INTEGRATION OF THE MAPE-K LOOP IN DIGITAL TWINS

Hao Feng
Cláudio Gomes
Santiago Gil
Peter H. Mikkelsen
Daniella Tola
Peter Gorm Larsen

DIGIT
Department of Electrical and Computer Engineering
Aarhus University
Åbogade 34, Aarhus N, DENMARK
{haof,claudio.gomes,sgil,phm,dt,pgl}@ece.au.dk

Michael Sandberg

Department of Mechanical
and Production Engineering
Aarhus University
Åbogade 34, Aarhus N, DENMARK
ms@mpe.au.dk

ABSTRACT

In recent years the concept of Digital Twins (DTs) has gained significant attention, in particular in the manufacturing industry. In proper DTs there is a bidirectional connection to/from their Physical Twins such that it is possible to leave humans out of the loop. However, the underlying ideas of DTs have been inspired from a number of different scientific communities, in particular in relation to autonomy including the "Monitor-Analyze-Plan-Execute over a shared Knowledge" (MAPE-K). In this paper, we bridge the two concepts by demonstrating an implementation of a MAPE-K loop for an incubator case study within the context of its DT. We also discuss some of the limitations of the MAPE-K concept when applied to Cyber-Physical Systems (CPS).

Keywords: Digital twins, self-adaptation, MAPE-K loop, automation.

1 INTRODUCTION

A DT is a virtual representation of a physical counterpart (the Physical Twin (PT)), where bi-directional communication endows a DT with the power to act on the physical system without human intervention. Digital representations with unidirectional communication only are called Digital Shadows (DSs) (Schuh et al. 2020). In practice, a DS needs a human to implement or intervene in control or other decisive actions, while a true DT does not necessarily. Based on this ability, a DT has the potential to develop a self-adaptation feature that allows an evolutionary strategy to adapt itself to a general class of problems, by reconfiguring itself accordingly, and to do this without any user interaction (Back 1996). As such, the concept of DT and the concept of self-adaptive systems (Weyns 2019, Chen et al. 2018) share the same goals.

This paper is motivated by our previous work, (Feng et al. 2021), in which we constructed a DT of an incubator system and proposed a self-adaptation loop as future work. In this paper, we report on the implementation of a self-adaptation loop using a MAPE-K feedback loop, which is the most influential reference model for self-adaptive systems (Arcaini et al. 2015). We also use UML diagrams to showcase the detailed

architecture of the self-adaptation loop within the DT. Our vision is that this architecture can be integrated into other DT frameworks and become more versatile. Finally, we discuss, with concrete examples, the limitations of the MAPE-K loop with respect to system safety.

The rest of the paper is organized as follows: Section 2 briefly introduces the concepts of DTs together with selected DT frameworks, followed by a short description of the incubator DT. Section 3 gives the details of achieving the self-adaptation loop through the MAPE-K feedback loop and uses UML diagrams to show how the self-adaptation loop is integrated into the incubator DT. Finally, section 4 concludes with discussions and proposals for future work.

2 BACKGROUND

2.1 Introduction to Digital Twins

Various definitions of DTs have been proposed (Liu et al. 2021, Rasheed et al. 2020), however, since it was first presented by Michael Grieves in 2003 (Tao et al. 2019), none of them have rigorously defined a DT. We view a DT as a system that integrates bidirectional communication of models and services to increase the value of the physical system. Notice that a DT is not equivalent to a model, nor to a simulation; models and simulations play an important role in a DT.

The concept of DTs is mainly attributed to manufacturing, but it has also been explored in other industries, such as agriculture, energy, healthcare, and smart cities (Fuller et al. 2020). Among the benefits and opportunities of using DT technology, modeling, and systems design are approaches that have been widely used (Leng et al. 2021), as it is intended for a DT to be a composition of several features of its physical counterpart. However, the setup of a DT can be a complex task, since it requires the integration of several methods and technologies, that can be complex. As a consequence, several DT contributions and frameworks are proposed in the literature, which alleviates some of the configuration work (Leng et al. 2021).

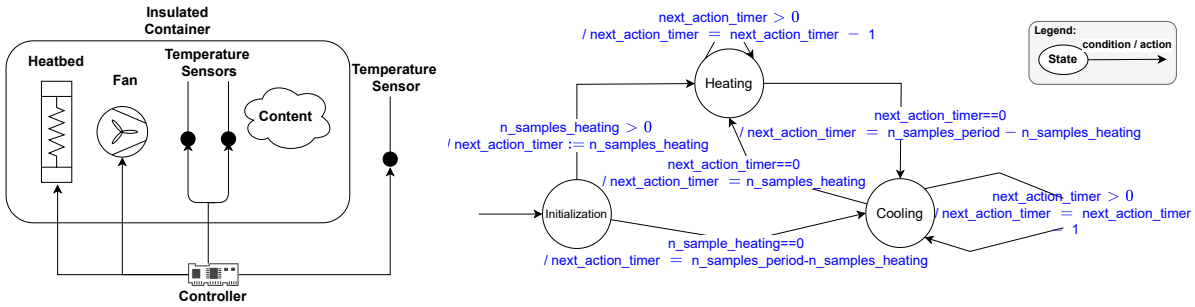
In the case of conceptual frameworks, some authors have analyzed and categorized different components that a DT-driven system should include. For example, He and Bai (2021) discuss and analyze some relevant aspects to be considered in DT-driven manufacturing, including product design, manufacturing, and operation. They also propose a framework for DT-driven sustainable intelligent manufacturing that includes: business architectural layers for intelligent manufacturing units and production lines; integration with manufacturing services; and the integration with the production automation system, manufacturing execution system, and enterprise resource planning. Moyne et al. (2020) also propose a conceptual object-oriented DT framework for DT applications considering a structured set of requirements derived from a general DT definition. Cheng et al. (2020) propose DT-II, a conceptual framework for smart manufacturing that comprises integration at the product lifecycle level, at an intra-enterprise level, and at inter-enterprise level considering a number of supporting technologies, including advanced sensing and communication technologies, ultra-high fidelity modeling, simulation, and verification, DT data construction and management, intelligent production, computing and analysis, and interoperability.

In the case of practical and deployable frameworks, similar conceptual designs exist, but additionally, they offer a set of tools to deploy the DT applications within certain scopes and domains. The Asset Administration Shell (AAS) (Bader et al. 2020, Ye and Hong 2019) is a good example of a DT framework which is based on the details of the AAS specification proposed by Platform Industrie 4.0. This framework provides common semantics to models and deploys DTs in computer networks within an industrial context. AAS has several implementations such as Eclipse BaSyx or AASX Package Explorer. The DT framework CPS-Twinning proposed in (Eckhart and Ekelhart 2018) is another available approach that uses AutomationML documents as inputs. On the other hand, Eclipse Ditto provides another alternative in the domain of Internet of Things.

There are also existing approaches with regards to DT-based state estimation of dynamic systems. Darbali-Zamora et al. (2021) propose a DT for state estimation and optimization of distributed energy resources for voltage regulation in electric grids composed of solar energy generators. Their approach implements a commercial state estimator tool called WinIGS that requires power system topology, locations, and distribution circuits. Then, it is combined with a particle swarm optimization method and integrated with co-simulation (Gomes et al. 2018) components to exchange data bidirectionally. Toothman et al. (2021) and He et al. (2021) propose a DT-based method for state estimation of pumping systems. The former approach proposes a DT framework for mechanical system health state estimation and modeling. The framework uses a hierarchical structure to provide aggregation of the system components and a binary qualitative healthy or faulty classification. The latter proposes a DT framework on state estimation for unsteady flow in a pumping station through frequency domain analysis and generalized predictive control theory. It also includes an adaptive model parameter calibration that uses a model-free adaptive control algorithm and a particle swarm optimization algorithm.

2.2 The Incubator Example Digital Twin

The system used in this work is the thermal incubator system that was originally proposed in Feng et al. (2021) as a case study for DT engineering. An overview of the incubator and its control logic is shown in fig. 1.



(a) Schematic overview of the incubator.

(b) Statechart of the open-loop controller.

Figure 1: Incubator overview.

The incubator system is composed of a styrofoam box, a fan, three temperature sensors, a heating device called a heatbed, and a controller. The controller can set the fan and the heatbed to be turned on or off. The fan is always on when the incubator is turned on, while the heatbed can be switched on or off by the controller.

Control Policy The controller is an open-loop controller and is deployed to control the heatbed through two parameters: the heating period (short for α) and the total period. The total period is a fixed amount of time proportional to the sampling time of the incubator (around 3s) and the heating period is the amount of time out of the total period where the heatbed is turned on. In previous work, we used a closed-loop controller. However, in order to decouple the controller from a system and make the self-adaptation more evident, in this work, we chose to use a simple open-loop controller whose parameters are set by the DT. The sampling time in the incubator is fixed, the total period is described using the number of samples. For example, we chose the total period to be 40 samples, which means the total period is around 120s. The heating period is the number of samples where the heatbed is turned on. By changing the heating period, we have different control policies. The control logic of the controller (see fig. 1b) is as follows. The controller turns the heatbed on first for the heating period, then turns it off during the rest of the time in the total period,

and then repeats during the next period. Changing the heating period also changes the heating rate, which is the energy pumped into the system during a period, to control the temperature.

Based on the incubator PT, we built a digital representation of the incubator and integrated it into our DT framework. We used a RabbitMQ server as a broker for communication between different services and InfluxDB as a database to store, access, and retrieve data, as shown in fig. 2, where the middle three blocks in the DT represent services in relation to a self-adaptation service. Relying on such a framework, we developed services such as monitoring, state estimation, and what-if simulations.

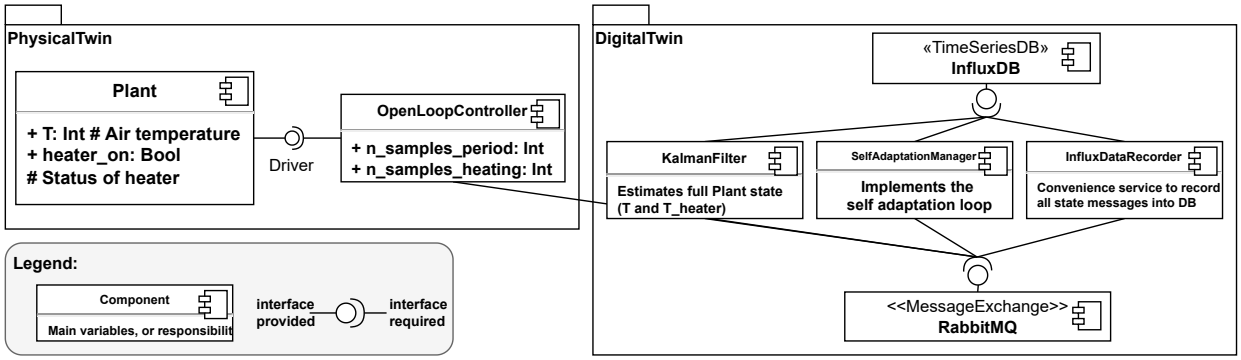


Figure 2: Exemplary communication of incubator DT in relation to a self-adaptation service. All components communicate via the RabbitMQ message exchange, and the data is stored in a time series database.

3 SELF-ADAPTATION LOOP IN THE INCUBATOR DIGITAL TWIN

Integrating a self-adaptation loop into a DT enables adaptation to new environments and thus relieves humans of the responsibility of directly managing the corresponding PT, thereby increasing the system autonomy. In our work, this has been realized through a MAPE-K loop, with the incubator DT used as a case study, and here we present the simulation and experimental results.

3.1 MAPE-K Loop

The MAPE-K loop, proposed in Kephart and Chess (2003) and shown in fig. 3, is a promising framework and well-recognized engineering approach (Arcaini et al. 2015) for self-adaptive systems.

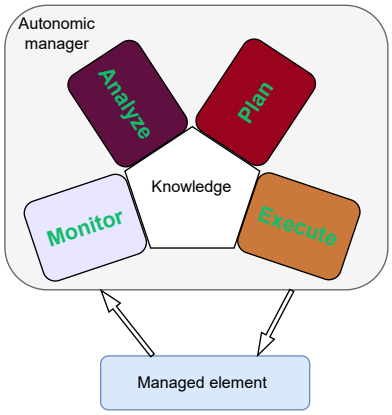


Figure 3: MAPE-K loop in an autonomic element (modified from Kephart and Chess (2003)). Different colors correspond to different stages.

An autonomic system consists of autonomic elements, each of which is an individual system containing resources and delivering services to humans and other autonomic elements, as shown in fig. 3. Every autonomic element includes a managed element that is equivalent to an ordinary nonautomatic system, such as a printer or a database, together with an autonomic manager that controls and represents them. Note that every autonomic element may have multiple managed elements since one autonomic manager can manage multiple elements, for example, multiple printers or multiple databases.

A key aspect in an autonomic manager is the knowledge about the managed element and/or other necessary information. For example, it may refer to the system configuration or mathematical models of the system.

Based on this knowledge, the MAPE-K loop does the following: by monitoring the managed element and its external environments, and constructing and executing plans based on an analysis of this information, the autonomic manager will relieve humans of the responsibility of directly administering the managed element.

There are similarities between an autonomic element and DTs. The managed element is analog to the PT of DTs and the autonomic manager to one of the services of the corresponding DT. Furthermore, the autonomic manager and the managed element have bi-directional communication, as a PT and its DT. Such similarities enable us to realize and deploy the MAPE-K loop into our DTs framework.

3.2 Self-adaptation Loop of the Incubator DT

The overview of dependencies of services in our DT for the self-adaptation loop is summarized in fig. 4.

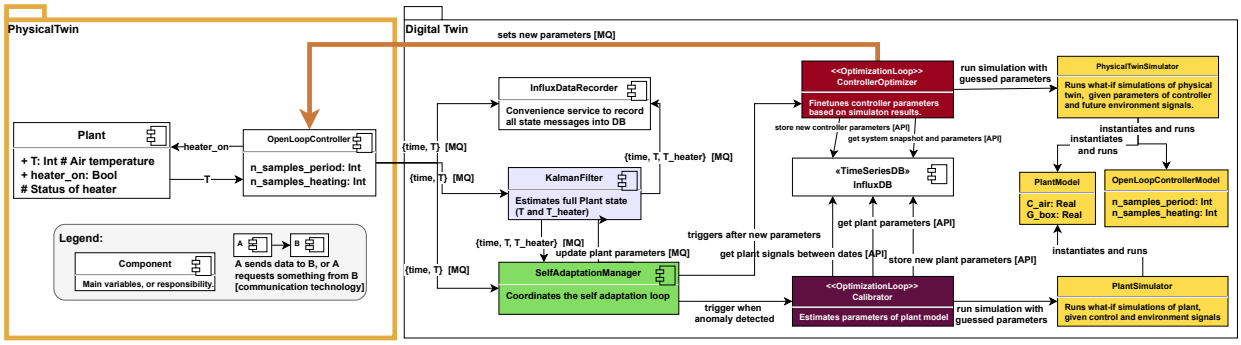


Figure 4: This diagram shows the dependencies of services that help achieve the self-adaptation loop.

The goal of the incubator is to regulate the temperature inside an insulated box. In order to showcase the ability of self-adaptation, we need a situation where potential human intervention would be needed, and therefore we changed the working conditions of the system by opening the lid of the styrofoam box. As the model (plant in fig. 4) and the controller (open-loop controller in fig. 4) of the incubator are designed to work with the lid closed, opening the lid during operation invalidates the model and the controller, and human input would normally be needed to recalibrate the model and re-optimize the controller, the process of which are colored in yellow in fig. 4. Since we only have this anomaly, it is not necessary for the system to analyze the type of the anomaly, it only requires the system to gather data for re-calibration and re-optimization. In this work, we deploy a Kalman filter (see Kalman Filter in fig. 4) to monitor and detect anomalies (Feng et al. 2021). When data is retrieved from the database (InfluxDB in fig. 4) in the DT, it is necessary to re-calibrate and optimize the model and the controller respectively. After obtaining the necessary parameters, the DT re-configures the model and controller with such parameters and delivers the controller parameters to the PT to be configured.

Knowledge Knowledge plays an important role in the MAPE-K loop in all its four stages. In the incubator DT, the knowledge mainly refers to the model of the incubator system as it contains the target information that is the temperature. The incubator is a thermal system, and its physical behavior is governed by the conservation of energy. Due to the nature of convective and conductive heat transfer through the internal airflow and the loss of energy to the ambient surroundings, the temperature of the incubator varies significantly in both time and space. However, as our studies in Feng et al. (2021) showed, the transient temperature response is well represented by a reduced-order model, in which the spatial temperature distribution is lumped into a two degrees-of-freedom model with four free parameters. This was confirmed using both experimental analysis and Computational Fluid Dynamics (CFD) simulations, in which the latter the governing partial

differential equations are discretized and solved in time and space (Feng et al. 2021). The four-parameter model reads as follows:

$$\frac{dT_{heater}}{dt} = \frac{1}{C_{heater}}(VI - G_{heater}(T_{heater} - T_{bair})) \tag{1a}$$

$$\frac{dT_{bair}}{dt} = \frac{1}{C_{air}}[G_{heater}(T_{heater} - T_{bair}) - G_{box}(T_{bair} - T_{room})], \tag{1b}$$

where T_{heater} is the lumped temperature of the heatbed and T_{bair} is the lumped temperature of the internal air, the fan, and the insulating container. In eq. (1), C and G hold the four free parameters that represent the lumped thermal capacitances and the effective heat transfer coefficients, respectively. T_{room} is the ambient room temperature, and the product of VI (voltage and current) is the power added to the heatbed. Finally, we note that the reduced-order approach which the four-parameter model represents is very suitable for use in a DT environment, because the computation time needed to advance the model in time is almost 100 000 times faster than a full-scale CFD model (Feng et al. 2021). This model is implemented in the *PlantModel* in fig. 4.

Monitoring Monitoring usually is the first stage for a self-adaptation loop because it provides the ability to distinguish whether the system works as intended or not. If the system does not work as expected, this might mean that there is an anomaly in the system and this can potentially have disastrous consequences. We deployed a Kalman filter to estimate the states of the incubator, which are the air temperature inside the insulated container and the heatbed temperature. A Kalman filter takes the model output and the measurement data from the sensors as inputs, and when these two signals align, the system works as expected. When the lid is opened, the properties of the incubator change which leads to the alternation of parameters in the model, but the model utilized by the Kalman filter does not know this change, it still uses old parameters. This results in a mismatch of temperature behavior between the model outputs and the measurement as shown in fig. 5, enabling the Kalman filter to detect such an anomaly. Further details can be found in Feng et al. (2021).

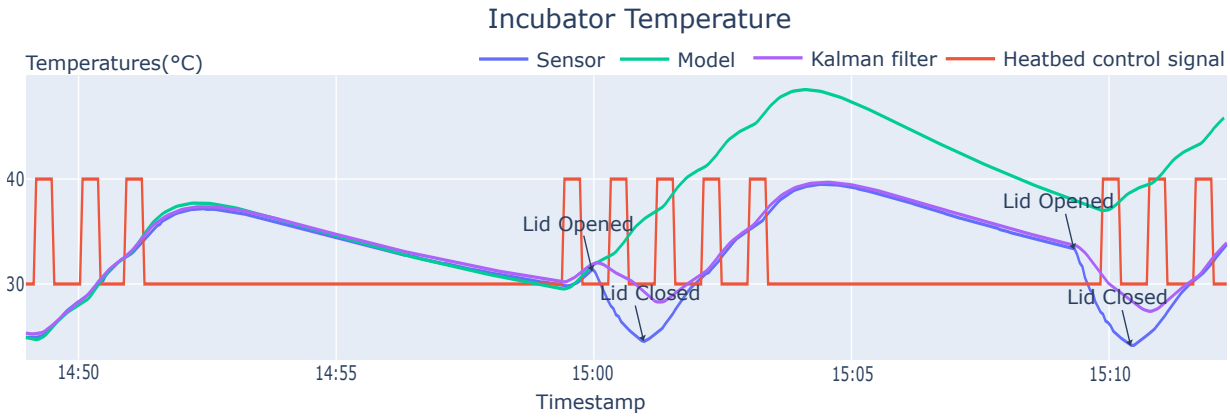


Figure 5: Results of the Kalman filter for anomaly detection (Feng et al. 2021).

Analyze When an anomaly is detected, the analyzing stage should be invoked to analyze the properties of the anomaly and come to a conclusion. In our case, the assumption we made is that opening the lid is the only anomaly happening in the system. Therefore, our analyzer only needs to determine whether the model needs recalibration and the controller needs re-optimization. To simplify the process, another assumption has been made that the model and the controller need to be calibrated and optimized when an anomaly is

detected as the model and the controller are designed for operation with a closed lid. At this stage, the loop also waits to collect more data before re-calibrating.

Plan According to the conclusion made by the analyzing stage, the planning stage determines the details of how to achieve the goal. After opening the lid, the model and the controller are no longer applicable to the incubator. To calibrate the model, historical data after the anomaly appears are needed. The more data, the better the calibration will be. Our model calibration leverages historical data stored in the database to calibrate the parameters of the model, with a non-linear optimization method. Then the model parameters in the DT will be stored in the InfluxDB database and accessed by relevant services. After updating the parameters of the model, a service called the physical twin simulator will run simulations with the updated model and estimated controller parameters, then optimize the controller parameters and store them in the database.

Execute This is the last stage of the MAPE-K loop. It executes what has been planned in the planning stage. In our system, the execution stage is to access the corresponding parameters from the InfluxDB database and then deliver and configure them to the model in the DT and controller both in the DT and PT.

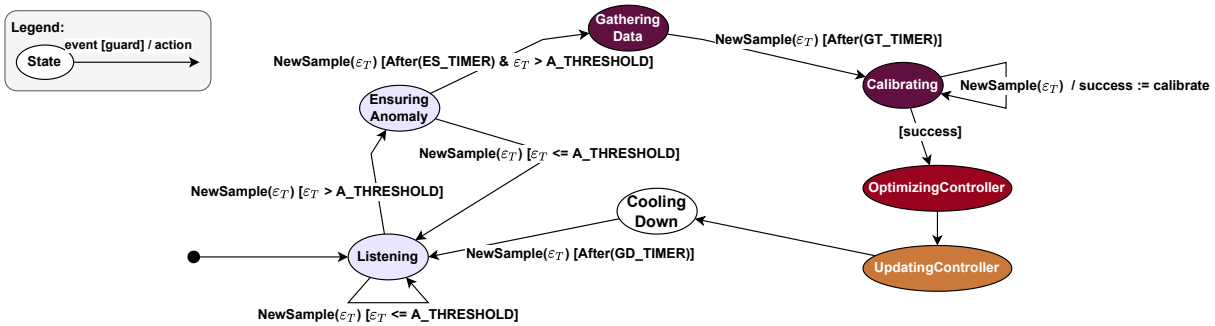


Figure 6: *SelfAdaptationManager* states. This diagram shows the main states the *SelfAdaptationManager* goes through in the self adaptation process. ε_T denotes the difference between the output of the Kalman filter and the sensor measurement.

This involves the services of the self-adaptation manager, a Kalman filter, and a data recorder. The states of the self-adaptation manager can be seen in fig. 6 and the interaction between the main entities participating in the self-adaptation loop are shown in fig. 7.

3.3 Experimental Results

In order to verify the performance of the self-adaptation loop, we conducted two types of experiments. As the incubator DT is a digital representation of our physical incubator and it can mimic the behavior of the PT, we first conducted simulation experiments in the DT framework. Conducting such a simulation experiment within the DT gives us more insights into the system. As the Kalman filter needs the measurement inputs from the temperature sensors, we replace those signals with the simulation outputs of our model, which means model A with parameters P_a is used for generating measurement signals and model B with parameters P_b for the Kalman filter. Notice that model A and model B have the same form described in eq. (1), but with different parameters. To emulate the operation of opening the lid, we change the parameters P_a of model A while the parameter P_b of model B remains the same. The experiment results, in which the desired temperature was set to $41^{\circ}C$, are shown in fig. 8a. From the behavior of the air temperature inside the box (air temperature for short), we can see that the self-adaptation loop works well because the estimated temperature aligns with the sensory temperature both after opening the lid and closing the lid. The controller

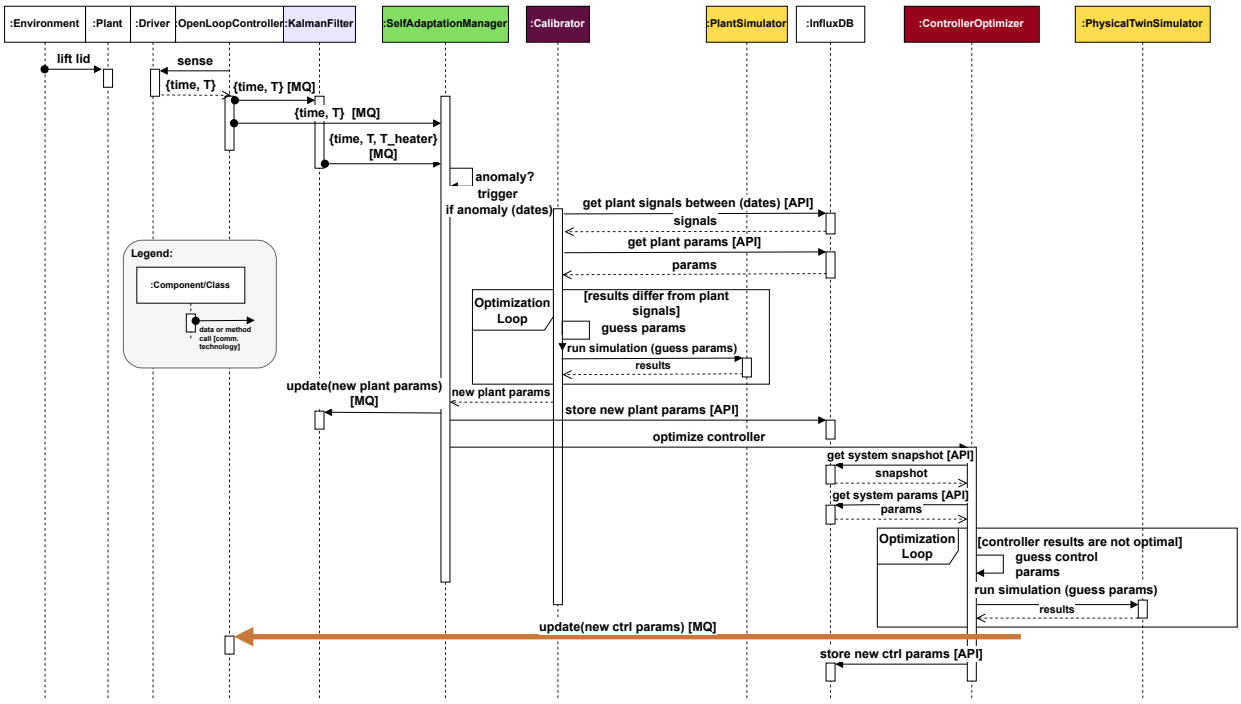


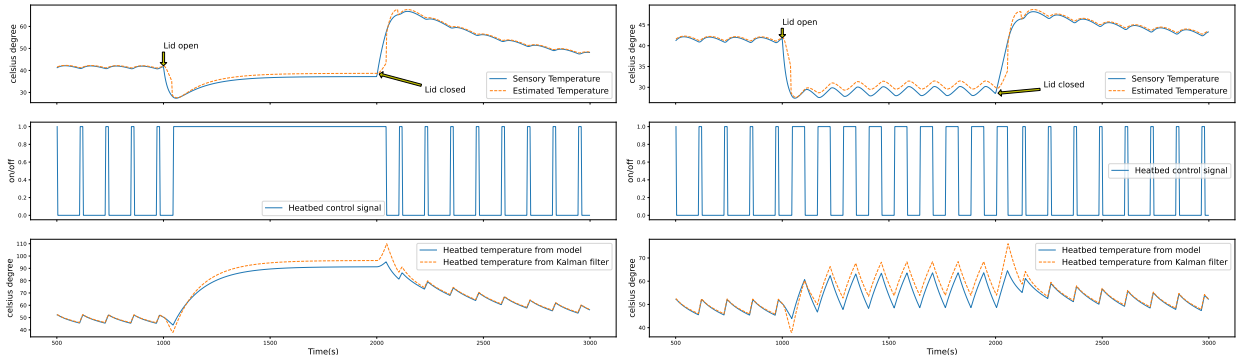
Figure 7: Self-adaptation sequence. This diagram shows the main interactions between the main entities that participate in the self-adaptation process. It is assumed that an anomaly has occurred due to the lid being open.

parameters are optimized, which can be seen from the frequency of the control signal (fig. 8a) both after opening and closing the lid.

We discovered two critical issues from the experiments. First, the optimization process of the controller parameters may fail as the control policy was to turn off the heatbed (set the heating period parameter to be 0) after the lid was closed, and therefore, the air temperature could not reach the desired temperature. There was not enough data gathered for the calibration process, and therefore, the newly found parameters were incorrect. Consequently, the controller optimization process was unsuccessful and the controller did not maintain the correct control policy. To fix this issue, we increased the time of collecting data for calibration to obtain a better model with which to optimize the control policy. Then we ran again the experiment, the result is in fig. 8a.

Secondly, as shown in fig. 8a, the highest air temperature after closing the lid was around 68°C, well exceeding the target temperature of the incubator. This was caused by the optimized controller policy. When the lid was open, the optimized control policy was to turn on the heatbed at its full power, which meant the heating period equals the total experiment period. In this period, the heatbed temperature increased as shown in the heatbed temperature signal in fig. 8a, and after the lid was closed, the energy was transferred to the air inside the box, causing this overshoot.

To overcome this issue, we capped the maximum amount of energy (during one cycle) released into the system by restricting the maximum heating period parameter as the energy can be approximatly calculated by $V * I * \alpha$. The experimental results in fig. 8b demonstrate that the issue is indeed solved as the heatbed control signal was not always turned on during the period of lid opening. We then conducted another experiment with the physical incubator and the incubator DT. In this experiment, we first let the incubator run to the steady state, then we opened the lid. Finally we closed the lid again after around 15 minutes. The



(a) This simulation result is without the constrain of energy released into the system.

(b) This simulation result is with the constrain of energy released into the system.

Figure 8: Simulation results. The first sub-plots are the air temperature behaviors. The second sub-plots are the control signals of the heatbed. The third sub-plots are the heatbed temperature behaviors.

result can be seen in fig. 9 and showing that our self-adaptation loop works as intended. Both the calibration process and the optimization process worked after opening and closing the lid and the heatbed was not always turned on when the lid was opened.

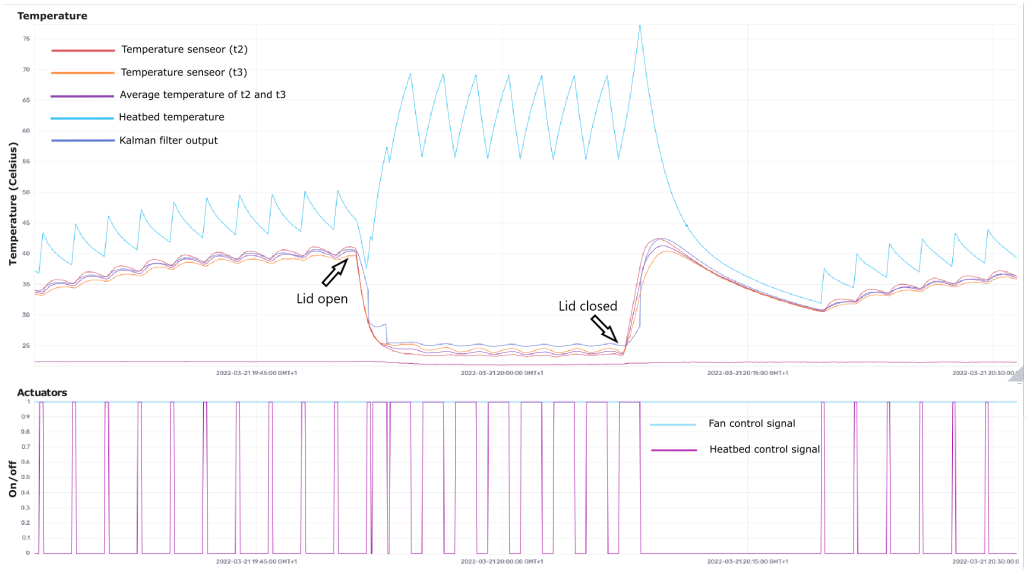


Figure 9: Experimental result of self-adaptation.

4 CONCLUSION

In this work, we demonstrated how the MAPE-K loop can be implemented within a DT framework in order to achieve self-adaptivity. We applied the MAPE-K loop to an incubator DT case study, making it one of the services of the DT. The case study uses an open-loop controller to showcase the benefit (in terms of performance) of using the MAPE-K loop. Note however that, from the point of view of safety, using a closed-loop controller would yield the same unsafe situation as the one reported in fig. 8a. One of the limitations of this paper is that we did not attempt to generalize this approach to other systems. Future work

will attempt to implement a similar self-adaptation loop in a robotic manipulator system, and an agricultural robot.

Our experimental results indicate that the self-adaptation loop works as we expected and it indeed automatically adapts to abnormal situations or changed working conditions. In addition, we investigated other DT frameworks and by studying the self-adaptation loop in our DT, we believe that our self-adaptation loop can be transferred to other DT frameworks if the framework has the service to acquire data and store, access, and retrieve data from a database.

Our experiments revealed two critical issues (described in section 3.3) in applying the MAPE-K loop to a CPS, which highlight the following research challenges.

Detecting Not-Enough-Data Failures An important part of the MAPE-K loop of the incubator DT is the analyzing phase, where data is gathered in order to correctly re-calibrate the models and adjust them to the new situation. At this stage, it is crucial that *checks are executed in order to ensure that the newly found model parameters make physical sense*, before progressing to the next stage. The research question is: for a given model, how much data is enough to successfully recalibrate the model. This question also occurs in machine learning and non-linear system identification, so techniques from those fields can be applied. In our case study, we determined this value from simulations, and did not have to resort to physical experiments. In a more general implementation, there should be some communication between the Analyze and Plan phases of the MAPE-K loop (recall fig. 3), where if the Plan phase did not succeed, then maybe the cause is that there's not enough data, and therefore the system should go back to the analyze phase.

Safety during the MAPE-K Loop The incubator case study highlights an interesting issue: when the lid is closed, there is too much latent energy in the system, caused by the previous self-adaptation loop. As the lid is closed, there is no way to dissipate that heat in a safe manner, leading to overheating. This shows that checks are needed before carrying out a self-adaptation, in order to ensure that *the resulting system configuration is kept safe not just in the immediate future, but also in the face of future anomalies*. This is a difficult challenge as it involves predicting the future. In our case, we circumvented this by degrading the system performance, trading it for safety.

Safety tradeoffs When safety is a concern, there is a tradeoff between gathering more data in the analysis phase (when the system is in a potentially unsafe state), and responding quickly to ensure that the system is reconfigured to the new situation. This means that ideally, for any possible anomaly, the MAPE-K loop should gather just enough data to successfully compute a new system configuration. A similar tradeoff happens in the time spent ensuring that a system configuration is safe.

Tuning MAPE-K Loops Even though the MAPE-K loop makes the system more robust to environmental changes, it still requires careful tuning of various parameters. In the DT case study, we had to fine-tune the following parameters: standard deviation, and initial co-variance matrix (for Kalman filter); threshold for anomaly detection; timer for ensuring that an anomaly is a real anomaly; timer to gather data; timer to wait before starting to monitor for anomalies; convergence tolerance and maximum iterations (for the two optimization loops in the system). To fine-tune these, being able to deploy the DT services in a co-simulation where the PT is also a simulation unit, was crucial in saving time.

ACKNOWLEDGMENTS

We are grateful to the Poul Due Jensen Foundation, which has supported the establishment of a new Centre for Digital Twin Technology at Aarhus University. Hao Feng also acknowledges support from China Scholarship Council.

REFERENCES

Arcaini, P., E. Riccobene, and P. Scandurra. 2015, May. "Modeling and Analyzing MAPE-K Feedback Loops for Self-Adaptation". In *2015 IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pp. 13–23.

Back, T. 1996, January. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press.

Bader, S., E. Barnstedt, H. Bedenbender, M. Billman, B. Boss, and A. Braunmandl. 2020. "Details of the Asset Administration Shell Part 1 - The exchange of information between partners in the value chain of Industrie 4.0". *Plattform Industrie 4.0* vol. 0, pp. 473.

Chen, T., R. Bahsoon, and X. Yao. 2018. "A Survey and Taxonomy of Self-Aware and Self-Adaptive Cloud Autoscaling Systems". vol. 51 (3), pp. 1–40.

Cheng, J., H. Zhang, F. Tao, and C. F. Juang. 2020. "DT-II:Digital twin enhanced Industrial Internet reference framework towards smart manufacturing". *Robotics and Computer-Integrated Manufacturing* vol. 62 (January 2019), pp. 101881.

Darbali-Zamora, R., J. Johnson, A. Summers, C. Birk Jones, C. Hansen, and C. Showalter. 2021. "State estimation-based distributed energy resource optimization for distribution voltage regulation in telemetry-sparse environments using a real-time digital twin". *Energies* vol. 14 (3).

Eckhart, M., and A. Ekelhart. 2018. "A specification-based state replication approach for digital twins". *Proceedings of the ACM Conference on Computer and Communications Security* (October 2018), pp. 36–47.

Feng, H., C. Gomes, M. Sandberg, C. Thule, K. Lausdahl, and P. G. Larsen. 2021, October. "Developing a Physical and Digital Twin: An Example Process Model". In *2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pp. 286–295.

Feng, H., C. Gomes, C. Thule, K. Lausdahl, A. Iosifidis, and P. G. Larsen. 2021, July. "Introduction to Digital Twin Engineering". In *2021 Annual Modeling and Simulation Conference (ANNSIM)*, pp. 1–12.

Feng, H., C. Gomes, C. Thule, K. Lausdahl, M. Sandberg, and P. G. Larsen. 2021, February. "The Incubator Case Study for Digital Twin Engineering". *arXiv:2102.10390 [cs, eess]*.

Fuller, A., Z. Fan, C. Day, and C. Barlow. 2020. "Digital Twin: Enabling Technologies, Challenges and Open Research". *IEEE Access* vol. 8, pp. 108952–108971.

Gomes, C., C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe. 2018. "Co-Simulation: A Survey". vol. 51 (3), pp. 49:1–49:33.

He, B., and K. J. Bai. 2021. "Digital twin-based sustainable intelligent manufacturing: a review". *Advances in Manufacturing* vol. 9 (1), pp. 1–21.

He, L., K. Wen, J. Gong, and C. Wu. 2021. "A multi-model ensemble digital twin solution for real-time unsteady flow state estimation of a pumping station". *ISA Transactions* (In press).

Kephart, J., and D. Chess. 2003, January. "The Vision of Autonomic Computing". *Computer* vol. 36 (1), pp. 41–50.

Leng, J., D. Wang, W. Shen, X. Li, Q. Liu, and X. Chen. 2021. "Digital twins-based smart manufacturing system design in Industry 4.0: A review". *Journal of Manufacturing Systems* vol. 60 (March), pp. 119–137.

Liu, M., S. Fang, H. Dong, and C. Xu. 2021, jan. "Review of digital twin about concepts, technologies, and industrial applications". *Journal of Manufacturing Systems* vol. 58, pp. 346–361.

Moyne, J., Y. Qamsane, E. C. Balta, I. Kovalenko, J. Faris, K. Barton, and D. M. Tilbury. 2020. "A Requirements Driven Digital Twin Framework: Specification and Opportunities". *IEEE Access* vol. 8, pp. 107781–107801.

Rasheed, A., O. San, and T. Kvamsdal. 2020. "Digital Twin: Values, Challenges and Enablers From a Modeling Perspective". *IEEE Access* vol. 8, pp. 21980–22012.

Schuh, G., A. Gützlaff, F. Sauermann, and J. Maibaum. 2020. "Digital Shadows as an Enabler for the Internet of Production". In *Advances in Production Management Systems. The Path to Digital Transformation and Innovation of Production Management Systems*, edited by B. Lalic, V. Majstorovic, U. Marjanovic, G. von Cieminski, and D. Romero, IFIP Advances in Information and Communication Technology, pp. 179–186. Cham, Springer International Publishing.

Tao, F., Q. Qi, L. Wang, and A. Y. C. Nee. 2019, August. "Digital Twins and Cyber–Physical Systems toward Smart Manufacturing and Industry 4.0: Correlation and Comparison". *Engineering* vol. 5 (4), pp. 653–661.

Toothman, M., B. Braun, S. J. Bury, M. Dessauer, K. Henderson, S. Phillips, Y. Ye, D. M. Tilbury, J. Moyne, and K. Barton. 2021. "A Digital Twin Framework for Mechanical System Health State Estimation". *IFAC-PapersOnLine* vol. 54 (20), pp. 1–7.

Weyns, D. 2019. "Software Engineering of Self-adaptive Systems". In *Handbook of Software Engineering*, edited by S. Cha, R. N. Taylor, and K. Kang, pp. 399–443. Springer International Publishing.

Ye, X., and S. H. Hong. 2019. "Toward Industry 4.0 Components: Insights into and Implementation of Asset Administration Shells". *IEEE Industrial Electronics Magazine* vol. 13 (1), pp. 13–25.

AUTHOR BIOGRAPHIES

HAO FENG is a PhD Student at Aarhus University specializing in the fundamental techniques that make up Digital Twins, including monitoring. His PhD combines software and robotics. His email address is haof@ece.au.dk.

CLÁUDIO GOMES is an assistant professor at Aarhus University. He received his PhD at the University of Antwerp, for his work on the foundations of co-simulation. His email address is claudio.gomes@ece.au.dk.

SANTIAGO GIL is a PhD Student at Aarhus University working on Digital Twins in Automation and Manufacturing Systems. His email address is sgil@ece.au.dk.

PETER H. MIKKELSEN is a PhD Student at Aarhus University specializing in the Digital Twin frameworks and their use in relation to production scheduling. His email address is phm@ece.au.dk.

DANIELLA TOLA is a PhD Student at Aarhus University working with robot system integration and digital twins. Her email address is dt@ece.au.dk.

MICHAEL SANDBERG is an assistant professor at Aarhus University. He received his PhD at Technical University of Denmark for his work on manufacturing engineering and multi-physics system simulation. His email address is ms@mpe.au.dk.

PETER GORM LARSEN is a professor at Aarhus University. He has been involved with >40 research projects and he currently leads the AU DIGIT Centre, the AU Centre for Digital Twins, as well as the research group for CPSs. He has published more than 200 articles and three books on modelling of CPSs and DTs. His email address is pgl@ece.au.dk.