Enhancing Product Test Management by Using Digital Twins for Device under Test Analysis

Dr.-Ing. Savarino, Philipp[1] and Dr.-Ing. Dickopf, Thomas[1]

[1] CONTACT Software Ltd., Wiener Str. 1-3 28359 Bremen

Abstract. New paths for product lifecycle management methods and IT tools appeared throughout traditional products evolving to smart products due to recent innovations in information and communication technologies. While some led to new, disruptive solutions, for example, reconfigurable smart products during their use phase based on digital twins, some others support or even enhance existing product lifecycle activities. Focusing on the latter, this contribution introduces how digital twins can be used in early product development phases to support the testing and subsequential evaluation of prototypes. Therefore, a test management approach is described that allows the integration of digital twins of devices under test, following a systematical exploration of their potential benefit on real-time analysis during and after the test run. The results were prototypically implemented by modeling a test environment in a product lifecycle management tool, including managing and analyzing digital twins of product instances.

Keywords: Digital Twin, Product Development, Digital Transformation, Digital/smart factory.

1 Introduction

The customer demand for innovative products and the increasing global production put pressure on companies and result in a fast development pace of highly complex and interdisciplinary smart systems. Based on the desire for ever more innovative systems, high customer satisfaction, and immense product reliability with the ever shorter product lifecycle times, the product testing phase and product test management are also subject to ever-increasing demands. These requirements range from the planning of test procedures and resource management to the test order and the provision and availability of test results throughout the entire lifecycle. In particular, data linking and provisioning for faster traceability and analysis in case of problems and claims poses significant challenges for today's companies. Concepts such as the digital twin can offer optimum support, particularly in providing and managing data and services for the later stages of the lifecycle. However, they must be integrated both methodically and technically into existing processes and the existing IT infrastructures.

This paper will present a concept of how digital twins can support and enhance product test management in terms of data consistency by automatically generating a virtual image for each device under test (DuT), which should also be available for subsequent life cycle phases. Introduced with an overview of state of the art in chapter 2, the approach's essential concept is elaborated in chapter 3. To this end, chapter 3.1 and chapter 3.2 first explain how the concepts of test management and digital twins are mapped to the data objects of the CONTACT Elements platform technology using a conceptual architecture representation. In chapter 3.3, these concepts are then merged to an overall concept, which is then prototypically implemented and validated in chapter 4 using the example of a test management scenario for washing machine tests. The paper ends with a summary of the findings and an outlook on further research activities in the presented context.

2 State of the Art

2.1 Test Management

Test management is a method dedicated to the organization of test assets and artifacts, such as test plans, test cases, and test results to enable accessibility and reuse [1]. When describing Test Management's evolution, three different types can be distinguished, starting with intuitive testing: Here, designers are given time to develop prototypical systems, which are initially tested by themselves. When the system is ready, the project manager invites end-users to test it, and if no critical issues are detected, the system goes into a productive phase. Given an unstructured manner, there is no test documentation, and the detection of critical issues varies according to the different experiences or expectations of the end-users. The next evolutionary step addresses testing based on design documents. Testing starts by collecting documentation, e.g., functional and technical designs. Responsibility for the accuracy of the designs lies with the development team. Tests are designed by functions to be tested based on a logical sequence without any indication of the testing priority or the design's connections

Consequently, the most important aspects of functionalities might not be given the necessary attention during a test cycle. A subsequent evolutionary improvement to intuitive testing and testing with design documents are tests based on requirements, as they are the most important specifications that are defined for the design project. They give advanced insight into the client's priorities and other stakeholders, and the testers answer the question if the product meets the specifications and allow detailed documentation according to the priorities of each requirement [2,3]. The scope of testing based on requirements is to verify each of them by specifically designed tests. Consequently, there may be the necessity for modifying requirements when developing an appropriate test, as they need to be verifiable [4].

Two main models developed out of the Information Technology (IT) industry regarding testing based on design project requirements: Waterfall and Agile. The Agile model provides the framework for incremental changes allowing a design environment to react with more flexibility to changing requirements. In contrast, the Waterfall model considers a more conservative approach, in which requirements are collected from early design phases, and feedback is provided only after the product release [5]. The Agile model itself can be split into two main versions: the Scrum method and the Test-driven development; however, new hybrid development models already offer combinations of these leading models [6].

Independently from the test level, for example, component or unit, integration, system or acceptance level, a typical test management process consists of a test preparation and a test execution phase [7]. While the planning phase considers organizational activities like the definition and preparation of the test steps and specifications, the execution phase consists of activities such as test monitoring and reporting [8,9]. Depending on the test phase, there can be different data objects that have to be managed. Regarding this contribution's scope, the main objects shall be introduced briefly: The test plan includes definitions of different test artifacts such as test objects, test resources, test types and procedures, test objectives, or testing goals. Test steps are the smallest artifact on describing a test process that defines the whole test procedure. The test report summarizes the testing efforts, such as variances, comprehensiveness assessment, results, and evaluation [10,11]. The test order is responsible for scheduling a test plan.

2.2 Digital Twins

A much-discussed and central element for smart products' IT support in terms of digital transformation along their lifecycle is the digital twin concept [12,13]. In general, the digital twin describes the virtual image or virtual twin [14] of a physical asset by reflecting its physical state in a current and historicized form [15]. Its exact characteristics depend on its use cases and the business models of its stakeholders. For this reason, there is no one digital twin and no one correct definition; instead, a large number of definitions have been published and refined over the past few years. An overview of definitions can be found in [16,17,18].

It is crucial that the digital twin forms the link between field data, service information, analysis models, and development data and thus the bridge between product development, production, and product use by linking data along the digital thread [15]. Considered in detail, this provides for comprehensive integration of product models from different development domains, integrated simulations, behavioral models, and product-related data [13]. Digital twins, as previously mentioned, can take different forms based on their intended use, as they consist of different sub-models based on context or focus on different scopes. For example, digital twins could be created for service processes (service twin), manufacturing (factory twin), or the product and its lifecycle (lifecycle twin) [19].

Considering the above references and summarizing the findings, the digital twin of a product has a lifecycle and results from the following essential conditions, which are intelligently and individually combined for specific applications.

The digital twin describes an instance of a master model. This master model, also called digital master, is created and formed during the development of the physical product and represents it in its entirety in all configurations, functions, and behavior. Functions and behavior are represented via digital prototypes, which are based on simulation models. This simulation can be used both for product validation and for the design of business applications on IoT platforms on which the digital twin is managed during product use. A corresponding approach in the form of a so-called twin-in-the-loop concept was presented in [20,21].

Furthermore, each digital twin includes an individual digital shadow [17]. The digital shadow describes the sum of all data generated by a physical asset during its lifecycle. This data includes, among others, operating data and status data, process information, usage data, but also reconfigurations and changes in the context of maintenance and servicing [13].

The digital twin's field of activity is diverse. The data of the digital twin serves as an essential information basis for typical tasks such as the monitoring of individual products, series/fleets, and their maintenance requirements, as well as for the monitoring and analysis of customer behavior, documentation of existing product and software variants and versions, and the simulation, validation, and optimization of product behavior and product-related services [13]. The logical linking of sensor data and product also offers immense potential for interpreting the information in the context of feedback to development and thus for an intended product improvement in the sense of retrofitting and generation planning [22]. In this context, the term "closed-loop engineering" [12,21] or "feedback to design" is often used.

3 Concept

3.1 Conceptional Architecture of Test Management in CONTACT Elements

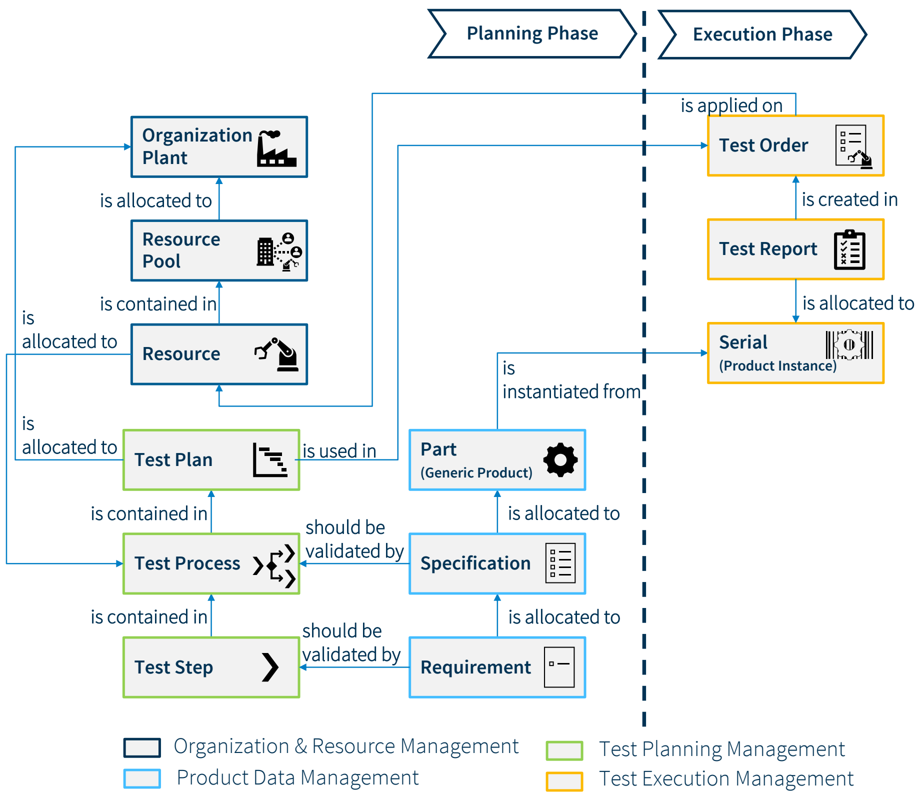
Test management relies on the management and provision of data objects that are part of the product lifecycle, particularly the product development phase. Thus, the product lifecycle management software CIM Database can provide all necessary data objects for a test management environment as part of the CONTACT Elements platform and will be used as a basis for the conceptional architecture.

As introduced in chapter 2, some data objects can be allocated in the test planning or the test execution phase. One organizational object that is part of the planning phase is the test resource pool allocated to an organization or plant. It contains resources needed to execute tests, for example, test beds, time, or the human workforce. Another object in the test management system is the test plan. The test plan is allocated to a plant or organization to which the test plan belongs.

Further objects are the test process and the test steps. The test steps are the hierarchically smallest object. They describe each test step, e.g., with a time slot needed for the specific test step, and can be enriched by a requirement whose satisfaction is validated by the step. The test process is assigned to a test bench of a plant or an organization and can be described in detail regarding time properties, such as waiting time, set-up time, or processing time. The test process aims to validate a specification. Hierarchically, test steps are contained in a test process, and the process is contained in the test plan and the addressed specification.

When the planning phase has ended, and the execution phase begins, the test plan is instantiated, including the assigned test process and steps. A test order is filed, where the test plan is added and the number of tests to be executed is stated. According to the test plan and the number of tests, the necessary resources are withdrawn from the resource pool. Simultaneously, the "part" object to be tested is instantiated by assigning serial numbers to each one, referring to the same amount of tests to be executed, in order to be able to allocate test results to each specific test object with a serial. The test results, as well as a test summary, are part of the test report. Figure 1 shows the test management model with all relevant objects based on the planning and execution phase.

Fig. 1. Test Management Model



3.2 Conceptional Architecture of Digital Twins in CONTACT Elements

The digital twin links static and dynamic product data via the digital thread as a link between the development, production, and operation of a product. Consequently, the digital twin is not just part of the IT infrastructure but is itself an object of development and an integral part of an interdisciplinary perspective on the product. [15] With CONTACT Elements for IoT (CE4IoT), CONTACT Software offers a software solution for managing digital twins, their data, and business logic based on them.

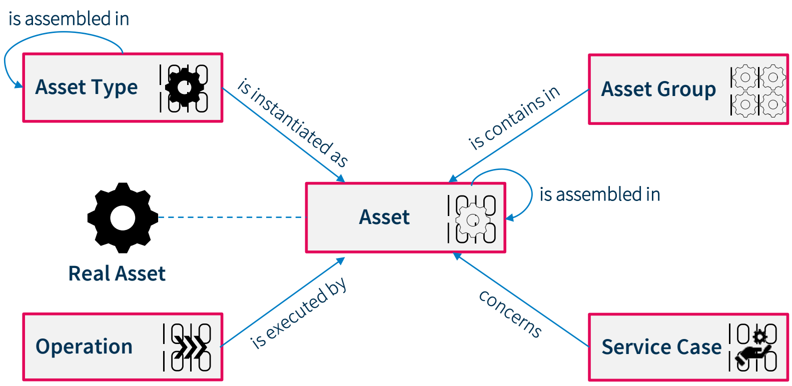
The essential data object in CE4IoT is the asset, representing the digital twin of a real asset. In this sense, it includes its current configuration like features, component structure, settings, or operating status on the one hand and the usage and environmental data (e.g., measurement data, production order and machine parameters, malfunctions and error codes, environmental and user data) that are generated during its use and continuously transferred to CE4IoT and processed there.

Assets are usually instantiated from so-called asset types. The asset type is a blueprint and is used to easily predefine the essential properties, settings, and functions of similar assets in the form of templates. Asset types thus support standardization in the management of assets and ensure data quality. In the sense of a continuous digital thread along the life cycle, the asset type can but does not have to be derived from a virtual product variant in the PLM system.

For monitoring, management, and analysis of similar assets or assets that should be considered in a common context, CE4IoT offers the asset group data object. It allows bundling assets according to any criteria, like their type, location, or other common characteristics. The assignment to asset groups can be done either manually by the user or automatically via defined rules. In the context of an asset group, information of the contained assets can be aggregated, analyzed, and displayed.

Other not insignificant data objects in CE4IoT are the operation and the service case. Operations are individual, time-limited processes of asset usage. Examples for such operations can be individual manufacturing steps of a machine in production, the individual measuring processes of a measuring device, or in the context of test management, the individual tests of a test bench. CE4IoT enables creating and log operations automatically, including all relevant information (e.g., program parameters of an operation or test results) by executing corresponding events and routines. Service cases allow to organize (i.e., plan, control, document) and bill (via an ERP system integration) all services related to an asset (such as spare parts orders, maintenance and repair orders, or pay-per-use billing). A single service case groups the services to be performed or provided for a customer within a certain period concerning one or more assets. Thereby, service cases can be created manually or automatically, cycle-, schedule- or event-controlled from templates. Figure 2 illustrates the relationships and dependencies between the above-mentioned essential elements in the CE4IoT data model.

Fig. 2. IoT Data Model



3.3 Conceptional Architecture of an IoT-Enhanced Product Test Management System

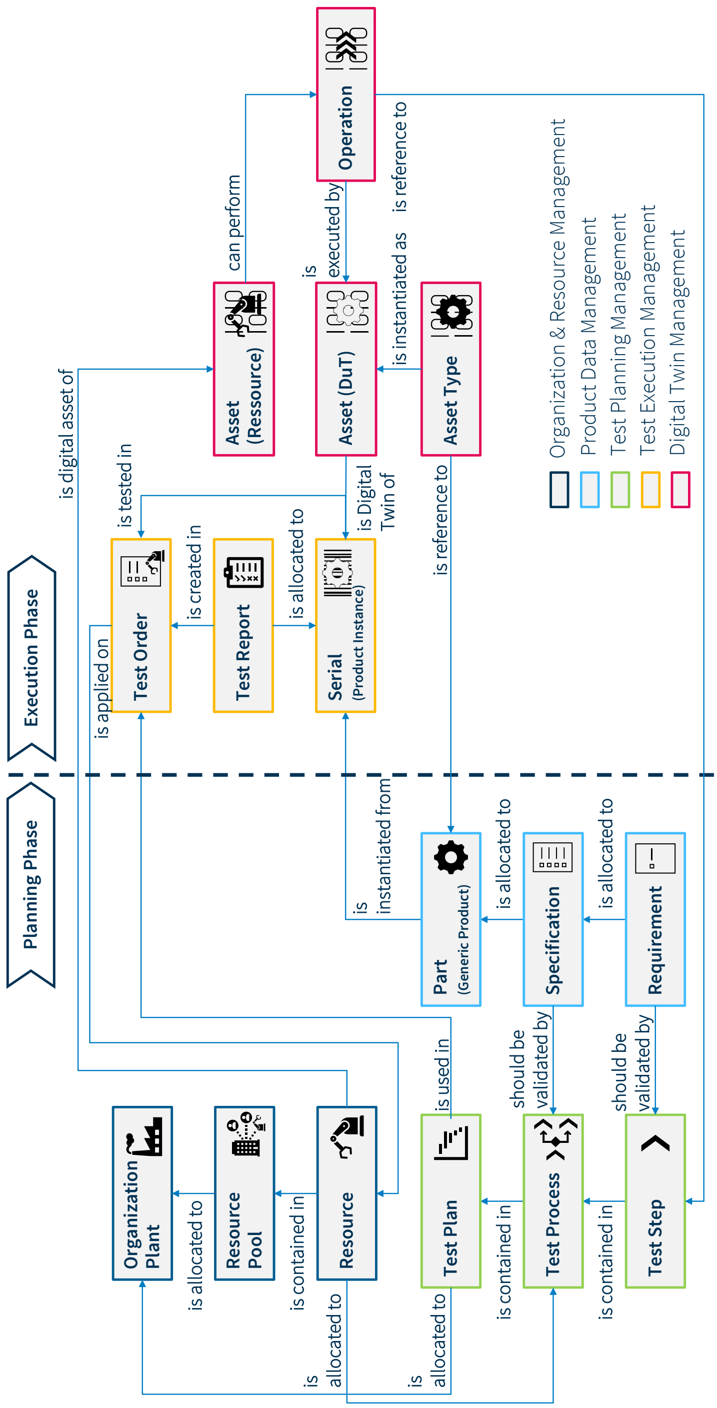
When enhancing product test management with IoT-components for a device under test analysis, the focus lies in the execution phase. There is a necessity to previously create the IoT-objects within the IT infrastructure described in chapter 3.2, which would be part of a planning process. However, as this contribution's scope lies in the analysis of devices under test, the objects in the execution phase and their relation to objects in the planning phase are of particular interest. The main objects related to an IoT-based enhancement of test management are the asset (digital twin), the asset type (digital master), and the operation, as shown in Figure 2.

The asset type is referring to the generic product object part. It can be considered the master template of each asset, which will be generated for each product instance object tested. Therefore, it references the generic product but includes just the information necessary for the DuTs or the device in the field. Furthermore, it defines essential properties, settings, and functions, such as sensors or actors' connectivity, configurations, and working or process patterns, which should be available in each DuT. The asset type has no direct relation to any object part of the test execution management but only to the product data management in the planning phase, as it has a rather organizational purpose regarding the instantiated asset.

The asset is the digital twin of the product instance (serial) that is under test. It consists of the real physical object's prototypical configuration to be tested, including features, component structure, settings, operating status, usage, and environmental data. Regarding a device under test, this can refer to the given bill of materials with the current features tested according to a set of requirements. Usage and environmental data in this context are data that sensors can trace during the actual test step to validate the requirements, e.g., a temperature or noise development. The test steps are contained in the process and, consequently, in the test order's test plan. The asset itself is tested according to the test order, which includes detailed planning of the test, e.g., by defining the targeted end and the test processes' sequence.

Another object that is part of the IoT-components in the execution phase is the asset of the test order's resource. It can be considered as the digital asset of the test bench and can provide additional knowledge about the current status of a test step, for instance, about consumables that are provided by the test bench and are subtracted from the generic resource pool or general status of components that are part of the test bench in the planning phase. Out of the asset resource, operations of the DuTs asset can be performed. As time-limited processes of the asset usage, they can be considered digital references to test steps. They support test steps regarding each step of the test plan's measuring process and create and log operations automatically, such as test results for each test step. Thereby more detailed descriptions and evaluations of each test are possible, as the test analysis is broken down into segments according to the operations mirroring each test step. Figure 3 shows the concept for IoT-enhanced test management.

Fig. 3. Concept for an IoT-enhanced Test Management



4 Implementation and Evaluation

To validate the conceptional architecture introduced in chapter 3, a prototypical system for a test management system, based on the CONTACT Elements technology, has been implemented. It consists of test management and necessary IoT-functionalities to allow real-time analysis of the devices under tests. In the given demo scenario, a washing machine was chosen as the device under test as it can be tested with different test steps that each include different possibilities for sensor analysis.

The first steps are to create the relevant organization and resource management objects, such as the organization that produces the washing machine and allocating a resource pool. As part of the resource pool, different test beds that perform test processes can be created, for example, a test bed where different test steps during a wash cycle are tested. After that, the washing machine's data object as part of the product data management needs to be created. In CONTACT Elements, these objects are called "parts" and each part is given a unique identification number. Additionally, specifications and requirements that belong to the washing machine can be created as well. Core elements of the test planning management are the test plan itself, the test process, and the steps.

Fig. 4. Possible Representation of a Test Plan and its Components in CONTACT Elements

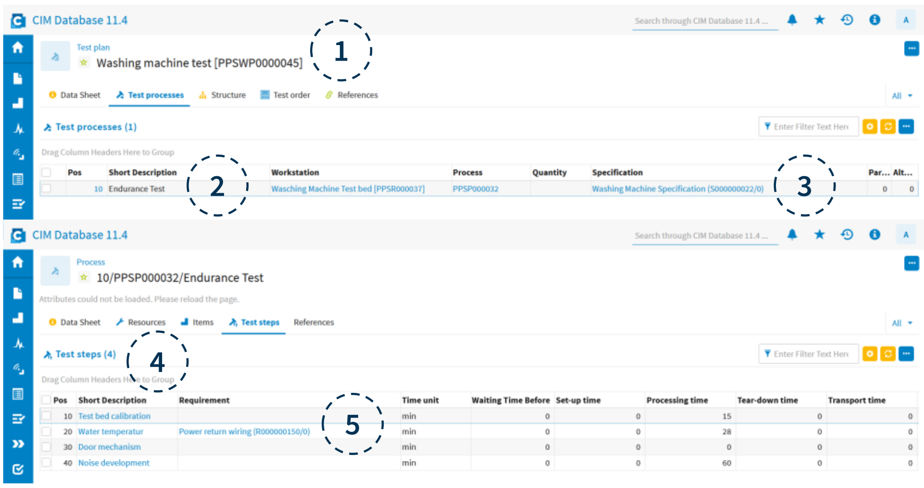
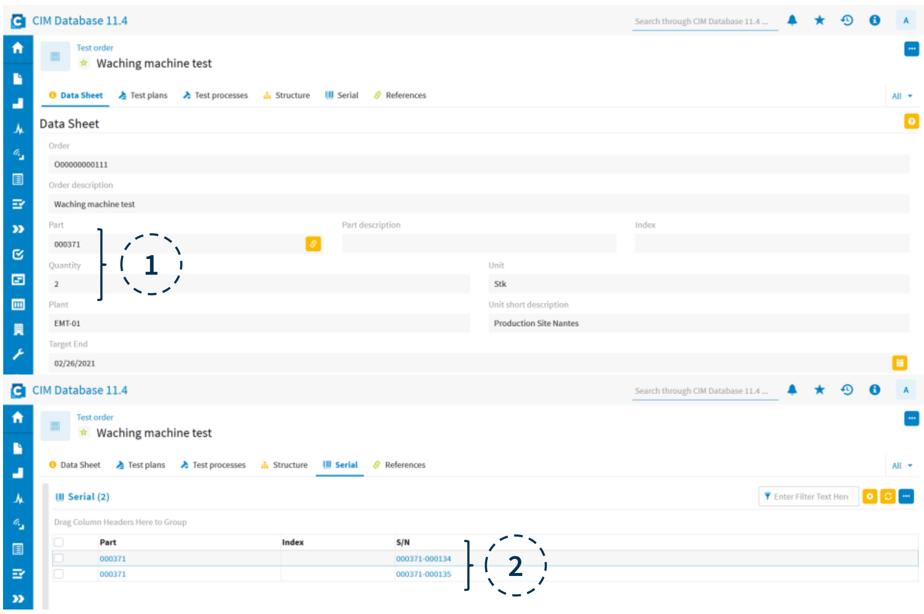


Figure 4. shows the user interface of the test plan of the washing machine test (1) with a specific test number in brackets. In this example, there is one test process called endurance test (2) to be scheduled, although more processes are possible. As shown in (3), this endurance test is satisfying a particular specification. When looking into the detailed description of the endurance test, the assigned test steps are shown (4). In this example, the test process "endurance test" consists of 4 test steps that describe different tests: a generic test bed calibration, a water temperature test, a test of the door mechanism, and a test for the noise development. Each test step can be de-

scribed with specific parameters regarding scheduling or process time. Apart from that, each test can satisfy a requirement, as shown in (5).

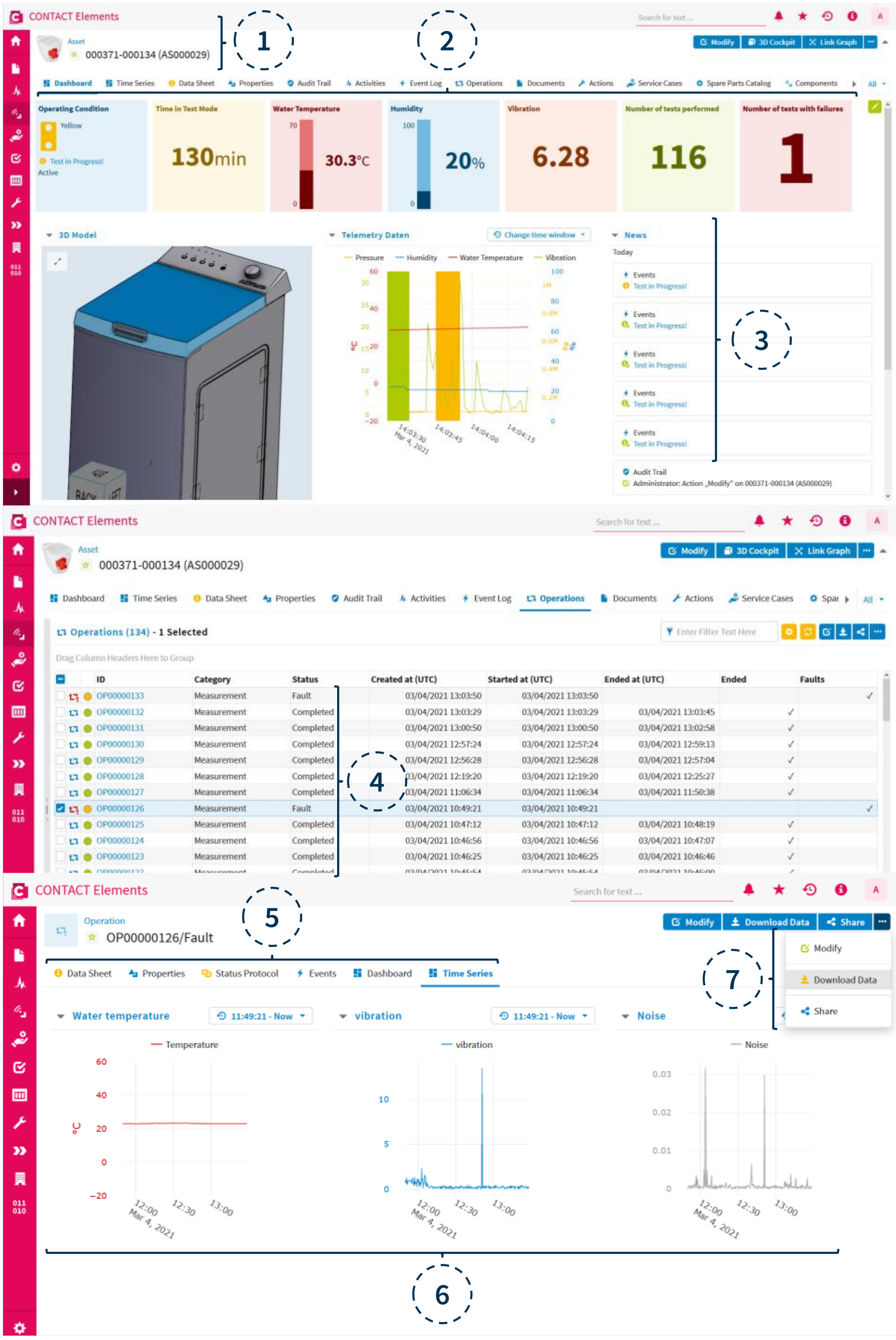
After having completed the test plan and its sub-components, the execution phase begins. The test plan is assigned to a test order, where the specific test or tests can be scheduled and described in further detail, as shown in Figure 5. A unique identification number is assigned to the newly created test order, and a quantity of tests of the part that will be executed is specified (1). Apart from that, a target end for the test is determined. Depending on the quantity of tests, a corresponding amount of serials is generated and assigned to the part-ID (2). In this example, two tests are scheduled for the part so that there are two serials generated for each test.

Fig. 5. Possible Representation of a Test Order and its Components in CONTACT Elements



When the test order is scheduled, an asset is created for each associated serial by instantiating it from the asset type that references the part on which the test order is based. By instantiating the asset type, preconfigured settings (e.g., field device connection, type of data processing and storage, component list, and the dashboard design) are adopted and do not have to be defined anew for each asset. Furthermore, different configurations can be made for different fields of application or phases in the asset's life cycle. For example, separate data providers or dashboards can be created for the individual phases (test phase, utilization phase) and stakeholders (test engineer, customer, service engineer), which are then only visible and applicable for them through roles and rights-based tenant concept. Concerning the test engineer, current information such as the asset's condition about the test execution, specific test values, or even KPIs based on statistical evaluations can then be visualized on the dashboard.

Fig. 6. Possible Representation of a Device under Test in CONTACT Elements for IoT



The upper part of Figure 6. shows the dashboard of a washing machine asset, which references one of the serials (1) of the test order presented in Figure 5. The dashboard header (2) represents the current operating condition of the DuT by a traffic light symbol followed by essential measurements. This case shows the total time of tests, current values for the measured water temperature, humidity, vibration, and the total amount of tests and failures. The dashboard body (3) illustrates supplement information according to the asset. Here it shows in its first row a preview of its 3D model, a specific time series, and a DuT related news feed.

The integrated trigger event logic can automatically create a corresponding operation, for example, when a test step is executed. Each operation has its status network. The status can be changed similarly to creating the operation by the trigger event logic already mentioned, which allows operations to be completed by completing a test step and transferring operations to other states based on the asset's data. For example, if relevant test values (water temperature, the vibration of the washing machine, measured noise) exceed or fall below a certain threshold, it would be possible for a predefined trigger to set the process to a corresponding fault mode to terminate it if necessary.

The operations tab provides an overview of all test steps executed on the washing machine, including the most important information (4). Besides, as shown in the lower part of Figure 6, an operation can also be analyzed in detail by different information categories (5). These include both static and dynamic information. Specifically, this includes master data (the type of test, status, context), test properties (e.g., test parameters and program settings of the test resource), status log, or an overview of all executed or occurred events during the test. Furthermore, the DuT's processed telemetry data are graphically processed and visualized over their course in time series for one or more values (6). In case of failures during the test, anomalies in the graph can be used as initial indicators of the failure's cause. During the device's use by the end-user, the data stored during the test phase can be used as a comparison value if error messages occurring or anomalies can be detected at an early stage with these. The data recording of a test ends with its termination or by the termination of a test process/step in the course of a malfunction. Consequently, an operation can also be considered a digital test report for the corresponding test step, which can also be downloaded or shared on request (7).

5 Conclusion and Outlook

Enhancing product test management by using digital twins enables fundamental new potentials for the device under test analysis, e.g., during the test as part of the real-time analysis and the instance-specific device management after tests have been executed. This contribution addressed this potential by describing fundamental approaches in both test and digital twin management at first. Subsequently, a conceptional architecture of a test management system and digital twins based on the CONTACT Elements technology have been introduced. Combining these two architectures shows how the test execution phase enriched by digital twins can qualify the device under test analysis in real-time and what dependencies between test management and digital twins need to be considered. The approach was prototypically implemented and validated using different test use cases for a washing machine test scenario. Future work might focus on activities after the execution of the tests. For example, if the digital twin management can directly evaluate requirements tested in specific test steps or based on test results, new test plans or orders with new test parameters are filed and scheduled. Another approach might be examining an asset group to allow an accelerated evaluation of a test row.

The concept presented in this paper describes a partial result of the reference kit developed in the AKKORD research project for the application of networked and integrated industrial data analysis in dynamic value networks. The core of this reference toolkit is a data backend system [23] that collects data from different source systems and or lifecycle phases and links them in a single meta-model to perform data analyses based on them. The results illustrated one aspect of this data backend system and were adapted on and prototypically illustrated by the use case of a manufacturer of white goods.

Acknowledgments

This research and development project is funded by the German Federal Ministry of Education and Research (BMBF) within the "Innovations for Tomorrow's Production, Services, and Work" Program and implemented by the Project Management Agency Karlsruhe (PTKA). The author is responsible for the content of this publication.

SPONSORED BY THE

Federal Ministry of Education and Research

akkord

BETREUT VOM

PTKA
Projektträger Karlsruhe
Karlsruher Institut für Technologie

For more details about the project, visit www.akkord-projekt.de/en

References

1. Parveen, T. et al.: A Case Study in Test Management. ACM-SE 45: Proceedings of the 45th Annual Southeast Regional Conference, pp. 82-87, Winston-Salem (NC), USA (2007).
2. Pinkster, I., et al..: Successful Test Management: An Integral Approach. Springer, Science & Business Media, Berlin (2013).
3. Hood, C., et al.: Requirements Management: The Interface Between Requirements Development and All Other Systems Engineering Processes. Springer, Science & Business Media, Berlin (2007).

4. Kim, J. et al.: Management of Software Test Using CASE Tool. 2019 International Symposium on Systems Engineering (ISSE), Edinburgh, United Kingdom (2019).
5. Stober, T., & Hansmann, U.: Agile Software Development: Best Practices for Large Software Development Projects. Springer, Berlin (2010).
6. Dima, A. M., & Maassen, M. A.: From Waterfall to Agile software: Development models in the IT sector, 2006 to 2018. Impacts on company management. Journal of International Studies, 11(2), pp. 315-326 (2018).
7. Homès, B.: Fundamentals of software testing. ISTE ltd and John Wiley & Sons, London, United Kingdom (2012).
8. Eldh, S. et al.: Towards Fully Automated Test Management for Large Complex Systems. 2010 Third International Conference on Software Testing, Verification and Validation, pp. 412-420. Paris (2010).
9. Kukreja, S. et al.: A critical survey on test management in IT projects. International Conference on Computing, Communication & Automation, pp. 791-796. Noida, India (2015).
10. Juhnke, K.et al.: Challenges concerning Test Case Specifications in Automotive Software Testing: Assessment of Frequency and Criticality. Software Quality Journal (2020).
11. Hooda, I. & Chhillar, R. S.: Software Test Process, Testing Types and Techniques. Software Test Process, Testing Types and Techniques, Volume 111 – No 13 (2015).
12. Dickopf, T. et al.: Closed-Loop Systems Engineering – Supporting Smart System Design Adaption by Integrating MBSE and IoT. Proceedings of the 4th International Conference Complex Systems Design & Management Asia, Beijing, China. (2021, to be published)
13. Lünnemann, P. et al.: Smart Industrial Products – Smarte Produkte und ihr Einfluss auf Geschäftsmodelle, Zusammenarbeit, Portfolios und Infrastrukturen (2019).
14. Abramovici, M. et al.: Reconfiguration of smart products during their use phase based on virtual product twins. CIRP Annals 66 (1), pp 165–168 (2017)
15. Göckel, N. & Müller, P.: Entwicklung und Betrieb Digitaler Zwillinge. ZWF - Zeitschrift für wirtschaftlichen Fabrikbetrieb – Digitaler Zwilling. Hanser, Band 115, pp 7-10 (2020)
16. Negri, E. et al.: A Review of the Roles of Digital Twin in CPS-based Production Systems. Procedia Manufacturing 11, pp 939–948 (2017)
17. Stark, R. & Damerau, T.: Digital Twin. The International Academy for Production Engineering, CIRP Encyclopedia of Production Engineering. Springer, Berlin, Heidelberg (2019)
18. Eigner, M. et al.: Definition des Digital Twin im Produktlebenszyklus. ZWF – Zeitschrift für wirtschaftlichen Fabrikbetrieb, Hanser, Band 11, Ausgabe 6, pp 345-350 (2019)
19. acatech: Cyber-Physical Systems. Innovationsmotor für Mobilität, Gesundheit, Energie und Produktion. acatech - Deutsche Akademie der Technikwissenschaften. Berlin, Heidelberg (2011)
20. Dickopf, T.: A holistic Methodology for the Development of Cybertronic Systems in the Context of the Internet of Things. Shaker (2020)
21. Dickopf, T. et al.: A Holistic System Lifecycle Engineering Approach – Closing the Loop between System Architecture and Digital Twins. Procedia CIRP - 29th CIRP Design Conference 2019, Póvoa de Varzim, Portugal, Elsevier Procedia, pp 538-44 (2019)
22. Massmann, M. et al.: Significance and Challenges of Data-driven Product Generation and Retrofit Planning. Procedia CIRP - 29th CIRP Design Conference 2019, Póvoa de Varzim, Portugal, Elsevier Procedia, pp 992-97 (2019)
23. Eiden, A. et al.: Anforderungen an ein Daten-Backend-System zur Unterstützung industrieller Datenanalyse-Anwendungen in digitalen Engineering-Prozessen dynamischer Wertschöpfungsnetzwerke. Proceedings of the 31st Symposium Design for X – DFX2020, pp 81-90 (2020)