NIST Advanced Manufacturing Series 300-10

Recommendations on Ensuring Traceability and Trustworthiness of Manufacturing-Related Data

Thomas Hedberg, Jr.
Moneer Helu
Sylvere Krima
Allison Barnard Feeney

This publication is available free of charge from:
https://doi.org/10.6028/NIST.AMS.300-10

National Institute of Standards and Technology
U.S. Department of Commerce

NIST Advanced Manufacturing Series 300-10

Recommendations on Ensuring Traceability and Trustworthiness of Manufacturing-Related Data

Thomas Hedberg, Jr.
Moneer Helu
Allison Barnard Feeney
Systems Integration Division
Engineering Laboratory

Sylvere Krima
Engisis LLC

This publication is available free of charge from:
https://doi.org/10.6028/NIST.AMS.300-10

July 2020

U.S. Department of Commerce
Wilbur L. Ross, Jr., Secretary

National Institute of Standards and Technology
Walter Copan, NIST Director and Undersecretary of Commerce for Standards and Technology

National Institute of Standards and Technology
Advanced Manufacturing Series 300-10
Natl. Inst. Stand. Technol. Adv. Man. Ser. 300-10, 44 pages (July 2020)

This publication is available free of charge from:
https://doi.org/10.6028/NIST.AMS.300-10

Abstract

The purpose of this recommendations report is to provide practical how-to guidance for ensuring traceability and trustworthiness of manufacturing-related data. The recommendations contained herein address both subtractive and additive manufacturing processes. Commercial-solution developers, technical-assessment personnel, and interested end-users are the intended audience of this recommendations report. The scope of this report is guidance for deploying a system, or set of systems, for ensuring traceability and trustworthiness of manufacturing-related data. Product-related data formats conforming to selected international, open, consensus-based standards are in scope. Proprietary data formats are out of scope. File-based and data-stream representations are in scope. Data collected from computer numerical control subtractive fabrication processes (e.g., milling, turning) and CNC metal-forming additive (e.g., directed-energy deposition, powder-bed fusion) fabrication processes are in scope. Data collected from polymer-based additive processes, fiber-reinforced plastic (FRP) composite processes, and mass-conserving fabrication processes (e.g., casting, forging) are out of scope.

Key words

digital signature; trustworthiness; data traceability

Table of Contents

List of Tables

List of Figures

List of Listings

1. Introduction

1.1 Purpose

This recommendations report provides practical how-to guidance for ensuring traceability and trustworthiness of manufacturing-related data. This report contains recommendations that address both subtractive and additive manufacturing processes. This report targets commercial-solution developers, technical-assessment personnel, and interested end-users as the intended audience.

1.2 Scope

The scope of this report is guidance for deploying a system, or set of systems, for ensuring traceability and trustworthiness of manufacturing-related data. Product-related data formats conforming to selected international, open, consensus-based standards described in Sec. 2 are in scope. Proprietary data formats are out of scope. File-based and data-stream representations are in scope. Data collected from computer numerical control (CNC) subtractive fabrication processes (e.g., milling, turning) and CNC metal-forming additive (e.g., directed-energy deposition, powder-bed fusion) fabrication processes are in scope. Data collected from polymer-based additive processes, fiber-reinforced plastic (FRP) composite processes, and mass-conserving fabrication processes (e.g., casting, forging) are out of scope.

1.3 Verbal Forms

1.3.1 Must

The verbal form "must" indicates requirements to be followed strictly to conform to this document and from which no deviation is permitted. Must is synonymous with "shall."

1.3.2 Should

The verbal form "should" indicates that a possibility among a set of possibilities is recommended as particularly suitable (without mentioning or excluding other possibilities) or that a certain course of action is preferred but not necessarily required.

1.3.3 May

The verbal form "may" indicates a course of action permissible within the limits of the document.

1.4 Overview of Document

This report makes three contributions: 1) overview of relevant standards and technology, 2) trades study, and 3) recommendations for implementation. Sections 2 and 3 provide the overview of relevant standards and technology for manufacturing-related data. Section 4 provides the trades analysis of viable traceability and trustworthiness solutions. Section 5 provides the recommendations for ensuring traceability and trustworthiness of manufacturing-related data.

2. Overview of Relevant Engineering Standards

Only internationally recognized, open, consensus-based standards were selected for review and recommendation. Standards-based data formats should be used at all times to ensure effective traceability, data-quality control, and long-term archival and retrieval (LOTAR). At a minimum, a standard must be developed by an American National Standards Institute (ANSI) recognized standards development organization (SDO) to be included in this review.

2.1 ISO 10303 (STEP) Application Protocol 242

The standard, ISO 10303-242:2014 [3] titled “Managed model-based 3D engineering,” commonly known as STandard for the Exchange of Product Model Data Application Protocol 242 (STEP AP242), is an international standard that shows promise for enabling linked data. The goal of STEP AP242 is to support a manufacturing enterprise with a range of standardized information models that flow through a long and wide “digital thread” that makes the manufacturing systems in the enterprise smart [4]. Digital data plays a central role in achieving the goal of STEP AP242.

First published in December 2014, STEP AP242 contains extensions and significant updates to other STandard for the Exchange of Product Model Data (STEP) Application Protocols (APs) for product and manufacturing information (PMI), kinematics, and tessellation [5]. PMI is the presentation and representation of geometric dimensions and tolerances (GD&T), material specifications, component lists, process specifications, and inspection requirements within a three-dimensional (3D) product definition [6]. A study [6], comparing drawing-based processes to model-based processes, concluded that PMI has the potential to make many lifecycle processes run faster, with fewer errors, and at lower cost, since STEP AP242 offers standards-based models that include the representation of PMI that is computer interpretable [4]. This is a major breakthrough that supports manufacturing’s need for model-based computer-aided manufacturing (CAM) and coordinate-measurement system (CMS) processes. STEP AP242 increases the effectiveness of model-based enterprise (MBE) by enabling a common path for model-based definition (MBD) and model-based manufacturing (MBM) integration [5, 7].

Typical uses of STEP AP242 in industry are:

- computer-aided design (CAD) to CAD
- CAD to CAM and computer-aided inspection (CAI)
- LOTAR of design data

The industrial stakeholders currently driving changes to the standard are:

- Industrial original equipment manufacturers (OEMs) (e.g., aerospace, automotive)
- Defense manufacturing suppliers
- Government

Considerations regarding STEP implementation include:

- All commercially available geometry implementations use the EXPRESS language, ISO 10303-11 [8] and the EXPRESS-based file format, ISO 10303-21 [9]. There is a risk of limited future support as EXPRESS expertise diminishes.
- STEP is widely adopted in all major CAD systems, but users must ensure that their computer-aided technologies (CAx) systems support the relatively new STEP AP242.

2.2 ISO/ASTM 52915 (AMF)

International Standards Organization (ISO)/American Society for Testing and Materials (ASTM) 52915:2016 [10] is an international standard that describes an interchange format for additive-manufacturing technology. The standard provides the specification for the Additive Manufacturing File Format (AMF), which is intended to replace the long-standing Stereolithography (STL) file format. STL only supports a surface mesh to define a part. In contrast, AMF supports the part's boundary representation and also includes support for color, texture, material specification, and substructures.

The ISO/ASTM 52915:2016 standard [10] claims AMF is backward compatible with STL[1] while also being extensible for future capabilities. The standard requires that all AMF files must conform strictly to the published Extensible Markup Language (XML) schema. Both, ISO and ASTM make a standard XML Schema Definitions (XSD) available for the AMF specification. The standard defines a minimum implementation for both the AMF file producer and consumer. The AMF file producer must, at a minimum, generate a compressed file with a single object and no additional properties. The AMF file consumer must, at a minimum, read a compressed file with a single object and ignore all additional properties.

Typical uses of AMF in industry are:

- CAD to CAD
- CAD to CAM

The industrial stakeholders currently driving changes to the standard are:

- Academia
- Government

[1]The ISO/ASTM 52915:2016 standard allows a STL file to be converted directly to an AMF file.

- Industrial users
- Machine-tool-manufacturers and solution providers

Considerations regarding AMF implementation include:

- The standard provides an implementation guide that includes a description of each AMF element, mathematical operation, and function.
- There is limited commercial vendor implementations of the standard. Support for the standard in CAD and CAM tools may be available in only the latest versions of the tools.

2.3 ISO 6983 (G-code)

ISO 6983-1:2009 [11] is an international standard that defines the data format to program position, line motion, and contouring control systems in the numerical control (NC) of machines. This data format is commonly known as G-code. G-code was created at MIT in the late 1950s and, like CAD, rose in popularity through the 1970s [12]. Today, G-code is the near-universal format for programming computer-based NC machines.

G-code is generated typically from a manufacturing plan using a CAM system. G-code files are defined using a standardized ASCII-based set of commands. Each line of the G-code is a new command to the machine. Header information is standardized to support some traceability.

Typical uses of G-code in industry are:

- CNC control
- CNC operations

The industrial stakeholders currently driving changes to the standard are:

- CNC control manufacturers and solution providers

Considerations regarding G-code implementation include:

- Code logic and control is based on decades-old technology, when paper-tape / punch cards were used to numerically control machines. G-code limits the ability to handle highly complex geometries or to correct issues in-process.
- G-code is the only standard-based format for CNC programming, but machine-tool vendors have their own ”flavors” of the standard and unique machine tool capabilities lead to data format variations between machines.

2.4 ANSI MTConnect

MTConnect [13] is an open-source, read-only data-exchange standard for manufacturing equipment and applications developed by the MTConnect Institute. It provides a standard domain model and a basic minimum infrastructure to enable the creation of structured, contextualized data for client applications. In this way, MTConnect enables semantic interoperability between different manufacturing systems and client applications. While other communication protocols may exist for data transfer, the information models defined in MTConnect provide a common vocabulary and structure created for manufacturing-equipment data. Perhaps the most important type of data addressed by the standard is real and near-realtime data from the equipment (e.g., speed, position, program blocks). This data enriches the digital thread by providing information about the as-built condition of a part.

The MTConnect standard defines four types of information models for manufacturing equipment: Devices, Streams, Assets, and Interfaces [13]. Devices provides a representation of the physical and logical configuration for a piece of manufacturing equipment and the definition of data that may be reported by that equipment. Streams provides a specification for time-stamped data values returned by a piece of manufacturing equipment. Assets provides models of items that are not considered integral to a piece of manufacturing equipment but are used in the manufacturing process, e.g., cutting tools. Interfaces models the information needed to coordinate actions between different pieces of manufacturing equipment.

Users access the data and information provided by the information models in MTConnect through the Agent [13]. The Agent implements the MTConnect protocol and generates the relevant response document, which is traditionally in XML. MTConnect only standardizes communication between the Agent and an application. The Agent returns data only when requested by an application. The Agent acts as an Hypertext Transfer Protocol (HTTP) server and uses Representational State Transfer (REST) when interacting with any data source. In other words, the data source is responsible for sending state updates to the Agent.

Typical uses of MTConnect in industry are:

- Overall equipment effectiveness and utilization
- Process monitoring
- Condition monitoring

The industrial stakeholders currently driving changes to the standard are:

- Manufacturing equipment vendors
- Manufacturing solution providers and developers
- Industrial end-users (e.g., aerospace, defense)

Considerations regarding MTConnect implementation include:

- Deploying one agent for the entire shop floor versus deploying one agent per machine on the shop floor (e.g., one agent per machine has higher maintenance overhead but does not requiring turning off data collection for all machines during maintenance, using one Agent for multiple machines may present challenges with the Agent's buffer)
- Different communication protocols may be used for transporting the MTConnect-compliant information (e.g., Message Queuing Telemetry Transport (MQTT), Open Platform Communications United Architecture (OPC-UA))
- Curation of data may occur at various times of collection and transport (e.g., curate at machine edge, curate aggregation of data for a shop, curate date by part)

2.5 ANSI/DMSC Quality Information Framework

The Quality Information Framework (QIF) [14] is an ANSI-accredited standard sponsored by the Dimensional Metrology Standards Consortium (DMSC). QIF defines an integrated set of XML information models that enable the effective exchange of metrology data throughout the entire metrology process. QIF handles feature-based dimensional metrology, quality measurement planning, first article inspection, and discrete quality measurement. QIF supports defining or importing the product definition and reusing data for inspection planning, execution, analysis, and reporting.

QIF uses terminology and semantics from the inspection world to represent the various elements in the QIF specification. QIF uses XSD to normalize the information models. The standard organizes QIF XSDs into six application areas for metrology: (1) MBD, (2) Rules, (3) Resources, (4) Plans, (5) Results, (6) Statistics. The standard combines the MBD (containing the product definition) with measurement rules and resources definitions to generate a plan. The plan is then executed and the results are captured. Multiple results are combined to generate statistics. QIF does not perform the task of statistics and the other metrology methods. Instead, QIF enables the ability to put raw inspection data into a quality context that is computer-processable.

Typical uses of QIF in industry are:

- Capturing and transmitting inspection results between organizations
- Describing available measurement assets

The industrial stakeholders currently driving changes to the standard are:

- Industrial users (e.g., aerospace, defense)
- Metrology solution providers
- Government

Some general implementation considerations are:

- All parts of QIF are not implemented by all commercial solution providers. QIF results is the most adopted part of the standard.
- Persistent ID of features and characteristics are optional in the QIF standard, but required for full traceability of quality across the product lifecycle.

2.6 ISO 32000 and ISO 14739 (PDF/PRC)

The Portable Document Format (PDF) [15] and Product Representation Compact (PRC) [16] International Standards are often combined into technologies for visualizing product-definition data. PDF/PRC is commonly referred to as a 3D PDF. While there are several flavors of 3D PDF, the combination of PRC embedded in a PDF document is emerging as the industry recommended practice. PDF/PRC enables the display of 3D product definition in any PDF reading software that conforms to the standard. Using PDF/PRC enables effective and efficient visualization of product data throughout the lifecycle for human-consumption. A significant amount of metadata may be included in the PRC and additional metadata can be included in the PDF document via XML namespaces wrapped in a Extensible Metadata Platform (XMP) package.

Typical uses of PDF/PRC in industry are:

- Visualization of design data without the use of CAD tools
- Container for transmitting a technical data package (TDP)

The industrial stakeholders currently driving changes to the standard are:

- PDF solution providers
- Industrial users (e.g., aerospace, defense)
- Government

Considerations regarding PDF/PRC implementation include:

- PRC includes a boundary representation (BREP), but no commercially available manufacturing or quality systems interact directly with the BREP in manufacturing or quality processes.

3. Overview of Other Relevant Standards and Technologies

3.1 ISO/IEC 9594-8 (X.509)

The X.509 standard [17], titled *Information Technology – Open Systems Interconnection – The Directory – Part 8: Public-key and Attribute Certificate Frameworks*, was first published in 1988 and since updated several times. The Telecommunication Standardization

Sector of the International Telecommunication Union (ITU-T) developed the standard, first as a recommendation, intended as the authentication framework for the X.500 series of electronic directory services. The X.509 standard normalizes two concepts for authentication and authorization. The first is Public Key Infrastructure (X.509-PKI) [18] and second is Privilege Management Infrastructure (X.509-PMI) [19]. X.509-PKI addresses authentication and X.509-PMI addresses authorization.

Figure 1 displays the basic components of X.509-PKI (1a) and X.509-PMI (1b). X.509-PKI creates and manages digital certificates – primarily for authentication with a certificate authority at the top of a certificate hierarchy. The hierarchy consists of hardware, software, people, policies, and procedures [1]. Common implementations of X.509-PKI today use asymmetric (public) key encryption, where a user is issued both a private key that is only known to the user and a public key that is known to everyone [1]. X.509-PKI is the most familiar certificate infrastructure used by end-users.

X.509-PMI is less known to end-users. X.509-PMI is similar to X.509-PKI, except X.509-PMI is used for authorization. X.509-PMI manages user authorizations with an attribute authority at the top of a certificate hierarchy [1]. The attribute authority references an X.509-PKI identity and delegates privileges to the identity based on the assigned privileges from a "source of authority." The attribute authority issues an "attribute certificate" that is linked to the identity provided by the X.509-PKI-based certificate. Adoption of the X.509-PMI in practice has been minimal with only a few commercially available applications.

In practice, X.509-PKI is implemented significantly more than X.509-PMI. X.509-PKI enjoys a broad range of applications – most notably Secure Sockets Layer (SSL)/Transport Layer Security (TLS) encryption of websites and Secure/Multipurpose Internet Mail Extensions (S/MIME) signing/encrypting of emails. X.509-PMI has seen minimal-to-no commercial adoption since its introduction to the X.509 standard in 2001. This is, in part, due to the rise of service-oriented architectures (SOAs) and attribute assertions via the Security Assertion Markup Language (SAML) specification [20] developed by Organization for the Advancement of Structured Information Standards (OASIS) [21].

X.509-PKI can be extended to include authorization information by embedding additional metadata in signatures to describe privileges. The recommendations contained herein use X.509-PKI primarily and include additional privilege metadata to manage authorization requirements. Taking this approach enables us to simplify the implementation of X.509 constructs while introducing traceability, authentication, and authorization to manufacturing-related data.

3.2 Digital Manufacturing Certificates Toolkit

The digital manufacturing certificate (DMC) toolkit is software developed at National Institute of Standards and Technology (NIST) and is described in Refs. [2, 22]. The DMC toolkit supports digital signing and embedding X.509-PKI certificates [18] into several open-data formats used in manufacturing: ISO 10303-242:2014 (STEP AP242) [3], ISO

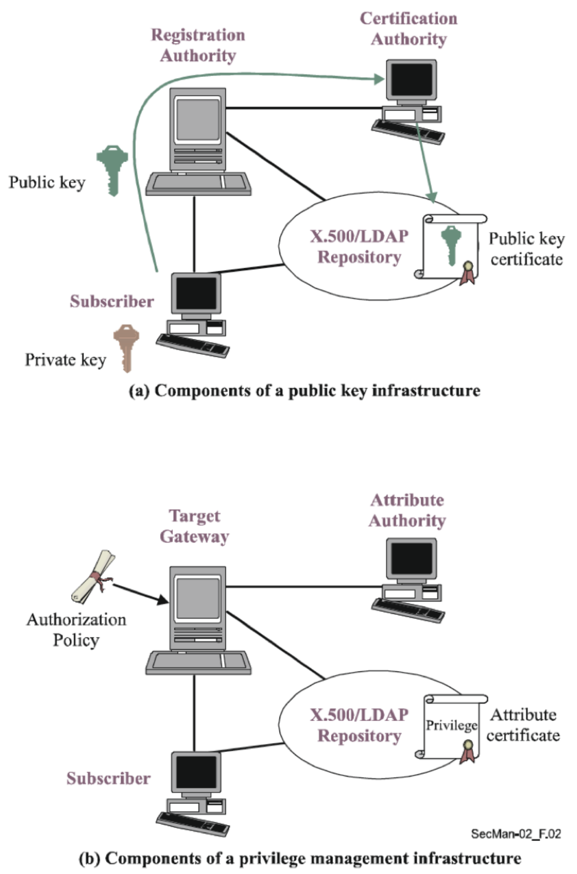

Fig. 1. X.509 components of public key infrastructure and privilege management infrastructure (from Ref.[1])

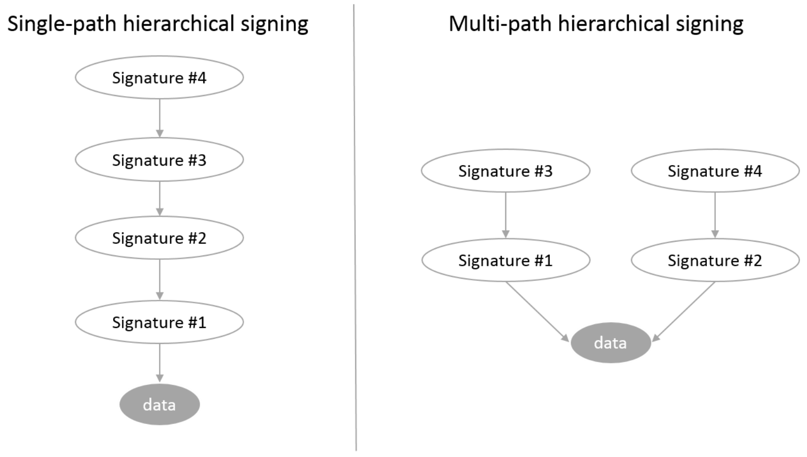

Fig. 2. Examples of the single-path and multi-path hierarchical signing employed in the digital manufacturing certificates toolkit (from Ref. [2])

6983-1:2009 (G-Code) [11], ANSI / DMSC QIF [14], and the combined standards of PDF [15] and PRC [16] to form a 3D PDF. People and systems can use the toolkit to digitally sign data using either a single- or multi-path signature hierarchy (see Fig. 2).

In single-path hierarchies, each signer must vouch for all the previous signatures. Mandatory vouching raises legal issues and sets contradictory constraints when data must be legally signed and endorsed by multiple actors from different organizations along the product lifecycle. In multi-path hierarchical signing, a new signature does not have to vouch for the previous signature in the file and multiple organizations can sign the same data while vouching only for signatures they choose. Signatories may also choose to vouch only for the data, and not for other signatures.

4. Trades Analysis for Implementation Aternatives

Data traceability is paramount to enabling trustworthiness throughout the product lifecycle. Simply providing a digital signature on data is neither sufficient nor feasible due to the complexity of the supply chain and the heterogeneity of the data exchanged. This gap was realized through validation of previous work in Refs. [2, 23].

In a complex environment composed of numerous partners and exchanges, embedding traceability data in only files can bloat the product data with information not required by every actor. A complete traceability can not be guaranteed due to the heterogeneity of the data and the need for every file format to support a traceability mechanism. Proprietary and/or binary files are heavily used and do not offer an efficient transparent way of auditing the traceability information. Moreover, numerous open-formats may not support such a mechanism either. Lastly, embedding traceability information in files makes the audit process cumbersome, requiring access to and processing of all the files, which is an enormous amount of data. Therefore, to overcome these challenges and to address efficient audit

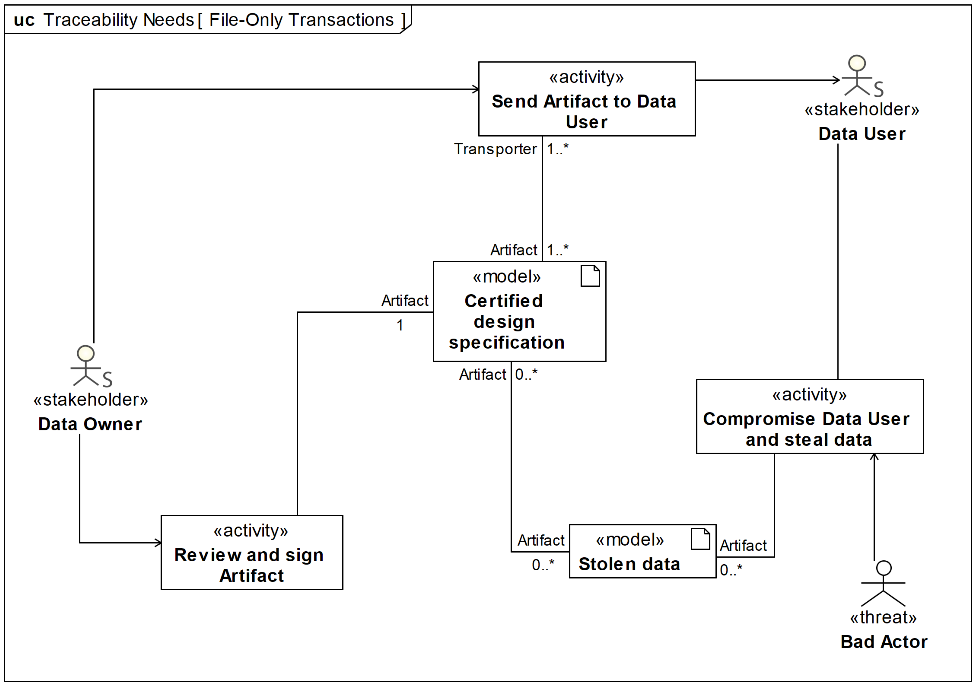

Fig. 3. Option for digitally signing a design specification (e.g., CAD model) and exchanging the data with another data user

needs, we suggest to combine previous work with recording traceability information externally in a safe and shared repository such as a distributed ledger [24] that offers a shared, trusted, and virtually tamper-resistant source of information.

We identified three options for tracing data transactions throughout the lifecycle. A data transaction occurs anytime data ownership is declared or when data is exchanged between two actors. The first options is file-only transactions, discussed in Sec. 4.1. The second option is distributed-ledger-registered transactions, discussed in Sec. 4.2. The third option is streaming-data packages, discussed in Sec. 4.3.

4.1 Option 1: File-Based Traceability

File-only transactions are asynchronous, require significant leveraging of X.509-PKI certificates, and require trust of the other actors with whom data is exchanged. Traceability is managed with metadata stored within the data files. Figure 3 presents a use-case diagram for digitally signing a design specification and exchanging the file with another data user, using the DMC toolkit described in Sec. 3.2.

Three actors are depicted in the file-based traceability option: 1) data owner, 2) data user, and 3) bad actor. The data owner (herein owner) and data user (herein user) are the normal roles that would typically share data while executing tasks. When the owner is prepared to release the data to the user, the owner could review and sign the data using the DMC toolkit. Then, the owner would send the data to the user. The owner and user would store that signed data in their respective data repositories. The user would use the data to complete all agreed-upon tasks for the owner (e.g., supplier fabricates a part for a customer). This portion of the use case represents typical manufacturing-related business relationships.

Data could be compromised and/or stolen from owners and users by bad actors. In the file-based traceability option, a bad actor could steal data from the user by compromising (e.g., gaining unauthorized access) the user's data repository. The bad actor would have access to the signed data. If the owner then found the signed data in the possession of an unauthorized actor, the owner could go back to his/her repository and determine all the users the data was sent to by querying and reviewing the certificate and metadata. This would provide the owner the ability to discover who received the data and request those users to investigate their systems for breaches. In this case, the owner would simply discover that he/she has a data problem, but the owner would not immediately know the root cause of that problem without further investigation.

However, the file-based traceability option represents a solid foundation with which to build data-traceability principals and methods. Having the ability to quickly impart additional metadata into a file and then later be able to trace where the data came from, its purpose, and potential uses would reduce the risk of errors being introduced due to the wrong data being used or because of changes that went unnoticed.

4.2 Option 2: Distributed Ledger

Distributed-ledger transactions are synchronous and require leveraging X.509-PKI certificates and a technology like blockchain [24]. Traceability is managed with transactions registered in a distributed ledger. Figure 4 presents a use-case diagram for digitally signing a design specification, using the DMC toolkit described in Sec. 3.2, and registering data-ownership and data-exchange transactions in a distributed ledger.

The same three actors depicted in the file-based traceability option are also depicted in the distributed-ledger traceability option. The owner and user are still the normal roles that would typically share data between each other for the purposes of executing tasks. However, in this case, when the owner is prepared to release and send the data to the user, the owner would review and sign the data using the DMC toolkit and register the signature fingerprint in a distributed ledger to prove ownership of the data. Krima et al. recommend storing only the signature fingerprint in the distributed ledger, registering the signature fingerprint in a transaction sent by the owner to him/herself for proving ownership, and then registering the signature fingerprint in transactions whenever the data is sent to a user [25]. The owner and user would still store signed data in their respective data repositories. The user would also still use the data to complete all agreed-upon tasks for the owner (e.g., supplier fabricates a part for a customer). This portion of the use case, like the file-only transactions, represents typical manufacturing-related business relationships with the only difference being that each action on the data is registered in a distributed ledger.

The strength of the distributed-ledger traceability option is in dealing with bad actors.

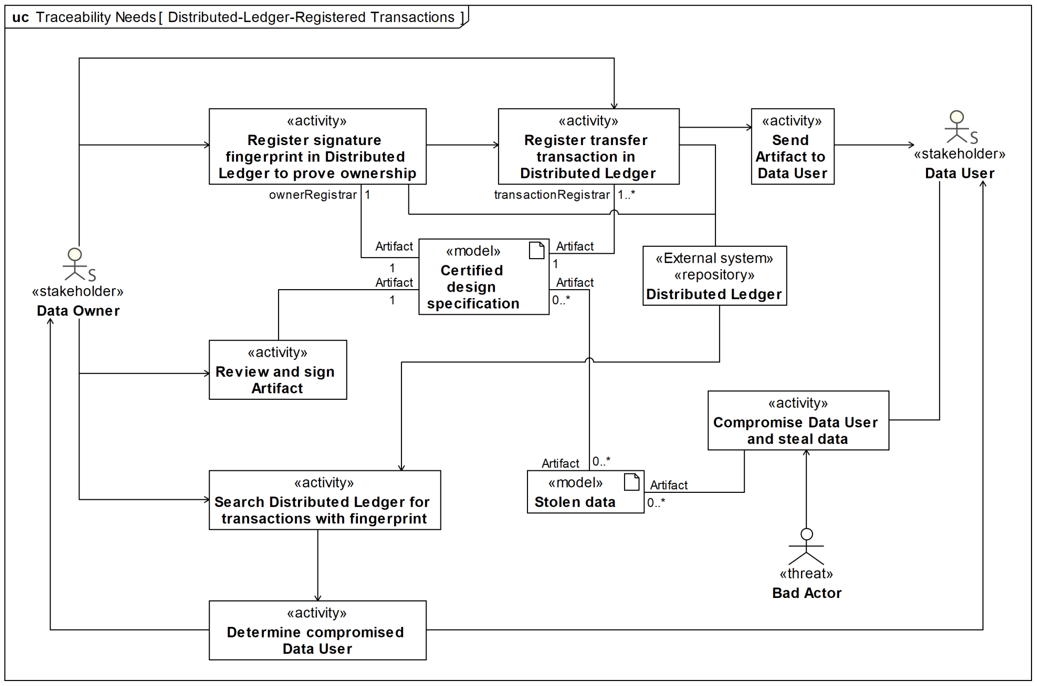
uc Traceability Needs [Distributed-Ledger-Registered Transactions]
«activity» Register signature fingerprint in Distributed Ledger to prove ownership
«activity» Register transfer transaction in Distributed Ledger
«activity» Send Artifact to Data User
«stakeholder» Data User
ownerRegistrar 1
transactionRegistrar 1..*
Artifact 1
Artifact 1
«model» Certified design specification
Artifact 1
Artifact 0..*
«External system» «repository» Distributed Ledger
«stakeholder» Data Owner
«activity» Review and sign Artifact
«activity» Compromise Data User and steal data
Artifact 0..*
«model» Stolen data
Artifact 0..*
«activity» Search Distributed Ledger for transactions with fingerprint
«threat» Bad Actor
«activity» Determine compromised Data User

Fig. 4. Option for digitally signing a design specification (e.g., CAD model) and registering ownership and data-exchange transactions in a distributed ledger

If the owner found signed data in possession of an bad actor, the owner could query the distributed ledger and determine the exact transaction that was related to the compromised data. This provides the owner the ability to discover exactly who was authorized to receive the data originally and request that user to investigate his/her systems for breaches. In this case, the distributed-ledger traceability option is differentiated from the file-based traceability option because the owner would discover that he/she has a data problem and immediately know the root cause of the problem without further investigation.

4.3 Option 3: Streaming-Data Packages

Streaming-data packages is an emerging option for providing data traceability and protecting the intellectual property (IP) included in the packages. In the additive-manufacturing domain, a few commercial proprietary platforms exist that claim to stream data directly to the manufacturing machines for fabricating hardware. However, all the commercial solutions are closed platforms and the state of their standards implementations are unknown.

The research literature also includes several papers related to distributed cloud manufacturing or manufacturing-as-a-service [26–29]. These papers claim the data could be streamed to a localized manufacturing services regardless of process. Putting the feasibility of the technologies aside, most of the research literature proposes different methods for streaming-data packages. The research literature does not currently propose a common method for streaming-data packages.

While there is a significant amount of activity and possible solutions available for streaming data packages, more work to achieve consensus on an approach is needed. None of relevant engineering standards listed in Sec. 2 support digital signatures or other traceability methods to build trust in the data streams. Further, there is little research in digitally signing streaming data. Therefore, the file-based traceability and distributed ledger options are the recommended approaches until consensus is achieved for streaming-data packages.

5. Recommendations for Traceability Implementation and Workflows

5.1 Schema Recommendations

The majority of manufacturing-related data standards used widely by industry are based on EXPRESS or XML information modeling methods. Therefore, we recommend traceability and trustworthiness extensions for EXPRESS- and XML-based implementations.

5.1.1 ISO 10303-21 EXPRESS-Based File Exchange

ISO 10303-21 does not yet support the flexibility required by a typical organization's business requirements for digital signatures.

ISO 10303-21 specifies Wirth Syntax Notation (WSN) [30] that define its formal grammar. A meta-syntax, or grammar, is a set of rules that describe and constrain a domain-specific language and its valid syntax and vocabulary. The current WSN of the ISO 10303-21 exchange structure is shown in Listing 5 in the Appendix.

The notation for optional signatures was added to edition 3 of the ISO 10303-21 specification, and denoted by the special token ”SIGNATURE”. A signature follows the file content that it verifies. A file may contain multiple signatures. If so, each signature verifies the content that precedes the ”SIGNATURE” token, including any previously defined signatures.

We recommend that ISO 10303-21 be enhanced to support multi-path hierarchical signing as shown in Fig. 2. In multi-path hierarchical signing, a new signature does not necessarily verify the previous signatures in the file. This allows multiple organizations to sign the same file while only verifying signatures issued by their organizations.

This enhancement to ISO 10303-21 require syntax extensions to support the following two requirements:

- A signature block must enable a signer to vouch for other existing signatures to support multi-path hierarchical signings
- A signature block must enable a signer to attach a set of metadata to document the signature and represent the file transformation information

This enhancement requires each signature block to begin with the special token `TRACE` and must terminate with the special token `ENDSEC;`.

The enhanced signature block contains three properties: 1) a SIGNATURE_TRACE that contains a set of metadata in a JavaScript Object Notation (JSON) representation, 2) a SIGNATURE_PATH that contains information about the signature itself and what it verifies, 3) a SIGNATURE that contains the signature itself.

The SIGNATURE_PATH property of a signature block has four attributes as follows: 1) sig_tech_type defining the technology used to generate the signature, 2) anchorID, a list of universally unique identifiers (UUIDs) in an ANCHOR section to identify the data being signed (or * when the ANCHORs should be ignored and the entire file signed), 3) binCrossSign, a Boolean that indicates whether this signature verifies other signatures, 4) tracesIDs, a list of IDs of other signature blocks verified by this signature, if binCrossSign is set to true. The SIGNATURE_PATH property is the key to configuring the signature and creating complex and hierarchical signature chains. If a signature block verifies the exchange content only, it must also verify the elements of the signature block, with the exception of the SIGNATURE property (see example below). If a signature block verifies the exchange content and other signatures, it must verify the exchange content, the signature blocks corresponding to its traceIDs (in the order they are given), and finally also the elements of the signature block itself, with the exception of the SIGNATURE property (see Listing 1 and Listing 2). In Listing 1, the SIGNATURE property verifies the exchange content, as well as lines 1, 2, 3, and 5. In Listing 2, the SIGNATURE property verifies the exchange content, as well as lines 1, 2, 3, 4, 5, 6, 7, 8, and 10.

Listing 1. A signature block that only verifies the exchange content.

```
1  TRACE: #122
2    #123 = SIGNATURE_TRACE({...}})
3    #124 = SIGNATURE_PATH('PKCS', *, N, [])
4    #125 = SIGNATURE('MIIGpgYJKoZIhvcNAQcCoIIGlzCCBpMCAQExCzAJBgUrDg
         ↪ MCGgUAMAsGCSqGSIb3DQEHAaCCA9cwggPTMIICu6ADAgECAgEEMA0GCSqGSIb3DQ
         ↪ EBCwUAMHoxEzARBgoJkiaJk/IsZAEZFgNjb20xGTAXBgoJkiaJk/IsZAEZFglzdG
         ↪ VwdG9vbHMxFzAVBgNV...')
5    ENDSEC;
```

5.1.2 XML Schema Definition

While ISO 10303-21 is used heavily to represent manufacturing data in accordance to the STEP standard, a significant number of other formats are XML-based. AMF, MTConnect, and QIF are all encoded in XML and structurally comply with their own XML schema following the XSD specification. An XSD formally defines the information structure of an XML document and is used to validate the compliance of said XML document. An AMF document must respect the AMF XSD, an MTConnect document must respect the MTConnect XSD, and a QIF document must respect the QIF XSD.

Listing 2. A STEP-based signature block that verifies the exchange content and a previous signature.

```
TRACE: #122
#123 = SIGNATURE_TRACE({...}})
#124 = SIGNATURE_PATH('PKCS', *, N, [])
#125 = SIGNATURE('MIIGpgYJKoZIhvcNAQcCoIIGlzCCBpMCAQExCzAJBgUrDg
    ↪ MCGgUAMAsGCSqGSIb3DQEHAaCCA9cwggPTMIICu6ADAgECAgEEMA0GCSqGSIb3DQ
    ↪ EBCwUAMHoxEzARBgoJkiaJk/IsZAEZFgNjb20xGTAXBgoJkiaJk/IsZAEZFglzdG
    ↪ VwdG9vbHMxFzAVBgNV...')
ENDSEC;
TRACE: #126
#127 = SIGNATURE_TRACE({...}})
#128 = SIGNATURE_PATH('PKCS', *, Y, [#122])
#129 = SIGNATURE('MIIGpgYJKoZIhvcNAQcCoIIGlzCCBpMCAQExCzAJBgUrDg
    ↪ MCGgUAMAsGCSqGSIb3DQEHAaCCA9cwggPTMIICu6ADAgECAgEEMA0GCSqGSIb3DQ
    ↪ EBCwUAMHoxEzARBgoJkiaJk/IsZAEZFgNjb20xGTAXBgoJkiaJk/IsZAEZFglzdG
    ↪ VwdG9vbHMxFzAVBgNV...')
ENDSEC;
```

In their current versions, only the QIF XML schema supports digital signature and traceability information. To provide a similar support, MTConnect, AMF, and any other XML-based format, may leverage the XSD import mechanism. Listing 6 in the Appendix provides the recommended XML schema for digital signature and traceability information. The XSD import mechanism provides a way to reuse existing definitions from another schema: define once, reuse everywhere. Not only does this ensure consistency across the different formats, this also reduces the cost of adoption.

The recommended XML schema imports the official World Wide Web Consortium (W3C) XML schema for digital signature. This schema defines the information structure to represent digital signatures and associated metadata (e.g., the type of signature, the issuer of the signature, the signature algorithm) documented in the XML Signature Syntax and Processing Version 2.0 [31].

A signature block is implemented through the `TraceBlockType` XML complex type and contains the following three elements: 1) a `SigTrace` element that contains a set of metadata in a JSON representation, 2) a `SigPath` element that contains information about the signature and what it verifies, 3) a `Signature` element, from the W3C XML schema for digital signature, that contains the signature.

A `TraceBlockType` contains an additional attribute (`traceBlockID`) used to represent its unique identifier. The `SigPath` element is composed of a set of three elements: 1) the `lstIndex`, a list of semi-colon separated indexes, representing the list of IDs of the XML elements being signed, or the '*' character, if the entire document is signed, 2) `bolCrossSign`, a Boolean that indicates whether this signature verifies other signatures, 3) `lstTraceID`, a list of IDs (from the `traceBlockID` attribute) of other signature blocks verified by this signature, if `bolCrossSign` is set to true.

Listing 3 provides a XML example with multiple signature blocks. The first signature block verifies the exchange content while the second signature block verifies both the exchange content and the first signature block.

Listing 3. A XML-based signature block that verifies the exchange content and a previous signature.

```
<TraceBlockType traceBlockID="122">
<SigTrace>{...}</SigTrace>
<SigPath>
    <lstIndex>*</lstIndex>
    <bolCrossSign>false</bolCrossSign>
    <lstTraceID/>
</SigPath>
<ds:Signature Id="signature122">
    <ds:SignedInfo>
        <ds:CanonicalizationMethod
            ↪ Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
        <ds:SignatureMethod
            ↪ Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
        <ds:Reference>
            <ds:DigestMethod
                ↪ Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
            <ds:DigestValue>dGhpcyBpcyBub3QgYSBzaWduYXR...</DigestValue>
        </ds:Reference>
    </ds:SignedInfo>
    <ds:SignatureValue>MIIGpgYJKoZIhvcNAQcCoIIGlzCCBpMCAQExCzAJBgUrDg
        ↪ MCGgUAMAsGCSqGSIb3DQEHAaCCA9cwggPTMIICu6ADAgECAgEEMA0GCSqGSIb3DQ
        ↪ EBCwUAMHoxEzARBg...</Sds:ignatureValue>
</ds:Signature>

</TraceBlockType>
<TraceBlockType traceBlockID="126">
    <SigTrace>{...}</SigTrace>
    <SigPath>
        <lstIndex>*</lstIndex>
        <bolCrossSign>true</bolCrossSign>
        <lstTraceID>122</lstTraceID>
    </SigPath>
    <ds:Signature Id="signature126">
        <ds:SignedInfo>
            <ds:CanonicalizationMethod
                ↪ Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
            <ds:SignatureMethod
                ↪ Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
            <ds:Reference>
```

```
                <ds:DigestMethod
                    ↪ Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
                <ds:DigestValue>dGhpcyBpcyBub3QgYSBzaWduYXR...</DigestValue>
            </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>MIIGpgYJKoZIhvcNAQcCoIIGlzCCBpMCAQExCzAJBgUrDg
            ↪ MCGgUAMAsGCSqGSIb3DQEHAaCCA9cwggPTMIICu6ADAgECAgEEMA0GCSqGSIb3DQ
            ↪ EBCwUAMHoxEzARBg...</Sds:ignatureValue>
    </ds:Signature>
</TraceBlockType>
```

5.2 Use Case Recommendations

We identified three uses cases that cover common manufacturing-related data traceability and trustworthiness needs across the product lifecycle. The first use case addresses communicating design information (see Sec. 5.2.2), the second use case addresses generating manufacturing programs (see Sec. 5.2.3), the third use case addresses generating and communicating inspection data (see Sec. 5.2.4).

5.2.1 Common Use Case Recommendations

An example component with part ID `TransverseFrame-001` is used for describing all the use cases. The use cases leverage a combination of file-based traceability (see Sec. 4.1) and distributed ledgers (see Sec. 4.2).

Listing 4 provides the signatures included in a STEP file for the component using the file-based traceability method described in Sec. 5.1.1. The `SIGNATURE_TRACE` element on lines 13, 18, and 23 of Listing 4 contains formatted metadata that describes the context under which the data was digitally signed. At a minimum, the attributes listed in Table 1 must be included in the `SIGNATURE_TRACE` (for STEP) and `SigTrace` (for XML) elements when using the file-based traceability method.

The `source` and `destination` attributes should contain Uniform Resource Identifiers (URIs) to enable link-data approaches [32]. The `usage` and `operation` attributes are proposed currently as strings. The values for the `usage` attribute should be limited to the *maturity states*, defined by American Society of Mechanical Engineers (ASME) Y14.47-2019 [33], which are: *conceptual (M1), developmental (M2), production (M3), and archive (M4)*. The values for the `operations` attribute should be added to a data dictionary negotiated by the sending and receiving parties, but we recommend at least including the terms: review, verification, validation, release, revision, supersede. The `date` attributes should be included using the ISO 8601 complete date-time format [34].

The recommended minimum set of metadata must be included using a JSON structure. This enables efficient processing of the metadata by both humans and machines. Using a JSON structure also allows for extending the minimum set of metadata, but the set proposed here must remain the minimum set of attributes for the metadata. Additional

Table 1. Recommended minimum traceability attributes to include in siganture-trace metadata.

Element	Description	Type
source	identifies the source data that was reviewed and digitally signed, may be a circular reference to data containing the `SIGNATURE_TRACE` element	URI per RFC 7320 [38]
destination	identifies the actual signed data, which may be a circular reference to data containing the `SIGNATURE_TRACE` element	URI per RFC 7320 [38]
usage	the purpose(s) / use(s) for which the signed data is authorized	string
operation	the reason why the data was signed (e.g., release, revision, validated)	string
date	the date the operation was completed	YYYY-MM-DDThh:mm:ss per ISO 8601 [34]

attributes and/or metadata can be included from standardized data dictionaries, such as ISO/IEC 11179 [35] and the Open Applications Group Integration Specification (OAGIS) [36, 37]. The need for extending the metadata and what additional metadata must be included should be determined by negotiations between the data owners, signers, and consumers.

Listing 4. Embedded *Verification*, *Release*, and *Revision* traceability information in a STEP document for a naval ship part

```
 ISO-10303-21;
HEADER;
FILE_DESCRIPTION((‘TransverseFrame-001 for demonstration of trust and
    ↪ traceability in the product lifecycle’),‘2;1’);
FILE_NAME(
    ↪ ‘TransverseFrame-001_rev01.stp’,‘2019-07-17T13:21:18’,(‘’),(‘’),‘’,‘’,‘’);
FILE_SCHEMA((‘AP242_MANAGED_MODEL_BASED_3D_ENGINEERING_MIM_LF { 1 0
    ↪ 10303 442 1 1 4 }’));
ENDSEC;
DATA;
#1=APPLICATION_CONTEXT(‘Managed model based 3d engineering’);
.
.
.
TRACE:#3415
#3416 =
    ↪ SIGNATURE_TRACE({source:‘URI:20.500.11993\734.13.TransverseFrame.001’,
    ↪ date:‘2019-06-14T11:39:54’, operation:‘verification’,
    ↪ usage:‘production’, result:‘pass with warnings’,
    ↪ report:‘URI:15.1115\734.13.TransverseFrame.001.verification’})
#3417 = SIGNATURE_PATH(‘PKCS’,*,N,[])
#3418 = SIGNATURE(‘MIIGpgYJKoZIhvcNAQcCoIIGlzCCBpMCAQExCzAJBgUrDg
    ↪ MCGgUAMAsGCSqGSIb3DQEHAaCCA9cwggPTMIICu6ADAgECAgEEMA0GCSqGSIb3DQ
    ↪ EBCwUAMHoxEzARBgoJkiaJk/IsZAEZFgNjb20xGTAXBgoJkiaJk/IsZAEZFglzdG
    ↪ VwdG9vbHMxFzAVBgNV...')
ENDSEC;
TRACE:#3419
#3420 = SIGNATURE_TRACE({source:'URI:20.500.11993\734.13.wingrib',
    ↪ destination:'URI:15.1115\734.13.TransverseFrame.001.release',
    ↪ usage:'production', date:'2019-06-20T15:18:36',
    ↪ operation:'release'})
#3421 = SIGNATURE_PATH('PKCS',*,Y,[#3415])
#3422 = SIGNATURE('MIIGpgYJKoZIhvcNAQcCoIIGlzCCBpMCAQExCzAJBgUrDg
    ↪ MCGgUAMAsGCSqGSIb3DQEHAaCCA9cwggPTMIICu6ADAgECAgEEMA0GCSqGSIb3DQ
    ↪ EBCwUAMHoxEzARBgoJkiaJk/IsZAEZFgNjb20xGTAXBgoJkiaJk/IsZAEZFglzdG
    ↪ VwdG9vbHMxFzAVBgNV...')
ENDSEC;
TRACE:#3423
#3424 =
    ↪ SIGNATURE_TRACE({source:'URI:20.500.11993\734.13.wingrib.release',
    ↪ destination:'URI:15.1115\734.13.TransverseFrame.001.change01',
    ↪ usage:'production', date:'2019-07-17T13:21:18',
    ↪ operation:'revision'})
#3425 = SIGNATURE_PATH('PKCS',*,Y,[#3419])
#3426 = SIGNATURE('MIIGpgYJKoZIhvcNAQcCoIIGlzCCBpMCAQExCzAJBgUrDg
    ↪ MCGgUAMAsGCSqGSIb3DQEHAaCCA9cwggPTMIICu6ADAgECAgEEMA0GCSqGSIb3DQ
    ↪ EBCwUAMHoxEzARBgoJkiaJk/IsZAEZFgNjb20xGTAXBgoJkiaJk/IsZAEZFglzdG
    ↪ VwdG9vbHMxFzAVBgNV...')
ENDSEC;
END-ISO-10303-21;
```

Using the distributed-ledger method is also recommended in the use cases to secure and authenticate product data due to tampering resistance. Distributed ledgers are ideal candidates to record and secure data exchanges. The set of data-exchange metadata necessary to identify transactions and product data should be included using the recommendations from Krima et al. when the distributed-ledger method is recommended in the use cases [25].

5.2.2 Use Case 1: Communication of Design Information

The first use case addresses communicating design information from a data owner to a data receiver. A use-case diagram is provided in Fig. 5. A *Data Owner* is typically an organization comprising one or more roles (e.g., Designer, Checker, Approver). A *Data Receiver* is also typically an organization comprising one or more roles (e.g., Manufacturing Engineer, Quality Engineer, regulatory agency). All roles are considered stakeholders. The *Data Owner* needs the ability to generate signatures, embed metadata and signatures in files, and register those files on a distributed ledger. The *Data Receiver* needs the ability to check files against a distributed ledger, read the embedded metadata and signatures in files, and

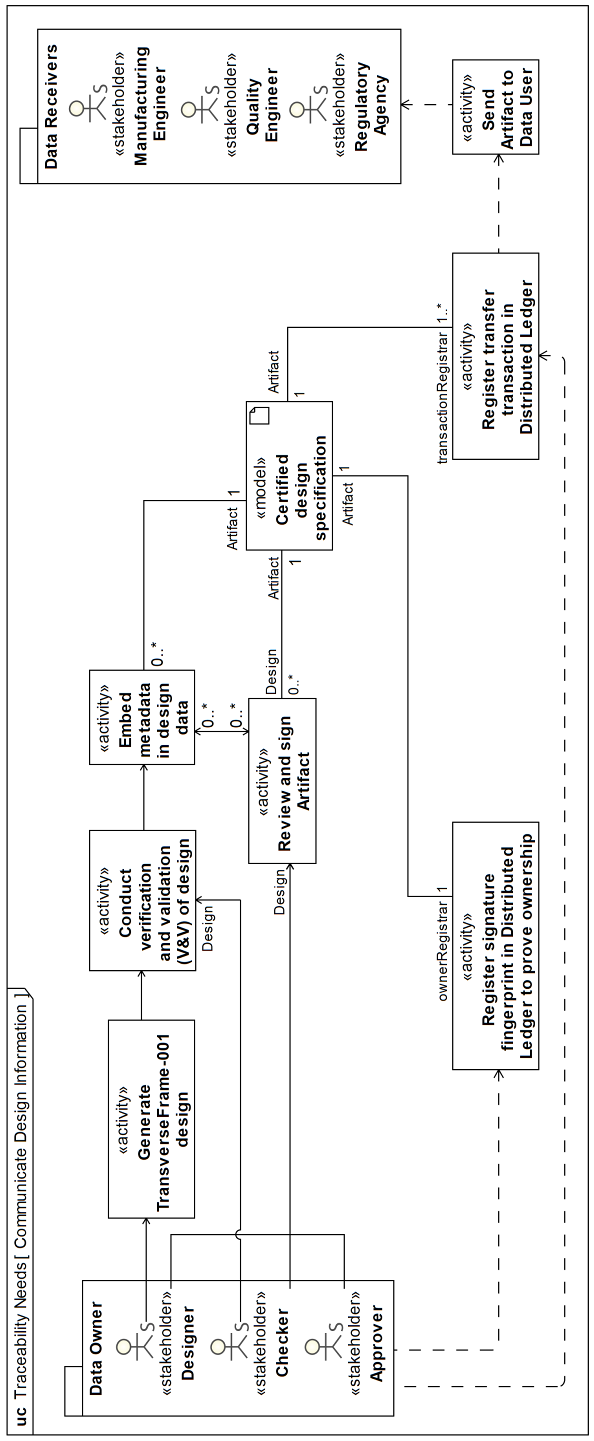
uc Traceability Needs [Communicate Design Information]
Data Owner
«stakeholder» Designer
«stakeholder» Checker
«stakeholder» Approver
«activity» Generate TransverseFrame-001 design
«activity» Conduct verification and validation (V&V) of design
Design
«activity» Embed metadata in design data
0..*
0..*
«activity» Review and sign Artifact
Design
Design
0..*
0..*
Artifact 1
«model» Certified design specification
Artifact 1
Artifact 1
Artifact 1
«activity» Register signature fingerprint in Distributed Ledger to prove ownership
ownerRegistrar 1
«activity» Register transfer transaction in Distributed Ledger
transactionRegistrar 1..*
«activity» Send Artifact to Data User
Data Receivers
«stakeholder» Manufacturing Engineer
«stakeholder» Quality Engineer
«stakeholder» Regulatory Agency

Fig. 5. Use case diagram for communicating design information from the data owner to the data receiver. The data receiver may be one or a combination of many roles (e.g., manufacturer, quality control, regulatory agency).

verify signatures.

Specific to the use case of communicating design information, after the Designer role generates the design (e.g., TransverseFrame-001), a verification and validation (V&V) of the design should be completed by the Checker role. The V&V could be a manual check if the design is a drawing or an automated check if the design is a CAD model. A URI for the results of the V&V should be included in the metadata of the digital signature embedded in the design file. The recommended minimum set of attributes in Table 1 must be included in the signature-trace metadata. The `source` attribute should include the URI for the certified design specification. The `destination` attribute may be omitted or include the same URI as the `soruce` attribute. In addition, the extended attributes provided in Table 2 should be added to the signature-trace metadata. Line 13 in Listing 4 provides an example of the signature-trace metadata for a V&V activity.

After the V&V completes, the design file should be signed by the appropriate roles. In this use case, only the Checker and Approver roles sign the design file. The Checker role signs first to attest the results of the V&V activity (see Line 12-16 of Listing 4) and then the Approver role signs to approve the release of the design data and vouch for the V&V activity (see Line 17-21 of Listing 4). An organization should use its configuration management (CM) process to decide what roles sign the design file and who vouches for whom. After all the appropriate signers provide their digital signatures, then the design should be considered a "certified design specification" and that artifact should be registered in a distributed ledger and shared with the *Data Receivers*.

Two types of transactions should be registered in a distributed ledger. The first transaction for any design should be an *ownership* transaction where the *Data Owner* generates a transaction block in the ledger using the `RecordOwnership` business rule defined in Ref. [25]. `RecordOwnership` business rule is used in transactions to claim ownership of a digital asset (e.g., a design). If any types of transactions are registered for a digital asset before the record of ownership, then the digital asset should be considered compromised.

After the *Data Owner* records ownership of the design in a distributed ledger, then *data exchange* transactions can be registered in the ledger using the `SendProductData` business rule defined in Ref. [25]. A data-exchange transaction can only be registered in a ledger by the entity who previously recorded ownership of the digital asset. The `SendProductData` business rule is used in transactions to capture the exchange of data between a sender and a receiver. The sender can also give a receiver permission share the data with others. If the receiver does not initially receive the permission to the share the data, then only the receive will have access to use the data. The *Data Owner* should register a data-exchange transaction every time data is sent to a *Data Receiver*.

5.2.3 Use Case 2: Generation of Manufacturing Programs

The second use case addresses generating manufacturing process plans using a certified design specification from the first use case (see Sec. 5.2.2) and sending process plans to a machine operator. A use-case diagram is provided in Fig. 6. The stakeholders for this

Table 2. Recommended traceability attribute extensions to include in signature-trace metadata for V&V activities.

Element	Description	Type
results	provides the overall results of the V&V activity (e.g., ’pass’, ’pass with warnings’, ’fail’)	string
report	identifies the location of the report that describes the results of the V&V activity	URI per RFC 7320 [38]

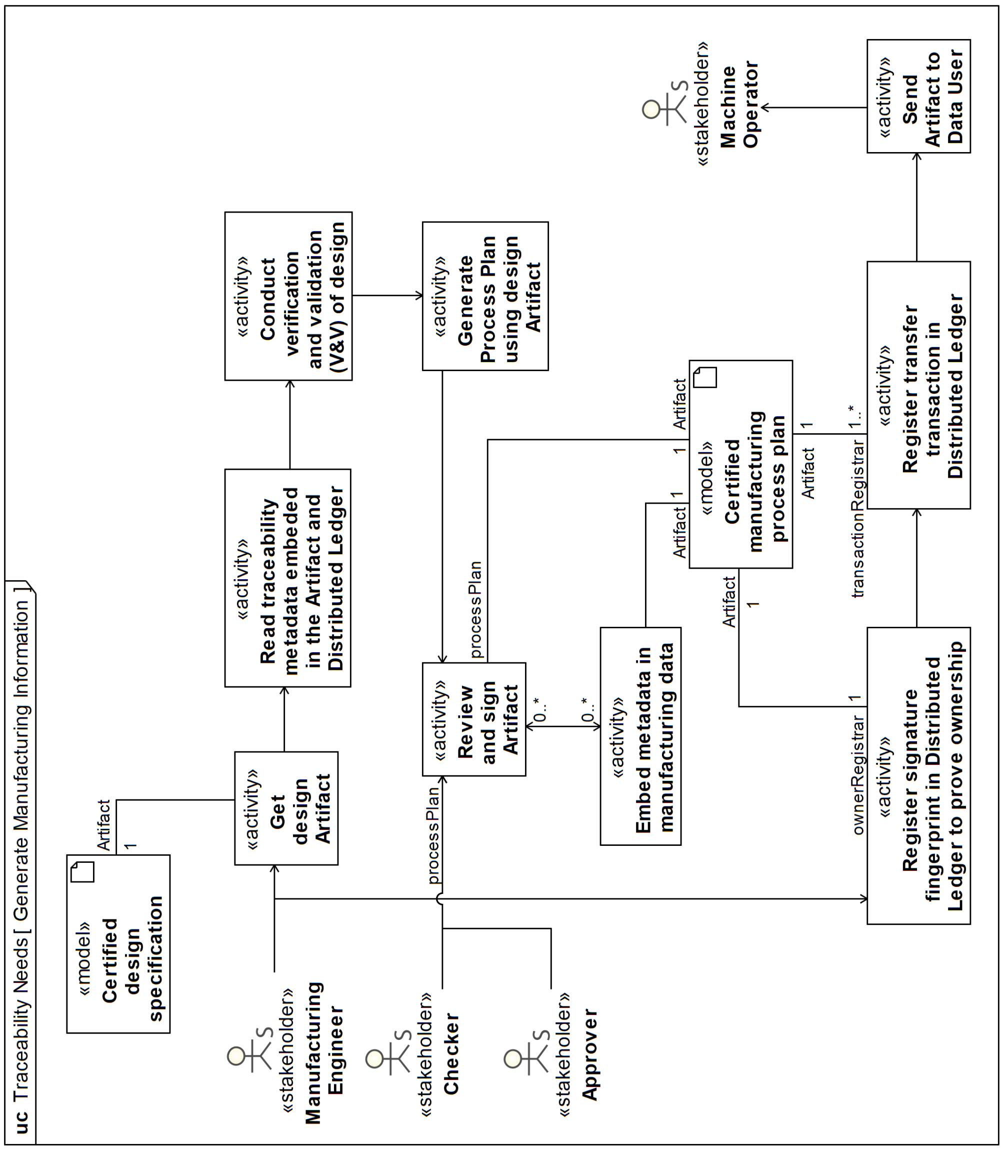
uc Traceability Needs [Generate Manufacturing Information]
«model» Certified design specification
Artifact
1
«stakeholder» Manufacturing Engineer
«stakeholder» Checker
«stakeholder» Approver
«activity» Get design Artifact
«activity» Read traceability metadata embeded in the Artifact and Distributed Ledger
«activity» Conduct verification and validation (V&V) of design
«activity» Generate Process Plan using design Artifact
«activity» Review and sign Artifact
processPlan
processPlan
0..*
0..*
«activity» Embed metadata in manufacturing data
«model» Certified manufacturing process plan
Artifact
1
1
Artifact
1
Artifact
1
Artifact
1
1..*
transactionRegistrar
ownerRegistrar
1
«activity» Register signature fingerprint in Distributed Ledger to prove ownership
«activity» Register transfer transaction in Distributed Ledger
«activity» Send Artifact to Data User
«stakeholder» Machine Operator

Fig. 6. Use case diagram for generating manufacturing information through consuming design information.

use case are the Manufacturing-Engineer, Checker, Approver, and Machine-Operator roles. The use case starts with the Manufacturing-Engineer role receiving or retrieving the certified design specification artifact, validating the artifact against a distributed ledger to ensure the artifact is valid, and then reading the traceability information embedded in the design file.

After the Manufacturing Engineer confirms the design artifact is valid, then he/she can generate a process plan using typical manufacturing capabilities. If the manufacturing plan is for a discrete subtractive process, then the plan will most likely be in the form of a G-Code file per ISO 6983 [11]. If the manufacturing plan is for a discrete additive process, then the plan will mostly likely be in the form of a STL file or an AMF file. Unfortunately, there are no standardized methods for embedding digital signatures into G-Code and STL files. However, Ref. [2] provides some guidance for working with G-Code. We recommend AMF files always be used when possible for additive processes because standard-conforming digital signatures can be included using the XML schema described Sec. 5.1.2.

When the process plan is complete, the process-plan file should be signed by the appropriate roles. In this use case, only the Checker and Approver roles sign the file. The Checker role signs first to attest the process plan meets the design requirements and then the Approver role signs to approve the release of the manufacturing data. An organization should use its CM process to decide what roles sign the process-plan file and who vouches for whom. The recommended minimum set of attributes in Table 1 must be included in the signature-trace metadata for each signature. The `source` attribute should include the URI for the certified design specification. The `destination` attribute should include the URI for the process plan. After all the appropriate signers provide their digital signatures, then the design should be considered a "certified manufacturing process plan" and that artifact should be registered in a distributed ledger and shared with the Machine-Operator role.

Like in the first use case, two types of transactions should be registered in a distributed ledger. The first transaction for the process plan should be an *ownership* transaction where the manufacturing organization generates a transaction block in the ledger using the `RecordOwnership` business rule defined in Ref. [25]. If any types of transactions are registered for a digital asset (e.g., process plan) before the record of ownership, then the digital asset should be considered compromised.

After the manufacturing organization records ownership of the process plan in a distributed ledger, then *data exchange* transactions can be registered in the ledger using the `SendProductData` business rule defined in Ref. [25]. The manufacturing organization should register a data-exchange transaction, including a *job ID* in the metadata, every time the process plan is sent to a Machine Operator, which ensures traceability for each job sent to a shop floor. We recommend that the manufacturing organization not give a Machine Operator permission in the distributed ledger to share the data with others.

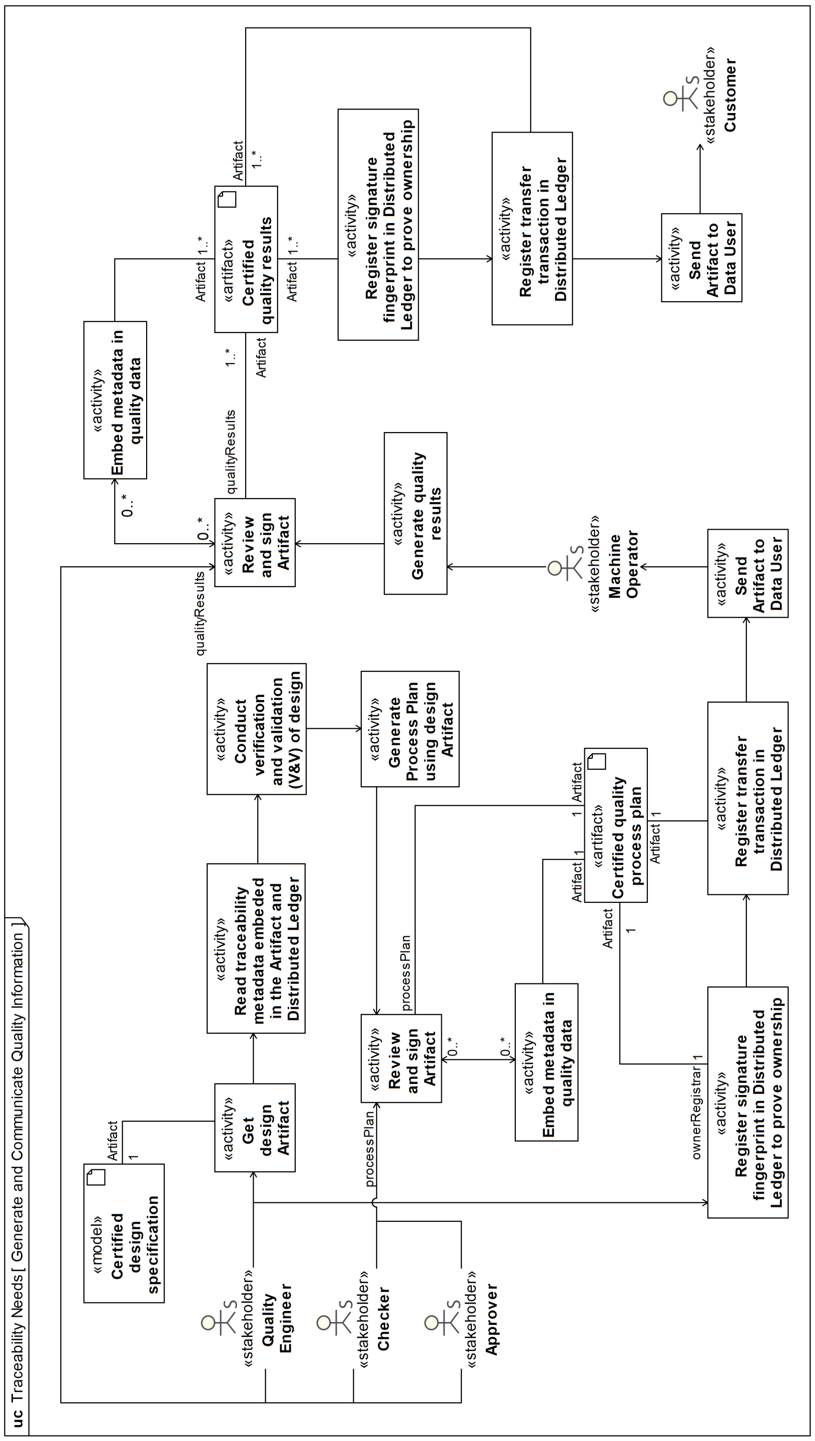
uc Traceability Needs [Generate and Communicate Quality Information]
«stakeholder» Quality Engineer
«stakeholder» Checker
«stakeholder» Approver
«model» Certified design specification
Artifact 1
«activity» Get design Artifact
«activity» Read traceability metadata embeded in the Artifact and Distributed Ledger
«activity» Conduct verification and validation (V&V) of design
«activity» Generate Process Plan using design Artifact
processPlan
«activity» Review and sign Artifact
0..*
«activity» Embed metadata in quality data
«artifact» Certified quality process plan
Artifact 1
ownerRegistrar 1
«activity» Register signature fingerprint in Distributed Ledger to prove ownership
«activity» Register transfer transaction in Distributed Ledger
«activity» Send Artifact to Data User
«stakeholder» Machine Operator
«activity» Generate quality results
qualityResults
«activity» Review and sign Artifact
«activity» Embed metadata in quality data
«artifact» Certified quality results
Artifact 1..*
«activity» Register signature fingerprint in Distributed Ledger to prove ownership
«activity» Register transfer transaction in Distributed Ledger
«activity» Send Artifact to Data User
«stakeholder» Customer

Fig. 7. Use case diagram for generating and communicating quality information through consuming design information.

5.2.4 Use Case 3: Generation and Communication of Inspection Data

The third use case addresses generating and communicating inspection data using a certified design specification from the first use case (see Sec. 5.2.2), sending the inspection plan to a Machine Operator, and returning inspection results to a Customer. A use-case diagram is provided in Fig. 7. The stakeholders for this use case are the Quality-Engineer, Checker, Approver, Machine-Operator, and Customer roles. The use case starts with the Quality-Engineer role receiving or retrieving the certified design specification artifact, validating the artifact against a distributed ledger to ensure the artifact is valid, and then reading the traceability information embedded in the design file.

After the Quality Engineer confirms the design artifact is valid, then he/she can generate an inspection plan using typical inspection capabilities. We recommend the QIF standard always be used when possible for inspection and quality data because standard-conforming digital signatures can be included using the XML schema described Sec. 5.1.2. Minimally, QIF *Plans* should be used for exchanging and communicating inspection plans and QIF *Results* should be used for exchanging and communicating inspection results.

When the inspection plan is complete, the QIF file should be signed by the appropriate roles. In this use case, only the Checker and Approver roles sign the QIF file. The Checker role signs first to attest the inspection plan meets the design requirements and then the Approver role signs to approve the release of the inspection plan. An organization should use its CM process to decide what roles sign the QIF file and who vouches for whom. The recommended minimum set of attributes in Table 1 must be included in the signature-trace metadata for each signature. The `source` attribute should include the URI for the certified design specification. The `destination` attribute should include the URI for the QIF file that contains the inspection plan. After all the appropriate signers provide their digital signatures, then the design should be considered a "certified quality process plan" and that artifact should be registered in a distributed ledger and shared with the Machine-Operator role.

Like in the second use case, two types of transactions should be registered in a distributed ledger. The first transaction for the inspection plan should be an *ownership* transaction where the quality organization generates a transaction block in the ledger using the `RecordOwnership` business rule defined in Ref. [25]. If any types of transactions are registered for a digital asset (e.g., inspection plan) before the record of ownership, then the digital asset should be considered compromised.

After the quality organization records ownership of the inspection plan in a distributed ledger, the *data exchange* transactions can be registered in the ledger using the `SendProductData` business rule defined in Ref. [25]. The quality organization should register a data-exchange transaction, including a *job ID* in the metadata, every time the inspection plan is sent to a Machine Operator, which ensures traceability for each job sent to the inspection department. We recommend that the quality organization not give a Machine Operator permission in the distributed ledger to share the data with others.

The Machine Operator will use the certified quality process plan to generate the inspection results. The Machine Operator should verify the QIF file including the inspection plan when he/she receives the QIF file. The same verification process used for verifying the certified design specification should also be used for the verifying the QIF file. After the Machine Operator generates a QIF file that includes the inspection results, the Quality Engineer, Checker, and Approver will review and sign the QIF file.

The Quality-Engineer and Checker roles sign first to attest the inspection results meets the design and inspection-plan requirements. The Quality Engineer and Checker do not need to cross-sign each other's signature. Then, the Approver role signs to approve the release of the inspection results. Like the inspection plan, an organization should use its CM process to decide what roles sign the QIF file and who vouches for whom. The recommended minimum set of attributes in Table 1 must be included in the signature-trace metadata for each signature. The `source` attribute should include the URI for the certified quality process plan. The `destination` attribute should include the URI for the QIF file that contains the inspection results. After all the appropriate signers provide their digital signatures, then the design should be considered a "certified quality results" and that artifact should be registered in a distributed ledger and shared with the Customer. The *ownership* and *data exchange* transactions for the inspection results should be registered in the distributed ledger in the same way as the inspection plan.

6. Conclusion

The objective of this report is to provide guidance for deploying a system, or set of systems, for ensuring traceability and trustworthiness of manufacturing-related data. These recommendations provide practical how-to guidance for ensuring traceability and trustworthiness of manufacturing-related data for both subtractive and additive manufacturing processes.

This report reviews manufacturing-related data formats conforming to international, open, consensus-based standards. The majority of those standards require capturing and exchanging data conforming to information models based on either EXPRESS or XML. This report also provides a trades analysis of three traceability implementation options. Based upon the available capabilities codified in standards and technologies, this report recommends schema extensions for capturing traceability information using EXPRESS and XML information models. This report also provides recommendations to address three common use cases that address manufacturing-related data traceability and trustworthiness needs across the product lifecycle.

While this report covers a majority of the manufacturing-related data and processes deployed in industry, gaps do remain. Industry lacks a consensus-based standard that provides a traceability method to deal with streaming data. Industry also widely uses two popular, out-dated data formats: G-code and STL. Both data formats are based on old technologies and do not support modern information-modeling methods. There are no methods for capturing digital signatures and signature-trace metadata in the G-code standard and STL specification.

The lack of traceability support in G-code is a minor issue because the manufacturing process plan that produces the G-code file can still be digitally signed. But the lack of traceability support for streaming data and STL are more significant issues because of the growing deployment of additive-manufacturing processes across industry. However, the gaps that remain account only for a small portion of the complete product lifecycle. Therefore, following the recommendations of this report provides industry with a strong foundation for deploying ubiquitous traceability and trustworthiness of manufacturing-related data.

Acknowledgments

The authors wish to thank Vijay Srinivasan from NIST and Paul Huang from the Office of Naval Research Laboratory for their comments and input to this report. The authors prepared this report as partial fulfillment to the requirements covered by the reimbursable agreement N0001418IP00053 between the National Institute of Standards and Technology and the Office of Naval Research Laboratory.

References

[1] Telecommunication Standardization Sector of ITU (2004) Security in telecommunications and information technology (Geneva),

[2] Hedberg Jr T, Krima S, Camelio JA (2016) Embedding X.509 digital certificates in three-dimensional models for authentication, authorization, and traceability of product data. *Journal of Computing and Information Science in Engineering* 17(1):11008–11011. https://doi.org/10.1115/1.4034131

[3] International Standards Organization (2014) Industrial automation systems and integration – product data representation and exchange – part 242: Application protocol: Managed model-based 3D engineering.

[4] Barnard Feeney A, Frechette SP, Srinivasan V (2015) A portrait of an ISO STEP tolerancing standard as an enabler of smart manufacturing systems. *Journal of Computing and Information Science in Engineering* 15(2):21001. https://doi.org/10.1115/1.4029050

[5] Trainer A, Hedberg Jr T, Barnard Feeney A, Fischer K, Rosche P (2016) Gaps analysis of integrating product design, manufacturing, and quality data in the supply chain using model-based defintion. *ASME 2016 11th International Manufacturing Science and Engineering Conference – Volume 2: Materials; Biomanufacturing; Properties, Applications and Systems; Sustainable Manufacturing* (ASME), Vol. 2, p V002T05A003. https://doi.org/10.1115/MSEC2016-8792

[6] Hedberg Jr T, Lubell J, Fischer L, Maggiano L, Barnard Feeney A (2016) Testing the digital thread in support of model-based manufacturing and inspection. *Journal of Computing and Information Science in Engineering* 16(2):021001. https://doi.org/10.1115/1.4032697

[7] Fischer A, Arthurs G (2015) INCOSE 2015 MBSE workshop breakout session. Available at http://www.omgwiki.org/MBSE/doku.php?id=mbse:incose_mbse_iw_2015:breakout_out_session_model_lifecylce_mgmt.

[8] International Standards Organization (2004) Industrial automation systems and integration – product data representation and exchange – part 11: Implementation methods: The express language reference manual.

[9] International Standards Organization (2016) Industrial automation systems and integration – product data representation and exchange – part 21: Implementation methods: Clear text encoding of the exchange structure.

[10] International Standards Organization, American Society for Testing and Materials (2016) Standard specification for additive manufacturing file format (AMF) version 1.2.

[11] International Standards Organization (2009) Automation systems and integration – numerical control of machines – program format and definitions of address words – part 1: Data format for positioning, line motion and contouring control systems.

[12] Suh SH (2008) *Theory and design of CNC systems* (Springer, London), .

[13] MTConnect Institute (2018) MTConnect standard version 1.4.0 (ANSI/MTC1.4-

2018). Available at https://www.mtconnect.org/s/ANSI_MTC1_4-2018.pdf.

[14] Dimensional Metrology Standards Consortium (2018) QIF standard version 3.0 (ANSI/DMSC QIF 3.0). Available at https://qifstandards.org/qif-download.

[15] International Standards Organization (2008) Document management – portable document format – part 1: PDF 1.7.

[16] International Standards Organization (2014) Document management – 3D use of product representation compact (PRC) format – part 1: PRC 10001.

[17] Telecommunication Standardization Sector of ITU (2017) Information technology – open systems interconnection – the directory – part 8: Public-key and attribute certificate frameworks. Available at http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=64854.

[18] The Internet Engineering Task Force (2013) Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile. Available at https://datatracker.ietf.org/doc/rfc5280/.

[19] The Internet Engineering Task Force (2013) An internet attribute certificate profile for authorization. Available at https://datatracker.ietf.org/doc/rfc5755/?include_text=1.

[20] Organization for the Advancement of Structured Information Standards (2005) Assertions and protocols for the oasis security assertion markup language (SAML) v2.0.

[21] Organization for the Advancement of Structured Information Standards (2015) About us. Available at https://www.oasis-open.org/org.

[22] Krima S, Hedberg Jr T (2016) Digital manufacturing certificate toolkit: Adding trust and traceability to product data. *NIST Journal of Research (NIST JRES)* 121. https://doi.org/10.6028/jres.121.027

[23] Hedberg Jr T, Krima S, Camelio JA (2019) Method for enabling a root of trust in support of product data certification and traceability. *Journal of Computing and Information Science in Engineering* 19(4):041003. https://doi.org/10.1115/1.4042839

[24] Yaga D, Mell P, Roby N, Scarfone K (2018) Blockchain technology overview (National Institute of Standards and Technology, Gaithersburg, MD), https://doi.org/10.6028/NIST.IR.8202

[25] Krima S, Hedberg Jr T, Barnard Feeney A (2019) Securing the digital threat for smart manufacturing: (National Institute of Standards and Technology, Gaithersburg, MD), AMS 300-6. https://doi.org/10.6028/NIST.AMS.300-6

[26] Wu D, Greer MJ, Rosen DW, Schaefer D (2013) Cloud manufacturing: Strategic vision and state-of-the-art. *Journal of Manufacturing Systems* 32(4):564–579. https://doi.org/10.1016/j.jmsy.2013.04.008

[27] Vincent Wang X, Xu XW (2013) An interoperable solution for cloud manufacturing. *Robotics and Computer-Integrated Manufacturing* 29(4):232–247. https://doi.org/10.1016/j.rcim.2013.01.005

[28] Zhang L, Luo Y, Tao F, Li BH, Ren L, Zhang X, Guo H, Cheng Y, Hu A, Liu Y (2014) Cloud manufacturing: a new manufacturing paradigm. *Enterprise Information Systems* 8(2):167–187. https://doi.org/10.1080/17517575.2012.683812

[29] Ren L, Zhang L, Wang L, Tao F, Chai X (2017) Cloud manufacturing: key character-

istics and applications. *International Journal of Computer Integrated Manufacturing* 30(6):501–515. https://doi.org/10.1080/0951192X.2014.902105

[30] Wirth N (1977) What can we do about the unnecessary diversity of notation for syntactic definitions? *Commun ACM* 20(11):822–823. https://doi.org/10.1145/359863.359883

[31] Bartel M, Boyer J, Fox B, LaMacchia B, Simon E (2015) XML signature syntax and processing version 2.0. Available at https://www.w3.org/TR/xmldsig-core2/.

[32] Bajaj M, Hedberg Jr T (2018) System lifecycle handler - spinning a digital thread for manufacturing. *28th Annual INCOSE International Symposium* (International Council of Systems Engineering, Washington, DC), Vol. 28, pp 1636–1650. https://doi.org/10.1002/j.2334-5837.2018.00573.x

[33] Y14 Standards Committee (2019) Model organization practices (American Society of Mechanical Engineers, New York), Standard Y14.47.

[34] International Standards Organization (2018) ISO 8601 date and time format. Available at https://www.iso.org/iso-8601-date-and-time-format.html.

[35] International Standards Organization (2015) Information technology – metadata registries (MDR) – part 1: Framework. Available at http://metadata-standards.org/11179/ https://www.iso.org/standard/61932.html.

[36] Group OA (2018) Oagis 10.4. Available at https://oagi.org/DownloadsResources/tabid/143/Default.aspx.

[37] Ivezic N, Kulvatunyou B, Srinivasan V (2014) On architecting and composing through-life engineering information services to enable smart manufacturing. *Procedia CIRP* 22(1):45–52. https://doi.org/10.1016/j.procir.2014.07.004

[38] Nottingham M (2014) URI design and ownership, . https://doi.org/10.17487/rfc7320. Available at https://www.rfc-editor.org/info/rfc7320

Appendix A: Syntax Block and Schema Definition Code

This appendix contains the raw code for each of syntax block and/or schema definition defined in this guide.

Listing 5. Wirth Syntax Notation (WSN) of the ISO 10303 Part 21 exchange structure

```
EXCHANGE_FILE = "ISO-10303-21;"
                   HEADER_SECTION [ ANCHOR_SECTION ]
                   [ REFERENCE_SECTION ] { DATA_SECTION }
                   "END-ISO-10303-21;" { SIGNATURE_SECTION }.

HEADER_SECTION = "HEADER;"
                   HEADER_ENTITY HEADER_ENTITY HEADER_ENTITY
                   [HEADER_ENTITY_LIST]
                   "ENDSEC;" .
HEADER_ENTITY_LIST = HEADER_ENTITY { HEADER_ENTITY } .
HEADER_ENTITY = KEYWORD "(" [ PARAMETER_LIST ] ")" ";" .

PARAMETER_LIST = PARAMETER { "," PARAMETER } .
PARAMETER = TYPED_PARAMETER |
                   UNTYPED_PARAMETER | OMITTED_PARAMETER .
TYPED_PARAMETER = KEYWORD "(" PARAMETER ")" .
UNTYPED_PARAMETER = "$" | INTEGER | REAL | STRING | RHS_OCCURENCE_NAME
                   | ENUMERATION | BINARY | LIST .
OMITTED_PARAMETER = "*" .
LIST = "(" [ PARAMETER { "," PARAMETER } ] ")" .

ANCHOR_SECTION = "ANCHOR;" ANCHOR_LIST "ENDSEC;" .
ANCHOR_LIST = { ANCHOR } .
ANCHOR = ANCHOR_NAME "=" ANCHOR_ITEM { ANCHOR_TAG } ";" .
ANCHOR_ITEM = "$" | INTEGER | REAL | STRING | ENUMERATION | BINARY
                   | RHS_OCCURRENCE_NAME | RESOURCE |
                      ↪ ANCHOR_ITEM_LIST .
ANCHOR_ITEM_LIST = "(" [ ANCHOR_ITEM { "," ANCHOR_ITEM } ] ")" .
ANCHOR_TAG = "{" TAG_NAME ":" ANCHOR_ITEM "}" .

REFERENCE_SECTION = "REFERENCE;" REFERENCE_LIST "ENDSEC;" .
REFERENCE_LIST = { REFERENCE } .
REFERENCE = LHS_OCCURRENCE_NAME "=" RESOURCE ";" .

DATA_SECTION = "DATA" [ "(" PARAMETER_LIST ")" ] ";"
                   ENTITY_INSTANCE_LIST "ENDSEC;" .
ENTITY_INSTANCE_LIST = { ENTITY_INSTANCE } .
ENTITY_INSTANCE = SIMPLE_ENTITY_INSTANCE | COMPLEX_ENTITY_INSTANCE .
SIMPLE_ENTITY_INSTANCE = ENTITY_INSTANCE_NAME "=" SIMPLE_RECORD ";" .
COMPLEX_ENTITY_INSTANCE = ENTITY_INSTANCE_NAME "=" SUBSUPER_RECORD
```

```
       ↪ ";" .
SIMPLE_RECORD = KEYWORD "(" [ PARAMETER_LIST ] ")" .
SUBSUPER_RECORD = "(" SIMPLE_RECORD_LIST ")" .
SIMPLE_RECORD_LIST = SIMPLE_RECORD { SIMPLE_RECORD } .

SIGNATURE_SECTION = "SIGNATURE" SIGNATURE_CONTENT "ENDSEC;".
```

Listing 6. Extensible Markup Language (XML) Schema Definition (XSD)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ elementFormDefault="qualified"
    targetNamespace="signatureTrace" xmlns="signatureTrace"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
        schemaLocation="https://www.w3.org/TR/2002/REC-xmldsig-core-2002
            ↪ 0212/xmldsig-core-schema"/>
    <xs:complexType name="TraceType">
        <xs:sequence>
            <xs:element name="strTraceMetadata" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="PathType">
        <xs:sequence>
            <xs:element maxOccurs="unbounded" minOccurs="1"
                ↪ name="lstIndex" type="xs:string">
                <xs:annotation>
                    <xs:documentation>* (all) or list of IDs of the
                        ↪ indexes being signed</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="bolCrossSign" type="xs:boolean">
                <xs:annotation>
                    <xs:documentation>binary (Y/N) switch to identify
                        ↪ single- or multi-path
                        ↪ signing</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element maxOccurs="1" minOccurs="0" name="lstTraceID"
                ↪ type="xs:IDREFS">
                <xs:annotation>
                    <xs:documentation>empty or list of PKCS entries
                        ↪ signed using multi-path</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="SigTrace" type="TraceType"/>
    <xs:element name="SigPath" type="PathType"/>
    <xs:complexType name="TraceBlockType">
        <xs:sequence>
            <xs:element ref="SigTrace"/>
            <xs:element ref="SigPath"/>
            <xs:element ref="ds:Signature"/>
        </xs:sequence>
        <xs:attribute name="traceBlockID" type="xs:ID"/>
    </xs:complexType>
    <xs:element name="Trace" type="TraceBlockType"/>
</xs:schema>
```

Appendix B: Acronyms

3D three-dimensional. 2, 7, 10

AMF Additive Manufacturing File Format. 3, 4, 15, 24

ANSI American National Standards Institute. 2, 6, 10

AP Application Protocol. 2

ASME American Society of Mechanical Engineers. 18

ASTM American Society for Testing and Materials. 3

BREP boundary representation. 7

CAD computer-aided design. iii, 2–4, 7, 11, 13, 22

CAI computer-aided inspection. 2

CAM computer-aided manufacturing. 2–4

CAx computer-aided technologies. 3

CM configuration management. 22, 24, 26, 27

CMS coordinate-measurement system. 2

CNC computer numerical control. 1, 4

DMC digital manufacturing certificate. 8, 11, 12

DMSC Dimensional Metrology Standards Consortium. 6, 10

FRP fiber-reinforced plastic. 1

GD&T geometric dimensions and tolerances. 2

HTTP Hypertext Transfer Protocol. 5

IP intellectual property. 13

ISO International Standards Organization. 3

ITU-T Telecommunication Standardization Sector of the International Telecommunication Union. 7

JSON JavaScript Object Notation. 15, 16, 18

LOTAR long-term archival and retrieval. 2

MBD model-based definition. 2, 6

MBE model-based enterprise. 2

MBM model-based manufacturing. 2

MQTT Message Queuing Telemetry Transport. 6

NC numerical control. 4

NIST National Institute of Standards and Technology. 8

OAGIS Open Applications Group Integration Specification. 19

OASIS Organization for the Advancement of Structured Information Standards. 8

OEM original equipment manufacturer. 3

OPC-UA Open Platform Communications United Architecture. 6

PDF Portable Document Format. 7, 10

PMI product and manufacturing information. 2

PRC Product Representation Compact. 7, 10

QIF Quality Information Framework. 6, 7, 10, 15, 16, 26, 27

REST Representational State Transfer. 5

S/MIME Secure/Multipurpose Internet Mail Extensions. 8

SAML Security Assertion Markup Language. 8

SDO standards development organization. 2

SOA service-oriented architecture. 8

SSL Secure Sockets Layer. 8

STEP STandard for the Exchange of Product Model Data. iii, 2, 3, 15, 18, 19

STEP AP242 STandard for the Exchange of Product Model Data Application Protocol 242. 2, 3

STL Stereolithography. 3, 24, 27, 28

TDP technical data package. 7

TLS Transport Layer Security. 8

URI Uniform Resource Identifier. 18, 19, 22–24, 26, 27

UUID universally unique identifier. 15

V&V verification and validation. iii, 22, 23

W3C World Wide Web Consortium. 16

WSN Wirth Syntax Notation. 14

X.509-PKI Public Key Infrastructure. 8, 11, 12

X.509-PMI Privilege Management Infrastructure. 8

XML Extensible Markup Language. 3, 5–7, 14–18, 24, 26, 27

XMP Extensible Metadata Platform. 7

XSD XML Schema Definitions. 3, 6, 15, 16