Brownfield Innovations: Digital Twin Development Guide

Contents

By
Max Ivannikov, Head of IoT, Industry 4.0 Expert, DataArt
Nikolay Khabarov, IoT Architect, Senior IoT Engineer, DataArt

DataArt

A **digital twin** is a virtual representation of a real-world physical system or product (a *physical twin*) that serves as the indistinguishable digital counterpart of it for practical purposes, such as system simulation, integration, testing, monitoring, and maintenance.

Wikipedia

Brownfield development is a term commonly used in the information technology industry to describe problem spaces needing the development and deployment of new software systems in the immediate presence of existing (legacy) software applications/systems. This implies that any new software architecture must take into account and coexist with live software already in situ.
Wikipedia

Introduction

The Digital Twin concept's benefits are quite obvious. It enables smarter decision-making, provides new means of process optimization, and makes manufacturing more manageable.

This whitepaper aims to provide useful, practical insights on developing a Digital Twin solution along with best practices, dos and don'ts, and possible timelines.

Brownfield modernization requires deep attention to the details of the existing infrastructure. It's considered to be more complicated than the greenfield approach in most cases. Nevertheless, brownfield innovations form the majority of cases due to business continuity and economic reasons.

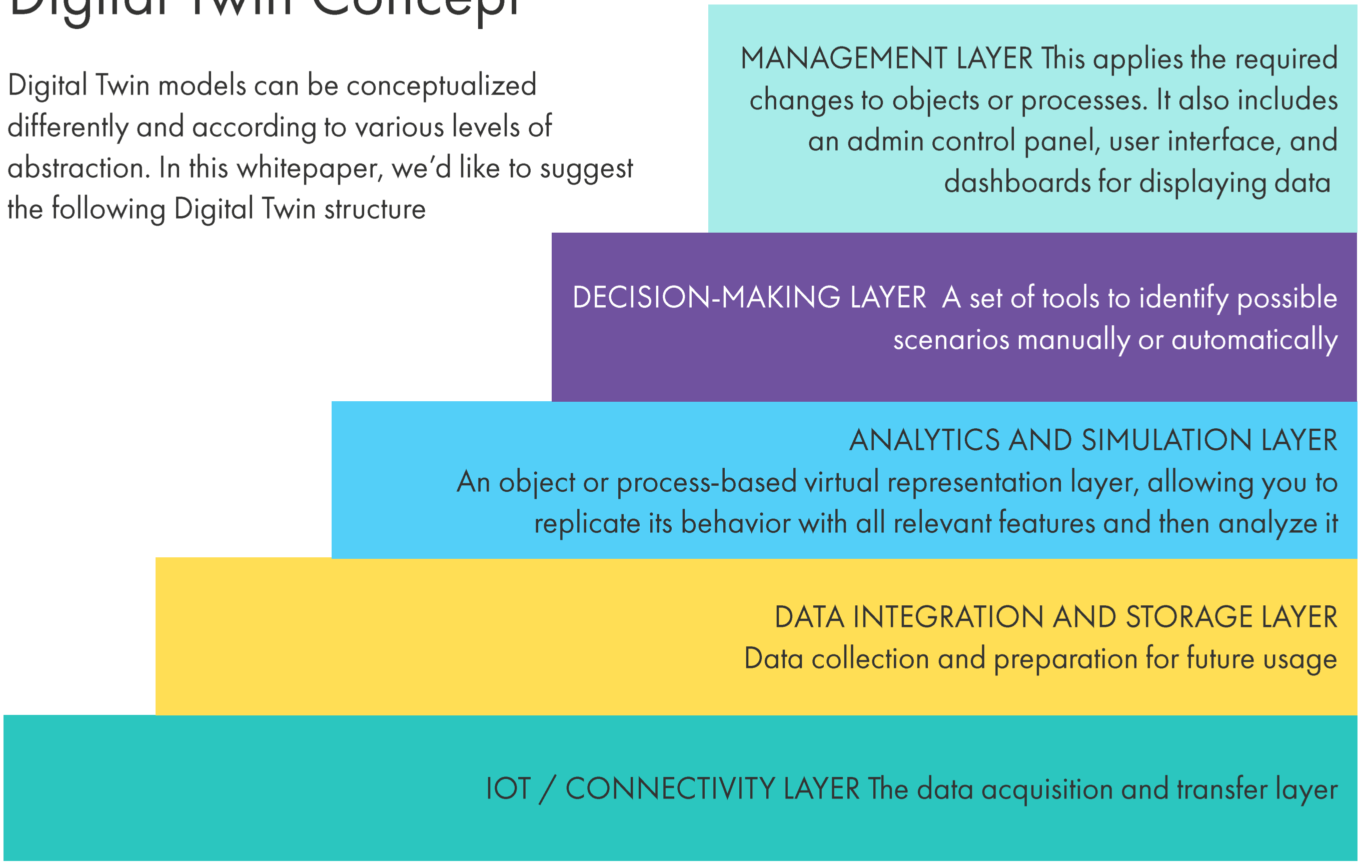
Generally, this document is aimed at providing context for evaluating each Digital Twin development step.

The focus of the whitepaper includes

- A typical roadmap of a Digital Twin project
- Step-by-step guide for development of a Digital Twin
- Possible risks, challenges, and solutions
- Variety of options and approaches on each step
- Practical examples and insights about Digital Twins

Digital Twin Concept

Digital Twin models can be conceptualized differently and according to various levels of abstraction. In this whitepaper, we'd like to suggest the following Digital Twin structure



MANAGEMENT LAYER This applies the required changes to objects or processes. It also includes an admin control panel, user interface, and dashboards for displaying data

DECISION-MAKING LAYER A set of tools to identify possible scenarios manually or automatically

ANALYTICS AND SIMULATION LAYER
An object or process-based virtual representation layer, allowing you to replicate its behavior with all relevant features and then analyze it

DATA INTEGRATION AND STORAGE LAYER
Data collection and preparation for future usage

IOT / CONNECTIVITY LAYER The data acquisition and transfer layer

Setup

There is a vast list of potential Digital Twin applications. We decided to narrow it down by picking an example applicable to several solutions and industries.

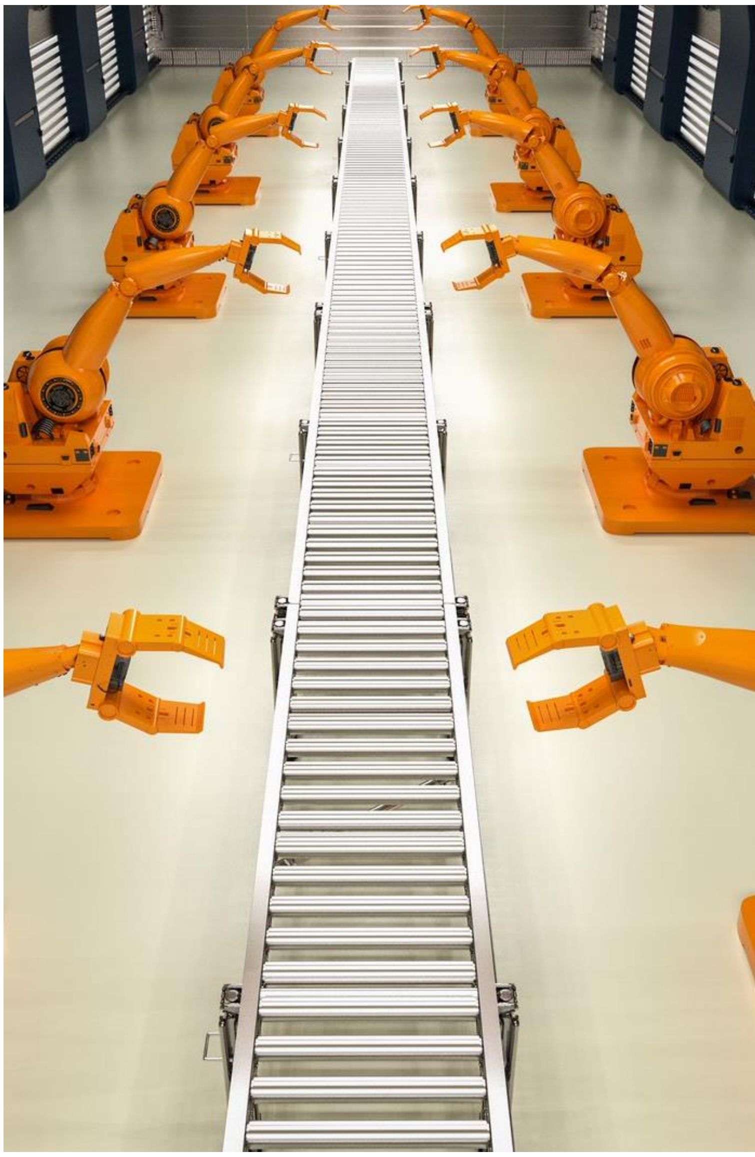
Conveyors are a significant part of almost every industrial project. They are widely used in warehouses, manufacturing, and heavy goods transportation, making it a good example of a way to maximize efficiency. At the same time, a broken conveyor may become a bottleneck for all operations resulting in quite measurable lost revenues.

In this whitepaper, we will describe a conveyor predictive maintenance system as an illustration of Digital Twin development. Our goal is to show the existing equipment's modernization process and introduce AI-based monitoring, simulation, and analytics technologies.

The cloud-based solution will offer real-time performance data, trigger alarms, provide historical data, and make maintenance suggestions and predictions.

Our example will be a fully automated conveyor with different activation and sensing devices across the belt. When all pieces of the conveyor work fine, there are no issues with the line. If something happens to one segment (e.g., a motor stops working), it causes additional expenses and affects the whole process. As a result, a local facility issue may cause supply chain delays in general.

Even though the conveyors have routine maintenance services, some mechanical parts may occasionally break down. We aim to develop |a Digital Twin solution to track equipment performance in real time and predict maintenance according to its current state.



IoT & Connectivity Layer

IoT network creation requires evaluating several things. Here's a list of some of them:

- Protocols
- Scalability
- Suggested traffic and message size
- Internet connection methods
- Security
- Equipment integration
- Device certifications

Brownfield innovations in manufacturing require some initial equipment investigation. The primary question is to understand whether the equipment can monitor its state and share data or if it needs some modernization by installing new sensors or devices to integrate with it. If it's the latter, we will add additional sensor (or device) integration steps to development of this layer.

Every manufacturing or industrial project has its constraints and preferences. Let's consider the most important points for our example.

Our conveyors are a circulatory system of our facility. Stopping it means that all operations will be delayed. Imagine the IoT network was compromised and started sending fake data.

> Security can't be over emphasized in this scenario, so we need to consider it.

The next step is to understand **which degradable parts we need to monitor**, how to install sensors there, and what data we need. Once we understand the number of sensors, we can clarify the device density, possible protocols, and types of devices.

Device certification is an important feature if there are any compliance regulations. Usually, it affects the total price of the sensors installed in your equipment.

IoT network scalability should be considered if some new equipment or new facilities may be added later.

Now that we understand our priorities better, it's time to continue our data research. Based on the conveyors' technical documentation for motors, gearboxes, and conveyor shaft bearings, we can start making suggestions. The most obvious are levels of vibrations and overheating. When something is wrong, the mechanical components start to vibrate and heat. That is the first hint of malfunction. So, can we somehow detect when such processes begin to happen, not when values are already out of a reasonable range?

Timeline ballpark: 1-3 months

Activities

- Requirements collection
- Priorities evaluation
- Technical documentation investigation
- Sensor and device installation
- IoT network setup

Data Integration and Storage Layer

This development phase usually consists of the following activities:

- Data collection and storage
- Data preparation (cleaning, aggregation, etc.)

Activities listed above specify the primary directions, though the whole picture will look more complex depending on the case. However, in most cases,

> our system should be able to stream and collect data, store historical data, and enable its usage for analytics.

Also, it's a good time to understand what additional data will be required in the future. Data enrichment may include weather conditions, geographical location, power consumption, inspection results and reports, maintenance logs, and even demographic statistics.

It's important to **design the data layer according to the solution's business goals .** Suppose our Digital Twin's goal is to predict the equipment performance and plan maintenance with a certain time of response. In this case, the solution will require the near real-time data streaming technology stack and probably some edge computing.

So, what can we use for our Digital Twin of the conveyor equipment? Here are some ideas utilizing the AWS stack:

- Build a data lake to collect all historical data available for analytics and data science. With Amazon S3, AWS Glue, and Amazon Athena, you can pilot your data lake quickly and low cost. It could be extended with Amazon Kinesis or Amazon MSK to support near real-time use cases.
- Build a data warehouse to fulfil BI and analytical use cases. With Amazon Redshift, AWS Glue, and Amazon QuickSight, it is possible to pilot DWH with a data integration layer and BI.
- Amazon RDS for PostgreSQL. It may be somewhere in the middle between performance pricing, and simplicity.

There are many other options. AWS is one of the most popular platforms, but others also have their equivalents for these services. The idea is to demonstrate how architectural decisions can be made.

Our conveyor Digital Twin doesn't have to be ultrafast, so there's no need to pay extra. But it shouldn't be too slow or overcomplicated either. So, let's pick the sweet spot: PostgreSQL.

Timeline ballpark: 1-3 months

Activities

- Evaluating data collection options
- Long-term data storage
- Development activities

Analytics and Simulation Layer

It can be hard to formalize the development of this layer, as it depends on the characteristics of the system we are trying to build. But we introduced our theoretical conveyor example precisely for this reason. It should help us illustrate the required activities to be performed.

> It's also important to understand the data we collect and how it influences our equipment.

Usually, a data science team runs a feasibility study to analyze how data affects the system and what it means.

In our conveyor example, we collected information about existing conveyors and possible variations of these machines. Usually, the technical specifications rarely contain the level of possible vibration acceleration, though information about suggested temperature ranges is quite common. Let's suppose that several sensors were installed on the actual conveyor to collect test data. For this example, we have decided to store the data in a straightforward PoC database (e.g., AWS RDS for PostgreSQL) to validate the possibility of collecting and storing data. And, of course, **our goal is to capture malfunctions in our data sample.**

Normally, motors are stable, so gathering enough failures may take some time. And the engineering team performs visual inspections to confirm all abnormalities along with the data science experts investigating all the data.

Is it possible to collect this data faster? Why not? Let's generate these faulty states ourselves. We can create a testing environment. To test the motors, we can install a three-phase motor with a VFD (Variable Frequency Driver) in the lab to collect data from it. We simulate malfunctions by installing artificial disbalances on the shaft and collecting sample data.

> At this stage, there are data sets for expected behavior, artificial malfunctions from our testing lab, and actual failures from our equipment monitoring. It should make our failure predictions more accurate.

The temperature regime from the specs and the abnormal vibrations from our tests allow us to set up a threshold-based alarm. Even such a simple approach will enable early abnormal behavior detection of mechanical parts during the first stages. These malfunctions can be repaired by a maintenance team and we can expect continuous data collection for new failures.

Timeline ballpark: 1-3 months

Activities

- Data science feasibility study
- Testing environment creation
- Simulations
- Alarms setup
- Prototype implementation

Decision-making Layer

We already introduced a simple monitoring tool using the equipment's data thresholds to detect deviations in the previous section. It can work surprisingly well in most cases, but the whole goal of the Digital Twin approach is to take it several steps further.

Based on our experience, the Machine Learning approach should be able to deliver more precise results. Let's use the LSTM (Long Short-Term Memory) algorithm. If we feed a neural network of the collected sensor data, we should be able to predict the future behavior of each mechanical part. Alerts will be raised when the actual data diverges from the predicted values of the LSTM.

So, we feed the LSTM with historical data up to the current moment, predict the current value and compare it with the actual one. Usually, this approach works well if we interact directly with motors. But there are always exceptions.

In one of our projects, the conveyor motors were controlled with the VFD by PLCs (Programmable Logic Controllers located in the conveyors' electric boxes. These motors are enabled in a random period based on the conveyor's sensors and a PLC program. Even though we had access to PLCs and read the states of the motor right from the PLC, it was impossible to develop a suitable algorithm to predict the motor's behavior during on-off cycles. We tried several other universal algorithms and existing models. But none of them were successful. The rate of false alarms and missed detections was too high.

The project requires a more advanced approach in these cases, and developing a Digital Twin solves these issues. **The universal models are trained to represent processes as black boxes without any detailed understanding of everything happening inside. But the Digital Twins do not require the training. They are based on physical models. Hence, we compare the predicted data with the physical model, not the black box.**

So, once the simulated physical model is developed, its behavior and measurements are compared to the equipment's actual data. The following features are usually available from the specifications:

- The supplied amount of the power
- The motors' efficiency
- The motors' load
- The thermal capacity of materials
- The physical weight of components

Having this data and using physics allows us to predict how the motors generate heat and change their temperature.

The Digital Twin model correlates better with the real data from the equipment than other models. And its benefits enable more precise results using a ML approach. The stage's outcomes allow simulation of the equipment's behavior, predicting its future state, detecting deviations earlier, and making decisions about its performance.

Timeline ballpark: 1-3 months

Activities

- Data science feasibility study
- Testing environment creation
- Simulations
- Alarms setup

Management Layer

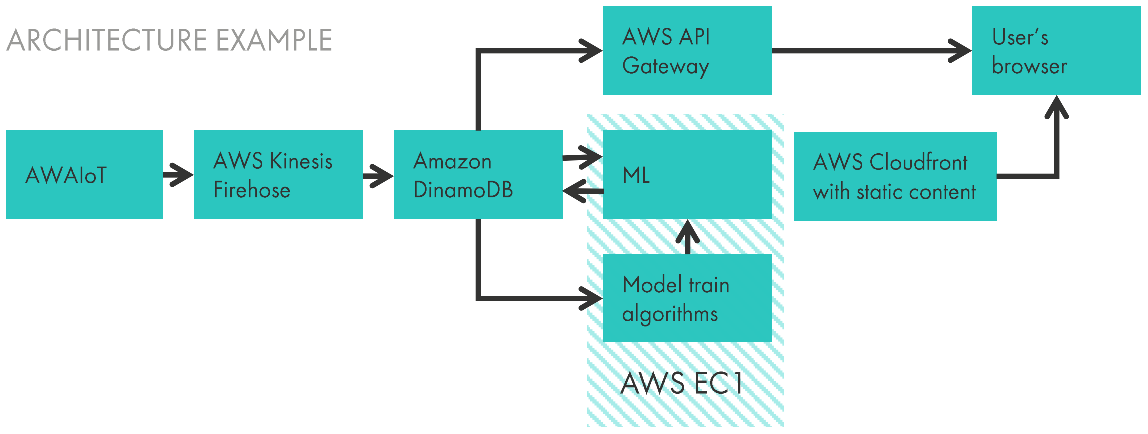
The goal of this stage is to provide means of process visibility, control, and automation (if required). Typically, the system should be able to visualize the historical data, provide all the required reports and predictions, show real-time feeds, and manage the equipment according to the decisions that were made.

We'll use a **serverless cloud architecture** to finish the implementation of our Digital Twin solution. It allows us to run all our logic in parallel, automatically solves the scalability challenges, and generally saves development time and effort. We should be able to easily connect at least several facilities with dozens of conveyors and thousands of sensors. And our algorithms will detect malfunctions in each one of them. **Cloud scalability makes it possible to run this algorithm for each sensor and host a web UI, an alarm system, user management, and some service components.**

Popular cloud providers have already introduced their Digital Twin frameworks, which can save some time in development and help make architecture decisions.

There may be some limitations to not allowing cloud infrastructure usage. For example, high-load systems will consume lots of cloud resources affecting the billing, some security risks, a vendor lock, decision-making time, or the existing infrastructure integration preferences. It will require a thorough analysis to pick the right option. So, if the cloud enablement roadmap doesn't fit your business, there are always other ways to achieve the results.

ARCHITECTURE EXAMPLE



We already have a connectivity layer to send all the management signals to the equipment. And our simulation layer provides insights about the equipment's performance and the resulting decision-making process. Once the management layer is fully developed, it will send alarms, show detailed performance information about our digital twin system, and control the whole process automatically or by sending information to the maintenance team.

Timeline ballpark: 1-3 months

Activities

- Dashboards development
- Visualization
- Facility integration
- Digital twin control system development
- Solution deployment

Find out more: https://www.dataart.com/services/industry-4-0

Summary

Brownfield innovation has many challenges, but this process can be made manageable with the right approach. The conveyor Digital Twin forecasts component degradation and provides a means for automatic disaster recovery. This example demonstrates the implementation steps addressing different aspects of possible requirements and evaluating development options and their effect on the outcomes.

Modern IoT technologies provide the equipment for connectivity despite their initial design . As a result, they open agile and cost-effective options for equipment modernization and other digitization choices.

Digital Twin solutions depend on the features of the systems they copy. And experience shows that simulating some objects or processes may be tricky. Most companies prefer to stick to the fail-fast strategy instead of spending significant resources pushing only one approach forward. Generally, this framework allows making the development cycles faster and keeping eye on the final goal.

Digital Twin solutions have benefitted from introducing AI / ML technologies. AI / ML model development is an iterative process, so sometimes the most popular options don't fit the equipment specifications or other limitations. In this case, the fail-fast strategy will enable the vendors to consider different scenarios.

The proposed 5-layer model splits the digital twin model into functional parts, which makes it easier to understand and design the system. Each layer development includes a set of activities that may be considered general guidelines.

Most cloud providers consider Industry 4.0 technologies crucial to their offerings and provide pre-designed frameworks and services to develop Digital Twins, predictive maintenance, supply chain, and other solutions.

The development timeline deeply depends on the scope of your project. As mentioned before, the Digital Twin solution described here consists of 5 layers. Usually, some parts of its development can be done in parallel, saving a significant amount of time. The first results can be seen after 5-7 months. The timelines provided here offer ballparks estimates for each stage.

The Digital Twin concept is hardly new, but modern technologies make it relatively easy to implement this kind of solution. As a result, Digital Twins can provide detailed data insights into issues that affect businesses.

The value of our Digital Twin example correlates with the scale of the conveyor usage. It may seem insignificant for a small warehouse, but it becomes irreplaceable for large-scale facilities, saving resources, time, and money.

> This whitepaper focused on developing the Digital Twins in manufacturing or industrial projects. DataArt is experienced in consulting and implementing Digital Twins in a variety of domains and business verticals.

DataArt

DataArt is a global software engineering firm that takes a uniquely human approach to solving problems. With over 25 years of experience, teams of highly-trained engineers around the world, and deep industry sector knowledge, we deliver high-value, high-quality solutions that our clients depend on, and lifetime partnerships they believe in. Headquartered in New York City, DataArt now employs over 6,000 professionals in 20+ locations throughout the US, Europe, Middle East and Latin America, with clients including major global brands like Unilever, Priceline, Ocado, Flutter Entertainment, among others.

Find out more about our Industry 4.0 offers: https://www.dataart.com/services/industry-4-0

Authors

Max Ivannikov

Max joined DataArt in 2013, where his passion for innovations and new technologies led him to IoT-focused expertise in the company. Now he is involved in delivering Industry 4.0, automotive, Smart City, and other digital transformation projects.

Prior to DataArt Max changed many roles and positions. His extended experience allows him to focus on combining creative and innovative approaches to solve emerging challenges with the best results.

Nikolay Khabarov

Nikolay has been working in the IT industry for over 14 years as a senior software developer, an embedded software engineer, and an architect. He has a specialist degree in the radio engineering.

Nikolay has developed many IoT projects for numerous hardware devices from tiny microcontrollers to modern ARM CPUs and related software for cloud backends, machine learning, end user applications and Android applications. Especially he likes working with edge technologies, on R&D projects, and invent new approaches.

He's passionate about ideas of open source software, enjoys public speaking and likes to share his own projects and experience with the community. In free time Nikolay likes to develop his own mini DIY projects related to automotive and computer numerical control machines.