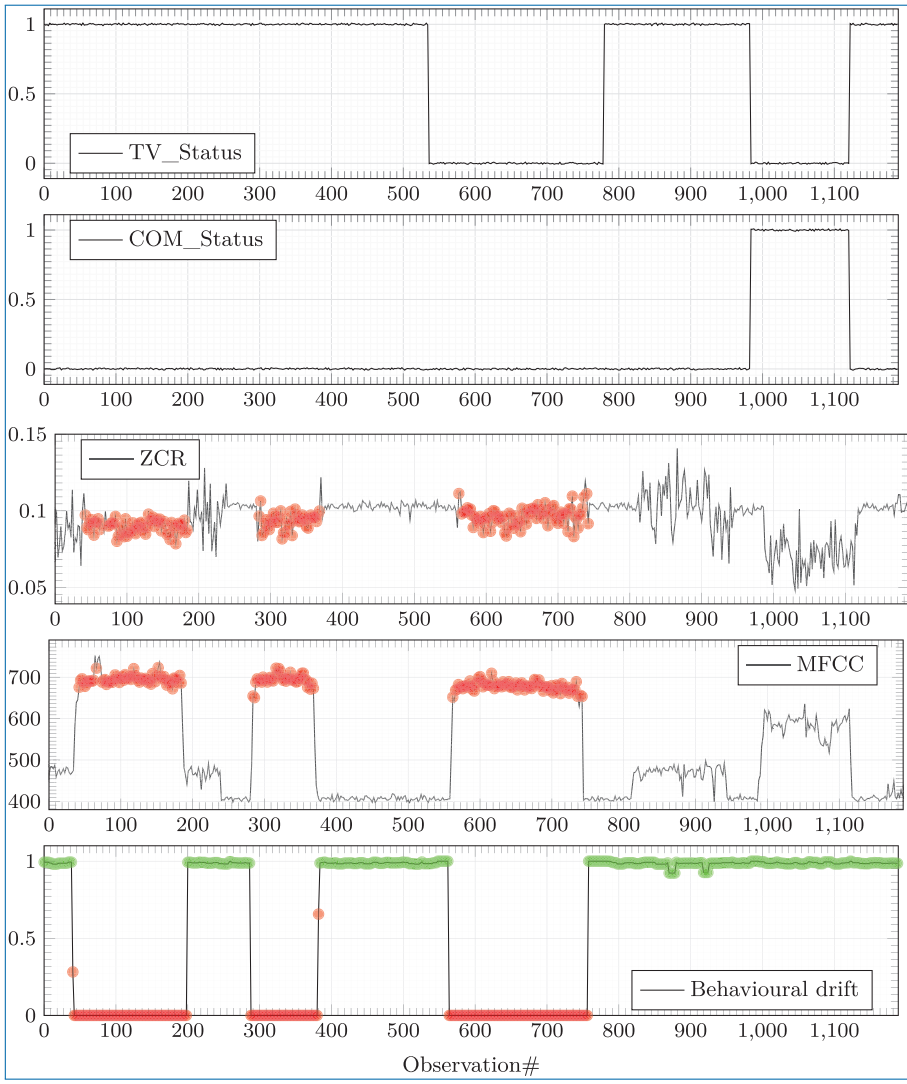

Figure 5.16. Observations from the field coloured in red denote the presence of the smart vacuum cleaner not anticipated/foreseen in the model of the legitimate behaviour (leading behavioural drifts to occur).

5.4.4 Software Development (Devs, Cycle 2)

The behavioural drift reported at run-time suggests that the model of the physical environment is incomplete. Indeed, the indirect conflict on the sound physical property should have been identified during the first phase of development. A new development cycle is therefore necessary to correct the model of the physical environment. The updated model is depicted in Fig. 5.18; an indirect

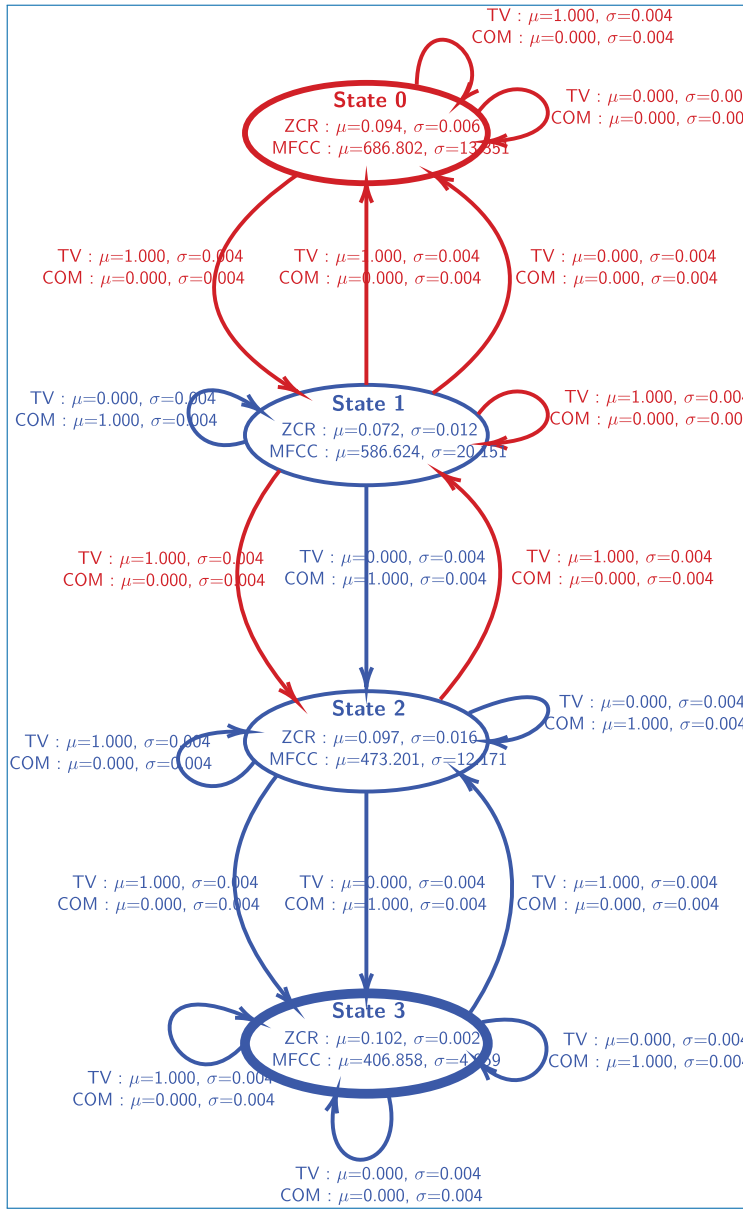

Figure 5.17. Dissimilarity graph. Nodes and edges coloured in blue correspond to the legitimate behaviour, those in red correspond to unexpected behaviour resulting from the appearance of the smart vacuum cleaner in the living-room.

actuation conflict is now detected between the *App_RC_TV*, *App_Phone_TV* and the *App_Vaccum_Cleaner* applications and a dummy ACM has been instantiated.

The management of this indirect actuation conflict implies semantic concerns that cannot be managed through generic off-the-shelf ACMs. A custom ACM

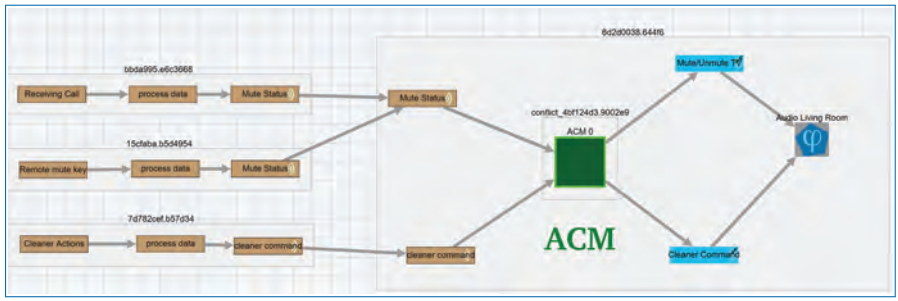

Figure 5.18. Excerpt of indirect conflict detection and management.

has to be developed. This task relies on "ECA" rules that define the ACM logic, further transformed into an FSM. More specifically, designers have to describe a control strategy that prevent the TV and the smart vacuum cleaner robot to produce sound simultaneously. The custom ACM thus receives commands from *App_RC_TV*), *App_Phone_TV* and *App_Vaccum_Cleaner*. On the basis of these input, the ACM control logic defines a strategy resulting in sending commands to the TV and the vacuum cleaner such that it is stopped while a communication is in progress or while inhabitants watch TV. The ACM control strategy defines some prioritization whose semantic has to be defined by designers.

Along with the logical strategy of the custom ACM, some logical and temporal properties to be formally verified have to be specified. To this end, ECA rules are extended into ECA+ rules (more details about this language are given in [7]) to provide the model with the properties to be verified by state-of-the-art MDE verification tools.

Logical properties are verified through NuSMV [8], a state of the art model checker. Model checkers are meant to ensure that logical properties are always verified, regardless of the state sequence being executed. Two main types of logical properties are formally verified: *safety* (i.e., something bad will never occur) and *Liveness* (i.e., something good will eventually happen). In the context of ACM design, safety properties ensure that conflicting states will never occur (e.g., the vacuum cleaner will never operate while a communication is in progress).

Temporal properties are verified through DEVS formalism. FSMs are defined by two functions: (i) the state-transition function computes the new state given the previous state $x_{(k-1)}$ at time $k-1$ and the current input $\vec{u}_{(k)}$ at time k, (ii) the output function computes the outputs $\vec{y}_{(k)}$ that solely depend on the state $x_{(k)}$ at time k. DEVS formalism allows an FSM to be encapsulated into DEVS atomic model coupled with a synchronizer managing asynchronous timings on the inputs of the FSM. The ECA+ language provides a syntax to specify asynchronous timings management strategies for the synchronizer. Temporal properties can then be

verified for each strategy within a DEVS simulation environment. It is worth noting that asynchronous timing strategies are meant to reproduce the asynchronous timings that govern the different hardware platforms ACMs are supposed to be deployed to, at the edge of the IoT infrastructure.

Once ACMs temporal properties have been verified in DEVS simulated operational contexts, associated DEVS Atomic models (embedding the synchronizer and the logical behaviour) can be directly used for implementation. Indeed, DEVS Atomic models can be translated into high level programming languages (C, C++, C#, Node.js, etc.), embedding a lightweight execution engine (DEVS kernel). This makes ACMs completely portable on the lightweight hardware platforms available at the edge of the IoT infrastructure.

Here again, the WIMAC model modified with the custom ACM is sent to GeneSIS for the deployment on the platforms, thus completing the second development cycle.

5.4.5 System Operations (Ops, Cycle 2)

Finally, SIS is deployed and the physical effects it produces in the physical environment are observed. The model of the legitimate behaviour is modified to take into account the autonomous vacuum cleaner, as learned during the first cycle. No behavioural drift is reported anymore. However, as the physical environment is complex, there are many reasons why the behaviour of the SIS may drift again and trigger a new development cycle:

- the introduction of a new software component that drives an actuator not correctly defined in the model of the physical environment,
- the introduction of a new device producing physical effects in the environment (as the vacuum cleaner in the use-case),
- unexpected changes in the physical environment in which the SIS operate (e.g., a tree growing in front of a window is likely to have an impact on the luminosity of the room in the long term).

5.5 Conclusion and Future Works

This chapter has introduced two innovative toolsets designed to enrich the DevOps eco-system and meant to address an issue that has been poorly addressed, to date, in the development of trustworthy SIS: the management of applications that can interact with their physical environment through actuators.

The first toolset, called "Actuation Conflict Management" toolset (ACM), is to secure the design of these applications at Devs time. The objective is to identify

and resolve, through local structural transformations of the SIS, the actuation conflicts that might arise as a result of the actuation effects produced in the physical environment a priori known and limited to its model. While such precautions at Devs time are necessary, they are not sufficient to guarantee that the SIS will always operate as expected in the physical environment. There is a risk of behavioural drift which is observed and quantified at Ops time, thanks to the second toolset called "Behavioural Drift Assessment & Analysis" toolset (BDA). This toolset makes it possible to assess and analyse the differences between the expected and observed effects of the SIS in real-world environment.

Beyond the interest of these toolsets as part of the DevOps methodology, their complementarity represents a major contribution in the realm of trustworthy SIS. Indeed, it is during consecutive DevOps cycles that the contribution of the couple BDA-ACM can be fully appreciated. The BDA toolset provides DevOps team with information on the observed behavioural drifts, thus motivating new development cycles which then results in changes to the model of the physical environment and/or corrections of the applications carried out in and with the ACM toolset. This complementarity is highlighted in this chapter throughout a smart-home use-case which involves two consecutive DevOps cycles to converge towards a SIS with satisfactory behaviour.

It is within the framework of this complementarity that two possible lines of work are foreseen. The first axis aims to reinforce the added-value of the information obtained from the BDA so as to accelerate the SIS/ACMs re-design cycle. Indeed, while the BDA toolset allows to obtain a model of the effects observed in the field and the symptoms of their dissimilarities with the legitimate ones, it is neither informative on the root-causes underlying these differences nor it provides insights to infer the corrections to be made to the model of the physical environment. The second axis is about leveraging behavioural drift measures as rewards towards self-adaptive ACMs automatically containing/mitigating behavioural drifts at run-time.

References

[1] A. Metzger (Ed.). *Software Continuum: Recommendations for ICT Work Programme 2018–2019*. NESSI White Paper, February, 2016.

[2] Abdullah Al Farooq *et al.* "Iotc 2: A formal method approach for detecting conflicts in large scale IoT systems". In: *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE. 2019, pp. 442–447.

[3] Robert C. Armstrong *et al.* "Survey of existing tools for formal verification". In: *SANDIA REPORT SAND2014-20533* (2014).

[4] Yoshua Bengio and Paolo Frasconi. "An input output hmm architecture". In: *Advances in neural information processing systems*, pp. 427–434, 1995.

[5] James Bruce. *Getting Started with OpenHAB Home Automation on Raspberry Pi*. 2015.

[6] Laurent Capocchi *et al.* "DEVSimPy: A collaborative python software for modeling and simulation of DEVS systems". In: *2011 IEEE 20th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*. IEEE. 2011, pp. 170–175.

[7] Franck Chauvel *et al. Risk-driven Continuous delivery of Trustworthy Smart IoT Systems*. 2020. URL: https://enact-project.eu/deliverables/D2.3.pdf.

[8] Alessandro Cimatti *et al.* "Nusmv 2: An opensource tool for symbolic model checking". In: *International Conference on Computer Aided Verification*. Springer. 2002, pp. 359–364.

[9] European Commission. *Horizon 2020 Work Programme 2016–2017 – Cross-cutting activities (Focus Areas), Call IoT-03-2017 : R&I on IoT integration and platforms*. Oct. 2015. URL: %7Bhttp://ec.europa.eu/newsroom/dae/document.cfm?%5C&doc%5C_id=11867%7D.

[10] Muhammad Fahim and Alberto Sillitti. "Anomaly detection, analysis and prediction techniques in IoT environment: A systematic literature review". In: *IEEE Access* 7 (2019), pp. 81664–81681.

[11] G. Rocher, J.-Y. Tigli, S. Lavirotte and N. Le Thanh. *Effectiveness assessment of Cyber-Physical Systems*. 2020.

[12] Thibaut Gonnin *et al.* "Actuation Conflict Management Enabler for DevOps in IoT". In: *10th International Conference on the Internet of Things Companion*, IoT '20 Companion. Malmö, Sweden: Association for Computing Machinery, 2020. ISBN: 9781450388207. DOI: 10.1145/3423423.3423474. URL: https://doi.org/10.1145/3423423.3423474.

[13] Edward R. Griffor *et al.* "Framework for cyber-physical systems: Volume 2, working group reports". In: (2017).

[14] Andrew Guillory *et al.* "Learning executable agent behaviors from observation". In: *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*. 2006, pp. 795–797.

[15] Hani Hagras. "Toward human-understandable, explainable AI". In: *Computer* 51.9 (2018), pp. 28–36.

[16] Andrew KS Jardine, Daming Lin, and Dragan Banjevic. "A review on machinery diagnostics and prognostics implementing condition-based maintenance". In: *Mechanical systems and signal processing* 20.7. (2006), pp. 1483–1510.

[17] Mengda Jia *et al.* "Adopting Internet of Things for the development of smart buildings: A review of enabling technologies and applications". In: *Automation in Construction* 101 (2019) 111–126. ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2019.01.023. URL: http://www.sciencedirect.com/science/article/pii/S0926580518307064.

[18] Chris Jones. "Attributed graphs, graph-grammars, and structured modeling". In: *Annals of Operations Research* 38.1 (1992), pp. 281–324.

[19] Stéphane Lavirotte *et al.* "IoT-based Systems Actuation Conflicts Management Towards DevOps: A Systematic Mapping Study". In: *IoTBDS*. 2020, pp. 227–234.

[20] Jean Louis Le Moigne. *La modélisation des systèmes complexes*. Bordas Paris, 1990.

[21] Renju Liu *et al.* "RemedIoT: Remedial actions for Internet-of-Things conflicts". In: *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*. 2019, pp. 101–110.

[22] Meiyi Ma *et al.* "Detection of runtime conflicts among services in smart cities". In: *2016 IEEE International Conference on Smart Computing (SMARTCOMP)*. IEEE. 2016, pp. 1–10.

[23] Mohammad Nasar and Mohammad Abu Kausar. "Suitability of influxdb database for iot applications". In: *International Journal of Innovative Technology and Exploring Engineering* 8.10 (2019), pp. 1850–1857.

[24] Dang Tu Nguyen *et al.* "IotSan: Fortifying the safety of IoT systems". In: *Proceedings of the 14th International Conference on emerging Networking EXperiments and Technologies*. 2018, pp. 191–203.

[25] Daniella Niyonkuru and Gabriel Wainer. "Towards a DEVS-based operating system". In: *Proceedings of the 3rd ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*. 2015, pp. 101–112.

[26] Lawrence R. Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286.

[27] Gerald Rocher *et al.* "An Actuation Conflicts Management Flow for Smart IoT-based Systems". In: *2020 7th International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*. Paris, France: IEEE, Dec. 2020, pp. 1–8. ISBN: 978-0-7381-2460-5. DOI: 10.1109/IOTSMS52051.2020.9340196. URL: https://ieeexplore.ieee.org/document/9340196/ (visited on 02/05/2021).

[28] Gérald Rocher *et al.* "A Possibilistic I/O Hidden Semi-Markov Model for Assessing Cyber-Physical Systems Effectiveness". In: *2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE. 2018, pp. 1–9.

[29] Gérald Rocher *et al.* "An IOHMM-Based Framework to Investigate Drift in Effectiveness of IoT-Based Systems". In: *Sensors* 21.2 (2021), p. 527.

[30] Gérald Rocher *et al.* "Overview and Challenges of Ambient Systems, Towards a Constructivist Approach to their Modelling". In: *arXiv preprint arXiv:2001.09770* (2020).

[31] Claude Rochet. "Public Management as a moral science". Habilitation à diriger des recherches, Université de droit, d'économie et des sciences – Aix-Marseille III, Dec. 2007.

[32] Alexander Schliep. "A bayesian approach to learning hidden markov model topology with applications to biological sequence analysis". PhD thesis, Universität zu Köln, 2001.

[33] Trusit Shah *et al.* "Conflict detection in rule based IoT systems". In: *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE. 2019, pp. 0276–0284.

[34] Antero Taivalsaari and Tommi Mikkonen. "A roadmap to the programmable world: software challenges in the IoT era". In: *IEEE Software* 34.1 (2017), pp. 72–80.

[35] Arjan J. Van Der Schaft and Johannes Maria Schumacher. *An introduction to hybrid dynamical systems*. Vol. 251. Springer London, 2000.

[36] Yentl Van Tendeloo and Hans Vangheluwe. "An evaluation of DEVS simulation tools". In: *Simulation* 93.2 (2017), pp. 103–121.

[37] Philippe Weber and Christophe Simon. *Benefits of Bayesian network models*. John Wiley & Sons, 2016.

[38] Muneer Bani Yassein, Wail Mardini, and Ashwaq Khalil. "Smart homes automation using Z-wave protocol". In: *2016 International Conference on Engineering & MIS (ICEMIS)*. IEEE. 2016, pp. 1–6.

[39] Bernard P Zeigler, Alexandre Muzy, and Ernesto Kofman. *Theory of modeling and simulation: discrete event & iterative system computational foundations*. Academic press, 2018.

[40] Xiangrui Zeng and Junmin Wang. "A stochastic driver pedal behavior model incorporating road information". In: *IEEE Transactions on Human-Machine Systems* 47.5 (2017), pp. 614–624.

[41] Tao Zheng and Gabriel A Wainer. "Implementing finite state machines using the CD++ toolkit". In: *Proceedings of the SCS Summer Computer Simulation Conference, 2003. atomic model, 1 CD++, 1 coupled model, 1 DEVS, 1 DEVS Graph, 1 discrete-event modeling*. Citeseer, 2003.

DOI: 10.1561/9781680838251.ch6

Chapter 6

Online Reinforcement Learning for Self-Adaptive Smart IoT Systems

By Alexander Palm, Felix Feit and Andreas Metzger

6.1 Introduction

In this chapter we explain how Reinforcement Learning (RL) techniques can be leveraged to improve the way self-adaptive smart IoT systems (SIS) adapt at run-time.

The concept of self-adaptation facilitates developing software systems that are capable of maintaining their quality requirements even if the systems' environment changes dynamically [3, 18]. Self-adaptation thereby helps developing systems that can operate in a resilient way at run-time. To this end, a self-adaptive software system (such as a self-adaptive SIS) can modify its own structure, parameters and behavior at run-time based on its perception of the environment, of itself and of the fulfilment of its requirements. An example is a self-tuning thermostat for a Heating, Ventilation and Air Conditioning (HVAC) system. Based on its perception of the outdoor and indoor temperature it can control the strength of its heating and cooling devices in order to proactively reach a set point temperature to maximize user comfort. On the other hand it can learn to reduce energy by reducing heating and/or cooling strength when no user is present in the room.

To develop a self-adaptive SIS, software engineers have to develop self-adaptation logic that encodes when and how the system should adapt itself. Software engineers, for instance, may specify event-condition-action rules that determine which adaptation action is executed in response to a given environment change. Developing self-adaptation logic requires an intricate understanding of the software system and its environment, and how adaptations impact on system quality [6, 7]. Among other concerns, it requires anticipating the potential environment changes the system may encounter at run-time to determine how the system should adapt itself in response to these environment changes.

However, anticipating all potential environment changes at design time is in most cases infeasible due to design time uncertainty [7, 17]. In addition, while the principal effects of an adaptation on the system may be known, accurately anticipating the effect of a concrete adaptation is difficult; e.g., due to simplifying assumptions made during design time [7, 10]. One emerging way to address design time uncertainty is to employ online RL [1, 2, 4, 8, 12, 14, 22–24].

Online RL can learn the effectiveness of adaptation actions through interactions with the system's environment. This means that instead of software system engineers having to manually develop the self-adaptation logic, the system automatically learns the self-adaptation logic via machine learning at run-time. The software system engineer expresses the learning problem in a declarative fashion, in terms of the learning goals the system should achieve. Online RL thereby automates the manual engineering task of developing the self-adaptation logic.

Therefore, the remainder of this chapter is structured as follows: Section 6.2 motivates the application of RL in the realm of SIS by briefly introducing the topics of self-adaptive software systems (SASS) and RL. Section 6.3 combines the aforementioned topics and introduces the concept of policy-based RL and explains how it helps to address large continuous state spaces which is a main shortcoming of state-of-the-art approaches leveraging RL at run-time. Section 6.4 shows our experimental results after using our policy-based RL approach for the realization of a self-tuning thermostat in the smart building domain. Section 6.5 exemplifies how the reward function of a RL problem can be decomposed into several reward streams with different semantics to make decisions of an RL agent explainable. Finally, Section 6.7 concludes the chapter.

6.2 Fundamentals

In this section the fundamentals of self-adaptive software systems and Reinforcement learning are briefly introduced.

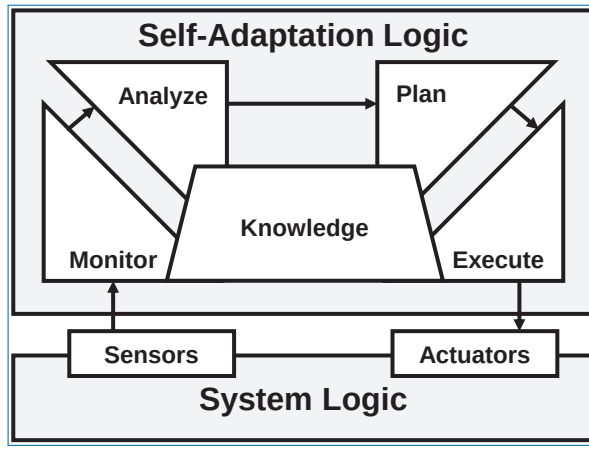

Figure 6.1. MAPE-K reference model for self-adaptive systems (based on [11]).

6.2.1 Self-adaptive Software Systems

A well-known reference model for self-adaptive systems is the MAPE-K model [11], which is depicted in Figure 6.1. Following this reference model, a self-adaptive software system can be logically structured into two main elements: the system logic (aka. the managed element) and the self-adaptation logic (the autonomic manager).

As shown in Figure 6.1, the self-adaptation logic can be further structured into four main conceptual activities that leverage a common knowledge base [9]. The knowledge base includes information about the managed system (e.g., encoded in the form of models at run-time), its environment, and its adaptation goals and adaptation policies (e.g., expressed as rules). The four activities are concerned with monitoring the system logic and the system's environment via sensors, analysing the monitoring data to determine the need for an adaptation, planning adaptation actions, and executing these adaptation actions via actuators, thereby modifying the system logic at run-time.

6.2.2 Reinforcement Learning

RL aims to learn suitable actions via an agent's interactions with its environment [20] as depicted in Figure 6.2. At a given time step t, the agent selects an action a (from its adaptation space) to be executed in environment state s. As a result, the environment transitions to s' at time step $t + 1$ and the agent receives a reward r for executing the action. The reward r together with the information about the next state s' are used to update the knowledge of the agent. The goal of RL is to optimize cumulative rewards. One fundamental problem in RL is the trade-off that must be made between *exploitation* (using current knowledge) and *exploitation*

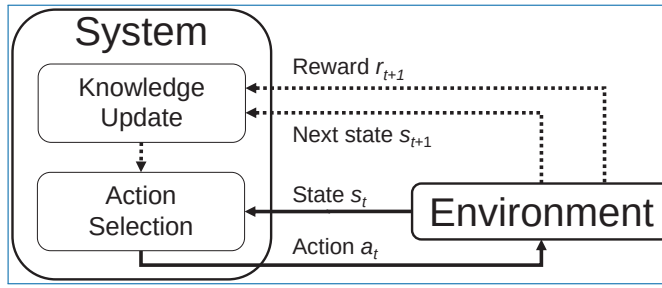

Figure 6.2. Schematic illustration of agent-environment interaction in RL (based on [20]).

(gathering new knowledge). For a self-adaptive service, "agent" refers to the self-adaptation logic of the service and "action" refers to an adaptation action [16].

6.3 OLE: Policy-based Online Reinforcement Learning

This sections introduces our online RL approach. In Section 6.3.1, we provide an overview, the conceptual ideas behind the approach how it differs from the state of the art. In Section 6.3.2, we explain how we prototypically realized the approach.

6.3.1 Overview of Our Approach

The innovative concept underlying the ENACT Online Learning Enabler (OLE) is that we use a fundamentally different type of reinforcement learning than what has been used in the state of the art. While the state of the art used value-based RL, we use policy-based RL. The main idea behind policy-based RL is to directly use and optimize a parametrized stochastic *action selection policy* [15, 21]. The action selection policy maps states to a probability distribution over the action space (i.e., set of possible actions). This means that actions are selected by sampling from this probability distribution. A *learning cycle* consists of a predefined number of n time steps. At the end of each learning cycle, the trajectory (comprising the selected n actions, states and rewards) are used for a policy update. During a policy update, the policy parameters are perturbed based on the rewards received, such that the resulting probability distribution is shifted towards a direction which increases the likelihood of selecting actions which led to a higher cumulative reward.

Figure 6.3 depicts the conceptual architecture of our approach, showing how the elements of policy-based RL are integrated into the MAPE-K loop. The dark-gray area indicates where the *action selection* of RL takes the place of the *analyze* and *plan* activities of MAPE-K. The learned *stochastic policy* takes the role of the self-adaptive system's *knowledge* base.

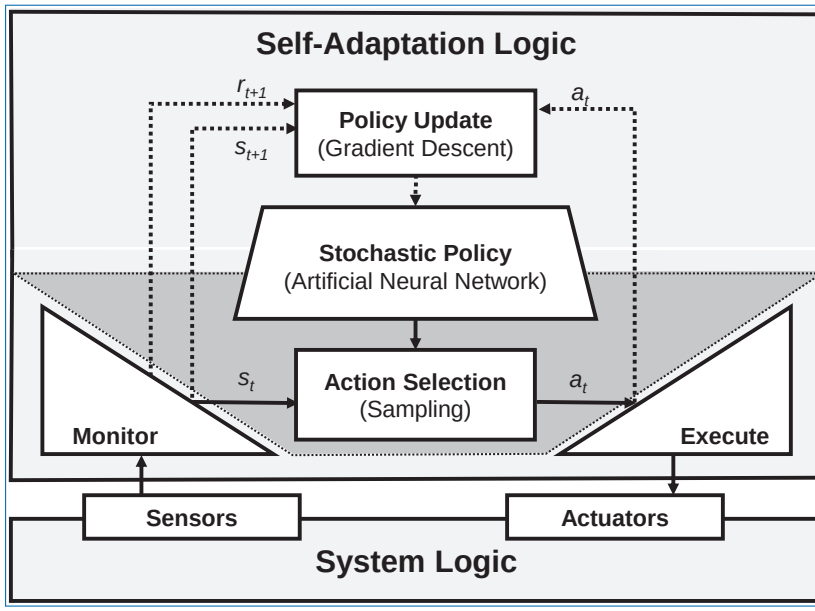

Figure 6.3. Conceptual architecture of policy-based approach.

At run-time the policy is used by the self-adaptation logic to select (via sampling) an adaptation action a_t based on the current state s_t determined by the *monitoring* activity. Action selection determines whether there is a need for an adaptation (given the current state) and plans (i.e., selects) the respective adaptation action to *execute*. In our approach, state s_t may be determined using observations of both the system's environment and the system logic itself. This differs from the basic RL model, where only observations from the system's environment are considered to determine the state s_t. A *policy update* utilizes the trajectory of actions a_t, states s_{t+1}, and rewards r_{t+1} to update the policy. In our approach, policy updates are performed via so-called policy gradient methods [20, 21], because the policy is represented as an artificial neural network. Policy gradient methods update the policy according to the gradient of a given objective function, such as the average reward per learning cycle to give a simple example. In our architecture, rewards are computed by the monitoring activity, as this activity has access to all sensor information collected from the system and its environment.

As mentioned above, the learning problem is stated in a declarative fashion. Typically, it can be formalized as a Markov decision process $MDP = (S, A, T, R)$, with

- S being the state space composed of a set of environment and system states $s \in S$ observable by monitoring via the system logic's sensors (e.g., system workload and performance of the system),

- A being the action space with a set of possible adaptation actions $a \in A$, i.e., possible ways the system may be adapted using the system logic's actuators (e.g., turning off or on different system features),
- $T : S \times A \times S \rightarrow [0, 1]$ being the transition probability among states with $T(s_t, a_t, s_{t+1}) = \Pr(s_{t+1}|s_t, a_t)$, which gives the probability that an adaptation action a_t in state s_t will lead to a state s_{t+1}, and
- $R : S \rightarrow \text{IR}$, being a reward function which specifies the numerical reward the system receives in state s_t. The reward function expresses the learning goal to achieve, which in our case expresses maintaining the quality requirements of the system (e.g., performance should not fall below a given threshold).

Policy-based reinforcement learning finds a solution to the MDP in the form of a parametrized stochastic policy $\pi_\theta : S \times A \rightarrow [0, 1]$, giving the probability of taking adaptation action a in state s, i.e., $\pi_\theta(s, a) = \Pr(a|s)$. The policy's parameters (weights of the artificial neural network) are given as a vector $\theta \in \text{IR}^d$.

Regarding design time uncertainty, we assume that we know A, S, and R, but do not know T. More precisely, even if we do not know the exact states and thus state space S, we know the state variables. As an example, even if we do not know exact workloads of a web application (and maybe not even the maximum workload), we can express a state variable workload $w \in \text{IN}^+$. We assume that we do not know T due to design time uncertainty about how adaptation impacts on system quality. As an example, we may not have an exact understanding of how different configurations of the system perform under different workloads.

6.3.2 Prototypical Realization

To select a concrete policy-based RL algorithm for the implementation of our approach, we took into account two main considerations. First, as we assume we do not know the transition function T, we need to use a model-free variant of policy-based RL. Second, to facilitate online learning, we need an algorithm that continuously updates the policy without waiting for a final outcome, i.e., without waiting for reaching a terminal state. Actor-critic algorithms are a model-free variant of policy-based RL algorithms that use bootstrapping (i.e., knowledge is updated continuously without waiting for a final outcome). We use proximal policy optimization (PPO [19]) as a state-of-the-art actor-critic algorithm. PPO is rather robust for what concerns hyper-parameter settings. Thereby, we avoid extensive hyper-parameter tuning compared to other actor-critic algorithms. In addition, PPO avoids too large policy updates by using a so called clipping function. A too large policy update may mean that RL misses the global optimum and remains stuck in a local optimum. To represent the actor and critic models of PPO, we used

multi-layer perceptrons with two hidden layers of 64 neurons each (neurons in the input and output layers depended on the respective number of action and state variables).

6.4 Validation in the Smart Building Domain

To show the applicability of our policy-based RL approach on SIS we performed a series of experimental validations in the smart building domain (cf. Section 10.5). Therefore, the experimental setup is summarized in Section 6.4.1 before the underlying RL problem is formalized as a MDP in Section 6.4.2. Finally, Section 6.4.3 shows the results of our experiments.

6.4.1 Experimental Setup

In the according use case scenario we show that it is possible to learn a control strategy for a simulated HVAC system by means of policy-based RL. The simulated HVAC system is based on the ground floor of the KUBIK building (cf. Section 10.5) and comprises six multi-sensors providing information about user presence and temperature, as well as 5 fan coils whose capacity is accumulated to treat them as one single fan coil. An excerpt of the KUBIK specification showing the room used for the simulation is depicted in Figure 6.4.

The learned control strategy thereby should control the HVAC system in such a way that thermal comfort is achieved whenever the controlled room is occupied and energy consumption is minimized otherwise. The run-time of each experiment corresponds to one year of simulation, while one time-step of an experiment corresponds to one minute (i.e. total time-steps of one experiment: 524000). After several iterations of code improvements of the simulation we were able to simulate

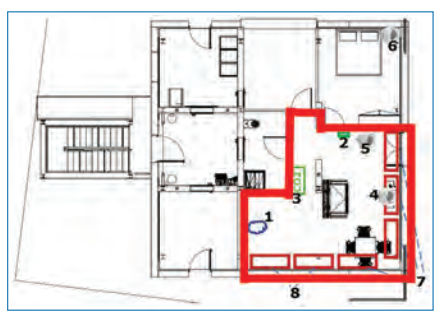

Figure 6.4. Excerpt of the KUBIK specification showing the room used for the simulation.

between 150 and 1000 time-steps per seconds depending on the hyperparameters (especially the size of the neural net) of the underlying algorithm.

As a baseline we implemented a simple on/off-controller, that heats or cools the room whenever the indoor temperature is not close enough to the user set-point and a user is present. If no user is present the thermostat controller remains inactive.

6.4.2 Problem Formalization as MDP

For the evaluation, in a first step we formulated the problem of learning an HVAC control strategy as a RL problem. The underlying Markov Decision Process (MDP) is thereby specified as follows:

State-space: The main state variables are stemming from the from simulation variables (e.g. indoor temperature). Furthermore, we created some crafted features resulting in variables relying on main state variables (e.g. deviation from setpoint). The variable predicted occupancy returns the probability that the room gets occupied within the next 30 min. This variable is computed based on the underlying occupancy pattern, as we assume that such variables might exist in modern HVAC systems (cf. [5]).

Action-space: As the main task of the control strategy is to properly control the HVAC device, the action-space comprises 7 discrete actions, where only one action could be selected at a time. The actions correspond to the different modes of the fan coil for heating and cooling, as well as a turning he device off. The different modes relate to different fan speeds resulting in different air flow rates and different temperatures concerning the integrated fluids for heating or cooling respectively.

Transition-dynamics: The transitions between the different states (depending on the selected action) are computed by the simulation according to the underlying thermal equations. As we employ model-free RL algorithms, the control strategy does not have access to the environmental model resulting from the simulation. It is learning through pure interaction with raw experience. The simulation could be treated as a real environment, with the main difference that with a real environment the run-time of an experiment would be much longer.

Reward: The main part of the MDP driving the algorithm in a direction to learn the right control strategy is the reward function. We defined the reward in such a way, that the algorithm gets a positive feedback whenever its control strategy keeps the indoor temperature close (i.e. within bounds of 1°C) to the user setpoint when a user is present and penalized otherwise (i.e. negative feedback corresponding to the deviation from the setpoint). As the control strategy should minimize energy consumption, the algorithm gets penalized according to the strength of the current action, whenever it performs an action other than turning the HVAC system off, if no user is present in the room (based on the simulated occupancy).

6.4.3 Results

We evaluate the results from two different perspectives: The domain perspective and the Reinforcement Learning perspective. For the domain perspective, we plotted the outdoor temperature curve and the indoor temperature curve showing the moving average of both variables. For the moving average we averaged the last 128 values to reduce the noise inside the diagram and improve the interpretability of the results. Furthermore, we plotted the average indoor temperature per occupancy phase. This gives us the option to see whether the learned control strategy is able to avoid heating or cooling actions in occupancy phases where no user is present and to keep the indoor temperature close to the user set-point whenever a user is present. Apart from this domain-related metric, we used the moving average reward of the last 10000 time-steps as a metric to evaluate the learning process from a RL-perspective. It is important to note that the values of the average reward are scaled to fit into the temperature diagram. The scaling factor is neglectable, as it is only the evolution of the reward curve that is important in this case. The exact evolution of the cumulative reward is shown in different plots to address solely the Reinforcement Learning perspective, when we compare the results of the RL approach to the baseline approach.

Figure 6.5 shows how the indoor temperature evolves according to the control strategy resulting from the baseline thermostat. As can be seen during the first

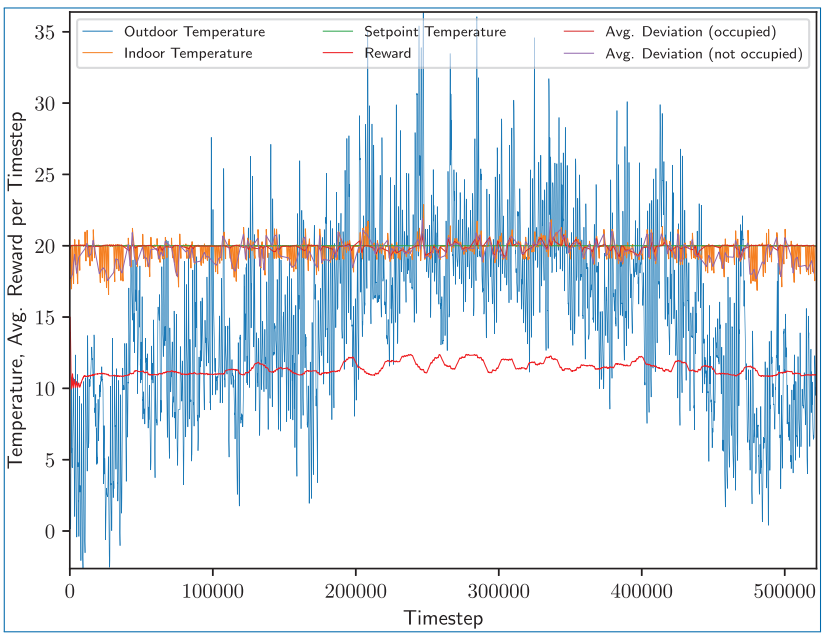

Figure 6.5. Temperature evolution of baseline thermostat.

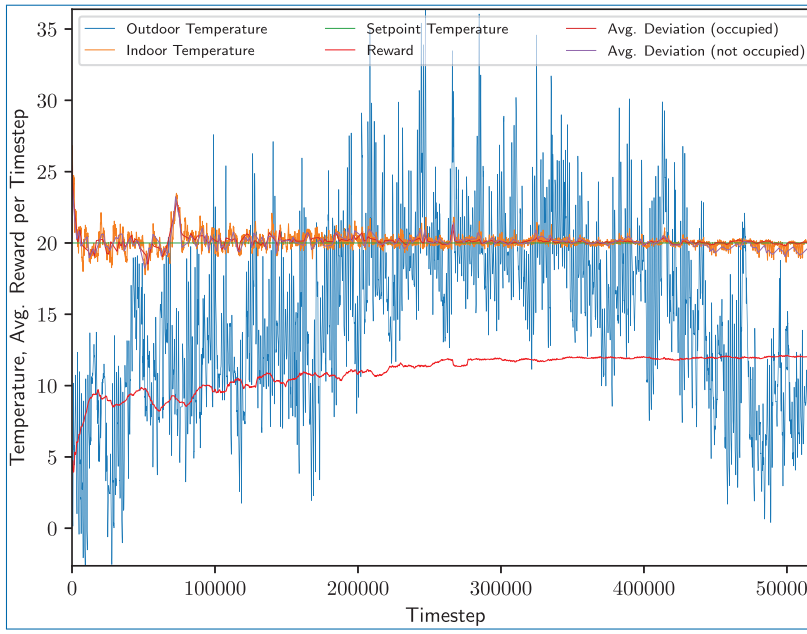

Figure 6.6. Temperature evolution of RL approach.

150.000 time steps, the indoor temperature drops if no user is present resulting from the heating device being turned off while the outdoor temperature is below the set point temperature. As the outdoor temperature becomes warmer (from time step 150.000 onwards) the indoor temperature is higher than the set point temperature (if not user is present) as the cooling device is turned off accordingly. In phases where a user is present the indoor temperature is kept around the set point temperature. However, from a RL perspective this leads to a non-optimal reward, because the thermostat controller is purely reactive and for every time step the indoor temperature is too far from the set point temperature this leads to a non-optimal reward.

In contrast, Fig. 6.6 shows how the indoor temperature evolves to a thermostat controller based on a policy-based RL approach. After a learning phase (until time step 260.000) the RL approach is able to keep the indoor temperature around the set point temperature if a user is present and reduce energy consumption by turning off the heating or cooling device otherwise. Especially during the end of the experiment, after learning has been converged and the approach can reuse its knowledge about low outdoor temperatures (from time step 450.000 onward), it can be seen that the same spikes can be observed as in 6.5. However, the drops in the indoor temperature are not as big as with the baseline approach and the reward is slightly higher.

Figure 6.7 shows the results from the RL perspective visualizing the cumulative reward evolution. The baseline approach outperforms the RL approach in terms of

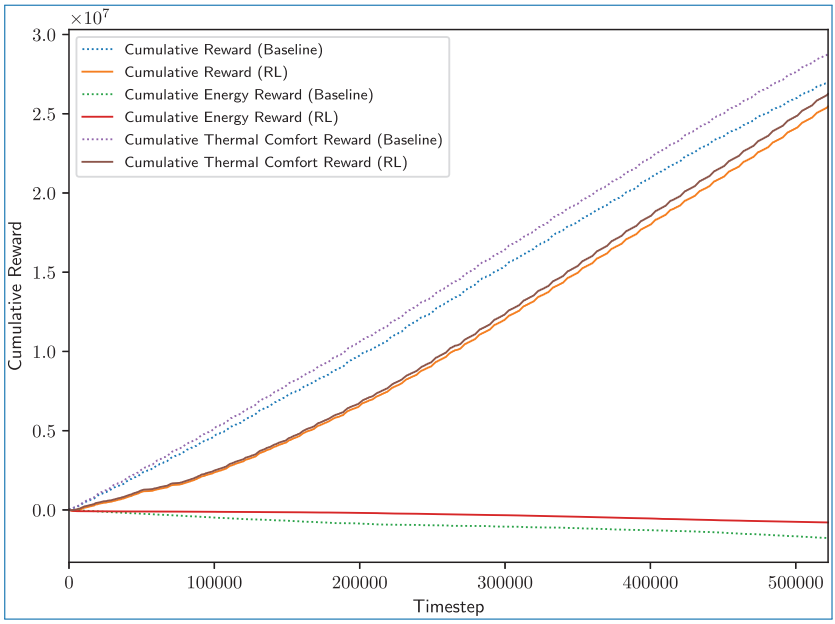

Figure 6.7. Cumulative reward (baseline vs. RL approach).

cumulative reward, resulting from the poor initial performance during the learning phase. This is due to the fact that the RL approach has no initial knowledge about a goal-directed behavior. However, after learning has converged (about time step 260.000) the increase in reward becomes more steep than that of the baseline approach, which can be seen from time step 400.000 onward.

After having shown that the RL approach is able to outperform the baseline approach after its learning process has converged, we did further investigate how it may perform if its knowledge has been initialized a priori. To do this we did set up a slightly modified version of the HVAC simulation, with the thermal dynamics being simplified (and not following complex thermal equations). To perform some kind of pretraining we let the RL approach learn with this simplified environment for the same amount of time steps as in the online experiment.

As it can be seen in Fig. 6.8 the pretrained version of the RL approach is able to adapt its control strategy to the real thermal dynamics pretty fast and after around 50.000 time steps the reward curve converges. This results from a goal-directed behavior with the pretrained RL approach being able to keep the indoor temperature around the set point temperature whenever a person is present and reduce energy consumption otherwise. The reason for the slightly better reward compared to the baseline approach can be seen in Fig. 6.9. The RL approach learns to avoid the indoor temperature moving too far from the set point temperature, to be able to reach the set point temperature within fewer time steps than the baseline approach.

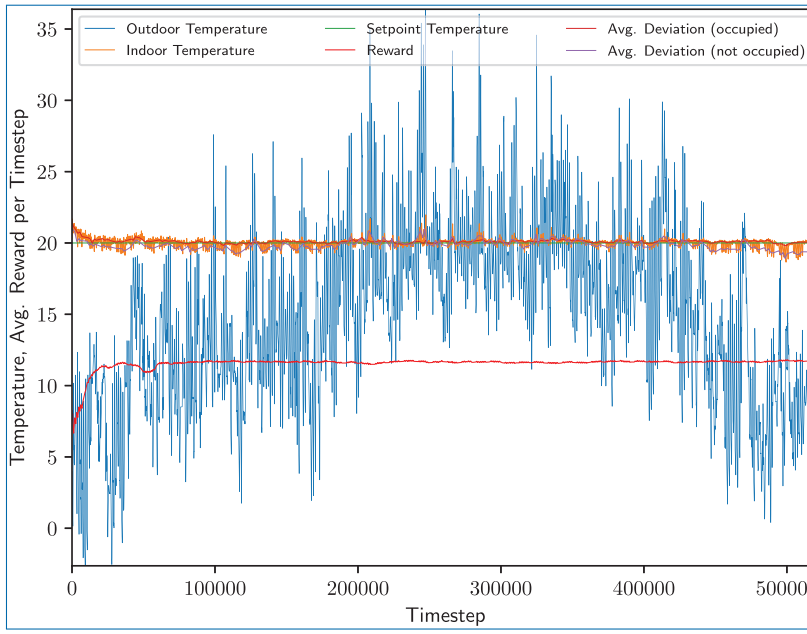

Figure 6.8. Temperature evolution of pretrained RL approach.

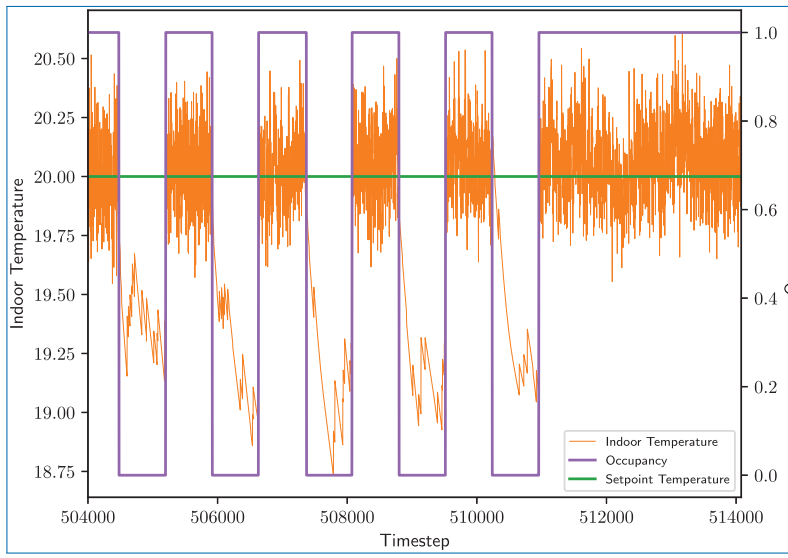

Figure 6.9. Temperature evolution of baseline thermostat.

This is done by proactively heating or cooling resulting in small penalty for consuming energy while the room is not occupied. However, this penalty is rather small to the penalty that would be received during a time step (with the room being occupied) where the indoor temperature is not around the desired set point temperature.

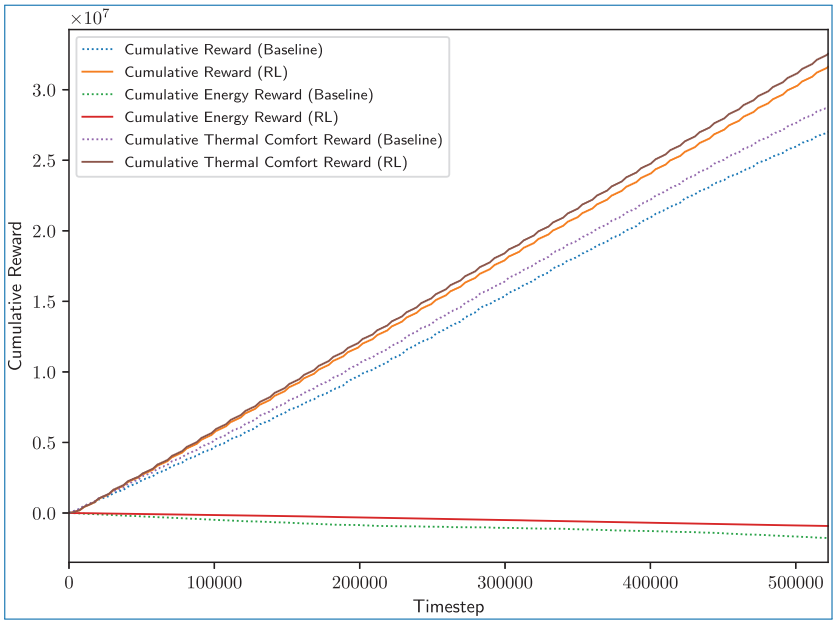

Figure 6.10. Cumulative reward (baseline vs. pretrained RL approach).

Finally, when having a look at the evolution of the cumulative reward as depicted in Fig. 6.10, the pretrained RL approach clearly outperforms the baseline approach.

6.5 Explaining Adaption Decisions via Reward Decomposition

Reward Decomposition is a method in which several value-based RL agents are trained in parallel on different aspects of an environment. At each time step the knowledge of the agents is aggregated to provide a global decision. To apply this method, it is necessary that the reward function of the examined environment can be decomposed into independent subfunctions. As a result, instead of returning a scalar value at each time step, the reward function returns a vector where each component reflects the reward of one subfunction or zero if there was no reward or punishment at the associated time step.

For each component of the reward vector an independent subagent is trained, which receives the global state as observation and as reward only one component of the reward vector. In order to derive the action of the overall agent, at each time step the action values of all subagents are summed up element-wise and on the basis of these aggregated action values an action is selected (e.g. by using epsilon greedy action selection).

This technique was applied to a simplified version of the HVAC environment (see Chapter 6.4) where the cooling capability was removed. In this environment the reward was split into the two subfunctions "thermal comfort" and "energy cost". The first of these functions always gives a negative value, i.e. a penalty if the person is present but the current temperature is not within a certain tolerance around the desired temperature. On the other hand, the second component "energy costs" contains a constant negative value if action "heating" was chosen in the last step. One value-based RL agent (e.g. DQN) is then trained for each of these two components of the reward vector. At each time step the action values of both agents are summed up for both actions "heating" and "not heating". The greater sum then marks the greedy action of the overall agent.

To generate explanations for actions, the action-values of the subagents can be put in relation to each other. For this purpose, the difference between the action-value of a particular action and the action-values of all alternative actions is calculated for each subagent. This is repeated for all actions and yields the relative importance per action per subagent. The higher the relative importance of an action, the more influence the subagent has on the selection of this action. By observing the behaviour and relative importance the reasoning of an agent can be deduced.

To further increase the explainability of the approach, the reward function can be broken down into situations instead of subfunctions. Each situation represents a set of special states of the environment. In addition, each situation receives a non-zero reward or punishment value that is always given when the agent is in that situation and zero otherwise. For example, the HVAC environment can be deconstructed into the following four situations:

- occupied and within the tolerance (reward: $+1$)
- occupied and out of tolerance (reward: -5)
- unoccupied and within the tolerance (reward: -1)
- unoccupied and out of tolerance (reward: $+0.1$)

For each of these situations, as before, a separate agent is trained and the relative importance is calculated. If the importance is then set in relation to the sum of the relative importance of all actions, the relative importance of an action in relation to a specific situation can be calculated. These values can then be summed up to obtain the relative importance of an action across all situations. Using these two metrics, the following natural language string can be generated for each time step:

"With my current knowledge I am 97% sure that action 'not heating' is better. Arguments in favour of action 'not heating' are the prevention of situation 'occupied and out of tolerance' (84%), the occurrence of situation 'occupied and within the tolerance' (9%), and the prevention of situation 'unoccupied and within the

tolerance' (4%). An argument in favour of action 'heating' is the occurrence of situation 'unoccupied and out of tolerance' (2%)."

The first sentence of this explanatory string contains the relative importance of an action across all situations (97%) and the following sentences describe the relative importance of an action in relation to a specific situation (84%, 9%, 4%, and 2%).

6.6 Synergies with Behavioural Drift Analysis

The main goal of our Online Reinforcement Learning approach is to learn an optimal control strategy for a MDP. Despite of evaluating the learned control strategy from a RL perspective, it needs to be evaluated from a domain perspective as well. The latter can also be used to guide the engineering of the reward function. As the behavioral drift analysis is based on a model capturing the desired control strategy in an abstract way, the computed signal might be used for the evaluation of the actual control strategy. Unexpected changes in the behavioral drift signal can then be interpreted as an indicator for context changes that make it impossible for the learning system to behave according to the obtained model. To showcase this synergy we derived a model for BDA based on the desired behavior described in Section 6.4.2 (cf. Figure 6.11).

The behavioural model depicted in Figure 6.11 represents the expected indoor temperature which depends on whether or not a person is present in the room and this, independently of the underlying temperature management system. It relies

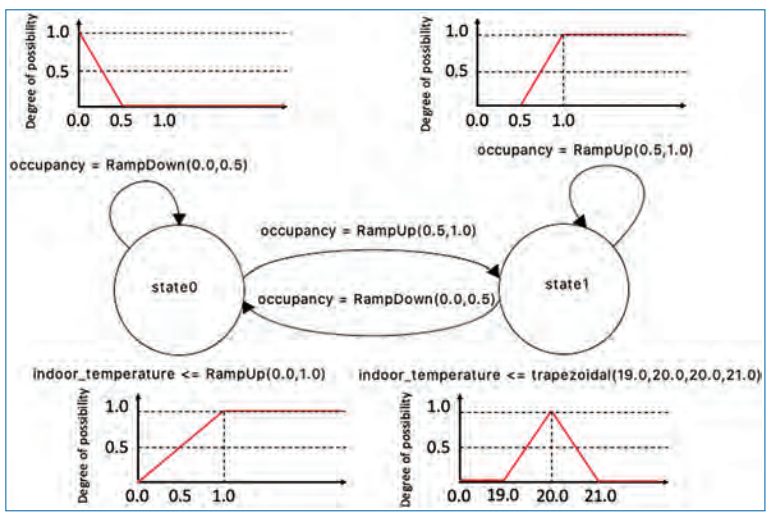

Figure 6.11. Possibilistic Input/Output Hidden Markov Model describing the expected indoor temperature depending on whether a person is present or no in a room.

on the possibility theory where distributions are defined as membership functions. The model defines the expected behaviour as follows: when a person is present in the room, the temperature must be equal to 20°C (state 1). When nobody is in the room, the temperature is expected to be greater than 1°C (state 0). In addition to fully accepted temperature values (where degree of possibility is equal to one), the model defines some tolerances. For instance, for the state 1, temperatures below 19.0°C and above 21°C are totally rejected (degree of possibility = 0) while temperatures in between 19.0°C and 21.0°C different from 20°C, while not being perfect, are not totally rejected (0 < degree of possibility <1). In conjunction with temperature and occupancy sensor values, this possibilistic Input/Output Hidden Markov Model (IOHMM) is used to compute the behavioural drift as the likelihood (possibility measure) of the observation sequences to have been generated by the model, i.e. the likelihood that the temperature is managed in such a way that it remains within the accepted boundaries defined by the model.

6.7 Conclusion and Outlook

We motivated the application of Reinforcement Learning as a means to enable a software system to adapt itself to changing context situations in the realm of SIS. Furthermore we introduced a concrete realization of an Online Learning approach which overcomes the main shortcomings of state-of-the-art approaches (e.g. ability to handle continuous parameters as actions and avoid manual fine-tuning of exploration). Our policy-based Reinforcement Learning approach for a self-adaptation logic has been validated in the smart building domain by applying it to an HVAC control problem. The experiment results have shown that our approach is able to outperform static thermostat implementations by dynamically learning to control the heating and cooling devices of a smart building. This has been achieved by finding a trade-off between the maximization of user comfort and minimization of energy consumption. Additionally, we introduced our conceptual work on the process of decomposing a reward function of a RL problem into several reward streams with different semantics to make decisions of an RL agent explainable and proposed how our Online Learning approach can be enriched by the concept of Behavioral Drift Analysis.

As future work, we envision extending our approach for online reinforcement learning for self-adaptive Smart IoT Systems along the following two main dimensions:

Better Pre-training As we demonstrated above, pre-training the reinforcement learning enabler may deliver better performance during operations. On the one hand, the initial performance (directly after deployment to run-time) can

be increased. On the other hand, the overall speed of learning and learning performance can be increased, in particular in real-world situations where rewards are sparse. Yet, such offline pre-training again faces the uncertainty issue when formulating the source learning task to be learned in the offline setting. It is not possible due to design time uncertainty that this source learning task faithfully captures the actual online setting. To capture the problem of uncertainty, existing solutions thus make certain assumptions about the system and its uncertainty in order to be able to perform the training in the offline setting. This is also what we did above, by taking certain assumptions about the building domain and even taking real, historic data into account. However, while this may mean that the reinforcement learning enabler learns a policy that solves this specific problem (i.e., under the given assumptions), the learned policy can be useless or may even perform worse than a policy only trained online when applied to the actual problem at run-time (which may violate these assumptions), even if it is relatively similar. One approach to this problem is to leverage the emerging concept of deep meta reinforcement learning.

Coping with large discrete action spaces Existing online reinforcement learning solutions for self-adaptive services propose randomly selecting adaptation actions for exploration he effectiveness of exploration therefore directly depends on the size of the adaptation space, because each adaptation action has an equal chance of being selected. Some reinforcement learning algorithms can cope with a large space of actions, but require that the space of actions is continuous in order to generalize over unseen actions. Self-adaptive Smart IoT Systems may have large, discrete adaptation spaces; *e.g.* if their adaptations entail reconfigurations of many system features or a large set of discrete parameters. In the presence of such large, discrete adaptation space, random exploration thus may lead to slow learning at run-time. One approach to this problem is to leverage the structure of the adaption space to better guide the exploration process. In related work, we have demonstrated that such improved exploration is possible for cloud services [13]. It thus can serve as a promising basis for applying to Smart IoT Systems.

References

[1] Mehdi Amoui *et al.* "Adaptive action selection in autonomic software using reinforcement learning". In: *Fourth International Conference on Autonomic and Autonomous Systems (ICAS'08)*. IEEE. 2008, pp. 175–181.

[2] Hamid Arabnejad *et al.* "A comparison of reinforcement learning techniques for fuzzy cloud auto-scaling". In: *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. IEEE. 2017, pp. 64–73.

[3] Rafael Aschoff and Andrea Zisman. "Qos-driven proactive adaptation of service composition". In: *International Conference on Service-Oriented Computing*. Springer. 2011, pp. 421–435.

[4] Enda Barrett, Enda Howley, and Jim Duggan. "Applying reinforcement learning towards automating resource allocation and application scalability in the cloud. In: *Concurrency and Computation: Practice and Experience* 25.12 (2013), pp. 1656–1674.

[5] Enda Barrett and Stephen Linder. "Autonomous hvac control, a reinforcement learning approach". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2015, pp. 3–19.

[6] Tao Chen and Rami Bahsoon. "Self-adaptive and online qos modeling for cloud-based software services". In: *IEEE Transactions on Software Engineering* 43.5 (2016), pp. 453–475.

[7] Nicolas D'Ippolito *et al.* "Hope for the best, prepare for the worst: multi-tier control for adaptive systems". In: *Proceedings of the 36th International Conference on Software Engineering*. 2014, pp. 688–699.

[8] Xavier Dutreilh *et al.* "Using reinforcement learning for autonomic resource allocation in clouds: towards a fully automated workflow". In: *ICAS 2011, The Seventh International Conference on Autonomic and Autonomous Systems*. 2011, pp. 67–74.

[9] Didac Gil de la Iglesia and Danny Weyns. "MAPE-K Formal Templates to Rigorously Design Behaviors for Self-Adaptive Systems". In: *TAAS* 10..3 (2015), 15:1–15:31.

[10] Pooyan Jamshidi *et al.* "Machine learning meets quantitative planning: Enabling self-adaptation in autonomous robots". In: *2019 IEEE/ACM 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, IEEE. 2019, pp. 39–50.

[11] Jeffrey O. Kephart and David M. Chess. "The Vision of Autonomic Computing". In: *IEEE Computer* 36.1 (2003), pp. 41–50.

[12] Tania Lorido-Botran, Jose Miguel-Alonso, and Jose A Lozano. "A review of auto-scaling techniques for elastic applications in cloud environments". In: *Journal of grid computing* 12.4 (2014), pp. 559–592.

[13] Andreas Metzger *et al.* "Feature Model-Guided Online Reinforcement Learning for Self-Adaptive Services".In: *Service-Oriented Computing – 18th International Conference, ICSOC 2020, Dubai, United Arab Emirates, December 14–17, 2020, Proceedings*. Ed. by Eleanna Kafeza *et al.* Vol. 12571. Lecture Notes in Computer Science. Springer, 2020, pp. 269–286. DOI: 10.1007/978-3-030-65310-1_20. URL: https://doi.org/10.1007/978-3-030-65310-1%5C_20.

[14] Ahmed Moustafa and Minjie Zhang. "Learning efficient compositions for qos-aware service provisioning". In: *2014 IEEE International Conference on Web Services*. IEEE. 2014, pp. 185–192.

[15] Ofir Nachum *et al.* "Bridging the Gap Between Value and Policy Based Reinforcement Learning". In: *Advances in Neural Information Processing Systems 12 (NIPS 2017)*. 2017, pp. 2772–2782.

[16] Alexander Palm, Andreas Metzger, and Klaus Pohl. "Online reinforcement learning for self-adaptive information systems". In: *International Conference on Advanced Information Systems Engineering*. Springer. 2020, pp. 169–184.

[17] Andres J. Ramirez, Adam C. Jensen, and Betty H.C. Cheng. "A taxonomy of uncertainty for dynamically adaptive systems". In: *2012 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE. 2012, pp. 99–108.

[18] Mazeiar Salehie and Ladan Tahvildari. "Self-adaptive software: Landscape and research challenges". In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 4.2 (2009), pp. 1–42.

[19] John Schulman *et al.* "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347* (2017).

[20] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[21] Richard S. Sutton *et al.* "Policy Gradient Methods for Reinforcement Learning with Function Approximation". In: *Advances in Neural Information Processing Systems 12 (NIPS 1999)*. 2000, pp. 1057–1063.

[22] Gerald Tesauro *et al.* "On the use of hybrid reinforcement learning for autonomic resource allocation". In: *Cluster Computing* 10.3 (2007), pp. 287–299.

[23] Hongbign Wang *et al.* "Integrating reinforcement learning with multi-agent techniques for adaptive service composition". In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 12.2 (2017), pp. 1–42.

[24] Tianqi Zhao *et al.* "A reinforcement learning-based framework for the generation and evolution of adaptation rules". In: *2017 IEEE International Conference on Autonomic Computing (ICAC)*. IEEE. 2017, pp. 103–112.

DOI: 10.1561/9781680838251.ch7

Chapter 7

Security of Smart IoT Systems

By Erkuden Rios, Eider Iturbe, Angel Rego, Saturnino Martinez, Anne Gallon, Christophe Guionneau and Arezki Slimani

7.1 Introduction

Ensuring data confidentiality, integrity, and availability while it is being processed, stored and transmitted by all parts of the environment are high-priority concerns in SIS. One of the most interesting approaches to ensure secure behaviour of the SIS is to embed security features such as monitoring, access control, encryption capabilities, etc. into the IoT Platform used as middleware to capture sensors' data and act as gateway to actuators. As SOFIA-SMOOL, or for short, SMOOL [9], is the IoT platform used in the ENACT Smart Building use case (see Chapter 11), the project has worked in extending this platform with built-in features that enable the platform to implement some of the required security controls to prevent integrity, confidentiality, access control and non-repudiation related issues. Chapter 7.2 describes how the SMOOL platform can be used in the SIS development and operation to monitor and control the desired security properties in the access to resources and communications between smart things of the SIS.

Security assurance at operation does require an external service, agnostic to system design but tailored to final system deployment, that is supervising at all times the security behaviour of the different elements in the IoT system. The role of this security monitoring service is to make sure that security incidents or anomalies are

early identified and corresponding alerts are raised to system operators. Chapter 7.3 describes the ENACT enabler supporting at operations the situational awareness of SIS, the so called, *Security and Privacy Monitoring Enabler*. The enabler is capable of collecting data from different layers of the IoT system: network, system and application layers. All these data are combined by the tool for advanced intrusion detection and anomaly detection. Artificial intelligence detection mechanisms are combined with a multi-layer surveillance so as accurate information of holistic security status of the SIS and all its parts is enabled.

Last but not least, Chapter 7.4 brings an innovative approach to access control in SIS. The tool implementing it is named *Context Aware Access Control* since it offers context-based authentication and authorisation of devices and services exchanging data within the SIS. The chapter describes the various manners in which this tool can be used to secure the IoT accesses, considering contextual information in form of a dynamic risk level computation. The context-awareness capability of the tool has been integrated and validated in the eHealth use case described in Chapter 9.

7.2 Built-in Security in IoT Platforms

7.2.1 Security-by-Design in IoT Platforms

Complex systems usually cover the security aspects by adding a layer intersecting or covering other business layers (user interface, data management, processes, etc.). When dealing with IoT systems, the heterogeneous nature of sensors, communication channels, Edge devices or Cloud services often demands the architects focusing on business logic, leaving unattended the needed security controls on sensitive areas (e.g. securing the Edge devices, credentials management for key devices,…), even if some sensors may use weak encryption or even produce data in clear because they rely on transport layer encryption.

Therefore, most of the security management is often handled by the developers creating dedicated solutions on the IoT platform. The platform could provide its own battle-tested security mechanisms but those may not fit well or at all with the security features required by the application developers. In these cases, they are impelled to provide additional security measures to the ones available in the IoT platform. And this may bring problems because when custom security is implemented it is likely that flaws occur, particularly when the developer is not a security expert.

We can introduce the security improvements performed in the SMOOL [1, 9] IoT middleware as an example of how adding security features "by design" generates important benefits, like the use of better security patterns, ensuring developer

confidence on the global security, and focus efforts on IoT application logic development and testing, rather than on security aspects.

In this example, we will analyse an application IoT client component exclusively. These components are always the weakest part in potential attacks, since they have limited resources to implement security mechanisms. While servers can also be attack targets, they are usually better prepared and include more or more robust security controls, and changes in the servers are always reviewed and tested exhaustively to prevent scalability problems or vulnerabilities. In SMOOL terminology, a KP (Knowledge Processor) is a client communicating with a SIB (Semantic Information Broker) or server. The KP can send sensor data or actuation orders, and it can also subscribe to messages emitted by other clients. The SMOOL KP clients compose all the messages in Smart Space Access Protocol (SSAP) format (particular of SMOOL), where the sensors, the data and the metadata are provided in the semantic W3C's Web Ontology Language (OWL) format. This allows other KPs or clients to subscribe to and consume information concepts in the same way for multiple types of sensors. If two different sources such as a complex industrial machine and a simple ambient sensor are providing temperature data to the SMOOL server, another KP could subscribe through the same mechanism to temperature concept in both source KPs to get the temperature value from each.

When embedding security mechanisms in SMOOL IoT platform, three different approaches can be followed, all of them were tested in ENACT and explained below.

7.2.1.1 Custom code of security controls in the KPs

The first implementation of security features within SMOOL clients consisted in adding security metadata to the business data, that is, for example, adding security metadata to the sensing data transmitted by sensors. For instance, if the client was transmitting temperature data (value, unit, timestamp), we could also attach the type and content of security information. These metadata were added as semantic concepts in SMOOL ontology so as they can be published and subscribed to by KPs just the same as KPs do with business semantic concepts. This way, specialized security KPs could only listen to which security data is flowing, instead of subscribing to all sensor types containing security metadata. The new security concepts added in ENACT to the SMOOL ontology covered authentication, authorization, confidentiality, integrity and non-repudiation. They were created to allow flexibility in the exact implementation of the security (for instance, integrity can allow symmetric or asymmetric key-based payloads). The new security concepts are shown in Figure 7.1 just as they appear in the Protégé application [13] used to visualize the ontology.

This way, the KP developers could create sensor KPs that publish sensor data with security information. Other KPs subscribed to the sensor data would check

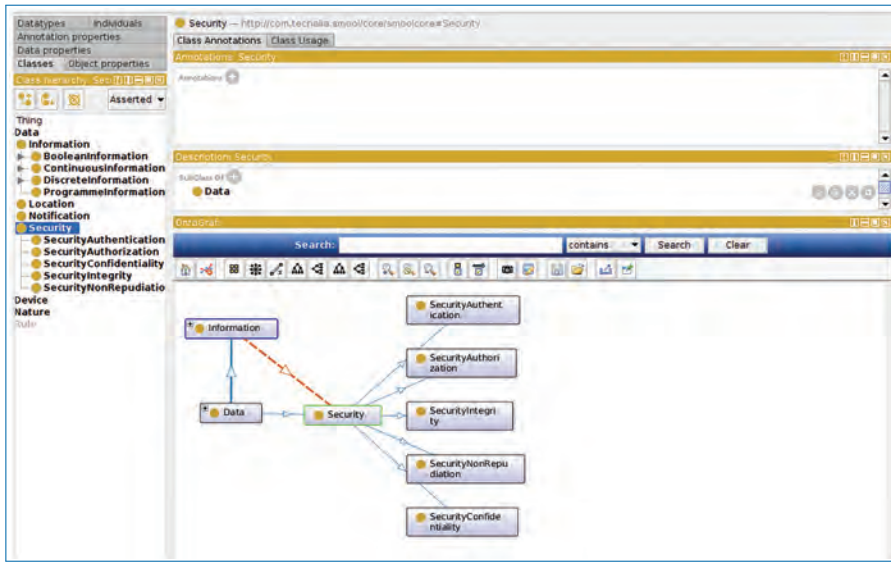

Figure 7.1. SMOOL ontology: Security metadata as ontology concepts.

the security data before accepting any values contained in the message. For instance, the publishers of sensor data could add integrity data, while actuation orders would be sent with valid authorization data.

This approach has several potential failures. The first one is that the developer must add extra code to manage security, which means more lines to peer-review and test, and the application implementation would be prone to insecure execution paths when running. The second problem is that the developer could miss some of the things to double-check when using one of the security concepts in the ontology. For instance, the need of salting before hashing encrypted payloads, or the need to authenticate the device before allowing it to publish data. The third issue is that freedom in security coding increases the list of potential security mistakes a non-expert may make and an expert should review.

In Figure 7.2 a simplified version of KP layers is displayed. When a message containing sensor data (in SSAP format) arrives, the first layer is the Comms layer or communications stack, responsible for accepting and assembling the message by using any of the allowed connectors (TCP, Bluetooth, etc.). The second layer is the Model layer where various operations on the message take place, such as parsing, data insertion into the ontology containers, comparison of previous and updated values, etc. These are the core layers, which facilitate the work of IoT application developers but their drawback is that they are black boxes for them. The next layers are the ones created for the real application, including the Custom code layer to collect, send or retrieve sensing data, and the Security layer with the security code. The figure shows these two layers separated logically, although in reality they

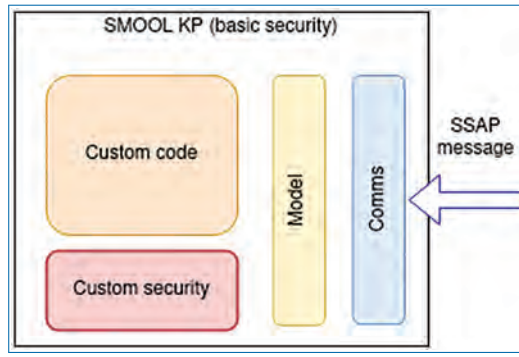

Figure 7.2. SMOOL KP (client) layers. Security is handled by the KP developer.

are glued together. The Security layer deals with the management of the security elements. Since these elements are also part of the ontology, in ENACT we have developed a user-friendly API for creating content of these elements.

7.2.1.2 Basic SecurityChecker in the core of the KPs

In the second approach to security for KPs, the aim was to provide a better experience for the KP developer by providing basic built-in security from the KP design, and therefore, preventing potential security flaws introduced by inexperienced developers. Security policy usage philosophy was added to the KP and implemented in the KP core layers. The security control is performed by a SecurityChecker class added in the security layer, which works on all messages received by a KP extracting any security concept present in the message and testing it against a list of policies. If the message does not conform to the policies, the message will be rejected directly from the core layer, so custom code layer will never be aware that the message was received. This solution is more efficient because there is an automatic security check installed on every newly generated KP, since the mechanism comes in the KP design itself. The developer has also fewer lines of security code to implement, because, instead of needing to program the checks of every message for different security constraints fulfillment, some simple one-line rules are defined as policies. For instance, all actuation orders to a specific actuator type (e.g. blinds in a smart building) must contain authorization metadata.

In summary, this second approach, depicted in Figure 7.3, introduces two major differences compared to the previous approach in Figure 7.2. First, the security layer is now part of the core layers, shown as a single vertical layer that works for all the messages, prior to the custom code execution. Second, custom security code is smaller in number of lines, because it would only be dedicated to the enforcement of advanced security features, such as the management of sensitive data, while most of the messages will be filtered or passed by the core security layer.

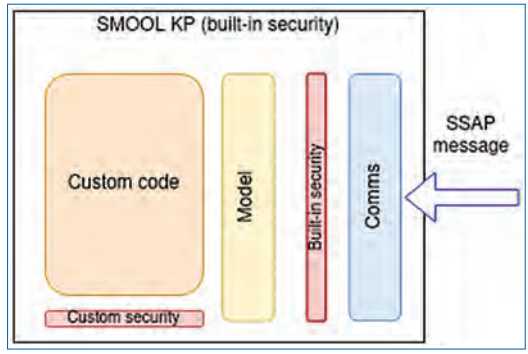

Figure 7.3. SMOOL KP (client) layers. Security is enforced for all messages.

7.2.1.3 External SecurityChecker called from the core of the KPs

The third iteration of built-in security in SMOOL IoT KPs takes advantage of the possibility to use enhanced or external security elements to enforce the needed security policies at all times. This way, the security policies would be completely independent from the KPs and adjustable when needed. The approach extends the built-in SecurityChecker of the KPs to provide better security controls. And these controls are still performed in the core layer, in the same manner as in the second approach. When designing the application, these elements are added as a dependency to replace the standard built-in SecurityChecker. The enhanced controls will be loaded when starting the KP.

To demonstrate this, we have used GENESIS for deploying refined security controls from the design phase. Since SMOOL and GENESIS were integrated in ENACT to allow deploying KPs with application extended features, we can also add security features to run either improved extensions of the KP security core layer, or custom security code. Depending on the security needs of the target IoT system, GENESIS will deploy a different implementation of the security policies, but from the design point of view, the declaration of the security enforcement of the policies is the same regardless what checks the external SecurityChecker will do. In Figure 7.4 below, the security core layer is bigger in lines of code. But for the KP developer the security complexity is the same as in the previous iteration.

Now, the core security box is bigger; however, the knowledge about how security is working in our system remains the same, thanks to the use of policies as main concept. Information reaching the custom code can be treated as secure, for all new security upgrades. The security schema remains straightforward, and the application can keep growing by focusing on the business logic features of the KPs (the custom code layer box) rather than security concerns that are handled outside of it.

Therefore, embracing security features of IoT environments from the design phase carries a set of benefits. The most important benefit is that the majority

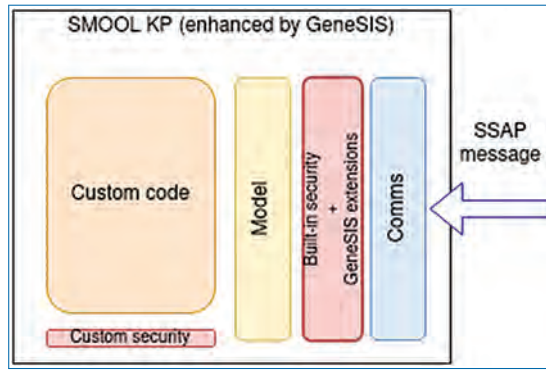

Figure 7.4. SMOOL KP (client) layers. Security is enhanced by GeneSIS redeployment.

of the security flaws can be avoided even when building the very first prototype. The software design can provide some security elements as mandatory and block non-secure messages, and all of it in a user-friendly manner. The second benefit is that developers can trust security elements provided from the design phase instead of needing to develop security features based on ad-hoc preferences. The application to be deployed will be more secure and it would be easier to make new releases in the future. Trends in IoT show that security remains as a main concern, and ENACT has demonstrated that IoT applications trusting on security-aware IoT platforms such as SMOOL can be created in a secure manner to prevent most known issues (legacy libraries or software pieces containing vulnerabilities, broken encryption mechanisms, credentials leakage, etc.). This way, securing applications during the design phase can prevent unexpected risky situations when deploying IoT solutions to production environments.

7.2.2 Reaction to Cyber Incidents and Anomalies

Smart IoT systems allow devices and data generated to be in the core of the system behaviour, leaving human interaction as trigger elements or passive receptors of actions. In SIS most of the elements must run autonomously, re-adapt based on rules, start and stop things, etc. In fact, the SIS behave as complex ecosystems where elements can have different degrees of intelligence but all of them share a high degree of autonomy. And in these systems, a preventive control monitoring what is happening in terms of security is important, but also a reactive control when things go wrong, i.e. issues are detected. For example, hacking only one of the devices in a SIS could create dangerous situations. Imagine a hacked temperature sensor sending low values to keep a heat system running all the time. Now, imagine a hack of a gas or smoke sensor to forge the sensed dangerous values and prevent them from being detected.

Therefore, apart from monitoring and identifying potential issues and attacks, SIS security must be reactive to take countermeasures in real time. The best way to ensure control is having administration rights on the smart devices and Edge, but not all IoT elements can be controlled (for instance, generic sensors or devices from external vendors ready for plug-and-run). Thus, control must sense the IoT system and must act on it, and in cases where the device cannot be managed from the inside, the control must be done from some other part of the communication or data processing chain.

Some security control systems are ready to detect general problems and react to them. Imagine a new device joins a weak security wireless network. This device could start flooding the communications in the network, creating a denial of service for every other legitimate device. The security control system can detect and block that element, no matter which type of device it is.

Now, a more intelligent and refined malicious IoT device could connect to the same security weak network and send legitimate data shaped in the same format other devices are using in their transmissions. The device is accepted, and the data is also accepted because it fits the format expected to be processed. In this case, a smart control element should understand operational data, so as to be able to detect abnormal values and provide feedback to the Security control system.

In the previous Chapter 7.2, we saw how smart IoT devices could enforce security controls based on policies. The enforcement was done inside the device itself. But not all issues could be detected in the device, and security updates may not be available once deployed. For the security issues not detected and blocked from the IoT devices, we need reactive security mechanisms dealing with them. In ENACT he have developed a reactive security control system that relies on SMOOL platform and its clients as explained below.

Let's go back to the SMOOL IoT platform we described in the previous Chapter 7.2. The clients or KPs connected to the IoT platform exchange complex messages. Some security features are already handled by the core security layer embedded in every KP, and some other security issues are handled by the generic security control system that reacts to incidents notified by the monitoring system described in Chapter 7.3. However, to enforce security in the communications of the SMOOL ecosystem (the server and the smart things connected to it) we have created a Security KP to control what information is really exchanged in the messages of the other KPs, detect non-secure elements, notify the incident, and block the insecure elements.

Figure 7.5 depicts an attack performed by an elaborated rogue application disguised as an IoT client, i.e. a SMOOL KP. The malicious KP behaves as a normal KP so connection to server and message exchange is allowed. Note that other type of rogue KPs will be rejected if their message structure does not conform to the one

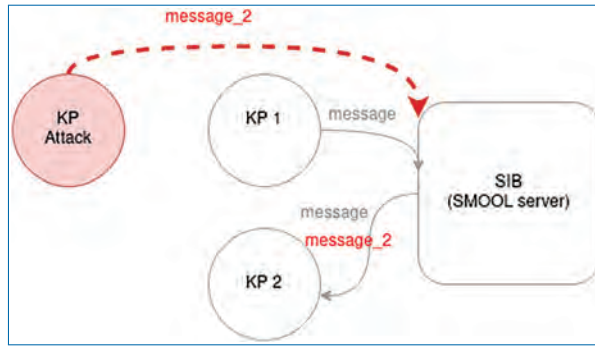

Figure 7.5. SMOOL. Malicious application disguised as a KP.

required by the IoT platform. Thus, this attacking KP is a real danger in the system and cannot be detected by the security infrastructure. The client KP2 receives messages from benign KP1 but also malicious messages from KP Attack (in red).

To solve this problem, a Security KP was created which is a special SMOOL client that has the unique ability to access and understand every SMOOL message. The Security KP, being a client rather than a library in the SMOOL server, has another characteristic: it can be upgraded with new features or customized controls faster than if it was allocated inside the server.

Instead of subscribing to all kinds of SMOOL messages and all the ontology concepts, the Security KP can subscribe to a subset of concepts corresponding to those security properties it needs to handle reactions for. Since security metadata was added in the ontology in the same manner as business data, the Security KP can process all or part of the messages to produce faster reactive responses. The first filter could be to check if messages are following the security polices, then inspecting the actual security metadata, and finally, looking for anomalies in the logic or operational data. If any unwanted message is detected, the Security KP has the right credentials to invoke the global Security control system to analyse the metadata or request it to block all communications from the device generating these messages.

Since the IoT server is the real link to the insecure device, a minor implementation for blocking KPs has been developed. This action can be performed only by the global Security control system. Figure 7.6 illustrates how the Security KP can detect refined attacks.

This time, even if the security metadata sent by the malicious KP Attack was fine (and therefore, the KP2 did not reject the message), the other metadata of the message were not compliant with the refined security rule set in the Security KP, so this KP requests the Security control system to block the KP Attack. The Security control system orders SMOOL server to cut off the connection for the offending KP, and add the offending IP to the black-list. The attacker KP would not be able to send further messages because it will not be even able to connect to the server.

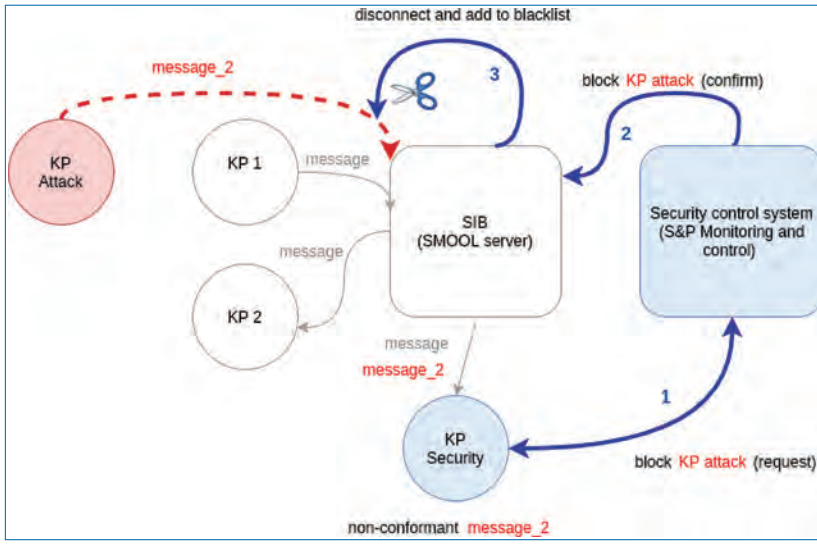

Figure 7.6. SMOOL. Malicious KP detected by Security KP.

By using the Security KP, the application-level messages can be tested against a detailed rule set, the IoT data can be transformed into another format that the Security control system could parse, and problems can be detected and blocking orders invoked too. The application can also detect even more sophisticated attacks depending on the use case, because it could have an additional white-list of allowed IoT devices based on historical activity, or detect abnormal behaviour values from legitimate devices, and then request the Security control system to inspect the device.

7.3 Continuous Monitoring and Detection in IoT System Operation

Information Security Continuous Monitoring (ISCM) is defined by NIST as *"maintaining ongoing awareness of information security, vulnerabilities, and threats to support organizational risk management decisions"* [7]. In an IoT environment, implementing ISCM provides the security administrator of the SIS with means for continuous situational awareness of the cybersecurity and privacy status of the system. This resource supports the security administrator by identifying cybersecurity incidents along with the targeted assets, as well as informing about the criticality and importance of those incidents so that the security expert can decide on the best cybersecurity strategy to mitigate the cyber threats and protect the assets.

In order to better comprehend the importance of the continuous security monitoring, a review of the NIST Security and Privacy Controls for Information Systems

and Organizations (SP 800-53 Rev. 5) [8] gives the following quick conclusion: at least 68 security controls of the catalogue are explicitly associated with the monitoring activity distributed in 9 control families: *Access Control, Audit and Accountability, Assessment, Authorization, and Monitoring, Configuration Management, Incident Response, Physical and Environmental Protection, Program Management, Risk Assessment, System and Communications Protection, and System and Information Integrity.*

The standard ISO/IEC 27035 identifies multiple technologies as sources of the required security information and events of continuous monitoring as part of detection and reporting phase within the security incident management process [12]. Mentioned technologies include: Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS), honeypots, log monitoring systems, security information and event management systems, and network monitoring systems, among others.

Implementing and deploying ISCM mechanisms into IoT environments may become a complicated task due to the high heterogeneity of standards, technologies, protocols and deployment architectures in use. Despite of the complexity, trust models based on security and privacy technologies deployed in IoT systems will be more and more necessary to ensure consumer acceptance [6].

7.3.1 Architecture and Main Capabilities

In ENACT, ISCM area is covered by the Security and Privacy Monitoring and Control Enabler (S&P Mon&Con), which aids the SIS operator in learning at all times the security status of the SIS and control the behaviour of the SIS in order to ensure it adheres to the security requirements designed. This Enabler delivers three main capabilities:

- Flexible and extensible continuous monitoring mechanism. A comprehensive security monitoring involves having granular, modular and dynamic security controls to be coordinated with. In this way, the enabler can be adapted to work properly with different types of security controls as data sources, and furthermore, the enabler can even be configured to respond through the use of specific security controls deployed in the SIS itself.
- Advanced anomaly detection through user and entity behaviour analysis using Artificial Intelligence (AI) techniques. Based on a zero trust security model, the enabler follows a cybersecurity strategy of addressing both internal and external threats. Particularly, all internal users and entities are considered for the anomaly-based Intrusion Detection System analytics.
- Scalability of the solution. Both the modular architecture and the technologies the enabler is based on guarantee the solution is able to rapidly scale up in large-scale IoT system scenarios, which also implies the need to deploy

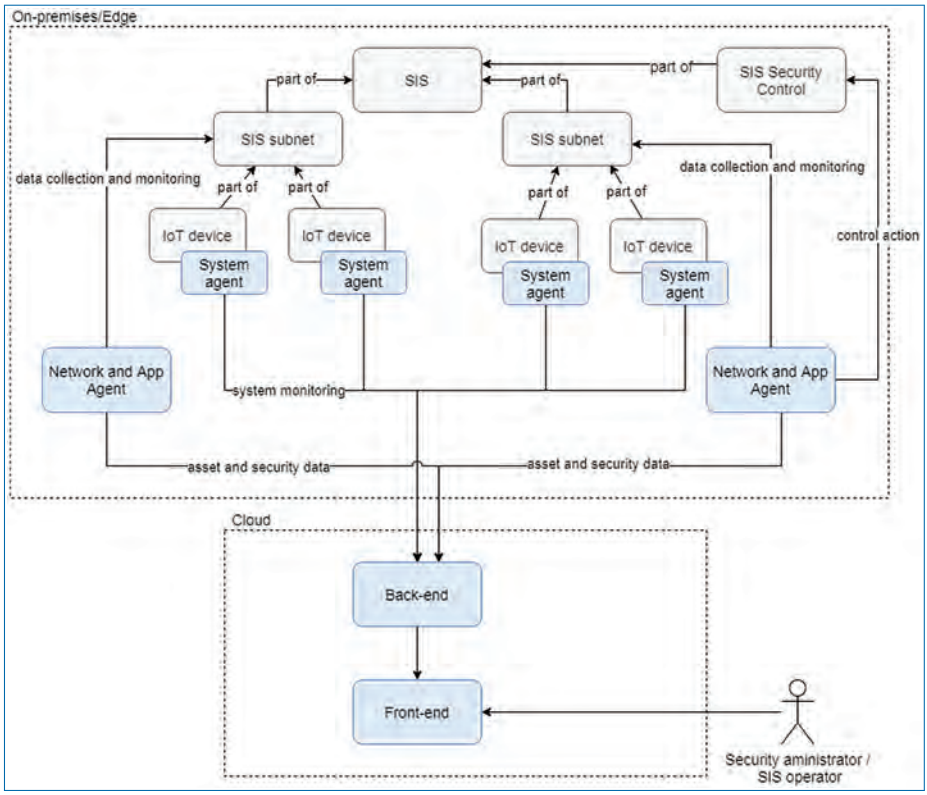

Figure 7.7. High-level architecture of the Security and Privacy Monitoring and Control Enabler.

hundreds or thousands of monitoring agents depending on the extension of the system.

Figure 7.7 shows the high-level architecture of the Security and Privacy Monitoring and Control Enabler. The enabler captures different kinds of data from the SIS through multiple distributed probes named *agents*. There are three types of monitoring agents:

1. *Network agent*, which captures network traffic data, network related security events and asset related data. It integrates an open source signature-based network IDS that is able to detect well-known security attacks and generate security events accordingly. Besides that, the agent includes capabilities to generate protocol-specific security events, e.g. related to ARP protocol so as to enable the detection of ARP spoofing attacks.
2. *System agent*, which gathers log information and data related to the activities and processes within the devices of the SIS; and,

3. *App agent*, which collects data at application layer and generates security events accordingly. This agent can be customized depending on the SIS characteristics; e.g. if the SIS is in intensive use of the MQTT protocol, this agent can be developed with specific MQTT based rules for monitoring and controlling that only authorized users and assets (smart things, devices, services, etc.) can communicate in the SIS.

All the data gathered by the monitoring agents is sent to the back-end of the enabler. Depending on the size of the SIS, the amount of data recovered after some time can be huge which would likely cause processing difficulties. In order to avoid a bottleneck at next phases of data processing and analytics, the entry of the back-end is implemented by a streaming bus (based on the open source distributed streaming platform Apache Kafka [4]. Moreover, the streaming bus provides extensibility to the solution by offering multiple channels where various kinds of data can be collected, and it also allows exchanging the outcome from the enabler in form of identified security events with external components.

The back-end of the enabler includes a data storage infrastructure which stores in a NoSQL database (based on Elasticsearch [2]) all the acquired and pre-processed data from the agents. Multiple indexes are created in the storage infrastructure depending on the different sources of data. For example, in a Smart Building IoT system where many communication protocols can be working at the same time, the network data can easily be stored with the definition of approximately 1800 network attributes as shown in Figure 7.8. Bearing in mind that network traffic is only one of the multiple data sources considered for the analytics within the enabler, dealing with enormous amounts of heterogeneous data is one of the major challenges addressed by this enabler in IoT environments.

The main ground-breaking part of the enabler is the anomaly detection service included in the back-end. AI techniques have been leveraged to analyse the collected SIS data and security events in order to detect anomalies in the system. Mainly, unsupervised Deep Learning techniques have been used to perform the SIS behavioural analytics and anomaly detection. Many Deep Learning techniques have been studied depending on the SIS characteristics so as to implement an accurate anomaly-based detection system. Figure 7.9 depicts Vanilla Long short-term memory (LSTM), a type of recurrent neural network (RNN) architecture, predictions for MQTT protocol in a Smart Building system showing as red points the anomalies detected correctly, where real MQTT traffic behaviour deviates largely from predicted one.

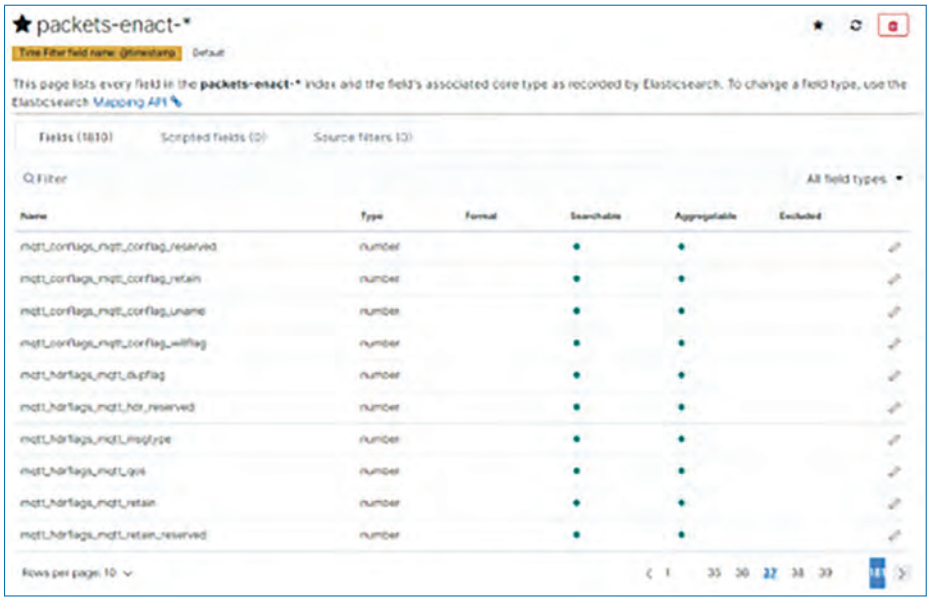

Figure 7.8. Extract of the network fields of stored indexes in the Smart Building.

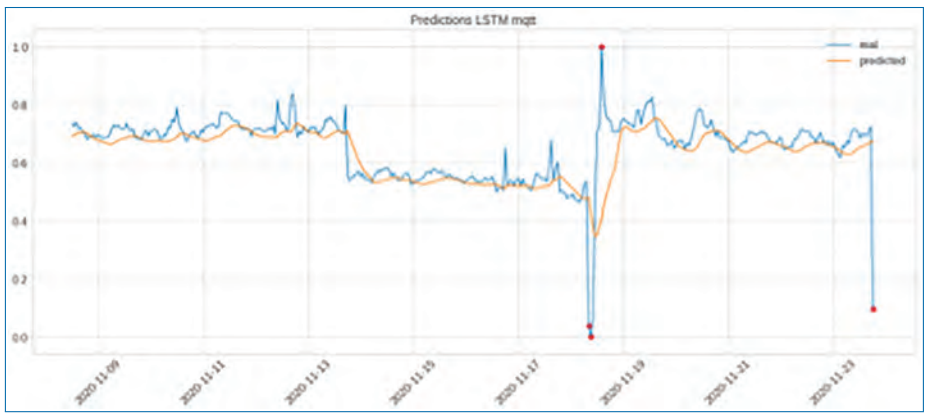

Figure 7.9. Vanilla LSTM predictions for MQTT protocol in the Smart Building.

Considering that each SIS can be completely different in terms of diversity of network protocols in the communications, type and number of devices, type and amount of operational data exchanged, etc. the anomaly detection capability of the enabler must be adjusted to each type of SIS, which means that the detection models need to be re-trained for each SIS.

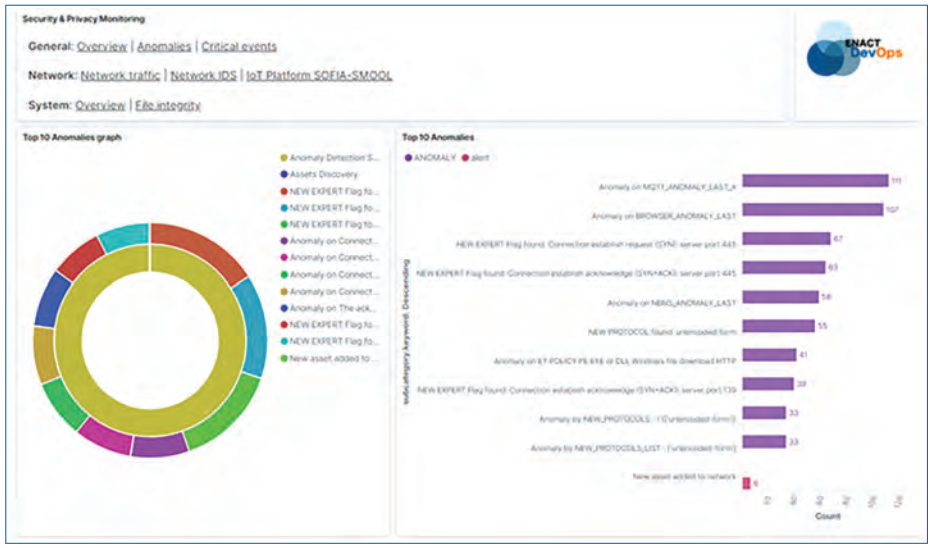

Figure 7.10. Extract of the overview dashboard view in the Smart Building.

The security administrator or SIS operator can continuously be aware of the cybersecurity status of the SIS using the front-end of the enabler, which displays all SIS status data, prediction data and detected security events in a user-friendly manner in form of alerts, statistics and graphs. Within ENACT, multiple viewpoints have been implemented in the front-end in order to have a comprehensive overview of the SIS security status; additionally, each of the viewpoints includes many dashboards. Nevertheless, the front-end can be adapted ad hoc in case the end user wants to have more details, or customize the graphs and the rest of the visualization objects.

Figure 7.10 shows an extract of the overview dashboard view of the General viewpoint in the enabler implemented for the Smart Building System use case (cf. Chapter 11). It offers the most important information related to the security events generated over the SIS to protect. The end user can navigate through the rest of the dashboards to learn more details about security events.

Figure 7.11 shows the network traffic dashboard of the Network viewpoint and Figure 7.12 shows an extract of the anomalies dashboard of the General viewpoint. Both dashboards have also been customized for the Smart Building System use case.

The Security and Privacy Monitoring and Control Enabler has been designed with the capability to integrate with an IoT platform, specifically with the SMOOL

Figure 7.11. Extract of the network traffic dashboard of the Network viewpoint in the Smart Building.

IoT Platform (cf. Chapter 7.2). This particular implementation allows the security administrator to continuously monitor all communications and data exchanged through the IoT Platform. Figure 7.13 shows the architecture of the enabler integrated with SMOOL IoT Platform.

The distinctive feature in this scenario is that a client of the SMOOL IoT Platform, called Security KP (cf. Chapter 7.2.2), works as an app agent for the enabler by monitoring all data and communications through the IoT Platform and identifying

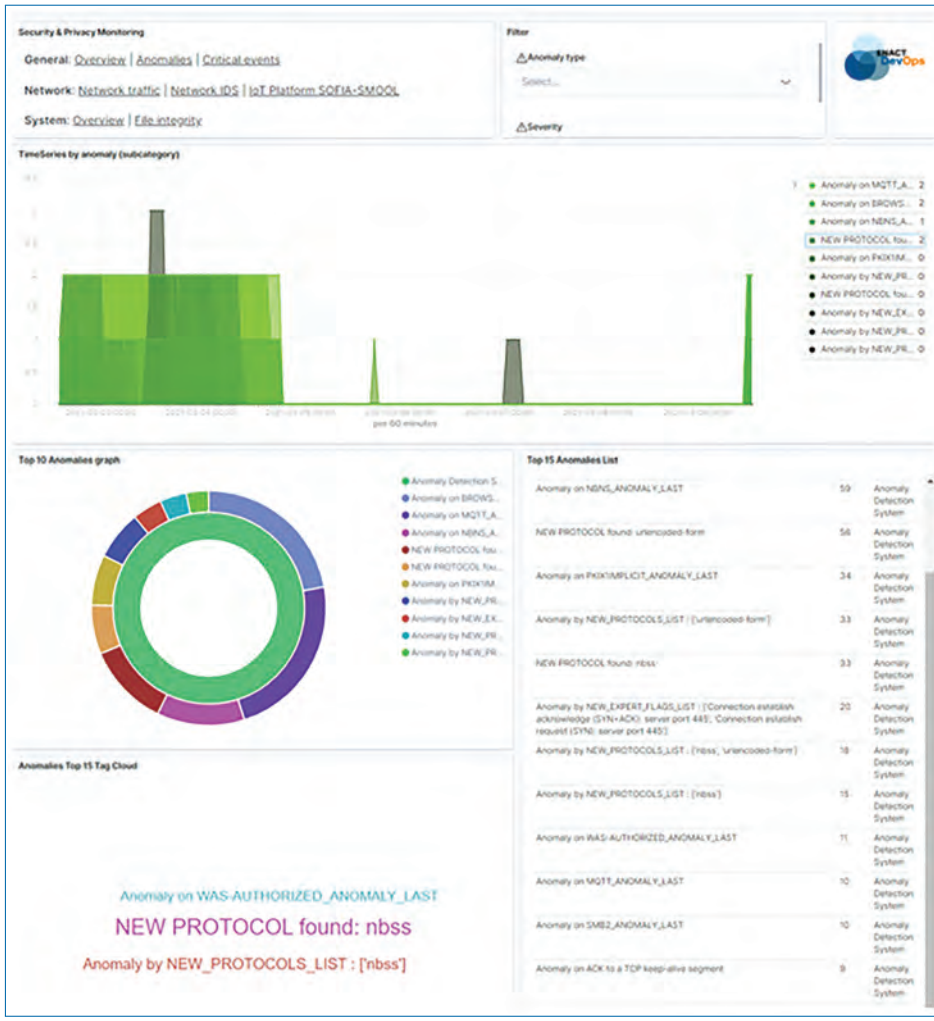

Figure 7.12. Extract of the anomalies dashboard of the General viewpoint in the Smart Building.

potential malicious intrusions. Furthermore, the integration with SMOOL IoT Platform provides the enabler security control capability to react to security incidents. For example, when detecting an unauthorized communication within SMOOL IoT Platform by the app agent (i.e. the Security KP), the enabler can respond by blocking all communications coming from the unauthorized client. All these security events registered by the SMOOL IoT Platform are shown in the front-end of the enabler.

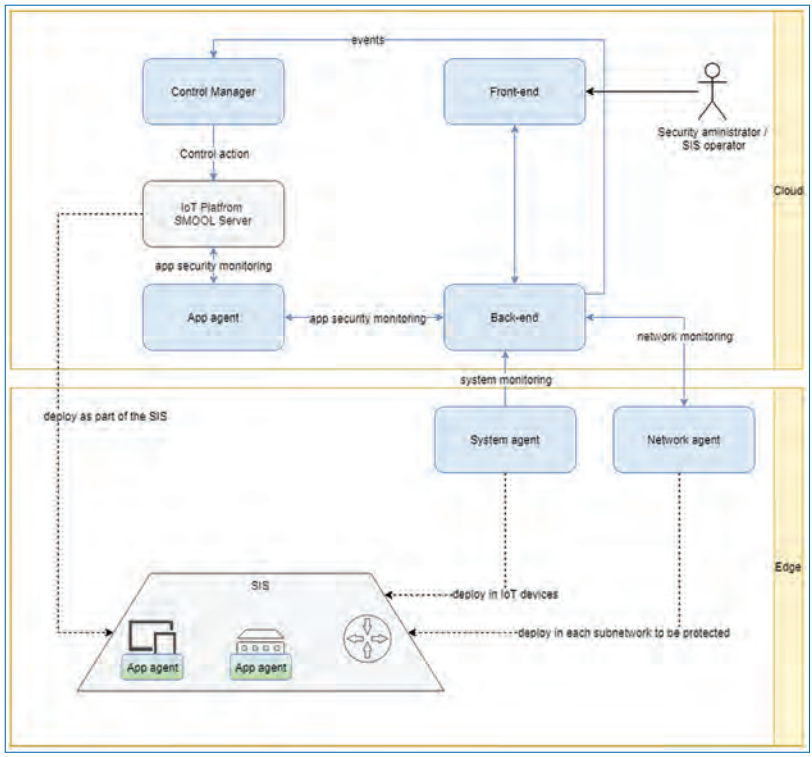

Figure 7.13. Architecture of the the Security and Privacy Monitoring and Control Enabler integrated with SMOOL IoT Platform.

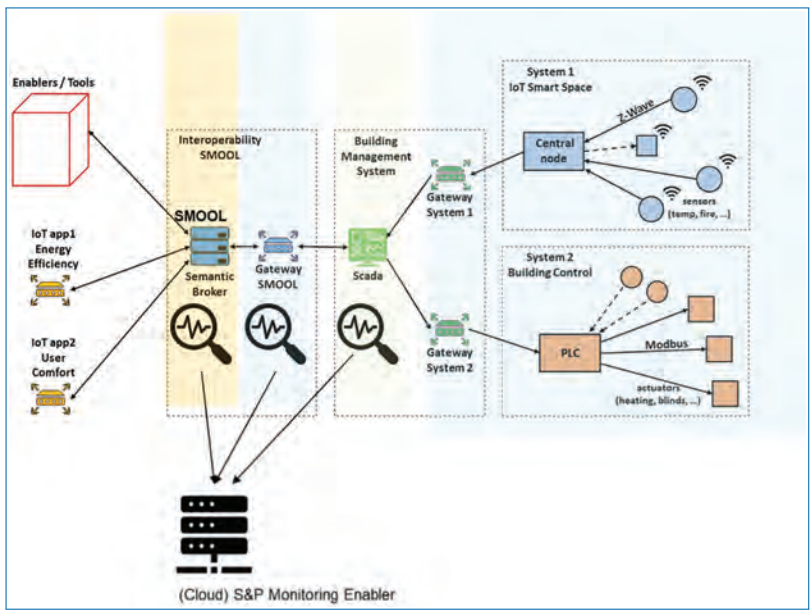

Figure 7.14. Smart Building SIS high-level architecture with the Security and Privacy Monitoring and Control Enabler integrated.

7.3.2 Validation

The Security and Privacy Monitoring and Control Enabler has been validated in two different use cases in the ENACT project.

7.3.2.1 Smart Home System

The Smart Home System (cf. Chapter 11) has integrated the Security and Privacy Monitoring and Control Enabler together with the SMOOL IoT Platform in order to monitor the IoT applications of user comfort and energy efficiency of the building. In that way, the Smart Home System has been monitored at different layers: network layer covered by network monitoring agents, system layer covered by system monitoring agents in IoT devices such as Raspberry Pis, and app layer covered by SMOOL KP clients as app agents.

The SIS operator of the Smart Building System has been able to discover anomalies and security incidents by using the enabler. For example, Figure 7.15 shows an anomaly in the network protocols used by the Smart Building system related to HTTP protocol's content type formats (such as json, image-gif or png); this

Figure 7.15. Example of network anomaly detection in the Smart Building.

anomaly can be seen in different graphics of the dashboard: (i) in a timeseries graphic or (ii) in a top 10 anomalies list.

7.3.2.2 Intelligent Transport System

The Intelligent Transport System (cf. Chapter 10) has integrated the Security and Privacy Monitoring and Control Enabler in order to monitor the security status of the on board Edge part of the train system. The enabler has been used and validated for the following scenarios:

1. User and entity behaviour monitoring, which is mainly based on the anomaly detection capabilities offered by the enabler. Industrial protocol network traffic has been analysed in order to detect potential security incidents related to abnormal traffic behaviour. When an anomaly is detected, the enabler is able to react by enabling a specific security control for the SIS itself.
2. Intrusion detection, which uses rule-based detection capabilities implemented within network and app agents of the enabler to spot the unauthorized users and devices trying to communicate or get access to resources in the system.

7.4 Context-aware Access Control

7.4.1 Purpose

Access control and identity governance mechanisms are cornerstones of security and privacy, which is today focused on addressing people accessing IT applications. In the context of the Internet of Things, access control needs to be extended to address not only people accessing IoT, but also to manage the relationships between connected things. This requires designing and building new access control mechanisms for authorizing access to and from connected things, with ad hoc protocols while still being able to address traditional access to IT applications.

The key challenge for access control in IoT is dynamicity. IoT systems are changing continuously: Devices keep entering and exiting the system; The same devices may be used in different context; New connections emerge among the devices; etc. For such highly dynamic IoT systems, access rights from people to devices, and from devices to devices, are not immutable. The access rights may vary according to the context change. Take an eHealth scenario as an example, where senior adults use IoT devices to monitor their physiological data such as blood pressure. In the normal, day-to-day context, only the user himself should have the access right to the data, due to the privacy concern. However, in a special context, such as under

emergency rescue, medical staff should be granted with the access right to the journal with historical physiological data. Therefore, the decision of access right in IoT systems must be made with awareness of the context.

The objective of the Context-aware Access Control (CAAC) is to deal with these considerations, by providing dynamic access control mechanisms for IoT systems based on context awareness and risk identification, applicable to both IT (Information Technology) and OT (Operational Technology) domains, through an IAM (Identity and Access Management) gateway for IoT that includes next-generation authorization mechanisms.

Evidian Web Access Manager (WAM) provides security features for identity management and access control. The Context-Aware Access Control tool is an evolution of the authentication and authorization mechanisms provided by WAM intended for the Internet of Things.

7.4.2 Background: Industry Standards of Access Control Protocols

- **The traditional dynamic access control chain based on the XACML model**

  A first approach is to study how the traditional dynamic access control chain based on the XACML model [10] could help to answer the challenge of securing the Internet of Things.

  XACML is a policy-based management system that defines a declarative access control policy language implemented in XML and a processing model describing how to evaluate authorization requests according to the rules defined in policies. As a published standard specification, one of the goals of XACML is to promote common terminology and interoperability between authorization implementations by multiple vendors.

  XACML is primarily an Attribute-Based Access Control (ABAC) system, where attributes associated with an entity are inputs into the decision of whether a given entity may access a given resource and perform a specific action.

  The XACML model supports and encourages the separation of the authorization decision from the point of use. When authorization decisions are baked into client applications, it is very difficult to update the decision criteria when the governing policy changes. When the client is decoupled from the authorization decision, authorization policies can be updated on the fly and affect all clients immediately.

  The access control chain based on the XACML model is depicted in Figure 7.16.

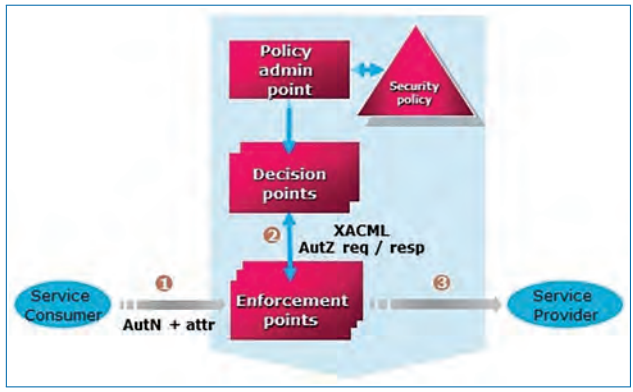

Figure 7.16. The dynamic access control chain based on the XACML model.

In this chain:

- The Policy Decision Point (PDP) evaluates access requests against authorization policies before issuing access decisions.
- The Policy Enforcement Point (PEP) intercepts the user's access request to a resource, makes a decision request to the PDP to obtain the access decision (i.e. access to the resource is approved or rejected), and acts on the received decision.

In fact, this approach is dynamic by essence, since the access control decisions are made based on attributes associated with relevant entities. In addition, it offers a powerful access control language with which to express a wide range of access control policies.

But the following points make this approach prohibitive:

- An approach based on rules is difficult to administer. Defining policies is effort consuming. You need to invest in the identification of the attributes that are relevant to make authorization decisions and mint policies from them. In addition, the ABAC system introduces issues, most notably the 'attribute explosion' [3] issue and, maybe more importantly, the lack of audibility.
- Although Service-Oriented Architecture and Web Services offer advanced flexibility and operability capabilities, they are quite heavy infrastructures that imply significant performance overheads.
- Since XACML has been designed to meet the authorization needs of the monolithic enterprise where all users are managed centrally, this central access control chain is not suitable for cloud computing and distributed system deployment, and it does not scale to the Internet.

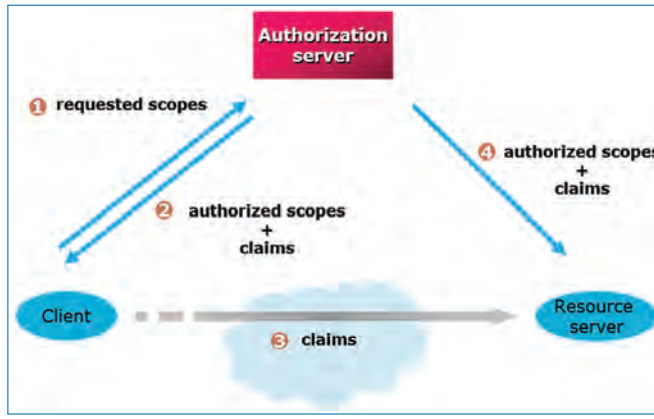

Figure 7.17. Another approach based on OAuth 2.0.

- **Another approach based on OAuth 2.0**

Another approach has been studied, based on the OAuth 2.0 industry-standard protocol for authorization [5].

This approach is depicted in Figure 7.17.

In this approach, a client can access a resource on behalf of a user through an authorization delegation mechanism. This assumes that the user has given his consent for the requested scopes.

As a major advantage, this protocol can be implemented in a light way, by leveraging HTTP and REST-based APIs. In fact, OAuth 2.0 supports the mobile device application endpoint in a lightweight manner. Its simplicity makes it the de-facto choice for mobile and also non-mobile applications. Due to the growing importance of Cloud technologies and APIs, the REST architecture is now heavily favoured.

In addition, OAuth 2.0 allows a fluid integration with role management: OAuth 2.0 scopes can be used to provide role-based authorization.

But this protocol does not have the granularity of XACML in terms of rules. And another point is still an obstacle to meet the need of an IoT context-aware access control: the dynamicity, allowing to take into account the context, is not provided by design.

7.4.3 A Solution for a Context-aware Access Control Approach for IoT

Due to the disadvantages observed on the traditional dynamic access control chain based on the XACML model, it appears that a solution based on OAuth 2.0 is more appropriate.

But to provide an IoT context-aware access control mechanism, the gap must be filled to deliver dynamic authorizations based on context by using the OAuth 2.0 protocol.

Starting from security features for identity management and access control based on the protocols OAuth 2.0 and OpenID Connect (OIDC) [11], the approach is to develop an evolution of these authentication and authorization mechanisms intended for the Internet of Things. Due to the dynamic nature of the data regarding the environment of the connected devices and the persons they belong to, this contextual information must be used to manage and adjust the security mechanisms, i.e. consider contextual information in the identification of the entity requesting access and in the evaluation of the conditions to grant access.

By assessing the applicability of OAuth 2.0, the ENACT IoT context-aware access control leverages it as a key protocol for interoperability, by adding dynamicity to the authorization decisions produced by OAuth 2.0, although this was not originally intended in that protocol. This dynamic capability is in charge of taking contextual information into account and inserting it into authorization decisions.

7.4.4 Architecture

The Context-aware Access Control tool provides an authorization mechanism that issues access tokens to the connected objects after successfully authenticating their owner and obtaining authorization. An access token contains the list of claims and scopes that an authenticated user has consented for this object to access. These scopes and claims are used to restrict accesses to the back-end server APIs to a consented set of resources.

This authorization mechanism may be coupled with contextual information to adapt the access authorizations according to them (for example to make certain information more widely available in some urgent case).

To this objective, the Access Control Tool directly communicates with a Risk Server to make dynamic access controls based on the context information during the authorization phase. For example, it can reject the authorization if the access token is valid while other context information does not respect the authorization policy.

The authorization policy is a set of rules that define whether a user or device must be permitted or denied access to a back-end server. An administrator can control this adjustment and create special authorization rules based on the context data provided.

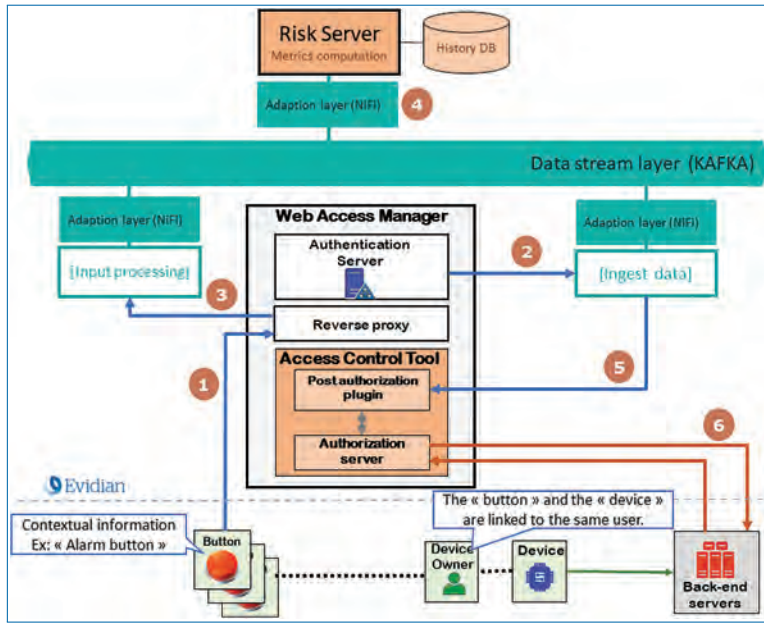

Figure 7.18. Context-aware Access Control global infrastructure.

The Context-aware Access Control tool is provided inside an infrastructure aimed to gather contextual information to deduce a risk level associated with a user. Figure 7.18 gives a global view of this infrastructure.

The infrastructure is based on the Apache Kafka event streaming platform, which allows to publish and subscribe to streams of events. The principle is to publish in this platform contextual information which may come (1) from connected devices (sensors, alarms, etc.) or (2) from audit events produced by the Evidian Web Access Manager. The contextual information is sent to an input processing interface (3) which then publishes it to a Kafka topic. An event is then received by the risk server from an Apache NiFi interface (4) which will take into account the contextual information in a dynamic risk level computation. Then, when a device tries to access a resource, (5) the CAAC retrieves the dynamic risk value associated to the device owner, and (6) this is transmitted to the back-end server to modulate the access accordingly.

In this architecture, two components are providing the Context-aware Access Control mechanisms:

- The **Access Control Tool**, composed of an Authorization server associated to a Post authorization plugin, to add more controls during the authorization phase. Its purpose is to check if the request is authenticated and is authorized to access the back-end server.

Indeed, each time a device sends a request to a back-end server (7), WAM can check the dynamic claims and scopes consented by the user associated to the device that performs the request and, in turn, realize special actions according to this information such as blocking the request or limiting the accessible resources.

The Post authorization plugin extends the basic authorization phase and is entirely customizable. Any operation can be executed during the authorization phase, including calling external programs, and in particular the Risk Server. The Post authorization plugin can create injection variables that can be reused and injected in the initial request sent to the back-end server.

- A **Risk Server** which essentially relies on WAM audit events to calculate a risk level for each user. This allows detection of abnormal behaviour such as connections from unknown IP addresses, or multiple failed connections.

  A user's risk level is a function of the level of trust given to that user. This level of risk determines the level of trust that can be placed in the devices owned by that user. The user's risk level is based on a system of sanctions/rewards depending on the user's behaviour. Its computation uses a ranking system based on a user-specific score: the Risk Score.

  Contextual information coming from external sources (sensors, other applications, etc.) makes it possible to modulate this risk, i.e., to increase or decrease it depending on the situation. For example, in the case of a fire, a smoke detector immediately sends information to the Risk Server, which will considerably reduce the user's risk and allow easier access to resources.

  The contextual information is sent to an input processing interface which then publishes it to a KAFKA topic. For this contextual information transmission to be controlled and secure, the transmitting device must be enrolled in WAM and associated with a user, and it must have received a valid access token which allows it getting its owner's userid to be associated with the contextual information. Each device is associated with a risk factor which can be used to modulate the user's risk score.

7.4.5 Integrating the Context-aware Access Control Tool

- **Device enrolment**

  The device enrolment procedure allows a device to be associated with the identity of its owner. The Access Control tool leverages on the OAuth 2.0 Device Flow protocol to achieve this.

  The only requirements to use this flow are that the device is connected to the Internet and able to make outbound HTTPS requests, and that it is

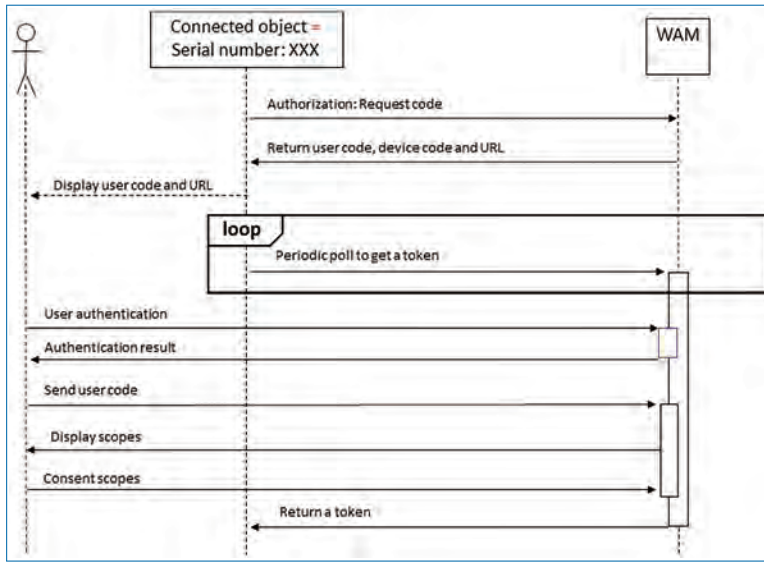

Figure 7.19. Device enrolment sequence diagram.

able to display or otherwise communicate a URI and code sequence to the user, and that the user (device owner) has a secondary device (e.g., personal computer or smartphone) from which to process the request. There is no requirement for two-way communication between the OAuth client (i.e., the connected device) and the end user's user-agent, enabling a broad range of use-cases.

During this procedure, the user gives his consent to the device to access data scopes on static attributes (username, email, etc.) and also a dynamic attribute (a risk level computed from contextual information on the user). At the end of the enrolment phase, the device receives an access token. The device has now access to the device owner profile that includes static attributes (username, email, etc.) but also the dynamic risk level.

The sequence diagram for this device enrolment procedure is described in Figure 7.19.

- **Context-aware Access Control with WAM used as reverse-proxy**

In this case, WAM is used as a Reverse proxy to protect the back-end servers. Additionally, WAM checks the token of the incoming request to verify if the device is authorized to access the back-end server. If this is the case, WAM injects in the header of the initial request the consent scopes of the device owner. This injection does not modify the request and the scopes injected contain some information about the device owner (username, email, etc..) and a risk level computed from contextual information. This allows the

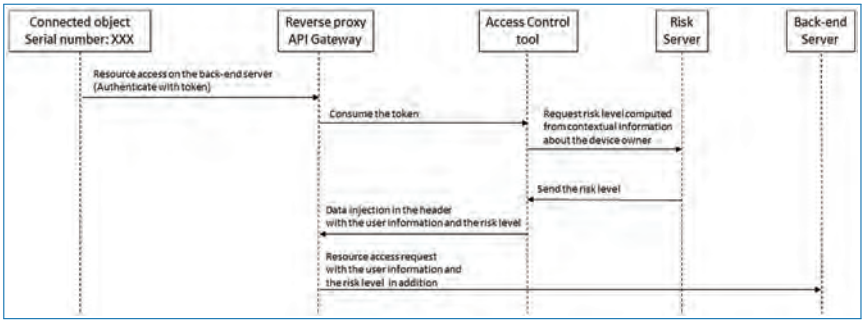

Figure 7.20. Context-aware Access Control with WAM as reverse-proxy — Sequence diagram.

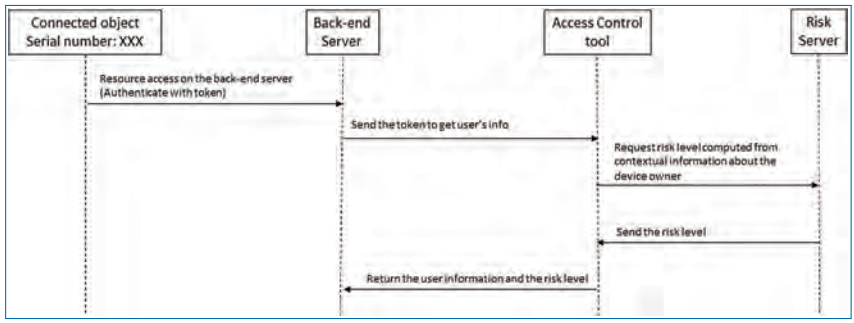

Figure 7.21. Context-aware Access Control with WAM as OpenID Connect IDP — Sequence diagram.

back-end server to make the link between the requesting device and the user associated with it, and to know the risk level depending on the context.

The sequence diagram for this working mode is described in Figure 7.20.

- **Context-aware Access Control with WAM used as OpenID Connect IDP**

In this case, WAM is used as an OpenID Connect IDP. WAM handles the access control part by checking if the token sent to a back-end server is valid. If this is the case, WAM responds with the consent scopes of the device owner. These scopes contain some information about the user (username, email, etc.) and the contextual risk level.

In the case where the authorization policy is not respected, the Access Control tool will inform the back-end server that it has to reject the request made by the device.

The sequence diagram for this working mode is described in Figure 7.21.

The device uses its token to access the back-end server (for example to push some data). The back-end server checks the validity of the token and retrieves the device owner's consent scopes for this token by calling the user-info endpoint of WAM (the userinfo endpoint from the Access Control tool

API consumes a token to retrieve information on the user). WAM returns the user information (for instance: username, email, address) and the dynamic contextual risk level associated to the user. The back-end applications can use this additional information to perform special actions.

7.4.6 Main Innovation

The main innovation brought by the features offered by the Context-Aware Access Control Enabler can be summarised as follows:

- The solution provides one unique tool to control in the same way the access of all the IoT actors (end-users, services, devices, administrators) to the operated data and resources, for both IT and OT (operational technologies) domains.
- The solution adds dynamicity to the authorization decisions OAuth 2.0 produces, by injecting dynamic scopes in the standard device flow.
- This allows to exploit contextual risk levels as dynamic attributes in the authorization mechanisms.
- Accordingly, the provided authorizations can be adapted based on a risk level computed from contextual data on the user and his devices, which allows context-aware dynamic access control behaviors.

7.5 Conclusion

This chapter was dedicated to the Security and Privacy Monitoring and Control Enabler designed to be used at the Ops phase of the DevOps life-cycle of SIS to address the security aspects of trustworthy SIS operation. The enabler is an innovative solution that supports SIS operation with multi-source data capturing, advanced detection combining signature-based IDS and AI techniques, and comprehensive situational awareness through a rich multi-viewpoint dashboard. By using this enabler, it is possible to assemble all or only some of the components in the enabler architecture. This brings flexibility to the continuous monitoring since it is possible to tailor the enabler design to the particular needs of the SIS under study in terms of e.g., how many security agents are deployed and where, which security policies are used by the clients, which metrics are monitored, and the needed tailored alarms and data visualisations can be created ad hoc.

The continuous monitoring offered is holistic in the sense that it correlates data captured in the three main layers of the SIS: network, system and application layers. And this makes possible a high richness and accuracy of security incidents

and anomalies detection which leverages signature-based detection together with machine learning and deep learning-based detection.

The enabler also answers to the needs of rapid elasticity and full scalability required by SIS that involve large amounts of sensors and actuators, while it still is able to offer the required visualisations and notifications that constitute the basis for the informed situational awareness of the overall system.

In order to be able to take advantage of the insights gained by the tool over the SIS, the enabler was designed with a security event bus for integration with other cybersecurity threat intelligence platforms and services, such as those of forensics analysis and cybersecurity information sharing with third parties.

Last but not least, the enabler design permits a seamless integration with controls at application layer, for example those developed on top of the SOFIA SMOOL IoT platform monitoring and control agents' management, which are able to monitor and control secure communications among the smart things of the SIS.

As part of the future lines of work in the enabler, the automatic reaction capabilities will be researched and enriched by extending the controls with intelligent security orchestrators and decision making support that facilitate the combination of multiple reaction measures when needed in the different layers of the SIS, so as the security level of the system is increased in the face of attack or incident symptoms.

References

[1] Rego *et al. SMOOL source code*. https://bitbucket.org/jasonjxm/smool, 2011–2020.

[2] Elasticsearch B.V. *The Elastic Stack. Retrieved March 11, 2021*, 2021. URL: https://www.elastic.co/elastic-stack.

[3] Prosunjit Biswas, Ravi Sandhu, and Ram Krishnan. *Attribute Transformation for Attribute-Based Access Control*. https://profsandhu.com/confrnc/misconf/abac17-prosun.pdf, 2017.

[4] Apache Software Foundation. *Apache Kafka. Retrieved March 11, 2021*, 2017. URL: https://kafka.apache.org.

[5] Dick Hardt. *The OAuth 2.0 Authorization Framework*. 2012. URL: https://tools.ietf.org/html/rfc6749.

[6] Wazir Zada Khan *et al.* "Industrial internet of things: Recent advances, enabling technologies and open challenges". In: *Computers & Electrical Engineering* 81 (2020), p. 106522.

[7] NIST. *NIST Computer Security Resource Center Glossary. National Institute of Standards and Technology. Retrieved March 11, 2021*. 2020. URL: URL=%20 https://csrc.nist.gov/glossary/term/information_security_continuous_monitoring.

[8] NIST. *NIST SP 800-53 Rev. 5. (December 2020). Security and Privacy Controls for Information Systems and Organizations. National Institute of Standards and Technology. Retrieved March 11, 2021*. 2020. URL: https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final.

[9] Adrian Noguero, Angel Rego, and Stefan Schuster. "Towards a Smart Applications Development Framework". *Social Media and Publicity* 27 (2014). URL: https://bitbucket.org/jasonjxm/smool,%202011-2020.

[10] OASIS. *eXtensible Access Control Markup Language (XACML) Version 3.0. Retrieved July, 2019*. 2019. URL: http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html.

[11] OpenID. *OpenID Connect*. 2014. URL: https://openid.net/connect/.

[12] Inger Anne Tøndel, Maria B. Line, and Martin Gilje Jaatun. "Information security incident management: Current practice as reported in the literature". In: *Computers & Security* 45 (2014), pp. 42–57.

[13] The Board of Trustees of the Leland Stanford Junior University. *Protégé, open-source ontology editor and framework.* 2020. URL: https://protege.stanford.edu/.

DOI: 10.1561/9781680838251.ch8

Chapter 8

Validation, Verification and Root-Cause Analysis

By Luong Nguyen, Vinh Hoa La, Wissam Mallouli and Edgardo Montes de Oca

8.1 Motivation

Verification and validation are two significant software development processes for checking that software meets its requirements and specifications and fulfills its intended purpose. In these processes, various test cases (e.g., unit tests, integration tests, regression tests, system tests) need to be designed and executed in a production-like environment that reproduces the same conditions where the software under test would run. However, having access to such an environment is usually tricky or close to being an impossible task. It is even particularly challenging in the IoT arena. The access to IoT devices might be nontrivial or limited due to many factors. Networks of physically deployed devices are typically devoted to production software. Testing applications on top of those networks might involve additional testing software, which might affect overall performance and the revenue generated by the devices (e.g., applications need to be stopped to load their new versions).

Software simulators proved to be valuable in easing the verification of the software requirements. They provide software developers a testing environment to at

least manage the execution of test cases. IoT Testbeds play a similar role in testing IoT applications. They offer a deployed network of IoT devices where developers can upload their applications and test their software in a physical environment. IoT-Lab [1] and SmartSantander [9] are good examples of IoT testbeds. Testbeds often have a predefined fixed-configuration and architecture. They are also usually shared with other users, which can be a problem for measuring application quality. Hence, this problem might make simulators more attractive since they provide a more customized and controlled environment. Furthermore, simulators avoid the need for a more expensive physical network of devices.

In recent years, both academia and the commercial market have proposed solutions for the IoT simulation field. These solutions are often entirely different, although their objectives are similar. The academic solutions implement cutting-edge technology as proofs-of-concept, and they are usually not ready for production systems. By contrast, the commercial solutions are designed to be stable and flawless, even though the technology behind them might not be state-of-the-art.

The ENACT project has brought an opportunity to create the Test and Simulation (TaS) tool. Collaborating with universities and research institutions such as SINTEF and CNRS, we provide a state-of-the-art test and simulation tool with cutting-edge technology behind it. We have evaluated our solution with several industrial use cases, such as eHealth (Tellu), Smart Building (Tecnalia), and Intelligent Train System (Indra). The case studies have shown that it is stable and ready for production systems.

The TaS provides the possibility to test the IoT system based on test scenarios using pre-prepared datasets. The datasets can be the recorded data from a real system or the data generated using some data mutation operators. The TaS also allows stressing the boundaries of the scenarios to detect potential problems.

We focus on the network of sensors and the applications on top of them. Therefore, we do not consider the physical behavior of the sensors. We take it for granted that the sensors are reliable and correctly react to the physical changes (e.g., if the physical temperature rises 2 degrees, the sensor will immediately send a message with a 2 degree higher reading).

On the other hand, it is also important to note that failures usually propagate in complex systems through causal chains and produce evolving fingerprints of noisy symptoms. One of the first tasks to accomplish for an automated tool helping humans troubleshoot a system is to group events that are causally connected (and keep unrelated events separated). Achieving this is often not straightforward since components of a system can exhibit similar symptoms of two unrelated failures. We need a higher level of granularity in the monitoring indicators and a deeper analysis to distinguish two unrelated failures. Moreover, it is frequent that failures are recurrent. The system administrators, who have some experience dealing with

failures, can react more quickly and efficiently against their recurrence. They can take the impact estimation and the mitigation action (e.g., reset a particular server every night) promptly.

Indeed, all aforementioned points lead to the need for a Root-Cause Analysis (RCA) tool which enables systematizing the experience in dealing with faults and problems to identify the root cause of a newly detected issue. Thanks to RCA results, remediation actions and reactions could be timely and wisely taken to prevent or mitigate the damage of the recurrence of problems.

The IoT world has promised to connect everything and create systems with an enormous number of devices. The need for RCA to implement and operate IoT systems is evident; IoT represents a generic framework that an RCA solution can target. However, several characteristics of these types of systems need to be considered: First, IoT networks are often very dynamic environments, with devices frequently joining and leaving a system (e.g., mobile devices connecting to a particular antenna). Nevertheless, most of the communications are likely to be wireless. This can introduce a higher degree of unreliability. The failure, however, can present symptoms very similar to a normal activity. For example, when we no longer receive sensed data from a sensor, it is difficult to determine whether the sensor is no longer in range or the communication has failed. Second, in many cases, the number of components/ indicators to be taken into the analysis could be enormous. This can lead to a big volume of data processed. Reducing the data dimension by avoiding less relevant attributes (i.e., noises) is a natural need. Finally, battery-powered devices may have a low-activity mode to extend their operation autonomy. In this mode, inputs may not be synchronized and have the same frequency as other information RCA uses for the diagnosis. Therefore, RCA must be able to deal with out-of-order data. In the context of ENACT, our RCA enabler would try to address all the challenges we mentioned above.

In summary, this chapter focuses on TaS and RCA, two primary parts empowering the validation and verification in an IoT DevOps cycle, which have been developed and evaluated in the context of the ENACT project. To the best of our knowledge, no similar tool had ever been created for IoT. On the one hand, the TaS tool enables the simulation and testing of an IoT system. It collects the events of a running IoT system without impacting its normal behavior. The recorded events can be used to simulate the system, inject different kinds of "problems", and collect all relevant data for detecting errors, failures, and unwanted symptoms. On the other hand, the RCA tool monitors the real system and performs the diagnostic analysis when some errors or failures occur. The enabler allows determining unknown incidents' symptoms and evaluating how much the unknown incident is similar to a known/learned one. We discuss more technical details regarding TaS and RCA in the following sections.

8.2 Test and Simulation (TaS)

In this section, we first present an overview of the TaS enabler. We then give the details of the enabler.

8.2.1 Overview and Approach

The TaS enabler is a test and simulation solution well adapted to IoT environments. It allows simulating different IoT topologies and performing various tests to detect potential errors and security failures. We first present the main features and components of the enabler that simulate an IoT system. Then, we give a detailed description of the TaS enabler architecture.

8.2.1.1 Smart IoT system components

Figure 8.1 shows some main components in a Smart IoT System:

- **The Sensor Node:** captures, pre-process, and sends the sensor data to the gateway that can be a Raspberry PI, an Arduino, etc. It implements some basic modules:
  - The Sensor module captures the environment information.
  - The Onboard Processing module reads the sensor data and pre-processes it (e.g., performs calculations and formalizes and validates data). It can be an IoT application, a Node-RED flow, etc.
  - The Communication module component communicates with the gateway to send or broadcast the processed data.
- **The Gateway device:** receives data from the sensor nodes and processes or just forwards it to the other components/services such as cloud-based application and control center.
- **The Actuator node:** reacts and controls the actuator based on the reactions of the IoT system. It contains some basic modules:
  - The Actuator module triggers a change on the IoT device, such as opening a door and activating an alarm system, etc.
  - The Onboard processing module reads the actuator data signal and converts it into an action.
  - The Communication module communicates with the gateway to receive the actuation data signal.
- **Other components:** Other components are higher-level components that can provide a service or an application that receives and processes the data and performs actions depending on the business logic.

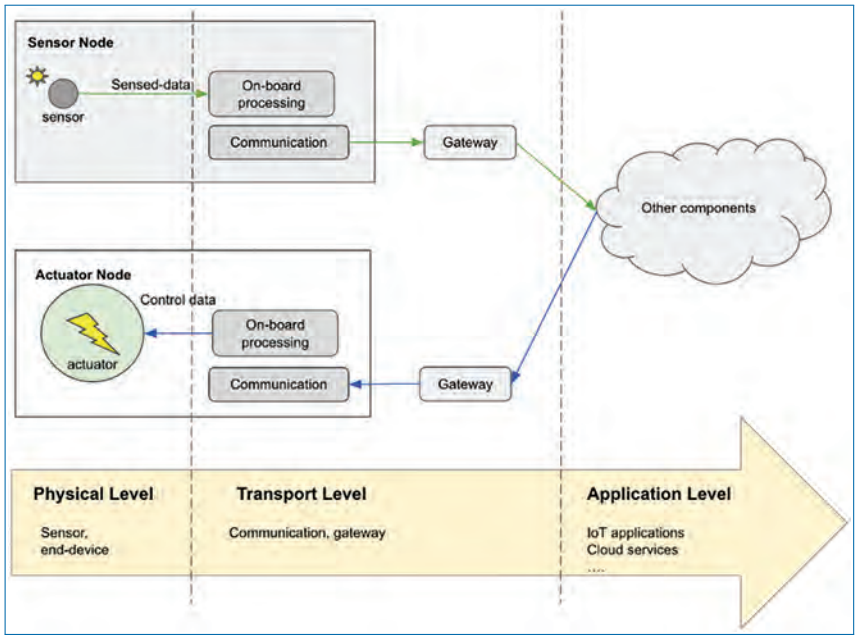

Figure 8.1. IoT system components.

The SIS components can be organized in a multi-layer architecture which we present in the next section.

8.2.1.2 Simulating a smart information system (SIS)

An SIS can be represented in three levels depicted in Figure 8.1. The physical level contains all the physical components, such as sensors and actuators, produced by a manufacturer and cannot be changed by developers. The transport level is responsible for transmitting the data within the SIS network. Developers can configure the transport level to use a specific port number or protocol. Finally, the highest level is the application level, which contains the application written by developers. The application receives the data from sensors, processes it, and produces an action to be performed by the actuators (e.g., turn the light on or off). Software engineers usually work on the application level.

When it comes to developing a software application, a software application needs to be tested every time there is a change in its source code or in the infrastructure it uses. The planned tests aim to cover many scopes involving different testing scenarios. While coping with multiple test scenarios, the testing environment needs to be flexible for manipulating input and measuring output. It is not easy to have such a testing environment for IoT applications since sensors at the physical level depend on the physical environment. Therefore, only the scenarios matching the current condition of the environment can take place.

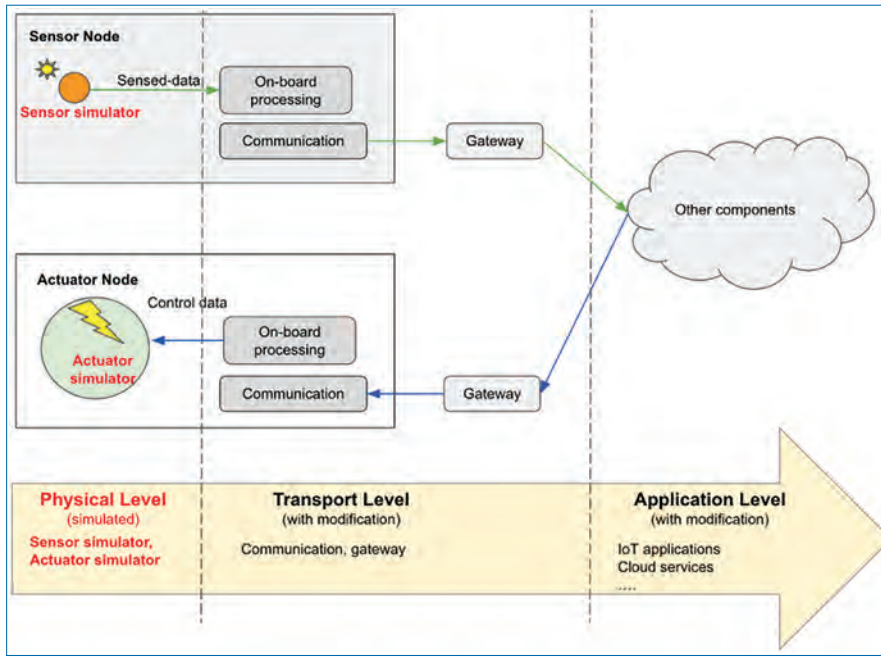

Figure 8.2. An IoT network architecture with simulated components.

Within an IoT network, a sensor captures the information of its surrounding environment at a specific time. This information is transformed into a digital format. Since it is not reasonable to wait for the change in the environment to test IoT applications, simulating various sensor measurements is very beneficial for testing them. It allows the developer to control sensor values and, thus, to simulate and test the IoT application in all scenarios without waiting for environment changes.

An actuator presents the SIS reactions in a specific situation, for example, switching on a light bulb. In such cases of testing system reactions, it is sufficient to measure the actuated data sent by the SIS to actuators. In the TaS enabler, the actuator is simulated by simply creating a hub to receive the actuated data instead of using a real actuator. Note that the impact of an actuator on a sensor is not yet considered.

With the TaS enabler, we only need to simulate the components at the physical level. The other (software) components can be cloned from the system under test and configured to work in a classical test and simulation environment, avoiding the need for communicating with a production environment, as we can see in Figure 8.2.

8.2.1.3 The TaS enabler's global approach and architecture

In this subsection, we present the architecture of the TaS enabler, which is based on the concept of Digital Twins [3]. Figure 8.3 illustrates the TaS enabler architecture.

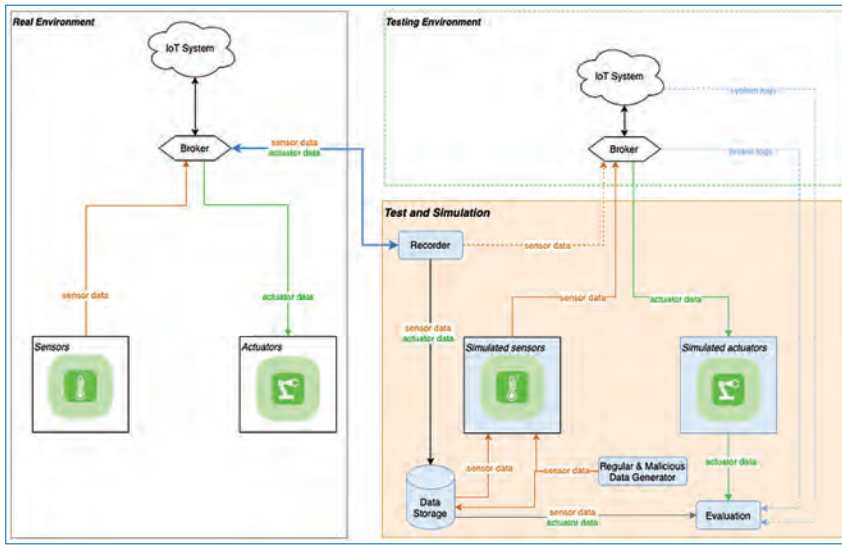

Figure 8.3. Test and Simulation (TaS) Enabler approach and architecture.

On the left-hand side, we have the system in a real (production) environment. The communication between the sensors, actuators with the IoT component is typically done via a broker. The sensors capture and send the surrounding information (e.g., temperature) to the IoT system. Based on input data, the IoT system reacts differently and sends actuation data to change the actuator settings (e.g., "change the heating level").

On the right-hand side of the figure, we have the SIS in a test environment and the TaS enabler. The system under test is the SIS that needs to be tested. The TaS enabler simulates sensors and actuators. The topology on the left side is very similar to the topology on the right side. The only difference is the simulated sensors and actuators. The simulated actuators collect the actuation data sent from the IoT system. The simulated sensors play the same role as the physical sensors providing the data signal to the IoT components. However, they are much more valuable than a physical sensor in terms of testing in the following ways:

- Firstly, by using the dataset recorded from the physical environment, the simulated sensors can repeatedly simulate the surrounding environment at a specific time. In reality, an event may happen only once, but the simulated sensor can generate the same event as many times as needed for testing purposes.
- Secondly, the physical sensors passively capture the state of the surrounding environment. It can be challenging to obtain different data from the physical sensors. In contrast, the simulated sensors use the dataset in the Data Storage as a data source. Therefore, we can generate various testing scenarios by modifying the event in the Data Storage.

- Moreover, the TaS enabler also provides a module to manipulate the data from the sensors. The Regular and Malicious Data Generator can generate regular data to test the functionalities, operations, performance, and scalability. It can also generate malicious data to test the resiliency of the system to attacks.

Besides the simulated sensors and actuators, the TaS enabler also provides some modules which support the testing process 8.3. The *Data Recorder* module records all the messages going through the broker in the physical environment. Each message can be considered as an event happening in the physical environment. Then, the recorded messages are forwarded to the broker in the testing environment. In this way, we have a "twin version" of the physical environment. What has happened in the physical environment is reproduced in the testing environment. Besides, the recorded messages are stored in a *Data Storage* as a dataset for later testing. The recorded dataset can be modified (muted) to create a new dataset, e.g., "change the event order", "delete an event", "add a new event". All the testing datasets are stored in the Data Storage. The *Regular and Malicious Data Generator* enables the simulation of different sensor behaviors, from normal behavior to abnormal behavior, such as a DOS attack (the sensor publishes massive data messages in a short time), node failure (the sensor stops sending data). With data mutation, the TaS enabler can help build datasets for testing many different cases hard to produce in real life. Finally, the *Evaluation* module analyses the simulation input and output and combines them with the logs collected from the IoT system to provide the final result of a testing process.

The next section presents more details on how the TaS enabler simulates an SIS.

8.2.2 Simulation of a Smart IoT System

Most of the testing scenarios are defined by the information about the surrounding environment captured by sensors. The following section goes into detail about the simulation of sensors.

8.2.2.1 The simulation of sensor

The sensor provides the input data of an IoT system. The simulation of a sensor corresponds to the simulation of the data stream it provides. The simulated sensor has been designed for flexibility in the following ways:

- It supports different types of data report formats:
  - PLAIN_DATA: the measurement value is published directly without any transformation, it can be a number, a string or an object, for example: 15

- JSON_OBJECT: the measurement value is transformed to be an object in JSON format, with the key is set by user, for example: "temp":15
- IPSO_FORMAT: the reported data follows the Internet Protocol for Smart Object (Object and Resource Registry). A temperature sensor can report the data in IPSO format as follows (Temperature Sensor in IPSO):

```
{
    "InstanceId": 5,
    "ObjectId": 3303,
    "TimeStamp": 1601498832,
    "TimeAccuracy": 364449977,
    "Resources": {
        "5700": 15,
        "5701": "celcius"
    }
}
```

- It supports different data sources which are used for simulation:
  - Dataset: The data source is from the data storage where the data has been recorded or created before simulating.
  - Data Generator: The data will be generated at run-time during the simulation.
  - Data Recorder: The data source is the data recorded from a real system and forwarded to the testing system.
- It supports simulating several abnormal behaviours, such as, low energy, node failure, DOS attack, and slow DOS attack.
- It supports multiple measurements with the different data types, such as Boolean, Integer, Float and Enum. For each measurement, there are several abnormal behaviours that can be selected, such as “fixed value”, “value out of range”, and “invalid value”.

Figure 8.4 presents the definition of a temperature sensor, which generates (data source: *DATA_SOURCE_GENERATOR*) a measurement value every 5 seconds. The measurement value is published in *PLAIN_DATA* format to a MQTT/MQTTS message bus communication channel defined by the topic *enact/sensors/temp-01*. The sensor does not have any abnormal behaviour.

8.2.2.2 The simulation of actuator

An actuator can be considered as a device that receives the IoT system reaction based on the input data. We simulate the actuator as a component that will receive the reaction signal (actuation data) from the IoT system. Figure 8.5 shows the configuration of a Heater. The actuator listens for the actuation data on an MQTT/MQTTS message bus communication channel defined by the topic:

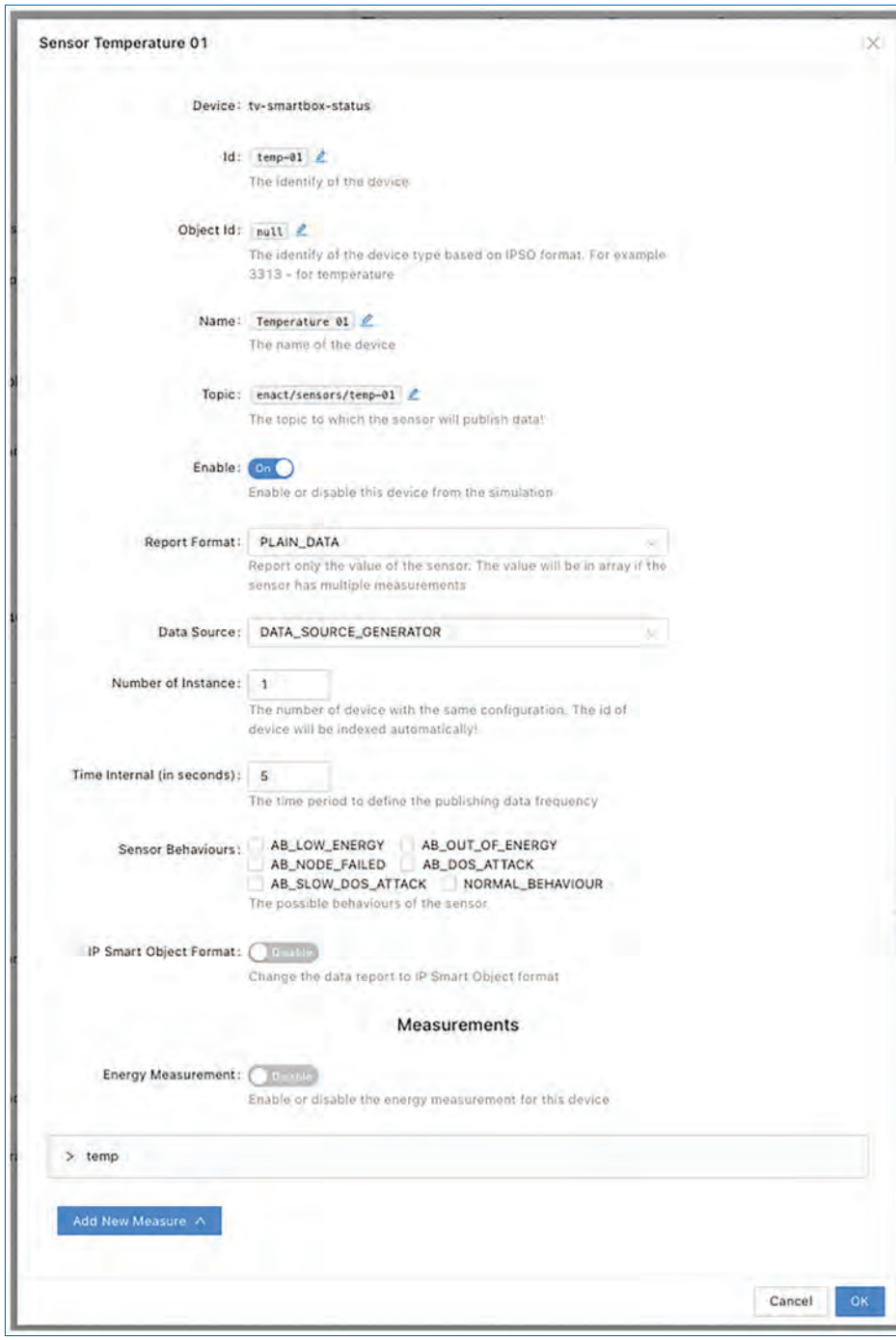

Figure 8.4. A temperature sensor.

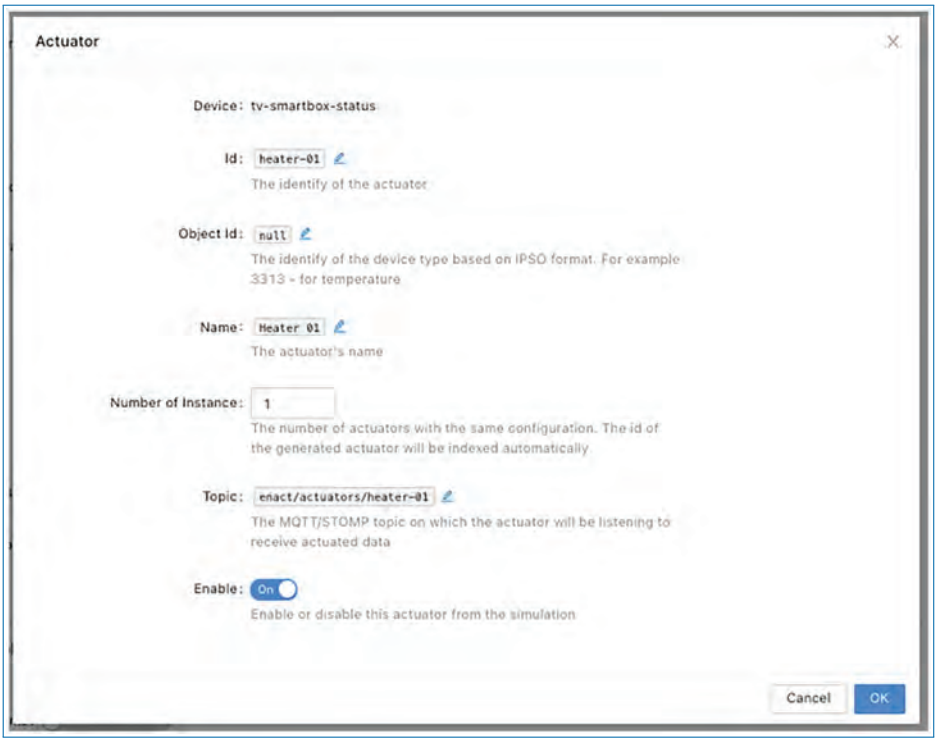

Figure 8.5. A Heater actuator.

enact/actuators/heater-01. The topic defines the channel on which the actuator will connect to obtain the actuation data.

8.2.2.3 The simulation of an IoT device

In an IoT system, the sensor and actuator are usually part of the same device. An IoT device can contain one to many sensors as well as one to many actuators. Figure 8.6 illustrates the configuration of a Heating System Control device. The device has one sensor and one actuator. The data is published by the sensor and received by the actuator via the MQTT protocol.

8.2.2.4 The simulation of a network topology

Figure 8.7 presents a simple simulated network topology.

A list of simulated IoT devices forms the simulated network topology. Besides the list of devices, a network topology can also provide the identifier of the dataset (*datasetId*),which contains the data to simulate the SIS in a given time, the global replaying options, the configuration to connect with the database, and

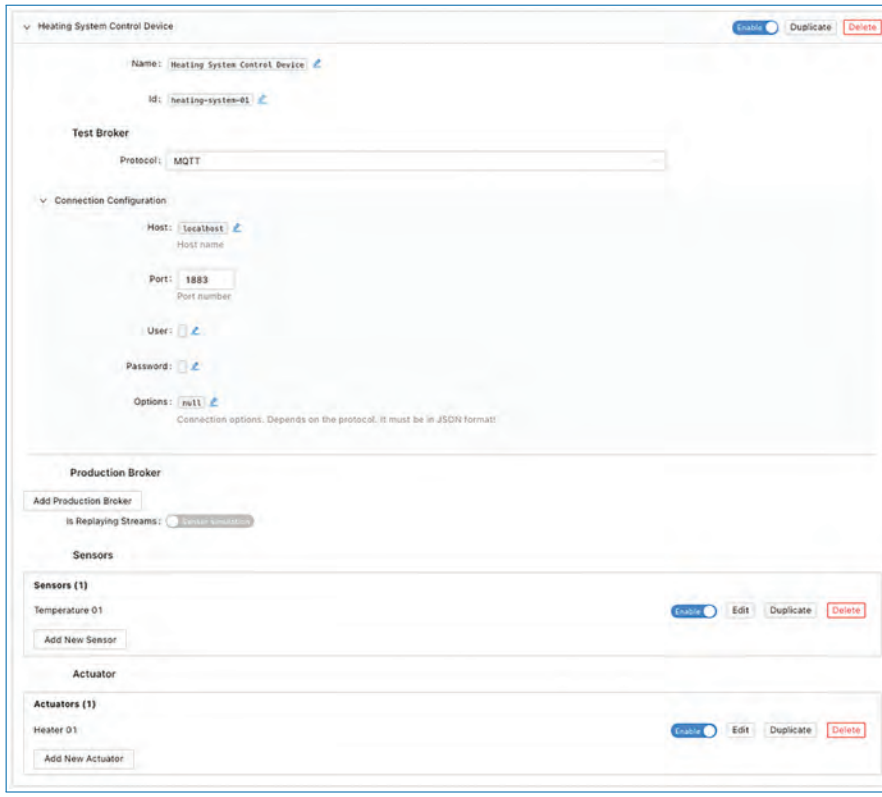

Figure 8.6. A Heating System Control device.

the definition of the new dataset where the data generated from the simulations will be stored.

8.2.2.5 The communication between the TaS enabler and the system under test

In the ENACT project, the communication between the TaS enabler and the system under test has been implemented based on message queue protocols such as MQTT and MQTTS. Figure 8.8 presents the Message Bus class diagram, similar to the interface for all Message Queue Protocols.

Some basic message queue bus protocol methods have been implemented, such as subscribe, unsubscribe, publish, connect, and close. By design, each IoT device can have its way of communication with the SIS systems.

8.2.3 The Testing of a SIS

In this section, we present the testing methodologies and techniques we have adapted in the TaS enabler.

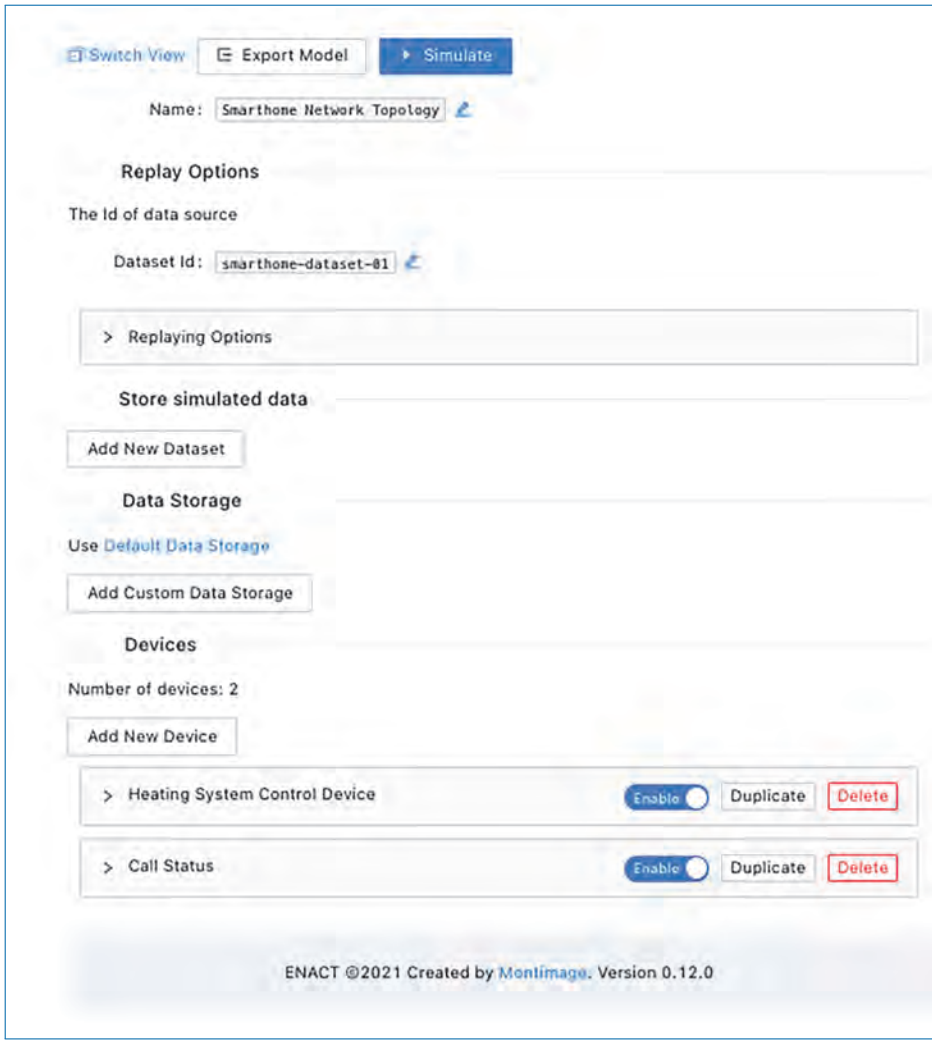

Figure 8.7. Smarthome network topology.

8.2.3.1 The testing methodologies

This section covers the testing methodologies that the TaS enabler can support. In this first version of the enabler, we have implemented only data-driven and data-mutation testing methodologies. The other ones described below are possible future extensions.

Data Driven Testing

Figure 8.9 presents the data flow of the Data-Driven Testing method. The Data Storage contains the datasets recorded from the IoT system or entered manually. Each dataset contains sensor data (inputs for TaS) and expected actuator outputs.

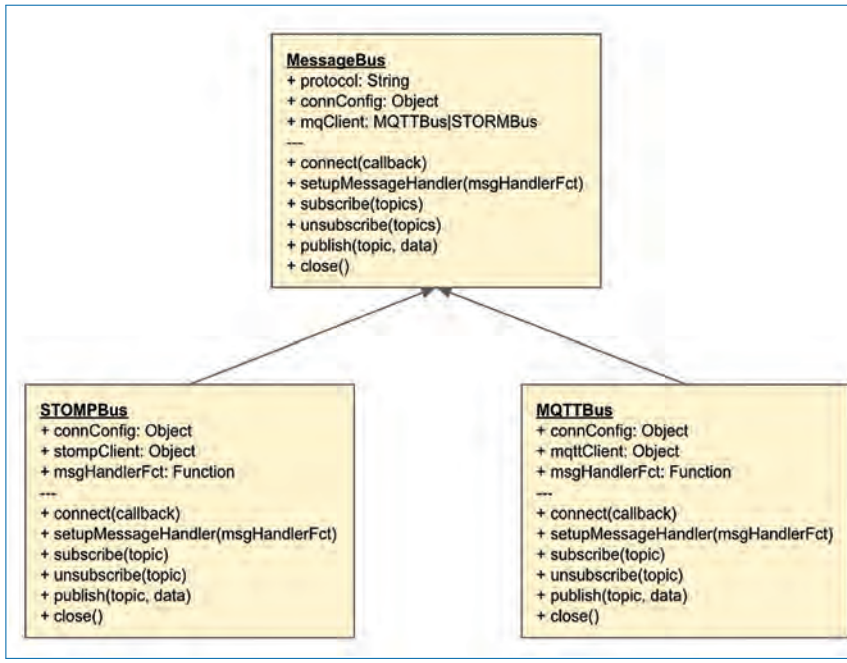

Figure 8.8. Message queue bus class diagram.

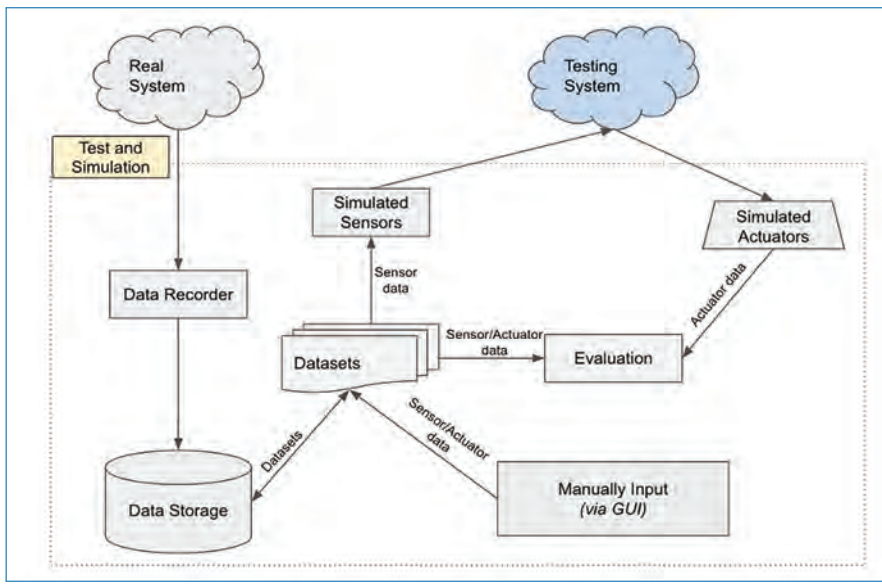

Figure 8.9. Data Driven Testing.

The expected actuator outputs can be the value recorded from the IoT system in a normal scenario. Engineers can also enter them manually via the Graphical Interface. The Evaluation module will use the expected outputs to compare them with the simulation output to determine if they match. A test case passes if the simulation output is the same as the expected output. The Data-Driven Testing method is suitable for functional and regression testing.

The Data-Driven Testing has been implemented as the main testing methodology of the TaS enabler.

Data Mutation Testing

Figure 8.10 illustrates the Data Mutation Testing architecture. The Mutant Generator generates new sensor data from existing data stored in the Data Storage by applying one or many mutated functions, such as "change the event order", "change a value", and "delete an event". The mutated data are input for the simulation. The Evaluation module generates a report about the output differences when testing the system with the mutated and the original input data. The Data Mutation Testing method is for penetration, robustness, security, and scalability testing (e.g., mutating the device identifier to obtain new devices). In the TaS enabler, we can mutate the device identity to generate many devices while testing the system scalability. There is also an interface to apply some mutation functions to a dataset manually.

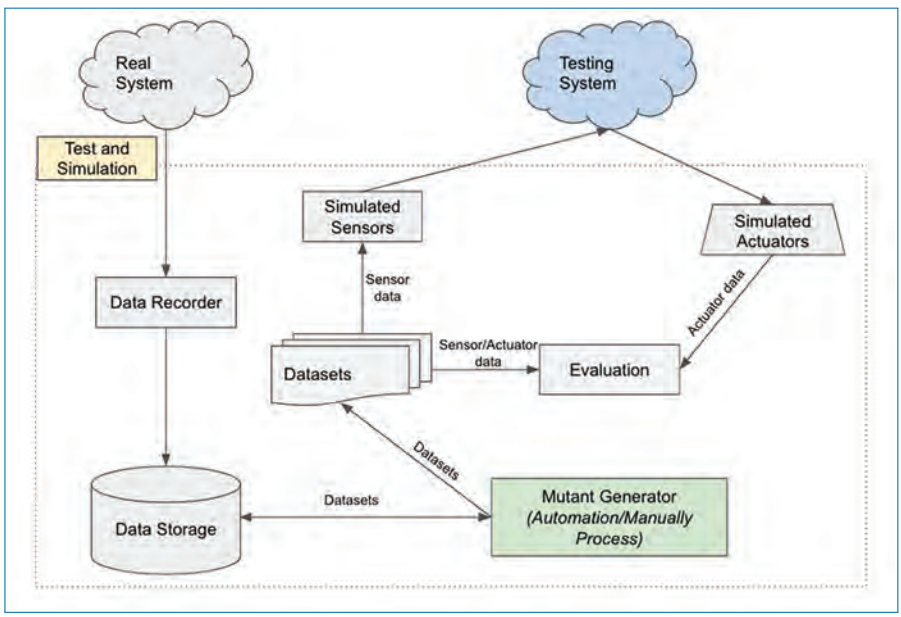

Figure 8.10. Data Mutation Testing.

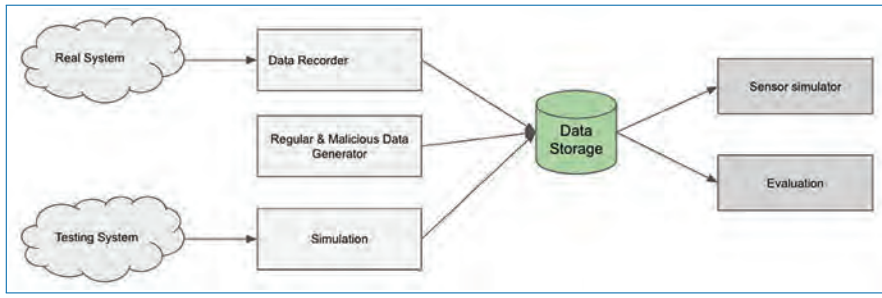

Figure 8.11. DataStorage.

Model-Based Testing [10] and Risk-Based Testing [6] are two other methodologies that we have studied but not yet implemented in the TaS enabler at the time of writing of this book chapter.

8.2.3.2 The testbeds

To simulate and test an IoT system in some specific scenarios, one of the easiest methods is to use a testbed. With testbeds, the developer can define exactly what is the input and what should be the corresponding output. This way, the tests can be done automatically and easily integrated in the DevOps Continuous Integration and Continuous Deployment processes. In TaS, testbeds are built from datasets which are recorded from a real system or generated by the TaS based on scenarios.

DataStorage

The Data Storage contains all the datasets for testing and simulating.

As depicted in Figure 8.11, the datasets are fed into the DataStorage via three sources: the data from the real system recorded by the Data Recorder, the data generated by the Regular and Malicious Data Generator, and the data generated by the simulation. The datasets in the Data Storage are used to simulate the sensors and to validate the simulation output.

The database connection of the TaS enabler is flexible. Two simulations can use different databases. If there is no configuration specified, the TaS enabler uses a default database. The database to connect to can be any database that the TaS enabler can reach.

Event

An event represents a message sent through the communication channels. It can be a data message sent by a sensor or data received by an actuator. Figure 8.12 presents the format of the event Schema.

The *timestamp* attribute indicates the time when the event has been captured. The *topic* represents the MQTT/MQTTS bus channel related to the event. It is the

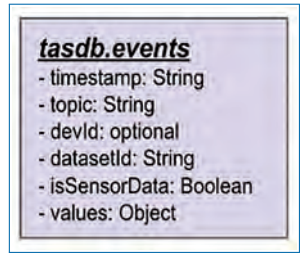

Figure 8.12. Event Schema.

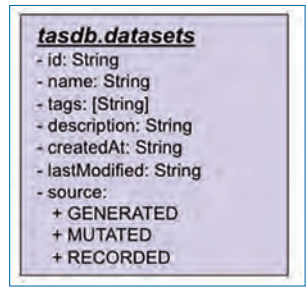

Figure 8.13. Dataset Schema.

source channel of the event where the data message corresponds to sensor data. It is the destination channel of the event where the data message is data received by an actuator. Its value is crucial for identifying the event to be replayed. The *datasetId* attribute represents the dataset to which the event belongs. The *isSensorData* is set to True if the event presents a data message sent by a sensor. It is False if the event is a data message received by an actuator. The *values* attribute contains the value of the message data. This value can be a number, a string, or an object. This design helps make the event generic and making it possible to consider any message data type.

Dataset

A dataset contains a series of events for a specific scenario. Figure 8.13 presents the schema of a dataset. Each dataset has a unique *id*, *name*, and *description* to describe the dataset objective. The *source* attribute indicates the dataset source. A dataset can be created from a recording session by the Data Recorder (source: RECORDED) or generated by the Regular and Malicious Data Generator (source: GENERATED). A dataset can also be derived by cloning and modifying data from another dataset (source: MUTATED).

By grouping the events by the dataset Id, we have all the events belonging to a dataset.

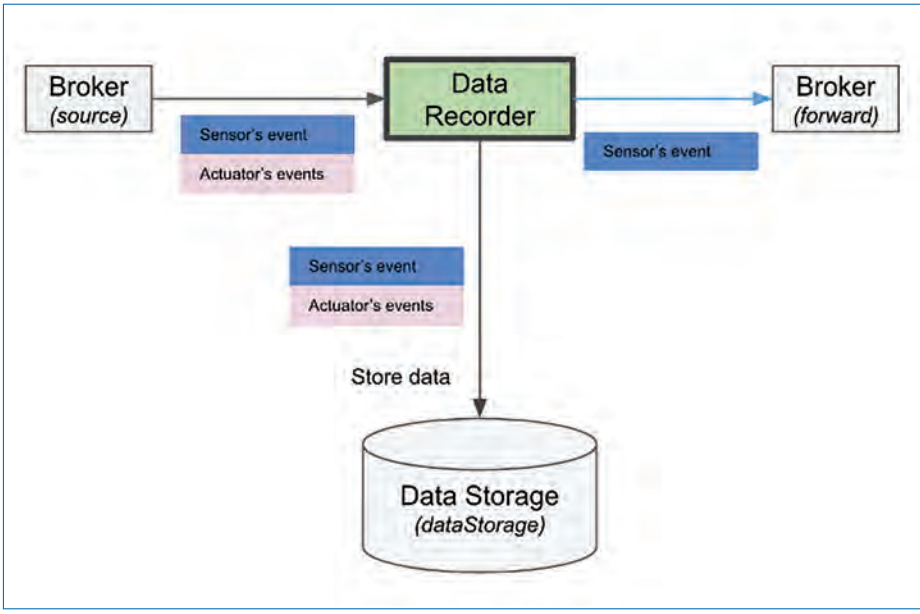

Figure 8.14. Data Recorder data flow.

8.2.3.3 The data recorder and digital twins concept

The TaS enabler provides the possibility to simulate an IoT system using historical data. To this end, a Data Recorder module is needed.

Figure 8.14 presents the data flow of the Data Recorder. All the events in the real system (coming from the broker) will be recorded. This data (including both sensor and actuator data) is stored in the Data Storage as a dataset. The sensor data can be forwarded directly to the testing system (using the forwarding broker). With the recorded data from the real system, the SIS can be tested with real input. The more data from sensors are recorded, the more test scenarios are tested. By synchronizing the Sensor simulator timestamps with the Data Recorder, it is possible to simulate a particular SIS (following the Digital Twin concept). By monitoring the SIS input and output, we can build an automatic testing process for a complex IoT system.

The recorded data can be used as a source for simulation. It can also be mutated so that it can contain different values for obtaining a modified testing scenario. In the next section, we will explain how to generate a new dataset using a given behavior profile.

8.2.3.4 The regular and malicious data generator

When testing the IoT system, there are many testing scenarios and cases that do not frequently occur in reality. With the real IoT system, it is almost impossible to collect the datasets for many testing scenarios. The TaS enabler provides a powerful

Behaviour / Data Type	Boolean	Integer/Float	Integer / Float + Value Constraint	Enum	Composed
Fix value (*Always send the same value*)	Yes	Yes	Yes	Yes	Yes
Value out of range (*Send the value out of possible range*)	NA17	Yes	Yes	NA	*
Value out of regular range (*Send the value out of the regular range*)	NA	NA	Yes	NA	*
Value change out of regular step (*The data change step is out of the regular step*)	NA	NA	Yes	NA	*
Invalid value (*Send invalid value - attack to crash the system*)	Yes	Yes	Yes	Yes	Yes
Low battery (*Reduce the sending data frequency - 1/2*)	Yes	Yes	Yes	Yes	Yes
Run out of battery (*Stop sending data*)	Yes	Yes	Yes	Yes	Yes
Possible node failed (*Stop sending data after some period of time*)	Yes	Yes	Yes	Yes	Yes
Possible DOS attack (*Send data with the period less than the minimum time period*)	Yes	Yes	Yes	Yes	Yes
Possible Slow DOS attack (*Send data with the period more than the maximum time period*)	Yes	Yes	Yes	Yes	Yes

Figure 8.15. Abnormal behaviours based on the data type and the constraints.

tool to solve this problem. The Regular and Malicious Data Generator module helps developers create a testbed. It enables generating sensor data for various scenarios, e.g., making the temperature too high or too low. By combining multiple data, one can create a testbed that includes many incidents or attack scenarios, such as DDoS and data poisoning. The Data Storage stores all the generated data for further use. Based on the data type and constraints on the time, values, or energy use, many abnormal behavior types may exist as depicted in Figure 8.15.

The abnormal sensor behaviors are defined by energy, reporting time, and value constraints.

Figure 8.16 illustrates how a data value is generated based on the selected behaviours of the sensor. In the beginning, the energy constraint is checked. There are two behaviors related to energy. If the sensor is in the low-battery mode, we can reduce the reporting frequency. Notice that the user initially sets the frequency. If the sensor is out of battery, it stops sending data. In the next step, we consider the time constraint. We can select among three behaviors. The possible DOS attack

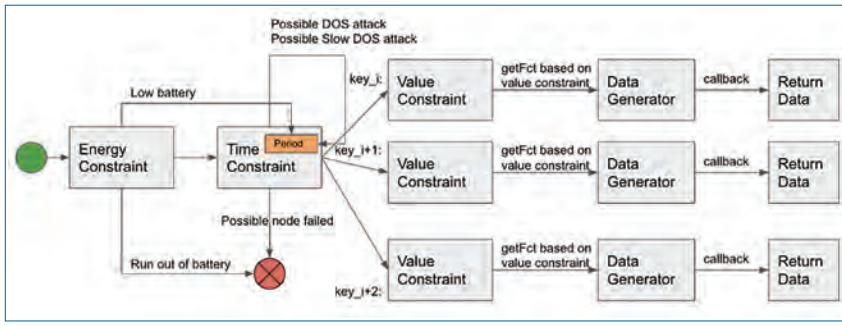

Figure 8.16. Data generating flow.

increases the reporting frequency to simulate a DOS attack, and the sensor sends a lot more data than what is considered normal. If there is a constraint on the maximum delay time for reporting data, we can simulate a behavior like a possible slow DOS attack. For example, if the system expects to receive the temperature information every 5 seconds maximum, then a slow DOS attack could change the sensor behavior that the sensor sends a message every 6 seconds. Based on the time constraint, we can simulate a sensor that stops sending data for a certain period (node failed). Finally, for each measurement provided by a sensor, the value constraint is checked. There are many behavior types based on the measured data type, such as invalid value or fixed value. Based on the selected behavior, the Data Generator function returns a specific value for the measurement.

Besides the sensor's behavior, it is also possible to change the behavior of the IoT devices. For example, the GATEWAY_DOWN behavior makes the simulated IoT device stop working after some time. When an IoT device stops working, all the sensors and actuators belonging to that device also stop working.

8.2.3.5 Automatic testing

The TaS enabler has been designed to be easily integrated into any Continuous Integration and Continuous Delivery processes. Figure 8.17 illustrates the TaS enabler concept. One of three events trigger the TaS process: code commit, new component (software module or hardware device) added, new scenario added. Several tests are executed by simulating the different testing scenarios on the system under test. The tests can cover functional, operational, security, performance, and scalability testing. If all the tests pass, we can deploy the new changes in the real environment.

Following the process we depict above, we can automatically test every change in the system and cover every test scenario.

Test case

While testing an IoT system, we may want to test different network topologies, such as adding a new device, removing a device, or just changing the way to

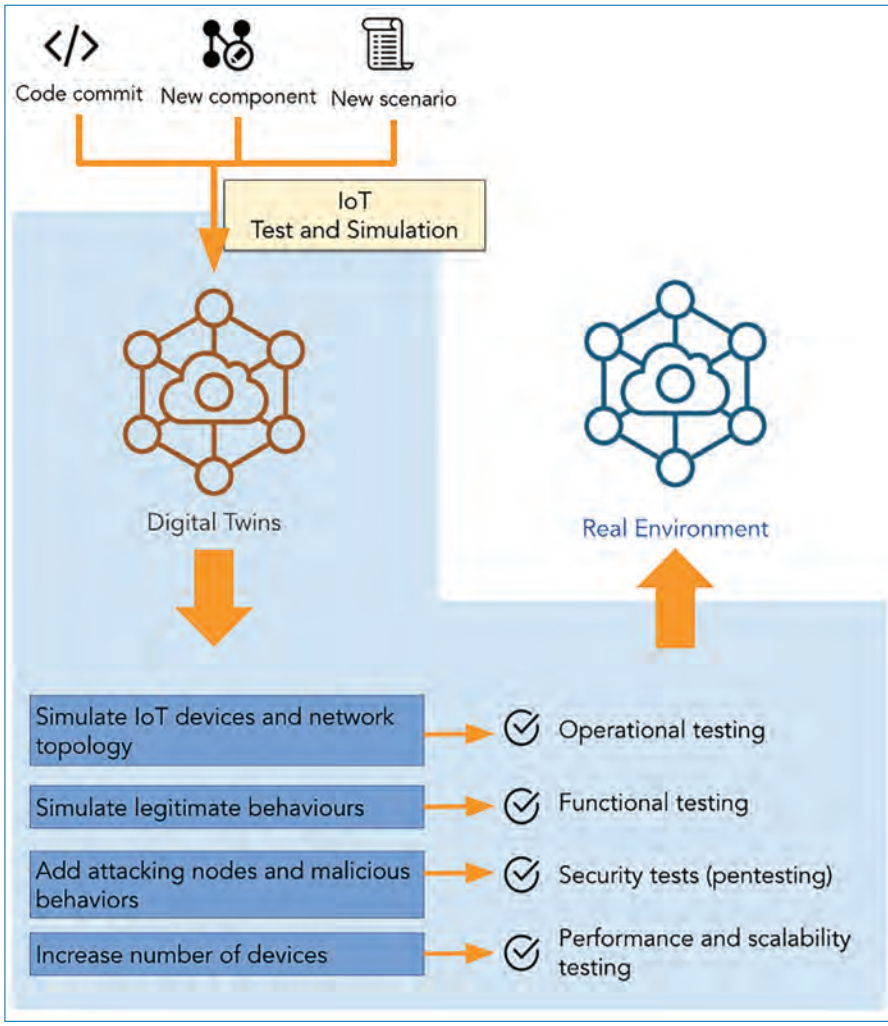

Figure 8.17. Test and Simulation (TaS) Workflow.

connect devices. A test case is a collection of tests executed on one network topology. There can be many different testing types (e.g., functional, security, or scalability testing). A dataset defines a test. For test execution, the TaS enabler runs the simulation and testing using each dataset by following the test order in the test list. We can change the order of the datasets via the web interface.

Test campaign and the integration into DevOps cycle

While the test case groups the test by the defined network topologies, the test campaign contains all the test cases that should be executed for each change in the IoT system. The test campaign is the global test that covers every testing scenario and testing aspect. The test campaigns are executed automatically every time there is

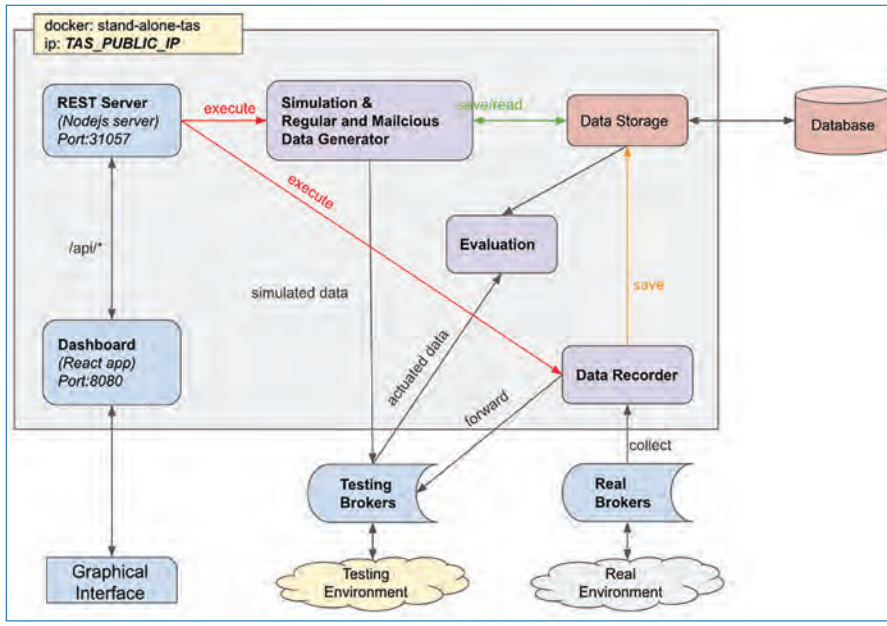

Figure 8.18. Test and Simulation (TaS) Enabler Docker image.

a change in the system. For each test campaign, the TaS enabler runs test cases according to the order in the list.

8.2.3.6 The evaluation module

The Evaluation module collects the simulated actuator data as well as the other metrics of the system. Then, it performs the evaluation based on the testing methodology (see Section 8.2.3.1 for more details).

The next section presents the implementation of the TaS enabler.

8.2.4 Implementation

8.2.4.1 The test and simulation (TaS) docker image

The TaS enabler has been designed to be portable. It can be installed as a Node.js application and packaged as a docker image. Figure 8.18 presents the communications between the modules inside a docker container and between the docker container and other modules.

The REST Server provides an API to interact with the tool. Via this API, we can execute the module Data Recorder, Simulation, and Regular and Malicious Data Generator. The Database is external to the docker container and can be connected via the Data Storage module. The dashboard is the graphical interface implemented using ReactJS [27].

Table 8.1. Basic APIs to integrate into a DevOps cycle.

Path	Method	Data	Response
/devops/	GET		Get automation testing configuration
/devops/	POST	{webhookURL, testCampaignId}	Update the automation testing configuration
/devops/start	GET		Trigger the simulation and testing process
/devops/stop	GET		Stop the simulation and testing process
/devops/status	GET		Get the status of the current execution

8.2.4.2 Basic APIs

Table 8.1 presents the list of basic APIs exposed by the tool for integration into a DevOps cycle.

8.2.5 Evaluation

The TaS enabler has been evaluated in several use cases in the ENACT project.

8.2.5.1 Itelligent train system

Figure 8.19 shows the data flow of the TaS enabler in the Intelligent Train System (ITS) use case. The Data Recorder records the WSN Coordinator data from broker-01 part of the ITS system. The recorded data is stored in the Data Storage. The Simulated Wire Sensor Network (WSN) Coordinator uses the recorded data to simulate a several WSN Coordinators for testing the scalability of the ITS system. All the simulated WSN Coordinators publish the data to broker-02 which is in the ITS system under test. By evaluating the gateway status, we can assess the scalability of the ITS system.

8.2.5.2 E-Health system

The TaS enabler is used in an e-Health use case to test if the parsing data function works correctly in the e-Health gateway. Figure 8.20 presents the data flow of the e-Health use case. The simulated sensors send some valid and invalid data messages to the internal broker, and then these messages are consumed by the CloudAgent. By mutating data messages, we can test the CloudAgent in various test scenarios, such as “invalid data format” and “invalid value”.

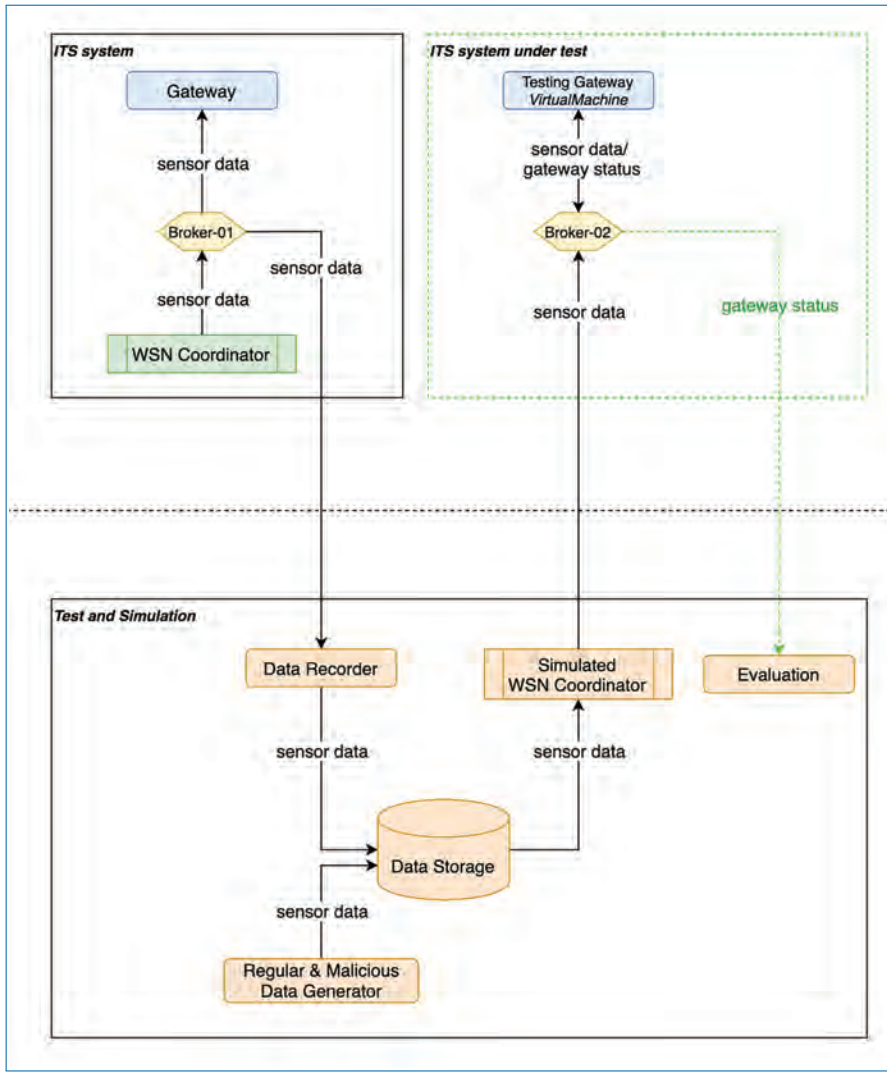

Figure 8.19. Intelligent Train System.

8.2.5.3 Smart home system

In the Smart Home use case, we used the TaS enabler as part of the DevOps cycle. Figure 8.21 shows the data flow of the TaS enabler. First, the Data Recorder records and builds the testing dataset from the real system. For each system change, such as new features, and software updates, the TaS enabler uses the recorded dataset to check the system reaction. By comparing the recorded system output with the simulated system output, we can detect miss behaviors in the updated system.

Using the TaS enabler, we can automatically test the SIS in various test scenarios. However, due to the SIS complexity, a problem may happen at any moment while

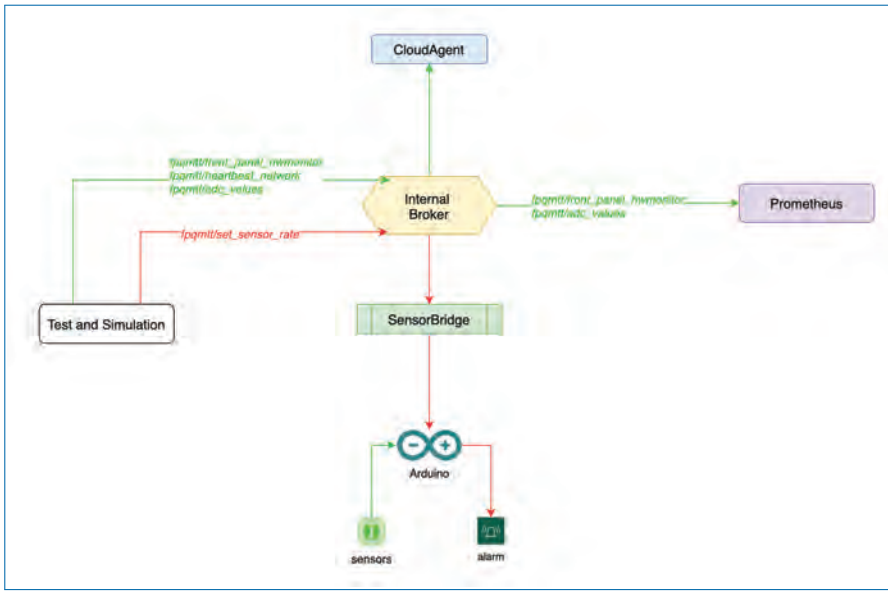

Figure 8.20. E-Health System.

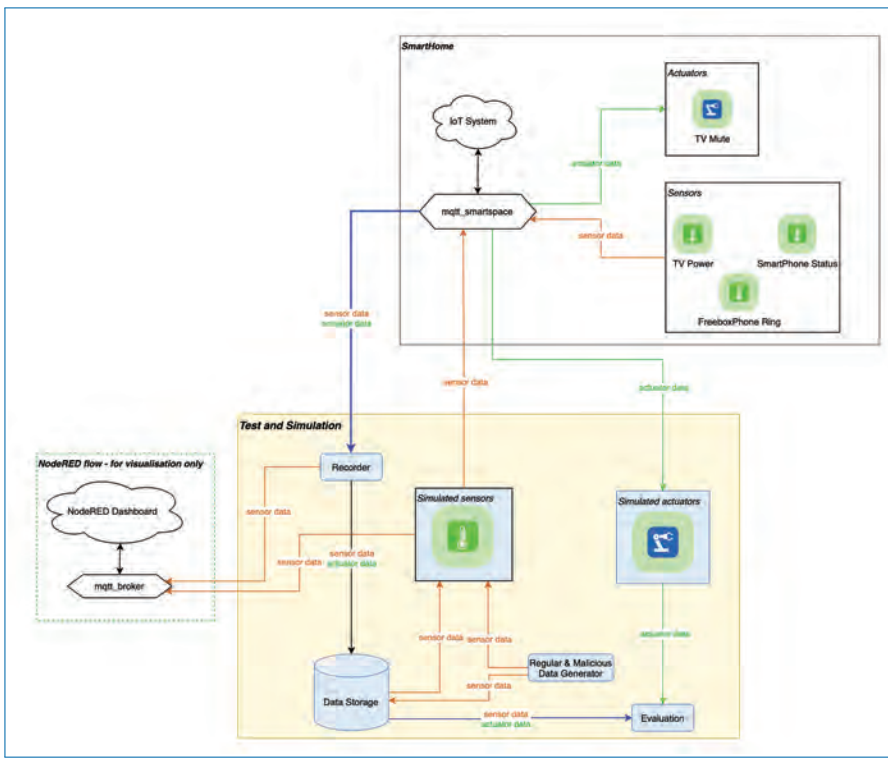

Figure 8.21. Smart Home System.

the system is running. Knowing the root cause of the problem is very important to find the solution. Therefore, we need a tool to identify the root cause of the problem while running an SIS.

8.3 Root-Cause Analysis (RCA)

Root Cause Analysis (RCA) is a systematic process for identifying "root causes" of problems or events and for responding to them. System administrators and DevOps engineers use RCA not only for detecting the problems but also for understanding their root-causes to prevent the recurrences and/or mitigate the impact. In the context of ENACT, the RCA enabler relies on Machine Learning algorithms to identify the most probable cause(s) of detected anomalies based on the knowledge of similarly observed ones. Figure 8.22 presents the high-level architecture of the implemented enabler.

The **data collector** allows gathering information from different sources (e.g., network, application, system, hardware) by relying on dedicated monitoring agents. It has a plugin architecture that enhances its extension to new data formats. Parsing such data allows extracting various attribute values relevant to the origin of any detected incident. We automatically select the most relevant attributes by using several machine learning algorithms. These attributes increase the analysis accuracy and reduce the data dimensions as well as the computation resources needed.

The **historical data** is a set of data used for learning purposes. It consists of labeled records collected over time. These records describe the original cause of several incidents (e.g., a sensor is no longer permitted to send data to the central gateway) and the relative attribute values (e.g., downstream data bit-rate measured

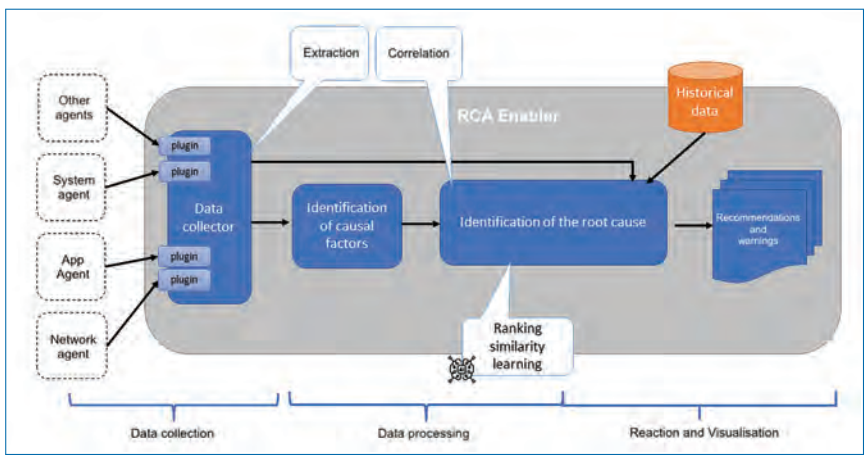

Figure 8.22. RCA Enabler high level architecture.

in the central gateway decreased). The **historical data** is constructed by two means:

- Active learning: We deal with controlled systems. Therefore, the collected data can be easily labeled by actively performing different tests injecting known failures and attacks.
- Passive learning: Once an incident is detected and alerted (e.g., by a third-party tool) without knowing its origin, thanks to the aid of the system experts, manual RCA is performed by debugging different logs and correlating various events to determine the corresponding root causes. The result of this work can be stored in the database with its relevant attribute values.

The historical data are derived from these two sources. The idea is to determine when the system reaches a known undesirable state with a known cause. It involves using the concept of Similarity Learning [8], i.e., Ranking Similarity Learning. The RCA tool calculates the similarity of the new state with the known ones. It presents the most similar states in the relative similarity order. The final goal is to recognize the incident's root origin by using historical data. In this way, the tool can recommend to the operator which countermeasures to perform based on known mitigation strategies.

The RCA Enabler works following two phases: the knowledge acquisition phase (Figure 8.23) and the monitoring phase (Figure 8.24). The former is for building a historical database of known problems and incidents. The latter consists of monitoring the system in real-time, analyzing the newly-coming incident by querying the historical data, and suggesting possible root causes. It is worth noting that passive learning in the knowledge acquisition phase can be continuously run during the monitoring phase. We describe the details of each module in the following subsections.

8.3.1 Data Collection

Analyzing an SIS requires different statistics and data, i.e., the logs, metrics, network traffic, and any data that could identify the system state. A data collector is necessary and can be provided by the system (e.g., in the ITS use case, the metrics are collected and sent to the RCA enabler via the MQTT broker), or an enabler can be deployed to collect different types of data, namely:

- Capturing network traffic: For example, the MMT-Probe [11] (TCP/IP networks) and the MMT-IoT [4] (IoT- 6LoWPAN) are able to sniff and record the network traffic.

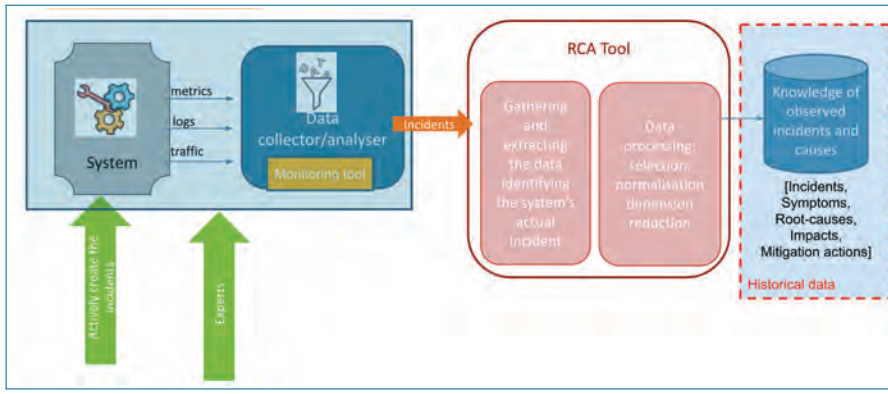

Figure 8.23. RCA-Knowledge acquisition phase.

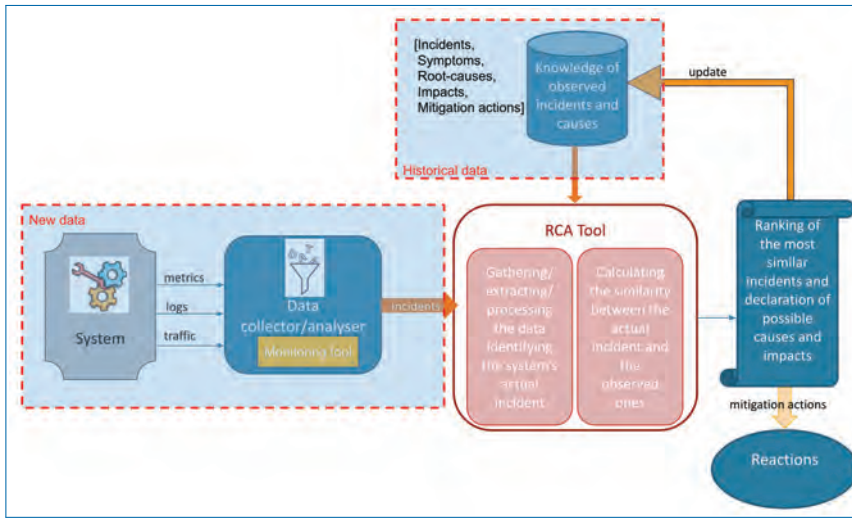

Figure 8.24. RCA-Monitoring phase.

- Reading and extracting logs: The current version of the RCA enabler supports by default reading the data input in the form of JSON and CSV files. Other formats can be rapidly taken into account thanks to the extensibility of MMT-Probe (e.g., creating new plugins).

In the knowledge acquisition phase, the data can be collected in two ways:

- Actively injecting or reproducing known failures and attacks, then collecting labeled corresponding data.
- Passively monitoring the system, debugging different logs and traces, correlating various events and particularly by consulting the system experts to determine the corresponding root causes as well as the relevant data.

In the monitoring phase, the data is collected and transmitted to the RCA enabler in real-time. In theory, there is no restriction in the type of data to be gathered. On the contrary, a maximum of data for identifying the system functionalities is desirable. Even though some data could be redundant, data processing steps are performed to extract the most pertinent data.

8.3.2 Data Processing

As we mentioned in Section 8.1, there can be an enormous number of components/indicators in the analysis of an IoT system. In the following subsections, we discuss our techniques that avoid data noise, deal with heterogeneous data, and calculate the similarity between two different data sets.

8.3.2.1 Attribute selection

Attribute selection (also known as feature selection) [5] is one of the core concepts in Machine Learning that tremendously impacts the model performance. For complex systems, it is common that the data collected is too complicated or redundant. In other words, there might be some irrelevant or less important attributes (i.e., noises) contributing less to the target variable. Removing the noises helps not only to improve the accuracy but also to reduce the training time. It is the first and most essential step that should be performed automatically based on the feature selection techniques or manually by system experts.

The current version of the RCA enabler has been integrated with the following feature selection techniques:

- Univariate feature selection: The selection of the best features is based on univariate statistical tests. Each feature is compared to the target variable while the other features are temporarily ignored. The goal is to determine whether there is any statistically significant relationship between them. Each feature has its test score. The bigger the score is, the more likely the feature is important. The features with top scores should be selected. The test score is the average of the scores calculated based on the chi-square test, the f test, and the mutual information classification test [5].
- Recursive feature elimination (RFE): It is about selecting features by recursively considering smaller and smaller sets of features. The idea is to use an external estimator (logistic regression model and random forest model [7]) that assigns weights to features (e.g., linear model coefficients). The least important features are pruned step-by-step from the current set of features. This procedure is recursively repeated on the pruned set until the desired number of features left is eventually reached. Compared to univariate feature selection, RFE considers all features at once, thus can capture interactions.

8.3.2.2 Data normalisation

Normalization is a concept informally used in statistics, and the term "normalized data" may have different meanings. In principle, data normalization means eliminating heterogeneous data measurement units and making the attributes comparable despite different value ranges.

In our perspectives, data normalization consists of two steps:

- Standardizing data to have a **mean** of zero and a **standard deviation**(s) of 1 (Equation (8.1) and Figure 8.25):

$$x_{standardized} = \frac{x - mean(x)}{s} \tag{8.1}$$

- Re-scaling the data to have values between 0 and 1 (Equation (8.2)):

$$x_{normalized} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{8.2}$$

8.3.2.3 Similarity calculation

Suppose that the system's temporal state can be reflected by n metrics (i.e., n attributes). This set of n attributes can be represented by a vector in a multi-dimensional space of n dimensions. Calculating the similarity and dissimilarity of two states becomes the problem of measuring the distance of orientation (the angle) and magnitude (the length) of their two representing vectors. Figure 8.26 depicts an example in a 3-dimensional space.

The current version of the RCA enabler has been integrated with the following similarity and distance measures:

- Cosine similarity [2]
- Adjusted cosine similarity [2]
- Jaccard similarity [2]
- Euclidean distance [2]
- Manhattan distance [2]
- Minkowski distance [2]

These measures are used to calculate the similarity score whose value is between 0 and 1. The bigger the similarity score is, the more similar the two compared states are (e.g., if the similarity score is equal to 0.95, there is a 95% probability that two compared states are considered the equivalent). The similarity score can be computed based on one or multiple similarity and distance measures. In the training phase, we determine the measures. Therefore, when we compare a known

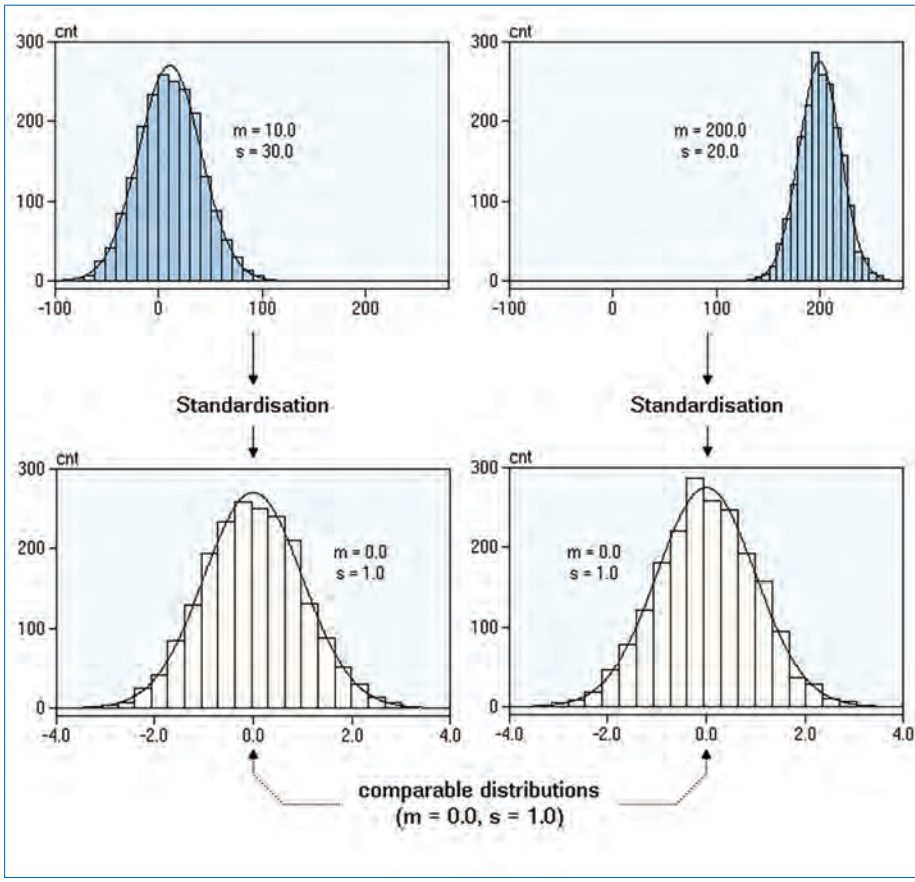

Figure 8.25. Data standardization.

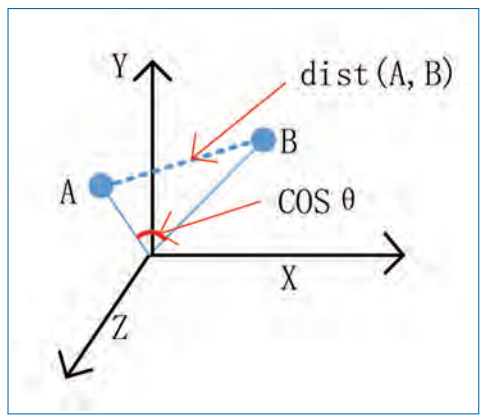

Figure 8.26. Similarity calculation in 3-dimensional space.

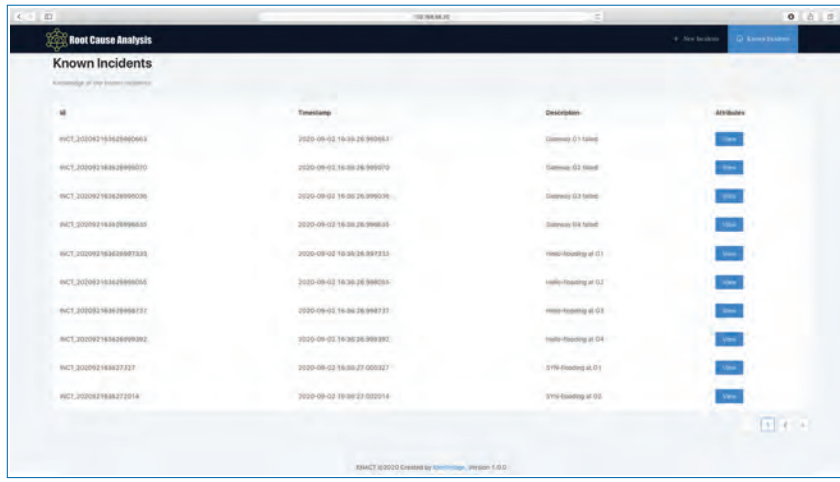

Figure 8.27. Historical database of known incidents.

state and its repetition, we have a similarity score as big as possible. Besides, to avoid false positives, the similarity between a common proper state and each known malicious state should be as low as possible.

8.3.3 Reaction and Visualization

First, the RCA enabler analyses live the data originating from the system under monitoring. It reports back the similarity score of the current state, its most similar known incident, and the corresponding root causes. If the similarity score is higher than a given threshold, an alert is generated. The alert helps the system administrators foresee how the system evolves from a normal functioning state to a known fault or failure and determine the root causes to perform the most appropriate mitigation actions. For example, in the ITS use case, the RCA enabler communicates with the ITS through an MQTT connection. When the RCA enabler receives a message with all the relevant attributes, it identifies the level at which the system state and a known incident are similar. It publishes the result to the MQTT exchange.

Regarding the visualization, the results of the RCA enabler can be viewed intuitively on a GUI. Figure 8.27, Figure 8.28, Figure 8.29 displays some screenshots of RCA's GUI. More results are presented in the following sections.

8.3.4 Evaluation

8.3.4.1 Performance evaluation with generated testing data

To evaluate the RCA enabler, we first generated a learning data set in CSV format with several known records. Each record describes an "incident" with different

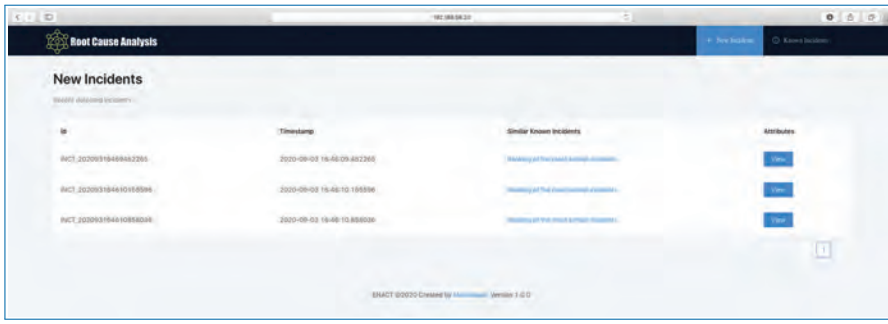

Figure 8.28. Newly detected incidents.

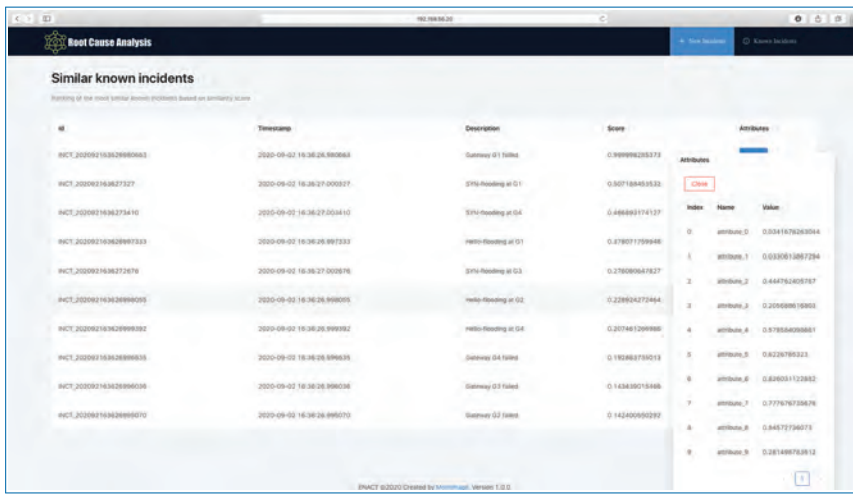

Figure 8.29. A newly detected incident and its similarity scores in comparison with known ones.

"attribute" values. These learned "incidents" are stored together with the potential origin causes. The "attributes" refer to the metric values that can be gathered. Afterward, we generate new attributes and check whether the system recognizes them as a known incident among the ones that are already in the historical data. The RCA enabler measures the similarity of each new record and each learned incident. It ranks them by determining the most likely similar ones. To assess the performance, we calculate the enabler's response time with the function taking as input the number of known states and the attributes identifying a state. In this evaluation, the similarity score is the average of all similarity measures mentioned in Section 8.3.2.3.

Figure 8.30 and Figure 8.31 show the results of our experiments. As expected, more time is needed when the data volume handled increases. However, we observe that the number of known states has less impact on the response time than the

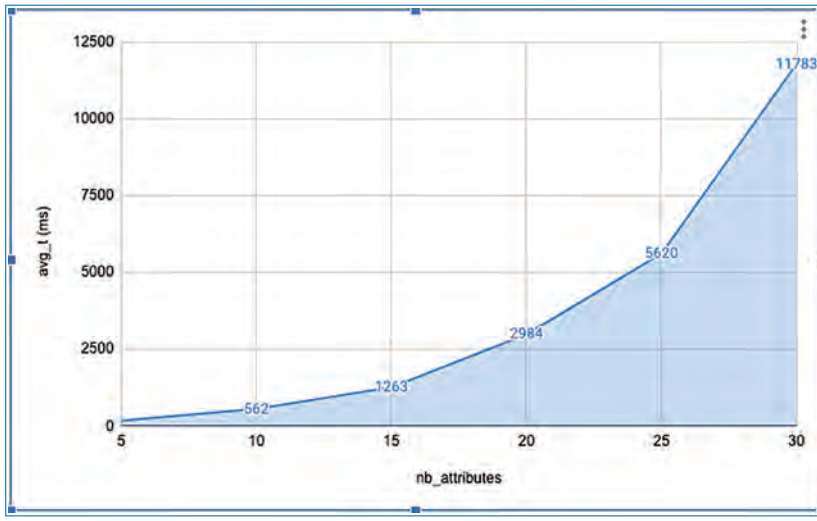

Figure 8.30. RCA Enabler's response time towards the number of attributes.

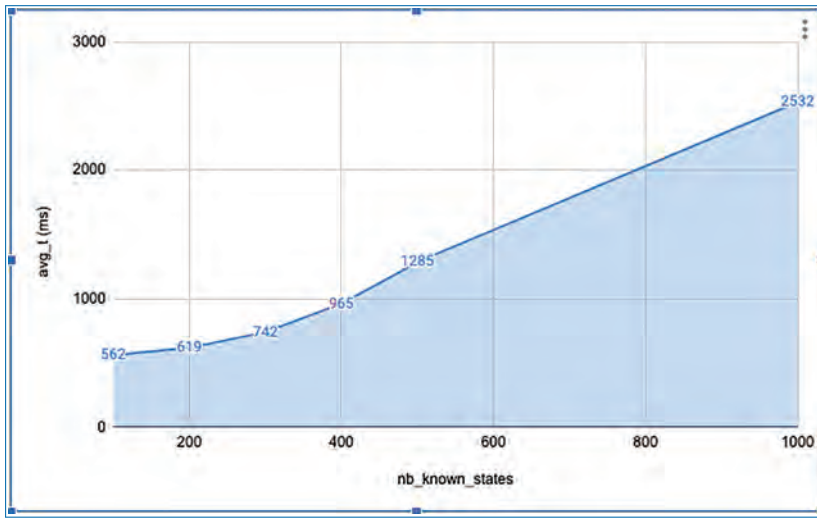

Figure 8.31. RCA Enabler's response time towards the number of known incidents.

number of the attributes has. The processing time needed increases more drastically when more attributes are under consideration than when there are more known states. This increase reaffirms the need for integrating "attribute selection" as aforementioned in Section 8.3.2.1. In our evaluation, we did not apply any selection technique because the data was generated randomly. The attributes are, thus, seemingly equal and make the selection not useful. However, there will probably be a different story when real systems are involved (further discussed in Section 8.3.4.2).

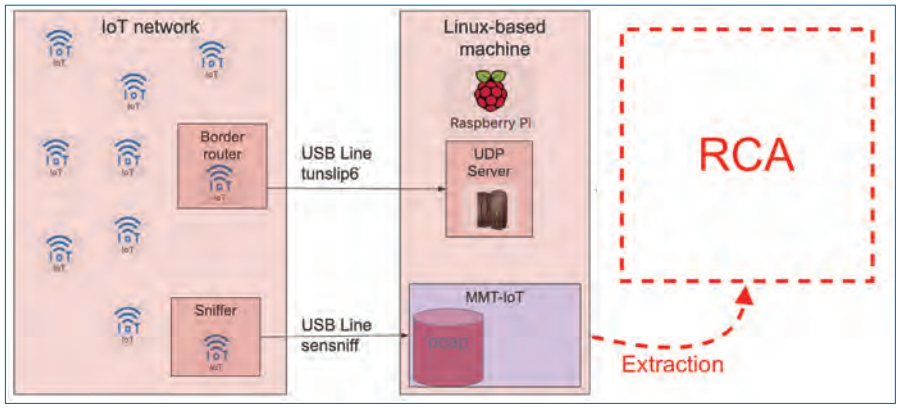

Figure 8.32. General architecture for the evaluation.

8.3.4.2 Evaluation on a real IoT Testbed

Set-up of the experiment

To evaluate the RCA enabler, we performed several experiments on a real IoT testbed called w-iLab.t[1] provided by Imec.[2] Figure 8.32 presents the general architecture of the experiments. Several IoT devices (Zolertia Re-Mote[3]) formed an IoT network where the clients reported sensed data periodically to the border router before these reports were forwarded via a USB line to a server installed in a more powerful Linux-based machine. There was an IoT device to perform sniffing tasks: capturing network traffic and piping via the USB line to the Linux-based machine where MMT-IoT was deployed to analyze the traffic and extract the metrics for the RCA enabler. Besides, the IoT network consisted of normal clients reporting sensed data every 10 seconds, and one (or several) attacker(s) behaved interchangeably in three modes:

- Normal mode reporting data every 10 seconds.
- DoS (Denial of Service) attack mode reporting data 100 times faster (10 messages/s) and with incorrect Frame Check Sequence (FCS[4]).
- Dead mode not reporting data at all (node failure).

1. https://www.fed4fire.eu/testbeds/w-ilab-t/
2. https://www.imec-int.com/
3. https://zolertia.io/zolertia-platforms/
4. FCS: The FCS field contains a number calculated by the source node based on the data in the frame. This number is added to the end of a frame sent. When the destination node receives the frame, the FCS number is recalculated and compared with the number sent in the frame. If they are different, the frame is considered malformed (intentionally or not) or modified between the source node and the destination node.

Table 8.2. Attributes extracted and sent to RCA.

Ref.	Attribute	Description
(1)	Network throughput (bps, pps)	The whole network traffic throughput, computed in bits per second (bps) and packets per second (pps)
(2)	Throughput at devices (bps, pps)	Throughput estimated at each device.
(3)	Traffic transmitted on links (bytes, packets)	Traffic volume transmitted on each link during a parameterized period (e.g., 10 seconds)
(4)	Number of routing-related packets (packets)	Number of routing-related packets sent and received by each device during a parameterized period (e.g., 10 seconds)
(5)	Transmission delay (ms)	The duration in millisecond since the packet is created by a device (timestamp packaged in the sensed data) until it is captured by the sniffer (captured packet's timestamp)
(6)	CPU usage (%)	CPU usage at each device, packaged in the sensed data.
(7)	Memory usage (%)	Memory usage at each device, packaged in the sensed data.
(8)	Battery level (%)	Level of battery left at each device, packaged in the sensed data.
(9)	Power consumption (W)	Power consumption (DC) at each device in Watts, packaged in the sensed data.
(10)	Average packet size (bytes)	Average size of packets transmitted during a parameterized period (e.g., 10 seconds)
(11)	Probe ID	An integer number representing the ID of the MMT-Probe analysing the traffic and performing the extraction.
(12)	Protocol ID	An integer number representing the protocol ID

Table 8.2 summarizes the attributes extracted by MMT-IoT and transferred to RCA for further analysis.

Attribute selection and data normalisation

As the first step, the RCA enabler selects the significant attributes among the 12 listed in Table 10. The selection is done by applying the techniques aforementioned in Section 8.3.2.1.

An incorrect FCS can signify a malformed packet (e.g., due to a misconfiguration or an error in the implementation), a jamming attack (i.e., the attacker abuses the network by generating frames that should be ignored), or a message manipulation attack (i.e., the attacker intercepts and modifies a frame's content).

Table 8.3. Feature Selection results.

	Univariate feature selection			Recursive feature elimination	
Ref.	Chi-square test	f-test	Mutual information classification test	Logistic regression model	Random forest model
(1)	true	true	true	1	true
(2)	true	true	true	2	true
(3)	true	true	true	2	true
(4)	true	false	true	6	false
(5)	true	true	true	3	true
(6)	true	true	true	2	true
(7)	true	true	true	2	true
(8)	false	false	false	9	false
(9)	false	true	true	6	false
(10)	false	false	false	8	false
(11)	false	false	false	10	false
(12)	false	false	false	10	false

Table 8.3 summarizes the results when different Feature Selection models are used. There are six attributes, namely (1–3), (5–7), which are considered significant according to all the models. Four attributes (8), (10), (11), and (12) are concluded to be not relevant and can be left out. The attributes (4) and (9) are recommended by some models and not by others. We performed the analysis in the following subsection with these two attributes and the other six attributes recommended by all the models.

Similarity calculation and analysis

Firstly, regarding the DoS attack, one can see clearly in the statistics displayed by MMT-IoT that:

- The traffic volume increased significantly during the attack period (Figure 8.33).
- The attacker was evidently the most active device (Figure 8.34) and one end of the most active link (Figure 8.35).

From the RCA point of view, all other selected attributes were more or less affected by the DoS attack. The attack pattern was learned, and when repeated,

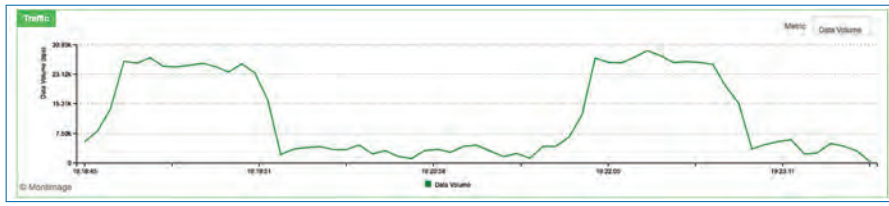

Figure 8.33. Traffic throughput increased remarkably when the attack took place.

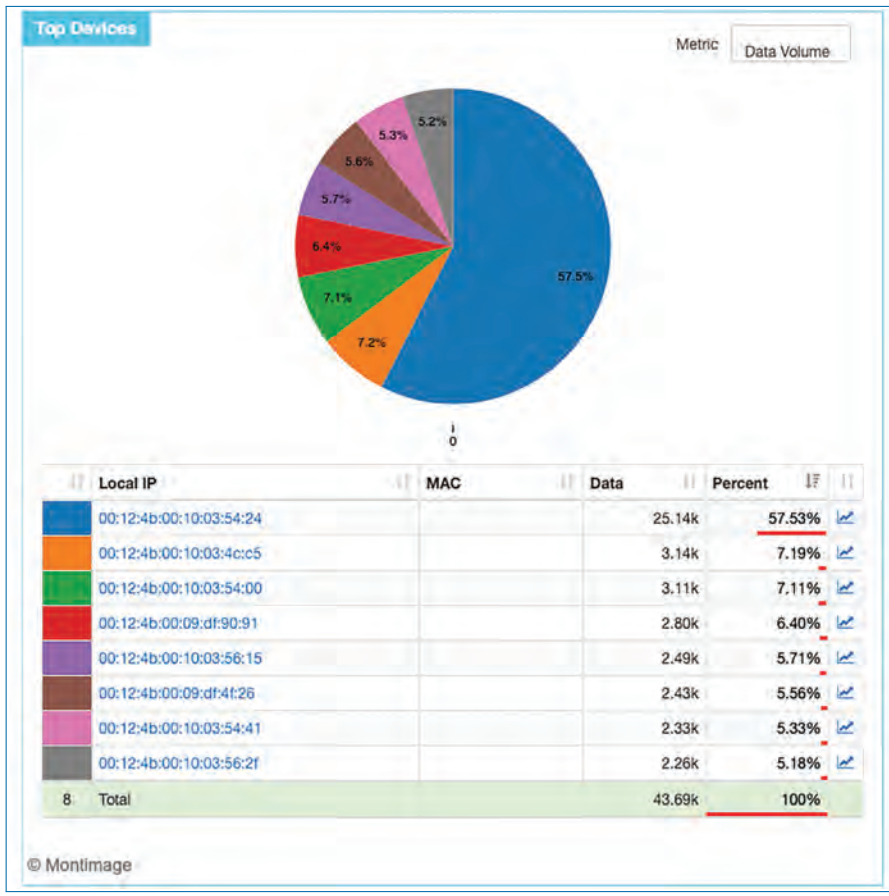

Figure 8.34. The attacker was the most active device.

the similarity score observed by the RCA was always no less than 0.92 (i.e., 92% similar). It is worth noting that, in this evaluation, we computed the similarity score based on the "adjusted cosine similarity".

In node failure (dead device), all the attributes related to the dead device were affected. The RCA enabler reported a similarity score between 0.84 and 0.87 when the failure repeated.

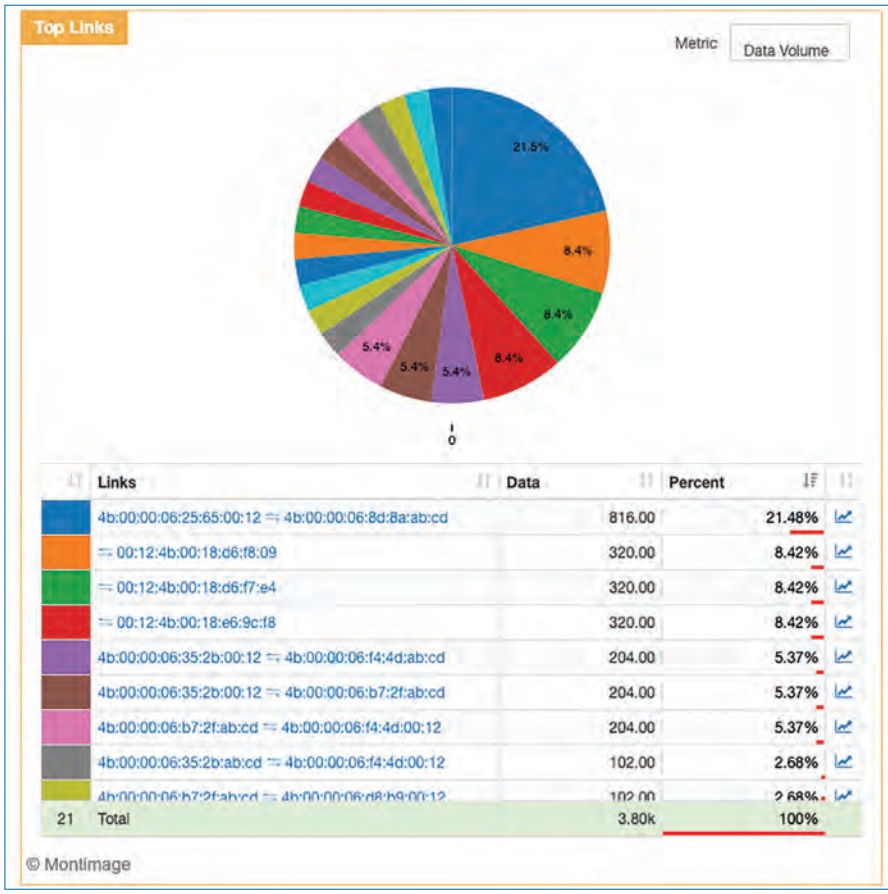

Figure 8.35. The attacker belonged to the most active link.

Lastly, when the border router was affected by the jamming attack of incorrect FCS values, its CPU usage jumped virtually. When this behavior happened again, the RCA enabler determined that it was up to 94% similar to the learned incident's observed symptoms. However, even when all the network devices worked normally, the RCA enabler identified that there was up to 78% similarity between this incident and the event "Possible jamming attack with incorrect FCS values".

8.4 Conclusion

In conclusion, This chapter presents two tools, i.e., the TaS and RCA enablers, that enable the validation and verification of IoT systems. The TaS enabler, based on the idea of "Digital Twins", is a Software-as-a-Service solution that provides

(i) a flexible simulation of sensor networks, (ii) a powerful data generator with real-time data recording, and (iii) support for Continuous Integration and Continuous Development. It helps IoT application developers save time and money on setting up the testing environment and thus supports faster application delivery. The RCA enabler systematizes the knowledge about the potential incidents that may occur in the system. It prevents the incidents or to quickly and intelligently react against their recurrences.

In practice, we can apply the TaS and RCA enablers to various systems other than IoT systems in the context of ENACT. In general, they can work on any system in which the data about the system's functioning state can be collected. For the RCA enabler, it would be beneficial if the owner or administrator has already acquired a certain level of understanding about the system to facilitate the training phase and the database creation for known incidents and root causes. Otherwise, we can perform penetration tests to discover potential vulnerabilities so that the attacks and failures can be injected and learned.

Moreover, both tools have been developed to be generic enough so that adaptations can be easily made to make them applicable to various types of systems (e.g., industrial SCADA systems, 5G mobile networks). We plan to adapt and use these two tools in several other collaborative projects in different contexts. We hope they will play a crucial role in the Montimage ecosystem and be commercialized within the MMT Monitoring solution.[5]

References

[1] C. Adjih *et al.* "FIT IoT-LAB: A large scale open experimental IoT testbed". In *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. 2015, pp. 459–464. DOI: 10.1109/WF-IoT.2015.7389098.

[2] Marc Sebban Aurélien Bellet, Amaury Habrard. *Metric Learning, Similarity-Based Pattern Analysis and Recognition*. Morgan and Claypool, 2015. ISBN: 1939-4616.

[3] A. Fuller *et al.* "Digital Twin: Enabling Technologies, Challenges and Open Research". In: *IEEE Access* 8 (2020), pp. 108952–108971. DOI: 10.1109/ACCESS.2020.2998358.

[4] Vinh Hoa La, Raul Fuentes, and Ana R. Cavalli. "A Novel Monitoring Solution for 6LoWPAN-based Wireless Sensor Networks". In: *Proceedings of 22nd Asia-Pacific Conference on Communications (APCC 2016)*. 2016.

5. https://montimage.com/products/MMT_DPI.html

[5] Richard Lowry. *Concepts and Applications of Inferential Statistics*. Vassar College, 2008.

[6] Sara N. Matheu-García *et al.* "Risk-based automated assessment and testing for the cybersecurity certification and labelling of IoT devices". In: *Computer Standards & Interfaces* 62 (2019), pp. 64–83. ISSN: 0920-5489. DOI: https://doi.org/10.1016/j.csi.2018.08.003. URL: https://www.sciencedirect.com/science/article/pii/S0920548918301375.

[7] Kjell Johnson Max Kuhn. *Feature Engineering and Selection: A Practical Approach for Predictive Models*. Taylor & Francis, 2019.

[8] Marcello Pelillo. *Similarity-Based Pattern Analysis and Recognition*. Springer, 2013. ISBN: 978-1-4471-5628-4.

[9] Luis Sanchez *et al.* "SmartSantander: IoT experimentation over a smart city testbed". In: *Computer Networks* 61 (2014). Special issue on Future Internet Testbeds Part I, pp. 217–238. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.bjp.2013.12.020.

[10] M. Tappler, B. K. Aichernig, and R. Bloem. "Model-Based Testing IoT Communication via Active Automata Learning". In: *2017 IEEE International Conference on Software Testing, Verification and Validation (ICST)*. 2017, pp. 276–287. DOI: 10.1109/ICST.2017.32.

[11] B. Wehbi, E. Montes de Oca, and M. Bourdelles. "Events-Based Security Monitoring Using MMT Tool". In: *IEEE Fifth International Conference on Software Testing, Verification and Validation (ICST), 2012*. Apr. 2012, pp. 860–863. DOI: 10.1109/ICST.2012.188.

DOI: 10.1561/9781680838251.ch9

Chapter 9

SIS-based eHealth Application: The Tellu Use Case

By Arnor Solberg, Oscar Zanutto and Franck Fleurey

9.1 From Chronic to Pro-active Care

To date, an average of 80% of public health care resources in Europe are spent to respond to chronic diseases that are exacerbated in the last three years of people's lives, against a low investment of resources in the field of prevention, which can be pursued by changing lifestyles. Public health expenditure is among the largest and fastest growing spending items for governments. In 2015, public expenditure on health was 7.8% of GDP in the EU as a whole, with more than 70% of expenditure funded by the public sector in two thirds of Member States (EC 2017). In 2013, premature deaths due to major NCDs (cardiovascular diseases, cancers, respiratory diseases and diabetes) cost EU economies 0.8% of GDP (OECD/ EC 2016), with further losses incurred due to the lower productivity and employments rates of people living with chronic health problems. Due to population aging, chronic diseases and the diffusion of new diagnostic and therapeutic technologies, the share of GDP spending on health is projected to increase in the coming years (EC 2015, OECD/ EC 2016). In most high and middle-income countries, non-communicable diseases are responsible for the biggest share of such healthcare costs (EC 2014). Furthermore the ongoing pandemic situation has boosted the demand of online telecare, eHealth solutions. General practitioners and nurses have improved the

remote physical parameters gathering, almost regarding oxygen saturation and body temperature. Due to these kind of increasing use of technology, care provider organizations has improved and scaled up their organizational models also in terms of digital employees' digital skills and care workflows.

This panorama indicates that institutions and care providers adopt a pro-active care approach aimed at significantly influencing individual, collective and organisational behaviour so that people can consciously plan lifestyles in which the preventive role played by the behavioural determinants of longevity is valued: nutrition, physical activity, cognitive stimulation and sociality.

This last aspect, in particular, has a decisive role as a health protection factor. It has been shown that perceived loneliness has an impact in terms of mortality comparable to smoking fifteen cigarettes a day. Furthermore, it is frequently associated with anxiety, depression and reduced movement, which can lead to hypertension and metabolic disorders with chronic degenerative effects. In this sense, living, lifestyle and technological support are integrating into a unicum that characterises the ecosystem in which the health design of the future is embedded. Within this new perspective, the process of longevity requires social and health services to overcome the dichotomous logic of intervention structured on antitheses such as health vs. disease, autonomy vs. dependence, placing rather their offer within a continuum that contemplates paths of reversibility, compensation, homeostasis, and new dynamic adaptations to the needs of the subject.

It is therefore necessary to imagine a model of person-services relationship based on the concept of co-production of health. In such a framework, the intervention of technology is inserted in support of care in a co-decided way with the person: at one extreme, ICT assumes a role of support to the fitness and well-being (i.e. prevention) of autonomous and still healthy citizens, to reach at the opposite extreme the apex of the technological complexity connected to an increase in the intensity of health care, passing through moments in which it becomes possible to set up an "intermediate" action of technological support, for example in the management of chronicity at home and in the transitions between the services used by the subject. eHealth solutions have increased their presence, and their perceived usefulness, following the development of the Covid pandemic19. In the global context, and in the European context in particular, there has been a proliferation of state-sponsored applications that provide information on the disease, ensure contact tracing, and create an informed dialogue with one's doctor and the Covid19 emergency management team in one's territory. The applications have also made it possible to remove much of the bureaucracy involved in the relationship between citizens and the health system, since many of the activities relating to the booking of diagnostic examinations, their payment and their reporting have moved online. Lastly, one of the most interesting aspects of the increase in the use of wearable technologies

for measuring certain health parameters is the growing possibility of acquiring data capable of feeding machine learning and data processing systems capable of activating artificial intelligence systems capable of making diagnostic forecasts and providing useful information for personalising care in increasing numbers.

At the centre of this paradigm is the decision-making process involving experts and the person: it is based on the personalisation and timing of the use of technologies, and on the personalised design of the integrated care process structured in collaboration with the family (where present) and the social and health services, public and private, which may be activated.

9.2 e-Health and m-Health for the Digital Evolution of Services

The European Commission defines eHealth and Digital health and care as: "Digital health and care refers to tools and services that use information and communication technologies (ICT) to improve prevention, diagnosis, treatment, monitoring and management of health and lifestyle. Digital health and care has the potential to innovate and improve access to care, the quality of care and to increase the overall efficiency of the health sector".

In the field of care for the elderly, tele-assistance is one of the answers that best translates the above statement into concrete terms. It must be understood as the person's ability to communicate remotely with home care providers and their social surroundings through the use of devices such as tablets and smartphones. These devices, equipped with integrated adapted video communication applications, are often placed in dialogue with wearable devices capable of automatically acquiring information about certain significant critical parameters, such as blood pressure and blood sugar in the case of fragile people suffering from chronic diseases such as hypertension and diabetes. They are therefore able to signal the exceeding of individual critical thresholds, activating the subject and starting a pre-set alarm chain.

In this sense, in 2017 the European Commission launched the initiative "Blueprint strategy for a digital transformation of health and care in an ageing society" proposing a structured path that links four distinct but fundamental worlds in advancing care innovation alongside technology: universities, companies, public authorities and citizens. The objective pursued is to transform social challenges into opportunities for economic growth associated with an increase in citizens' well-being. This initiative foresees a "multiplier effect" to boost the digital transformation of the entire health care secotr. For companies, research organisations and care providers operating in the social and health sectors, the indications contained in the

Figure 9.1. Evidence of the positive results produced by the implementation of this approach, defined as "quadruple helix", is the success of numerous experiences mapped by the study of European excellence in the sites selected by the European Innovation Partnership for Active and Healthy Ageing.

"Blueprint strategy" represent a fundamental reference point for structuring digital innovation paths in care. This document directs corporate efforts towards the adoption of a perspective in which people and their needs are placed at the centre, aiming at their empowerment to achieve independent living in their own context. The elements connected to the participatory co-design of technological solutions, as well as the creation of sustainable business models capable of making care systems more efficient, represent the drivers to be followed for the digital transformation of services supporting frail persons.

A virtuous example is represented by the recent HoCare2.0 Project, done in the Interreg Central Europe Programme (https://www.interreg-central.eu/Content.Node/HoCare2.0.html) that is going to codesign and provide customer-centered home care by co-creation with citizens. The Project foresee the creation and the devices adaptation to the user needs in combination with the SMEs knowledge to come up with technological solutions that could be relevant and usable in the daily life.

Another experience is The "Electronic Health Care Record and Integrated Information Systems" that has been implemented by the Valencian Health Agency to improve the integration and interoperability of systems and guarantee their sustainability, with greater efficiency and quality of service and according to a citizen-centred approach.

Below are some examples of the impact of the programme:

- In the context of the Integrated Home Care Programme, approximately 7,000 patients were treated with an overall satisfaction index of 92.7%; 154

were saved for each stay in hospital, which was 30% less than the Spanish national average duration;
- the Electronic Dossier is accessible to 50,000 health professionals and 373 pharmacies. There are 5.1 million clinical pictures of patients, 43 million clinical documents are registered, 150,000 visits are conducted daily online. This information, integrated with each other, has enabled better control of treatment interactions and drug administration and increased quality support for professionals' decision-making;
- system development has created a boost for the IT industry in the region: 1,320 IT specialists and 107 companies have been involved at full capacity.

9.3 H2020 ENACT Project Pilot Testing Experience

This framework includes the experimentation conducted in ISRAA (ISTITUTO PER SERVIZI DI RICOVERO E ASSISTENZA AGLI ANZIANI) that is a Public care provider organization for older people, based in Treviso (Italy) concerning the investigation, and subsequent experimentation, of some technological solutions for the remote assistance of fragile people living in the residential context of "Borgo Mazzini Smart Cohousing" foreseen within the Horizon 2020 ENACT DevOps project on DevOps of trustworthy smart IoT systems.

The residential complex in which the elderly people who participated in the pilot reside is located in the historic centre of Treviso. It consists of 46 flats with a total surface area of 5,589 m2 in which elderly people live alone or in pairs with an average age of 75 years.

9.3.1 ENACT Pilot Scenarios on Smart Building and eHealth Impact

As the first step to the e-health IoT system design, in order to understand the attitudes and needs of the elderly residents in Cohousing towards technological innovations for the improvement of quality of life, a set of questions were defined to be asked to the residents in the form of an interview. Then, a focus group followed where eight residents representative of the elderly population used the designed technologies for environmental comfort and independent health management.

Below are some of the most significant elements taken from the survey conducted in August 2020:

The study showed a general inclination towards the adoption of technologies useful for monitoring both one's own health and the living environment for the benefit of one's comfort and safety.

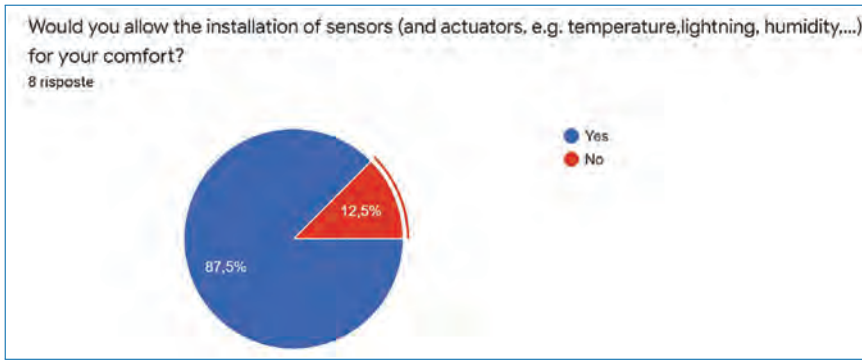

Figure 9.2. Percentage of users who agree to sensors installation for environmental comfort.

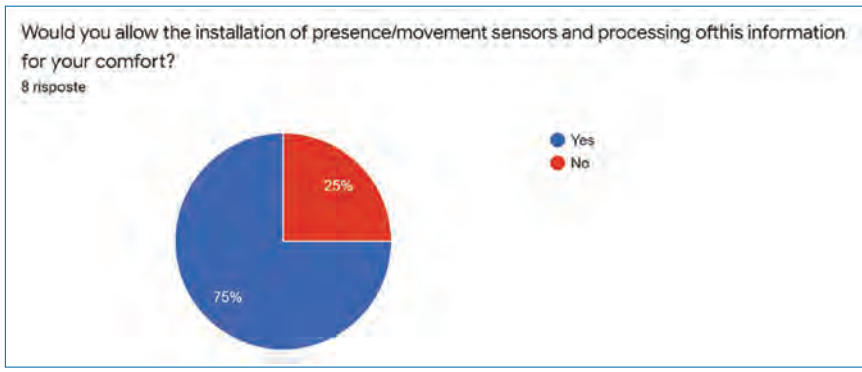

Figure 9.3. Percentage of users who agree to presence and movement sensors.

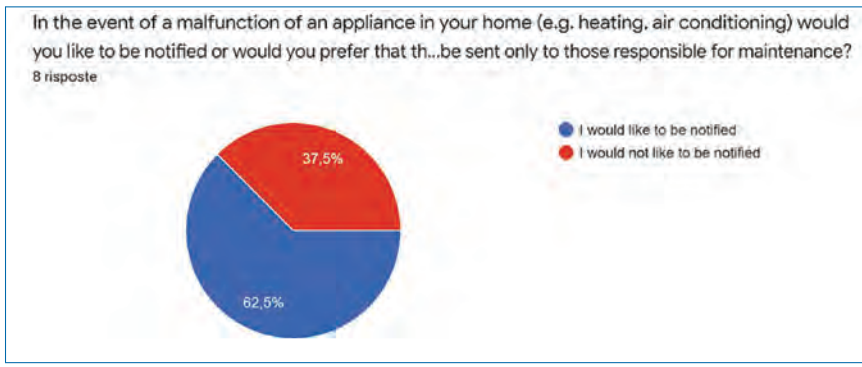

Figure 9.4. Percentage of users who want to get notifications of devices' malfunctioning.

On the basis of this attitude, some residents with conserved cognitive resources and a discreet functional autonomy were involved in order to test the devices provided by the Norwegian Company TellU that provide eHealth solutions in health care such as: thermometer, saturator, Oxymeter, sphygmomanometer capable of

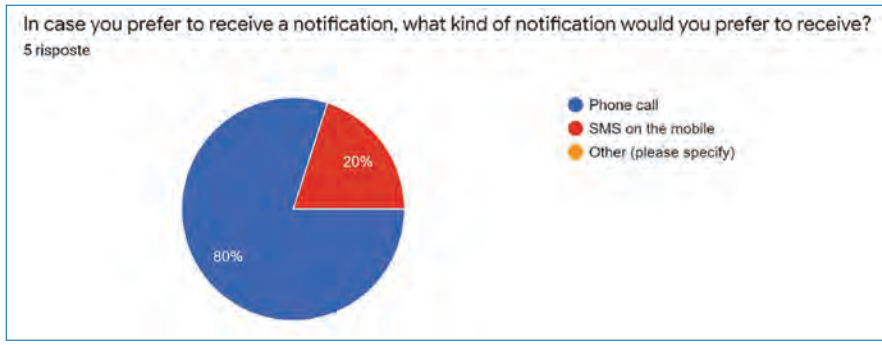

Figure 9.5. Percentage of notification modalities preferred from users in case of malfunctioning.

detecting and transmitting the parameters detected via Bluetooth in real time to the ISRAA care manager. In this way, on the basis of a personalised care plan, innovative teleprotection paths have been activated based on the detection of critical alarm thresholds, for each parameter, over which the care manager was able to act in a timely manner by innovating the care processes.

The experience was favourably in the eyes of the elderly people testing the solution, who were able to experience the benefits of these health support tools, highlighting the high usability of the devices throughout the trial.

With regard to the organisational impact, determined by the experimentation, it should be noted that the nurses and care management staff involved appreciated the time savings, the accuracy of information and the possibility of acting proactively, guaranteeing better health conditions for the people assisted.

9.3.2 Technical Overview of the eHealth Case Study

The industrial-based use case from TellU that was developed in ENACT is a Digital health system for supporting and helping various patients staying at home or in residencies such as cohousing to the extent possible during treatment and care, as well as to have tight interaction with health personnel through digital means in addition to adequate physical meeting points. This makes the patient more independent and it enables support for extensive self care. One type of "patients" supported by the provided digital services is elderly people, for whom the Digital health system will feature elderly care to allow the elderly to live safe at home. Another type of patients are people with chronic diseases such as Diabetes, Kidney diseases, Chronic obstructive pulmonary disease (COPD) and people with temporary diseases such as Covid-19 and cancer. These are patients that need to be regularly followed up and that would benefit from sensor based health status monitoring and digital self care services. For example, Diabetes patients apply sensors and devices to follow

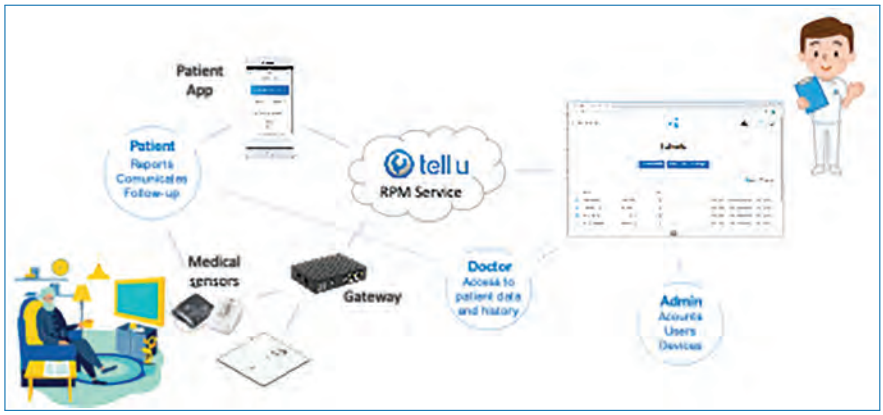

Figure 9.6. The general set up of the ENACT eHealth case study.

their glucose level and regularly provide measurements and questionnaire reports that can be followed up by health personnel. The general set up of the Tellu eHealth case study for medical telecare services is illustrated in Figure 9.6.

The digital health system controls both equipment that are deployed for remote supervision (such as bed sensors, motion sensors, sensors for indoor and out-door location, video based supervision, etc.) and various types of medical devices and specific sensors supporting the care and wellness for the specific patient (e.g., blood pressure meter, sphygmomanometer, Oxymeter, glucose meter, medicine reminder, etc.). In addition, the system can integrate with other systems, for instance to provide information or alarms to response centers, caregivers, physicians, family, etc., and to feed information to medical systems such as electronic health record systems.

In terms of managing the extensive distribution of devices, sensors and software across the IoT and edge space we exploit what we denote "the Personal Health Gateway" (PHG) which integrates the sensors and devices and that controls the edge and ensures the right data are provided to the various stakeholders and to the cloud based system. Thus, the handling of large numbers of largely distributed personal health gateways and their connected sensors has been a main focus in this case study for the validation and exploitation of the ENACT technologies. In particular, we have explored the potential of ENACT for the IoT, edge and cloud services, by having smooth integration of heterogeneous devices, DevOps process for the development of the edge components, as well as secure and trustworthy connections and data transfers. This case study is set up with a local/edge infrastructure consisting of a set of devices and a home gateway (GW). A set of such local infrastructures are then connected and aggregated into a cloud-based infrastructure. The overall technical architecture of the use case including the PHG are depicted in Figure 9.7.

The Personal Health Gateway architecture is the one depicted in the lower part of the figure, and is the element that is controlling the edge and connecting devices

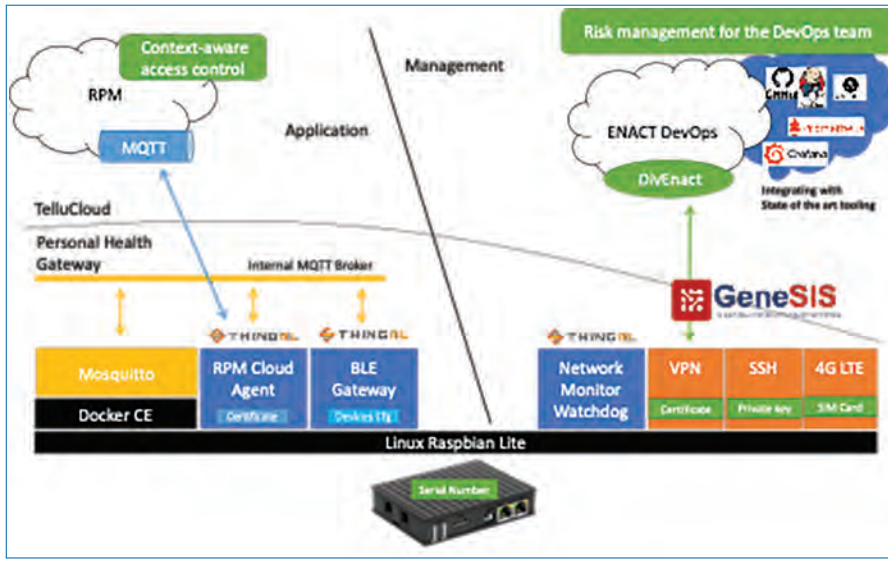

Figure 9.7. Overall architecture with the GW architectural components.

and sensors in the IoT and edge space, while the TelluCloud eHealth system resides in the cloud. The total system encompasses a complex ecosystem spanning IoT, edge and cloud. The Personal Health Gateway consists of a set of microservices to manage the various interactions with the devices and cloud services. The application level interaction and the management is completely separated. This is partly to ensure strict security and privacy requirements. The BLE gateway component manages Bluetooth Low Energy (BLE) enabled devices, for example blood pressure meter, scale, glucose meter, etc. The RPM cloud agent includes the application logic that resides at the edge level and interacts with the cloud level service. A set of microservices supports the management and DevOps process, providing access to system level operations of the Personal Health Gateway through secure channels. Moreover, it includes the monitoring component providing system and application level monitoring required for the continuous operation of the service. The Gateway includes an MQTT broker and support for standard internet communication protocols. The components run on docker containers. The application of the ENACT enablers is indicated in the overall architecture of Figure 9.7:

- The context aware access control is explored to provide more advanced application-level functionality in order to dynamically provide access to different stakeholders based on the context. Context can for example imply an escalated state or a crisis situation. For example, in case of a fire alarm in the patient's house, it may be important to provide further access to the installed camera for example to provide access to firefighters for them to better assess

the current situation, while in the normal state the camera will only be possible to be accessed by authorised health personnel;

- ThingML is fully exploited for the efficient coding and DevOps support of the Personal Health Gateway;
- GeneSIS together with DivENACT is explored for the efficient management and continuous deployment of potentially large scale deployments of our telecare service, where large amounts of IoT and edge devices such as welfare sensors and medical devices are managed through the deployed Personal Health Gateways (PHG) residing in people's homes. Note that the PHG is the software stack as depicted in the overall architecture figure above, thus, it may also be deployed on mobile gateways (e.g., smart phones) and we are currently releasing a new version of our PHG that can be deployed on Android and iOS based smart phones, enabling the patient to do medical measurements on travel.
- The ENACT Risk Driven Decision Support tool is explored as part of our DevOps process that needs to be compliant with standards such as ISO 27001, where risk analysis and risk management is required to be an integral part of the DevOps process.

Reference

[1] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, 1993.

DOI: 10.1561/9781680838251.ch10

Chapter 10

Intelligent Transport System: The Indra Use Case

By Francisco Parrilla, Sergio Jiménez Gómez, Modris Greitans and Janis Judvaitis

10.1 Introduction

The future of the railway market involves digitization, automation, connectivity and the use of intelligent systems that continue to add value to the society by improving management, operation and user experience, in order to face the challenge posed by the European Green Deal[1] as evidenced in the different Strategic and Innovation European agendas,[2] as well as in the plans and reports of public governs and relevant public organizations and Programs (such as the Shift2Rail European Innovation Initiative,[3] the Innovation Plan for Transport and

1. https://ec.europa.eu/info/strategy/priorities-2019-2024/european-green-deal_en
2. https://errac.org/publications/strategic-rail-research-and-innovation-agenda/
3. https://shift2rail.org

Infrastructures launched by the Government of Spain,[4] the French-Swedish Strategic Partnership for innovation, green solutions in the transport sector,[5] etc).

The digitalization of the railway market, as well as the automation and deployment of new intelligent systems, requires the design and implementation of new systems to be deployed in the railway ecosystem; systems that will pose a technological challenge to achieve the objectives set in the European agendas for the coming years, and that will entail major changes as well as the deployment of a large number of devices and subsystems both On Board and On Track. These systems and devices, to face a progressive and changing digitalization of the sector, must be prepared for agile development and deployment, where their control and monitoring provide the necessary mechanisms to guarantee an efficient, robust, safe, secure and updated operation, according to the demands and challenges to be met.

The ENACT results emerges as a facilitator to meet the proposed challenge, in line with the strategic lines and programs described for the future digitalization of the rail market providing DevOps enablers.

10.2 Rationale

The rail domain requires infrastructure and resources that are usually expensive and require a long-time planning and execution. Therefore, the usage of the rail systems must be trustworthy, following strict security and safety regulations. Several functionalities could be implemented within the rail systems to ensure that the system could tackle its high critical requirements as planned.

The proposed Use Case for railways shows how the use of the ENACT enablers can be used to enhance the DevOps cycle of new innovative systems – aligned with other innovation programs mentioned in the Chapter 10.1 – exploiting and evaluating their potential. The selected innovative systems have been analyzed, implemented and tested:

- **On Board WTI (Wireless Train Integrity)**[6]: This functionality is in charge of measuring, in real time, train composition parameters and evaluate them

4. Ministerio de Transportes, Movilidad y Agenda Urbana. Gobierno de España: "Plan de Innovación para el Transporte y las Infraestructuras". February 2018

5. https://www.tresor.economie.gouv.fr/Articles/2018/03/28/partenariat-franco-suedois-pour-l-innovation-et-les-solutions-vertes-french-swedish-partnership-for-innovation-and-green-solutions

6. Aligned with On Board Train Integrity Technology demonstrator tasks defined on TD2.5 addressed on X2RAIL-2 and X2RAIL-4 projects on which Indra is involved https://projects.shift2rail.org/s2r_ip_TD_D.aspx?ip=2&td=061d0fcf-51a6-4358-a74f-a4d34e8dac01 and making use of SCOTT https://scottproject.eu/ and DEWI results http://www.dewiproject.eu/

to report the train integrity status. The On Board system, based on WSN Sensors among the composition, provides the necessary information to determine the rolling stock material that composes the consists and evaluate and ensure, through an On Board unit, its integrity status. This integrity status is shown to the driver through the Train Management Systems (TMS) and a Cloud service interface.

- **Logistic and Maintenance System**[7]: This functionality provides information to register and locate both rolling stock material and on-track signalling devices and inform about their status. This functionality is required to solve the rail environment needs to locate and monitor the status of the big heterogeneity and flexibility of the compositions and signalling devices, making a special emphasis on the freight compositions, to optimize the rail business operation. The points that are optimized into the rail framework are the management of the rolling stock, cargo tracking, etc. To this end, it is required deploying IoT On Board and On Track, together with Cloud solutions to track and manage the rolling stock material data and to perform predictive maintenance.

These services are illustrated in the following Figure 10.1:

The DevOps role in the Use Case assists to reach this automation and digitalization objective, with the following focuses:

- **Security and Privacy Monitoring (S&P Mon&Con) tool**: This tool is responsible for monitoring and actuating over the On Board infrastructure to guarantee its security characteristics.
- **GeneSIS tool**: This tool monitor performs software remote deployments on the rail equipment to keep all the devices with the desired software version.
- **Behaviour Drift Analysis (BDA) tool**: This tool monitors the behaviour of the equipment to detect deviations related with the proper define behaviour for them.
- **Actuation Conflict Management (ACM) tool**: This tool detects conflicts that may appear in the Use Case operation and to help to resolve the conflicting actions.
- **Testing and Simulation tool**: It is a tool that simulates a part of the Use Case infrastructure and makes a Digital twin of it to be evaluated.

7. Aligned with Smart radio-connected all-in-all wayside objects demonstrator tasks defined on TD2.10 addressed on X2RAIL-1 and X2RAIL-4 projects on which Indra is involved https://projects.shift2rail.org/s2r_ip_TD_D.aspx?ip=2&td=061d0fcf-51a6-4358-a74f-a4d34e8dac01 and making use of SCOTT https://scottproject.eu/ and DEWI results http://www.dewiproject.eu/

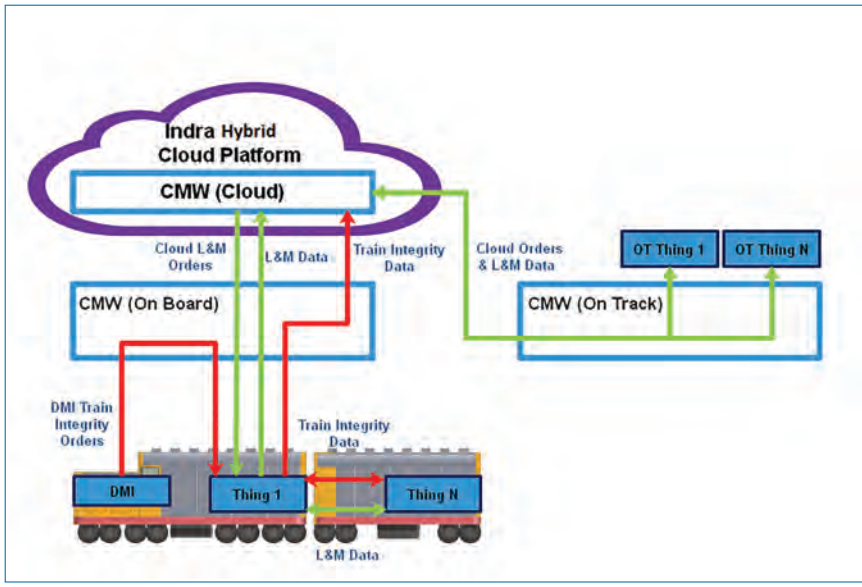

Figure 10.1. On Board WTI (Wireless Train Integrity & Logistic and Maintenance System Functionalities Scheme).
Source: INDRA.

- **Root Cause Analysis (RCA) tool**: This tool detects possible failures that may occur in the Use Case infrastructure informing about the most likely cause for that fail.

One of the challenges that the Use Case faces to reach the automation and digitalization of the rail environment is the scalability. The mentioned DevOps tools developed in ENACT for the Use Case have different objectives. However, these objectives converge on solving the scalability issues that the functionalities hide. To evaluate the scalability impact and to provide the issue's magnitude, a real rail scenario example is provided. An example to show the scalability issues is the Madrid-Barcelona French Border line, one of the first high speed lines built in Spain and one with the higher capacity (more trains per day). An example of a real line magnitude can be seen in Figure 10.2.

For this real example, focusing only on On Board systems, there are 94 trips circulating through this line: 47 of them from Madrid to Barcelona and the rest from Barcelona to Madrid. Each trip is accomplished by a single train composition that could be built following three different kinds of composition: simple, double, and mixed.

- **Simple**: There are 38 simple train compositions per day where there is only one locomotive wagon. Each simple composition is formed by 12 wagons.

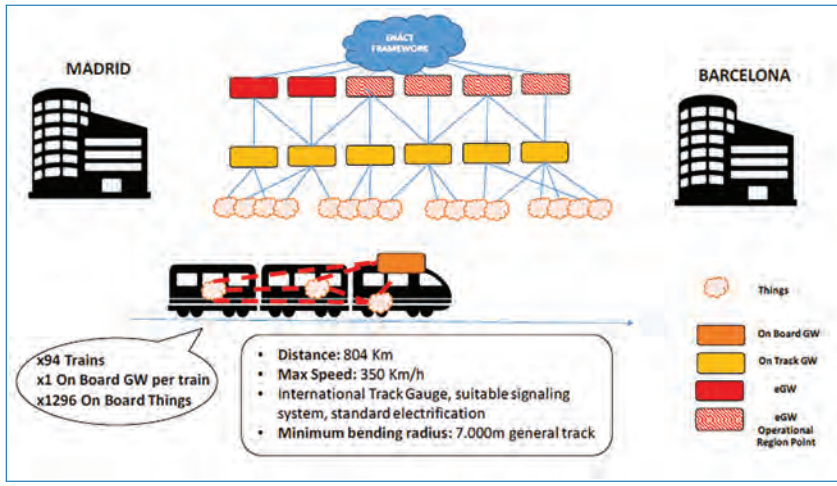

Figure 10.2. Madrid-Barcelona French Border rail line general characteristics. *Source*: INDRA.

- **Double**: There are 5 double compositions per day where there are two simple compositions joined. Each double composition is formed by 24 wagons.
- **Mixed**: There are 5 mixed compositions per day which are formed by different kind of pieces of rolling stock. Each mixed train composition is formed by an average of 18 wagons.

In the real line, there are running 1296 wagons in average, if both directions of the line are considered (94 trains per day). Based on the functionalities architecture, further details shown in Section 10.3, it is estimated that around 4000 units of On Board equipment are required to cover the On Board equipment for this line.

From the magnitude presented, we can verify that it is essential to provide a deployment mechanism to update all these systems in a controlled, automated and orchestrated way. It is important to remark that these systems should be developed with digital twins in mind to test the potential impact of new updates in a test environment. They faithfully reproduce the potential impacts of these updates in operating environment with mechanisms that ensure the cybersecurity of communications throughout the ecosystem and adjusts the deviations of the sensor networks involved. This serves to ensure the robustness of the different orders distributed throughout the global system.

10.3 Use Case Implementation

In this section, we explain the IoT platform brought to implement the rail functionalities and show how the ENACT enablers are applied. Moreover, it includes how

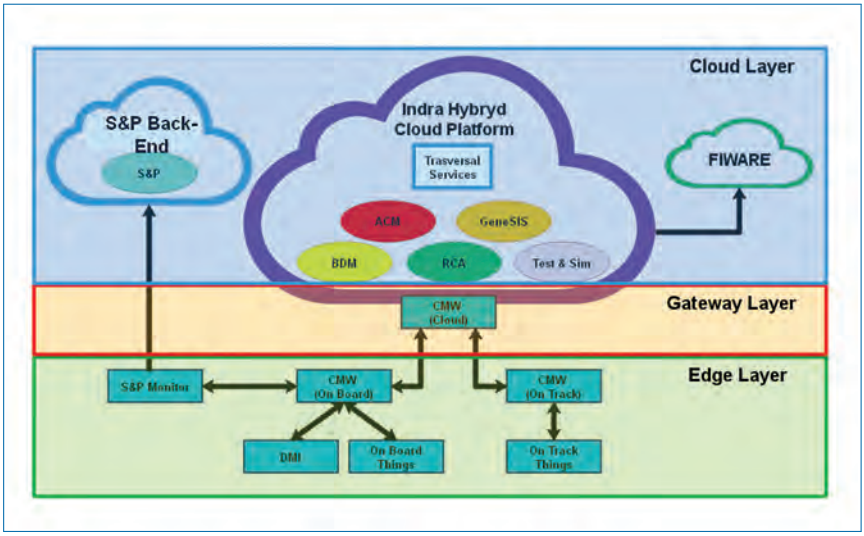

Figure 10.3. Rail Use Case architecture enriched with the DevOps tools. *Source*: INDRA.

the DevOps tools enhance the Use Case itself. As illustrated in the Figure 10.3, we identify three different architecture layers. These layers are the Edge, the Gateway, and the Cloud layers:

10.3.1 Edge

The edge layer is made up of the various objects (Wireless Sensor and actuator networks – WSAN – and other concentrating and/or communication devices) distributed both on the track and in the On Board equipment. This layer also serves to provide/send raw data to the functionalities. It is used by the BDA (10.4.5) and ACM (10.4.4) tools to perform conflict analysis and monitor the behavior of the devices Moreover, it is used by the Testing and Simulation tool to create a fair Things Digital Twin. The following elements form it:

- **Things**: The key element in this layer is defined under the name of Thing. The Things provide the functionality parameters described above.
  The Things are defined as a group of nodes which globes coordinators, sensors or actuators. In such a way, the Things have several capabilities: gather, actuate and communicate.

  It must be emphasized that the layer covers the On Board and On Track sections. WTI and a set to Logistic and Maintenance parameters are obtained and processed in the On Board section, while the On Track section participates in the provision of another set of Logistics and Maintenance

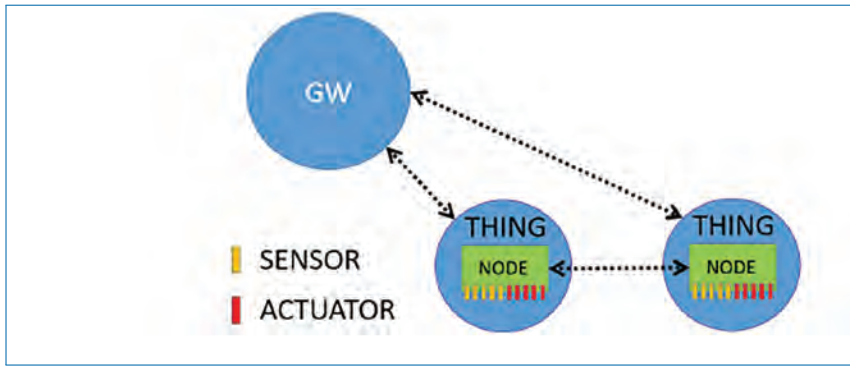

Figure 10.4. On Board and On Track Things.
Source: INDRA.

specific data. The On Track infrastructure uses RFID technology to obtain information from the train. These Things are powered up by an energy harvesting system to cover the case that no electrification systems are equipped On Board a train.

- Sensors:
  - On Board: an accelerometer, a Received Signal Strength Indicator (RSSI) sensor, and a Global Navigation Satellite System (GNSS) receiver, and Radio Frequency Identification (RFID) tag and reader.
  - On Track: RFID reader.
- Actuators:
  - On Board: Light-Emitting Diode (LED)s and displays.

The Things software consists of 5 modules: data managing module, inauguration module, integrity module, logistics module and maintenance module. They are working in synergy to provide the data and control over the On Board and On Track Things infrastructure deployed on the rolling stock.

- Data managing module:
  - *On Board*: The data is gathered at the nodes located on each wagon of the train. Data are sent over the air using the IEEE 802.15.4 based ZigBee protocol for secure and reliable communications. As typical for wireless sensor and actuator network, there is at least one so called base station, which forwards the data to the next processing point. The system is also capable of communication in the other direction sending commands from base station to the nodes.

    In this case the next processing point is WSN coordinator. WSN coordinator acts as a data forwarder from Things to the Communication middleware (CMW), while also ensuring that the data forwarded

are in the correct format. WSN coordinator is also responsible about the forwarding of commands to the Things from CMW. The commands from CMW can be: start inauguration, start/stop integrity control, start/stop logistics control, start/stop maintenance control and reset. All of the commands are part of modules described in next sections.

- *On-Track*: The on-track data gathering does not require the wireless network, the RFID sensor readings are gathered and forwarded to the CMW. On-track infrastructure listens to the following commands from CMW: start/stop logistics control and reset.

– Inauguration module: The inauguration module is implemented on the WSN coordinator. After receiving the command about the start of inauguration procedure from the CMW, the base station is notified to identify the nodes located on the wagons belonging to this train and check that they are operational. In this module also the physical order of the wagons is calculated using the GNSS and RSSI sensor values and RFID data. After successful inauguration the WSN coordinator sends corresponding message to CMW reporting the inauguration status.
– Integrity module: The integrity module is operated only in the On Board infrastructure and is responsible for continuously validating the train integrity while it is operational. Integrity control can only be started after a successful inauguration phase is finished and integrity control is not already operational. When the WSN coordinator receives the command from CMW to start the integrity control it notifies the base station to send out the integrity start command. After start command each of the nodes and base station reports sensor status every 250 ms or 4 times per second. The received sensor data is forwarded to the CMW as raw data consisting of GNSS position, accelerometer data and RSSI value measured at the base station for each wagon. Using this information also the train integrity is calculated, it is based on aforementioned sensors and train integrity is considered lost in case when at least two of three sensors report data that indicates that there is a train integrity issue. The train integrity information is also forwarded to the CMW. The train integrity system remains operational until the stop or reset command is received. If the start inauguration or start integrity command is received while the integrity system is operational, the command is ignored.
– Logistics module: The logistics module is responsible for providing the business information about the train logistics. Logistics module operates on the On Board and on-track infrastructure and uses the RFID reader data. On-track infrastructure provides wagon order information, but On

Board infrastructure provides information about the cargo and also allows to identify the wagon by RFID tag associated with it. The logistics module can be started after successful inauguration and stopped at will.

- Maintenance module: The maintenance module is responsible for providing the maintenance information about the wagons and infrastructure. While the maintenance module is operational it reports the node energy consumption and battery charge status. The maintenance module can be started after successful inauguration and stopped at will. Also the maintenance module handles the reset command, which can be issued by CMW and will reset the Things software in case of any errors. The reset command will not be accepted by the WSN coordinator if inauguration process is running or integrity control module is operational.
- Testing and validation: The developed infrastructure was validated using the demo setup of Lego train with minimal changes to the setup described above due to physical limitations of Lego train. The demo setup with Lego train was used to provide the partners with sensor, logistics and maintenance data from the on-track and On Board infrastructure while located in the lab environment but still providing non-generated data, thus making it easier to test, integrate, debug, and showcase developed technologies.

- **DMI**: The DMI (Driver Machine Interface) is included into this layer and it is responsible for managing the Things into the train by the driver and to perform Safety data treatment for the functionalities.
- **S&P Monitor**: This equipment is the key element in the security properties for the Use Case Edge layer. The S&P Monitor is an architecture component located On Board in charge of monitoring the traffic that is carried by the gateway layer 10.3.2. It must be emphasized that the traffic analysis ignores the business data related with the ITS functionalities. The reason behind this decision is that the functionality is in charge of covering the security aspects of the On Board equipment out of functionality focus. This elements is formed by several differentiated entities:
  - Hardware: The hardware offers the computing capacity to the software, the connectivity with the gateway by Ethernet and the connectivity with the S&P Mon&Con Back-end through a 3G/4G interface.
  - Monitor SW: The monitor software sniffs the traffic from the gateway in order to be treated and to be sent to the S&P Mon&Con Back-end for a further analysis.
  - User's removal Software: The S&P Mon&Con Back-end analysis could throw as a result that the users that generates the traffic monitored are intruders, therefore, a notification with the users is published to this

entity. This entity, developed by Indra, revokes the user in the ITS central authentication services.

10.3.2 Gateway

The gateway, so called CMW (Communication Middleware), is the element that connects all the Use Case system elements. It gathers the edge data and provides it to the Cloud. Moreover, it gathers the orders to the Things to start the functionality services. This CMW is based on Indra developed solution certified for rail environments.

From a general perspective, the basic functionalities of this CMW are as follows:

- Routing the messages from the different services to be sent to the different architecture entities. This routing follows a preconfigured topology to guarantee the provision of the information in a safe and secured manner.
- Synchronization tasks to keeps all the devices working with the same time reference.
- Authentication tasks at different OSI levels to enable the connectivity of the edge elements and to enable the connection of the CMW with the Cloud layer.
- Provision of CMW status metadata to be evaluated by the DevOps tools.

The Use Case, into the ENACT project framework, relays part the scalability issues to the CMW. As the CMW equipment cost is high to provide several of these devices to the project, considering the budget, the scalability is tests will be done in a single physical CMW and several virtual ones will be provided for testing. Therefore, the general architecture is formed by a single physical CMW located On Board and the several virtual CMWs.

10.3.3 Cloud

The Cloud layer is formed by the Cloud platforms that participate on the Use Case and how they are related. Three different Cloud platforms are considered: FIWARE [1], Indra Hybrid Cloud Platform, and S&P Back-end.

- **Indra Hybrid Cloud Platform**[8]: The cloud is a hybrid solution that combines a private and public part. The private par is in charge of the internal

8. https://www.igi-global.com/chapter/security-in-rail-iot-systems/258896

ITS task such as edge data storage, edge elements authentication, and external partner's authentication. The public part is in charge of integrating external elements such as tools or other Cloud platforms. This is the integration point made for all the Cloud resources and DevOps tools that require it. All the DevOps tool, except the S&P tool, are integrated in this layer to accomplish their functions. Moreover, it is the integration point also for the FIWARE Cloud platform.

- **FIWARE**: FIWARE is an open source platform that the European Union supports as a future platform to provide Cloud services. For this Use Case, the FIWARE tool can allocate several functionalities that goes from systems authentication, storage, DevOps tools integration. However, for the ENACT project the functionalities uses this Cloud platform to provide to the Use Case user several dashboard to track the Use Case functionalities (making use of the FIWARE ORION component for integration tasks and FIWARE Grafana component as the presentation tool).
- **S&P Mon&Con Back-end**: It is a service in a separate Cloud that supports the security monitoring capabilities. The traffic and security data treated in the S&P Monitoring tool is sent to this back-end to be evaluated. The service provides to the Use Case user with an interface to interact with the traffic behaviour and security rules that are desired for the On Board installation, as well as it offers the situational awareness functionalities for the user to get informed on the security status at all times.

10.4 DevOps of ITS System Powered by ENACT Tools

This section is intended to introduce the specific functionalities that the DevOps tools use and how these has been implemented and tested to enhance the Use Case itself.

10.4.1 Security Monitoring (S&P Mon&Con)

The Security and Privacy Monitoring and Control Enabler (for short, S&P Mon&Con) is in charge of providing the security layer that it is required for the Edge, as IoT technologies can suffer multiple types of attacks at this layer. The monitoring service of the tool is the one used in the Use Case. The S&P Mon&Con is the only tool that it not integrated into the Use Case at the Indra Hybrid Cloud Platform, as it acts as supervisor of the Use Case IoT system checking from outside whether resources at the Edge could be under compromise, thus, it has a parallel

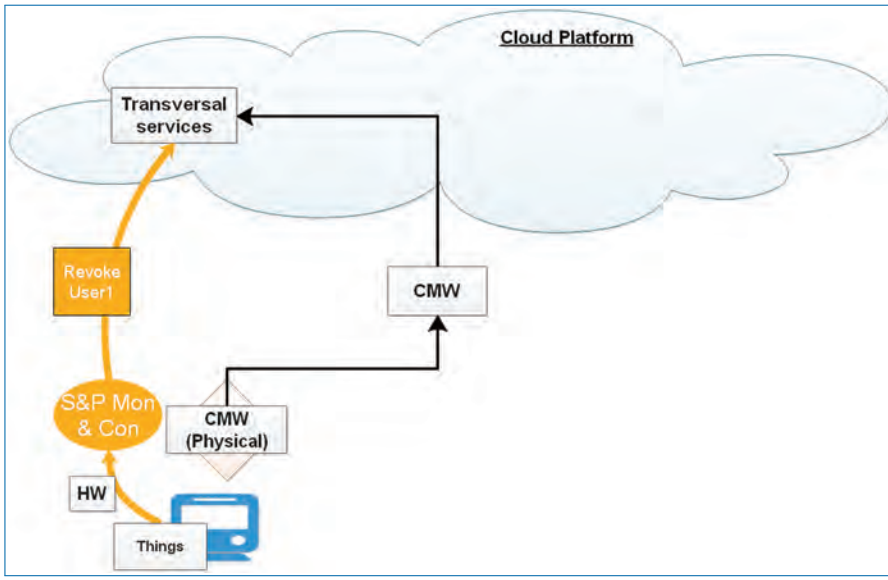

Figure 10.5. S&P – Rail Use Case integration synoptic.
Source: INDRA.

deployment and a different integration philosophy. A synoptic of the scenario can be seen in the Figure 10.5.

The tool is divided into two sections, the monitoring and the actuation part. The monitoring part is in charge of collecting the data that are going through the On Board gateways, this data includes the Things and the DMI data only. These data is monitored and sent throw a specific On Board device to the S&P Monitoring Back-end to be analysed. From this analysis the following parameters are analysed:

- **Traffic behaviour monitoring**: The traffic behaviour is considered as one of the characteristics that defines the rail functionalities is regular. The Safety and Security requirements that defines this kind of functionalities require a regular traffic that may be affected by intruders. The S&P tool is focus on detecting deviations in this edge layer traffic and revoking the user, from the Use Case central authentication servers, that generates that deviations.
- **Intrusion detection**: The authorised users are those systems that can publish/subscribe to IoT Platform (authenticated in the central Use Case authentication server), and are registered in the S&P Monitoring tool registry (both user id and Media Access Control (MAC) addresses). In case a user is authenticated in the central Use Case authentication server but not in the S&P Monitoring tool database, that user or/and the device they use will be revoked

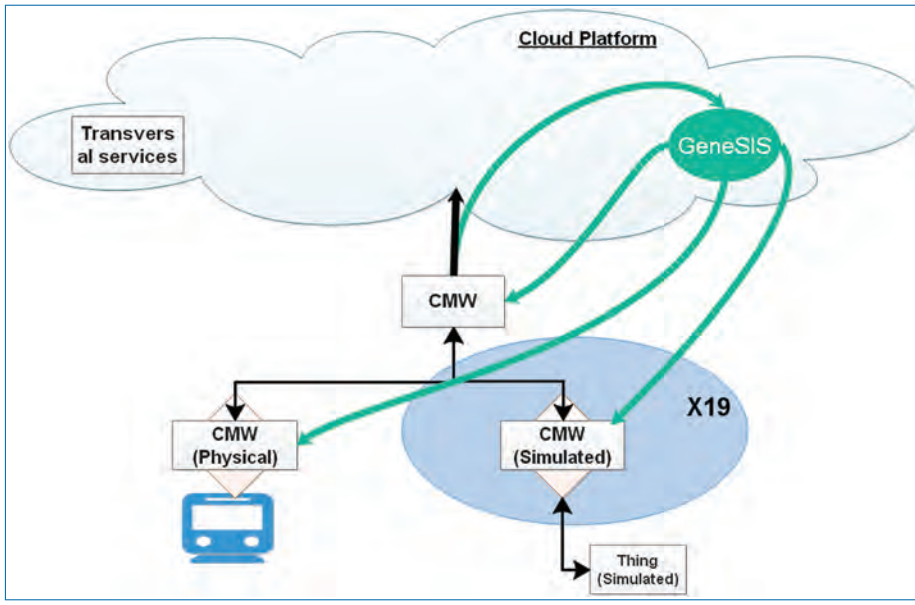

Figure 10.6. GeneSIS – Rail Use Case integration synoptic.
Source: INDRA.

in the Use Case central authentication servers, banning its connectivity to the whole IoT Platform.

10.4.2 Automatic Deployment – GeneSIS

The tool is in charge of managing and controlling the software that is running in the ITS Use Case.

It deploys the Docker images running in the gateways and ensures that they are correct and deployed with the access credentials requires to be integrated into the ITS infrastructure. A synoptic of the scenario can be seen in the Figure 10.6.

The tool also is aware of the status of the system deployed. The tool monitors the status of the Use Case infrastructure to check if it is possible making a deployment or not. In case the system is running the Use Case functionalities, the deployment cannot be performed. Moreover, the tool is able to evaluate if the deployment is correctly performed and if it has a conflict with previous deployments made in the same device.

10.4.3 Testing and Simulation

The testing and simulation tools has two roles into the ITS Use Case. The first one is monitoring the infrastructure in order to validate that the ITS infrastructure

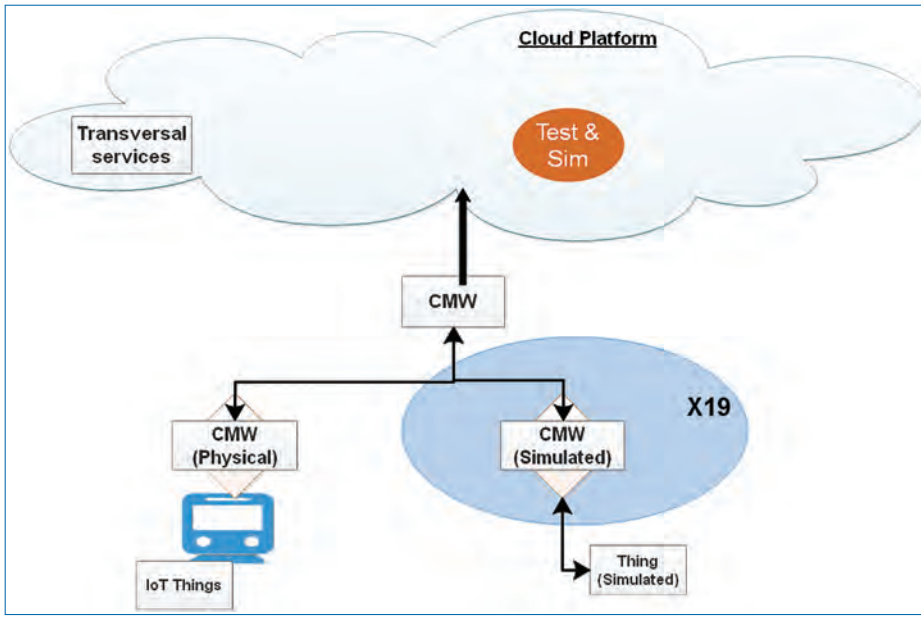

Figure 10.7. Testing and Simulation – Rail Use Case integration synoptic. *Source*: INDRA.

is running properly and to simulate failures into the infrastructure to enhance the validation value against issues that may appear into the operation of the Use Case itself. A synoptic of the scenario can be seen in the Figure 10.7.

The Testing and Simulation tool collects all the data gathered by the gateway layer to monitor its workload when it is operating the rail functionalities' tasks. In this case, all the traffic from the Edge and Cloud layers is monitored, in other words, the 100% of the traffic managed during an operation by the gateway.

Using these data the tool is able to replicate a single gateway, in a simulated virtual environment, taking as a basis the behaviour of the monitored gateway. This procedure that replicates the devices in a virtual infrastructure is called Digital Twin. Generating several Digital Twins as many times as desired provides the tool's user the possibility to generate a virtual scenario with all the gateways desired; hence, the scalability of a scenario can be proved. Moreover, as the virtual infrastructure is generated, the tool is able to simulate certain situations that may compromise the infrastructure and check its reliability.

This simulated environment is not enough to test the system's scalability. Several metrics to evaluate the simulated gateways are required. A specific report about the gateways status is required to know how the gateway is dealing with that monitored data. As it is mentioned in the Section 10.3.2, the gateways report specific data about its physical and routing status to evaluate their operation. Therefore, the tool

is able to infer, using these reports, the behavior of this simulated infrastructure in any situation proved.

10.4.4 Actuation Conflict Management (ACM)

The ACM tool is intended to manage conflicts that may appear in the train operation. This tool is based on generating behavioural models that can be applied to the On Board devices to fix an specific behaviour depending on the rail system inputs. A synoptic of the scenario can be seen in the Figure 10.8.

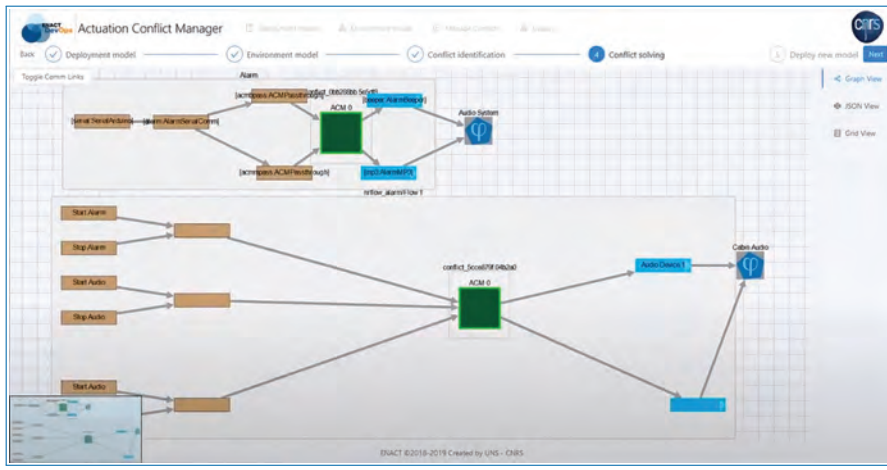

Figure 10.8. ACM – Rail Use Case integration synoptic.
Source: INDRA.

Based on the operation iterations, the developer can check the conflicts that may appear. The Safety and Secure functionalities defined are designed to not generate conflicts in the operation; however, the non-Safety systems that uses the safety ones may generate conflicts between their behaviours (e.g., audio announcements, alarms, etc.).

The ACM tool is able to generate models, as shown in the Figure 10.8, which can be deployed in any device that is the root cause of the conflict.

10.4.5 Behavioral Drift Analysis (BDA)

The BDA in charge of checking that the Things behaviour matches with the real modelled behaviour defined for them. The tool monitors the business data published by the Things to the Cloud layer to check in real time deviations with the mentioned model.

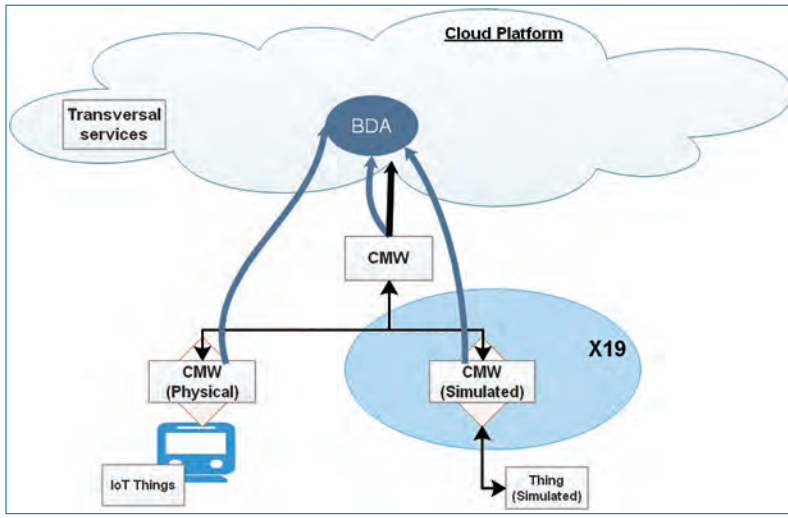

Figure 10.9. BDA – Rail Use Case integration synoptic.
Source: INDRA.

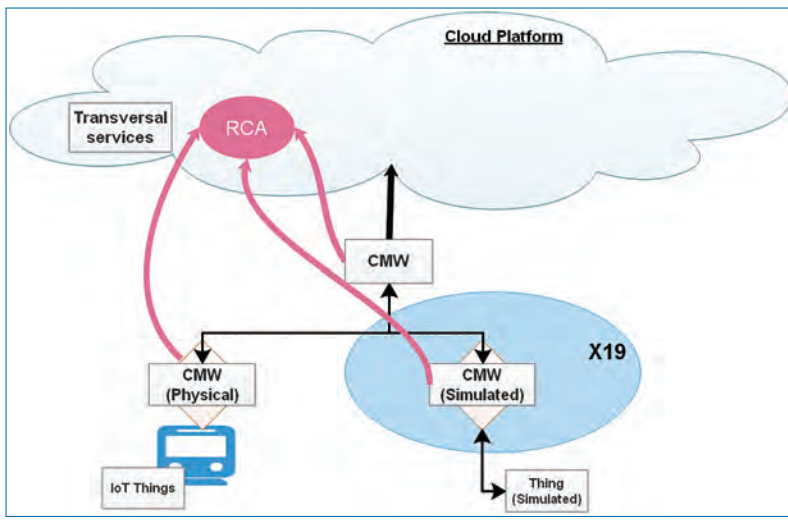

Figure 10.10. RCA – Rail Use Case integration synoptic.
Source: INDRA.

10.4.6 Root Cause Analysis (RCA)

The RCA tool is in charge of checking the security perspective of the ITS infrastructure from the gateway to the cloud layer finding the reasons for a failure that may occur (Figure 10.10). This tool serves to identify the cause of network mistakes that in a scalable systems is critical due to the quantity of devices deployed.

Several failure scenarios are simulated and recorded by the RCA tool in order to store the behaviour of the infrastructure in case of those failures appear. The data monitored is the physical and routing data reported by the gateways.

10.5 Conclusion

We have tested and validated – focused on developing trustworthy and ready-to-use scenarios to test the DevOps Enablers – the ENACT DevOps framework. From the Use Case perspective, the provision of an IoT platform with the characteristics exposed along the chapter makes a first step to enhance the rail environment and improves the new systems for the future digitalization and automation to an industry that is experiencing a huge evolution in the last decades.

The main impacts that the DevOps philosophy have on the rail environment, based on the mentioned DevOps tools that are listed along the document, are focus on:

- Ensuring the scalability of the system before deploying it. Introducing a Testing and Simulation tool permits the optimization of the infrastructure components and reduces the uncertainty about the system reliability. Therefore, the on-site testing time is reduced.
- Fast and agile deployment of new software versions in a remote manner, this reduces the infrastructure functionalities updated and the maintenance costs.
- Provision of a security backup to the internal rail authentication mechanisms covering since the devices to the application layers. The S&P tool locates the system intruders in a more accurate manner and the by design rail Security and Safety aspects are increased.
- Monitor the deviation in the behaviour of this systems in real time during the operation. The BDA tool helps to locate devices issues in a more accurate manner and, then, as the diagnosis time is improved the maintenance time and tasks is reduced.
- Ensure the interaction between the drivers and the rail functionalities to reduce the human errors. The main impact is the reduction of the failures that may occur in the system caused by the driver. This is highly relevant as the driver is a key factor to rely on the security and safety aspects in the rail environment.

Reference

[1] Indra Sistemas S.A. Contribution. *X2Rail-1 Deliverable D7.2 – Railway requirements and Standards application conditions – Indra Sistemas S.A Contribution*. Tech. rep. May 2018.

DOI: 10.1561/9781680838251.ch11

Chapter 11

Smart Building: The Tecnalia KUBIK Use Case

By Miguel Ángel Antón, Rubén Mulero, Sheila Puente, Larraitz Aranburu and Sarah Noyé

11.1 Introduction

Buildings have long been equipped with sensors and actuators to automate their control. Smart buildings are those whose facilities and systems (air conditioning, heating, lighting, access control systems, etc.) allow integrated and automated building management and control to increase energy efficiency, security, and usability. With the democratization of the Internet of Things (IoT), the number of sensors and actuators is constantly increasing, giving ways to new applications. The reduction of sensors and actuators cost is driving a digital shift in the building sector.

The need for better energy resource control and the requirement to provide better comfort for the user has led to a new market of complex Smart IoT Systems able to provide a vast array of new services or applications to the end-user. Extending legacy system to take advantage of those new services can be expensive, thus limiting possibilities. There is a need for a seamless way to integrate solutions from different manufacturers as well as to ensure effective design, deployment, and operation of simultaneous IoT applications that respect security and privacy requirements.

In that sense, software needs to be changed when new IoT devices are added or new functionalities for user comfort are developed. Therefore, it is necessary to

ensure the continuous development and update of the IoT applications. The thermal and climate control needs to be continuously adapted to the environmental changes to keep the occupants' comfort in the buildings. At the same time, potential conflicts between IoT applications acting on the same actuator or the same physical parameters must be identified to guarantee the buildings' trustworthiness. And finally, cybersecurity threats need to be identified and mitigated to preserve the security and data privacy.

By leveraging the ENACT DevOps framework, secure and trustworthy IoT applications can be developed based on the interoperability and orchestrated operation of multiple sensors and actuators.

The smart building KUBIK, situated close to Bilbao, Spain, was inaugurated in 2010 as an experimental infrastructure for developing and validating innovative products and systems to optimize energy efficiency in buildings [1]. It is a three floors building owned by Tecnalia and designed for testing and research ranging from passive systems such as modular insulating components for roofs and facades, to energy generation based on renewable energy and climate control systems. It includes more than 700 sensors and actuators, central Building Management System (BMS), local Renewable Energy Systems RES (RES), local weather station, and Combined Heat and Power (CHP) equipment on-site. In the context of the ENACT project, KUBIK provides the required equipment and well-known boundary conditions for the testing and validation of the enablers developed in the project.

KUBIK experimental infrastructure is relevant to the ENACT project due to its special needs and characteristics such as the combination of legacy building automation systems and new smart IoT devices, this fact requires an interoperability platform to communicate both systems. At KUBIK, several energy efficiency applications and user comfort applications share common actuators (fancoils, lights, blinds, controlled sockets, etc.) which generate actuation control conflicts. Thermal control of a building is also a trade-off between energy consumption and user comfort that must be adjusted to the specific physical characteristics of the building and user preferences. Behavioral drifts in the control of building systems also need to be identified and addressed. And finally, security and privacy of the communications is a must, paying special attention in secure actuation.

The ENACT enablers in combination with the SMOOL middleware platform have been used to solve the challenges described in the previous paragraph. Now, IoT applications are designed, developed and improved using the DevOps strategy, as ENACT enablers ensure no actuation conflicts, security (secure communications, access control, threat detection, etc.) and trustworthiness (self-learning controls, behavioural drifts identification, etc.) saving time and effort.

As a general result of the ENACT project, KUBIK building was leverage to become a place to develop new applications for energy efficiency and user comfort

for Smart Buildings that are based on IoT equipment. This exploitation of KUBIK as an infrastructure/testbed for the IoT domain is completely new. Now, the KUBIK building offers an environment where new IoT devices can be introduced and a wide range of IoT applications for smart buildings can be deployed and tested.

Section 11.2 describes the KUBIK building that was used as an experimental platform to validate the ENACT enablers for smart buildings. Section 11.3 gets into the detail of the technical architecture of the Smart IoT System of the KUBIK building. In Section 11.4, we expose different test scenarios and the benefit of the ENACT enablers. Finally, Section 11.5 concludes.

11.2 The KUBIK Smart Building

KUBIK is an experimental infrastructure focused on the development of new products and systems that provide energy consumption reduction for the building and increase user comfort (Figure 11.1). Its uniqueness lies in its ability to generate realistic scenarios to test energy efficiency resulting from the integration of constructive solutions, air conditioning and lighting systems, and energy supply from conventional and renewable energies. The building contains three floors with different testing zones and a cellar. Its ground floor is an apartment. It has a bedroom, a kitchen, a living room, and a corridor where engineers can test the Energy Efficient Building scenarios for a real home.

The ground floor has various IoT devices installed. Figure 11.2 shows the sensors and actuators installed at the ground floor. There is a flood sensor, sensors on doors

Figure 11.1. KUBIK by Tecnalia.

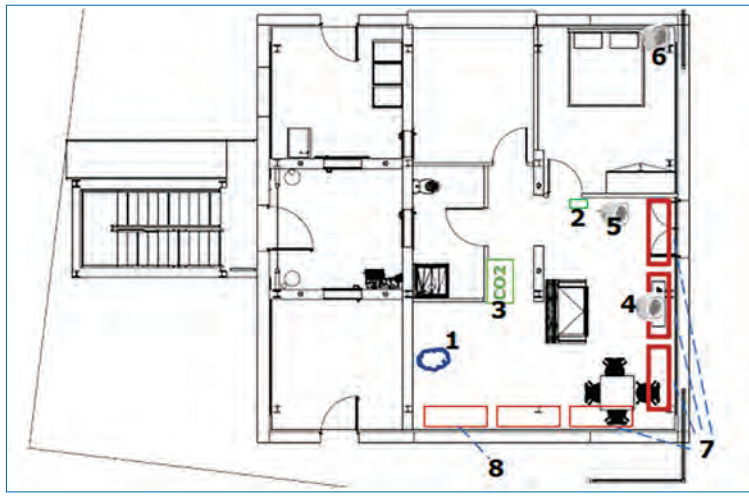

Figure 11.2. Floor plant of the Ground Floor of KUBIK.

Table 11.1. Device/Signals on the Ground Floor and Floor Plant of KUBIK Building. (K) values represents Kitchen, (B) values represents Bedroom, (L) value represents Living Room, (C) value represents corridor.

ID	Type	Device type	Location	System	Signal
1	Sensor	Water Flood	K	Z-Wave	Flood alarm state: ON/OFF
2	Sensor	Door Multisensor	B	Z-Wave	Position: OPEN/CLOSED
3	Sensor	CO2 Sensor 1	K/L/C	Z-Wave	CO2 level 0: 0 ppm to 200 ppm
4	Actuator	Remote Socket 1	L	Z-Wave	Switch state: ON/OFF
4	Sensor	Remote Socket 1	L	Z-Wave	Energy consumption: Watts
5	Actuator	Remote Socket 2	L	Z-Wave	Switch state: ON/OFF
5	Sensor	Remote Socket 2	L	Z-Wave	Energy consumption: Watts
6	Actuator	Remote Socket 3	B	Z-Wave	State: ON/OFF
6	Sensor	Remote Socket 3	L	Z-Wave	Energy consumption: Watts
7	Actuator	4 Blinds motors	L	PLC	Position: UP/Down
8	Actuator	2 Blinds motors	K	PLC	Position: UP/Down

that indicate open or closed status, various electrical sockets sensors and actuators, and motors for the blinds. Except for the blind motors, the devices are wireless sensors and actuators integrated as an additional layer to the building control system. Table 11.1 shows a detailed description of each device represented in the floor plan of the ground floor of KUBIK, its location in a specific room, its belonging to the IoT Smart Space or the wired Building Control group, and finally, the measures or commands it provides.

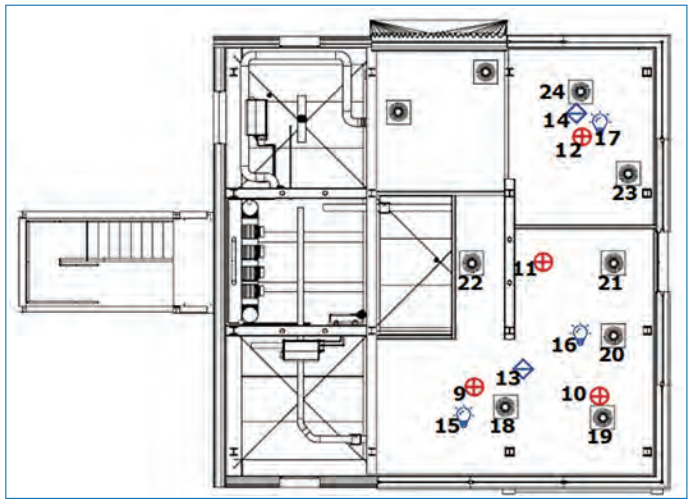

Figure 11.3. Reflected Ceiling plant of the Ground Floor of KUBIK.

Similarly, the reflected ceiling plan of the ground floor of KUBIK with sensors and actuators in their approximate location is shown in Figure 11.3. Those sensors are wireless Z-wave sensors monitoring ambient conditions (temperature, humidity, lighting, and occupancy) and smoke detectors. Sensors and actuators of the fan coil units of the space are connected to the building wired control system. Table 11.2 gives the details of different sensors and their signal types.

The cohabitation between hard-wired legacy sensors and easy-to-install additional wireless sensors presents an interesting case in line with the desire for flexibility and evolution of the smart building applications in the market.

11.3 Technical Architecture

A set of sensors, actuators, and devices are deployed inside each floor of the KUBIK building to capture real-time data and store it in a persistent environment. The stored data is analyzed to find potential solutions to automate different processes. Each connected device uses a standardized communication protocol to enable interoperability among them. A hub acts as a middleware between connected devices and external software programs and manages the communication protocol. The communication protocol may vary because of the installed devices' connectivity and cause additional complexity for large deployment scenarios. Therefore, it is necessary to create a general-purpose system to centralize the connections no matter what device type is connected.

Table 11.2. Devices/Signal on Ground Floor and Reflected Ceiling of KUBIK Building. (K) values represents Kitchen, (B) values represents Bedroom, (L) value represents Living Room, (C) value represents corridor.

ID	Type	Device type	Location	System	Signal
9	Sensor	Ceiling Multisensor 1	K	Z-Wave	Motion: YES/NO
9	Sensor	Ceiling Multisensor 1	K	Z-Wave	Temperature: degrees Celsius
9	Sensor	Ceiling Multisensor 1	K	Z-Wave	Light: 0 lux–1000 lux
9	Sensor	Ceiling Multisensor 1	K	Z-Wave	Relative humidity: 20%–95%
10	Sensor	Ceiling Multisensor 2	L	Z-Wave	Motion: YES/NO
10	Sensor	Ceiling Multisensor 2	L	Z-Wave	Temperature: degrees celsius
10	Sensor	Ceiling Multisensor 2	L	Z-Wave	Light: 0 lux – 1000 lux
10	Sensor	Ceiling Multisensor 2	L	Z-Wave	Relative humidity: 20%–95%
11	Sensor	Ceiling Multisensor 3	L/C	Z-Wave	Motion: YES/NO
11	Sensor	Ceiling Multisensor 3	L/C	Z-Wave	Temperature: degrees Celsius
11	Sensor	Ceiling Multisensor 3	L/C	Z-Wave	Light: 0 lux–1000 lux
11	Sensor	Ceiling Multisensor 3	L/C	Z-Wave	Relative humidity: 20%–95%
12	Sensor	Ceiling Multisensor 4	B	Z-Wave	Motion: YES/NO
12	Sensor	Ceiling Multisensor 4	B	Z-Wave	Temperature: degrees Celsius
12	Sensor	Ceiling Multisensor 4	B	Z-Wave	Light: 0 lux – 1000 lux
12	Sensor	Ceiling Multisensor 4	B	Z-Wave	Relative humidity: 20%–95%
13	Sensor	Smoke Detector 1	K/L/C	Z-Wave	Alarm state: ON/OFF
14	Sensor	Smoke Detector 2	B	Z-Wave	Alarm state: ON/OFF
15	Actuator	Ceiling light 1	K	PLC	Light state: ON/OFF
16	Actuator	Ceiling light 2	L	PLC	Light state: ON/OFF
17	Actuator	Ceiling light 3	B	PLC	Light state: ON/OFF
18	Actuator	Fan Coil 1	K	PLC	Operation state: ON/OFF
18	Actuator	Fan Coil 1	K	PLC	Temperature setpoint: Celsius
19	Actuator	Fan Coil 2	L	PLC	Operation state: ON/OFF
19	Actuator	Fan Coil 2	L	PLC	Temperature setpoint: Celsius
20	Actuator	Fan Coil 3	L	PLC	Operation state: ON/OFF
20	Actuator	Fan Coil 3	L	PLC	Temperature setpoint: Celsius
21	Actuator	Fan Coil 4	L	PLC	Operation state: ON/OFF
21	Actuator	Fan Coil 4	L	PLC	Temperature setpoint: Celsius
22	Actuator	Fan Coil 5	C	PLC	Operation state: ON/OFF
22	Actuator	Fan Coil 5	C	PLC	Temperature setpoint: Celsius
23	Actuator	Fan Coil 6	B	PLC	Operation state: ON/OFF
23	Actuator	Fan Coil 6	B	PLC	Temperature setpoint: Celsius
24	Actuator	Fan Coil 7	B	PLC	Operation state: ON/OFF
24	Actuator	Fan Coil 7	B	PLC	Temperature setpoint: Celsius

The devices installed in KUBIK use two different communication protocols. The first one is called Z-Wave,[1] a wireless communication protocol that integrates smart sensors inside a building. Z-Wave devices are widely used in domestic environments due to their ease of installation and low cost. However, its signal quality can be affected by interference and its battery level. The second one is called the MODBUS[2] communication protocol. This protocol is an industry-standard that is robust, fast, and secure. The connected devices using the MODBUS communication protocol need a central node called *industrial PC* or Programmable Logic controller.[3] They need to be connected by a physical connection (a wired cable). The MODBUS communication protocol is widely used in industrial processes to obtain information from machines and active processes. Thus, the main difference between a Z-Wave and a PLC device is that the former does not need any physical connection, and the latter requires a physical connection to an industrial PC. In terms of installation, Z-Wave-based devices are more convenient than a PLC device, but a PLC device offers robust connectivity and high reliable data speeds.

Having two different communication protocols to acquire data or perform actuation processes, we need to implement a middleware that enables interoperability among various sensors and actuators. The SMOOL IoT middleware that has a semantic broker for connecting heterogeneous devices or sources of information. In addition, the Building Management System also centralize all the information of new wireless sensors/actuators and the legacy building control systems of the building using a Scada software.

To address the challenges of sharing sensors/actuators between IoT applications that are running at the same time and also add a security layer to the data streams, the ENACT project provides tools and enablers to ensure the trustworthiness of the IoT applications in the KUBIK infrastructure. Figure 11.4 depicts the high-level architecture of the communications architecture in the KUBIK building.

The high-level architecture of the communications in KUBIK building is divided into three modules. The first module contains two subsystems: (i) **system 1**, the *wireless* system where each device is connected to a central node or *network hub* managing wireless connections; and (ii) **system 2**, a *wired* system where each device is physically connected to a central node or PLC device managing each wired connection. The second module contains the Building Management System (BMS) having three main elements: (i) a gateway device managing the connections between PLC/Z-Wave nodes, (ii) a persistent database that gathers the

1. https://www.z-wave.com/
2. https://modbus.org/
3. https://en.wikipedia.org/wiki/Programmable_logic_controller

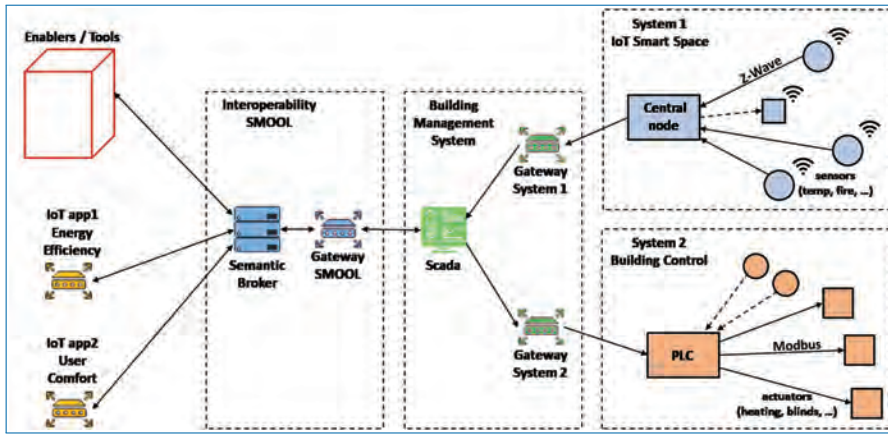

Figure 11.4. High level communications architecture in the KUBIK building.

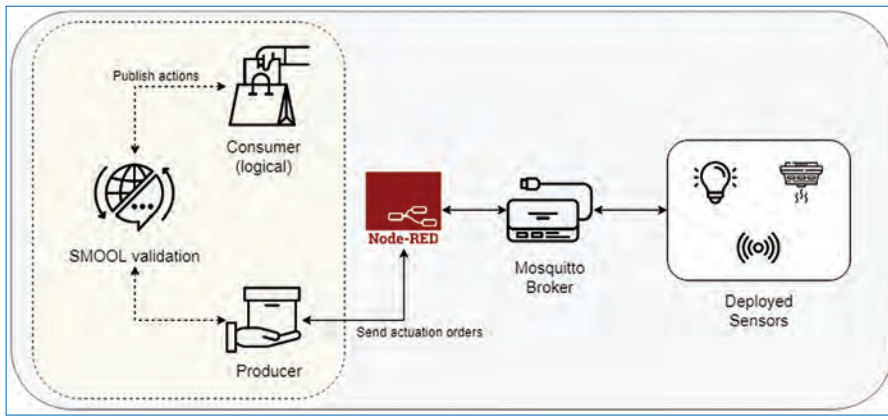

Figure 11.5. Technical components used in KUBIK architecture.

total connected devices and stores the data read by each device, (iii) a SCADA interface that graphically shows the current status of each deployed device and enables engineers to update device configurations. The last module is the SMOOL interoperabilty layer [2]. The entire system is managed by the decisions made according to a set of rules programmed in the IoT applications.

Figure 11.5 depicts the technical components used in the above architecture. The "Deployed sensors" part represents the acquisition process in which a set of sensors using both Z-Wave (system 1) and legacy PLC (system 2) obtain data measures to be sent to a middle device called "Mosquitto MQTT broker".[4]

4. https://mosquitto.org/

Figure 11.6. Jeedom configuration interface.

The SMOOL middleware has two components, i.e., the Producer and Consumer modules. The Producer module sends secure actuations when the Consumer module requires a security token. The Consumer module executes a set of expert rules using the obtained data from the Producer module. An expert encodes the expert rules. He/she decides which recommended actions to take when the data acquired from sensors meets a condition. For example, if the illumination sensors detect too much light, the actuators open the blinds for sunlight to get inside the KUBIK's living room.

At the low-level in the smart building Architecture (Figure 11.5), the data acquisition and actuation processes are managed by some hub systems called Gateways. These hubs are configured to allow direct communication between different sensors/actuators and a Mosquitto MQTT broker. Each hub uses its communication protocol to acquire or send actuation orders to the target device. For example, one hub is configured to manage only Z-Wave communications towards connected devices, while another hub is configured with the MODBUS communication standard. These hub systems use an internal operating system, JEEDOM,[5] that enables a graphical configuration of the connected devices and external services. In this regard, each hub is configured to make a direct connection to the Mosquitto MQTT broker. Figure 11.6 shows how JEEDOM is configured to send the data read from a set of Z-Wave devices directly to the Mosquitto MQTT broker.

Once the connection between the Z-Wave/PLC hub (JEEDOM) and Mosquitto broker is established, the next step is to program the Node-RED programming tool to develop a bridge between MQTT messages, SMOOL middleware and logic of IoT applications. Node-RED[6] is a visual flow-based programming environment

5. https://www.jeedom.com/site/en/index.html

6. https://nodered.org/

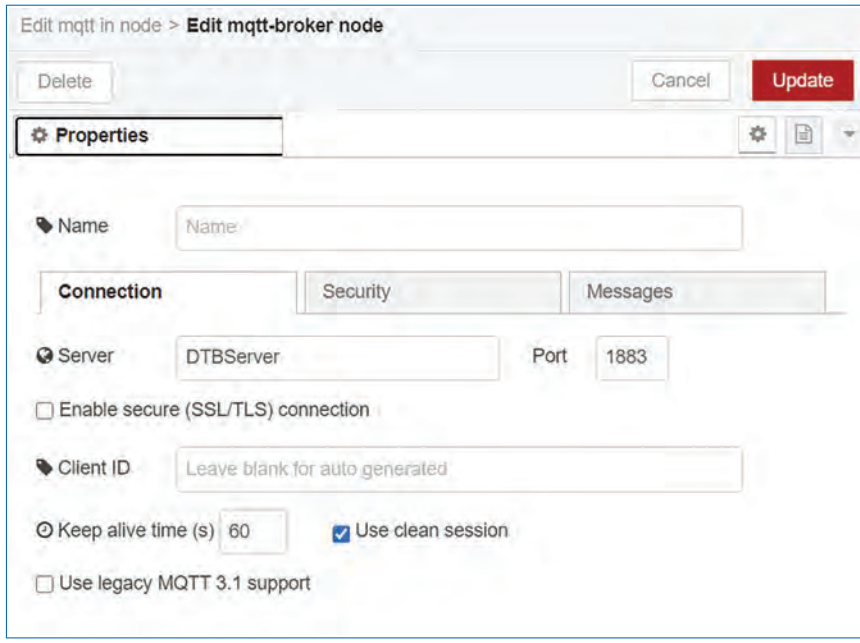

Figure 11.7. Node-RED programming to connect to Mosquitto MQTT Broker.

designed by IBM for the Internet of Things. Figure 11.7 depicts the configuration parameters needed to connect the Node-RED with the Mosquitto broker. These parameters enable the configuration of different workflows to make the required subscriptions to each connected device.

After having the logical connection between Node-RED and Mosquitto, it is necessary to program the required Node-RED flows for the data acquisition process with the SMOOL module. Figure 11.8 shows the acquisition process flow programmed in Node-RED. It reads the data from the Mosquitto broker and exposes it directly to its internal REST API module. The Producer SMOOL component reads the data exposed in the Rest API endpoints. Each connected line in Figure 11.8 represents a device inside the KUBIK building. There are several sensors deployed in the KUBIK building, and each one has its action flow. For convenience, Figure 11.8 presents only a minor part of the devices.

Figure 11.9 exhibits the secure actuation process provided by SMOOL middleware having two action flows: (i) one flow to send the available orders from Node-RED to SMOOL using a security token (top of the image) and (ii) another flow to sent the secure actuation orders checked in SMOOL to the actuator via Mosquito broker (low part of the image).

The flows are configured to enable the SMOOL components to send actions to the hub and write the needed information to perform the move up or move down actuation orders of the living room/kitchen blinds.

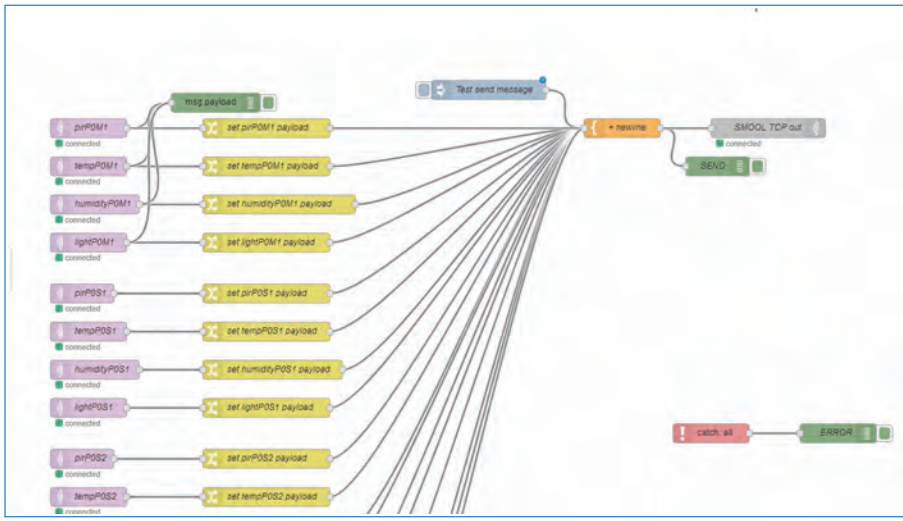

Figure 11.8. Node-RED flows to publish sensorized data in SMOOL.

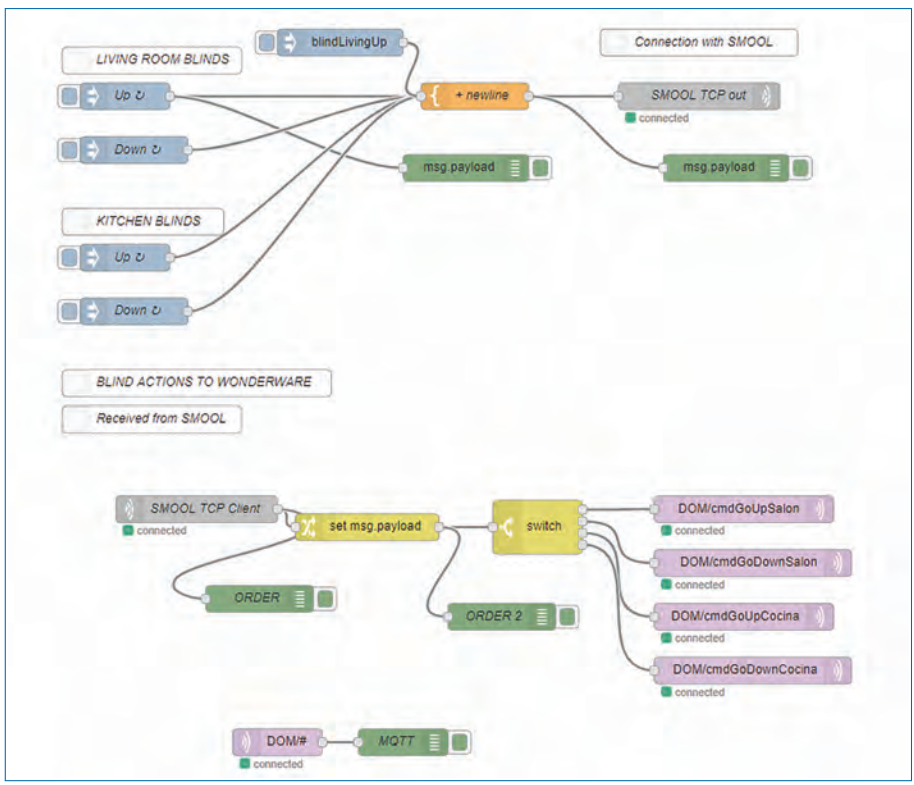

Figure 11.9. Node-RED flows to execute orders over devices in KUBIK through SMOOL.

11.4 KUBIK as an Experimental Platform for the ENACT Scenarios

In the ENACT project, the KUBIK building has been used as an experimental platform to test enablers in the *Smart Building* use case. Multiple scenarios have been implemented to test the ENACT tools relevant to this use case. We give the details of the scenarios and their main results in the following section. Those scenarios involve concurrent IoT applications related to energy efficiency and user comfort. These applications implement thermal comfort controls and smart building alerts that assure the safety of their occupants. They employ Z-wave and PLC systems described in the previous section.

11.4.1 Scenario 1: Thermal Comfort Control – Heating Design

Thermal comfort control is an essential smart building function. Sensors measure the users' comfort and enable the HVAC control system to keep the temperature at the level requested. By adding thermostat and temperature sensors with another protocol than the one of the HVAC system, it is easier to retrofit old systems and give more flexibility to deploy sensors. On the other hand, this strategy poses potential threats and risks to the thermal comfort system that must be identified and addressed.

One of the threats identified is when one of the thermal control devices is replaced with a similar device but not the same one. The Risk Management enabler is then used to analyse potential threats to the HVAC control system when new IoT devices are combined with legacy systems, provide the list of mitigation actions that the new IoT device needs to fulfil, and support the selection of security controls to minimize risks. By means of this enabler, HVAC control system designers and maintainers can also decide the risk level that that it is tolerable.

The retrofitting of old HVAC control systems is usually combined with changing the logic of the HVAC control program. In addition, the SMOOL middleware is used to communicate with new IoT devices and ThingML language can program those devices to define system behaviours and generation of executable code. The Orchestration and Continuous Deployment enabler, aka. GeneSIS, enables the continuous deployment and update of applications. In the ENACT project, GeneSIS has been fully integrated with SMOOL semantic middleware and ThingML language. In that way, SMOOL and ThingML are automatically deployed by GeneSIS as any other software component when adding new IoT devices and changing the HVAC control logic. The programming of the HVAC control logic, the communication characteristics of the new IoT devices in SMOOL and the

programming of ThingML devices are done as part of the same project in the same Integrated Development Environment (IDE).[7]

The Orchestration and Continuous Deployment enabler has also been used with the Actuation Conflict Manager enabler in Scenario 2.

11.4.2 Scenario 2: Thermal Comfort Control – Conflict in Heating Actuator Use

In the continuation of scenario 1, the integration of multiple systems can result in two or more applications sending different temperature preferences to the same heating actuator, which causes a fluctuating operation of the thermal control. The Actuation Conflict Management (ACM) enabler allows findings at design time the actuation conflicts that may lead to these fluctuations, and then it helps solving the conflict, thereby fixing the cause of the fluctuation.

A direct conflict occurs when two applications try to access the same node, e.g. an actuator. These two applications accessing the same actuator might send contradictory commands resulting in an indeterministic behaviour. An example of direct actuation conflict has been programmed and tested in the ACM enabler (see Figure 11.10). Figure 11.10 shows two IoT applications fed with temperature values from two different sensors. The applications try to change the actuator state when the temperature values reach a threshold that mimics the thermostat operation. The temperature sensors are deployed at different parts of the room, and thus different temperature values are likely obtained. The actuator behavior is similar to a relay that can switch ON and OFF the thermal heater.

Figure 11.11 shows a Node-RED flow with the implementation of the previously described scenario. In that flow, two identical subflows represent sensor processes that access the "command" node to send an actuation command to the same heating

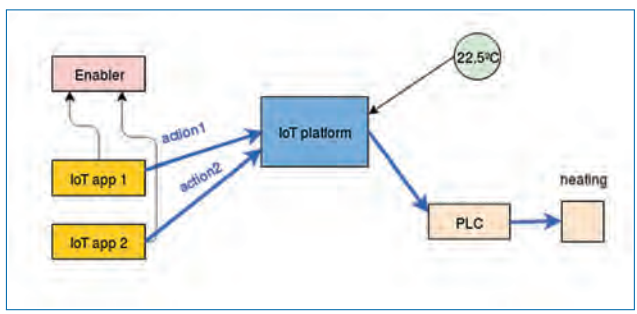

Figure 11.10. Thermal comfort control with conflict in heating actuator use.

7. Details about this integration can be seen in the following video https://www.youtube.com/watch?v=mfT_AwfkXNc

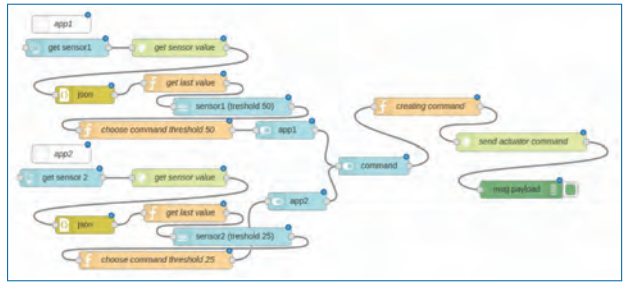

Figure 11.11. Node-RED flow example for thermal comfort control to be used by the ACM enabler.

actuator. The ACM enabler helps the software developer solve actuation conflicts occurring when two or more concurrent IoT applications try to send simultaneously conflicting actuation orders to the same actuator. In this scenario, a docker component that contains the Node-RED flows of the IoT application is deployed using GeneSIS. The ACM enabler then imports the model of the IoT system created by GeneSIS. GeneSIS enables the creation of an architecture and deployment model of the Smart IoT System by adding components and links. Although several model formats can be imported into the ACM enabler, Node-RED and GeneSIS are the main tools supported by the ACM enabler (Figure 11.11).

Once everything is imported and set up in the ACM enabler, a click on the "find conflicts" button automatically detects actuation conflicts and add a placeholder for actuation conflict management component in the model where conflicts might happen. The ACM enabler proposes several out-of-the-box components to solve a detected actuation conflict. The user chooses the right component, and the actuation conflict management placeholder is automatically replaced accordingly. Then, the IoT application is updated to become a conflict-free thermal comfort control system that can be redeployed using GeneSIS.

11.4.3 Scenario 3: Luminosity Comfort Control – Indirect Conflict in Luminosity Level Actuation

Two or more applications may also send actuation orders to different actuators that cause an indirect actuation conflict. An indirect actuation conflict affects the building thermal control of a building when two IoT applications managing two actuators act concurrently over the same physical variable, e.g. setting a high-temperature setpoint to the HVAC and opening a window. For instance, one app opens the window (lower temperature), and another one increases the setpoint temperature in the HVAC control system (higher temperature). An indirect actuation conflict can also occur in the room's luminosity level control when acting on lights and blinds. One switch controls the light to set it set ON or OFF, and another one controls the blind

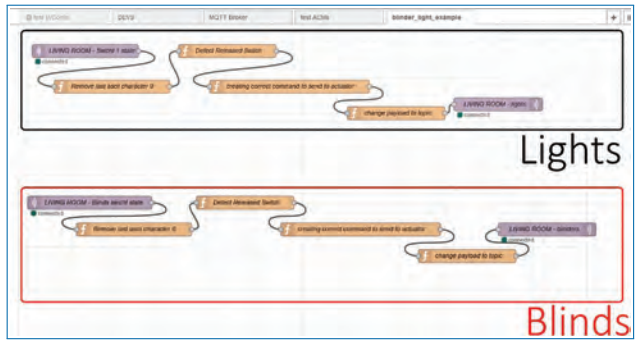

Figure 11.12. Node-RED flow representing the classic behaviour of the control of lights and blinds using switches.

to set it UP or DOWN. A physical model of the building is needed to find the indirect conflict on brightness, i.e. the impact of lights and blinds on the luminosity level.

The Actuation Conflict Management (ACM) enabler also corrects the operation of a physical system subject to uncertainties to deal with indirect conflicts. To achieve that, the physical system model is added to the enabler to detect and solve such indirect actuation conflicts. The ACM enabler imports the IoT system model created using GeneSIS and Node-RED tools. Once the model is imported, a physical process configuration is specified. The configuration allows establishing the interaction between a logical node and the environment. For instance, when there is an activation of a light, it is linked to a physical process representing luminosity. The utility of this process is to find an indirect conflict. If two different actions are linked to the same physical process, it may be an unplanned conflict to be solved.

In this scenario, the control of lights and blinds are associated with their corresponding switches using Node-RED. The classic behaviour of the system is that the lights are controlled by one switch to set it ON or OFF and the blinds are controlled by another switch to set it UP or DOWN (see Figure 11.12).

The ACM enabler then is fed with the previous described Node-RED flow for classic behaviour and the physical representation of the system to find actuation conflicts, i.e. the impact of lights and blinds on the luminosity level. Then, the ACM tool detects an actuation conflict in the luminosity level when we turn on the lights having sufficient brightness outside. The ACM tool creates a new Node-RED flow that resolves that conflict by adding new ACM components between the application logic and the actuation command (see Figure 11.13). The modified new Node-RED flow for luminosity level control can then be redeployed. The updated behaviour open the blinds instead of turning on the lights when the outside luminosity is high enough. Also in this sceanario, the Behavioural Drift Analysis

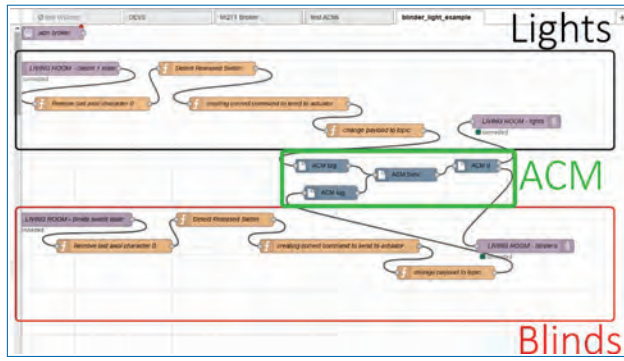

Figure 11.13. Node-RED flow example for solving the indirect conflict in luminosity level actuation.

(BDA) enabler was used to evaluate how much the observed behaviours of the IoT System are different from the expected ones.

11.4.4 Scenario 4: Smart Building Alerts for User Comfort

Data integrity and confidentiality must be ensured during data communication, especially if the communication is an important alert about equipment and user safety. Therefore, tampering with the alarms by external people should be avoided. The Security & Privacy Monitoring & Control (S&P Mon&Con enabler, cf. Chapter 7.1) enabler ensures the integrity and privacy of the communications through two services: (i) the Security and Privacy Monitoring for the surveillance of data security and (ii) the Security and Privacy Adaptation for data security enforcement.

This scenario addresses smart building alerts for IoT apps monitoring building aspects, such as an abnormally high or low temperature and smoke presence. The Security and Privacy Monitoring and Control (S&P Mon&Con) enabler ensures that the alarm system is not tampered with by external people and safeguard people's privacy. These enablers have been applied to the communication architecture in Figure 11.4.

The Security & Privacy Monitoring & Control enabler uses port mirroring to send a copy of the network packets that contain sensitive information of the smart building to a server where the enabler is running. Three different nodes in the smart building communication architecture have been mirrored: (i) the SMOOL communication broker, (ii) the Raspberry Pi used as a gateway by the SMOOL, and (iii) the SCADA node of the Building Management System of the KUBIK building.

When sensing information is transmitted, the S&P Mon&Con enabler checks if the messages are sent by the authorized nodes that belong to the KUBIK network. If an external device tries to send or receive a message to/from the monitored nodes, the S&P Mon&Con enabler notifies the system administrator to block network traffic if necessary. The S&P Mon&Con enabler also ensures the sensing data integrity by blocking any attempt of tampering with the message.

Scenario 4 addresses sensor devices that trigger alarms. These alarms may also trigger a security action or siren. The security actions consist of an ON/OFF command to a siren.

11.4.5 Scenario 5: Thermal Comfort Control – Self-optimizing Controller Design

The thermal comfort control of a building is a trade-off between user comfort and energy consumption, e.g., the heating is off when the room or building is not occupied. Monitoring the building's thermal inertia and real-time occupation can significantly improve energy efficiency and comfort compared to traditional controllers. We used the Online Learning enabler (OLE) to find and update the optimal control parameters at run time. We employed GeneSIS to properly deploy those control parameters and orchestrate the involved IoT devices.

The Online Learning (OLE) enabler is a module with an agent-based Artificial Intelligence (AI) algorithm that performs actions based on the sensors readings gathered from the KUBIK building infrastructure. This enabler aims to provide a thermal comfort solution that reduces the energy costs in the building. We have developed a thermal model of the ground floor of the KUBIK building to implement the OLE. We have demonstrated the potential for energy saving and increased thermal comfort.

11.5 Conclusion

Buildings are becoming increasingly smart, and new IoT tools are needed to ensure the safe and trustworthy operation of the different services they offer. The KUBIK experimental building is a smart building testbed for the ENACT project tools.

We successfully tested the ENACT enablers that enable continuous deployment, solving actuation conflicts, ensure security and privacy of the communications, identify risks at design time, correct behavioral drifts, enforce security and privacy, and self-optimizing controller.

References

[1] José A. Chica *et al.* "Kubik: Open Building Approach for the Construction of an Unique Experimental Facility Aimed to Improve Energy Efficiency in Buildings". In: *Open House International* 36.1 (Jan. 2011), pp. 63–72. ISSN: 0168-2601. DOI: 10.1108/OHI-01-2011-B0008.

[2] Adrian Noguero, Angel Rego, and Stefan Schuster. "Towards a smart applications development framework". In: *Social Media and Publicity* 27 (2014), pp. 2011–2020.

DOI: 10.1561/9781680838251.ch12

Chapter 12

Looking Ahead

By Andreas Metzger, Cristóbal Costa Soria, Juan Garbajosa, Ana M. Moreno, Daniel Pakkala, Jukka Rantala, Valère Robin, Jukka Saarinen, Bjørn Skjellaug, Hui Song, Mike Surridge, Tuomo Tuikka, Josef Urban and Thorsten Weyer

12.1 Introduction

In this book, we reported the main outcomes of the EU Horizon 2020 project ENACT. ENACT developed a toolkit that facilitates the development and operations (DevOps) of trustworthy Smart IoT Systems.

Concluding this book, we look ahead and offer perspectives on future research and innovation opportunities in the area of Smart IoT Systems. These future research and innovation opportunities are based on research challenges that were jointly developed with partners from the European Technology Platform *NESSI* (Networked European Software and Services Initiative [1]). These research challenges were contributed as input to the forthcoming European Key Digital Technologies (KDT) Partnership. We thus use the term *KDT applications* as an umbrella term for Smart IoT Systems, and thereby include closely related areas such as embedded systems, cyber-physical systems and edge-based systems.

12.2 Research and Innovation Opportunities

12.2.1 Software-driven Integration of KDT Applications

Without interaction with each other, KDT applications are information silos, which become an obstacle for potential business value creation. Software-driven integration, *i.e.*, developing new software to integrate existing applications, is a trend in the software industry. Mainstream cloud applications already underwent platformization, *e.g.*, Facebook or Salesforce are now platforms that allow third parties to develop and offer value-added services.

Because platformization of KDT applications requires offering powerful APIs to external components and systems, new research and innovation into such platforms is needed. Platformization also requires novel kinds of software architectures within the applications to achieve the flexibility for deep customization. Supporting integration is challenging because custom code will share the already constrained resource of electronic components and may also bring vulnerability to applications that are security- or safety-critical. New design methods are needed, with performance and vulnerability assessment considering resource and hardware aspects, together with novel isolation mechanisms on low-level electronic components, potentially supported by virtualization techniques.

KDT applications also need to be integrated with traditional enterprise and consumer software applications, since the latter are currently managing the data and business processes. Such integration introduces electronic components into the traditional human-data interaction, and thus calls for novel software-hardware co-design in an agile and continuous way, in order to bridge the social-cyber-physical sensing powered by new electronic components with the business data and process controlled by the traditional enterprise software systems. This integration with traditional enterprise and consumer software applications, also require a higher-level analysis of vulnerability and risks. Organizations try to run high level business processes over inefficient digital ecosystems, made out of different parts, some of them new, some of them legacy-based, and in many cases non-interoperable. There is a lack of mechanisms to collect information about these processes (transversal analytics) and control risks from data pieces coming from different hardware and software components.

Software integrators play an important role in the software industry and in the software value chain. New abstractions, orchestration languages and integration methodologies are needed to facilitate the interdisciplinary thinking of integrators. Future research and innovation actions in this direction require close collaboration among software engineering researchers, application providers, software integrators and the integration platform providers, resulting in the extension of mainstream

iPaaS (integration Platform as a Service) solutions to reach the lower-end devices. Platformization and integration are also important for KDT application providers to develop ecosystems involving component providers, software vendors and integrators. Research and innovation is needed to investigate business models, legal issues, data sharing strategies, etc., in order to integrate scattered businesses into prosperous European KDT ecosystems.

12.2.1.1 Managing complexity, dynamics, and uncertainty of KDT applications

Digital systems and groups of collaborating systems, as well as the environments in which these systems operate, are becoming more complex, show highly dynamic behaviour, and increasingly face uncertainty during operation. The trend toward digitalization is accompanied by a significant increase in complexity, dynamics and uncertainty. The increasing complexity can be seen in individual KDT systems, groups of collaborating systems, as well as the environments in which a particular KDT system operates.

Managing the dynamics of systems, collaborating groups and their environment poses an important challenges for the engineering KDT applications. Furthermore, KDT applications increasingly face uncertainty during runtime, *i.e.*, KDT applications have incomplete or ambiguous information about the environment in which they operate. This is especially the case when systems are operating in open contexts where the relevant properties of the environment cannot be completely anticipated at design-time, and therefore cannot be fully handled by predefined adaptations. In many future scenarios, such as autonomous driving or smart factories, systems must be able to meet their goals even on the basis of incomplete or contradictory information about the environment, *e.g.*, the intentions of other systems or humans, or the preferences and skills of human users.

While ENACT has indicated how – via machine learning at runtime (see Chapter 6) – a KDT application may capture uncertainties in the environment, additional research is needed to answer questions such as how to guarantee appropriate and safe cognitive adaptability in complex, highly dynamic and uncertain environments, and how to verify – at runtime and under hard real-time constraints – emergent system behavior resulting from the interactions of subsystems and the ambiguity faced in such environments.

A central challenge will be the modelling and implementation of human-machine interactions under those conditions; for example the efficient and effective transfer of control between systems and human users to avoid effects of mode confusion in autonomous driving. Developing software for complex and dynamic systems, able to deal with uncertainties, will require engineering processes involving multiple disciplines such as cognitive science and sociology.

Promising approaches to deal with the increasing complexity, dynamics and uncertainty must consider both design-time and runtime. Such approaches will be based on innovative combinations and improvements of software technologies in the following categories:

- innovative technical solutions in the area of software technology including design-time and runtime techniques for collaborative information fusion;
- collaborative runtime verification (including digital twin technology);
- environment perception with shared models of the environment;
- prediction of future behaviour and cognitive adaptability of individual systems, collaborative groups and the environment.

Innovative process-related solutions in software technology should include approaches from design science, agile methods, and the creation of appropriate team culture. Design science offers a disciplined approach to analyze a problem in a real-world context, systematically derive solutions (in the form of software artefacts and prototypes) and validate and evaluate them in order to generate new knowledge [2]. In design science digital solutions are produced and studied in an operational application context (real world use context), where the maturity, quality and value of the solutions can be evaluated. The solutions often involve combinations of evolving operational processes, software, hardware and ICT systems governed by multiple organizations, which creates a complex context for design and deployment of new digital solutions. At design- time, the related engineering activities, including software engineering, need to be well aligned with the overall solution goals and requirements, and hardware and network requirements and limitations, as well as business requirements, need to be considered in parallel with the software engineering process. At runtime, the resulting technical system (or system-of-systems) may be widely distributed across different embedded, networked and cloud computing nodes governed by multiple different organizations. Accordingly, to discover ways to manage the complexities involved in design and deployment of new digital solutions, the role of governance boundaries and multi-organization collaboration, both at design-time and runtime, deserve further research.

12.2.2 Leveraging Spatial Computing for KDT Applications

Spatial Computing is an emerging interaction mechanism for digital content in a converged cyber-physical world. Advances in devices and user interfaces (*e.g.*, mixed reality glasses, gesture recognition, haptic feedback, interfaces built from new materials) and their integration into a spatial computing system will allow for more adaptable, responsive and immersive interactions with the digital world. The services offered by the spatial computing system will provide new ways of

augmenting user experience and will allow us to 'feel' with all senses the virtual environment around us. Just like the real word, spatial computing offers a rich environment for multi-user interaction, and empowers a human-centric approach for future digitalisation. Taking industrial automation as an example, the introduction of spatial computing will help put human needs and interests in a central role, focusing not only on how to automate and optimize the production process, but also on how to get workers involved in the process.

Spatial computing systems will be complex constellations of software and content components operated by a multitude of ecosystem participants. The intelligence required for smart interaction mechanisms will depend on collecting and analysing massive amounts of social cyber-physical sensing data across all stakeholders in the ecosystems into digital super twins. The required computing power will not be provided by the involved devices only, and thus limited to their capabilities – it will be provided by the cloud or at the edge also, offering additional and typically more powerful capabilities.

Software engineering approaches for designing and developing spatial computing systems will need to cope with the challenging environment of diversity in devices and application domains, with intelligent deployment and adaptation capabilities. New programming models, languages and methodologies will emerge in the spatial computing era, and research and innovation actions will help to create breakthrough progresses. This will require interdisciplinary research integrating advances in media technologies (new coding technologies for digital content), cognitive psychology (new human-machine interaction for better attention and perception from the users), social science (new methods and processes for better human collaboration), etc.

12.2.3 Sustainable and Energy-efficient KDT Applications

Although digital technologies and software may provide very powerful tools to optimize the energy efficiency in vertical domains, their absolute and relative energy and resource consumptions continue to increase, even if hardware itself improves (notably for embedded systems as a by-product of autonomy optimisation or to reduce the cost of big data centres). The increasing functional scope of software and applications, the introduction of data intensive algorithms and systematic logging of events, the use of complex middleware stacks (hypervisors, virtual machines, containers, languages runtimes, bloated framework) all contribute to the environmental impact of software-based systems, often sacrificing frugality for the sake of ease of development and time-to-market.

The sustainability concern needs to be natively addressed in the development and execution phase of all digital systems (embedded, personal, large-scale,

communication equipment, etc.). New tools and models are needed to optimize the interactions between hardware and lower software layers, to adapt to runtime context, and continuously minimize energy and resource consumption, based on monitoring not only the internal behaviour of software systems but also the external physical environment through advanced sensing and learning.

Sustainability also calls for simplified and efficient architectural patterns, along with the appropriate education of key actors such as developers, software architects, system integrators and data centre management teams. Interdisciplinary research will provide novel solutions towards sustainable digital systems, *e.g.*, the use of energy harvesting from the environment (wind, solar, pressure, etc.) or other energy sources (body heat, foot strikes, etc.) to power lower-end devices, making them battery-free. This also calls for new programming models, software architectures and self-adaptation approaches to cope with novel and potentially unstable power sources.

12.3 Conclusion

The ENACT project paved the way towards bringing established software engineering processes and tools to the realm of Smart IoT Systems and KDT applications. As indicated above, this is a mere start and challenging research and innovation opportunities are ahead of us. Jointly addressing these challenges will contribute to European companies, SMEs, and research institutes to remain competitive in this traditionally strong area of Europe.

References

[1] NESSI ETP. *Software and Key Digital Technologies*. Networked European Software and Services Initiative, Brussels. (http://www.nessi.eu/)

[2] R. Wieringa. *Design Science Methodology for Information Systems and Software Engineering* Springer, 2014.

Index

About the Editors

Nicolas Ferry is an Associate Professor at University Côte d'Azur. Prior he was a Senior Research Scientist at SINTEF. He holds a Ph.D. degree from the University of Nice. His research interest includes model-driven engineering, domain-specific languages, Internet of Things, cloud-computing, self-adaptive systems, and dynamic adaptive systems. He has actively contributed to various national and international research projects such as the REMICS, CITI-SENSE, MC-Suite and MODAClouds EU projects, and is the technical manager of the H2020 ENACT project. He has also served as a program committee member of international conferences and workshops.

Erkuden Rios received the B.S. and M.S. degree in telecommunication engineering from University of Basque Country, Bilbao, Spain, in 1997. In 2020 Erkuden has received a Ph.D. in multi-Cloud security assurance from University of Basque Country. After working six years for Ericsson Spain, she is currently senior scientist of Cybersecurity research team of ICT Division in Fundación Tecnalia Research & Innovation, Derio, Spain. She is currently the coordinator of the Security WP in the H2020 ENACT project on Secure and Privacy-aware Smart IoT Systems as well as in the H2020 SPEAR project on Secure Smart Grids. Previously, she was the coordinator of the H2020 MUSA project on Multi-cloud Security as well as the chair of the Data Protection, Security and Privacy in Cloud Cluster of EU-funded research projects, launched by DG-CNECT in April 2015. (https://eucloudclusters.wordpress.com/data-protection-security-and-privacy-in-the-cloud/).
Furthermore, she has worked in multiple large European and Spanish projects on cybersecurity and trust such as POSEIDON, PDP4E, TACIT, RISC, ANIKETOS, SWEPT, CIPHER and SHIELDS. Her main research interests include Trust and Security, Risk Management, and AI for Cybersecurity. Mrs. Erkuden collaborates with Technology Platforms and Forums such as Cybersecurity PPP ECSO, ETSI Secure Artificial Intelligence Working Group, AIOTI WG4 Policy and Privacy and

the Spanish National Network on Cybersecurity. She has been member of Programme Committees of Journals and Conferences.

Andreas Metzger is senior academic councilor at the University of Duisburg-Essen and heads the Adaptive Systems and Big Data Applications group at paluno, the Ruhr Institute for Software Technology. He holds a Ph.D. in computer science from the Technical University of Kaiserslautern. His background and research interests are software engineering and machine learning for self-adaptive systems. Among other leadership roles, he was technical coordinator of the European H2020 lighthouse project TransformingTransport and work package leader in the H2020 ENACT project. Andreas serves as steering committee vice chair of NESSI, the European Technology Platform dedicated to Software, Services and Data, and as deputy secretary general of the Big Data Value Association (BDVA/DAIRO).

Hui Song is a senior researcher with SINTEF Digital, Norway. He has a Ph.D. in computer science from Peking University in China. His research interests include software engineering methods and tools, and their applications on developing cloud and IoT systems. He has contributed to a number of European and Norwegian research projects, and is the coordinator of the H2020 ENACT project.

Contributing Authors

Miguel Ángel Antón
TECNALIA, Basque Research and Technology Alliance (BRTA), Donostia-San Sebastián, Spain
mangel.anton@tecnalia.com

Larraitz Aranburu
TECNALIA, Basque Research and Technology Alliance (BRTA), Donostia-San Sebastián, Spain

Franck Chauvel
SINTEF Digital, Oslo, Norway
franck.chauvel@sintef.no

Tommaso Crepax
KU Leuven, Belgium
tommaso.crepax@kuleuven.be

Rustem Dautov
SINTEF Digital, Oslo, Norway
rustem.dautov@sintef.no

Franck Dechavanne
Université Côte d'Azur, I3S/CNRS Sparks, Sophia Antipolis, France
franck.dechavanne@sintef.no

Jacek Dominiak
Beawre, Barcelona, Spain
jacek.dominiak@beawre.com

Felix Feit
paluno – The Ruhr Institute for Software Technology University of Duisburg-Essen, Essen, Germany
felix.feit@paluno.uni-due.de

Nicolas Ferry
Université Cote d'Azur, I3S/INRIA Kairos, Sophia Antipolis, France
nicolas.ferry@univ-cotedazur.fr

Franck Fleurey
Tellu, Asker, Norway
franck.fleurey@tellu.no

Anne Gallon
Evidian – ATOS, France
anne.gallon@evidian.com

Juan Garbajosa
UPM, Spain

Thibaut Gonnin
Université Côte d'Azur, I3S/CNRS Sparks, Sophia Antipolis, France
thibaut.gonnin@univ-cotedazur.fr

Elena Gonzalez-Vidal
Beawre, Barcelona, Spain
elena.gonzalez-vidal@beawre.com

Modris Greitans
Institute of Electronics and Computer Science (EDI), Riga, Latvia
modris_greitans@edi.lv

Christophe Guionneau
Evidian – ATOS, France
christophe.guionneau@evidian.com

Vinh Hoa La
Montimage, Paris, France
vinh.hoala@montimage.com

Eider Iturbe
TECNALIA, Basque Research and Technology Alliance (BRTA), Derio, Spain
eider.iturbe@tecnalia.com

Sergio Jimenez Gomez
Indra Sistemas S.A, Madrid, Spain
sjimenez@indra.es

Janis Judvaitis
Institute of Electronics and Computer Science (EDI), Riga, Latvia
janis.judvaitis@edi.lv

Stéphane Lavirotte
Université Côte d'Azur, I3S/CNRS Sparks, Sophia Antipolis, France
stephane.lavirotte@univ-cotedazur.fr

Wissam Mallouli
Montimage, Paris, France
wissam.mallouli@montimage.com

Saturnino Martinez
TECNALIA, Basque Research and Technology Alliance (BRTA), Derio, Spain
saturnino.martinez@tecnalia.com

Andreas Metzger
paluno – The Ruhr Institute for Software Technology University of Duisburg-Essen, Essen, Germany
andreas.metzgger@paluno.uni-due.de

Yuliya Miadzvetskaya
KU Leuven, Belgium
yuliya.miadzvetskaya@kuleuven.be

Guillaume Mockly
Trialog, Paris, France
guillaume.mockly@trialog.com

Edgardo Montes de Oca
Montimage, Paris, France
edgardo.montesdeoca@montimage.com

Ana M. Moreno
UPM, Spain

Rubén Mulero
TECNALIA, Basque Research and Technology Alliance (BRTA), Donostia-San Sebastián, Spain

Victor Muntés-Mulero
Beawre, Barcelona, Spain
victor.muntes-mulero@beawre.com

Luong Nguyen
Montimage, Paris, France
luong.nguyen@montimage.com

Phu Nguyen
SINTEF Digital, Oslo, Norway
phu.nguyen@sintef.no

Sarah Noyé
TECNALIA, Basque Research and Technology Alliance (BRTA), Donostia-San Sebastián, Spain

Daniel Pakkala
VTT, Finland

Alexander Palm
paluno – The Ruhr Institute for Software Technology University of Duisburg-Essen, Essen, Germany
alexander.palm@paluno.uni-due.de

Francisco Parrilla
Indra Sistemas S.A, Madrid, Spain
fparrilla@indra.es

Sheila Puente
TECNALIA, Basque Research and Technology Alliance (BRTA), Donostia-San Sebastián, Spain

Jukka Rantala
Nokia, Finland

Angel Rego
TECNALIA, Basque Research and Technology Alliance (BRTA), Derio, Spain
angel.rego@tecnalia.com

Erkuden Rios
TECNALIA, Basque Research and Technology Alliance (BRTA), Derio, Spain
erkuden.rios@tecnalia.com

Valère Robin
Orange, France

Gérald Rocher
Université Côte d'Azur, I3S/CNRS Sparks, Sophia Antipolis, France
gerald.rocher@univ-cotedazur.fr

Jukka Saarinen
Nokia, Finland

Arezki Slimani
Evidian – ATOS, France
arezki.slimani@evidian.com

Arnor Solberg
Tellu, Asker, Norway
arnor.solberg@tellu.no

Cristóbal Costa Soria
ITI, Spain

Bjørn Skjellaug
SINTEF Digital, Oslo, Norway

Hui Song
SINTEF Digital, Oslo, Norway
hui.song@sintef.no

Mike Surridge
IT Innovation, UK

Tuomo Tuikka
VTT, Finland

Jean-Yves Tigli
Université Côte d'Azur, I3S/CNRS Sparks, Sophia Antipolis, France
jean-yves.tigli@univ-cotedazur.fr

Josef Urban
Nokia Bell Labs, Germany

Thorsten Weyer
paluno – The Ruhr Institute for Software Technology University of Duisburg-Essen, Essen, Germany

Oscar Zanutto
ISRAA, Italy
faber@israa.it