10th SummerSOC
29. June 2016, Crete

Situation Model as Interface between Situation Recognition and Situation-Aware Applications

SATOPT

Mathias Mormul, Pascal Hirmer, **Matthias Wieland**, and Bernhard Mitschang
Institute of Parallel and Distributed Systems
University of Stuttgart
Stuttgart, Germany

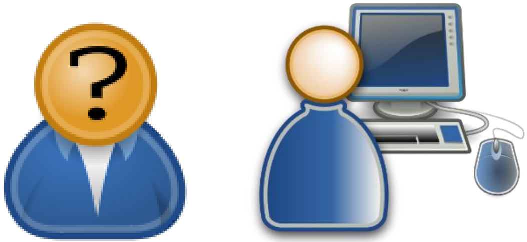
University of Stuttgart
Germany

IAAS

as
Anwendungssoftware

Motivation – The Challenge of The Internet of Things

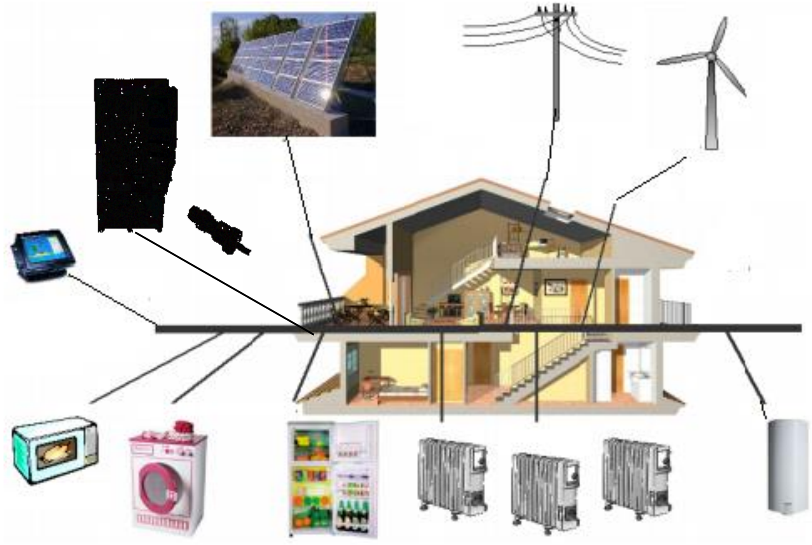
The user has to process the huge amounts of data



Setup applications or trigger actors manually

Sensors providing context data about the environment

Actors for changing the environment



intechopen.com, CC

Smart Devices

intechopen.com, CC

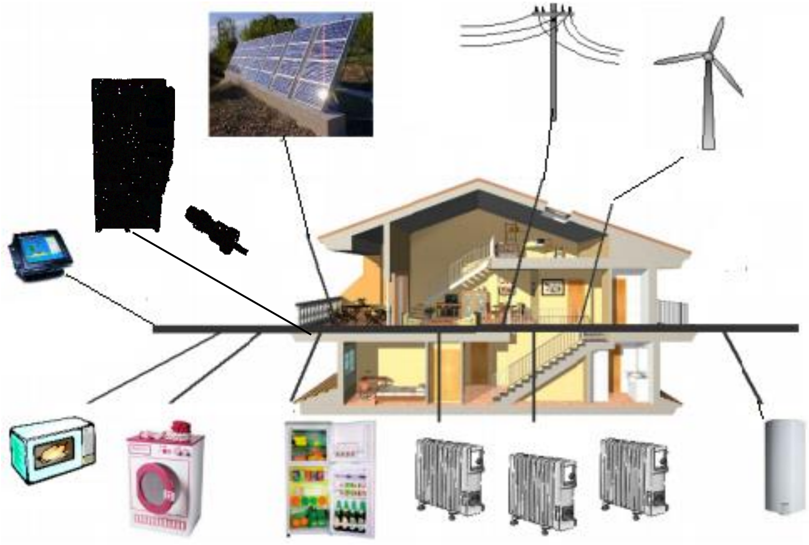
Motivation – Situation Model for Decoupling

If a specific **situation** occurs in my smart environment then **trigger** the following action.

Situation Model for management of: Things, Sensors, Actuators, Situations

Sensors providing context data about the environment

Actuators for changing the environment



intechopen.com, CC

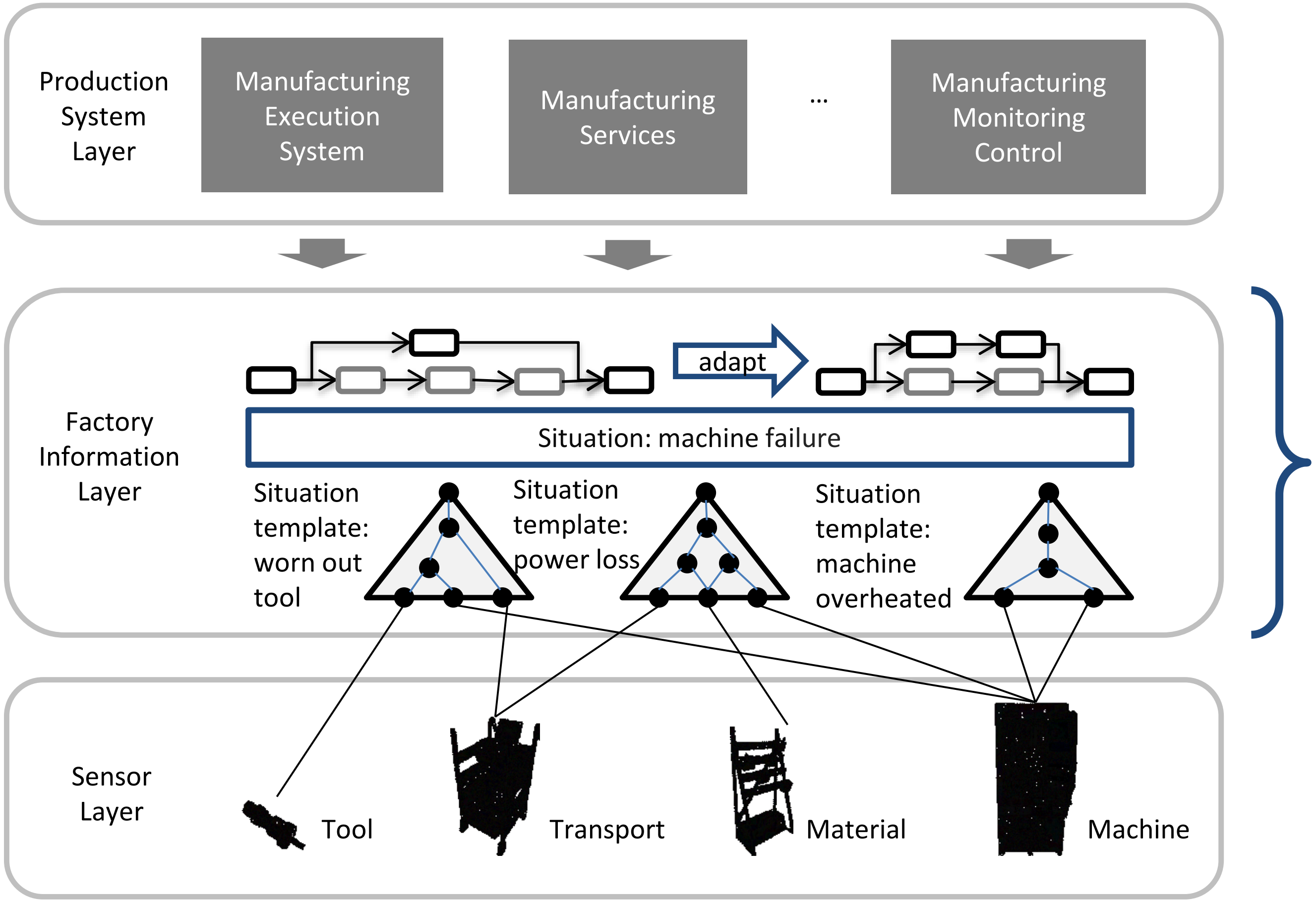
intechopen.com, CC

Paper Contribution and Agenda

- Situation Model for modeling and management of Situations in Internet of Things environments
- Optimized integration of Situation Recognition and Situation-aware Applications
- Agenda
  - (1) Problem Statement and “Industrie 4.0” Scenario
  - (2) Definition of the Situation Model
  - (3) Extended SitOPT Architecture
    - (1) Situation Model Management (SMM)
    - (2) Situation Management Layer Architecture
  - (4) Implementation and Evaluation of the SMM
  - (5) Summary and Future Work

Problem Statement – Situation Recognition in “Industrie 4.0”

If a specific **situation** occurs in my smart environment then **adapt** my workflow



Production System Layer

Manufacturing Execution System

Manufacturing Services

...

Manufacturing Monitoring Control

Factory Information Layer

adapt

Situation: machine failure

Situation template: worn out tool

Situation template: power loss

Situation template: machine overheated

Sensor Layer

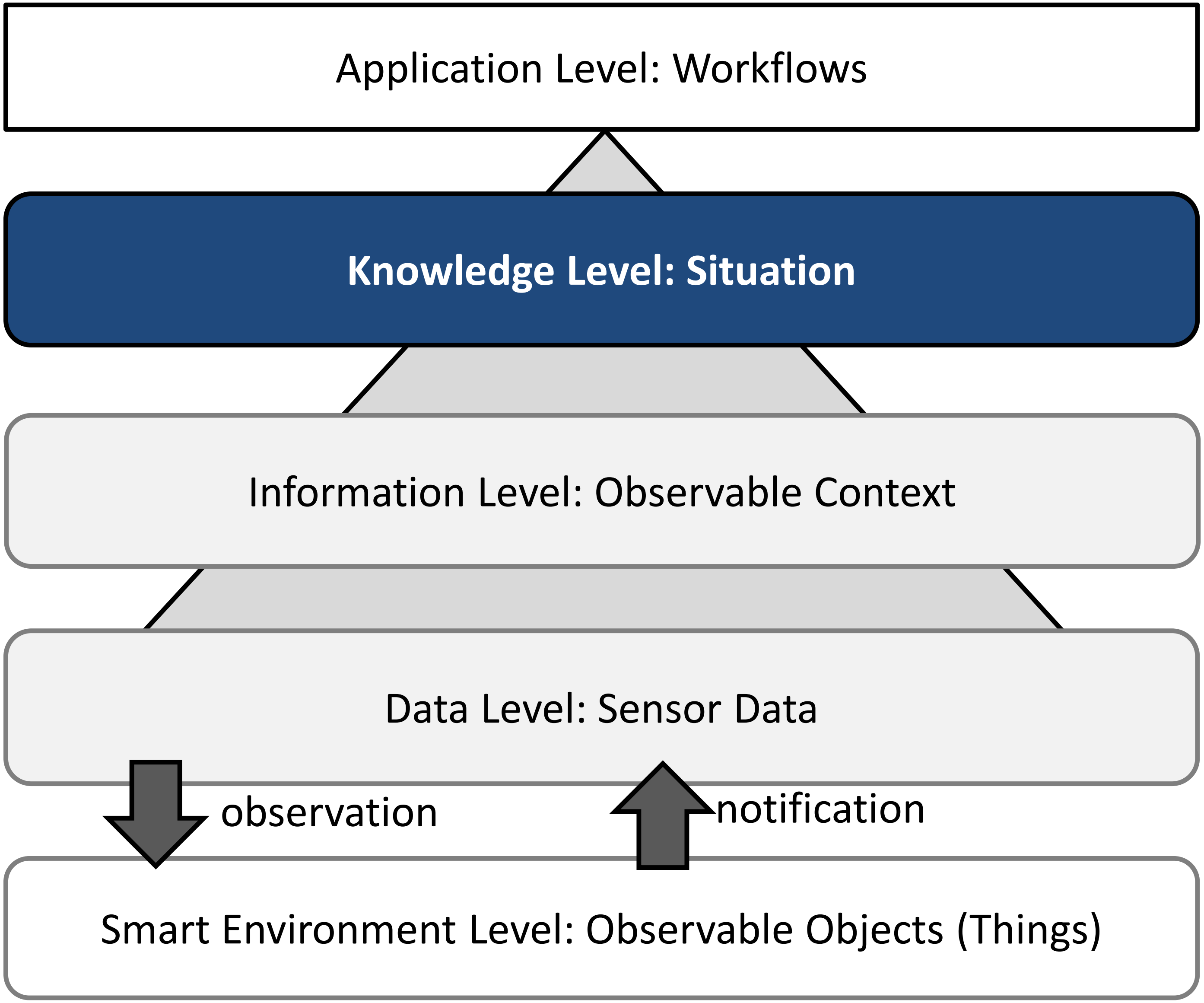
Tool

Transport

Material

Machine

What is a Situation? Context Based Knowledge Generation



Application Level: Workflows

1. Register situation recognition for machine
2. Trigger actions based on situations occurring

Knowledge Level: Situation

Situation-Model, e.g. state of machine “OK” or “Failure”

Information Level: Observable Context

Context-Model, e.g. Machine.Temp

Data Level: Sensor Data

Basic data types, e.g. °C of Sensor

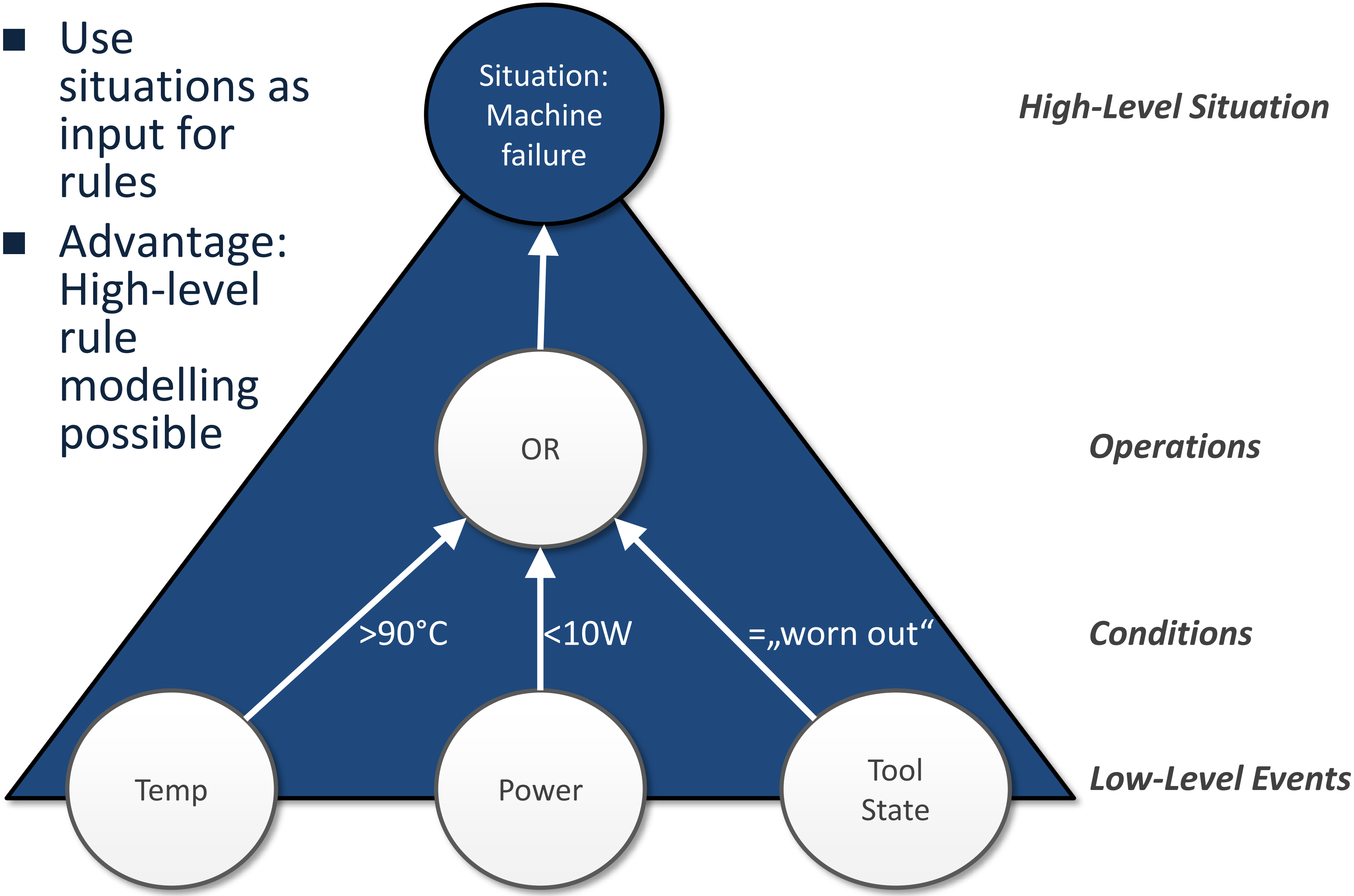
observation

notification

Smart Environment Level: Observable Objects (Things)

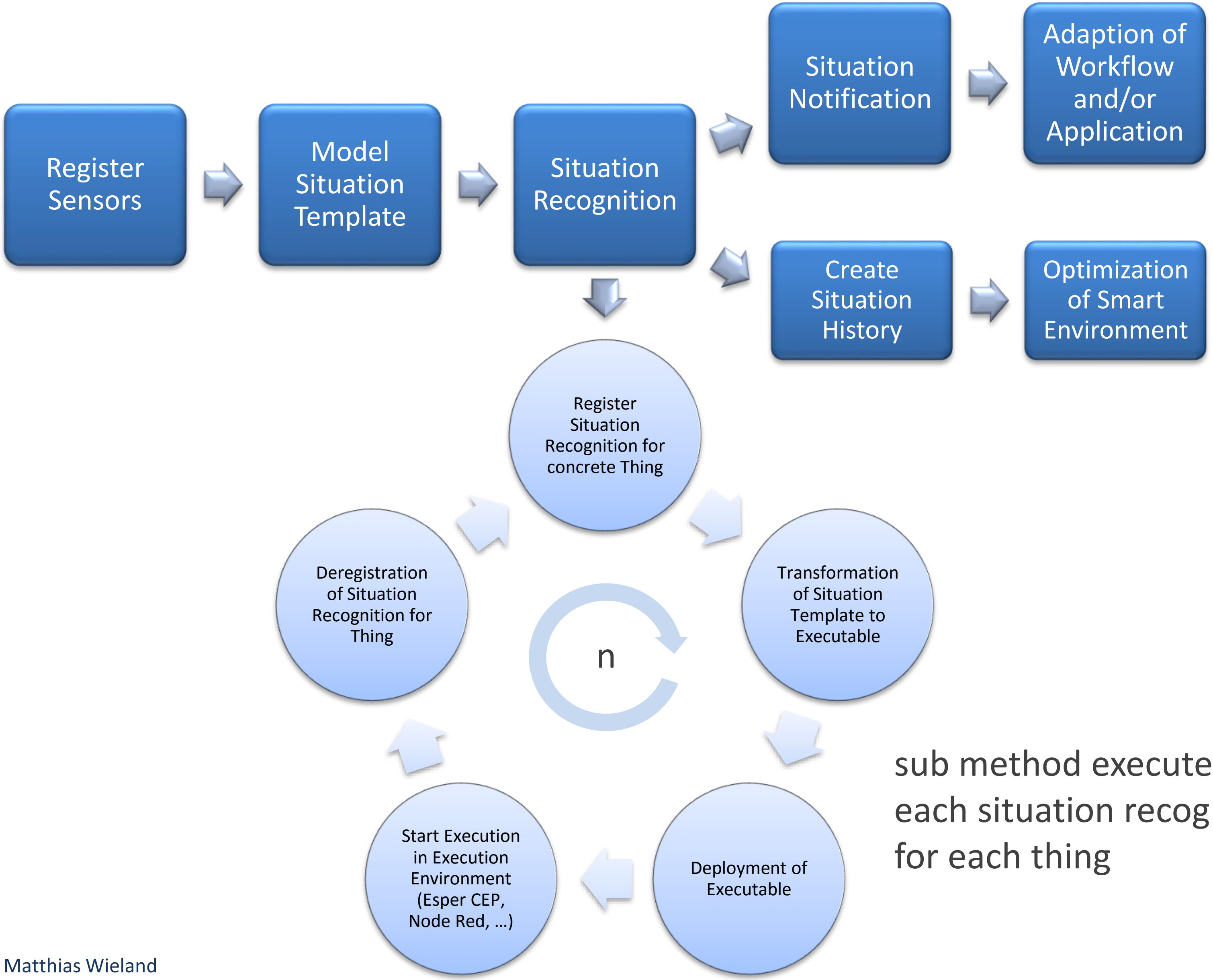
Modeling of Situations as Situation Templates

- Use situations as input for rules
- Advantage: High-level rule modelling possible



© Matthias Wieland

Complete Method for Situation Recognition



Register Sensors

Model Situation Template

Situation Recognition

Situation Notification

Adaption of Workflow and/or Application

Create Situation History

Optimization of Smart Environment

Register Situation Recognition for concrete Thing

Transformation of Situation Template to Executable

Deployment of Executable

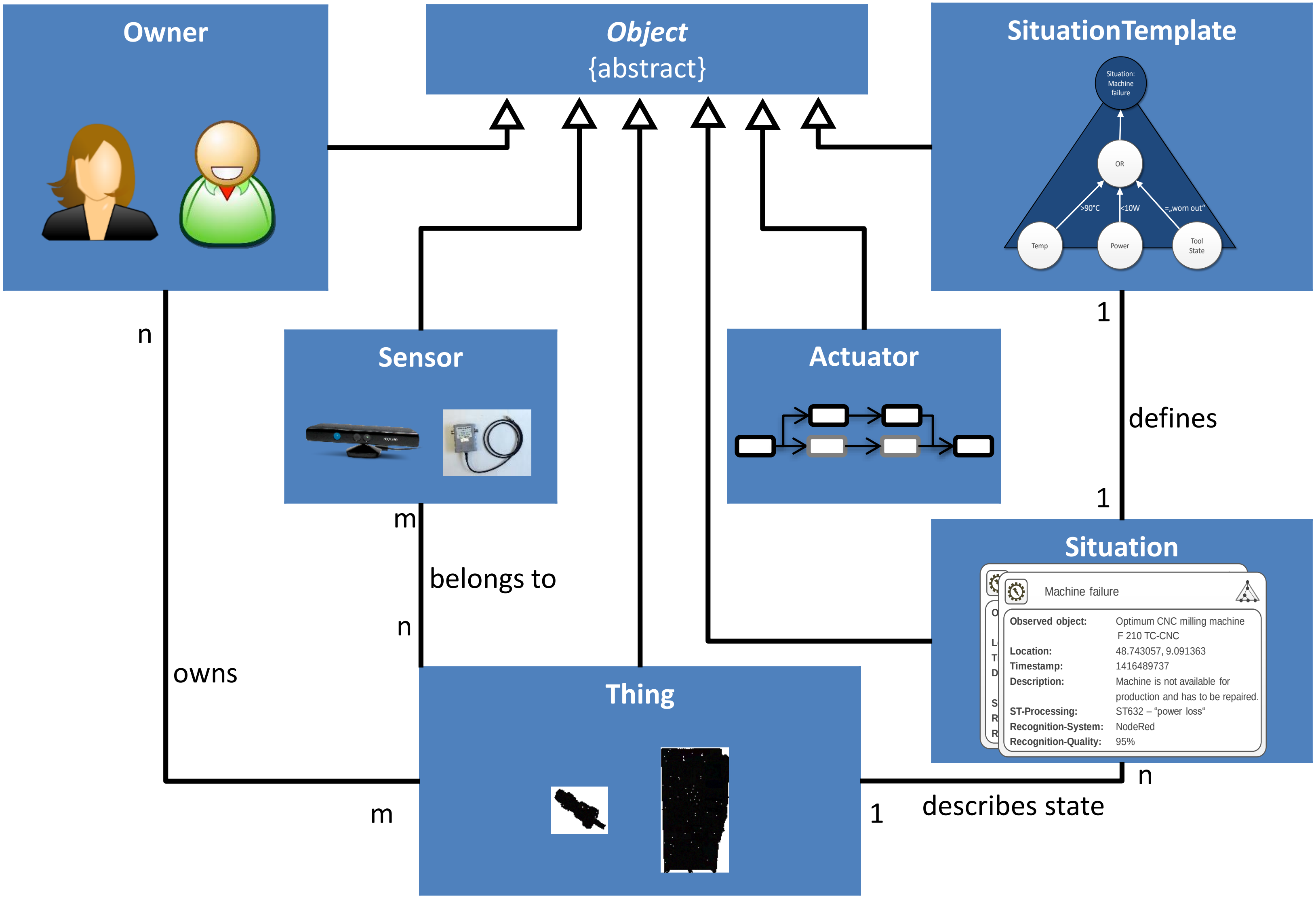
Start Execution in Execution Environment (Esper CEP, Node Red, …)

Deregistration of Situation Recognition for Thing

n

sub method executed for each situation recognition for each thing

Definition of the Situation Model – Relationships



Machine failure

Observed object: Optimum CNC milling machine F 210 TC-CNC

Location: 48.743057, 9.091363

Timestamp: 1416489737

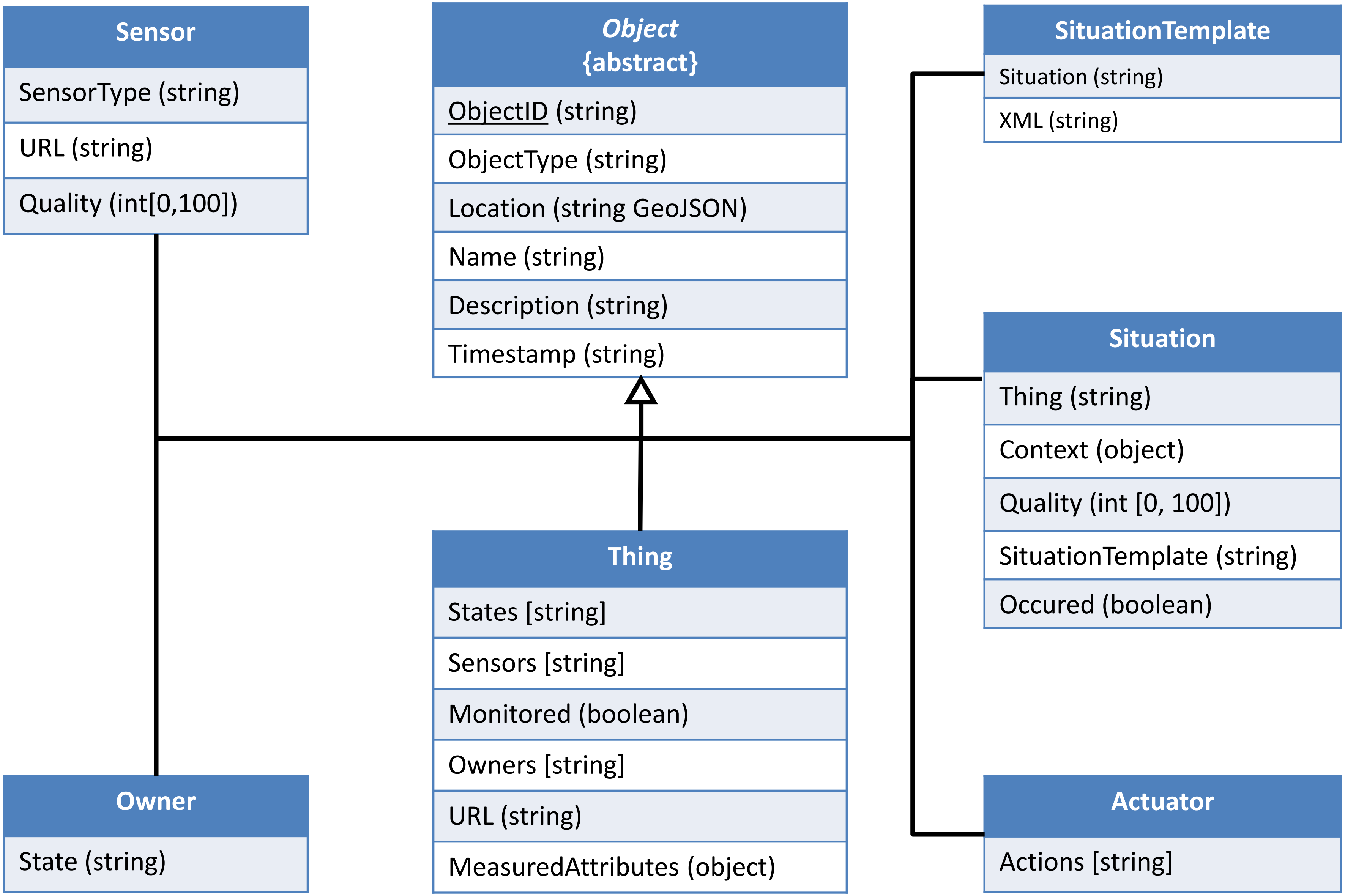
Description: Machine is not available for production and has to be repaired.

ST-Processing: ST632 – “power loss“

Recognition-System: NodeRed

Recognition-Quality: 95%

Definition of the Situation Model – Attributes



Sensor
- SensorType (string)
- URL (string)
- Quality (int[0,100])

Object
{abstract}
- ObjectID (string)
- ObjectType (string)
- Location (string GeoJSON)
- Name (string)
- Description (string)
- Timestamp (string)

SituationTemplate
- Situation (string)
- XML (string)

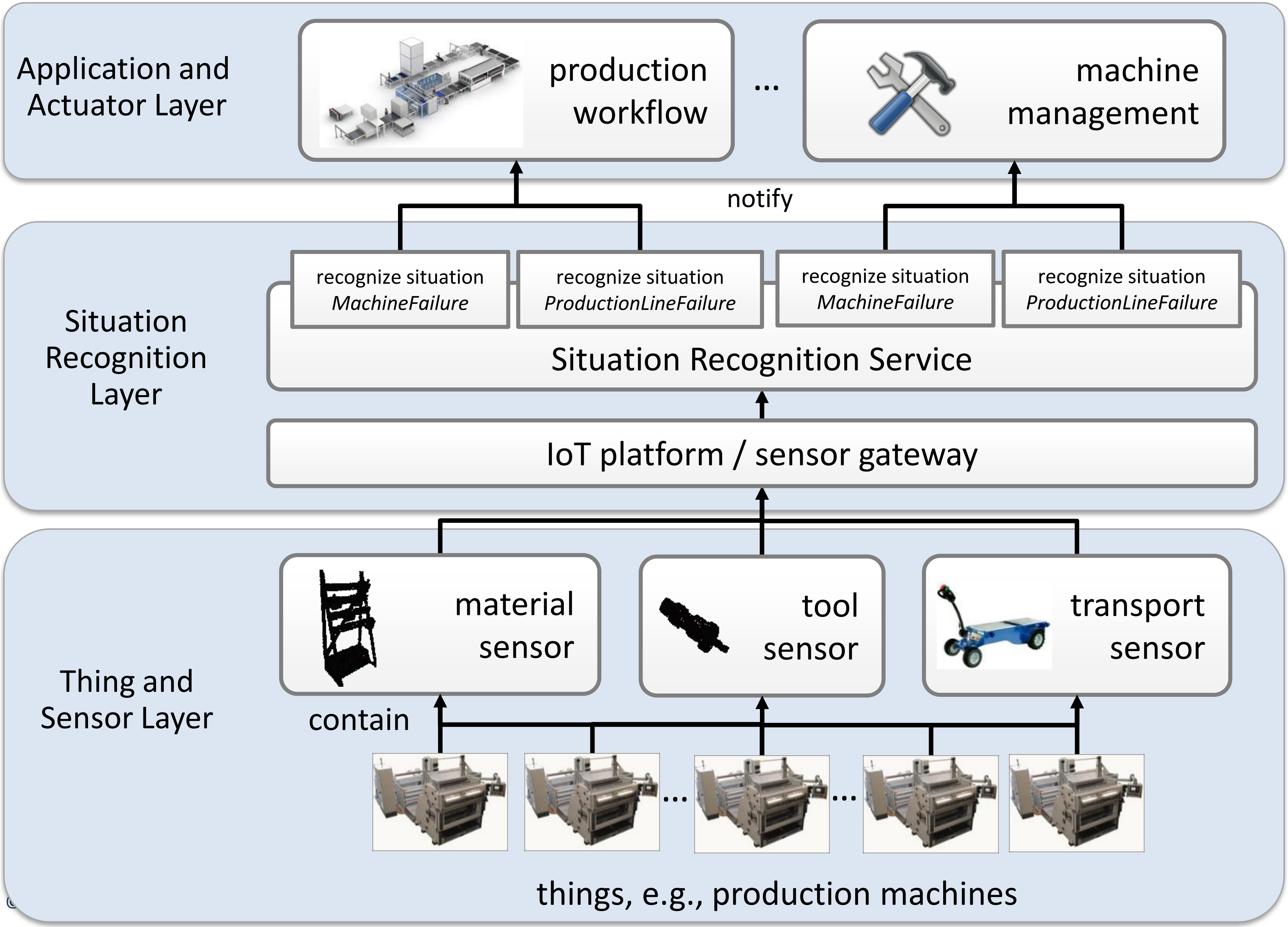
Situation
- Thing (string)
- Context (object)
- Quality (int [0, 100])
- SituationTemplate (string)
- Occured (boolean)

Thing
- States [string]
- Sensors [string]
- Monitored (boolean)
- Owners [string]
- URL (string)
- MeasuredAttributes (object)

Owner
- State (string)

Actuator
- Actions [string]

SitOPT Architecture Without Extension



Application and Actuator Layer

production workflow

…

machine management

notify

Situation Recognition Layer

recognize situation *MachineFailure*

recognize situation *ProductionLineFailure*

recognize situation *MachineFailure*

recognize situation *ProductionLineFailure*

Situation Recognition Service

IoT platform / sensor gateway

Thing and Sensor Layer

material sensor

tool sensor

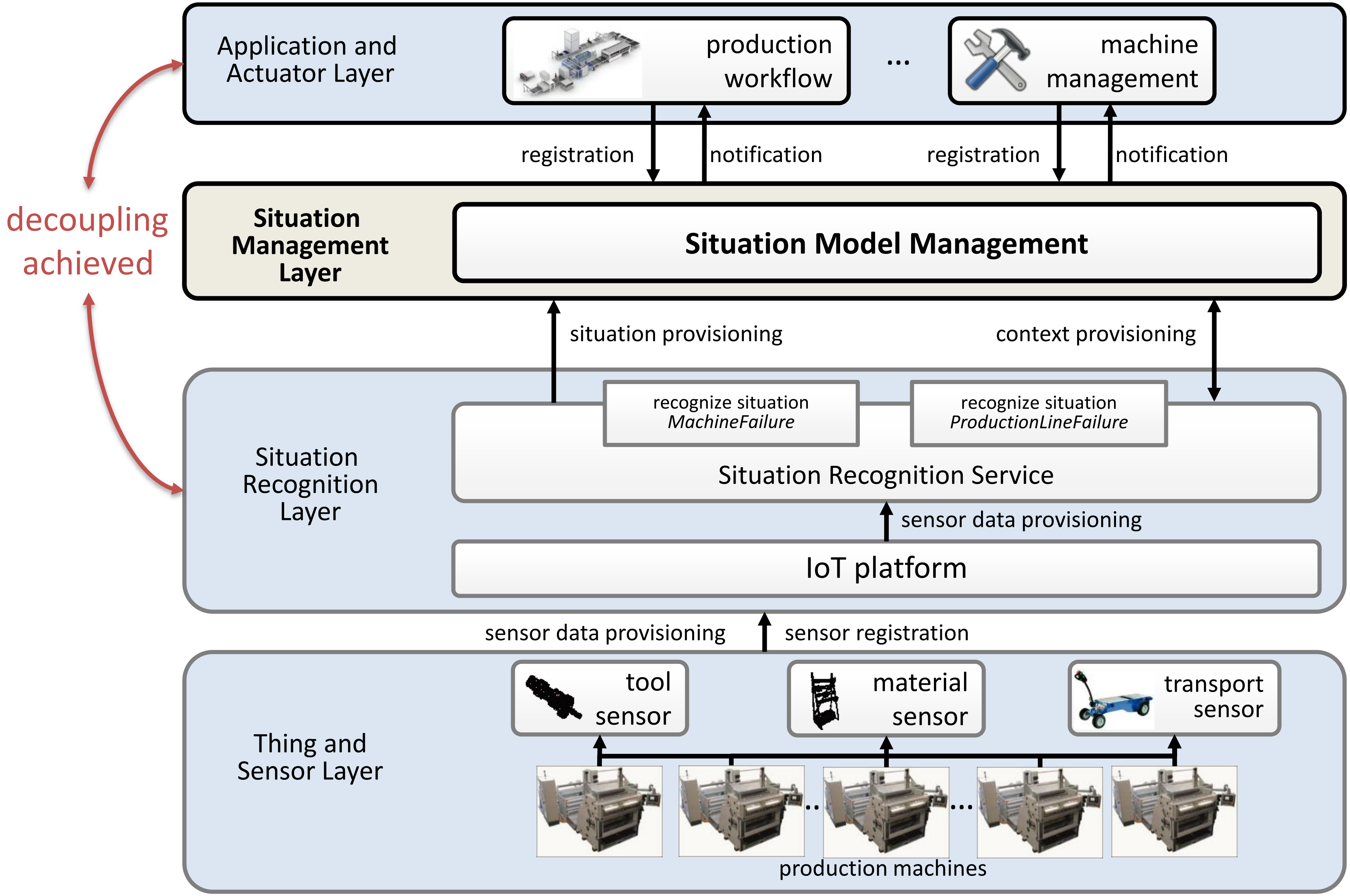
transport sensor

contain

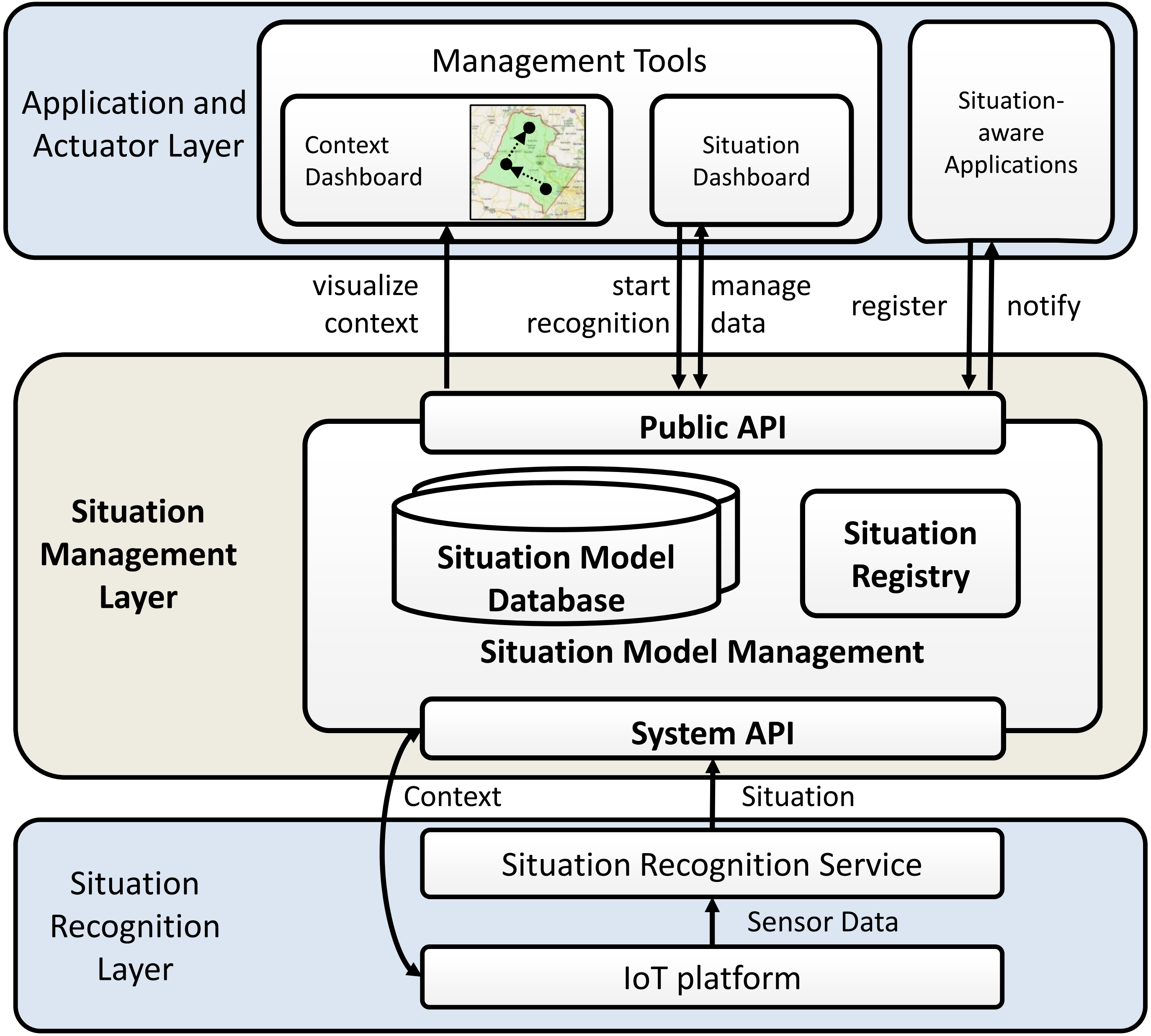
… …

things, e.g., production machines

Extended Architecture – New Situation Management Layer



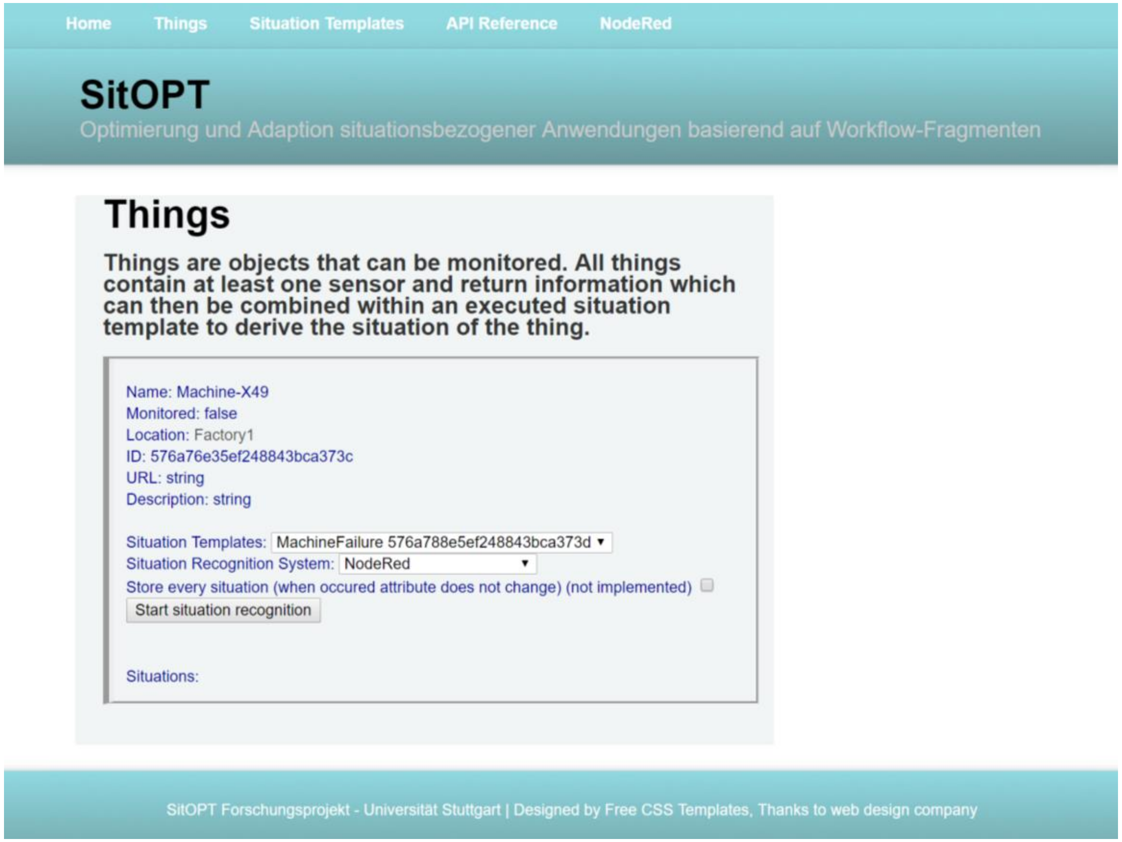
Architecture of the Situation Management Layer



© Matthias Wieland

Situation Dashboard

- List all things
- Show available Situation Templates
- Start situation recognition using different systems



SitOPT

Optimierung und Adaption situationsbezogener Anwendungen basierend auf Workflow-Fragmenten

Things

Things are objects that can be monitored. All things contain at least one sensor and return information which can then be combined within an executed situation template to derive the situation of the thing.

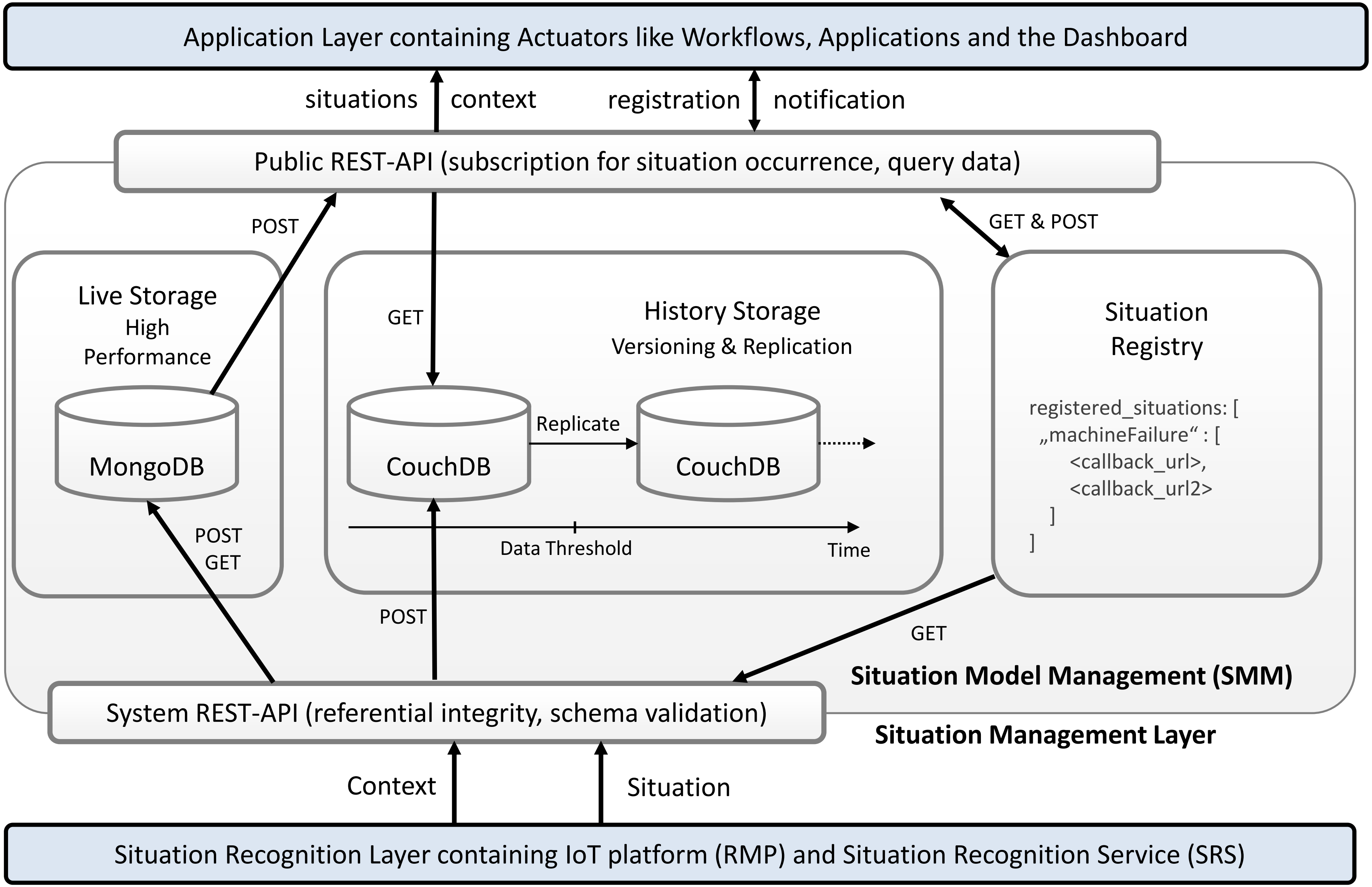
Name: Machine-X49
Monitored: false
Location: Factory1
ID: 576a76e35ef248843bca373c
URL: string
Description: string

Situation Templates: MachineFailure 576a788e5ef248843bca373d ▾
Situation Recognition System: NodeRed ▾
Store every situation (when occured attribute does not change) (not implemented) □

Situations:

SitOPT Forschungsprojekt - Universität Stuttgart | Designed by Free CSS Templates, Thanks to web design company

Distributed Implementation of the SMM



Application Layer containing Actuators like Workflows, Applications and the Dashboard

situations

context

registration

notification

Public REST-API (subscription for situation occurrence, query data)

POST

GET & POST

GET

Live Storage

High Performance

MongoDB

History Storage

Versioning & Replication

CouchDB

Replicate

CouchDB

Data Threshold

Time

Situation Registry

```
registered_situations: [
  „machineFailure“ : [
      <callback_url>,
      <callback_url2>
   ]
]
```

POST

GET

POST

GET

Situation Model Management (SMM)

System REST-API (referential integrity, schema validation)

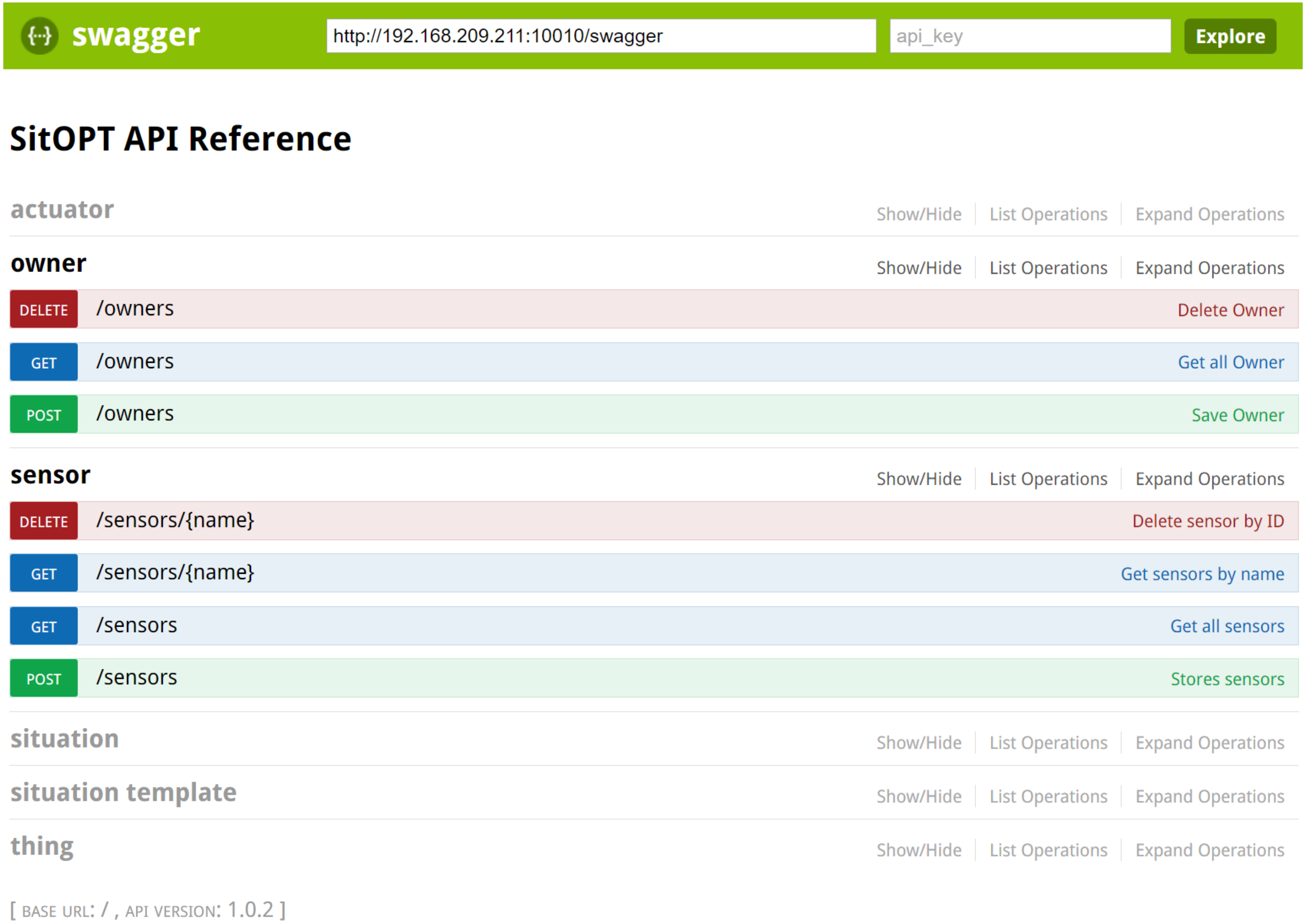
Situation Management Layer

Context

Situation

Situation Recognition Layer containing IoT platform (RMP) and Situation Recognition Service (SRS)

Public and System API Definition Using Swagger



SitOPT API Reference

actuator · Show/Hide · List Operations · Expand Operations

owner · Show/Hide · List Operations · Expand Operations

- DELETE /owners — Delete Owner
- GET /owners — Get all Owner
- POST /owners — Save Owner

sensor · Show/Hide · List Operations · Expand Operations

- DELETE /sensors/{name} — Delete sensor by ID
- GET /sensors/{name} — Get sensors by name
- GET /sensors — Get all sensors
- POST /sensors — Stores sensors

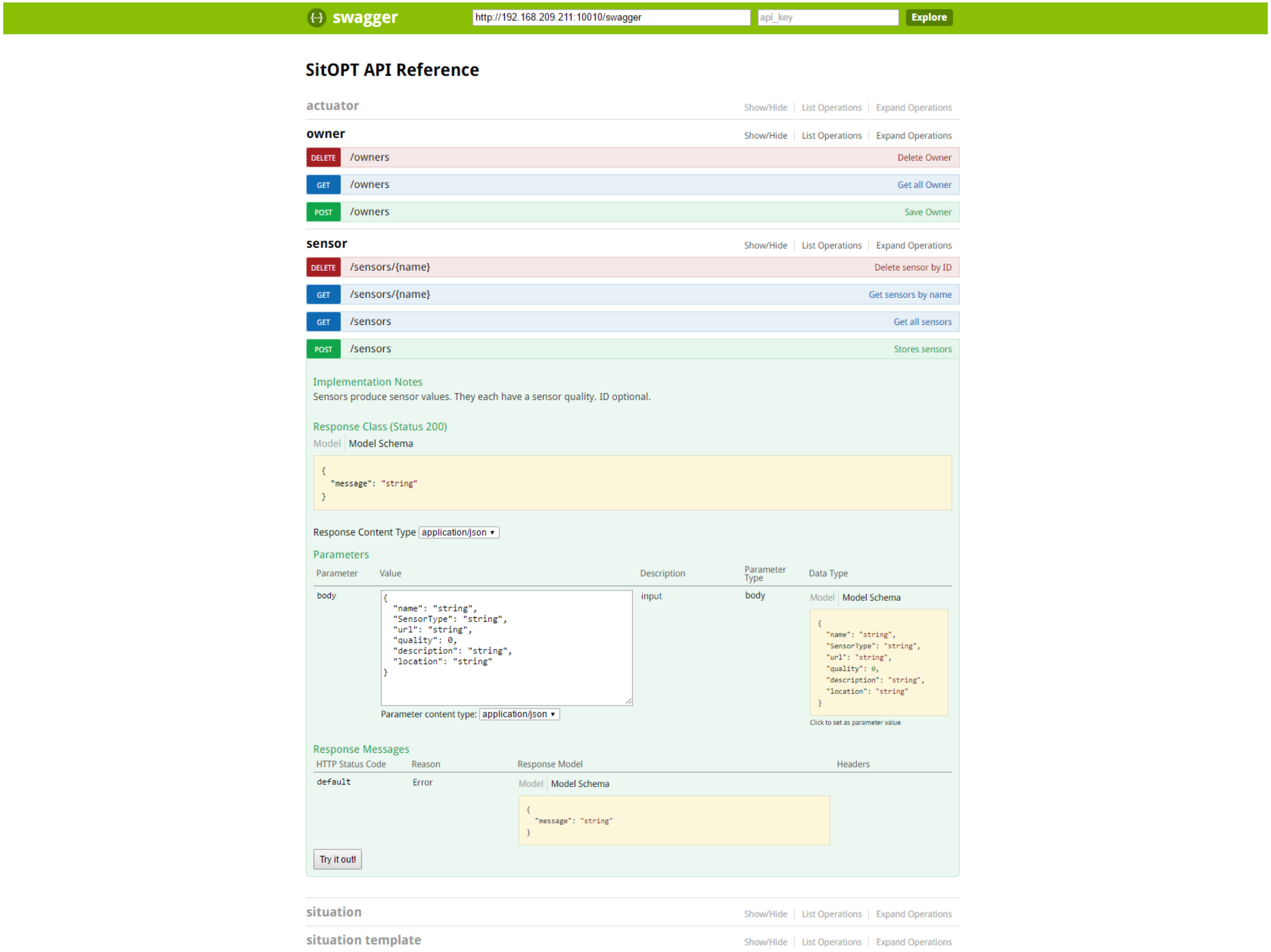
situation · Show/Hide · List Operations · Expand Operations

situation template · Show/Hide · List Operations · Expand Operations

thing · Show/Hide · List Operations · Expand Operations

[BASE URL: / , API VERSION: 1.0.2]

Web-Based Testing and Debugging Provided



SitOPT API Reference

actuator — Show/Hide | List Operations | Expand Operations

owner — Show/Hide | List Operations | Expand Operations

DELETE	/owners	Delete Owner
GET	/owners	Get all Owner
POST	/owners	Save Owner

sensor — Show/Hide | List Operations | Expand Operations

DELETE	/sensors/{name}	Delete sensor by ID
GET	/sensors/{name}	Get sensors by name
GET	/sensors	Get all sensors
POST	/sensors	Stores sensors

Implementation Notes

Sensors produce sensor values. They each have a sensor quality. ID optional.

Response Class (Status 200)

Model | Model Schema

```
{
  "message": "string"
}
```

Response Content Type: application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
body	{ "name": "string", "SensorType": "string", "url": "string", "quality": 0, "description": "string", "location": "string" } Parameter content type: application/json	input	body	Model | Model Schema { "name": "string", "SensorType": "string", "url": "string", "quality": 0, "description": "string", "location": "string" } Click to set as parameter value

```
{
  "name": "string",
  "SensorType": "string",
  "url": "string",
  "quality": 0,
  "description": "string",
  "location": "string"
}
```

Response Messages

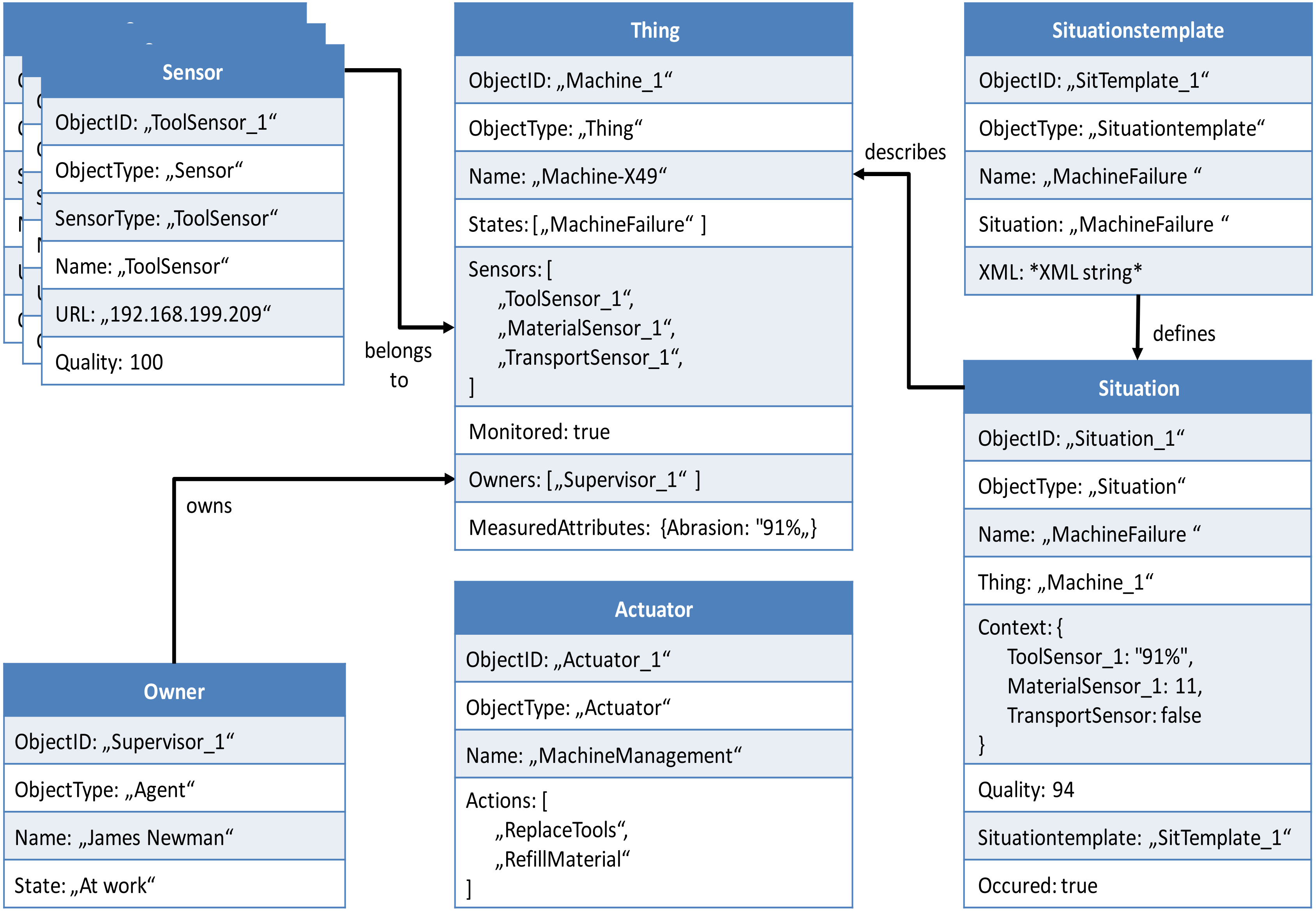
HTTP Status Code	Reason	Response Model	Headers
default	Error	Model | Model Schema { "message": "string" }	

Try it out!

situation — Show/Hide | List Operations | Expand Operations

situation template — Show/Hide | List Operations | Expand Operations

Example Instance of the Scenario for Document Store



Sensor

ObjectID: „ToolSensor_1“

ObjectType: „Sensor“

SensorType: „ToolSensor“

Name: „ToolSensor“

URL: „192.168.199.209“

Quality: 100

Thing

ObjectID: „Machine_1“

ObjectType: „Thing“

Name: „Machine-X49“

States: [„MachineFailure“]

Sensors: [
„ToolSensor_1“,
„MaterialSensor_1“,
„TransportSensor_1“,
]

Monitored: true

Owners: [„Supervisor_1“]

MeasuredAttributes: {Abrasion: "91%„}

Situationstemplate

ObjectID: „SitTemplate_1“

ObjectType: „Situationtemplate“

Name: „MachineFailure “

Situation: „MachineFailure “

XML: *XML string*

Situation

ObjectID: „Situation_1“

ObjectType: „Situation“

Name: „MachineFailure “

Thing: „Machine_1“

Context: {
ToolSensor_1: "91%",
MaterialSensor_1: 11,
TransportSensor: false
}

Quality: 94

Situationtemplate: „SitTemplate_1“

Occured: true

Owner

ObjectID: „Supervisor_1“

ObjectType: „Agent“

Name: „James Newman“

State: „At work“

Actuator

ObjectID: „Actuator_1“

ObjectType: „Actuator“

Name: „MachineManagement“

Actions: [
„ReplaceTools“,
„RefillMaterial“
]

Evaluation of Situation Model Management

Runtimes (ms)	Min	Max	Avg
MongoDB (Live)	5 ms	66 ms	21.87 ms
CouchDB (History)	124 ms	300 ms	246.13 ms

Situation object with #context	data / day	data / year
10	33,8 MB	12 GB
50	103 MB	36,6 GB
200	370,7 MB	131,8 GB
1000	1,74 GB	625,9 GB

Summary and Outlook

SitOPT is a general purpose, situation-aware, and adaptive workflow-system that can be used in different use-cases

- Summary
  - **Situation Model** for modeling and management of **Situations in Internet of Things** and for defining **Situation Objects** characterizing the state of the environment
  - **Optimized** integration of Situation Recognition and Situation-aware Applications with new **Situation Management Layer**
  - SitOPT is capable of **integrating different processing technologies**
    - Data streaming, Complex Event Processing, Internet of Things technologies
    - **Provides decoupling of workflow-modeling and its adaption** logic from the situation recognition

- Planned future work
  - **High quality** in manufacturing needed → No unnecessary downtime
  - Model quality **throughout** the recognition → sensors, values, process
  - **Prediction** of situations based of created context and situation history

Dr. rer. nat. Matthias Wieland

Contact

Phone	+49 711 685 88235
Fax	+49 711 685 88424
Email	Matthias.Wieland@ipvs.uni-stuttgart.de
Address	Universitätsstraße 38
	D-70569 Stuttgart

Situation Model as Interface between Situation Recognition and Situation-Aware Applications

Mathias Mormul, Pascal Hirmer, **Matthias Wieland**, and Bernhard Mitschang
Institute of Parallel and Distributed Systems
University of Stuttgart
Stuttgart, Germany

University of Stuttgart
Germany