm:ss").format('DD/MM/YYYY HH:mm:ss');
  var html_inbody = '\
    <div class="container-fluid">\
     <form class="form-horizontal">\
     <div class="form-group">\
      <div class="row">\
       <label class="col-sm-4 control-label" for="textUpd-timestart">Time Start</label>\
       <div class="col-sm-7 input-group date" id="datetimepicker1">\
        <div class="input-group-addon">\
         <span class="glyphicon glyphicon-calendar"></span>\
        </div>\
        <input type="text" class="form-control" id="textUpd-timestart" value="'+vts+'">\
       </div>\
      </div>\
      <div class="row">\
       <label class="col-sm-4 control-label" for="textUpd-timeend">Time End</label>\
       <div class="col-sm-7 input-group date" id="datetimepicker2">\
        <div class="input-group-addon">\
         <span class="glyphicon glyphicon-calendar"></span>\
        </div>\
        <input type="text" class="form-control" id="textUpd-timeend" value="'+vte+'">\
       </div>\
      </div>\
```

```
        <div class="row">\
         <label class="col-sm-4 control-label" for="textUpd-timestep">Time Step</label>\
         <div class="col-sm-4" style="padding-left:0px;">\
                                  <span   id="textUpd-timestep"   style="font-size:10px;vertical-
align:bottom;">'+time_res+'\
          </span>\
         </div>\
        </div>\
       </div>\
       </form>\
      </div>';
     $(function () {
         $('#datetimepicker1').datetimepicker({format: 'DD-MM-YYYY HH:mm:ss'});
         $('#datetimepicker2').datetimepicker({format: 'DD-MM-YYYY HH:mm:ss'});
     });

     //call sim time modal
     $("#simTimeModal").css("z-index", "8000").modal();
     $('#sim-time-modal-body').html(html_inbody);
     $('#sim-time-modal-msg').html('');
     $("#sim-time-btn_ok").off('click').on('click', function (event, data) {

       var timestart = $("#textUpd-timestart").val();
       var timeend = $("#textUpd-timeend").val();
       var timestep = jsonSimsTimes['timestep'];
                timestart_t=moment(timestart,  'DD/MM/YYYY  HH:mm:ss').format("YYYY-MM-
DDTHH:mm:ss");
                 timeend_t=moment(timeend,  'DD/MM/YYYY  HH:mm:ss').format("YYYY-MM-
DDTHH:mm:ss");

       var date_s = new Date(timestart_t);
       var ds = date_s.getTime() / 1000; //from milliseconds to seconds
```

```
    var date_e = new Date(timeend_t);
    var de = date_e.getTime() / 1000; //from milliseconds to seconds
    var ts = parseInt((de-ds)/timestep);
    if ((ds>0)&&((de>0))&&(ds<de)&&(ts<TIME_STEPS_LIMIT)) {
      $("#simTimeModal").modal("hide");
      //send sql to update times
            var obj = {"sim_id":sim_id, "timestart":timestart_t, "timeend":timeend_t,
"timestep":timestep};
      d3.request("./sim_mode/update_sim_time.php")
      .mimeType("application/json")
      .header("Content-Type","application/x-www-form-urlencoded")
      .response(function(phpResponseObj2) {
       //sse_source.close();
       //design_rd(design_id);
      })
      .send("POST","q="+JSON.stringify(obj));
    } else {
        if (ds>de) $('#sim-time-modal-msg').html('<div class="form-group has-error"><div
class="col-sm-10"><span class="help-block">Time End should be after Time
Start.</span></div></div>');
        else if (ts>TIME_STEPS_LIMIT) $('#sim-time-modal-msg').html('<div class="form-
group has-error"><div class="col-sm-12"><span class="help-block">Too many Time Steps!
('+ts+'). The limit is '+TIME_STEPS_LIMIT+'.</span></div></div>');
    }
   });
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: shareSim

Description: Share a simulator

```
function shareSim(user_id) {
 var obj = {"user_id":user_id};
 d3.request("./sim_mode/get_own_sims.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
//   read Designes from db to select for opening ...
  jsonSims=JSON.parse(phpResponseObj.response);

  //call share modal
  $('#sim_share_menu').removeClass("active");
  $("#simShareModal").modal();
  $('#sim-share-modal-msg').html('');
  $('#sim-share-modal-body').treeview({
  collapseIcon: "fa fa-angle-down",
  data: jsonSims,
  expandIcon: "fa fa-angle-right",
  nodeIcon: "fa fa-folder",
  onNodeSelected: function(event, data) {
   //code to chown design ...
    var sim_id=data.id;
    var obj3 = {"user_id":user_id,"sim_id":sim_id};
    d3.request("./sim_mode/perm_sim.php")
     .mimeType("application/json")
     .header("Content-Type","application/x-www-form-urlencoded")
     .response(function(phpResponseObj3) {
      var json3=JSON.parse(phpResponseObj3.response);
      $("#chownModal").css("z-index", "8000").modal();
                $('#chown-modal-head').html('<button type="button" class="close" data-
dismiss="modal">&times;</button><h4 class="modal-title" >Change Ownership for
'+data.name+'</h4>');
```

```
        $('#chown-modal-msg').html('');
        create_table_sim(user_id, sim_id);
      }).send("POST","q="+JSON.stringify(obj3));
    },
  levels:1,
  showBorder: false
  });
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: give_permission_sim

Description: Giving permissions to a user for a simulator

```
function give_permission_sim(user_id,user_id_to_change,sim_id,permission){
 var obj = {"user_id":user_id_to_change, "sim_id":sim_id, "permission":permission};
 d3.request("./sim_mode/give_sim_permission.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
  JSON.parse(phpResponseObj.response);
  create_table_sim(user_id, sim_id);
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: create_table_sim

Description: Create permissions table for a simulator

```
function create_table_sim(user_id, sim_id) {
 var obj3 = {"user_id":user_id,"sim_id":sim_id};
```

```
 d3.request("./sim_mode/perm_sim.php")
  .mimeType("application/json")
  .header("Content-Type","application/x-www-form-urlencoded")
  .response(function(phpResponseObj3) {
   var jsonSim3=JSON.parse(phpResponseObj3.response);
   $('#chown-modal-body').html('');

       $('#chown-modal-body').html('<table style="width:100%" class="table table-striped table-bordered table-hover" id="table1">\
    <thead><tr>\
     <th>User Name   </th>\
     <th>Name   </th>\
     <th>Permission   </th>\
     <th>Actions         </th>\
    </tr></thead></table>');
    $('#table1').DataTable({
     data: jsonSim3,
     columns: [
      { data: "uname" },
      { data: "name" },
      { data: "permission" },
      { data: null,
       searchable: false ,
       className: "table-view-pf-actions",
       render: function (data, type, full, meta) {
       // Inline action kebab renderer
       return '<div class="dropdown dropdown-kebab-pf">' +
                    '<button class="btn btn-default dropdown-toggle" type="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="true">' +
         '<span class="fa fa-ellipsis-v"></span></button>' +
                                  '<ul class="dropdown-menu dropdown-menu-right" aria-labelledby="dropdownKebabRight">' +
```

```
                    '<li><a  href="#"  onclick=give_permission_sim('+user_id+','+data['id']
+','+sim_id+',"full");>give Full permission</a></li>' +
                    '<li><a  href="#"  onclick=give_permission_sim('+user_id+','+data['id']
+','+sim_id+',"copy");>give Copy permission</a></li>' +
                    '<li><a  href="#"  onclick=give_permission_sim('+user_id+','+data['id']
+','+sim_id+',"order");>give Order permission</a></li>' +
                    '<li><a  href="#"  onclick=give_permission_sim('+user_id+','+data['id']
+','+sim_id+',"test");>give Test permission</a></li>' +
                    '<li><a  href="#"  onclick=give_permission_sim('+user_id+','+data['id']
+','+sim_id+',"view");>give View permission</a></li>' +
                    '<li><a  href="#"  onclick=give_permission_sim('+user_id+','+data['id']
+','+sim_id+',"");>delete permission</a></li></ul></div>';
        }
       }
      ]
    });
 }).send("POST","q="+JSON.stringify(obj3));
}
```

Function: simOperate

Description: Simulation Operation

```
function simOperate() {
 var n = 0;
 var ok2in = '0';
 for (var mId in machSt) {
  //output operation
  for (var mOut in machSt[mId].outs) {
   //n - last state bit
   n = machSt[mId].outs[mOut].prodTime;
   if (machSt[mId].outs[mOut].state[n].bit=="1") {
```

```
      if (machSt[mId].outs[mOut].storageId!="0") {
        if ((+storA[machSt[mId].outs[mOut].storageId].a + (+machSt[mId].outs[mOut].a)) <=
(+storA[machSt[mId].outs[mOut].storageId].aMax)) {
                                                    storA[machSt[mId].outs[mOut].storageId].a     =
(+storA[machSt[mId].outs[mOut].storageId].a)+(+machSt[mId].outs[mOut].a);
       } else console.log('Warning storage '+machSt[mId].outs[mOut].storageId+' overflow');
      } else  console.log('Warning machine '+mId+' output '+mOut+' not connected');
    };
   };
   //input operation
   ok2in = '0';
   if (machSt[mId].t2ni == 0) {
    ok2in = '1';
    for (var mIn in machSt[mId].ins) {
     if (machSt[mId].ins[mIn].storageId != "0") {
      if (+storA[machSt[mId].ins[mIn].storageId].a < +machSt[mId].ins[mIn].a) ok2in = '0';
     } else ok2in = '0';
    }
    if (ok2in == '1')
     for (var mIn in machSt[mId].ins) {
      if (machSt[mId].ins[mIn].storageId != "0") {
                                                      storA[machSt[mId].ins[mIn].storageId].a     =
(+storA[machSt[mId].ins[mIn].storageId].a) - (+machSt[mId].ins[mIn].a);
       } else console.log('Warning machine '+mId+' input '+mIn+' not connected');
      }
   }
   //change states operation
   for (var mOut in machSt[mId].outs) {
    //n - last state bit
    n = machSt[mId].outs[mOut].prodTime;
    machSt[mId].outs[mOut].state[0].bit = ok2in;
    for (var i=(+n); i>0; i--)
```

```
      machSt[mId].outs[mOut].state[i].bit = machSt[mId].outs[mOut].state[i-1].bit;
    if (ok2in == '1')
      machSt[mId].t2ni = +machSt[mId].inputTime-1;
    else if (machSt[mId].t2ni != 0)
      machSt[mId].t2ni--;
                                                machSt[mId].outs[mOut].stateN =
binArr2Num(JSON.stringify(machSt[mId].outs[mOut].state));
   }
  };
  for (var sId in storA) {
   for (var p=0; p<20; p++) {
    storA[sId].bars[+p+1].bit=(storA[sId].a>=(p+1)*(storA[sId].aMax/20)?"1":"0");
   }
  }
}
```

Function: viewSim

Description: View a simulator

```
function viewSim(user_id) {
 var obj = {"user_id":user_id};
 d3.request("./sim_mode/get_sims_and_versions.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {

//   read Sims from db to select for opening ...
   jsonSims=JSON.parse(phpResponseObj.response);

   //call sim view modal
   $('#sim_view_menu').removeClass("active");
```

```
$("#simViewModal").modal();
$('#sim-view-modal-body').treeview({
collapseIcon: "fa fa-angle-down",
data: jsonSims,
expandIcon: "fa fa-angle-right",
nodeIcon: "fa fa-folder",
onNodeSelected: function(event, data) {
 $("#simViewModal").modal("hide");

 //hide sim, buttons and info
    d3.select("#SVG_ID").remove();
    $('#selected_sim_name').text('');
    simId='';
    $('#sim_info').hide();
    $('#sim_buttons').hide();
    sim_v1_menu_hide();
    sim_v2_menu_hide();

    // show menu
        $('#selected_sim_name').html("<abbr style='border: none;text-decoration: none;'
title='based on version "+data.number+"."+data.description+" of design
"+data.des_name+"'>"+data.name+"</abbr>");
    simId=data.sim_id;
    $('#sim_info').show();
    $('#sim_close_menu').show();

    //code to read and display sim ...
    //acquire objects in case of sim selected
    if (data.ver_id=='sim_selected'){
     sse_source.close();
     machStorEdit="no";
     version_rd(data.des_ver_id,true);
```

```
    }
    //code to read and display version ...
    //acquire objects in case of version selected
    else {
      sse_source.close();
      machStorEdit="no";
      version_rd(data.des_ver_id,true);
    }
   },
   levels:1,
   showBorder: false
   });
 })
 .send("POST","q="+JSON.stringify(obj));
}
```

Function: closeResults

Description: Close Results

```
function closeResults(user_id) {
 //hide design, buttons and info
 d3.select("#SVG_ID").remove();
 $('#selected_results_name').text('');
 resultId='';
 $('#results_info').hide();
 results_v1_menu_hide();
 results_v2_menu_hide();
 d3.selectAll("g").remove();
 $('#ResultsArea').html('');

 // show menu
```

```
 results_v1_menu_show();
}
```

Function: productGraph

Description: Create charts for products

```
function productGraph(user_id) {
console.log(user_id);
console.log(resultsId);
 var obj = {"user_id":user_id,"sim_run_id":resultsId};
 d3.request("./results_mode/get_products_name.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {

// read Products from db to check if name exists ...
 jsonProducts=JSON.parse(phpResponseObj.response);

console.log(jsonProducts);
   var html_table='\
     <h3>Products</h3><table style="width:100%" class="table table-striped table-bordered table-hover" id="product_table">\
    <thead><tr>\
     <th>Name   </th>\
     <th>Unit   </th>\
                                                <th style="padding-left:18px;padding-right:19px;">Actions      </th>\
    </tr></thead></table>';


  //call modal
```

```
  $("#selectProductModal").modal();
  $('#select-product-modal-body').html(html_table);

  $('#product_table').DataTable({
    data: jsonProducts,
    columns: [
     { data: "name" },
     { data: "unit" },
     { data: null,
      searchable: false ,
      className: "table-view-pf-actions",
      render: function (data, type, full, meta) {
      // Inline action kebab renderer
      var html_return;
       html_return='<div><button type="button" onclick=showGraph('+user_id+','+data['id']
+',"'+data['name']+'","'+data['unit']+'");>' +
        'Graph</button>';
            html_return+='<button type="button" onclick=showTable('+user_id+','+data['id']
+',"'+data['name']+'","'+data['unit']+'");>' +
        'Table</button></div>';
      return html_return;
      }
     }
    ]
  });
 })
 .send("POST","q="+JSON.stringify(obj));
}

function showGraph(user_id,id,name,unit) {
 var obj = { "run_id":resultsId};
 d3.selectAll("polyline").remove();
```

```
 d3.selectAll(".machine").remove();
 d3.selectAll(".storage").remove();
 $("#selectProductModal").modal("hide");
 d3.selectAll("g").remove();
 $('#ResultsArea').html('');

 d3.request("results_mode/get_run_data.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
  jsonRunData=JSON.parse(phpResponseObj.response);
  console.log(jsonRunData);
  showTimeGraph(jsonRunData,name,unit);
  })
 .send("POST","q="+JSON.stringify(obj));
}

function showTable(sim_run_id,id,name,unit) {
 var obj = { "run_id":sim_run_id};
 d3.selectAll("polyline").remove();
 d3.selectAll(".machine").remove();
 d3.selectAll(".storage").remove();
 $("#selectProductModal").modal("hide");
 d3.selectAll("g").remove();
 $('#ResultsArea').html('');

 d3.request("results_mode/get_run_data.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {
  jsonRunData=JSON.parse(phpResponseObj.response);
  console.log(jsonRunData);
```

```
  showTimeTable(jsonRunData,name,unit);
  })
 .send("POST","q="+JSON.stringify(obj));
}

function showTimeGraph(data,name,unit) {

 var svg = d3.select("svg").append("g").attr("height",800).attr("width","100%");
 var parseDate = d3.timeParse("%Y-%m-%d %H:%M:%S");

 data.forEach(function(d) {
  d.timer = parseDate(d.timer);
  d.amount = +d.amount;
  return d;
 })

 // modify data to create a bar for every point
 var newArr = [];
 var d0 = new Date(data[0].timer);
 var d1 = new Date(data[1].timer);
 var half_barwidth = (d1.getTime()-d0.getTime())*0.45;

 for ( var i = 0; i < data.length; i++){

  var d = new Date(data[i].timer);
  var nt1 = new Date(d.getTime()-half_barwidth);
  var nt2 = new Date(d.getTime()+half_barwidth);

  var newObj1 = {};
  newObj1.timer = nt1;
  newObj1.amount = 0;
  newArr.push(newObj1);
```

```
  var newObj2 = {};
  newObj2.timer = nt1;
  newObj2.amount = data[i].amount;
  newArr.push(newObj2);

  var newObj3 = {};
  newObj3.timer = nt2;
  newObj3.amount = data[i].amount;
  newArr.push(newObj3);

  var newObj4 = {};
  newObj4.timer = nt2;
  newObj4.amount = 0;
  newArr.push(newObj4);
}

var chartGroup = svg.append("g")
  .attr("id", "chartGroup")
  .attr("class", "chartGroup")
  .attr("transform", "translate(200,50)")
  .attr("height",250)
  .attr("width",500);

//product unit text
chartGroup.append("text")
    .attr("x","41")
    .attr("y","10")
    .attr("font-size","12px")
    .attr("fill","#003d44")
    .attr("text-anchor","middle")
    .attr("alignment-baseline","middle")
```

```
    .text(unit);

var drag1 = d3.drag()
  .on("start", function () {dragStartFunc("#chartGroup");})
  .on("drag", function () {dragFunc("#chartGroup");});

d3.select("chartGroup"),
  margin = {top: 20, right: 20, bottom: 90, left: 40},
  margin2 = {top: 190, right: 20, bottom: 30, left: 40},
  width = +chartGroup.attr("width") - margin.left - margin.right,
  width2 = +width/2,
  height = +chartGroup.attr("height") - margin.top - margin.bottom,
  height2 = +chartGroup.attr("height") - margin2.top - margin2.bottom;

var x = d3.scaleTime().range([0, width]),
  x2 = d3.scaleTime().range([0, width]),
  y = d3.scaleLinear().range([height, 0]),
  y2 = d3.scaleLinear().range([height2, 0]);

var xAxis = d3.axisBottom(x),
  xAxis2 = d3.axisBottom(x2),
  yAxis = d3.axisLeft(y);

var brush = d3.brushX()
  .extent([[0, 0], [width, height2]])
  .on("brush end", brushed);

var zoom = d3.zoom()
  .scaleExtent([1, Infinity])
  .translateExtent([[0, 0], [width, height]])
  .extent([[0, 0], [width, height]])
  .on("zoom", zoomed);
```

```
var area = d3.area()
  .curve(d3.curveStep)
  .x(function(d) { return x(d.timer); })
  .y0(height)
  .y1(function(d) { return y(d.amount); });

var area2 = d3.area()
  .curve(d3.curveStep)
  .x(function(d) { return x2(d.timer); })
  .y0(height2)
  .y1(function(d) { return y2(d.amount); });

chartGroup.append("defs").append("clipPath")
  .attr("id", "clip")
    .append("rect")
  .attr("width", width)
  .attr("height", height);

var focus = chartGroup.append("g")
  .attr("class", "focus")
  .attr("transform", "translate(" + margin.left + "," + margin.top + ")");

var context = chartGroup.append("g")
  .attr("class", "context")
  .attr("transform", "translate(" + margin2.left + "," + margin2.top + ")");

x.domain(d3.extent(newArr, function(d) { return d.timer; }));
y.domain([0, d3.max(newArr, function(d) { return d.amount; })]);
x2.domain(x.domain());
y2.domain(y.domain());
```

```
focus.append("path")
   .datum(newArr)
   .attr("class", "area")
   .attr("d", area);

focus.append("g")
   .attr("class", "axis axis--x")
   .attr("transform", "translate(0," + height + ")")
   .call(xAxis);

focus.append("g")
   .attr("class", "axis axis--y")
   .call(yAxis);

context.append("path")
   .datum(newArr)
   .attr("class", "area")
   .attr("d", area2);

context.append("g")
   .attr("class", "axis axis--x")
   .attr("transform", "translate(0," + height2 + ")")
   .call(xAxis2);

context.append("g")
   .attr("class", "brush")
   .call(brush)
   .call(brush.move, x.range());

focus.append("text")
    .attr("x",width2)
    .attr("y","-20")
```

```
    .attr("font-size","16px")
    .attr("fill","#003d44")
    .attr("text-anchor","middle")
    .attr("alignment-baseline","middle")
    .text(name);

chartGroup.append("rect")
   .attr("class", "zoom")
   .attr("width", width)
   .attr("height", height)
   .attr("transform", "translate(" + margin.left + "," + margin.top + ")")
   .call(zoom);

var radius=3;
chartGroup.append("circle")
  .attr("cx", (margin.left/2-1)/2)
    .attr("cy", (height2-1)/2)
    .attr("r", radius)
  .style("stroke", "black")
  .style("fill", "none")
  .attr("transform", "translate(" + margin2.left/2 + "," + (1+margin2.top) + ")");

chartGroup.append("polygon")
  .attr("points",((margin.left/2-1)/2)+", "+((height2-1)/2-radius-2-radius*1.41)+", "+
        ((margin.left/2-1)/2+radius)+", "+((height2-1)/2-radius-2)+", "+
        ((margin.left/2-1)/2-radius)+", "+((height2-1)/2-radius-2))
  .style("stroke", "black")
  .style("fill", "none")
  .attr("transform", "translate(" + margin2.left/2 + "," + (1+margin2.top) + ")");

chartGroup.append("polygon")
  .attr("points",((margin.left/2-1)/2)+", "+((height2-1)/2+radius+2+radius*1.41)+", "+
```

```
        ((margin.left/2-1)/2+radius)+", "+((height2-1)/2+radius+2)+", "+
        ((margin.left/2-1)/2-radius)+", "+((height2-1)/2+radius+2))
  .style("stroke", "black")
  .style("fill", "none")
  .attr("transform", "translate(" + margin2.left/2 + "," + (1+margin2.top) + ")");

 chartGroup.append("polygon")
  .attr("points",((margin.left/2-1)/2-radius-2-radius*1.41)+", "+((height2-1)/2)+", "+
        ((margin.left/2-1)/2-radius-2)+", "+((height2-1)/2+radius)+", "+
        ((margin.left/2-1)/2-radius-2)+", "+((height2-1)/2-radius))
  .style("stroke", "black")
  .style("fill", "none")
  .attr("transform", "translate(" + margin2.left/2 + "," + (1+margin2.top) + ")");

 chartGroup.append("polygon")
  .attr("points",((margin.left/2-1)/2+radius+2+radius*1.41)+", "+((height2-1)/2)+", "+
        ((margin.left/2-1)/2+radius+2)+", "+((height2-1)/2+radius)+", "+
        ((margin.left/2-1)/2+radius+2)+", "+((height2-1)/2-radius))
  .style("stroke", "black")
  .style("fill", "none")
  .attr("transform", "translate(" + margin2.left/2 + "," + (1+margin2.top) + ")");

 chartGroup.append("rect")
    .attr("class", "drag")
    .attr("width", margin.left/2-1)
    .attr("height", height2-1)
    .attr("transform", "translate(" + margin2.left/2 + "," + (1+margin2.top) + ")")
    .call(drag1);

 function brushed() {
    if (d3.event.sourceEvent && d3.event.sourceEvent.type === "zoom") return; // ignore
brush-by-zoom
```

```
  var s = d3.event.selection || x2.range();
  x.domain(s.map(x2.invert, x2));
  focus.select(".area").attr("d", area);
  focus.select(".axis--x").call(xAxis);
  svg.select(".zoom").call(zoom.transform, d3.zoomIdentity
    .scale(width / (s[1] - s[0]))
    .translate(-s[0], 0));
 }

 function zoomed() {
   if (d3.event.sourceEvent && d3.event.sourceEvent.type === "brush") return; // ignore
zoom-by-brush
  var t = d3.event.transform;
  x.domain(t.rescaleX(x2).domain());
  focus.select(".area").attr("d", area);
  focus.select(".axis--x").call(xAxis);
  context.select(".brush").call(brush.move, x.range().map(t.invertX, t));
 }
}

function dragStartFunc(id) {
 d3.select(id).raise();
 xMouseDiff = d3.event.x - d3.select(id).node().transform.baseVal[0].matrix.e;
 yMouseDiff = d3.event.y - d3.select(id).node().transform.baseVal[0].matrix.f;
}

function dragFunc(id) {
   d3.select(id).attr("transform", "translate("+(d3.event.x - xMouseDiff)+","+(d3.event.y -
yMouseDiff)+")");
}

function showTimeTable(data,name,unit) {
```

```
 data.forEach(function(d) {
  d.timer=moment(d.timer, "YYYY-MM-DD HH:mm:ss").format('DD-MM-YYYY HH:mm:ss');
  d.amount = +d.amount;
  return d;
 })
//console.log(data);
 var html_table='\
   <h3>Product: '+name+'</h3><table style="width:100%" class="table table-striped table-
bordered table-hover" id="results_table">\
  <thead><tr>\
   <th>Date & Time   </th>\
   <th>Amount   </th>\
  </tr></thead></table>';

 $('#ResultsArea').html(html_table);
 $('#results_table').DataTable({
  data: data,
  columns: [
   { data: "timer" },
   { data: "amount" }
  ]
 });
}

function showTimeTableNest(data) {
 var parseDate = d3.timeParse("%Y-%m-%d %H:%M:%S");

 data.forEach(function(d) {
//   d.timer = parseDate(d.timer);
  d.amount = +d.amount;
  return d;
```

```
})

var dataMetrics = d3.nest()
        .key(function(d) { return d.timer.split(" ")[0]; })
        .rollup(function(v) { return {
        count: v.length,
        total: d3.sum(v, function(d) { return d.amount; }),
        avg: d3.format(".1f")(d3.mean(v, function(d) { return d.amount; }))
        }; })
        .entries(data);

            var flatten = JSON.parse(JSON.stringify(dataMetrics).replace(/"value":
{/g,'').replace(/}}/g,'}'));
var sortAscending = true;
var table = d3.select('#ResultsArea').append('table').attr('class','scroll');
var titles = d3.keys(flatten[0]);
var headers = table.append('thead').append('tr')
    .selectAll('th')
    .data(titles).enter()
    .append('th')
    .text(function (d) {return '\u2195 '+d;})
    .on('click', function (d) {
      //headers.attr('class', 'header');
      if (sortAscending) {
      rows.sort(function(a, b) { return d3.ascending(b[d],a[d]); });
      sortAscending = false;
      this.className = 'aes';
      } else {
      rows.sort(function(a, b) { return d3.descending(b[d],a[d]); });
      sortAscending = true;
      this.className = 'des';
      }
```

```
    });

 var rows = table.append('tbody').selectAll('tr')
          .data(flatten).enter()
          .append('tr');
 rows.selectAll('td')
     .data(function (d) {
      return titles.map(function (k) {
        return { 'value': d[k], 'name': k};
      });
     }).enter()
     .append('td')
     .attr('data-th', function (d) {return d.name;})
     .text(function (d) {return d.value;});
}
```

Function: selectRun

Description: Select run to see the results

```
function selectRun(user_id) {
 var obj = {"user_id":user_id};
 d3.request("./sim_mode/get_sims_and_versions.php")
 .mimeType("application/json")
 .header("Content-Type","application/x-www-form-urlencoded")
 .response(function(phpResponseObj) {

//   read Sims from db to select for opening ...
   jsonSims=JSON.parse(phpResponseObj.response);

   //call results select modal
   $('#res_select_menu').removeClass("active");
```

```
$("#resSelectModal").modal();
$('#res-select-modal-body').treeview({
collapseIcon: "fa fa-angle-down",
data: jsonSims,
expandIcon: "fa fa-angle-right",
nodeIcon: "fa fa-folder",
onNodeSelected: function(event, data) {
 $("#resSelectModal").modal("hide");

 //hide sim, buttons and info
    d3.select("#SVG_ID").remove();
    $('#selected_results_name').text('');
    resultsId='';
    $('#results_info').hide();
    results_v1_menu_hide();
    results_v2_menu_hide();

    // show menu
        $('#selected_results_name').html("<abbr style='border: none;text-decoration: none;'
title='based on version "+data.number+"."+data.description+" of design
"+data.des_name+"'>"+data.name+"</abbr>");
    resultsId=data.sim_id;
//     console.log(resultsId);
    $('#results_info').show();
    results_v2_menu_show();

    //code to read and display sim ...
    //acquire objects in case of sim selected
    if (data.ver_id=='sim_selected'){
     sse_source.close();
     machStorEdit="no";
     version_rd(data.des_ver_id,true);
```

```
    }
    //code to read and display version ...
    //acquire objects in case of version selected
    else {
      sse_source.close();
      machStorEdit="no";
      version_rd(data.des_ver_id,true);
    }
   },
   levels:1,
   showBorder: false
   });
  })
  .send("POST","q="+JSON.stringify(obj));
}
```

Bibliography

1: W3C, HTML 5, , https://www.w3.org/html/

2: PatternFly , "PatternFly home page", 2013

3: The jQuery Foundation, jQuery Site, , https://jquery.com/

4: David Dailey, Jon Frost, Domenico Strazzullo, "Building Web Applications with SVG", 2012

5: Ændrew Rininsland, Swizec Teller, "D3.js 4.x Data Visualization", 2017

6: W3C, W3C HTML5 Server-Sent Events, , https://html.spec.whatwg.org/multipage/server-sent-events.html

7: W3C, SVG Working Group, SCALABLE VECTOR GRAPHICS (SVG), , https://www.w3.org/Graphics/SVG/

8: Ændrew Rininsland, Michael Heydt, Pablo Navarro Castillo, "D3.js: Cutting-edge Data Visualization", 2017

9: Amr Tarek, Stamboliyska Rayna, "Practical D3.js", 2016

10: Mike Bostock, D3.js WebSite, , https://d3js.org/

11: Jonathan Chaffer, Karl Swedberg, "Learning jQuery", 2013

12: Sai Srinivas Sriparasa, "JavaScript and JSON Essentials", 2013

13: ECMA, Introducing JSON, , https://www.json.org/

14: Ilya Grigorik, "High Performance Browser Networking", 2013

15: Microsoft, Microsoft Edge Platform status, , https://developer.microsoft.com/en-us/microsoft-edge/platform/status/serversenteventseventsource/