BRAINE

BRAINE - Big data Processing and Artificial Intelligence at the Network Edge

Project Title:	**BRAINE - Big data Processing and Artificial Intelligence at the Network Edge**
Contract No:	876967 – BRAINE
Instrument:	ECSEL Research and Innovation Action
Call:	H2020-ECSEL-2019-2-RIA
Start of project:	1 May 2020
Duration:	43 months

Deliverable No: D5.9

BRAINE industry 4.0 use case

Due date of deliverable:	30 November 2023
Actual submission date:	25 November 2023
Version:	1.0

Project funded by the European Community under the
H2020 Programme for Research and Innovation.

Project ref. number	876967
Project title	BRAINE - Big data Processing and Artificial Intelligence at the Network Edge

Deliverable title	BRAINE industry 4.0 use case
Deliverable number	D5.9
Deliverable version	Version 1.0
Contractual date of delivery	30 November 2023
Actual date of delivery	25 November 2023
Deliverable filename	D5.9 – BRAINE industry 4.0 use case
Nature of deliverable	Report
Dissemination level	PU
Number of pages	21
Work package	WP5
Task(s)	T5.4
Partner responsible	IFX
Author(s)	Hans Ehm (IFX), Thomas Kaminski (IFX), Ming-Yu Tu (IFX)

Editor	Ming-Yu Tu (IFX)

Abstract	This technical report, delivers the detailed information about the development of a semiconductor wafer analysis system designed to integrate and utilize the features available on the BRAINE platform to analyse semiconductor wafer fabrication data in real-time at the edge in order to improve wafer yields.

Keywords	Edge computing, Industry 4.0, AI

Copyright

Deliverable history

Version	Date	Reason	Revised by
0.1	01.03.2023	Table of Contents	Sean Ahearne
0.2	31.05.2023	First Draft	Ming Yu Tu
1.0	20.11.2023	Final Review	Ming Yu Tu, F. Cugini

List of abbreviations and Acronyms

Abbreviation	Meaning
AI	Artificial Intelligence
EMDC	Edge Mobile Data Center
EUR	Euro
GDPR	General Data Protection Regulation
GPU	Graphics Processing Unit
IoT	Internet of Things
KPI	Key Performance Indicator
QSD	Qualified Synthetic Data

Table of Contents

List of Figures

1. Executive summary

This section is brief, and will describe

- Create an AI to improve the quality of semiconductor manufacturing with a microservice architecture to deploy in a Kubernetes cluster.
- Find the optimal exposure time and yields from the optimization and regression models of wafer fabrication
- BRAINE platform provides the resource of edge computing to realize the complex computation
- The impact on KPI’s compared to the state of the art
- BRAINE platform supports reducing in computation time of manufacturing optimization and increasing the wafer yields

2. Use case overview

2.1. Background

Securely connecting manufacturers' devices both locally and to the cloud is paramount for customers to take up the connected service offering. People living in smart homes and working in smart buildings can benefit from the seamless interaction of the sensors. Infineon-enabled solutions in energy, light management, health care, and building operations can improve the quality of life and deliver substantial cost savings. In BRAINE, Infineon will continue investigating new technologies for edge computing that enable the extensive use of AI for different technology fields, in view of further commercialization. Infineon sees BRAINE as an opportunity to explore looking forward to real-time semiconductor manufacturing optimization.

2.2. Motivation

The motivation of this use case is to move from cloud-operated interval planning to more real-time on-premise planning in the semiconductor supply chains and manufacturing for Infineon one of the lead users of the Arrowhead Framework (2013-2017, EUR 65 million, 81 participants) and Tools (2019-2021, EUR 91 million, 81 participants) enabled by the EMDC. Arrowhead was and is Europe's largest automation and digitization project enabling the creation and engineering of IoT-based automation systems. Results of Arrowhead's WP1 (Architecture & Concepts for the digital industry), and WP7 (Productive4.0 framework) of Productive4.0 (2017 to 2020, EUR 109 million, 106 partners) are used. In those two WPs already some adaptation (e.g., with the Semantic Web-based Digital Reference) of Arrowhead toward the semiconductor environment was done. The distributed nature of the Arrowhead Framework based on local clouds allows to separate operations and activities, ensuring e.g., engineering, operation, maintenance, evolution, real-time platform, security, and safety, while still having a common integration platform that will be merged with BRAINE results coming out from the research work packages. Infineon the leading consortium member of Productive4.0 will not only provide the use case but also supports the research to merge former H2020 ECSEL work with BRAINE. Infineon especially brings in its know-how on practical industrial Semantic Web and ontologies as well as supply chain knowledge to make the transfer from research to use case more efficient.

2.3. Objective

- More accurate process control of gate thickness and width, exposure resistance, yields, and supply and turn it into an automatic process with the support of BRAINE.
- Integrate and value-add the BRAINE EMDC federated distributed platform as an appliance that deploys big data and AI tools to complement Arrowhead for the benefit of semiconductor supply chains and supply chains containing semiconductors.

2.4. Goals (KPI’s)

- Reduce the calculation time of optimization and regression models for wafer manufacturing.
- Increase the yield percentage for the batch production.

3. Implementation and Integration

Describe how the use case was implemented and integrated with the BRAINE platform here.

3.1. Use case demonstration

The goal of the use case is to complete the prototype of algorithm to predict the best lithography exposure time to improve gate resistance. The gate resistance is the function of the gate area, which is the multiplication of gate width and layer thickness. The layer thickness is fixed when the substrate of the wafer is formed. In contrast, the gate width is a variable that can be decided by the exposure time set by the machine.

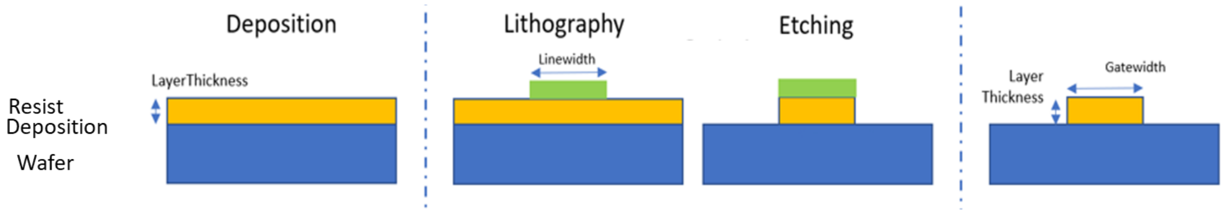

Figure 1: The form of gate area

$$resistance = f(exposure\ time) = f(area) = layer\ thickness * gate\ width$$

3.2. Use case implementation

- Collect process data and create an **AI** to predict the optimal lithography exposure time to maximize yield.
- Collect process data and transform them into **Qualified Synthetic Data** (QSD) to use for the AI.
- Develop an algorithm as **a microservice** which will use the AI model to predict parameters in real time.
- Deploy the microservice in the **Kubernetes** cluster of Infineon with the OpenVPN profile provided by CNIT.

3.3. Integration with the BRAINE platform

- BRAINE supports to generate QSD based on a few wafer measure points.
- BRAINE allows dynamic adjustment in real-time the light exposure time for each coordinate field within a wafer.
- BRAINE accelerates the computation of optimization and regression models to increase yield through edge computing

4. Results

4.1. Data complexity in the semiconductor supply chain

The data domain of the semiconductor supply chain can be represented by the Digital Reference ontology introduced in Productive 4.0. It is a supply chain-related Semantic Web mirror of the semiconductor industry depicting a combination of different supply chain domains and concepts to enable use cases.

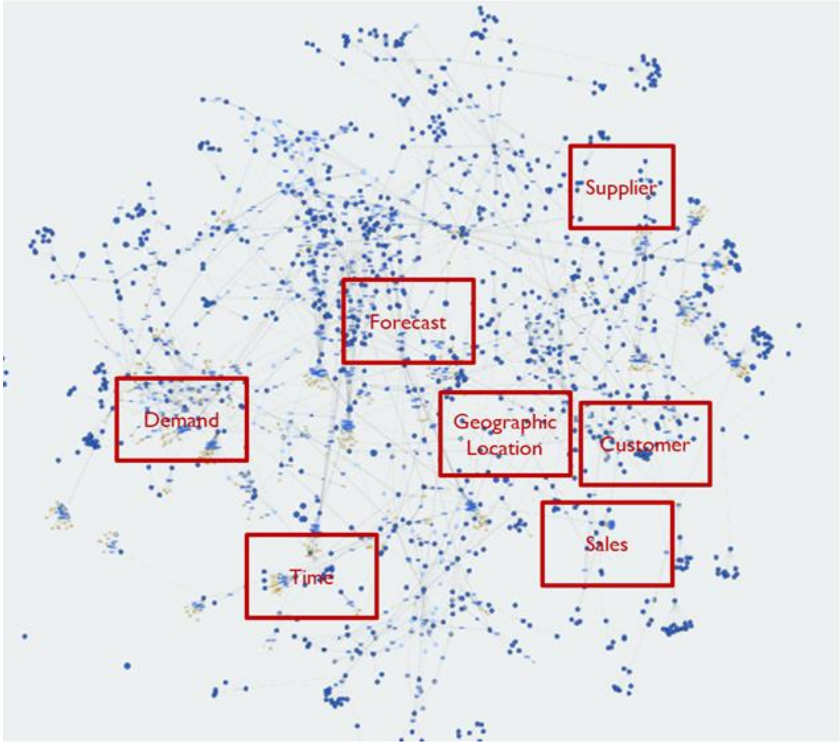

Figure 2: The Digital Reference in the semiconductor supply chain

Further, the domain of the application focuses on the manufacturing step: lithography. Therefore, the intercorrelation of the gate oxide, width, thickness plays a critical role in the data analysis of the use case.

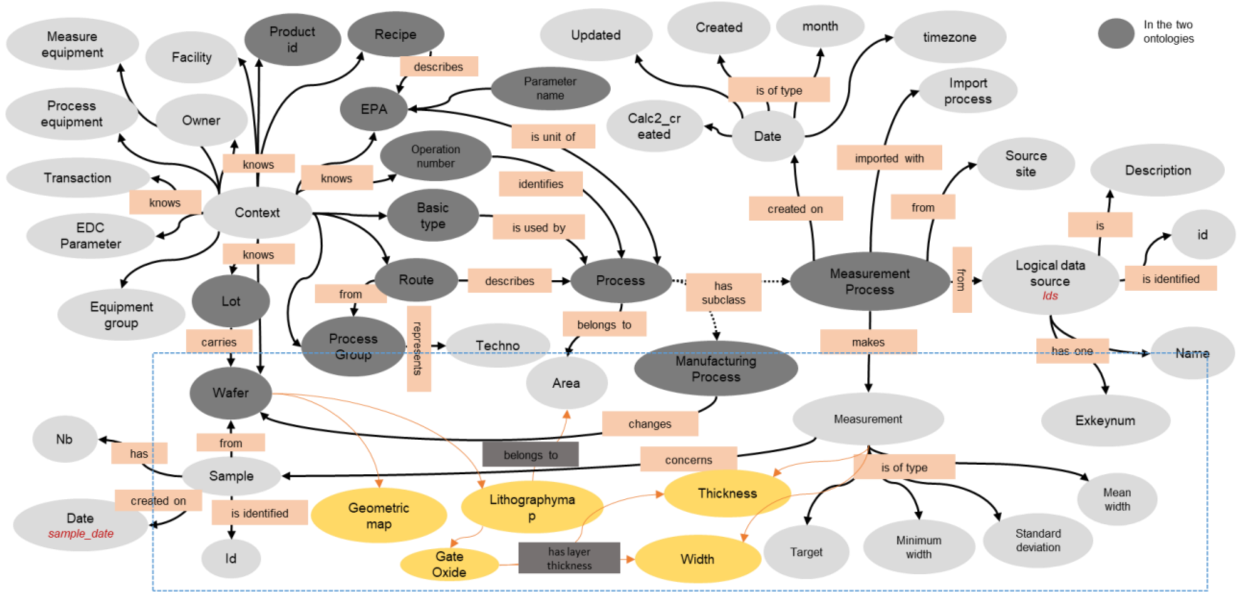

Figure 3: Datalake ontology representation

4.2. Qualified Synthetic Data (QSD)

Gaussian Distribution is the method used for generating Synthetic Data. Wafer, Lot, Die data will also be included in the Synthetic Data. This distribution function is included in the Copulas library, so it is the main function for generating synthetic data.

4.3. Advance of QSD

When the dataset is more complex and the number of columns is high, accuracy is lost. A more robust model that adapts to more complex datasets is needed: Neural Networks. These networks are trainable, they learn to generate patterns through prediction and correction. Two neural network architectures: Generative Adversarial Networks (GAN) and Variational Autoencoders (VAE).

4.4. Simulation of impact on wafer fabrication

We include temperature, pressure, humidity, and thickness parameters in the Qualified Synthetic to simulate the wafer fabrication measures. Qualified Synthetic Data is generated for each die in the Wafer Map, which is a map showing the performance semiconductor devices on a substrate as a color-coded grid.

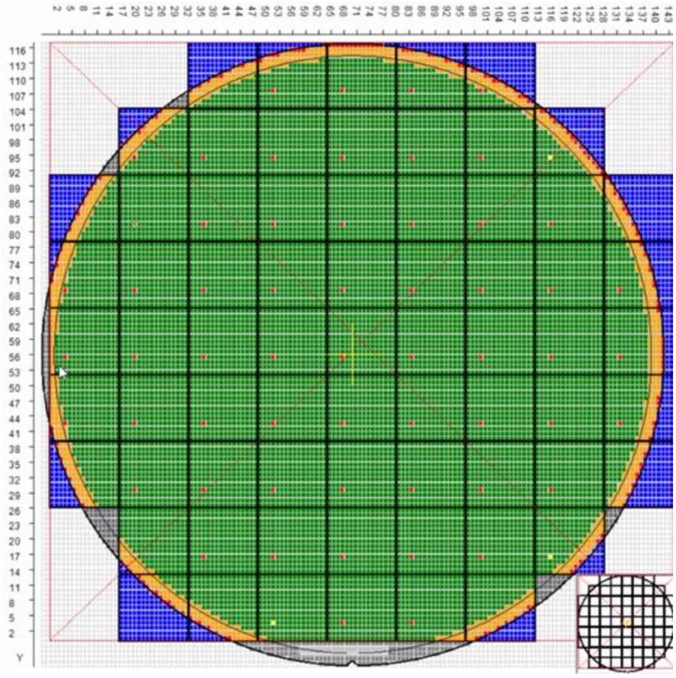

Figure 4: The wafer map

4.5. Optimization model

The goal of the optimization model is to maximize yields while constraining exposure time. We define the following elements required for the model development, including a set, parameters, decision variables, a objective function, and constraints.

<u>Set:</u>

$i \in \{1, \dots, n\}$: a finite set of exposure field

<u>Parameter:</u>

p_i: yield on exposure field i per area

t_i: gate thickness on exposure field i

C : exposure time of a wafer

<u>Decision variables:</u>

w_i: gate width on exposure field i

x_i: decide if the exposure field i is exposed

<u>Objective function:</u>

$$max \sum_{i=1}^{n} p_i t_i w_i x_i \; maximize \; yields$$

Constraints:

$$s.t.\ \sum_{i=1}^{n} t_i w_i x_i \leq C \quad capacity\ constraint$$

$$x \in \{0,\ 1\}^n \quad binary\ constraint$$

$$w_i \geq 0 \quad non - negativity\ constraint$$

4.6. Application Deployment on Kubernetes

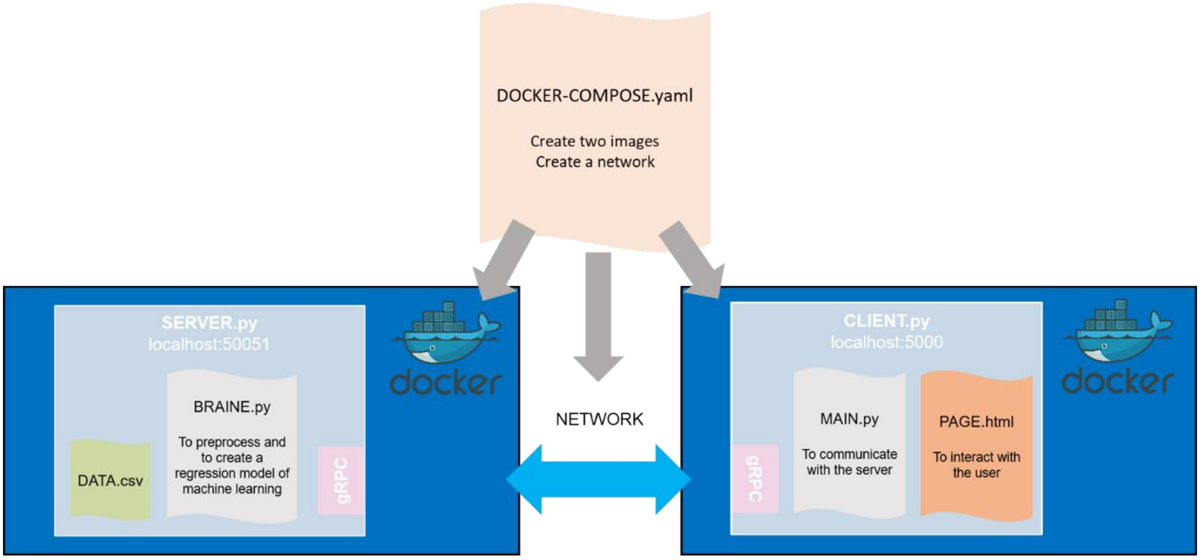

Figure 5: Create docker containers

Building Docker Containers

- Build the server Docker container

The Dockerfile for the server microservice is located within the server directory, navigate to that directory and run the following command from within that directory:

```
docker build -t server-image:latest .
```

This will create a Docker image of the server microservice.

- Build the client Docker container

Same is ture for the client microservice, navigate to the client directory and run the following command from within that directory:

```
docker build -t client-image:latest .
```

This will create a Docker image of the client microservice.

Registering Docker Containers

- Register the server Docker container

To register the server Docker container, we need to have access to a container registry. In this case, we will use our internally hosted BRAINE container registry, which is accessible via the IP address `172.30.101.1`. First, we log in to our registery using the `docker login` command:

```
docker login 172.30.101.1
```

Then, tag the server image with our registery IP address `172.30.101.1` and the name of the repository we want to push the image to:

```
docker tag server-image:latest 172.30.101.1/server-image:latest
```

Finally, we push the server image to our registery:

```
docker push 172.30.101.1/server-image:latest
```

- Register the client Docker container

We follow the same steps as above to register the client Docker container, but use the client image instead:

```
docker tag client-image:latest 172.30.101.1/client-image:latest
docker push 172.30.101.1/client-image:latest
```

Deploying Docker Containers

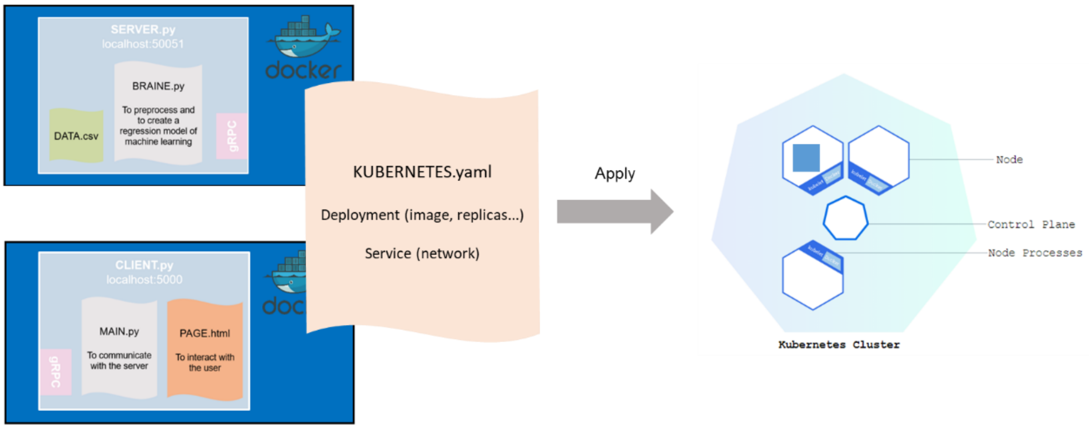

Figure 6: Deploy docker containers

- Deploy the server Docker container

To deploy the server Docker container, we will use a container orchestration tool, namely Kubernetes. The `kubernetes.yaml` file is located in the project root, we can create the deployment by running the following command from within the project root directory:

```
kubectl apply -f kubernetes.yaml
```

This will create a Kubernetes deployment of both defined microservices.

- Perform some Kubernetes checkups

We can check the status of the deployment by running:

```
kubectl get deployments
```

We expect something like this to be the output:

Step 3 Example Output

```
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
client    1/1     1            1           4m47s
server    1/1     1            1           4m47s
```

To check whether the pods are running, we can use this:

```
kubectl get pods
```

Our output should match the following:

Step 4 Example Output

```
NAME           READY   STATUS    RESTARTS   AGE
client-75f8    1/1     Running   0          4m52s
server-7656    1/1     Running   0          4m52s
```

Lastly for more information we can look into the events by running the command:

```
kubectl get events
```

We should have something simmilar to the following as a result:

Step 5 Example Output

```
LAST SEEN   TYPE     REASON              OBJECT
MESSAGE
5m9s        Normal   Scheduled           pod/client-75f8
Successfully assigned default/client-75f8 to uc4-w1
5m8s        Normal   Pulling             pod/client-75f8
Pulling image "172.30.101.1:5000/client_py"
```

```
4m49s        Normal   Pulled               pod/client-75f8
Successfully pulled image "172.30.101.1:5000/client_py" in
19.196572875s
4m48s        Normal   Created              pod/client-75f8
Created container client-py
4m48s        Normal   Started              pod/client-75f8
Started container client-py
5m9s         Normal   SuccessfulCreate     replicaset/client-
75f8          Created pod: client-75f8
5m9s         Normal   ScalingReplicaSet    deployment/client
Scaled up replica set client-75f8 to 1

5m9s         Normal   Scheduled            pod/server-7656
Successfully assigned default/server-7656 to uc4-w2
5m8s         Normal   Pulling              pod/server-7656
Pulling image "172.30.101.1:5000/server_py"
4m35s        Normal   Pulled               pod/server-7656
Successfully pulled image "172.30.101.1:5000/server_py" in
33.424867625s
4m34s        Normal   Created              pod/server-7656
Created container server-py
4m34s        Normal   Started              pod/server-7656
Started container server-py
5m9s         Normal   SuccessfulCreate     replicaset/server-
7656          Created pod: server-7656
5m9s         Normal   ScalingReplicaSet    deployment/server
Scaled up replica set server-7656 to 1
```

Run the deployed application

After deploying the app we can run the following command to execute the application

```
kubectl exec client-75f8 -- curl localhost:5000
```

- where `client-75f8` is our client container
- and everything after the `--` are the commands we execute in the pod

As a result we get the html file:

```
braine@UC4-m:~$ kubectl exec client-75f8 -- curl localhost:5000 %
Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed 100 519 100 519 0 0 18 0
0:00:28 0:00:28 --:--:-- 126 <!-- homepage.html --> <!doctype
html> <html lang="en"> <head> <title>Prediction of
photolithography width</title> </head> <body> <h1>Creating QSD
dataset and testing it on the Braine_solver_cplex algorithm to
measure the computing time </h1> <p> Data sent: ext_mv = 10.5,
ext_min = 11.8, ext_max = 14.2, spec_upper = 7, ext_q1 = 6,
ext_q3 = 3, raw_values = 12, ext_ewma_mv = 4, ext_ms_mv = 20,
ext_mv_ucl = 10 </p> <p>computation_time: 131.878357 </p> </body>
```

5. Impact

5.1. Comparison to existing systems

The previous system used by Infineon is not able to consider multiple manufacturing parameters in the optimization process. In contrast, BRAINE enables real-time optimization by loading all the parameters required in production. The following table shows the statistical results of the execution time from 20 iterations of 10,000 samples.

Average	Standard deviation	Minimum	Maximum	Medium
0.0162 s	0.0027 s	0.0121 s	0.0205 s	0.0166 s

5.2. Potential Impact to the semiconductor wafer manufacturing

- Improve the current state of decision making for gate width, directly leading to optimal exposure time and yields for chip production.
- Improve the development of secured cloud connection-enabled applications; these can range from motion detection up to situational awareness, by leveraging AI and machine learning algorithms.
- Enable a more reliable decision making with support of data integration.

5.3. Advantages of the BRAINE platform

- Provide a platform for real-time computation and reduce the calculation time to optimize manufacturing yields.
- Support efficient data integration from sensors into the cloud service.

5.4. Business solutions and economic advantages of BRAINE for semiconductor wafer fabrication

- Drastically reduce the computation time to make optimal manufacturing planning and decision
- Increase wafer production yields in every manufacturing batch and improve overall production quality.

6. Conclusion

Supported by BRAINE, Infineon improves the current state of real-time decision making of wafer production. This enables a more reliable decision making, due to more data integration in the cloud. The BRAINE platform is an opportunity for the supply chain department to increase production yields and improve the overall quality of the supply chain management at Infineon.