POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Digital twins management with blockchain technology: a case study for a new protocol implementation

Laurea Magistrale in Aeronautical Engineering - Ingegneria Aeronautica

Author: Niil Petruccioli

Advisor: Prof. Sergio Ricci

Co-advisor: Stefano Francesco Pitton

Academic year: 2022-2023

1. Introduction

A digital twin (DT) is a virtual replica of a physical object, process, or system that extends the dimension of its physical counterpart, the physical twin, to the virtual space, realizing a bidirectional connection in which one may influence the other. This technology is a relatively new concept that has gained significant attention and traction in recent years. It is considered one of the main fields of research of the next future.
The fundamental principle of a perfect DT is that any information that would be available by inspecting the physical twin is also available through the DT diagnosis and prognosis capability. A system difficult or impossible to be physically inspected, such as a vehicle in a space mission, can be inspected through its DT. Moreover, any input given to change the DT's state will be reflected in the physical twin.
DTs will be powerful entities, capable not only of simulating the physical counterpart but also to predict the future state of it, evolving with it and taking preventive actions from their diagnosis and prognosis capability. This could lead to security issues regarding DT integrity. If the DT is maliciously manipulated, this could result in wrong predictions that may escalate to the complete failure of the system the DT represent.
As DT technology evolves, issues about the security of these advanced models have to be considered. In the era of Industry 4.0, blockchain technology can help enhance the protection of these models from malicious tampering and unauthorized changes. In this thesis, a blockchain application for DT life-cycle management has been developed starting from a conceptual case study of a damaged composite plate for which a structural health monitoring DT has to be realized. The DT model realized is based on a feedforward neural-network. The blockchain realized implements a distributed training protocol that will enable the re-training of the DT by authorized users when new data will be available through the continuous collection of the physical twin sensors.
Sections 2 and 3 provide an overview of, respectively, DT and blockchain technology. Section 4 describes the case study. Section 5 provides the implementation details of the blockchain application.

2. Digital twins

DTs concept origins in 2002 from Dr Michael Grieves who theorized this framework for prod-

uct lifecycle management. NASA was the first to use the term “digital twin” in its 2010 draft-version of its technological roadmap. The idea behind this concept was to build "an integrated multi-physics, multi-scale, probabilistic simulation of a vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to mirror the life of its flying twin" [6]. In NASA’s concept, DT technology was based upon extremely sophisticated physics-based models, but now also data-driven or hybrid models are used to model DTs.
NASA acknowledged that this technology would have been essential for space missions since it can provide (i) the possibility to simulate future missions before the actual start of the mission, (ii) monitoring the system during operation, extending its life by performing prognosis and diagnosis analyses, and (iii) using the DT for certification through simulation, reducing developments’ time and cost.
Publications about DTs saw exponential growth, starting to bloom in 2015 [7]. This surge has however generated confusion about the concept. DT technology comprehends various levels of maturity of DT implementations. The *DT prototype* or *pre-DT* is the first level of the implementation and is used before the physical part is created, to simulate its working conditions and design it in the best way for its specific application. A *cognitive DT* is the most high level of implementation and is an intelligent DT which possesses abilities of perception, attention, memory, and reasoning and can operate independently of the physical twin [3]. Based upon the implementation level of the bidirectional connection between physical and digital world: (i) *digital model*, where no connection is established; (ii) *digital shadow*, where only the connection from physical to digital is implemented (the DT can’t reflect automatically itself on the physical twin); (iii) *digital twin*, where the bidirectional connection is realized. Li et al. [3] found that the most common DT implementation found in literature was a *basic DT*, which lacks the ability to evolve automatically with its counterpart when the environment changes and the ability of autonomous problem-solving. Currently, many big players in the industry use or plan to use the DT technology. Among these there are NASA, Airbus, Boeing, General Electric, Siemens, British Petroleum, Dassault Systems. Fields of application of this technology are many. The fields in which it is most used are manufacturing, aerospace, energy, civil and healthcare.

3. Blockchain technology

Blockchain technology is a relatively new technology, born in 2008 for financial applications [4], and constantly evolved over the years. Thanks to its great versatility, it can be applied also in other fields.
A blockchain is a type of *distributed ledger*. Distributed ledger technology (DLT) is a multi-party system in which a shared database (the ledger) is managed and maintained with no central operator or authority, despite parties who may be unreliable or malicious (”adversarial environment”). In a blockchain, the ledger assumes the shape of a chain of blocks. A block contains the inputs sent to the blockchain by the users who interact with the blockchain, and it is created after a certain time has passed from the previous block creation.
A blockchain is based upon a distributed system of interconnected machines where each machine state is the exact replica of the others’ machine states. For this to be possible, inputs must be deterministic and the input sequence must be agreed upon, because in general inputs are not commutative. A blockchain is composed essentially of three layers: an *application layer*, a *consensus layer*, and a *networking layer*. The application layer manages the state transitions, the consensus layer solves the problem of providing an agreed-upon version of the inputs’ sequence, and the networking layer takes into account the broadcasting of users’ inputs, called *transactions*, that are what users send when they want to change the blockchain state.
Common functions used in blockchains are the *hash functions*. These are functions that take in input a message of arbitrary length and produce as output a fixed-length message, called digest. Cryptographic secure hash functions must preserve two fundamental properties: (i) *Collision resistance*, so it must not produce hash collisions (where two messages produce the same digest); (ii) *One-wayness*, so it must not be infeasible for an attacker to revert the process without unlimited computational power. The most used

hash function in blockchain applications is called SHA256, which produces a 256-bit hash digest from a message of 2^{64} bit length.
A user of a blockchain is identified by an *address*, which is a unique alphanumeric string. Blockchain technology is based upon *public-key cryptography*. A user owns a couple of keys: a public key, that can be shared with anyone, and a private key, which has to remain secret. Transactions to the blockchain are digitally signed using the private key, and the resulting digital signature can be verified by anyone with the public key of the user (which is sent along with the digitally signed transaction and registered by the blockchain application). The private key must remain secret because the public key is derived from it and the address is derived from the public key hashing. These derivations are based upon cryptographic functions that do not allow to revert the process without an unlimited amount of computational power.

4. Case study description and digital twin model

The case study under exam consists of a composite plate with a central cutoff, instrumented with embedded fiber optics Bragg sensors, damaged and under tensile load. Damages reduce the strength properties of the plate and a reduction in the ultimate load is expected, correlated with the damage entity. A DT for the plate under exam has to be realized for structural health monitoring of the plate. This case study is a conceptual study: plate under exam has not been built and the physical twin is represented by the finite element model of the plate.

4.1. Case study description

The plate has a length of 1200 mm and width of 600 mm, and an internal rectangular cut-off, centred in the plate center, of 450 mm length and 150 mm width. Plate geometry and laminate coordinate system are represented in figure 1. The plate is made of composite material obtained from lamination of unidirectional graphite-epoxy plies and it is instrumented with 8 embedded fiber Bragg grating (FBG) sensors disposed around the central cutoff as depicted in figure 1. These sensors will measure the local strains in the x direction of the global coordinate system, which is aligned with the longitudinal direction of the plate. Composite material choice has been made based upon an experimental study found in literature that has been used to perform preliminary numerical simulations with MSC Nastran and validate the results comparing them to what reported in literature. Composite material used is T300/1034-C, which properties are defined in 1. The lamination sequence is $[0/(\pm 45)_3/90_3]_s$, for a total of 20 plies, and each ply has a thickness of 0.1308 mm, resulting in a laminate of thickness 2.616 mm.

Property	Value
E_{xx}	146858 [MPa]
E_{yy}	11376 [MPa]
G_{xy}	6185 [MPa]
G_{yz}	6185 [MPa]
G_{xz}	6185 [MPa]
ν_{xy}	0.3
X_t	1730.5 [MPa]
X_c	1379 [MPa]
Y_t	66.5 [MPa]
Y_c	268 [MPa]
S	133.7 [MPa]

Table 1: T300/1034-C material properties.

Plate is clamped on the left edge (AB). On the right edge (CD) the plate is free only to translate in the global x direction. Plate will be loaded in traction along the longitudinal direction resulting in an in-plane state of stress. Load is applied as imposed displacement on the right edge.

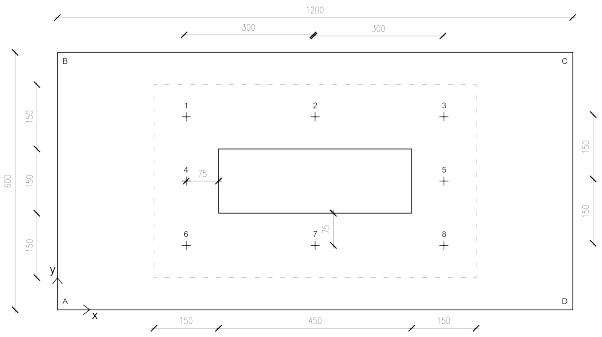

Figure 1: Geometry of the composite plate. +: FBG sensors. Units in mm.

The plate may be damaged by small accidental damages or manufacturing defects. The maximum damage entity is assumed to be that which does not reduce the ultimate load of the plate

under 70% of the undamaged ultimate load. Damages are assumed to be located only in the region defined by the dashed line in figure 1. The ultimate load of the plate is 268.8 kN (it has been determined by numerical analyses on the undamaged model of the plate). The composite plate is intended to operate in a loaded condition in the range of 100 kN to 150 kN. a finite element model has been realized and analyses have been performed using MSC Nastran. The plate has been modelled with shell elements of type CQUAD4 due to the nature of plane state of stress. Mesh elements are square elements of 10 mm edge. Each sensor is represented by the element in which the sensor is located, for a total of 8 elements representing each a different sensor. Strains measured from these are obtained from the analysis strains outputs of these elements. Material property definition in Nastran is obtained with cards PCOMP and MAT8. The degradation model considered is instantaneous degradation in wich the post-failure residual stiffness fraction has been set to 0.01 in MATF card. This last value has been set based upon the results of preliminary analyses that recreated an experimental study of a specimen in traction found in literature. Material failure is considered using the MATF card. Failure criterion used is a max strain criterion, with max strains derived from the material max strengths reported in table 1.

4.2. Digital twin model

The DT model realized is a neural-network model built using PyTorch library. It has been modelled starting from what S. Pitton done in his research [5].

DT model has as input parameters the strains measured from the FBG sensors and produces as output: (i) the prediction of the ultimate tensile load; (ii) the ratio between the plate stiffness and the undamaged linear model stiffness; (iii) a prediction of the same ratio of point (ii) that will be observed if the applied load is increased of 20%.

To generate the DT model, first a database resulting from different analyses on the composite plate has been generated. The composite plate region subjected to damages has been divided into twelve square patches evenly distributed along the damage region, in a similar manner of [2]. To simulate a degradation, the patches' stiffness properties have been scaled down by a knock-down factor. Each patch has been analyzed for a knock-down factor varying from 10% to 60% with a 2% step. Resulting analyses have been post-processed to eliminate analyses failed due to convergence problems or violating the assumptions made in 4.1. Since the plate's operation range varies from 100 kN to 150 kN, the output quantities included in the dataset are obtained for input loads in this region, extended also by 20 kN to add an extra margin to the DT model knowledge. Numerical analyses are performed using the modified Newton-Raphson method implemented in MSC Nastran. A multistep approach has been used for the displacement step increments. This has been done to reduce computational time. Load increment steps considered are finer going toward the ultimate failure region, where failure is expected, for a total of 71 steps. The resulting dataset is composed of 400 design points and has been split into training, validation and testing sets considering 60% for training, 20% for validation and 20% for testing. Figure 2 shows the envelope obtained from the analyses that has been used to generate the dataset.

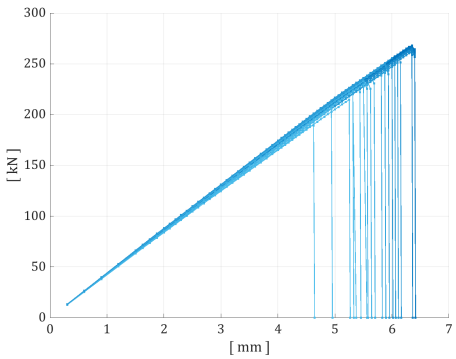

Figure 2: Load-displacement envelope for the damaged plate.

Cost function used for the optimization problem of the neural network is the Mean Square Error (MSE) and optimization is performed using Adam optimizer with parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1e-8$ set as found in the original Adam implementation. Activation functions considered are rectified linear units (ReLU). Weights have been initialized with Xavier initialization.

4.3. Results

The following accuracy metrics have been defined.

1. R square error (R^2), called also coefficient of determination, which defines the overall accuracy of the model and it is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{N}(\bar{y} - y_i)^2} \quad (1)$$

2. Mean Absolute Percentage Error (MAPE), which represents the average absolute error between the predicted outputs and the correct ones:

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}|\frac{\hat{y}_i - y_i}{y_i}| \cdot 100 \quad (2)$$

3. Relative Maximum Absolute Error (RMAE), which is a local indicator of the accuracy, indicates if the model fails to predict accurate values for some specific output.

$$RMAE = \frac{max_i(|\hat{y}_i - y_i|)}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(\bar{y} - y_i)^2}} \quad (3)$$

The best result is obtained from 3 hidden layers and 30 neurons per layer with a learning rate of 0.0001. Table 2 resumes the best results obtained.

Metric	Train	Test
R2	0.991	0.846
MAPE	0.002	0.008
RMAE	0.427	0.891

Table 2: Performance metrics of the best neural network model generated.

For the testing set, R2 is high and MAPE is very low, meaning the model captures very well the dataset relations. RMAE is high, meaning that there are points in the dataset that the network fails to correctly predict. Testing set outputs show that the stiffness prediction error is very close to 0%, with spikes in some points around 1%. For the life prediction the neural network performs well, but there are error spikes around 10%-12% that should be handled. The resulting overall performance is thus very good, but the spikes in the life prediction error should be addressed for the DT model to be the best reliable as possible. The dataset should be refined, adding more analysis and trying to consider smaller patches. Analysis used to generate the database could be differentiated in load increments so that more values of failure points can be taken into account. Also, the sensors' placement should be investigated to find the optimal position. These are beyond this case study purpose, so these refinements have not been done in this work.

5. Blockchain life-cycle management

In situations where a consortium of different corporations cooperate in the creation and updating of a complex DT, then a blockchain solution can enhance the DT life-cycle management's security. The different entities inside the consortium would maintain the blockchain network and if sufficient decentralization is reached then the resulting blockchain network will securely store the updatings history. For decentralization it is meant the spread of decisional power among the different entities of the network. Where sufficient decentralization is reached, it will not be possible for a small group of entities to manipulate the data recorded in the blockchain, and so the updating history will be secured.

To manage the DT lifecycle, a new blockchain, Vesta, has been developed, which also integrates a distributed training protocol. It is available at https://github.com/niilptr/vesta. The application layer has been developed using the Cosmos-SDK framework (https://v1.cosmos.network/sdk), while consensus and networking layers are handled by Tendermint consensus engine [1].

In this implementation a central server will be used as not-fully trusted side store, where DT models will be saved.

A DT is represented in Vesta as a data structure that records four fundamental properties: *name*, *creator*, *hash*, *last updater*. The DT hash is used as identifier of the DT. It will be the hash of all the files that composes the DT, where complex models have to be handled. The full DT will be stored in remote server(s), and its hash will be saved on-chain. If any malicious manipulation of the twin happens, the on-chain hash would differ from the remote twin hash and this will prevent using compromised twins. The last modifier is

saved to take actions in response to malicious modifications of the twin. Users will interact with the blockchain by sending transactions to it. Transactions (also called *messages*) are: (i) `create-twin`, to be used when a new twin has to be instantiated; (ii) `update-twin`, to update the twin hash from an authorized address; (iii) `delete-twin`, to be used when the DT has been dismissed and there is then no means to continue storing it on-chain; (iv) `train`, to be used when the DT needs to be retrained from a new database expanded with lifetime operation data collected from the physical twin sensors.
Along with these messages, other two "satellite" transactions have been implemented to handle the distributed training. These are `training-phase-end` and `best-result-is`, and are two confirmation messages that will be broadcasted by trainers involved in the training procedure. Their role will be explained in section 5.1.

5.1. Distributed training protocol

For *distributed training* it is here intended a neural network training scheme based on parallelizing different training jobs taken from a training population among different nodes of a blockchain.
In distributed training, results will be different from trainer to trainer since each trainer will perform a different training job from the whole training jobs population. Blockchain nodes will instead have to reach consensus among a new state, deterministically. Moreover, results must be validated by the blockchain protocol, otherwise it would simply be more convenient to use a central authority and get rid of the distributed ledger. In this section a protocol which can solve these problems is presented.
In Vesta implementation the distributed training is based upon a group of authorized accounts, called *trainers* that will start a local training job on their machines and agree on the training best result.
The distributed training procedure will act in the way described below.

1. When the training message is received the training phase is started.
2. In training phase, blockchain application triggers each trainer node to start the local analysis.
3. Local analysis program makes trainers who complete the training upload their results to the central server.
4. In training phase, blockchain application triggers the following behaviour on nodes authorized to read the central server:
   (a) they check periodically if all trainers have uploaded their results on the server;
   (b) in case they witness that all trainers completed their job they inform the blockchain sending a specific `training-phase-ended` message. This transaction is registered in the blockchain.
5. In training phase, if a sufficient number of nodes, greater than a predetermined threshold, agrees on the ending of the training phase, then training phase is ended and validation phase is started.
6. Eventually a predetermined timeout for the training phase is reached and validation phase is started.
7. In validation phase, blockchain triggers each authorized node to get the training results from the central server and select the best result following a predetermined strategy. If the result is not valid then the next best result is analyzed. Eventually, each node selects a result (or a `none` result) and informs the blockchain by sending a specific `best-result-is` message.
8. Eventually a predetermined timeout for validation phase is reached and validation phase is ended.
9. Blockchain application checks periodically for agreement on the best result. If agreement is reached (greater than a predetermined threshold) the twin state is updated with the new state. If no agreement is reached, or agreement on `none` result is reached, then the twin state is left unchanged.

Implementation details are too long to be reported here and can be found in chapter 5 of the thesis.

5.2. Distributed training testing

The protocol has been tested on a local network with 3 trainers. Tests have been conducted to test the application handling of unauthorized in-

teractions.

1. Not allowed accounts were correctly detected by the application when training was requested including unauthorized accounts in the training configuration file.
2. The application correctly refused to start the training procedure when the training configuration file uploaded to the remote server and the one specified in `train` message were different.
3. A test where a results file was modified after its uploading has been done. Results file was altered by changing the network parameters' values. The application correctly detected the hash mismatch discarding the altered results.

Training tests have been performed requesting for each trainer, in the training configuration file, a different learning rate value to be used in the training process of the DT model this thesis. Values set as learning rate were 0.0001 for trainer one, 0.0002 for trainer two, and 0.0005 for trainer three. Block-time has been set to 6 seconds.
Training requested has been correctly handled: agreement was reached in the different phases, results were correctly uploaded to the remote private server and the twin hash was been correctly updated with the new one.

6. Conclusions

In this thesis, the implementation of a blockchain solution for DT life-cycle management has been proposed. Vesta, a blockchain built using Cosmos-SDK, has been used to generate and manage the state of a DT.
The main target of this solution is consortia of enterprises that are involved in the developing and management of shared DTs and that choose to operate in a trustless environment to enhance security over data integrity and ownership, but also choose to adopt central databases to store the DT models. Models saved in these databases will have their unique identifier (hash) saved also on the blockchain ledger so that any unauthorized modification of the models saved in the central database will be detectable and consequent actions can be taken.
The blockchain developed also integrates a protocol to orchestrate distributed training for neural-network-based DT models. The blockchain solution proposed is based upon addresses authorized to manage DTs, or request training to the network, providing the input data to generate a pool of analyses that will be parallelized on specific nodes of the blockchain.
Vesta distributed training protocol can be extended to any optimization algorithm that starts from a pool of analysis that can be parallelized, and from which a unique best result has to be obtained. This includes also any analysis job in general that fits this analysis procedure. The application features can also be extended by implementing logics to automatically trigger the physical twin management from the DT diagnosis analyses.

References

[1] Ethan Buchman. *Tendermint: Byzantine Fault Tolerance in the Age of Blockchains.* University of Guelph, 2016.

[2] Michael G. Kapteyn and Karen E. Willcox. From physics-based models to predictive digital twins via interpretable machine learning, 2020.

[3] Luning Li, Sohaib Aslam, Andrew Wileman, and Suresh Perinpanayagam. Digital twin in aerospace industry: A gentle introduction. *IEEE Access*, 10:9543–9562, 2022.

[4] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Cryptography Mailing list at https://metzdowd.com*, 03 2009.

[5] Stefano Francesco Pitton. *Artificial intelligence techniques for the optimization of variable stiffness cylindrical shells.* Politecnico di Milano, april 2019.

[6] Mike Shafto, Michael Conroy, R Doyle, E Glaessgen, C Kemp, J LeMoigne, and L Wang. *Modeling, Simulation, Information Technology and Processing Roadmap.* may 2010.

[7] Maulshree Singh, Evert Fuenmayor, Eoin Hinchy, Yuansong Qiao, Niall Murray, and Declan Devine. Digital twin: Origin to future. *Applied System Innovation*, 4(2):36, may 2021.