Assuring Trustworthiness in Dynamic Systems Using Digital Twins and Trust Vectors

A Digital Twin Consortium Foundational Paper

2022-10-25

Authors

Anto Budiardjo (Padi), Jon Geater (RKVST), Frederick Hirsch (Upham Security), Michael Pfeifer (TÜV SÜD), Detlev Richter (TÜV SÜD)

Contents

FIGURES

TABLES

Digital transformation promises to deliver cleaner, greener and more efficient technology solutions by bringing data-driven operations to the real world. Smart buildings that adapt to required usage, smart vehicles that optimize transit and smart machines that sense their environment and adapt to optimize output are all within reach.

Digital twins are an essential part of this transition, but they need to operate securely and safely and need an understandable and interoperable model for maintaining security and safety assurance that satisfies all stakeholders: technical, business and regulatory, if they are to be adopted at scale. Best practices for trustworthiness characteristics such as safety and security should be followed but more is needed.

In a complex software- and data-driven environment things can change rapidly. As a result of new vulnerabilities or a lack of maintenance, something that was once fit for purpose may no longer be. This raises a need for continuous assurance of meeting security and safety requirements while considering changes, even dynamic changes, in system composition or operating parameters.

Because cyber-physical systems have real-world consequences, all the trustworthiness characteristics (safety, security, privacy, resilience and reliability) must be considered holistically. Many initiatives and standards for security exist, but they do not consider all of these characteristics that can result in losses. Similarly, safety standards for mechanical systems are mature and respected but are not necessarily able to address all concerns with dynamic and complex software-based systems. To be successful using digital twins, operators must have visibility and control over all five trustworthiness characteristics.

The *trust vector* is a model for these critical decisions.

1 A New Approach

Today's safety and security landscape is largely *static* and *avoidance based*. Typically, it relies on a list of known things not to do and control measures needed to support safety and security based on a concrete understanding of exactly how the system is composed, how it has operated in the past and the static environment for which it is intended. This static approach is safe but inflexible, too inflexible to deal with the realities of today's software-based and highly connected systems. The price paid for relative certainty at the design stage is the inability to move to new operating models or adapt to new environmental conditions during the much longer operational stage. To communicate, devices needed explicit, design-time integration and special code adaptations to speak each other's protocols, making combinations static and favoring pre-existing relationships.

The advent of technologies that enable new, even *ad hoc* connections between systems provides more flexibility in system design and operation and gives access to more data from more sensors

that can be invaluable in making good safety decisions. This requires an approach for certifying dynamically composed systems that is different from the static 'proven-in-use' approach.

1.1 Dynamic Trust as Foundation for DT-Based Eco-Systems

Digital twins can provide a means to achieve trustworthiness for systems that require continuous decisions in changing situations. Time-critical assurance and decision-making needs a flexible system that is able to make decisions in real time to prevent vulnerable situations that would result from static measures, while reaching new productivity levels. Digital twins provide a means to achieving this.

According to the Digital Twin Consortium, a digital twin is defined as follows:

- A digital twin is a virtual representation of real-world entities and processes, synchronized at a specified frequency and fidelity.

The key phrase is "synchronized at a specified frequency and fidelity." Risk assessment involving threat and hazard analysis requires an understanding of the appropriate frequency and fidelity of synchronization as they are essential to ensure resilience of dynamic use cases.

Only when assessments of changes within the lifecycle have been built into digital-twin capabilities, making trustworthiness part of the digital twin architecture, is it possible to make changes to machines or configurations that are safe and that do not generate an unacceptably high downtime during validation. Digital twins can achieve a previously unattainable level of adaptability, flexibility and uptime in production environments.

To achieve a trustworthy and dynamic digital twin system you need to take the following steps.

1.2 Step 1: Enabling Communication with Connection Profiles

Interoperability as a primary concern: An essential consideration whenever systems communicate is that they be able to understand each other and act confidently and predictably on the information they exchange. The Digital Twin Consortium *system interoperability framework* tackles this topic in depth.

Connection profiles: A *connection profile*[1] (CP) is a named and immutable model of how two entities exchange information for a specifically defined purpose. Each CP defines the properties that each side of the connection provides when an instance of a profile is created to connect two specific systems. The two sides of a CP are referred to as client and server nodes, although information can flow in either direction.

[1] *https://github.com/CNSCP*

1.3 Step 2: Ensuring Data Provenance and Integrity with Zero Trust

Now that machines and software can communicate, we need to ensure they do so only under the right circumstances. Taking in good new data is a great opportunity for improvement; taking in bad data is an equally great opportunity for disaster. If "knowledge is power" then you need to be careful of your sources of knowledge. This is where zero-trust principles come in.

A *zero-trust* approach aims to *increase* trust in the system by *driving down toward zero* all the assumptions, shortcuts and blind spots that came with traditional network security approaches. It's not "trust but verify;" it's "always verify *then* trust."

The core principle of zero trust is to "assume breach." This means accepting the reality that nothing is 100% secure and sooner or later an attack will get through. Nothing is 100% reliable and sooner or later it will break down and need maintenance to return to reliable operation. If systems are built to verify their data and component relationships regularly, then they will be far more resilient and inspire confidence.

Traditional silo models of network security operated mainly on confidentiality and keeping data inside the castle. This can mitigate some attacks but does not address insider attacks for example. Such an approach can also make it harder to share data when it is useful to do so. Keeping data locked up in silos makes it unavailable for use in critical decision-making for other system components. A zero-trust approach enables data to flow and provides the tools to validate the integrity and usefulness of that data in the context where it is needed.

1.4 Step 3: Enabling Business Confidence through Continuous Assurance

Connection profiles and a zero-trust approach provide the basis for the next step: achieving continuous assurance. *Continuous assurance* is the practice of evaluating assurance cases in real time against the best, most up-to-date, operating and environmental information, rather than the traditional point-in-time practice of annual audits, approval of components and long-term confidence from static approvals.

Trustworthiness can change over time, or with a change in context. Something that was a reliable source of data in the past might not be trustworthy now. This could be for any number of reasons:

- *Cyber*: vulnerabilities may have emerged that were not known when the system was originally approved. For example, a supplier may have had a security breach.
- *Privacy*: appropriate measures have not been taken to protect data from inappropriate use, re-use and retention.
- *Compliance*: required maintenance may be outstanding, changing risk calculations.
- *Environmental*: the operating environment may have changed in a way that changes the assumptions of the original safety case. e.g. a unit designed and approved for indoor use may have moved.

- *Commercial*: business and legal changes may occur, increasing risks if certain expected actions are not taken.
- *Political*: data may be subject to political risks if situated in a foreign territory, or if a party that is part of the system may have been acquired by a foreign entity.

Any of these reasons can move a control or data source from trustworthy to untrustworthy.

Achieving trustworthiness may be complex. For example, the direct verification of trust in an Industry of Things (IoT) device may be to verify its digital identity cryptographically, while the indirect validation may be to check vulnerability reports or withdrawal notices against the device's software bill-of-materials. If this indirect step is not done, a compromised device with a digital certificate might be able to send secure, yet false messages to the rest of the system. This requires validation of all claims, using multiple sources and responsible entities.

Adopting zero trust means understanding enough of the detail of the system and its use to understand the necessary claims. "Which environment am I in right now? What action am I about to take? How much information do I need from how many sources before I'm confident that I won't make a bad decision? How can I believe those sources?" Checking enough, but not too much, is key to achieving a safe and secure outcome within practical boundaries.

1.5 Step 4: Implementing Trust Vectors for Resilient Dynamic Systems

Steps 1 ~ 3 enable communication, then implement technical trust so the data can be trusted, then implement business trust so the processes and sources can be trusted. The final piece is to put all that into action using techniques that can quantify the abstract notions of 'risk' and 'trust.'

Trust vectors are standardized connection profiles that convey the five trustworthiness characteristics: security, privacy, safety, reliability and resilience. Trust vectors allow two entities to exchange and negotiate scores of each of the characteristics on a range between a score of 1 (least trustworthy) to 5 (most trustworthy). The consumer side presents its requirement scores, while the provider side presents its ability scores. When the provider's score matches or is greater than the consumer side in all characteristics, the relationship of trust is established. Moreover, a trust vector allows both sides to provide, for each characteristic, a URL to a machine- and human-readable justification for their respective scores. This justification could be an assurance case, for example.

The trust-vector approach identifies which characteristics are necessary for collaboration of the digital twins to enable more trustworthy operation. Creating smarter systems using trust vectors enables better decisions, making systems more productive, not shutting down production unnecessarily. The trust-vector approach enables system components to communicate and answer the question: is this other component going to help me achieve my outcomes in a better, trustworthy way, or do they represent an unreasonable risk?

Trust vectors can be especially useful for cooperation and collaboration between different digital twins, using the trustworthiness level assessed in real time. The trust-vector principle enables a fast negotiation procedure, using similar principles to the following:

> *Assume somebody is planning a business trip to a customer and is looking for a hotel. The search for a nice hotel can be done based on the price, luxury level of the hotel, parking spaces, distance to the customer, reachability with means of transport, surroundings like restaurants and parks. This research and decision-making process can consume a lot of time, especially when the chosen hotel is not vacant. If we reduce our research and decision making on the distance to the customer and the hotel's rating (e.g. 3-star hotel) we will find a vacant one faster. This increase of speed was possible by reducing the number of parameters to be assessed.*

The same approach can be found behind the idea of the trust vector to enable time critical decisions in digital twin systems.

The steps outlined lead to *dynamic* and *outcome-based* approaches to trustworthiness with the flexibility to enable new business cases and the resilience to handle unanticipated events. Enhanced communications channels with connection profiles, securing those channels through zero-trust techniques, and assuring safe and mature operations through standardized and formally verified trust vectors, enables safe and resilient operations in digital twins.

2 TRUST-VECTOR SCORES

The definition of the trust-vector score targets and the capability of the digital twin to assess and change these values over the technical lifecycle are key skills assuring trustworthiness. The meanings of trust-vector scores must be established in context with an understanding of the vertical industry, concerns and details of the situation.

The five dimensions of the vector are based on the trustworthiness characteristics developed by the Industry IoT Consortium:

- *Security*: the system being protected from unintended or unauthorized access, change or destruction.
- *Privacy*: the right of an individual or group to control or influence what information related to them may be collected, processed and stored, and by whom, and to whom that information may be disclosed.
- *Safety*: the system operating without causing unacceptable risk of physical injury or damage to the health of people, either directly or indirectly, as a result of damage to property or to the environment.
- *Reliability*: the ability of a system or component to perform its required functions under stated conditions for a specified period of time.

- *Resilience*: the emergent property of a system that behaves in a manner to avoid, absorb and manage dynamic adversarial conditions while completing the assigned missions, and reconstitute the operational capabilities after causalities.

A maturity model can be useful in understanding the characteristics that comprise a score and can be used for establishing targets and assessing values. The IIC IoT Security Maturity Model (SMM)[2], for example, organizes the complex security space into eighteen practices covering governance, security enablement and hardening with guidance regarding four comprehensiveness levels for each, as well as a process for applying the model. The trust vector security score can be derived from the comprehensiveness levels of all the practices in the SMM. The SMM Digital Twin Profile[3] offers detailed guidance relevant to digital twins. The SMM 62443 mapping for Asset Owners, Product Suppliers and Service Providers[4] maps IEC 62443 requirements to the security maturity levels. All of this can be applied (as well as related work such as the NIST Cybersecurity Framework[5]).

Similarly, privacy models may provide a basis for privacy scores. An example is the MITRE Privacy Maturity Model.[6] Regulations such as GDPR also offer relevant guidance. Safety scoring standards such as ISO 13849 for electric and electronic components[7] may also be used as well as other best practices addressing mechanical safety aspects.

Reliability may be scored using the mathematics of fault modeling. Resilience scoring is more difficult since it relates to adaptability and learning, but organizational measures may be used.

The scoring approach of trust vectors can be adapted by industry verticals as appropriate.

Automatic down and upgrade of the trust-vector values using digital-twin-based capabilities during operations combines the results of “tested and certified after installation” with data analytics at run time for reliability figures, changes of recipes, hazards between assets moving around and so on into one vector: the trust vector.

The different trust vector trustworthiness levels can be evaluated into a risk/outcome of interacting with the other party. A low score does not mean you should not use it, but it does indicate that you need to take more direct responsibility to protect against risks arising from bad data or bad interactions. The generic meaning of the trust vector scores is as follows:

[2] *https://www.*iiconsortium*.org/pdf/IoT_SMM_Practitioner_Guide_2020-05-05.pdf*

[3]*https://www.digitaltwinconsortium.org/wp-content/uploads/sites/3/2022/06/SMM-Digital-Twin-Profile-2022-06-20.pdf*

[4]*https://www.iiconsortium.org/wp-content/uploads/sites/2/2022/08/SMM-62443-Asset-Owner-Product-Supplier-Service-Provider-Mappings-2022-08-16.pdf*

[5] *https://www.nist.gov/cyberframework*

[6] *https://www.mitre.org/sites/default/files/2021-11/pr-19-3384-privacy-maturity-model.pdf*

[7] *https://www.iso.org/standard/69883.html*

- Level 1: connectivity probably makes your life worse, at least if this is your only source of data.
- Level 2: you can connect and use data for operations, but you need all your manual processes in place to verify it before trusting anything sensitive in the data.
- Level 3: you can automate processes confidently based on data and virtual models to achieve known, static outcomes production assets are interoperable.
- Level 4: you can automate processes confidently based on data and virtual models to achieve known, static outcomes, production assets are interoperable and capable of run-time communication.
- Level 5: you can confidently automate and dynamically adapt processes based on data and virtual models to achieve optimized dynamic outcomes.

The following sections go into more detail regarding connection profiles and dynamic relationships, followed by examples related to the manufacturing industry vertical.

3 Introduction to the Trust Vector

3.1 Trust Vector Connection Profile–Rules and Capabilities

A digital twin system-of-systems has trustworthiness requirements for integration of DT-based assets. The trust vector that is appropriate to the capabilities needed and the trustworthiness maturity required by the situation ensure the required level of trust. In this case, the trust vector and the profile describe the capabilities of the digital twin incorporating the level of trust needed and the way to achieve it at run time. The figure shows that basic information for communication, information about the functionality and trust vector requirements for trustworthiness is needed.

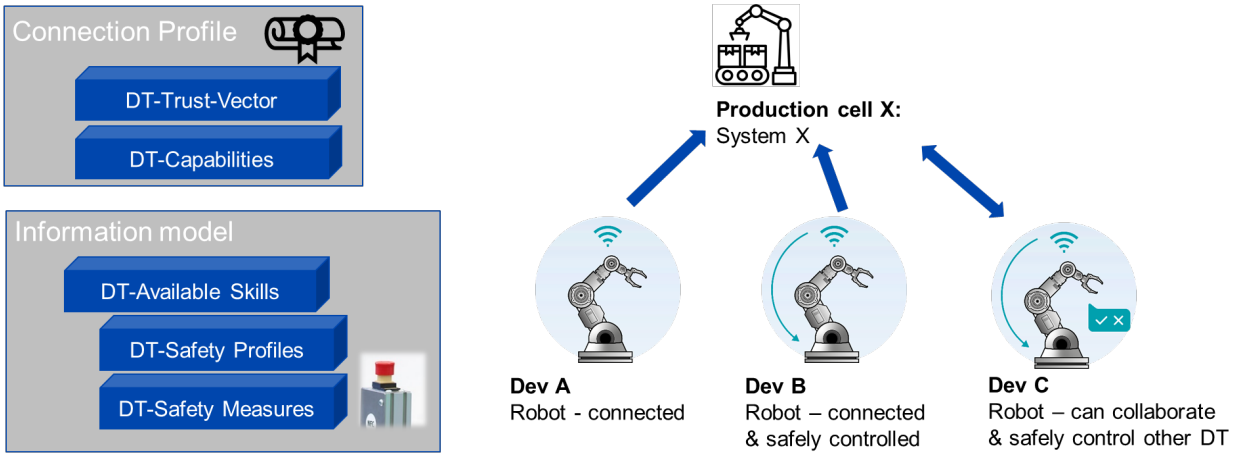


Figure 3-1: Types of information needed for communication.

A significant benefit of trust vectors is that they can be used without having completed a detailed examination of the reasoning behind the trust-vector values. Introspection is possible but not essential. Once values are established, they can be used easily and quickly in a variety of situations, with the knowledge that the values will be independently updated as needed. The trust vector enables the possibility of mission-critical decisions at run time in an efficient use of resources of computing and communication capacity.

3.2 TRUST VECTOR INTERMEDIARIES

The production world generates requirements that system X has to fulfill. Therefore, system X has to determine which subsystems it can rely upon to achieve its objectives while also meeting the trustworthiness requirements. For this reason, it has to monitor and determine which assets (A, B and C) fulfill the trustworthiness requirements.

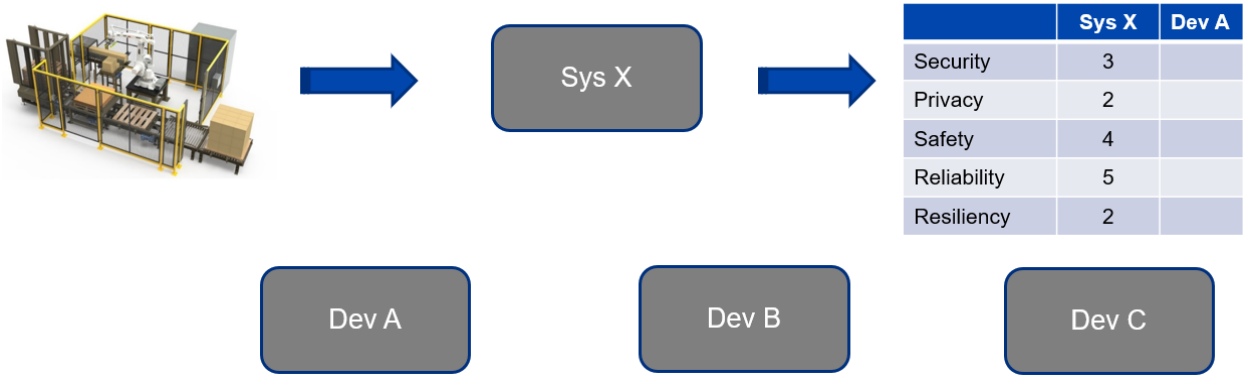
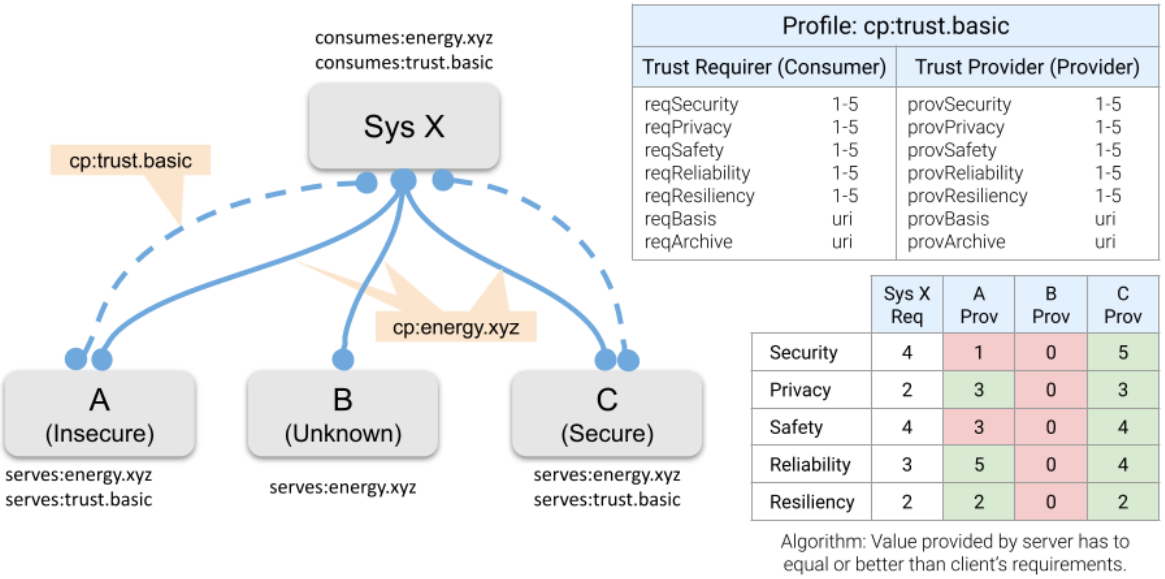


Figure 3-2: A system of system with a trust vector intermediary.

Connection profiles and trust vectors enable a flexible way to achieve the required level of trust by incorporating, at real-time, the capabilities of the assets that are necessary to fulfill the operational requirements for the operation to be executed. This shows that a system such as system X, acting as a client, requires certain attributes from other sources (server).



Profile: cp:trust.basic			
Trust Requirer (Consumer)		Trust Provider (Provider)	
reqSecurity	1-5	provSecurity	1-5
reqPrivacy	1-5	provPrivacy	1-5
reqSafety	1-5	provSafety	1-5
reqReliability	1-5	provReliability	1-5
reqResiliency	1-5	provResiliency	1-5
reqBasis	uri	provBasis	uri
reqArchive	uri	provArchive	uri

	Sys X Req	A Prov	B Prov	C Prov
Security	4	1	0	5
Privacy	2	3	0	3
Safety	4	3	0	4
Reliability	3	5	0	4
Resiliency	2	2	0	2

Algorithm: Value provided by server has to equal or better than client's requirements.

Figure 3-3: Basic trustworthiness using connection profiles and trust vectors.

How does the trust vector look like in terms of digital twins? The trust vector is a function of:

- aspects of a specific situation that are relevant for the asset to which the viewed trust vector belongs to and
- the asset's capabilities needed for this specific situation.

As long as all relevant situational information is available, the domain is not an important factor. For example, a robot does not differentiate between domains. They care about specific parameters that are dependent of specific situations, for example, allowed speed limits.

As another example, no one should be able to rent a truck based on their motorcycle driving license.

The specific trust vector score is in most cases domain specific, because of liability reasons.

4 Connection Profiles and the Trust Relationship Pyramid

Relationships between constituent systems in systems-of-systems are inherently multi-faceted. For such relationships to be modeled, they have to be composable to handle different types of information to be exchanged between them in different use cases, different circumstances and constantly changing dynamic systems.

A person can carry multiple credential documents such as a driving license, passport or pilot license. If the police were to stop them on the highway, they present their driving license; at a flight school, they show their pilot license. These are all "standard" and accepted forms of information about the person in specific contexts, some more general than others.

Connection profiles are models that define how a capability can be represented, so *cp:trust.xyz* is all about trust, and *cp:energy.xyz* is all about energy, etc. Each connection profile has two ends with complementary roles; one side is the provider of the capability, and the other side is the consumer of the capability. Each connection profile describes these roles in their own terms, so for *cp:trust.basic*, they are "requirer" and "provider" of trust scores. For *cp:energy.xyz* they could be "generator" and "load" of energy. We use the terms "provider" and "consumer" as generic terms for the two roles in all connection profiles. A connection profile broker instantiates a connection between systems that need to share information within a defined context by using connection profiles.

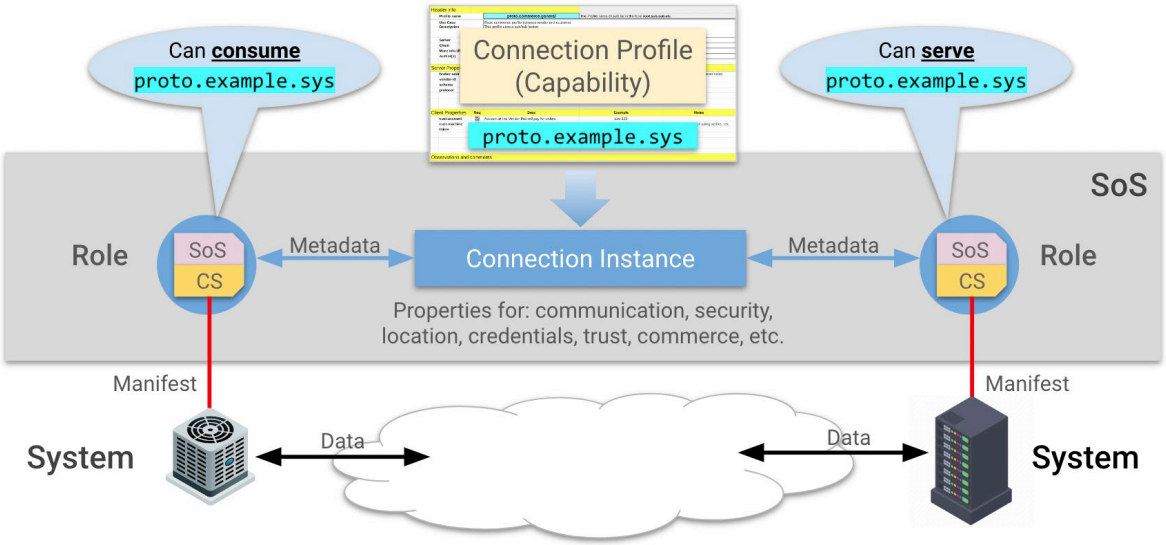


Figure 4-1: Operation diagram of connection profile mechanism.

Systems declare which connection profiles they support either as a provider of information (server) or as a consumer of information (client) to the broker in a manifest, and they provide the context and any necessary properties as specified in each CP model. The broker is responsible for matching compatible roles in a specific context, where the parties share a common set of circumstances and profile, and for passing the property values between matched systems.

Like the documents humans use, each CP is designed for a specific purpose, some more general than others. An issue like trust is general since, in a well-designed system, every system should have a way to understand trust in other systems regardless of function, just as driving licenses provide a way to verify that a person can drive. A relationship between supply-chain entities, on the other hand, would be concerned about package movements and quite different from the relationship between an electric motor and its energy provider.

5 System Metadata

To understand the relationship between systems, we need to understand the types of metadata that each system has that may be relevant in relationships with other systems. This metadata will

be used by servers that advertise their capabilities and clients that seek capabilities compatible with their needs. We can think of these types of metadata as a layered pyramid.

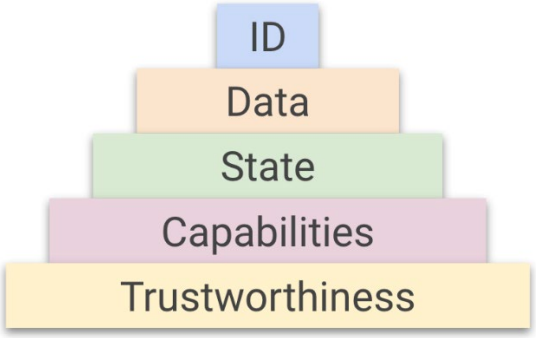


Figure 5-1: The types of system metadata.

With the exception of the ID layer, the metadata types are dynamic in nature and are assumed to change over time as circumstances change.

The types of metadata in the pyramid range from the specific (ID) to the most general (e.g. trustworthiness). Each layer is an aspect of the relationship a system may have with other systems that have a similar set of system metadata necessary to be in such a relationship.

ID—Identity of the entities: The ID layer provides instance-specific information about the unique ID of a system. If there is a relationship between two systems, then the identities of both need to be consistent and known to each other within the same namespace. On the internet, these IDs could be URIs or GUIDs, while in a closed system, these IDs could be locally created IDs unique within that system.

All systems within a Systems of Systems (SoS) must exist in a compatible ID namespace so that they can establish unique relationships with each other.

Data—Application-specific data: Application data will be needed to pass between systems for the successful operation of the SoS of which they are a part. The specific nature of this information, with respect to issues like bandwidth, latency tolerance, protocols, semantics and security, is highly dependent on the type of application.

Application data can be conveyed using the properties of CPs or they can be conveyed using other communication methods. When this information flow does not use CPs, the configuration of such a communication path can be established using the properties of appropriately designed CPs.

State—Instance-specific information of the entities: State metadata is unique to the instance of each system. Examples include availability, location, serial number and application parameters such as speed or temperature setpoint. In relationships, the state of a system is necessary for other systems to know and can be presented as properties of specific CPs.

An example electric motor presenting *cp:energy.demand* property may include a property *cp:energy.load* set to 1,000 watts in a specific installation of the motor. The connected electric meter knows the required load of the motor using *cp:energy.demand*.

Capabilities—Inherent capabilities of the system: Relationships between systems are based on each system declaring their roles and capabilities that can be matched by the broker with the compatible roles and capabilities of other systems within the context of the SoS.

Capabilities are presented using different CPs as per the needs of the SoS. An electric motor consuming energy may present a *cp:energy.demand* in the role of a client, while an energy supply entity may present a *cp:energy.demand* as a server. The motor and supply system would be matched by the broker, and the two entities would exchange information as defined in the specification of *cp:energy.demand*.

Systems can have multiple roles and capabilities by declaring any number of CPs, each with their role as client or server. This enables a system to have multiple relationships with other systems with matching and compatible CPs within an SoS and context.

In trust vectors, the two ends of the connection are defined as the requirer and provider of trustworthiness. Entity A could require scores of 3:4:4:4:3 in the five characteristics, so when matched with entity B providing scores of 4:4:4:5:5, it will result in the relationship being trustworthy since B's scores are all equal to or greater than the required scores. If a requirer is not interested in a specific trust dimension, best practice is to require a score of 1.

This layer of the metadata can use the standard connection profile called *cp:trust.basic*. Other trust-related CPs (such as *cp:trust.nuclear* and *cp:trust.military*) could be created for more specific and demanding trustworthiness requirements. These CPs could contain additional or different dimensions and scores than those specified in *cp:trust.basic*.

Other domain-specific trustworthiness connection profiles can also be exchanged in this layer.

6 RELATIONSHIP PYRAMID

When two systems need to interact with each other, the broker, responsible for orchestrating relationships within an SoS, would establish connection instances between them in accordance with the compatible and matching CPs, roles and context declared by each side. To have a useful interaction two systems need to have an appropriate trust relationship, have capabilities that are provided and needed for a useful interaction, be in a compatible state to interact (e.g. both are operational) and share compatible information models.

The resulting relationship will be based on two system metadata pyramids constructed from multiple connection profiles as shown between constituent system X and constituent system Y.

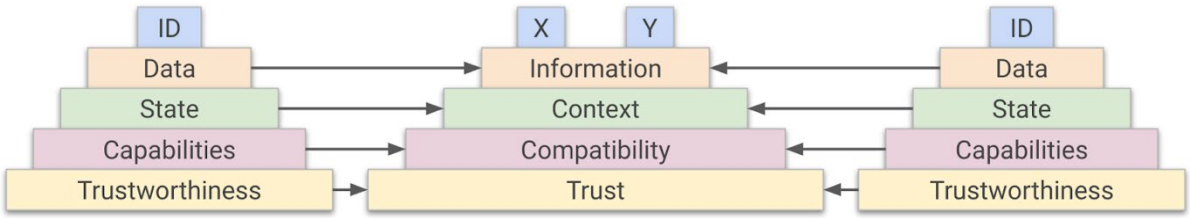


Figure 6-1: Combining two system metadata to make a relationship.

This diagram shows the metadata needed to form a relationship at each end, with the middle part of the diagram showing the relationship-kind corresponding to a type of metadata. For example, the information layer of the relationship pyramid represents the syntax and semantics of data understood by both parties, as defined by the capability layer. A trusted relationship depends on the trustworthiness metadata, the compatibility of systems depends on the capability attributes, a shared context depends on a consistent state and information depends on compatible data.

Each system's metadata pyramid is shown above on the left and right of the diagram. When these two constituent systems are matched into a relationship within a system-of-systems, each layer of the metadata must be compatible to interact successfully.

The combination of the two trustworthiness metadata of X and Y form the trust of that relationship. The combination of the capabilities of X and Y, when matched by a broker, would create compatibility of the two systems to provide a relationship. The combination of the states of X and Y create a specific context where such two systems can provide useful information in a relationship. Lastly, the combination of data in X and Y creates useful information being conveyed in the relationship.

The following terms define properties of the relationship (e.g. trust) and these relate to the properties of the entities (e.g. trustworthiness).

System IDs: In a relationship between two systems, there are two distinct IDs that exist to represent each system. These IDs, and the relationship they create, is managed by the orchestration function.

A combination of two system IDs is inherently unique as there can only be a single relationship between two given systems that considers all relationships between the systems despite the possibility of various communication protocols between them. This is different from the identities that might be used for different protocols used between systems. It is essential for parties to be clear with whom they are communicating, both from a protocol perspective and the context (i.e. a clear understanding of the properties related to that communication). This is essential for dynamic systems, especially when the context can change. Then they can meet regulatory and other requirements.

Information (data exchange): The information layer of the relationship pyramid represents the syntax and semantics of data understood by both parties, as defined by the capability layer. This enables a flow of data between the two systems in accordance with the trust and other attributes defined between each system, the matching context, role and capabilities of the systems.

The flow of information at this layer represents the actual, application and instance-specific value of the relationship between the systems.

Context (state alignment): The context is the set of constraints that make it meaningful for systems to operate together. The context of a relationship is the matching combination of the states of each of the two systems in the relationship. For example, this context can be consistent physical location, consistent operating procedures or consistent operating states.

For two systems to have a useful relationship, the relationship must be within a common context applicable to both systems.

When two systems have a relationship on the context layer, the two systems have a reason to exchange data related to the context. We can think of context as a physical entity, such as manufacturing plant XYZ, or a specific piece of equipment, or it could be a virtual entity such as an invoice or a maintenance ticket.

Compatibility (capability alignment): In relationships, the compatibility layer represents the matching of the capability and role of the two systems in the relationship.

When there is compatibility between systems, the two systems have a functional reason to have a relationship.

Trust (trustworthiness alignment): Relationships are built on trust. The same applies to constituent systems within an SoS. The trustworthiness metadata shared between two systems can be viewed as the metadata representing the trust relation between those entities.

In static use cases, the relationship metadata could simply be the comparison of the *cp:trust.basic* scores between 1 ~ 5 in the five characteristics. The trust is either there or not, using simple arithmetic based on the trustworthiness information.

In dynamic use cases, again using the *cp:trust.basic*, the trust metadata can be used as a way to negotiate the relationship constantly as circumstances change.

7 DYNAMIC RELATIONSHIPS

Composable relationships in system-of-systems provide a flexible way to enable dynamic behavior as required by the changing circumstances of the SoS, its constituents and the environment. The ability to combine relationships dynamically allow the needs of applications to be met and does not require consideration of a very large number of combinations in advance.

7.1 COMPOSABLE CONNECTIONS

The following example illustrates the composability and dynamics of the relationship approach to managing system-of-systems using connection profiles.

Consider a system-of-systems environment such as a factory or home depicted in Figure 7-1 as the grey oval. System 1 is instantiated with a manifest of four connection profiles shown as A, B, C, and D. System 1 declares each CP as having either a server role of a CP capability (shown as squares) or as having a client role of a CP capability (shown as circles). The depiction of *cp:a.x/s* is shorthand for declaring the capability of connection profile named a.x as a server (the /s means server, while /c means client).

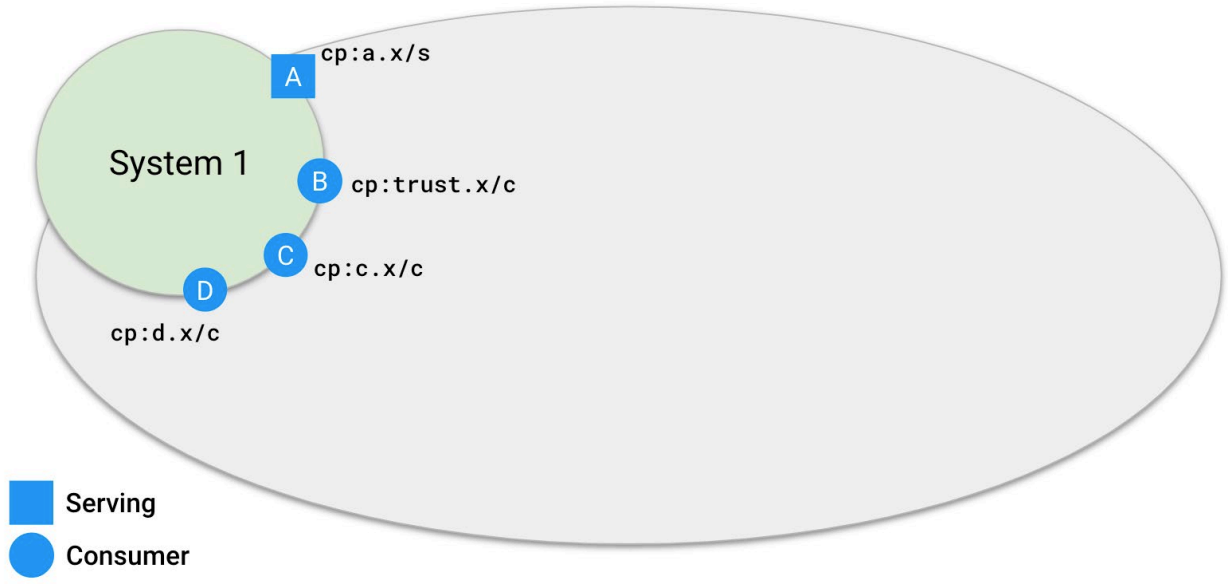


Figure 7-1: System 1 instantiated in the system of systems.

Since System 1 is currently the only system, no connections are created by the broker since no matches are possible.

Note that System 1 declares a client of *cp:trust.x/c* (shown as B) to share its trust vector capabilities with other systems. System 1 can therefore be considered as a system that includes functionality that uses trust-vector information to regulate domain-specific connections with other systems.

At some later time, System 2 instantiates itself as seen in Figure 7-2. The broker discovers a match since System 1 has declared as a server of *cp:a.x*, and System 2 has declared itself as a client of *cp:a.x*. A connection instance is created between the two systems shown as the line between them, and data can now flow between them as specified by *cp:a.x*.

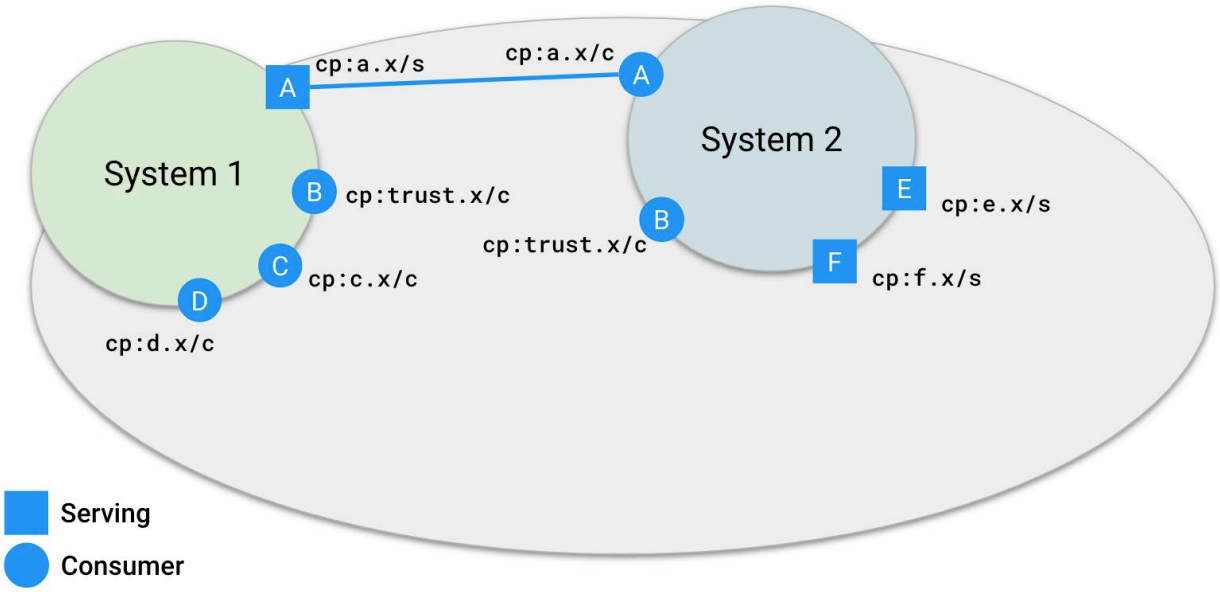


Figure 7-2: System 2 is instantiated, matched and connected to System 1.

At this point, Systems 1 and 2 could be considered to have a partial relationship as described by the pyramids above. Their respective IDs are known and being used (let's say 1 & 2), their capabilities are matched (using *cp:a.x*), their context is established (the system of systems of the grey oval), and data can flow between them based on the matched capability of *cp:a.x*, but there is no trust information between them.

In some use cases, where trust is less important, such as when using publicly available information, or where trust can be assured through other means, this partial relationship may be satisfactory, and useful data can flow between 1 and 2. Where greater trust assurance is needed, we can consider two approaches:

- Systems 1 and 2 could have established a *cp:trust.basic* connection directly between them, where System 2 provides its requirements of the five characteristics, and System 1 provides its scores accordingly. This method works well in use cases that are modest in size, but as the number of systems increase the number of point-to-point trust connections would become prohibitive.
- Alternatively, we can consider adding a trust intermediary that could be used by all systems in the system of systems to receive, manage, and provide specific trust scores of any system to any other system. Assuming the connection profile *cp:trust.x* as a way for each system to be part of such a trust management framework, each system would consume the capability of the trust intermediary as *cp:trust.x/c*, and the trust intermediary would be a trust intermediary by declaring *cp:trust.x/s*.

Introducing System X into the system of systems as an intermediary, there are two cases:

- If System X is a temporary system, such as an automated vehicle entering the system of systems to perform a task safely, it would declare *cp:trust.x/s* so that when the vehicle arrives, it will be matched with Systems 1 and 2, and once its task is complete, those connections would be removed.
- Alternatively, if System X is a permanent management system such as a Supervisory Control and Data Acquisition (SCADA) manufacturing/enterprise system or home automation application/device, then it could have multiple services that can be offered via CPs, including that of a trust intermediary. In this case, System X would declare *cp:trust.x/s* and other CPs as needed for their respective functions.

In either of the above cases, *cp:trust.x* would need to be defined accordingly as a variation of the *cp:trust.basic* principles to convey trust vector information. Other creative patterns could be envisaged for other use cases.

As seen in Figure 7-3, System X is instantiated and declares the server capability of a *cp:trust.x/s* as a trust intermediary (shown as square B), which the broker connects with both System 1 and System 2 to establish trust vectors between all of the systems in the system-of-systems.

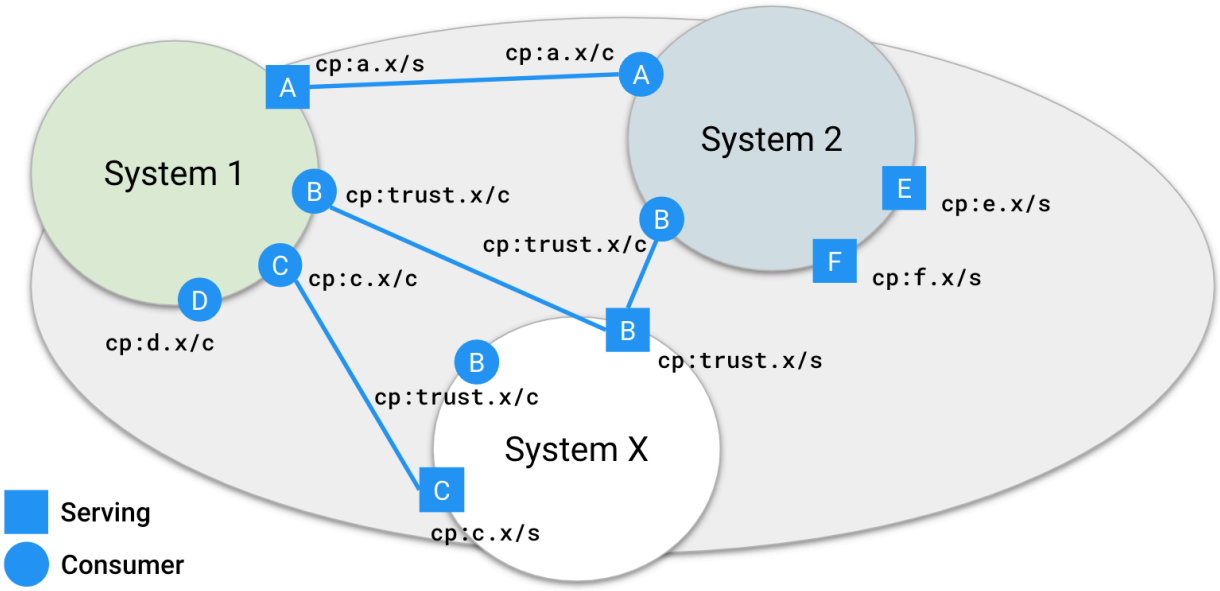


Figure 7-3: System X instantiated with a trust server connected to Systems 1 & 2.

System 1 and System 2 now have trust information about each other. With this, the server end of the A Connection (System 1) can now dynamically change what information it is providing to System 2 based on the newly discovered trust vector information.

Also, note that System X establishes a connection with System 1 using *cp:c.x/s* (shown as C). This connection can use the trust already established.

Note also that System X declares a *cp:trust.x/c* client to provide trust data if necessary, possibly to a different context (not shown).

Another scenario is if System X is removed, possibly due to a problem or failure. In this case, the trust information that Systems 1 and 2 are using would cease to exist, plunging the connection between Systems 1 and 2 to revert to an untrusted connection.

The behavior continues with other systems instantiating into the SoS. System N, in Figure 7-4, is shown to provide *cp:trust.x/c* so that it can communicate with other systems based on its trust vector.

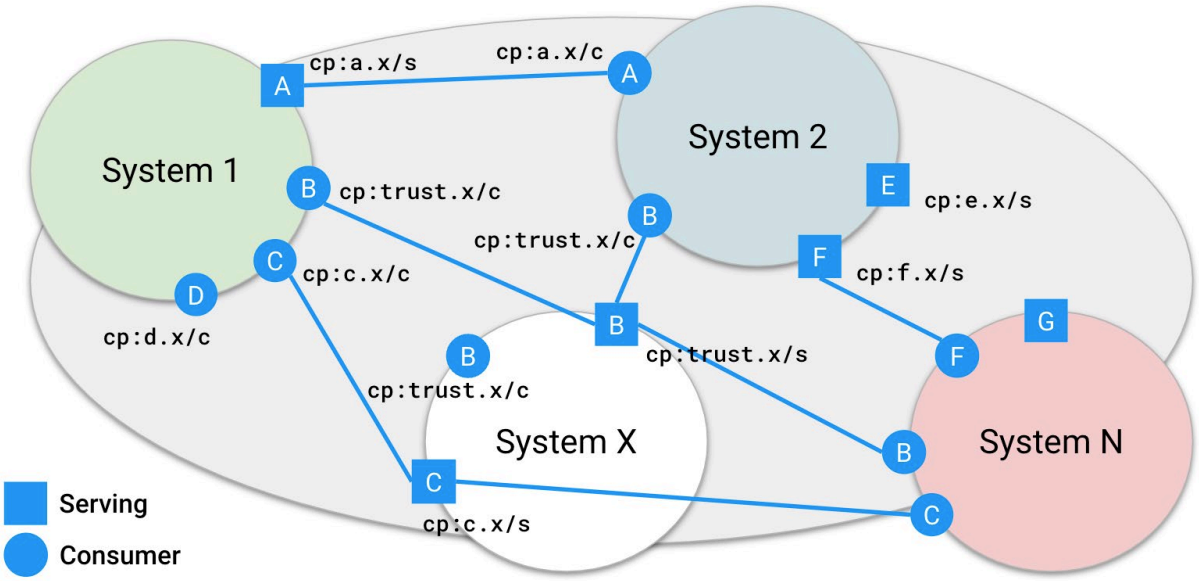


Figure 7-4: System N instantiated showing additional connections.

A connection instance based on F (*cp:f.x*) is established between System 2 (*cp:f.x/s*) and System N (*cp:f.x/c*) again based on trust vectors as appropriate.

Composability by chaining capabilities based on trust is also possible in this example. Note that System 1 does not have a direct relationship with System N but has an indirect relationship using CP *cp:c.x* via System X. As noted above, if System X is removed, possibly due to a problem or failure, then this indirect relationship between System 1 and System N would no longer be there.

7.2 Multiple Contexts

In the example described above, all the systems exist in a single system of systems in which any systems instantiating connection profiles will be considered to be matched with other systems by the broker. In connection profile terminology, this is considered as a single context.

Multiple contexts would separate the different areas of concern into their own contexts that could be overlapping or have some other association with each other.

Consider an environment such as a factory or home with two distinct physical spaces. Here, those two spaces could be considered as two different contexts necessary to invoke the matching capability of connection profiles to only those systems within each context.

Because those two contexts exist separately, some systems may pertain to both contexts, for example, the electricity supply, or a management system for the factory automation or smart home system. Other systems, however, pertain only to a single context, such as equipment located in a specific area of the factory floor, or a smart device in a specific room of the home.

The following example, in Figure 7-5, places the systems from the previous single context example into multiple contexts. Systems are instantiated in either both contexts (System 2 and System X) or just in one context (System 1 and System N). The contexts are labeled as Context 1 and Context 2.

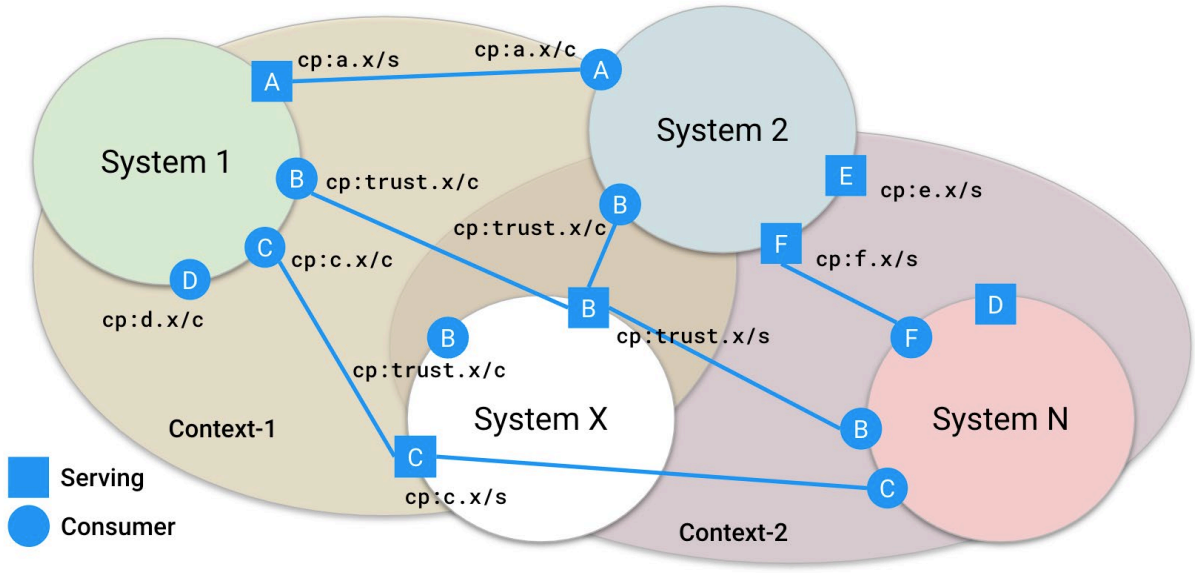


Figure 7-5: Example of a multiple context environment.

Note that System 1's *cp:d.x/c* is not matched with System N despite having a matching CP declaration since they are in different contexts.

The combination of context (defined by specific use cases), capabilities (provided by domain specific CPs such as *cp:a.x* and *cp:c.x*), trust specific CPs (as exampled by *cp:trust.x*) provides for a scalable, dynamic, composable and trustworthy flow of information between systems that have a need to do so.

8 Reliable Collaboration of Digital Twins

This section illustrates three scenarios focusing on the trustworthiness characteristic of "safety." The interaction between digital twins forming a digital twin system will be described for both a stationary and a moving asset: a robot. The system behavior under changed environmental conditions is described in the third scenario where a new context affects the system capabilities.

8.1 Scenario 1: Using a Digital Twin to Manage Risk for a Stationary Asset

We start with the typical factory application of predictive analytics using a stationary asset analyzing its own operation (work piece and recipe) and add DT-based communication (connection-profile based) to the digital-twin based environment of the turning machine.

Scenario 1: stationary asset (DT) and handling of residual risk (DT-profile) within a DT-based environment.

A stationary asset and the environment will see changes over its lifetime. Every asset comes with a risk assessment made for an intended use of this machine. There is no absolute safety; all risks are mitigated to a risk-acceptance level. Therefore, every machine has remaining residual risks, which are sometimes forgotten and lead to accidents, some of them serious. Digital twins enable handling these risks based on communication profiles that incorporate trust-vector scores.

Figure 8-1: Turning machine accident – workpiece knocked out the protective door (source: TÜV SÜD).

Scenario 1 comprises a turning machine (DT 1.1), a work piece (DT 3.1) and people passing this machine e.g. a worker (DT 2.1).

How can a DT-based factory handle the situation for different types of people and the “residual risk?”

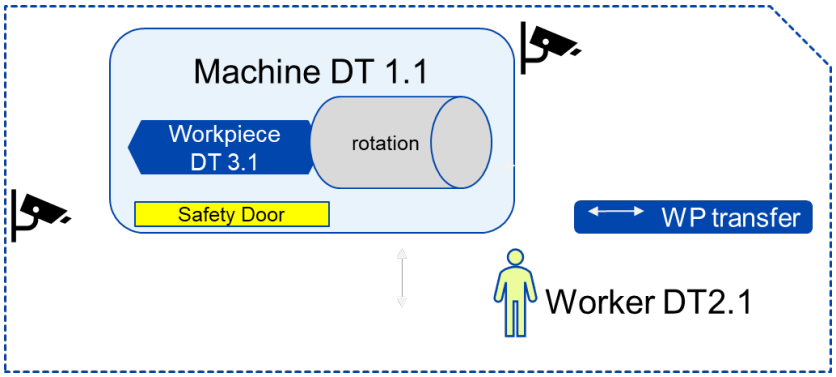


Figure 8-2: Scenario 1 showing turning machine, work piece and worker.

According to machine standard *ISO 23125 - Machine tools - Safety – Turning machines*, safety doors are allowed to have a limited mechanical resistance, at times significantly lower than the maximum workpiece load capacity of the machine. Current I3.0-practice is to include a notice in the instruction manual informing the operators that "The safety door minimizes the risk of ejection, but it does not eliminate it entirely." However, relying on the safety manual may be less effective in practice, especially over time.

The digital twin-based turning machine understands that the weight of the working piece is higher than the protection level of the protective door. In this case, the residual risk has to be considered if people are in front of the machine. The trust-vector level to the environment of the machine has to be checked and fulfilled for a safe operation addressing the residual risk. As part of the zero-trust approach, the machine does not simply depend on the configured weight of the working piece but confirms this understanding by measuring the resistance to turning and thus knows that the weight is in the appropriate range. Zero trust does not require new sensors or costs, but it does need an understanding of the approach and better software. Using this knowledge in conjunction with sensing that people are near the machine (e.g. by video analysis), safety can be significantly improved without unnecessarily stopping production.

What kind of DT-based communication starts if a visitor is crossing a factory floor and stops in front of the machine?

A DT-based turning machine with trust level 3 and 4 has the capability to handle the situation within the context. The context understanding can be achieved by communication to all DT-based systems and people within this factory cell. The required decisions are derived from the safety maturity level specified, for example, worker classification. The unacceptable risk level for visitors is automatically detected and properly reduced by using trust-vector values only.

Sys Req	Turning Machine	People		Action
Safety Maturity Level 3	Safety Maturity Level 4	Level 5	Trained safety expert	Safety expert is aware about the situation, its hazards and risks; this person can execute safety-related decisions and communicate these with the system, e.g. for maintenance or troubleshooting reasons.
		Level 4	Trained operator for this machine	Operator is aware and has been trained appropriately, no actions.
		Level 3	Operator from our company	Inform about residual risk: **Action:** Fly-path warning.
		Level 2	Unknown person; can communicate	Inform about residual risk: **Action:** Fly-path warning, if unknown person enters danger zone, slow spindle down.
		Level 1	Unknown person	**Action:** Slow spindle down, inform factory operator to handle the situation.

Table 8-1: Safety Trust Vector Options for Scenario 1

The DT-based turning machine with trust level 4 can handle the situation within the context correctly since the trust vector indicates it is acceptable with the underlying reason that the operator is trained. The unacceptable risk level for a visitor is automatically detected and properly reduced by slowing down the spindle. A visitor is treated differently from an operator due to liability, for example. In this example, the trust level 4 incorporates a kind of broker or agent, which has the capability to derive a decision. The broker or agent could also be part of the digital twin of the factory cell, for example.

Example for the levels and the reference to the corresponding maturity model developed by TÜV SÜD:

- Level 1: no-safety or unknown.
- Level 2: safety level 2 according to safety maturity model—communication supported.
- Level 3: safety level 3 according to safety maturity model—cooperation supported.

- Level 4: safety level 4 according to safety maturity model—coordination supported.
- Level 5: safety level 5 according to safety maturity model—collaboration supported.

8.2 Scenario 2: Using a Digital Twin and Trust Vectors to Manage Risk for Interacting Stationary and Mobile Assets

When we add an automated guided vehicle (AGV) to a stationary asset we need the digital twin trust vector safety approach (scenario 2).

The trust vector can be used for predictive operation, handle failures properly and manage a degraded operation with the required safety level, understanding the differences between the assumptions within the intended use case and the conditions in the real use case.

Scenario 2—Superposition of trust vector values of DT-based systems—"degraded operation/intended use case."

How can a DT-based factory handle this situation for different types of failures and modifications?

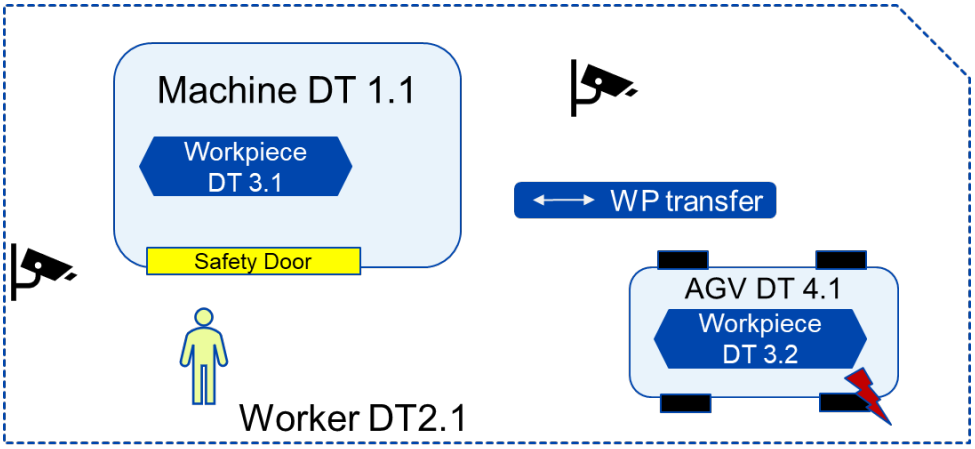


Figure 8-3: Scenario 2 showing turning machine, work piece, worker and AGV with wheel damage.

This scenario demonstrates handling two dimensions of the trust vector (reliability and safety). This scenario also demonstrates the importance of the digital twin definition from the DTC:

> "A Digital Twin is a *virtual representation of real-world entities and processes*, synchronized at a specified frequency and fidelity."

A hardware failure within one wheel impacts the reliability trust vector value. The safety trust vector value is dependent on reliability values of certain components. The change of the trust vector values in real-time of the vehicle allows the factory cell to derive the appropriate countermeasures. The virtual model of the digital twin corresponds to the asset and allows a smart way of handling this changed condition.

The turning machine system in our example consists of the turning machine and a work piece transfer device.

Such a machinery system with a safety level 4 requires a skilled worker with safety level 4 and knowledge of both the machine and the AGV to derive the correct next steps (see first example in Table 8-2).

A system with a safety level 5 could investigate the weight of the workpiece (DT 3.2) and the impact of the reduced AGV braking capability further to assess, decide and allow autonomous positioning of the AGV with reduced speed and the autonomous workpiece transfer (see second example in Table 8-2). The system with the safety level 5 would be, in this example, the broker previously described. It could be the turning machine (DT 1.1) or another system in the factory with the capability to be the broker.

Sys Req	Turning Machine System	People		Action
Safety Maturity Level 4	Safety Maturity Level 4	Level 4	Trained operator for this machine and the AGV.	Operator is aware, no actions. Manual docking by operator.
	Safety Maturity Level 5	Level 2, 3 & 4	Reduce speed if weight smaller than limit or broken AGV wheel.	AGV executes autonomous docking for workpieces with qualified weight.

Table 8-2: Safety Trust Vector Options for Scenario 1

By using trust vector values, the specified synchronization of the virtual representation of the digital twin, an appropriate model of control and safety functions and a context understanding of the situation can be used to detect hazards and properly reduce the risk by offering two options while maintaining the main production function:

- Target—degraded operation managed by trustworthiness vector.
- Target—extend operation mode for a hardware failure. For example, if the AGV has wheel damage, this can require a longer braking distance and require it to slow down. The dynamic trust vector values ensure transparency about this situation.
- Target—incomplete information—safety profiles cover must have DT dependencies and enable guidance within the context for autonomous decisions to enable safe and continuous production.

8.3 Scenario 3: Using a Digital Twin and Trust Vectors to Manage Risk for Assets in Unanticipated Situations

We enrich the indoor scenario of the last section with brownfield conditions (modification of assets, recipes, worker training procedures and environmental conditions) to illustrate that these real-world conditions are not always properly considered in today's static safety concepts.

The trust vector can be used to handle unexpected and unknown environmental conditions within the context and in a structured way. In scenario 3, we focus more on the environmental conditions and the DT capability to predict the next steps.

Scenario 3—Superposition of DT-based capabilities and trust vector values—"Higher productivity/dynamic resilience."

1x AGV, 1x machinery and sensors within the building—unknown ground floor conditions.

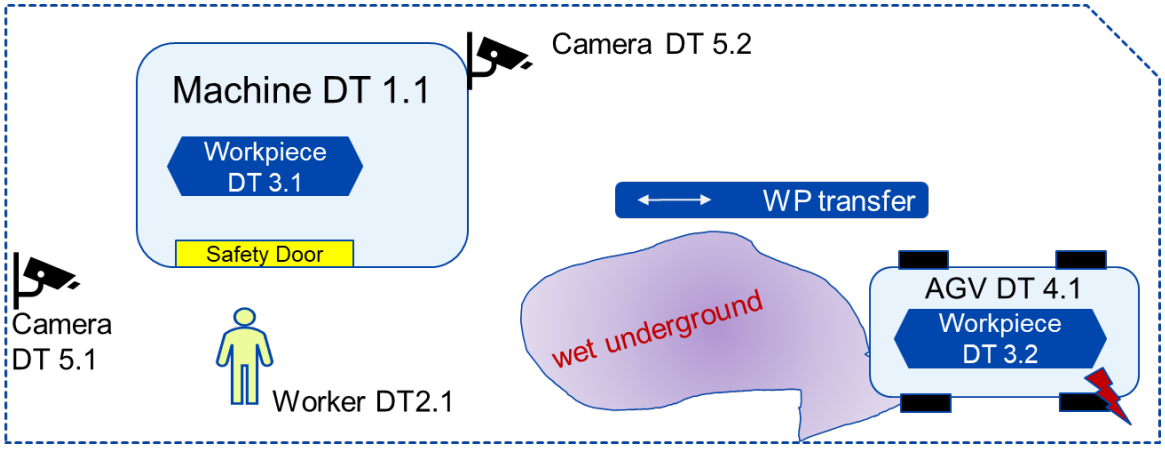


Figure 8-4: Scenario 3 showing turning machine, work piece, worker, AGV and wet underground.

A turning machine system with a safety level 4 can communicate with the camera (DT 5.2) and can recognize a deviation for ground floor conditions in front of the docking station of the turning machine.

The AGV (DT 4.1) has to reduce the speed before entering the wet area. The wet area impacts the performance of braking and detection by the AGV that would cause it to be too fast to guarantee the required safe speed in front of machine (DT 1.1). This would result in a dangerous situation for the worker and raise the possibility of damage to the machine.

By using trust-vector values, incorporating the detection capabilities of all cameras with the specified synchronization of the virtual representation of the DT environment, the context understanding of the situation can be achieved to detect the area floor change and properly reduce the speed of the AGV in advance of the wet area (to maintain the main production function).

This scenario describes the level of information required behind the trust vector on the safety profile side to ensure a decision capability within the context. An implementation example is given in 9.1 Manufacturing Domain.

In a situation where all relevant situational information is unavailable, the domain is needed for setting the parameters in compliance with the domain's socially recognized residual risk e.g. the safe speed needed. If we imagine a transport robot, like an AGV in a hospital, due to an implied

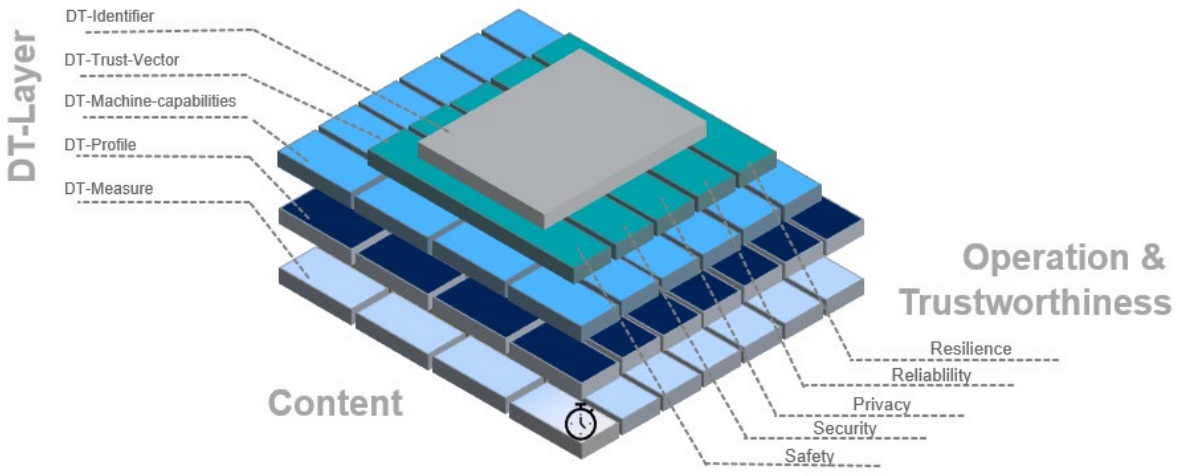
event the AGV is not able to differentiate between a patient and the hospital's technician. In this case the AGV should always pass the person as slowly as prudent for a patient.

9 VERTICAL SPECIFIC APPLICATION OF TRUST-VECTOR PRINCIPLE

9.1 MANUFACTURING DOMAIN

The practical implementation of "plug & produce"[8] on the shop floor from a safety perspective shows that there are functional relationships between parameters and information that must also be represented in the digital representation and especially between assets within a system-of-systems. A domain-specific implementation of the trust-vector principle incorporating interdependencies of data and information in different layers is shown in the 4D-DT-model in Figure 9-1. Mission-critical decisions in the manufacturing domain have to be based on a complete set of machinery properties, which are embedded in the 4D-DT-model in a structured way. In addition to the "identifier," the trust vector, and the machine-capabilities (first three layers) the machine's functions are also specified by DT-profiles and DT-measures (4 and 5 layer). An AGV will have a safety profile named "prevent collision" with which the available measures and corresponding functions are linked, e.g. braking for stopping or steering for route change. This detailed information model is required to ensure the compatibility of capabilities offered by different digital twins as described in Chapters 7 and 8 thus providing a basis to assess the trust vector score.

The fourth dimension of this digital-twin model represents the time variability of parameters or components a key parameter for mission critical real-time decisions. How to understand this model is illustrated by the following example.



[8] Industrie 4.0 term used to describe modular and flexible production.

Figure 9-1: Graphical visualization of the 4D-digital twin model (Source: TÜV SÜD).

The generic contents of the relationship pyramid are also included in this 4D DT model, except that they have been structured for illustration purposes from the point of view of technical implementation. This means that when digital twins interact, the relationship pyramid can be supplied with the necessary information from the corresponding layer and building block of the 4D model of the corresponding DT.

Mission-critical decisions require a digitalized risk management at run-time that is possible if the system-of-systems is based on a composable digital-twin architecture that follows the pyramid model, including integrity level represented by the trust vector and the corresponding profiles defining the validation and zero-trust criteria to make such decisions.

Why do we need more than the digital twin relationship pyramid for specific domains?

> The safety characteristic of the trust vector indicates a certain intrinsic level of hazardous potential of this specific asset or the integrity level of a safety measure. The understanding of the context of the situation requires the source of the hazard and the required counter measures in a pre-defined way and is handled by a strong link between the trust vector and the profile layer of all involved digital-twin-based assets. Some things are either safe or not safe, so partial negotiation of some safety requirement is not possible, although parties may decide which risks to accept under consideration of the socially accepted residual risk or to reject a potential loss by having a safety stop and down time as a result.

Safety measures are part of the safety case, well described and becoming part of most of the information models (OPC-UA, IND 4.0 AAS).

> Modularity makes it easy to adapt a digital twin to a modified asset, e.g. because a new braking system has been installed. The dependencies on the safety side are the same, except that the individual braking characteristics have changed, enabling a shorter braking distance.

The fundamental advantage of the 4D-DT model, including the trust vector is that an intelligence (“broker”) can select the best safety measure depending on the situational needs (within the context of the situation).

Instead of always assuming the worst-case scenario, which is state-of-the-art in safety, the most economically efficient measure that also meets the safety requirements is selected. A route change, for example, prevents in the same way as stopping a collision, with the only difference that in one scenario the component still reaches its next processing point on time and productivity can thus be maintained at a high level.

Figure 9-2 shows a visualization of a digital twin that exemplifies various functional relationships between parameters. The structure shown in it serves to make it easier to understand and assign individual aspects or parameters.

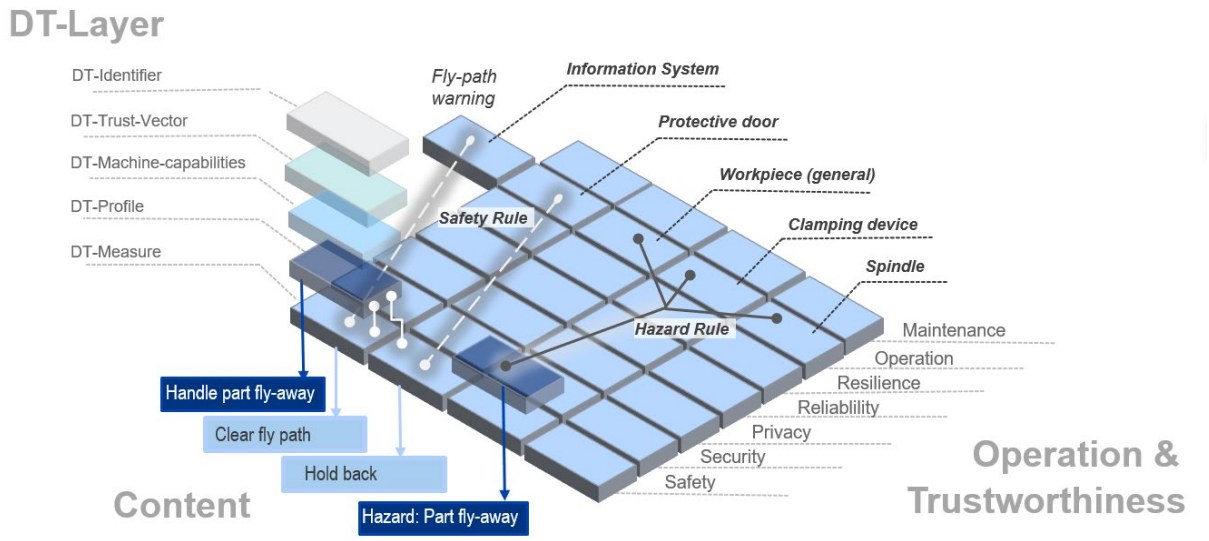


Figure 9-2: Graphical visualization of a digital twin of a turning machine (Source: TÜV SÜD).

The workpiece clamped in the clamp chuck, which is firmly connected to the spindle, “activates” the hazard of “part fly-away” when the spindle is required to rotate. The safety profile "handle part fly-away" is linked to the protective door and danger area warning as an example. For workpieces whose calculated kinetic energy exceeds the retention capacity of the protective door, a warning is issued with the possibility of defining supplementary measures as described above.

10 Conclusion and Outlook

Holistic understanding is one of the major benefits brought by digital twins.

Trustworthiness processes and technologies are not useful in isolation, they are implemented to support business operations in achieving business outcomes with less risk and more confidence.

Traditionally, this is achieved with *static* security and safety assurance cases: fixed checklists of system properties and process activities that everyone is judged against, planned in advance against contemporary threats and then fixed and followed rigidly.

Static assurance cases are not fit for purpose in a digital-first world: things move too fast, supply chains are too deep and systems are too complex for manual checklists, outdated standards and

annual (at best) audits. We are reaching the limits of assessment concepts. While safety software must not be changed to avoid re-testing, patches are mandatory from a security point of view.

Instead, we need to move to a world of *dynamic* assurance cases where standards are *outcome-based,* and target processes and system properties change according to need.

Such an approach may feel unfamiliar and unsettling for many practitioners, and there is little yet that can show dynamic assurances cases proven in use, but trustworthiness in digital twins must be able to account for a constantly changing environment, evolving threat landscapes and rapid decision-making based on incomplete and imported information. To achieve the first step, when the provider's score matches or is greater than the consumer side in all dimensions, the relationship of trust is established.

11 REFERENCES

[GDPR] General Data Protection Regulation (GDPR) Compliance Guidelines, 2016-04-16
https://gdpr.eu

[IEEE-IC2000] Electronic Commerce Trust Models and Metrics, IEEE Internet Computing, March/April 2000, pp. 35-43.
https://ieeexplore.ieee.org/abstract/document/832944

[IIC-IIRA2019] The Industrial Internet, Volume G1: Reference Architecture Technical Report, version 1.9, 2019-06-19, retrieved 2020-04-29
https://www.iiconsortium.org/pdf/IIRA-v1.9.pdf

[IIC-IISF2016] Industrial Internet of Things Volume G4: Security Framework, 2016-09-26, retrieved 2019-01-24
https://www.iiconsortium.org/pdf/IIC_PUB_G4_V1.00_PB-3.pdf

[IIC-IIV2019] The Industrial Internet, Volume G8: Vocabulary Technical Report, version 2.2, 2019-11-06, retrieved 2020-01-24
https://www.iiconsortium.org/pdf/IIC_Vocab_Technical_Report_2.0.pdf

[IIC-SMMD2020] IoT Security Maturity Model: Description and Intended Use, version 1.2, 2020-05-05, retrieved 2020-05-05
https://www.iiconsortium.org/pdf/SMM_Description_and_Intended_Use_V1.2.pdf

[IIC-SMMP2020] IoT Security Maturity Model: Practitioner's Guide, Version 1.2, 2020-05-05, retrieved 2020-05
https://www.iiconsortium.org/pdf/IoT_SMM_Practitioner_Guide_2020-05-05.pdf

[IIC-SMMRP2020] IoT SMM: Retail Profile for Point-of-Sale Devices, 2020-08-01, retrieved 2020-11-18
https://www.iiconsortium.org/pdf/SMM-Retail-Profile.pdf

[IIC-SMM-AP62443-Mappings]
The IoT Security Maturity Model (SMM): 62443 Mappings for Asset Owners and Product Supplier
https://www.iiconsortium.org/wp-content/uploads/sites/2/2022/08/SMM-62443-Asset-Owner-Product-Supplier-Service-Provider-Mappings-2022-08-16.pdf

[IIC-SMM-DTProfile]
IoT Security Maturity Model (SMM) Digital Twin Profile
https://www.digitaltwinconsortium.org/wp-content/uploads/sites/3/2022/06/SMM-Digital-Twin-Profile-2022-06-20.pdf

[MITRE-Privacy-Model]
Privacy Maturity Model Version 1. MITRE, 2019-10-20
https://www.mitre.org/sites/default/files/publications/pr-19-3384-privacy-maturity-model.pdf

AUTHORS & LEGAL NOTICE

This document is a work product of the Digital Twin Consortium Security & Trustworthiness Subgroup, chaired by Detlev Richter (TÜV SÜD) and David Shaw (Intuitus Cyber).

Authors: The following persons contributed substantial written content to this document: Anto Budiardjo (Padi), Jon Geater (RKVST), Frederick Hirsch (Upham Security), Michael Pfeifer (TÜV SÜD), Detlev Richter (TÜV SÜD).

Contributors: The following persons contributed valuable ideas and feedback that significantly improved the content and quality of this document: Marcellus Buchheit (WIBU-Systems).

Technical Editor: Dan Isaacs (DTC CTO) oversaw the process of organizing the contributions of the above Authors and Contributors into an integrated document. Stephen Mellor (OMG Executive VP) acted as IIC Technical Editor and ensured consistency across the IIC and DTC programs.