2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C)

An API Development Model for Digital Twins

Jim Scheibmeir
jimscheibmeir@gmail.com

Yashwant K. Malaiya
Colorado State University
malaiya@cs.colostate.edu

Abstract— **Digital twins are software implementations of their physical counterparts. These software representations act through application program interfaces (APIs) to the physical devices they monitor, engage with and possibly control. Here a development model is proposed to ensure the desired reliability and performance of the API which is critical to the overall success of the Digital Twin.**

Keywords—Application Programming Interface, API, Digital twin, test automation, software quality

I. Introduction

Digital twins are a logical conduit between physical entities and humans, geared towards using monitoring and control as a way to improve life [8]. The digital twin is becoming more important, and not just in understanding assets within our environment, but also in controlling them [2]. Within the engineering process for digital twin APIs, design and run-time are critical [2]. These APIs as components are more important than ever in modern software development [1]. Here an approach is proposed that quickly produces a development model to meet growing expectations.

II. Software Development Model of Digital Twins

An optimum digital twin is virtual and can generate any behavior and control any state that would be accomplished by engaging its physical twin [4]. Such abstraction allows maintenance and reuse of both the physical devices and the software twin over time. The software acts as a proxy, allowing monitoring or controlling of the physical device.

A. Digital Twins Needs Analysis

Engineering a digital twin should be done in a modeled approach by performing a needs analysis from the physical device and it's operating environment. From these needs, we can visualize the software proxy for the remote operations. Defining the systems functions and components begins with an objectives tree which then progresses into a contextual diagram. The branches of the objectives tree align to monitor and control, and later translate into GET and POST type API transactions.

The physical device exists within an operational environment which is a critical design factor. The contextual diagram is key to understanding the relationship of the physical device to the environment conditions and constraints, and for building the digital twin to facilitate these relationships and limitations. These concerns will be implemented as objects, functions, and components utilizing design tools and specifications such as OpenAPI, visual modeling and design tools such as Node-RED, or implementation and runtime tools such as Azure Digital Twins. OpenAPI designs list the various objects, paths, and operations that the API will present [6], creating documentation and hence specifying the contract between design and implementation.

B. Digital Twin Code Design

The proposed development model focuses on quality code design and testing early, or 'shift-left', through Test Driven Development (TDD). TDD also helps detect defects early and reduce the cost of software development [3]. Software defects that leak into the production digital twin can be disruptive to the physical counterpart, and expensive to remediate.

By writing our test cases first, we create a design contract between what the successful implementation necessitates and the code that is later written. By creating and deploying the tests first, we create code that accomplishes two things. First, it focuses on quality by sufficing the test. Second, code that is easily unit tested is more reliably integrated with other code, improving the API usability. Going forward, during refactoring or other product enhancements, compliance with the design contract built through TDD will be achieved.

C. Performance Engineering

In a digital twin ecosystem, performance could become a limiting factor. Since the digital twin will monitor and control the physical twin, if performance is degraded, the physical device could suffer from untimely decisions in extreme environmental conditions. This could result in critical losses including loss of life, in manufacturing or medical systems. For this reason, performance testing is a requirement, and within our development model, we test performance iteratively, early and often.

While HTTP is common in APIs, IoT introduces new protocols into the API and Digital Twin ecosystem. A performance engineering approach is required that can be applied to the various implementations.

D. API Test Framework

Beyond the previously established unit and performance testing, there exist vast areas where an API can experience failures and degradations. To limit these and control experience, we implement test contracts in the form of automated functional and nonfunctional tests. We interrogate how the API should behave, with focus on service level agreements, security and entitlement. Our functional tests assure that the operation of the API satisfies functional specifications and that our user has the correct permissions. We can accelerate defect discovery timeline and reduce costs over time by introducing automation [5]. Implementing test automation into the development model reduces some of the tedious and error prone tasks, such as manual testing, defect analysis and reporting.

The International Organization of Standards 25010 quality model is composed of eight attributes. One of these eight is the product's usability. This is frequently considered as a part

978-1-7281-3925-8/19/$31.00 ©2019 IEEE
DOI 10.1109/QRS-C.2019.00103

of user interface testing, and thus in the case of the digital twin, we must assure the usability of the API. Table 1 outlines focus areas for API usability.

TABLE I. API USABILITY CHARACTERISTICS

Concepts and Artifacts	Explanation of applicability and importance
API Documentation	When starting a digital twin initiative, there may be many APIs from different providers. Proper documentation enhances usability and promotes initial use.
API Example Code	API Authentication examples are common.
Consistent Operation Naming	Method naming standards accelerate learning how to use the API by providing a repeatable verb and noun association
Consistent Return Types	Developers will expect an API method to return a consistent data type, even if a failure occurs.
Limited Arguments	Going above six arguments being passed into an API method will begin to deteriorate the usability of the API

E. API Mediation

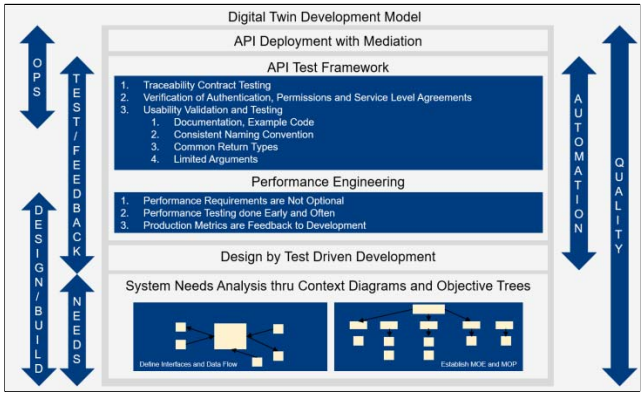
Thirty eight percent of respondents in a Gartner survey reported that APIs are "very important" to their digital platform [7]. If the API service is important, it should be secured, managed and measured, and mediation is the solution for all three. API mediation is not new, but it is critical to the digital twin development model and is the deployment method of choice. Inner APIs attach directly to the digital twin. Outer APIs are exposed through mediation. Outer APIs can be composite endpoints created by joining two or more inner APIs. Using this mediation management layer allows us to better secure and maintain the integration experience through an additional layer of management. This mediation enables metering and measuring usage and performance.

F. Concepts within the Digital Twin Development Model

Our model begins with creating objective trees and context diagrams. These supply the starting points to creating architecture and specifications around the interfaces and performance characteristics. Our model then moves into Test Driven Development and OpenAPI for the purpose of designing and coding to a contract (Fig 1). Performance engineering is our next major initiative. Early feedback is critical as performance issues can stem from poor design, and design changes are more costly to correct than feature changes. After automated functional testing, we cover deployment and operations of the API through mediation. This allows us to maintain the API and the digital twin, even to the point of swapping them out.

This development model takes into consideration requirements, design, build, test and operations, with a focus on automation and quality.

Fig. 1. The Digital Twin API Development Model



G. Future Work

This paper proposes a development model of a digital twin, from requirements and design, with a shift-left approach to testing early and often, up to the API mediation layer. The UI of the digital twin would then utilize the mediation layer. Further work is required to develop an effective user interface for the digital twin.

REFERENCES

[1] Burns, C., J. Ferreira, T. D. Hellmann, and F. Maurer. "Usable Results from the Field of API Usability: A Systematic Mapping and Further Analysis." *2012 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, November 12, 2012. doi:10.1109/vlhcc.2012.6344511.

[2] Cearley, David, Alfonso Velosa, Marc Kerremans, Mike Walker, and Brian Burke. "Top 10 Strategic Technology Trends for 2018: Digital Twins." Gartner IT Glossary. March 08, 2018. Accessed October 07, 2018. https://www.gartner.com/doc/3867164.

[3] Damm, Lars-Ola, Lars Lundberg, and David Olsson. "Introducing Test Automation and Test-Driven Development: An Experience Report." Electronic Notes in Theoretical Computer Science116 (2005): 3-15. doi:10.1016/j.entcs.2004.02.090.

[4] Grieves, Michael, and John Vickers. "Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems." *Transdisciplinary Perspectives on Complex Systems*, 2016, 85-113. doi:10.1007/978-3-319-38756-7_4.

[5] Kumar, Divya, and K.K. Mishra. "The Impacts of Test Automation on Softwares Cost, Quality and Time to Market." Procedia Computer Science, 79 (2016): 8-15. doi:10.1016/j.procs.2016.03.003.

[6] Martino, B. Di, M. Rak, M. Ficco, A. Esposito, S.a. Maisto, and S. Nacchia. "Internet of Things Reference Architectures, Security and Interoperability: A Survey." *Internet of Things*1-2 (2018): 99-112. doi:10.1016/j.iot.2018.08.008.

[7] O'Neill, Mark, Jason Wong, Elizabeth Golluscio, and Marty Resnick. "API Mediation Is the Key to Your Multiexperience Strategy." Hype Cycle Research Methodology | Gartner Inc. July 23, 2018.. https://www.gartner.com/doc/3883663.

[8] Saddik, Abdulmotaleb El. "Digital Twins: The Convergence of Multimedia Technologies." IEEE MultiMedia, vol. 25, pp. 87-92, Apr. 2018.