SCTE CABLE-TEC EXPO FAST FORWARD 2021
ATLANTA, GA
OCTOBER 11-14
2021 Fall Technical Forum
SCTE • NCTA • CABLELABS

Seeing Double: Network Digital Twin

A Technical Paper prepared for SCTE by

Guy Meador III
Senior Solutions Architect
Cox Communications, Inc.
6305 Peachtree Dunwoody Road
Atlanta, GA 30328
404-269-5625
Guy.Meador@cox.com

George Cave
Principal Data Center IP Architect
Cox Communications, Inc.
George.Cave@cox.com

Table of Contents

Title **Page Number**

List of Figures

Title **Page Number**

1. Introduction

Operators are facing disruptive change to their networks driven by technological and business forces. The business seeks to increase innovation, reduce costs, and manage risk while network technologies including virtualization, software defined networking, streaming telemetry, and automation drive additional complexity into the network. Typically, an operator's functional disciplines such as engineering, operations, analytics, and planning each have separate, uncoordinated views of the network with sometimes overlapping, sometimes disjoint, data, leading to inefficiencies and uncertainty. Network Digital Twin provides a comprehensive view of the network, combining disparate data sources such as telemetry, monitoring, inventory, provisioning, analytics, planning, and network automation with variable-fidelity models of equipment, network functions, network services, network operations, and analytics, creating a separate digital representation of the live network. The network digital twin is an integrated, consistent, comprehensive, lively, and accurate model of the network and its constituents, connected to and representing the state and behaviors of the actual entities of the operating network. Operators use the network digital twin through the accompanying dynamic views and visualizations to better understand the state and behaviors of the network and, through manipulation of the digital twin, affect changes in the actual network. Functional disciplines within the operator's organization use the network digital twin to break down data silos and leverage the consistent model of the network as the basis for their individual missions. The paper addresses the formation, concepts, and implications of digital twins and their application to the operator's network, including an example from Cox's application of aspects of digital twin to our data center network.

2. System Architecture

The three top-level functional building blocks of the network digital twin functional ecosystem are depicted in Figure 1. The functional building blocks have significant internal composition (other functional blocks) and form an enduring basis for multiple logical and physical architectures. This architecture description, therefore, provides discussion, primarily, of key aspects of a functional architecture for network digital twin.

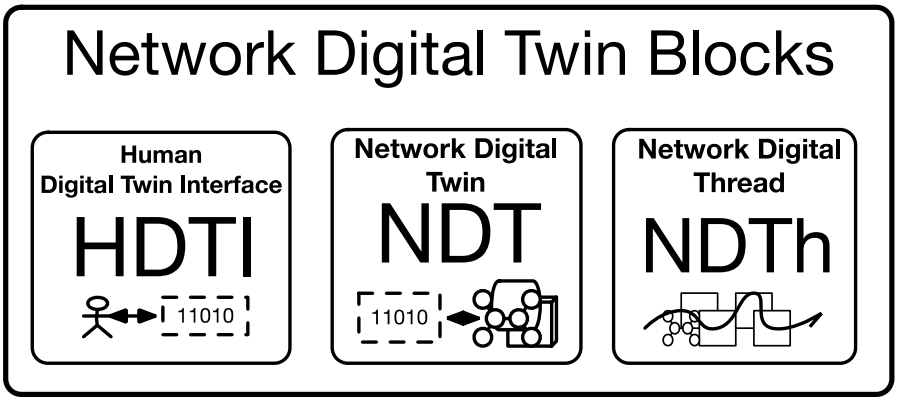

Figure 1 – Top Level Functional Blocks of the NDTE

The top-level functional building blocks of the Network Digital Twin Ecosystem (NDTE) are the Network Digital Thread (NDTh), the Network Digital Twin (NDT) itself, and the Human Digital Twin Interface (HDTI). These three top level building blocks work in concert to provide a transformative approach to the creation and operation of the network.

Network Digital Thread: The network digital thread contains engineering models for the network and its constituent subsystems and interfaces. Also included are any document-based information, specifications, drawings, images, and non-operational data relevant for the network that are created across the system development life cycle (SDLC), such as design and planning information.

Network Digital Twin: A network digital twin is a unified executing engineering model of an operating network entity coupled with normalized data pertaining to the actual operating network entity with which it is associated. The network digital twin can be queried and manipulated separately from the actual operating network entity, with the option for those manipulations to affect changes at the operating network entity. Network digital twins can be created to represent network entities at the logical, virtual, or service layer, as well as composites of any of these (including an entire network).

Human Digital Twin Interface: The human digital twin interface is a human-machine interface (HMI) providing the capability for a person to interact with one or more digital twin instances. It provides dynamic visualization of the digital twin instance and related data and supports human-driven query and intent-based manipulation.

2.1. Theory of Operation

The prototypical top-level functional connections and information flow between the network digital twin functional building blocks and the operating network entity are shown in Figure 2, adding an additional functional block to the mix: the Actual Twin (AT).

Actual Twin: The actual twin is the physical, virtual, or logical instance that is a counterpart to a digital twin instance. Examples of actual twins for the network include a network router appliance, a router network function in a virtual machine or container, and a layer 3 virtual private network (VPN) service.

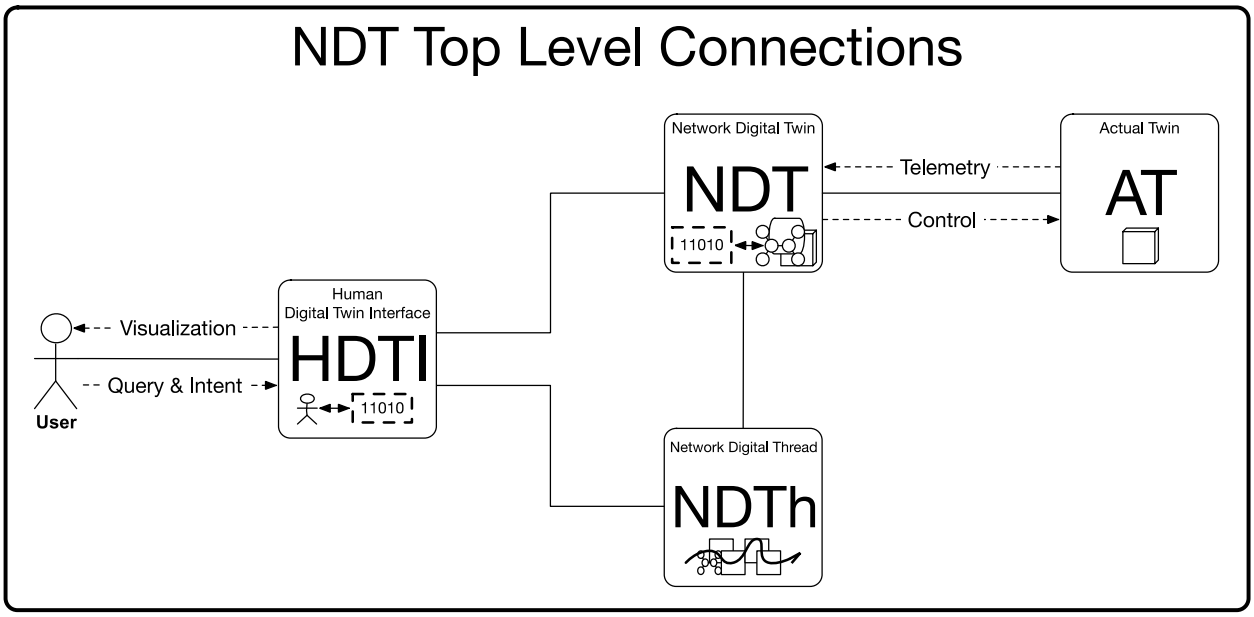

Figure 2 Top Level Functional Blocks and Connections

An overview of the theory of operation for a prototypical network operations situation using network digital twin is as follows.

An executable NDT instance is created with an integrated set of models pertaining to a specific type of network entity, e.g., the type of AT, in a given role in the operating network. The models are drawn from the Network Digital Thread, which is the authoritative source for this information. The NDT is also pre-initialized with data pertaining to operational intent for the instance.

The NDT instance is associated with an actual entity instance in the operating network, e.g. the AT. Telemetry data pertaining to the AT flows from the AT and other systems into the NDT. The telemetry data could be from streaming telemetry, monitoring systems, element managers, and other such intermediaries. The telemetry data is used to update the state of the NDT, causing the dynamic executing model to be evaluated, updated, and to react to the changing conditions of the AT. For example, the NDT may detect model constraint violations, a variance between operational intent and current state, or other such issues.

If the NDT models are constructed to provide control capabilities for the AT, then control signals are sent as needed toward the AT. The control signals could take many forms, depending on the specific implementation. The control signals could be of types such as configuration changes, commands, signals to intervening systems, and events. Incorporating this kind of model into the NDT provides intent-based closed loop control for the network.

The HDTI provides the user/operator experience for the NDT and the Network Digital Thread. It allows the user to see various static and dynamic views of various kinds pertaining to the network digital twin that are updated during operations. It also enables queries to be performed and the ability to express operational intent and other control input from the user perspective.

Ideally, any technical information from the Network Digital Thread is linked with the NDT and can be discovered and viewed using the HDTI environment.

2.2. Network Digital Thread

Network Digital Thread encompasses the engineering and analytical models for the network created across the entire system development life cycle (SDLC). The network digital thread building block is the digital thread created through digital engineering, with the subject being the operator's network.

The INCOSE Systems Engineering Body of Knowledge (SEBoK) states, in part, that digital engineering "…is the creation of computer readable models to represent all aspects of the system and to support all the activities for the design, development, manufacture, and operation of the system throughout its lifecycle. These computer models would have to be based on shared data schemata so that in effect a digital thread integrates all the diverse stakeholders involved…"; and further, "Everything from documenting requirements, technical reviews, architecture design, and so forth would be based on the models in a digital engineering environment (Vaneman and Carlson, 2019). The digital thread would be the authoritative source of truth concerning the system data." - [SEBoK/Digital Engineering].

The network digital thread contains engineering models for the network and its constituent subsystems, properties, interfaces, protocols, behaviors, and data. These models range across multiple disciplines, including network engineering, facilities engineering, reliability engineering, systems security engineering, software engineering, network planning, and operations engineering, to name but a few. The various discipline-specific models are all aligned and connected through the common context and consistency established and maintained in systems engineering models.

Within the engineering disciplines, a major class of model in the network digital thread are those models addressing forward engineering concerns. These models capture the "as-specified", and "as-designed" aspects for the network, addressing, for example, structural, behavioral, functional, non-functional, and operational concerns. In addition to the models that capture specifications and design decisions, this class of model also includes the accompanying engineering analysis models and engineering/test data sets.

Models of the forward engineering class are necessary to establish a correct model of the network as it is intended to be. They establish the interfaces, protocols, behaviors, data formats, and other characteristics of the network and its constituent subsystems, separate and independent of any hardware and software implementation. They are integral to the simulation, emulation, operational characterization, and operational control capabilities of NDT, forming the rational basis with which the live operational data is normalized and paired with the NDT.

One example of a forward engineering model relevant to NDT is a network engineering model for network layer 2-4 network functions, connections, configurations, and behaviors. A formal model is prepared for the functionality, interfaces, and behaviors of a give type of network function (ex. router forwarding), with the model providing state variable and state transition definitions, configuration parameter definitions, and the effect of configuration on those

behaviors. Other instance-independent characteristics of the network function can be included in the model. Given a sufficiently elaborated model, the designed layer 2-4 behavior of the network function can be determined given any set of internal state and external stimulus, as can an arbitrary network of many interconnected network function instances.

Another example of a forward engineering model relevant to NDT is a systems engineering model for a state machine-based autonomous operations closed loop controller. This type of controller is used at the network resource level (ex. device) or multi-resource network service level (ex. layer 3 VPN) to actively establish and maintain the operational state of the controlled entity according to operational intent. A formal model is prepared to specify the intent & configuration parameter definitions, states, state transitions, triggers, and actions of the controller, along with the input and output signals (ex. telemetry/events, control, configuration). Given a sufficiently elaborated model, the designed behavior of a closed loop controller can be determined given any set of internal state and external stimulus, as can an arbitrary set of many interconnected closed loop controller instances in communication with one another. Taken together, this type of model provides a key part of the executable specification for closed-loop controllers for network autonomous operations.

A third example of a model type relevant to NDT is a reliability engineering model for a manufacturer's network equipment model (in the last usage "model" means the manufacturer's model identifier rather than a formal engineering model of the equipment). A type of reliability engineering model is the Failure Mode and Effects Analysis (FMEA) model. The FMEA model identifies the inherent reliability characteristics of the equipment design, including the tree of relationships between possible failure modes and the resulting system performance degradation or failure. Given an FMEA model and a set of one or more fault conditions, the effect of the components failure modes to the system performance or safety can be determined and characterized. With relevant equipment test results and measurements from the population of a given equipment model in operation in the field, a predictive model for fault and failure probabilities can be created to live alongside the FMEA model. With a sufficiently elaborated predictive model of this type and the relevant time series state measurements for an instance of equipment, the probability of failure of the equipment can be predicted, indicating a need for proactive maintenance ahead of the failure. This second type of model may be developed using machine learning techniques and is closely associated with the area of predictive network maintenance.

Network digital thread entails more than models. For example, any document-based information or drawings and images relevant for the network created across the SDLC are part of the network digital thread. These non-model-based artifacts can be linked with the models, but do not play an active part in the executable network digital twin.

The preceding examples are but a few of the many types of models that are associated with network digital thread. Other examples include models for system security, network planning, network design (physical and virtual), and mechanical/physical design, among others. Some of these models are design models specifying correct characteristics of the described atomic or composite entities; others are analytical models of the resulting emergent characteristics of those

designs (ex. fault tree) or are predictive in nature based on the combination of the prior models with aggregated data from tests and measurements from the field.

All model types described above can be used for the purposes of engineering analysis. Some can be used for engineering discipline-specific model execution and/or simulation. Multi-discipline model execution and simulation is possible when coordinated through appropriate systems engineering models and co-simulation environments. As networks and network operations complexity rises, these techniques become more important to employ for the design and analysis of networks, their data, and their operation, even without the full capabilities of network digital twin.. They are used to support exploration of the network and operations design space and trade studies, for example, without connection to an AT.

2.3. Network Digital Twin

The models of the network digital thread designed for execution and simulation in the context of engineering design and analysis, along with the model execution platforms, can be developed beyond the scope identified in the previous section. Doing so enables a spectrum of model execution capabilities directly associated with an actual operating network and its constituent parts. A key characteristic of the approach is that the disparate models are linked together, coordinated, and updated with identical, or at least consistent, normalized, data during their execution.

For example, executable models from the network digital thread (at the needed level of fidelity and from the necessary engineering disciplines) pertaining to an instance of network equipment are brought together, connected, and co-executed in a coordinated fashion on a platform environment designed for that purpose. These executing models are then provided with data from and about the network equipment instance, forming a kind of digital copy of the equipment instance and its state. The digital copy consisting of these models bound with the network equipment instance's data forms the basis of an instance of a network digital twin for the actual network equipment instance.

The dynamic behavior, state changes, and other characteristics of the actual network equipment instance can be simulated, emulated, analyzed, represented, and predicted based on the network digital twin instance. The network digital twin execution platform provides additional data, from or about the actual network equipment instance, to the network digital twin instance as it becomes available.

The data pertaining to the actual network equipment falls into categories that include operational intent, target configuration and operational state, actual operational configuration and state, sensor readings (ex. temperature), network function-specific data (ex. firewall state, traffic counters, routing tables, etc.), and alarm conditions. Additional data about the actual network equipment instance can also include historical information, time-series data, failure data, scheduled maintenance actions, and other instance-related data.

The above data may be obtained for the network digital twin from a variety of sources, including telemetry and monitoring systems, element management systems, operations analytics systems, service provider data systems, or directly from the network equipment instance. The rate and

latency at which changing data from and about the actual network equipment is provided to the network digital twin will affect the extent to which the network digital twin instance reflects its twin's state.

Binding multiple, coordinated, executable models with a consistent set of instance data from an actual network equipment instance into the network digital twin instance enables the ability to achieve useful analytical and operational objectives.

The functional/behavioral models in the network digital twin instance combine with the actual twin data to enable analysis and simulation. For example, given a specification of network traffic arriving at one of the interfaces, the resulting forwarding plane and packet transformation functions and behaviors can be determined and/or simulated without further interaction with the actual network equipment instance. The case study later in this paper employs this type of network digital twin analytical model.

Anomalous or incorrect behavior or conditions in network equipment in the operational environment can be identified through the network digital twin's explicit and automatic checking of invariant conditions, constraints, policies, and state encoded in its models. Examples include the presence of disallowed configurations, variance between intent and operational configuration or operational configuration and relevant established policies, detectable functional behavior variances, and fault conditions.

With a network digital twin approach, the often-brittle workflow-oriented techniques using "golden config" templates and scripts for pre-checks and post-checks are eliminated in favor of an approach whereby the multi-discipline specification of correctness in context is expressed in the network digital twin models with continuous evaluation during model execution, and whenever new data is available from the operating environment.

Further, by employing state machine-based closed loop controller models as a part of network digital twins, the entire operational lifecycle of each actual twin in the operating network can be addressed, achieving intent-based network management with autonomous operations.

2.4. Human Digital Twin Interface

The human digital twin interface is a human-machine interface (HMI) providing the capability for a person to interact with one or more digital twin instances. It provides dynamic visualization of the digital twin instance and related data, and supports human-driven query and intent-based manipulation.

The HTDI is separate from the other functional blocks of the NTDE for several reasons. First, the pace of innovation between the NDT execution environments and HMI variety and innovation pace will be different; de-coupling these two facilitates the ability to take advantage of the pace of each independently. Second, the type of HDTI used for the same NDT is dependent on specific goals for the interaction, roles of the user, and so forth. Multiple HDTI experiences may be in simultaneous operation connected to the same NDT instance.

For example, an HDTI based on web technologies with user access via a web browser matches the typical network operations scenario of today. Static and dynamic views, queries, and controls appropriate to the user role and web browser environment are provided.

An application running on a mobile device in the field can also be created as an HDTI implementation, providing field technicians a new and improved way to view and interact with the autonomously operated network.

Another example of an HDTI is that provided by the class of tools, either web-based or native applications, classified as engineering design, simulation, and analysis tools. These tools are not used for operations but benefit from connection to simulation-only NDT instances, or connection to hybrid test environments containing a mix of simulation-only NDT instances and NDT instances associated to AT instances.

A final example with exciting promise for both operations environments and engineering exploration is that of virtual reality HDTI implementations. These will have a transformative effect on network operations centers enabling users to see and interact with the operating network in significantly new and effective ways.

In the operations environment, it may seem that HDTI is just another typical support portal, but that is not the case. The HDTI approach does not present an order/workflow-based view of the operating network, and user-initiated changes for the NDT are not user-initiated by manipulating orders. Rather, HDTI presents an active, intent-based way for the user to manipulate the NDT in a manner that is more natural and direct. With closed loop autonomous operations at work in the NDT layer, the user wields the HDTI to see the live state of the network and directly express intent for changes to the NDT, seeing those changes progressing and taking effect in the same view of the NDT. The overall approach embraces autonomous operations while hiding the complexities of its implementation, and, at the same time, provides improved visibility and control to the user.

2.5. Observations and Implications for Network Digital Twin

Applying network digital twin in the service provider network operations environment can take several forms ranging from passive uses for visibility, analytics, and verification/constraint checking on one end of the spectrum, to a full-up intent-based network with closed loop autonomous operations on the other.

The types and fidelity of models used in the NDT should be tailored to the specific purposes for which NDT is being used. An ideal network digital twin ecosystem should facilitate the flexible mixing, matching, coordination, and harmonization of the various models and uses.

Employing network digital twin effectively for many useful purposes does not require direct use of AI/ML at all. This is particularly true when using network digital twin for autonomous operations. Intent based network operation built upon executable models for state machine-based closed loop control, policy and constraint checking, service-to-resource configuration mapping, and other autonomous operations capabilities, along with integrated, consistent, and improved visibility, are achievable and valuable without direct reliance on AI/ML. It is preferable for the

automated operation of the network to be explicitly designed, implemented, and verified for desired characteristics through forward engineering disciplines. This approach ensures autonomous network operations remain consistent, explainable, and predictable.

On the other hand, AI/ML plays a vital role for the network digital twin ecosystem. It has already been discussed how machine learning plays an important role in engineering analysis and the formation of predictive models, playing a key role in the design of the executable models of the NDT. In connection with autonomous operations, machine learning has a key role to play in operations analytics (OA) within the network digital twin ecosystem, providing informational/advisory signals back into the closed-loop controller portion of the NDT. For example, the employment of OA functions based on machine learning and connected to the NDT used for evolving intrusion detection, identifying unusual network operational patterns, predicting equipment failure, and providing other operational recommendations. These OA advisory signals flow into the autonomous operations closed-loop controller layer of NDT; the controller layer is designed and configured to have a predictable and tested response, including the option of ensuring any appropriate human-in-the-loop decision making.

Of particular importance for success, and a challenge to be overcome, is the standardization of information across the NDT models and the normalization of network data (ex. from the AT) presented to those models. One of the benefits of moving to a network digital twin approach to network autonomous operations is that it will be a nexus forcing function to that normalization. It is a systems engineering challenge that begins early in the SDLC and affects multiple operator systems and organizational silos.

The network digital twin approach will transform the way provider networks are designed and operated, but it will not arrive overnight and all at once. Organizational inertia of existing tools, techniques, skills, and processes across the SDLC will need to be overcome. Here, too, it will take years to fully realize the complete architecture described in this first section.

However, network digital twin implementation has begun. The next section provides a case study of how one group at Cox has embarked on the journey employing a network digital twin operations analytics model for the data center network.

3. Case Study: Network Digital Twin in Cox's Data Center Network

With the goal of promoting the long-term health of Data Center Network operations within Cox Communications, Inc. (CCI), the Data Center Network Engineering (DCNE) team has embraced Intent-Based Networking (IBN) as outlined in "Intent-Based Networking – Concepts and Definitions (draft-irtf-nmrg-ibn-concepts-definitions-02), A. Clemm, *et. al.*" and in "Intent-Based Networking – Concepts and Overview (draft-clemm-nmrg-dist-intent-03), A. Clemm, *et. al.*", as a guiding framework in the implementation of a holistic automation strategy within CCI data centers. These documents, taken together, do not constitute a detailed blueprint for IBN, rather they define IBN and the structural components necessary to implement an autonomic network at a conceptual level. While investigating the promise and feasibility of IBN, it became

readily apparent that implementing automated change would require a high degree of visibility into the configuration and operation of network devices.

3.1. Background and Discussion of IBN Strategy

In the summer of 2020, the CCI Data Center network team began work developing an Intent-Based Networking strategy that could be implemented in successive stages with each stage providing value on its own and subsequent stages building upon the value of previous stages. Six stages were identified:

1. Visibility
2. Validation
3. Change Automation
4. Event Management
5. Auto-Remediation
6. Autonomic Network Operation

Visibility – Raw data about the network needs to be accessible at a *human* level. Considering the massive amount of network data that can be collected on a tier-3 Internet service provider network, it is important to distill that data into a digestible source usable by both humans and machines. This data must include both device and topology data as well as statistical traffic data. The refinement and analysis of this data into a usable, published dataset is the goal of this stage.

Validation – Testing using the data collected during the visibility stage provides a basis to ensure that the network is operating according to designed intent. In this stage, suites of tests are built to validate configurations, topologies, traffic flows, etc. These tests can then be used to ensure that changes to the network do not violate the original design intent of the engineer.

Change Automation – The risk in performing automated changes to the network is less stressful with a high degree of visibility into the operation of the network and the ability to validate that those changes had the desired result and did not break any existing network functions. This stage is often built in parallel to the previous two with the goal of automating simple, repetitive tasks. As Visibility and Validation mature, Change Automation can be trusted with riskier changes. Part of this stage is the identification of component configurations (config snippets) used to implement specific functions, i.e., implementing an access control list (ACL) to control access to the routing engine.

Event Management – Leveraging Visibility and Validation, AI and Machine Learning can be employed to detect anomalies in expected behavior. Analysis of these anomalies leads to the identification of events, which can be investigated, and appropriate action taken through human interaction.

Auto-Remediation – As the Change Automation and Event Management stages mature, trust in allowing the network to heal itself grows. Once a specific course of action has proven to resolve

an event that commonly occurs, Change Automation can be used to automatically fix or handle those events.

Autonomic Network Operation – This final stage requires significant investment in the previous five stages to the point where AI and Machine Learning algorithms can review high-level intents provided by engineers, validate them against the operating network, review the component configuration libraries of the Change Management stage to make determinations regarding how the network should be configured to realize the intended design of the network and finally schedule and implement the changes identified.

With this strategy in mind, the Data Center Network Design team began developing use cases focused on developing the Visibility and Validation stages.

3.2. Definition of Use Cases

Like most traditional network teams, the Data Center Network team at CCI is staffed primarily with engineers trained mainly on networking and security disciplines with a smaller number of individuals that have cross-training in programming and automation. Staff are comfortable working in a device command line interface (CLI), but less so writing code or leveraging tools like Ansible to make inquiries or changes to devices. To encourage the adoption of a 'NetDevOps mindset', the DC Network team began investigating various tools to facilitate movement *away from the CLI*.

A set of use cases was identified, and various network tools were researched for their feasibility in meeting those uses cases, as well as the part they could play within the broader context of the team's IBN strategy.

3.2.1. Functional validation of the network intent.

As discussed regarding Visibility and Validation, a fundamental component of our IBN Strategy is the ability to collect network data and test it based on an expected outcome. While many might consider it sufficient to validate that device configurations were matching some pre-defined standard template, it is also important to validate those devices were operating as expected and were implementing the intent of the design. The latter validation capability required a considerably more sophisticated testing framework to enable validating the output of various commands.

Examples of intents identified for the proof-of-concept testing were:

1. Route Propagation
2. Expected Path Analysis
3. FWs blocking/passing traffic as expected
4. VLAN Propagation within the Data Center
5. EVPN Functionality
6. Physical link connectivity and bi-directional traffic validation

3.2.2. Network database – searchable codex of devices and linkages.

Another aspect of visibility that was researched is the ability to perform a search and find various conceptual entities and where they might exist within the network. The goal was to quickly locate and isolate things like:

1. Devices and Device Interfaces
2. MAC and IP Addresses
3. VLANs
4. Routes
5. ACLs
6. L3VPNs
7. Text within a device's configuration, such as a neighbor hostname in a physical interface description.

3.2.3. Reporting – Automated Querying of Network Database

A final identified use case is the ability to perform ad-hoc queries on network data for the purpose of reporting. This use case could in turn facilitate other use cases by giving users direct access to search device data, create reports for management, or use search results as inputs to some other process.

3.3. Decision to implement a PoC with a Specific Vendor

Based on research and the use cases identified, an application was selected for implementation in a POC environment. One of the key factors in this decision was the fact that the application chosen builds a mathematical model of the network based on the transformations made to a packet as it passes through the network. Because the data center network at CCI implements EVPN with L3VPNs, it can be confusing to troubleshoot packets from the CLI. The application gives CCI engineers the ability to visualize how complex network configurations affect the path an IP packet takes through the network on a hop-by-hop basis.

3.3.1. Discussion of Implementation

After reviewing the possible security issues with the CCI internal security team, it was decided to implement the application in a SaaS model. In this model, an internal *collector* is installed within the security perimeter of the organization with the ability to make SSH connections to all devices participating in the network model. The collector logs into each device, runs a battery of commands and collects the resultant data generated by those commands. It then bundles up the data and encrypts it for shipping to the cloud environment, which processes the data, builds a mathematical model of the network, and provides a searchable map through the cloud-based GUI.

Depending on the number of devices requiring collection, the sizing of resources dedicated to the collector may vary. Appropriate sizing for the collector is something that needs to be determined

as part of the implementation process. It is important to note that the function of the collector is solely as a means of gathering information, not processing it. If the 'on-prem' application model is used, resource requirements are significantly higher.

Management of the collector is handled through automation provided by the vendor and upgrades to the cloud environment are all managed internally by the vendor.

The ease with which the application can be managed within CCI was an important factor in the decision to eventually move forward with purchasing the fully licensed version of the application.

3.3.2. Discussion of adoption and usage

The CCI DCNE team manages network connectivity for multiple data centers with the organization. There are two primary data centers, one in Alpharetta, GA and one in Phoenix, AZ as well as a disaster recovery data center in the Metro Atlanta area. In addition, there are distributed regional data centers servicing edge applications in the various CCI markets. To quickly gain benefit from the application, maps of each data center were built to assess if there were possible issues at those locations by running basic checks to validate connectivity between devices. After this initial stage and discovery process, a full map of the network including the network backbone was created. This map allows for the analysis of path data between the various data centers and ensures that traffic follows the expected paths based on where it originates and where it is destined.

Adoption of the new application has not happened as quickly as hoped. The initial collection of data in the primary data center included many devices. Because of this, the first maps which were drawn by the application were highly complex and difficult to understand. Though the search features worked as expected and gave accurate results, the complicated nature of the maps discouraged some users from engaging with the tool. To counteract this complexity, the application allows devices to be aggregated into 'clusters' represented by a single node on the map. This has been effective at making the map more digestible but requires a significant investment in effort to curate the maps. Leveraging the application's API, automated tools for map curation are being developed.

3.3.2.1. Device selection and curation of map

As previously discussed, having an accurate map is vital to developing trust in the tool by network personnel. While it may seem obvious on the surface for highly complex networks, determining the set of devices to collect requires some forethought. While the process of collecting and mapping the network can provide significant insights, a haphazard approach to adding devices can lead to a map that requires significant effort to curate. Built within the tool itself are the capabilities to automatically curate connectivity between devices, but this requires LLDP/CDP on devices interfaces, something that for security reasons is not viable on all devices (such as FWs and some LBs). Consequently, manually linking those devices together can become a time-consuming part of the curation process.

A recommendation to make the process smoother is to start from a central point or 'core' network device and then expand the collection of devices towards the edge of the network. By doing this, devices can be added to the network in a manageable way and the location and placement on the map more easily be curated.

3.3.2.2. Usage/adoption among front line staff

A key aspect of adoption is the use of the application as a troubleshooting tool. The search capability allows users to do things like locate hosts within the network or search paths through the network to ensure that end-to-end traffic is following the expected path. During the information gathering phase of the troubleshooting process, being able to quickly collect this data can have a significant positive impact on MTTI (mean time to innocence) and MTTR (mean time to resolution).

While the application has shown significant promise, there has been some difficulty in getting some network staff to adopt a 'new' tool when they are used to using other tools or using the CLI. A four-session training course was presented to show the efficacy of the tool as a troubleshooting resource. Three sessions were presented by internal staff and a fourth was presented by the vendor. Sessions were recorded and made available for the broader network community at CCI. Although this did increase the user base slightly, it did not reach the expected goal for number of users in the allocated time frame.

While some staff have seen value right away and have begun adopting the tool, these staff tend to be higher-level engineers. We are seeking further engagement and feedback from the potential user community to increase future adoption.

3.3.2.3. Expected growth in adoption

Recognizing the need to integrate this new application into the typical troubleshooting workflow for front line network support, we have undertaken to analyze the typical use cases seen by operations staff. More training on how to solve specific troubleshooting problems will be developed. During this process, operations staff will be engaged to provide feedback to the development process.

Another point of attack in growing adoption is the development of network queries to provide value to users by using the application's query engine to gather data, generate reports, etc. Queries that provide detailed validation of proper device configuration, for example, proper MTU sizing on an interface, can be adopted as part of a workflow process to remediate devices that are out of standards compliance. While this may initially be implemented as a manual process, the application provides a facility to make testing an reporting an automated process on all collections.

3.4. Discussion of PoC Results

Based on the use cases described above, the POC was very successful. In fact, very early on in the POC, issues with the network topology requiring immediate attention to prevent possible outages were discovered.

3.4.1. Efficacy of network entity searches.

One of the most useful toolsets is the ability to search the network for various network elements, such as IP addresses, MAC addresses or VLANs. This search capability has proved very useful and provided great insight into how VLANs are distributed in the various data centers. After searches are entered valid results are shown both as tabular data and as graphical representations on the map. When searching for a VLAN, all points that the VLAN is distributed to are shown on the map.

3.4.2. Efficacy of network path searches

Probably the most powerful of all the tools in the application's arsenal is the network path search: given two IP addresses on the network a graphical representation of the path traffic takes between those points is given. That path information includes detailed information from each device in the flow, including how the packet changes from entry to exit as it moves in and out of the device. The detailed information provided is extremely useful because it allows us to view the relationship between underlay and overlay parts of the EVPN configuration.

3.4.3. Use of network queries to identify misconfigured network devices

Another useful tool of the application is the query engine. This tool allows us to generate queries against all the devices in a map, extract data from those queries, and run tests against that data. During the POC we used this facility to validate the configuration of MTU size on device interfaces. Manually validating the MTU size on thousands of device interfaces is a tedious and error prone process. By writing a properly formed query, we can ensure that all device interfaces that are administratively and operationally 'UP' have an MTU of a certain size or greater. This function works very quickly and gives us the information needed to hand off to our operations teams in csv-formatted report so that the incorrect interfaces can be remediated.

3.5. Realities of Digital Twin maintenance

As described in this case study, the digital twin shows great promise in helping validate the proper operation of the network based on the intent of the network design. There are, however, some considerations that affect the construction of a network digital twin as it is being created and managed.

3.5.1. Time requirements for staff to manage.

The initial time commitment for building a digital twin is very high but should taper off as the number of devices in the collection reaches the full set of devices in the network. This is especially true if accommodations can be made to device configurations that make automated mapping of the network easier, such as the implementation of low-level device discovery protocols is enabled, such as LLDP or CDP. While the application was able to infer connections between devices without direct LLDP/CDP data by analyzing MAC addresses and ARP tables on the device, inference of connections can only work if the application can collect the data necessary to make those inferences. In some cases, network configuration can prevent the collection of that data.

Once the topology of the network is reviewed, and any curation of the twin required is completed, the application will maintain an accurate assessment of the topology unless some change requires an update to the map. It is important to note that the time needed to manage map curation at this point relates to arranging the placement of devices/cluster nodes on the map in such a way that it is readily understandable by all network staff.

3.5.2. Assigning overall responsibility/technical advocacy for map(s)

It is important that a RACI matrix be developed for ownership and management of digital twins. Responsibility and accountability for the accuracy of each map must be vested in individuals who have an interest in the value provided by the map. In a highly complex data center environment, especially a service provider environment, it is likely there is no one person who has a full understanding of the entire data center network. Being able to ensure that the map is accurate is an effort that should be shared between design and operations staff and should enlist enough staff to ensure expertise throughout the environment.

An example of the need for this effort would be the case where a connection between two devices was not properly discovered, requiring manual configuration in the digital twin model. In this case, path searches may not resolve as expected, which would be an obvious indication of an issue to someone familiar with the expected path that traffic should take.

3.5.3. Cost/Benefit discussion.

While a full ROI analysis has not been performed on the application, there have already been several instances where the manpower required to analyze and remediate issues in the network discovered by the tool would have been orders of magnitude greater if done manually (if they would have been identified and fixed at all). It is important to note though, that fully realizing the value of the tool requires investment in manpower to customize the tool to the specific environment within which it is deployed, which is something that must be done on case-by-case basis. In addition, a fair accounting of the fully realized ROI would require a significant analysis in time savings for network staff.

4. Conclusion

We hope that the paper has provided a blended view of network digital twin, including both a forward-looking glimpse of the long-term landscape with some of the exciting transformative possibilities, and a practical account of how one such possibility is being realized today.

In the words of Lao-tzu, “A journey of a thousand miles must begin with a single step.” The introduction, development, and wielding of network digital twin is a journey worth taking. We welcome fellow travelers on this journey as we all take the first steps!

Abbreviations

ACL	access control list
AI	artificial intelligence
AT	Actual Twin
CCI	Cox Communications, Inc.
DCNE	CCI Data Center Network Engineering
FMEA	Failure Mode and Effects Analysis
HDTI	Human Digital Twin Interface
HMI	Human Machine Interface
IBN	Intent-Based Networking
ISP	Internet Service Provider
INCOSE	International Council on Systems Engineering
MAC	Media Access Control
ML	machine learning
NDT	Network Digital Twin
NDTE	Network Digital Twin Ecosystem
NDTh	Network Digital Thread
SCTE	Society of Cable Telecommunications Engineers
SDLC	System Development Life Cycle
SEBoK	Systems Engineering Body of Knowledge
VPN	Virtual private network

Bibliography & References

Intent-Based Networking – Concepts and Definitions (draft-irtf-nmrg-ibn-concepts-definitions-02), A. Clemm, *et. al.*; IRTF

Intent-Based Networking – Concepts and Overview (draft-clemm-nmrg-dist-intent-03), A. Clemm, *et. al.*; IRTF

Header Space Analysis: Static Checking for Networks, Kazemian, Peyman, G. Varghese and N. McKeown; *NSDI* (2012).

SEBoK: *Systems Engineering Body of Knowledge*; International Council on Systems Engineering

SEBoK/Digital Engineering: Article titled *Digital Engineering* in the *Systems Engineering Body of Knowledge*, 2021; International Council on Systems Engineering

The Way of Lao-tzu, 64, Lau-tzu, c. 604-c.531 B.C.E.; Bartlett's Familiar Quotations, 18th edition

