This is an Accepted Manuscript of an article published online by ASME in the Journal of Computing and Information Science in Engineering on 19 SEP 2019, available online: https://doi.org/10.1115/1.4044921

Using graphs to link data across the product lifecycle for enabling smart manufacturing digital threads

Thomas D. Hedberg, Jr.[*]
National Institute of Standards and Technology
Gaithersburg, Maryland 20899

Manas Bajaj
Intercax
Atlanta, Georgia 30346

Jaime A. Camelio
Grado Department of Industrial and Systems Engineering
Virginia Tech
Blacksburg, Virginia 24061

ABSTRACT

Smart manufacturing promises to provide significant increases in productivity and effectiveness of manufacturing systems by better connecting the data from people, processes, and things. However, there is no uniform, generalized method for deploying linked-data concepts to the manufacturing domain. The literature describes and commercial vendors offer centralized data repository solutions, but these types of approaches quickly breakdown under the intense burden of managing and reconciling all the data flowing in and out of the various repositories across the product lifecycle. In this paper, we introduce a method for linking and tracing data throughout the product lifecycle using graphs to form digital threads. We describe a prototype implementation of the method and a case study to demonstrate an information round-trip for a product assembly between the design, manufacturing, and quality domains of the product lifecycle. The expected impact from this novel, standards-based, linked-data method is the ability to use digital threads to provide data, system, and viewpoint interoperability in the deployment of smart manufacturing to realize industry's $30 Billion annual opportunity.

digital thread, model-based enterprise (MBE), linked-data, manufacturing graphs

Acronyms
3D three-dimension. 5

ALM application lifecycle management. 7
API application programming interface. 12, 19, 21, 23, 30
ASME American Society of Mechanical Engineers. 15

BOM bill of materials. 13

CAD computer-aided design. 4, 5, 11, 13, 15, 17, 18, 29
CAI computer-aided inspection. 4
CAM computer-aided manufacturing. 4, 8, 11
CM configuration management. 12
cUAV configurable unmanned aerial vehicle. 14–16, 29

DOI Distributed Object Identifier. 9, 24

[*]Corresponding Author, tdh1@nist.gov

ERP enterprise resource planning. 4

FAIR first article inspection reporting. 15, 16

GID global identifier. 11–14, 18

HTTP Hypertext Transfer Protocol. 13

I/O in-out. 5
INCOSE International Council on Systems Engineering. 15, 16
IP intellectual property. 9, 24
IT information technology. 12

JSON JavaScript Object Notation. 12, 18, 21, 30
JT Jupiter Tesselation. 4

LHS Lifecycle Handler System. 9, 11–14, 23, 24, 29
LIFT Lifecycle Information Framework and Technology. 5, 10, 29

MBSE Model-Based Systems Engineering. 15, 16
MES manufacturing execution system. 4
MTC Manufacturing Technology Centre. 15, 16

NC numerical control. 15
NIST National Institute of Standards and Technology. 15, 16

OSLC Open Services for Lifecycle Collaboration. 4

PDM product data management. 4, 8, 13, 14
PLCS Product Life Cycle Support. 4
PLM product lifecycle management. 6, 7, 29

QIF Quality Information Framework. 8, 15–17
QMS Quality Management System. 4, 8

REST Representational State Transfer. 13, 19, 21, 30
RII receiving and incoming inspection. 15–17, 20, 30

STEP Standard for the Exchange of Product Data. 4
SysML Systems Modeling Language. 4, 15

TDP technical data package. 5

UML Unified Modeling Language. 4
URI Uniform Resource Identifier. 12–14

XML Extensible Markup Language. 15, 17

1 Introduction

Between 1998 and 2015, U.S. manufacturing productivity grew three times faster than the service economy [1]. While manufacturing exhibited growth and success, significant opportunity remains. For design through production portion of the product lifecycle, one study found that simply transitioning from paper-based processes to (digital) model-based processes would achieve an approximate 75 percent reduction in cycle-time [2]. Further, enhanced sensing and monitoring, seamless transmission of digital information, and advances in analyzing data and trends would save manufacturers $30 Billion annually [3]. This paper contributes a standards-based, linked-data approach that would help manufacturers realize the significant savings.

Industry is approaching the fundamental limits to the amount of data its people, tools, and processes can manage. The challenges with managing manufacturing-related data are well understood [4, 5, 6, 7]. Further, data, system, and viewpoint interoperability is an increasing challenge for industry [8, 9, 10, 7, 11]. Industry needs connected systems and linked-data federated across enterprises. Point-to-Point interoperability (e.g., file-based data translation) is no longer enough [12]. Considering the identified challenges and needs, industry should stop thinking about data interoperability through mapping exercises and instead focus on domain and interface interoperability.

Domain interoperability (e.g., design to manufacturing, design to quality) requires a normalized method for accessing and contextualizing data at different points of the product lifecycle. Often the focus of interoperability has been confined to the formats in which the data is stored and not the semantics. Focusing on the information for the "thing" being represented in the data would help industry keep more focus on solving problems for the thing than focusing on communication and data exchange. Further, actors in industry must also consider the interfaces, outputs, and inputs, on the boundaries of their domains. Standard interfaces between domains must be developed and understood to support efficient flow of required information through the product lifecycle. Enabling effective communication of information brings with it an almost $8 Billion return-on-investment annual opportunity [3].

We recognize that context varies based on the phase of the lifecycle (e.g., design, manufacturing, quality). Each phase of the product lifecycle has different viewpoints and concerns, which lead to different levels of abstraction in modeling and simulation [13, 14]. In addition, context varies based on the particular viewpoint interacting with data (e.g., systems, operations, enterprises) [11]. The various viewpoints lead to information models and systems being developed for a specific purpose, which results in different information models across the product lifecycle to look at the same data in different ways. Thus, geometry and manufacturing specification is not enough to define products – behavioral and contextual definitions are required too. Furthermore, all three aspects must be generated, documented, and communicated using an agile and dynamic method.

Traceability is another important aspect to industry as data is generated, used, and linked across the lifecycle, because one must know the provenance of data and/or parts to ensure they are trustworthy [15, 16]. Our standards-based, linked-data approach provides seamless traceability that must be supported between the systems, designs, manufacturing operations, and maintenance of products. Seamless traceability across the product lifecycle, enables high-quality manufacturing, and supports trusted enterprise knowledge reuse. High-quality manufacturing remains a goal of industry because industry wants to make parts faster, cheaper, and better.

Last, enterprise knowledge reuse supports industry's need in retaining and generating knowledge regardless of what human resources are available. People come and go in organizations, but the knowledge must remain. The goals are not achievable without first connecting data across the enterprise to spin a digital thread. Different contextual models can be generated as information moves across disciplines. Also, tracking changes as well as comparing, synchronizing, and repairing connections are topics of interest related to linking data across enterprises. Achieving the goal of a standards-based linked-data approach for distributed, smart manufacturing is the major contribution of the work presented here.

This paper, in Section 2, provides a brief background on previous works. Section 3 describes a methodology for linking and tracing data throughout the product lifecycle. Section 3.1 specifically addresses the information requirements and architecture proposed for making connections across enterprises to form smart manufacturing digital threads. Further, Section 3.2 proposes a method for ensuring persistent global identification. Section 4 presents a case study to demonstrate the method applied to an information round-trip between design, manufacturing, and quality. Before concluding, Section 5 will discuss generating connections dynamically, forming frequently asked queries enabled by the method, contextualizing graph-based viewpoints, and knowledge generation.

2 Background

Various standards and technologies exist for industry to connect and/or integrate data and systems. However, each standard and technology is often built for a specific purpose and may not apply to all viewpoints of the product lifecycle. For instance, Unified Modeling Language (UML) [17] and Systems Modeling Language (SysML) [18] are used for architecture and system modeling. Computer-aided design (CAD), computer-aided manufacturing (CAM), computer-aided inspection (CAI) are used to generate planning and specification models in design, manufacturing, and inspection, respectively. The tools to generate each model vary widely both intra-domain and inter-domain. Standards, such as Standard for the Exchange of Product Data (STEP) [19] and Jupiter Tesselation (JT) [20], enable file-based data exchange between domains, but they have been deployed primarily in limited design contexts.

Then, each product-lifecycle domain has its own type of client-support systems for managing the models built within each phase of the product lifecycle. Examples of these systems are product data management (PDM), manufacturing execution system (MES), enterprise resource planning (ERP), and Quality Management System (QMS). There are multi-million dollar market sectors built around configuring, customizing, and managing these systems. Further, standards such as Open Services for Lifecycle Collaboration (OSLC) [21] and Product Life Cycle Support (PLCS) [22] or non-standard point-to-point integrations all assume the same schema and/or behavior can be used across these systems. This is impractical in today's distributed manufacturing environments. The literature propose and commercial vendors offer centralized data-repository solutions, but these types of approaches quickly breakdown under the intense burden of managing and reconciling all the data flowing in and out of the repositories.

One estimate for the cost of digitally connecting and managing all artifacts in one program across its lifecycle using the tools available today is approximately \$80 to \$180 billion [23]. The reason for the high cost is the tools today do not support effective linked-data and require significant amounts of manual intervention to maintain. Industry needs a capability for linking all the different models and systems in distributed and universal ways to enable rapid data curation, query, discovery, and retrieval. Industry would benefit from the Semantic Web being applied to manufacturing – forming a sort of Engineering and Manufacturing Internet. Further, there is evidence that shows enabling an integrated smart-manufacturing approach could provide industry with a $100 billion annual savings opportunity [3].

2.1 Digital Thread

The digital thread concept shows promise for supporting industry's needs. The digital thread is an integrated information flow that connects all the phases of the product lifecycle using accepted authoritative data sources (e.g., requirements, system architecture, technical data package (TDP), three-dimension (3D) CAD models) [24, 2, 25]. The aim of digital thread is to deploy an integration framework that brings all phases and systems of a product lifecycle together for making efficient and effective measurements of the lifecycle in support of data-driven methods. Specific interests relate to knowledge building, decision support, requirements management, and control. A major goal for enabling the digital thread is linking universally heterogeneous information systems and data sets across the various domains of the product lifecycle (e.g., design, manufacturing, quality) in dynamic ways without requiring one-to-one data mapping. An expected impact of achieving this goal is a significant reduction in the cost of deploying digital thread.

Hedberg Jr et al. [12] proposed a Lifecycle Information Framework and Technology (LIFT) idea to support effective implementation of the digital thread. The framework is comprised of three levels: (1) product lifecycle data, (2) data certification and traceability, and (3) data-driven applications. In general, the LIFT idea strives to link data across the information silos, while building trust through traceability, for driving applications with data. Hedberg Jr et al. [12] also introduced an example schematic (see Figure 1) of technology for linking and integrating systems. The proposed technology would utilize agent-based adapters connected to client support systems used in industry. The adapters include micro-services (e.g., query, data retrieval, control) that manage the curating, discovering, retrieving, and observing of data across a federated product lifecycle.

The adapters connect the various client support systems through a master handle system that acts as a data "traffic cop" to aggregate all related data to a particular thing and present unified results back to user in the client support system with which he/she is most comfortable working [12]. We consider the master handle system to be a type of in-out (I/O) bus in an identifier system for data to flow and link between cyber and physical things (e.g., specifications, parts, organizations, people). This cyber-physical manufacturing I/O bus enables linking people, organizations, machines, federated data, hardware, and systems together – thus, supporting a digital-thread infrastructure that brings full product lifecycle connections to reality. To accomplish the goals of the digital thread, we leverage graph theory, various management techniques, and system and information modeling.

The work presented by this paper extends and describes the theory behind the authors' previous work [26]. The previous work provided the requirements and a feasibility study that led to the outcome of the work described herein. Further, LIFT [12] is used as the foundation for the architecture of the work presented in this paper. Links and relationships between artifacts are generated and tracked in near-real-time using agent-based adapters. Generating a graph of all data and links across the product lifecycle for a product is assumed to be a large

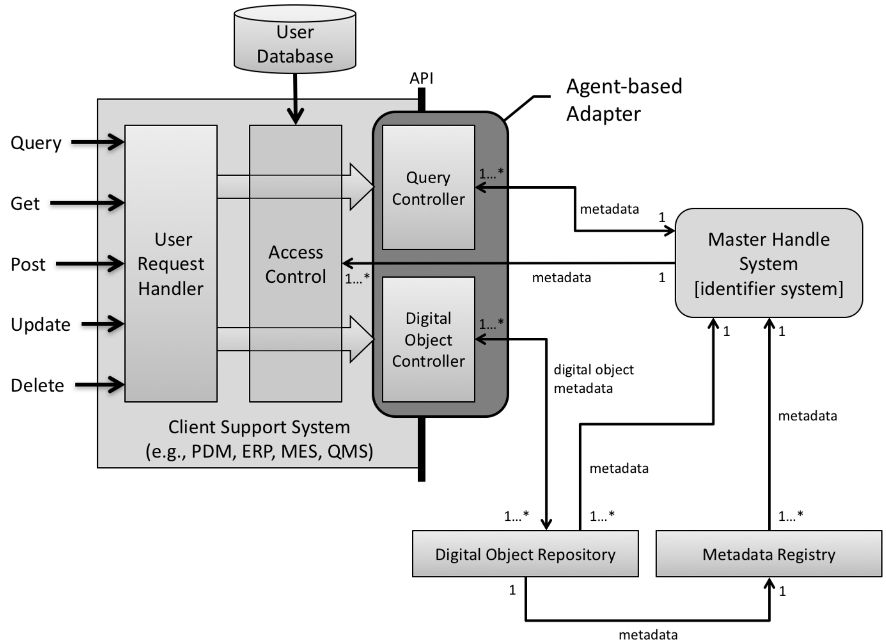

Fig. 1. Example technology schematic for partially enabling the digital thread with linked data through the use of agent-based adapters, applications programming interfaces, and a master data handling system [from [12]].

unstructured data set. However, by tracing the links and relationships while users interact with the data, structure could be inferred through observation and lifecycle-wide information models could be generated dynamically to provide context.

2.2 Graph Theory Applied to Product Lifecycle Management

A graph is defined as consisting of a set of nodes and connecting edges. Assuming there is only one node per each domain of the product lifecycle[1] that can be connected, the number of undirect and directed graphs show there could be between 1,024 $\left(2^{10}\right)$ and 1,048,576 $\left(2^{20}\right)$ graphs [2]. While a real-world manufacturing example probably has more nodes than five, the considerable range of possibilities shown here is a significant risk for introducing uncertainity into the product lifecycle. Trying to manually manage connections of data across the product lifecycle is incomprehensible and a prime reason for the many challenges industry faces today. While graph theory applications to engineering receive sizable attention in the literature, product lifecycle management (PLM) is one area where graphs have not been significantly studied.

A reason for the lack of graph-based research in PLM is because the majority of research is still focused on data management in manufacturing [27]. However, interest in bringing "smart" technologies to manufacturing is motivating studies in graph theory applied to PLM viewpoints. Shilovitsky [28] bridged the gap between data management and PLM by suggesting different types of database technologies for use in PLM. Table 1 presents the

[1] Defined here as (1) marketing, (2) engineering, (3) manufacturing, (4) quality, and (5) sustainment.

[2] In a worst case scenario, the number of graphs that may be formed from a n number of labelled nodes is $2^{\frac{n(n-1)}{2}}$ for undirected graphs and $2^{n(n-1)}$ for directed graphs.

Table 1. Strengths and weaknesses of various database types and their suggested use in product lifecycle management (PLM). Adapted from [28].

Database Type	Strengths	Weaknesses	Suggested use in PLM
Relational	Known data layout and structure	Variable and hierarchical data	Transactional data in specific models
Key-value Pairs	Little or no need of indexes	Create, read, update, and delete and miscellaneous queries	Vaulting. Media.
Columnar	Horizontal scale. clustering.	Undefined data use patterns	Suppliers access. Design collaboration.
Documents	Unknown data structure	Joins and relationships	Vaulting. Media.
Graph	Flexible types of relationships	Limited scale, query-ability	Configurations. Product structure.

strengths, weaknesses, and suggested PLM uses for five types of databases. Graphs are suggested for dealing with configurations and product structure, which aligns well with the types of relationships that must be managed as data is shared throughout the product lifecycle. The work presented in this paper accepts Shilovitsky's suggestion for using graphs in PLM to propose a method for connecting, discovering, and retrieving data across the product lifecycle.

2.3 Identified Gaps

Maturing a new product idea to commercialization requires nurturing and oversight, which only proper management controls can provide [29]. Simons [30] defines management-control systems as "formal, information-based routines and procedures managers use to maintain or alter patterns in organizational activities." Industry applies various management techniques to all aspects of product-lifecycle activities. However, managing and contextualizing data from across the product lifecycle is not as simple as deploying some software. Making effective design decisions is challenging because data use varies based on the role that is interacting with the data [12, 11]. Taking advantage of digital-thread concepts and graphs theory could overcome some of the challenge by helping industry manage different contextual viewpoints based on what role is using the data.

Further, service-oriented architectures provide a significant integration benefit over point-to-point integration. Point-to-point integration of tools is fragile and expensive to develop and maintain because a ripple affect of changes occurs as one tool is modified or replaced [31]. Context is also harder to manage during point-to-point integrations because the individual tools are centered on one discipline while the integrations must support multiple viewpoints, which could lead to the deployment of multiple point-to-point integrations for connecting a single tool to a suite of other tools. Conversely, service-oriented architectures offer the benefit of composing systems dynamically to meet changing demands in the operation of manufacturing systems [32]. Therefore, we believe the gaps identified here could be closed with a standards-based, linked-data approach that leverages the digital thread and graph theory.

3 Information Model and Architecture

A federated digital thread includes artifacts originating from different discipline, tool, and repository ecosystems, such as PLM or application lifecycle management (ALM) systems, in the design-manufacturing-supply-chain

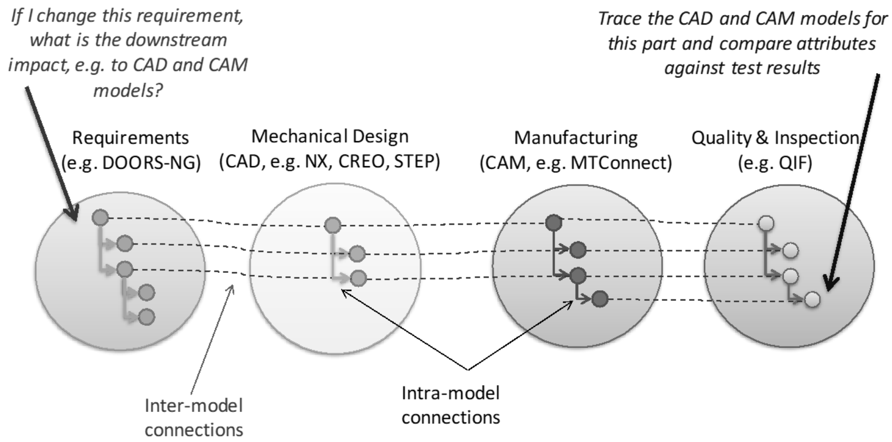

Fig. 2. A representative example of a digital thread for manufacturing [from [33, 34, 26]].

network. Although each repository provides tools to manage the artifacts originating in that repository (e.g., versioning, configuration management, verification and validation), curating artifacts originating from different repositories poses a challenge.

Consider a representative subset of a digital thread (See Figure 2), which connects artifacts originating in four different repositories: (1) product requirements originating in a requirements management system, (2) mechanical design models originating from a PDM system, (3) CAM models originating from manufacturing process-planning tools with MTConnect data coming from manufacturing machines, and (4) quality inspection reports in Quality Information Framework (QIF) originating from a QMS. Even though the artifacts originating in a given repository may be seamlessly linked to artifacts originating in the same repository (intra-model connections), it is the inter-model connections between the artifacts across the repositories that enable a federated digital thread. Both intra- and inter-model connections are necessary to traverse and query a graph-based digital thread. Two sample queries shown in Figure 2 are:

- If a product requirement changes, can we assess the impact of the change downstream to mechanical/electrical design and manufacturing process plans? The impact may be measured in terms of time, resources, and cost to affect the change.
- If a part fails during operation, can we trace upstream to the mechanical/electrical design and product requirements?

Using the idea of a federated digital thread, artifacts can be connected across entire enterprises. Making connections across enterprises is about abstracting up to a higher level to solve problems and make decisions. The connections must be made using a technology-agnostic approach. Technologies change over time, but the information needs of the functions and roles using the technologies do not. Therefore, the method for making connections must use immutable attributes of the artifacts being linked. Some examples of these attributes are location, ownership, or any other attribute that has the possibility of changing without changing the identity of the referent. The goal is to ensure persistent connections regardless of how artifact attributes may change.

Kahn and Wilensky [35] purposed a framework for distributed digital object services. An original motivation for the framework was the need to identify and retrieve information over long periods of time (e.g., tens of years, hundreds of years). Therefore, persistence was a critical design requirement. While Kahn's and Wilensky's framework originally addresses digital objects, the manufacturing sector requires an approach that can manage artifacts that are digital (i.e., cyber) or physical. Connecting only digital objects is not enough for industry because it must also include the connections to the physical world during decision making and problem solving (e.g., traceability analysis, accident investigation). Therefore, we propose an extension to Kahn's and Wilensky's work. Section 3.1 provides our extended digital object architecture that encompasses the Kahn and Wilensky framework and Section 3.2 addresses persistent global identification in the context of manufacturing-specific intellectual property (IP) and data-rights issues.

3.1 Lifecycle Handler System

Starting with the technology schematic (Figure 1) presented by Hedberg et al. [12], the architecture developed for making connections across manufacturing enterprises is shown in Figure 3. The architecture forms what we call the Lifecycle Handler System (LHS). The LHS includes the global handle registry and local handle services from Kahn and Wilensky [35], but also adds client support systems, local graph databases, and agent-based adapters. The LHS enables exposing the digital thread as a set of services so that higher-level analysis and verification applications can be built and deployed for teams across the product lifecycle (e.g., design, manufacturing, and operation).

The LHS system leverages the Handle System [36] to connect to the global handle registry and deploy local handle services. The Handle System was selected as a starting point for our work here based on our previous analysis in [12] and because the Handle System was developed to connect, track, and access information when the storage locations are not always known [35]. Further, the underlying architecture of the Handle System accounted for IP issues as a critical component of the undertaking. Last, backed by the ISO 26324 standard [39], the Distributed Object Identifier (DOI) system[3] is based on the Handle System and the International DOI Foundation has expanded the scope of a handle to be a digital identifier of an object, which they define as "thing: physical, digital, or abstract" [40]. The DOI system serves primarily the media and publication sectors and has approximately 175 million DOI names assigned to date with over 5 billion DOI resolutions per year.

In the LHS, handles are generated and managed in accordance with RFC 3650 [36], RFC 3651 [37], and RFC 3652 [38]. A handle is composed of a naming authority and local name. The handle syntax is shown in Listing 1 and `20.500.11993/nist.tdh1`[4] is an example of a real handle where "20.500.11993" represents the <NamingAuthority> and "nist.tdh1" is the <LocalName> of the object that represents the digital persona for one of the authors of this paper. Figure 4 provides an overview of the process defined by RFC 3650 for resolving

[3] `http://www.doi.org`

[4] The metadata for the example handle may be retrieved via the HTTPS protocol by visiting `http://hdl.handle.net/20.500.11993/nist.tdh1?noredirect` in any web browser.

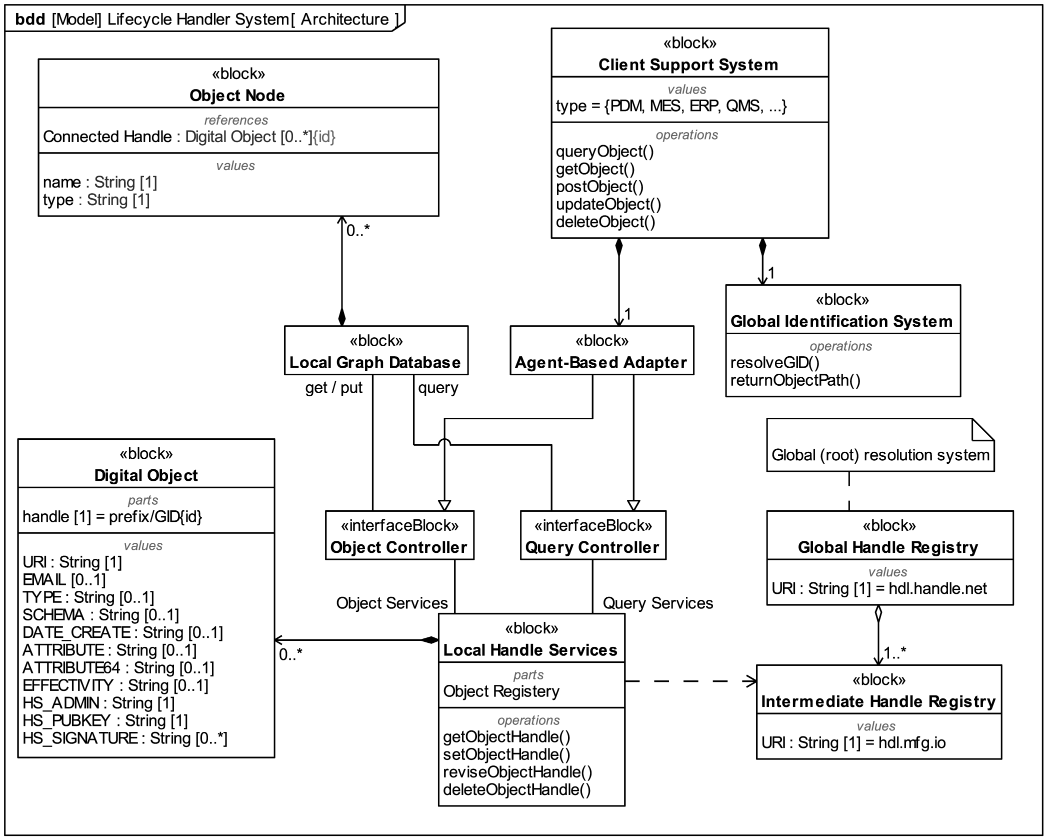

Fig. 3. An architecture for making connections across enterprises based on the Lifecycle Information Framework and Technology (LIFT) concept [12]. The definition of the Global Handle Registry, Intermediate Handle Registry, and Local Handle Services are the work of [35] and standardized in accordance with RFC 3650 [36], RFC 3651 [37], and RFC 3652 [38]. The remain components of the archicture are proposed herein.

Listing 1. Handle syntax from RFC 3651 [37].

```
<Handle> = <NamingAuthority> "/" <LocalName>

<NamingAuthority> = *(<NamingAuthority> ".") <NAsegment>

<NAsegment> = 1*(\%x00-2D / \%x30-3F / \%x41-FF )
                ; any octets that map to UTF-8 encoded
                ; Unicode 2.0 characters except
                ; octets '0x2E' and '0x2F' (which
                ; correspond to the ASCII characters '.',
                ; and '/').

<LocalName> = *(\%x00-FF)
                ; any octets that map to UTF-8 encoded
                ; Unicode 2.0 characters
```

handles from the global handle registry to the local handle services. The client queries the global handle registry to determine which local handle service manages the handle's prefix. Then, the client queries that local handle services to retrieve the information about the handle. Finally, the client processes the returned information in accordance with the requested action.

```
  ------------------------
 |                        |    4. Result of client request
 | Client with global     |  <------------------------------.
 |  service information   |                                 |
 |                        |  ----------------------------.  |
  ------------------------     3. Request to responsible |  |
            |   ^                 Local Handle Service   |  |
1. Client   |   |                                        |  |
query for   |   |                                        |  |
naming      |   | 2. Service information                 |  |
authority   |   |    for "10.1045"                       V  |
"10.1045"   |   |                                 ----------------------
            |   |                                |                      |
            V   |                                | Local Handle Service |
       ---------------                           | responsible for the  |
      |               |                          | naming authority     |
      | Global Handle |                          | "10.1045"            |
      |   Registry    |                          |                      |
      |               |                           ----------------------
       ---------------
```

Fig. 4. Handle resolution from global handle registry to local handle service from RFC 3650 [36].

An agent-based adapter composed of micro-services for query and object control is attached to client support systems in the LHS. The adapter tracks activity within the client support systems and captures links between artifacts. The adapter stores the handles of artifacts as nodes in a local graph database. The handles of each linked connection are also captured for the nodes in the database. For example, for a CAM model generated using a portion of a CAD model, a node is generated for both models, the handles of each model are captured, and a directed or undirected edge is generated between the two nodes depending on how the CAM model references the CAD model.

After a handle is created for an artifact, when information is required about that artifact, one can query the handle of that artifact to discover and retrieve its metadata. RFC 3651 provides predefined data types for metadata repositories attached to local handle services [37]. We propose the additional data types described in Table 2 be included at a minimum for metadata repositories attached to a manufacturing-oriented LHS. Then, invoking the object controller micro-service triggers tasks with the local handle services and global identifier (GID) sub-system to resolve the path of the digital object and fetch the complete artifact.

In cases where artifacts are digital objects, the user may retrieve an artifact through the LHS if the user has the appropriate permissions. The user must have the appropriate authentication and authorization to discover and retrieve artifacts. The LHS respects three user access scenarios: (1) objects are not discoverable and retrievable, (2) objects are discoverable and not retrievable, and (3) objects are discoverable and retrievable. The access scenarios respect permissions negotiated by the agent-based adapters and the repositories to which the adapters are attached.

For the work described in this paper, capturing nodes and edges in the local graph database is not automatic, but require input to the agent-based adapter describing how to form the nodes and edges. A desired future

Table 2. The schema for the metadata repositories attached to manufacturing-oriented local handle systems in the LHS system.

Data Type	Index	Requirement	Description
URI	1	Required	The URI type provides a Uniform Resource Identifier (URI) that is passed to a general-purpose name service for accessing the artifact referenced by a handle.
EMAIL	2	Optional	The EMAIL type provides a UTF8-encoded email addresses for a handle that points to a person.
TYPE	3	Optional	The TYPE type provides the type of artifact that is referenced by a handle. The TYPE notation is based on the proposed structure presented in Figure 5.
SCHEMA	4	Optional	The SCHEMA type provides the schema used to provide the data provided by the ATTRIBUTE type. The SCHEMA type is required when the ATTRIBUTE type is included for a handle.
DATE_CREATE	5	Optional	The DATE_CREATE type captures the timestamp for when the artifact referenced by the handle was originally created.
ATTRIBUTE	6	Optional	The ATTRIBUTE type provides informative data about the artifact in JavaScript Object Notation (JSON) form according to the schema provided by the SCHEMA. type.
ATTRIBUTE64	7	Optional	The ATTRIBUTE64 type is a base64 encoding of the data provided by the ATTRIBUTE type. While it is not a best practice to duplicate data, this data type is intended to enable automation by providing computer-interpretable data and overcoming a limitation in how the handle system's metadata repositories capture and deliver encoded data as strings through its application programming interface (API). Policy must be in place to ensure the data provided by both ATTRIBUTE and ATTRIBUTE64 always match. If the metadata repositories are capable of capturing encoded data without breaking the structure of the data, then this data type may not be needed.
CM_STATUS	10	Optional	The CM_STATUS type provides type of configuration management (CM) tracking and the date effectivity or serial effectivity for the artifact.
HS_ADMIN	100	Required	The HS_ADMIN as defined by RFC 3651 [37].
HS_PUBKEY	300	Required	The HS_PUBKEY type provides encoded information describing a public key for authenticating entities in the handle system.
HS_SIGNATURE	400	Optional	The HS_SIGNATURE type provides the digital signature of an entity that vouches for the metadata included for a handle.

extension of the LHS is to deploy an inference micro-service that can dynamically track activity in near-real-time and capture autonomously the required information to be curated in the local graph database. Further, several nuances around how industry operates its information technology (IT) networks presents challenges for ensuring effective and persistent identification, addressing, and accessing of artifacts. Section 3.2 describes how we overcame the challenges.

3.2 Persistent Global Identification

The LHS provides a unique, global identifier (GID) system for addressing and searching all artifacts and their inter-relationships in the digital thread. This is a challenging task because the information about the artifacts par-

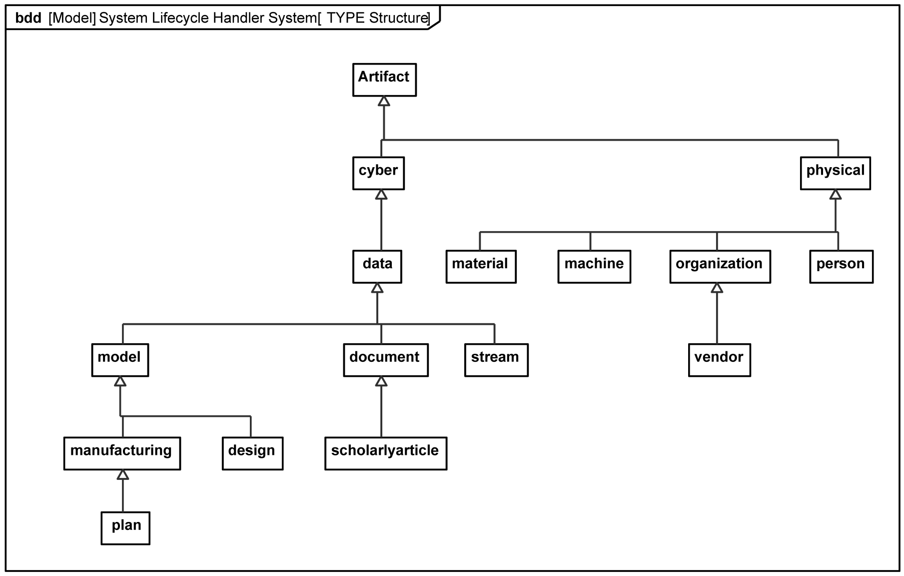

Fig. 5. Preliminary TYPE structure proposed for describing artifacts referenced in the LHS. The type value (e.g., cyber.data.document.static) would be included in the TYPE metadata element of the artifact as proposed in Table 2.

ticipating in the digital thread originates from multiple repositories, databases, requirements, system-architecture models, product-structure information (e.g., bill of materials (BOM) and CAD models), and manufacturing plans and data streams. The digital thread includes data originating from multiple sources, such as static files (e.g., spreadsheets, documents), computer models, real-time data streams, and hardware. Each type of repository and/or database provides its own identification system that is local to artifacts and relationships managed by such tools. The identifier for the data may range from cells in a spreadsheet, to unique string-based identifiers for a part in a computer model, to URIs provided by a Representational State Transfer (REST)-based, Hypertext Transfer Protocol (HTTP) service. However, when we build a digital thread by federating artifacts from multiple repositories, we need a GID system that can provide an address for every artifact or inter-relationship throughout the product lifecycle.

We propose the GID to deprecate the need for “smart” naming conventions for artifacts. Instead, industry should be enabled potentially with the ability generate dynamic information models and ontologies from the artifact metadata. We observed industry have too many arguments over artifact naming schemas instead of discussing the metadata to describe the artifacts and the information requirements for what the artifacts contain.

The Handle System addresses a majority of the tasks for unique, persistent addressing, but does not meet all of the manufacturing sector’s needs. The digital thread must work in a global environment that include(s) computer firewalls, network security, and multi-layer authentication. The GID for an artifact in the digital thread may not be a single URI as in a generic handle approach (e.g., publications). Instead, the GID is often an ordered set of addresses (e.g., URIs or other identifier types) that must be resolved recursively to navigate through multiple layers of namespaces, firewalls, and authentication servers.

Consider the example shown in Figure 6 where the design model of a part managed in a PDM system needs

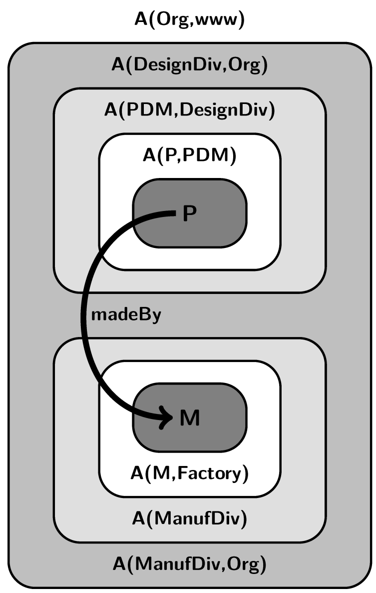

Fig. 6. Multi-level addresses for locating artifacts across enterprise layers

to be assigned to a specific machine in the factory that will make this part. For simplicity, we will assume that the same organization is designing and manufacturing the part. In a globally distributed supply chain, the challenge presented here will be compounded. The abbreviation `A(x,base)` is used to represent the address of an artifact x in the context of the base artifact. The address can be a URI or some form of an identifier that can be resolved.

At the highest level, an organization artifact *Org* may have a gateway server available on the internet (world wide web) for all incoming requests, denoted as `A(Org, www)` in the figure. Next, the gateway servers for the various divisions in the organization are generally not reachable directly from the open internet due to firewalls, but reachable from the organization's gateway server. `A(DesignDiv,Org)` and `A(ManufDiv,Org)` are the respective addresses of the design and manufacturing divisions in context of the organization. Similarly, `A(PDM,DesignDiv)` is the address of the PDM server reachable from the design division, and `A(P,PDM)` is the address of the part-specification, P, in the context of the PDM server. Hence, the GID and address for P is an ordered set of addresses: {`A(Org,www), A(DesignDiv,Org), A(PDM,DesignDiv), A(P,PDM)`}. Traversing the GID is similar to *hops* in a packet-switching network (e.g., the route a packet takes to go from its source to its destination).

To reach P, the LHS is augmented by a resolver system that recursively traverses the chain of addresses, authenticating the request at each base artifact to reach the next artifact. The process for traversing recursively the chain of addresses is shown in Algorithm 1. A similar resolution process must be followed to reach machine, M. Once P and M can be reached uniquely in this manner, we can establish a relationship that P is realized by M. A collection of these relationships spins the digital thread.

4 Case Study

A case study using data from a real design and manufacturing process was used to test and validate the method described in this paper. The use case is an enclosure box for a payload assembly used in a configurable

Algorithm 1: Algorithm for a resolver system that recursively traverses the chain of addresses.

Input: A as {GID}
Output: recursive traversal of A

```
initialization;
if (A = null) then
    return
end
s ← empty stack;
s.push(A);
set s.pos to first address of s;
while s.pos is not end of stack do
    read address at s.pos;
    send request to address at s.pos;
    move s.pos to next address of s;
end
```

unmanned aerial vehicle (cUAV). The payload assembly is a subsystem of the overall cUAV system. The enclosure box is an assembly composed of eight components – four design-build parts and four standard procured parts. This case study focused on only three of the design-build parts: 1) box, 2) internal plate, and 3) cover.

The dataset for the case study comes from the International Council on Systems Engineering (INCOSE) 2015 Model-Based Systems Engineering (MBSE) Model Lifecycle Management Workshop [41] and a collaborative project between National Institute of Standards and Technology (NIST) and the Manufacturing Technology Centre (MTC) [42]. Several data types are included in the dataset. The SysML model of the cUAV was retrieved from the results of the INCOSE workshop. All other data was retrieved from the NIST-MTC project. Using Solidworks 2016, CAD models captured the digital product definition in accordance with American Society of Mechanical Engineers (ASME) Y14.41-2012 [43]. The manufacturing operations were executed using numerical control (NC) programs in an ISO 6983 [44] compliant format. The manufacturing execution was monitored using an Extensible Markup Language (XML)-based implementation of the MTConnect version 1.3 standard [45]. First article inspection reporting (FAIR) and receiving and incoming inspection (RII) reports were produced using the QIF version 2.1 standard [46].

Figure 7 depicts the layered approach for organizing the digital artifacts using in the case study. Three layers were used. The top layer is the Product Concept Level, which contains the high-level stakeholder needs and product requirements. The requirements for the case study were managed in Jama requirements management tool[5]. The middle layer is the Design Variant Level, which contains the digital product definitions and specifications. Four variants of the assembly, four variants of the box, one variant of the plate, and two variants of the cover were available. The CAD models were managed in a GitHub[6] repository and the status of each variant was managed in the Jira issue and project tracking software[7]. The bottom layer is the Part Instant Level, which contains information about the realized product and parts. Twenty instances of each part were fabricated. The

[5] `https://www.jamasoftware.com`
[6] `https://github.com`
[7] `https://www.atlassian.com/software/jira`

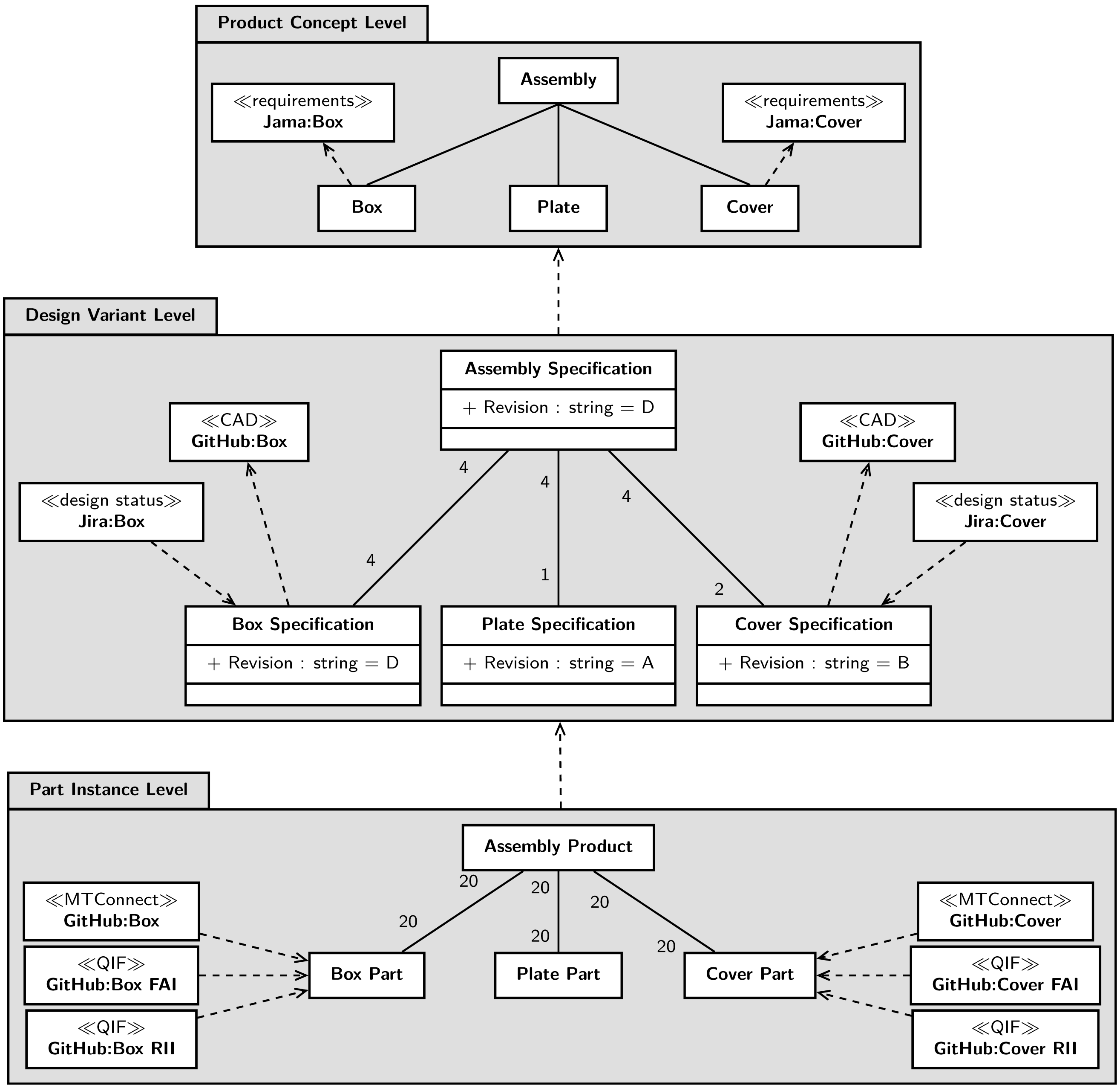

Fig. 7. An overview of the layered approach for organizing the digital artifacts of the enclosure box for an payload assembly used in the cUAV use case. Data came from the INCOSE 2015 MBSE Model Lifecycle Management Workshop [41] and a collaborative project between NIST and the MTC [42]. FAI = first article inspection, RII = receiving and incoming inspection

MTConnect data, QIF FAIR reports, and QIF RII reports, were managed in a GitHub repository.

Artifacts were linked and managed using the method described in Section 3. The implementation prototype of the architecture and services used several commercially and/or freely available software tools. The Syndeia tool[8] was used to build and manage all the links between the enclosure box assembly artifacts. The graph database was built in Neo4j[9]. Cypher query language [47] was used to query the database.

For this case study, only the enclosure box assembly, box, and cover artifacts were managed in the prototype.

[8] http://intercax.com/products/syndeia/
[9] https://neo4j.com/product/

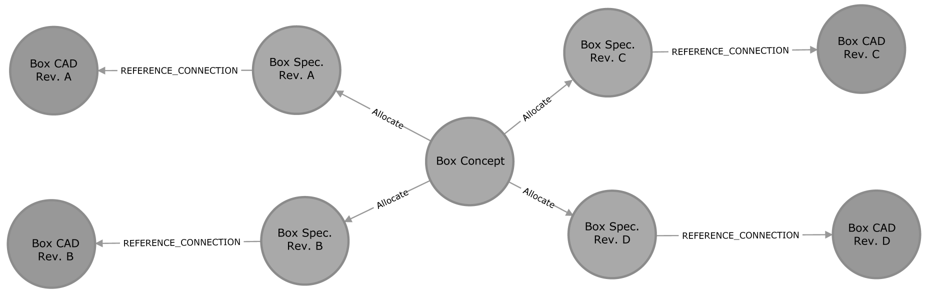

Fig. 8. Sub-graph showing all Design Variants of Product Concept 'NIST_MTC_CRADA_BOX' and associated CAD files. The raw node names are changed in this figure to help the reader align the sub-graph with the artifacts shows in Figure 7.

Overall, 145 nodes and 436 edges were generated between the assembly, box, and cover. The nodes consisted of requirements in a Jama instance, CAD files stored in a GitHub repository, tracking information in a Jira instance, and MTConnect and QIF data captured in XML files stored in additional GitHub repositories. Reviewing the cover part component in the assembly would reveal there are 44 nodes and 66 edges related to the cover part. When considering a larger product, such as an aircraft or automobile, the number of connections could grow into the thousands, if not millions. This presents evidence to the significant amount of data and links that must be managed, which requires considerable amounts of human resources if managed manually. Searching for and retrieving the right data for a particular purpose could take a person hours up to days [48]. Using the method described in this paper, discovering information, retrieving it, and extracting knowledge took seconds to complete.

Listing 2 provides several Cypher queries used for discovering information, retrieving it, and extracting knowledge in this case study. The goal of this case study is to show traceability from multiple viewpoints of the product lifecycle. Using the query on line 2 of Listing 2 would return all the nearest and next-nearest neighbors to the part instance of the assembly with serial number "D01." Using the query on line 5 of Listing 2 would return all the tasks connected to the part instance for the box component with serial number "D01." Using the query on line 8 of Listing 2, all the design variants of the product concept for the box and the associated CAD files are displayed in Figure 8. Using the query on line 11 of Listing 2 provides all the part instances of the design variant for "Revision D" of the box.

Data traceability can be displayed starting with manufacturing and quality viewpoints too. Using the query on line 14 of Listing 2, all the manufacturing and quality files managed and associated with the part instance for the assembly with serial number "D01" are shown in Figure 9. Using the query on line 17 of Listing 2 would return the CAD files managed and associated with the manufacturing data for the box with serial number "5." Using the query on line 20 of Listing 2 would provide all the requirements connected to the part instance of the box with serial number "D01."

Lastly, one part instance of the box was misplaced during shipping and did not go through RII. Using the query on line 23 of Listing 2, it can be determined that 20 boxes were fabricated. However, using the query on line 26 of Listing 2, it can be seen that only 19 boxes went through RII. The 19 instances of the box can be listed using the query on line 29 of Listing 2 and a snippet of the result of the query is shown in Listing 3.

Listing 2. Cypher Query Language entries for the prototype tested in the case study.

```
// Show all nearest and next-nearest neighbors to Part Instance `NMC_ASSBLY_D01'
MATCH (n1)<-[r1]-(n)-[r]-(s:Block) WHERE s.name=~'NMC_ASSBLY_D01' AND NOT n:Repository AND
    NOT n:Package AND NOT n1:Package AND NOT n1:Repository RETURN n1,r1,n,r,s

// Show all JIRA Tasks connected to Part Instance `NMC_BOX_D01'
MATCH (m:JIRA_Task)-[r]-(s1)-[r1]-(s:Block) WHERE s.name=~'NMC_Box_D01' RETURN m,r,s1,r1,s

// Show all Design Variants of Product Concept `NIST_MTC_CRADA_BOX' and associated CAD files
MATCH (m:File)-[r1]-(n:Block)-[r:Allocate]-(s:Block) WHERE s.name=~'NIST_MTC_CRADA_BOX'
    RETURN m,r1,n,r,s

// Show all Part Instances of Design Variant block `NIST_MTC_CRADA_BOX RevD'
MATCH (n:Block)<-[r:Allocate]-(m:Block) WHERE m.name=~'NIST_MTC_CRADA_BOX RevD' RETURN n,r,m

// Show all manufacturing and quality files in GitHub associated with Part Instance
    `NMC_ASSBLY_D01'
MATCH (m:GitHub_File)<-[t]-(n)<-[r]-(s:Block) WHERE s.name=~'NMC_ASSBLY_D01' RETURN m,t,n,r,s

// Show the CAD files in GitHub associated with manufacturing data `Box-Hurco02-05of20.xml'
MATCH (m:File)-[t1]-(n1)-[t]->(n)-[r]->(s:GitHub_File) WHERE s.name='Box-Hurco02-05of20.xml'
    RETURN m,t1,n1,t,n,r,s

// Show all Jama requirements connected to Part Instance `NMC_Box_D01'
MATCH (n:Jama_Requirement)-[r2]-(s2)-[r1]-(s1)-[r]-(s:Block) WHERE s.name=~'NMC_Box_D01' AND
    NOT s1:Repository RETURN n,r2,s2,r1,s1,r,s

// How many instances of Box were fabricated?
MATCH (m:Block)-[t:Allocate]-(n:Block)-[r]-(s:Block) WHERE s.name=~'NIST_MTC_CRADA_BOX'
    RETURN count(m)

// How many instances of Box were through receiving and incoming inspection?
MATCH (n1:GitHub_File)-[r1:REFERENCE_CONNECTION]-(m:Block)-[t:Allocate]-(s:Block) WHERE
    n1.name=~'BoxResults_19_samples.QIF' AND s.name=~'NIST_MTC_CRADA_BOX RevD' RETURN count(m)

// List the instances of the box that went through receiving and incoming inspection
MATCH (n1:GitHub_File)-[r1:REFERENCE_CONNECTION]-(m:Block)-[t:Allocate]-(s:Block) WHERE
    n1.name=~'BoxResults_19_samples.QIF' AND s.name=~'NIST_MTC_CRADA_BOX RevD' RETURN m
```

The example Cypher queries present evidence of the power of graphs applied to manufacturing contexts. Each node also has a handle associated to it, which provides additional metadata and linking capabilities to enable quickly identifying the type of artifact being referenced. Since the data for the enclosure box resides across several systems in different enterprises, the multi-level addressing method shown in Figure 6 was required. An example of the GID for the graph of associated CAD files up through the design variant to the product concept for Revision D of the box component is: {`20.500.11993/NIST.MTC.CRADA.BOX`, `20.500.11993/NIST.MTC.CRADA.BOX.SPECIFICATION`, `20.500.11993/NIST.MTC.CRADA.BOX.REV.D`}. Listing 4 presents a snippet the JSON result for the handle “20.500.11993/

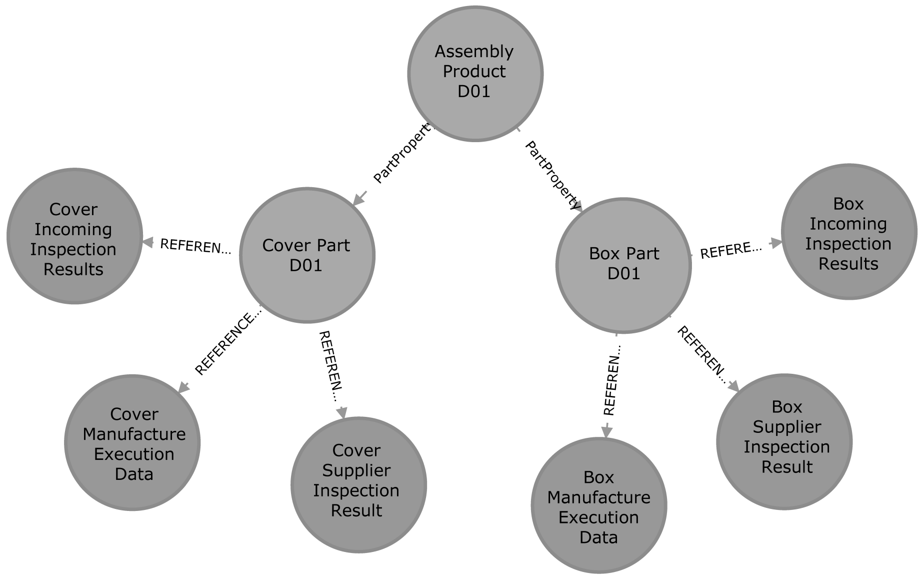

Fig. 9. Sub-graph showing all all manufacturing and quality files in GitHub associated with Part Instance 'NMC_ASSBLY_D01.' The raw node names are changed in this figure to help the reader align the sub-graph with the artifacts shows in Figure 7.

nist.mtc.crada.box.rev.d" from the REST-based API [10].

This case study shows the magnitude of data that must be managed in manufacturing, even when the dealing with a relatively small product assembly. The case study provides evidence that the method proposed in this paper potentially overcomes the challenges associated with managing manufacturing-related data [4, 5, 6, 7]. The combination of the graphs and handles associated to each node enables rapid querying, discovering, and retrieving of artifacts based on the users access permissions when proper links are established between related nodes (see Section 3.1.

5 Discussion

Purpose-built modeling is currently the recommended approach because it enables "expert systems" that support making decisions in contextual ways related to a specific function and role [49]. Conversely, purpose-built models are not scalable. Data requires context when related to decisions [50]. Data alone is not sufficient for decision making because the decision maker must understand the scope and type of the problem the decision is intended to solve. As the scope of the problem changes, the models must also change. Thus, connecting heterogeneous information and systems introduces a paradox to the steadfast approach of purpose-driven modeling. A trade-off of how purpose-built to make a model versus how scalable (i.e., generalized) to make a model must be considered. This requires integrating domains in multiple directions while providing scalable contextual models. Overcoming these challenges is not easy, but we believe a standards-based linked-data approach, using the digital thread, provides the best opportunity for maximizing the successful deployment of smart manufacturing.

[10] Manufacturing-related handle metadata can be resolved against the local handle servive API at `https://hdl.mfg.io/api/handles/`.

Listing 3. Snippet of text result of a query to list the 19 RII quality files for the box part instance.

```
 "m"
{
 "name": "NMC_Box_D019",
 "gid": "PROJECT-b11f2583-da67-4515-b8d9-1304d22c06a7 |
     _18_5_3_63e021c_1521994265350_944094_15411"
}
{
 "name": "NMC_Box_D018",
 "gid": "PROJECT-b11f2583-da67-4515-b8d9-1304d22c06a7 |
     _18_5_3_63e021c_1521994264434_175058_15408"
}
{
 "name": "NMC_Box_D017",
 "gid": "PROJECT-b11f2583-da67-4515-b8d9-1304d22c06a7 |
     _18_5_3_63e021c_1521994264123_336079_15405"
}
{
 "name": "NMC_Box_D016",
 "gid": "PROJECT-b11f2583-da67-4515-b8d9-1304d22c06a7 |
     _18_5_3_63e021c_1521994263698_60315_15402"
}
.
.
.
```

5.1 Contextualizing Graph-Based Viewpoints

Contextualizing data from across the product lifecycle to make design decisions is challenging because data use varies based on the role that is interacting with the data [12, 11]. Graphs can overcome some of the challenges by managing different contextual viewpoints based on what role is using the data. Sub-graphs can be extracted from the graph to enable observing the connections that matter most to a role. Further, trees – specialized graphs – can be extracted from graphs to enable decision making and sorting [51]. The root of the trees changes dynamically based on the domain expert's required context and the types of decisions he/she would need to make. Fortunately, various sort, search, reduction, and decision algorithms for "decision trees" and other types of graphs are widely available to solve large, computationally intensive, practical problems that are often encounter in engineering contexts [51].

Using graphs to link data across the product lifecycle enables quickly extracting domain-specific viewpoints. For instance, consider the graphs shown in Figure 10. Figure 10(a) provides a product lifecycle viewpoint for the box component from the case study in Section 4. Which Figure 10(b) and (c) present an emphasized sub-graph for a manufacturing-specific viewpoint and a materials-specific viewpoint, respectively.

Viewpoint-identification capability provides efficient and effective segmentation of massive datasets generated by enterprises in the manufacturing sector. Without this approach, industry spends considerable amounts of time searching for data related to its products. With this approach, domain experts can quickly find information from

Listing 4. Snippet of JSON result from the REST-based API for the “20.500.11993/nist.mtc.crada.box.rev.d” handle.

```
{
  "responseCode":1,
  "handle":"20.500.11993/nist.mtc.crada.box.rev.d",
  "values":[
    {
      "index":1,
      "type":"URL",
      "data":{
        "format":"string",
        "value":"https://smstestbed.nist.gov/tdp/mtc/CAD/NIST-MTC-CRADA-Mo...
      },
      "ttl":86400,
      "timestamp":"2018-08-24T19:47:28Z"
    },
    {
      "index":2,
      "type":"TYPE",
      "data":{
        "format":"string",
        "value":"cyber.data.model.design"
      },
      "ttl":86400,
      "timestamp":"2018-08-24T19:47:28Z"
    },
    {
      "index":3,
      "type":"SCHEMA",
      "data":{
        "format":"string",
        "value":"http://schema.org/ProductModel"
      },
      "ttl":86400,
      "timestamp":"2018-08-24T19:47:28Z"
    },
.
.
.
```

across the enterprise that relates to their needs, and they can quickly move to gathering actionable intelligence.

5.2 Knowledge Generation

Traceability, impact analysis, and continuous validation and integration of the digital thread are important aspects of configuration management. The greatest impact of the digital thread is in the continuous analyses that can be performed. In the simplest form, we have basic traceability where one can traverse the digital thread using the intra- and inter-model connections, starting with any artifact. However,a greater capability is to use graph-pattern matching and graph traversals to assess the upstream and downstream impact of changes in any

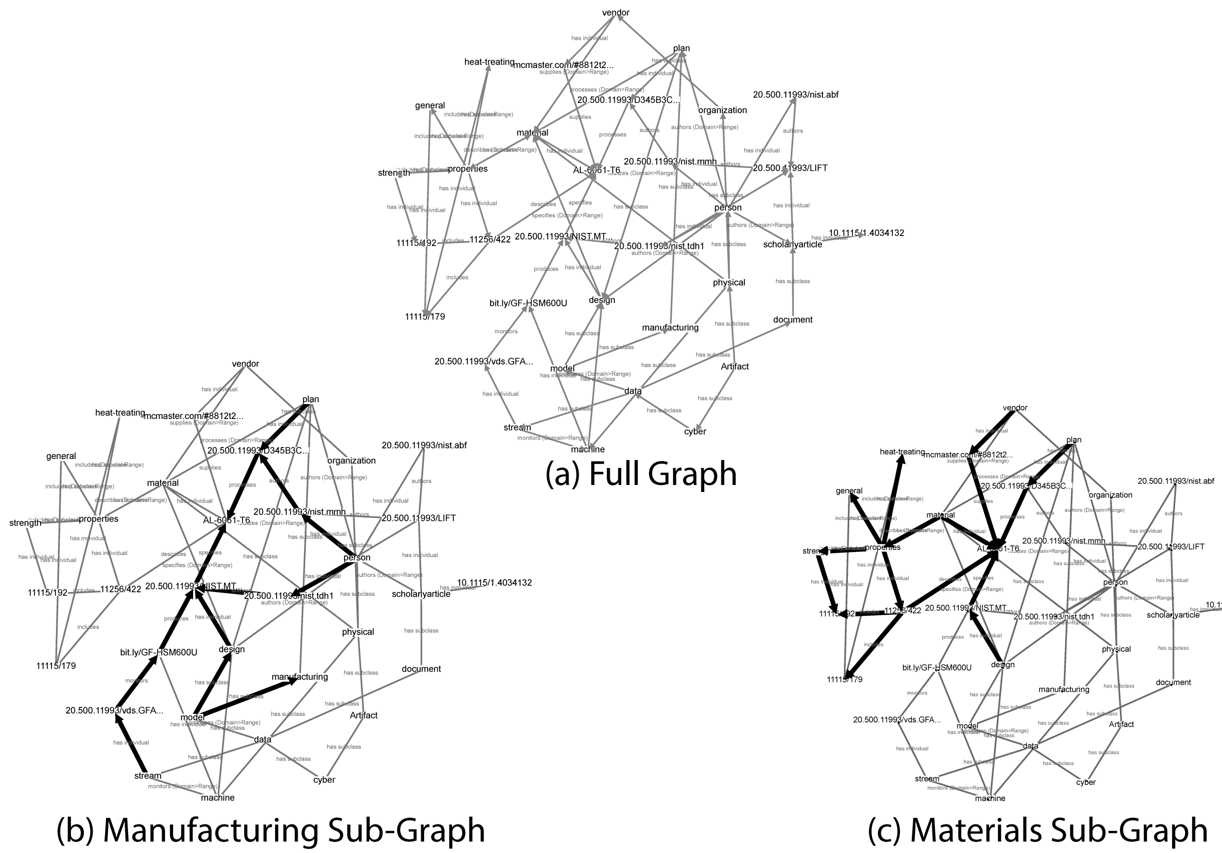

Fig. 10. Contextual Graph-Based Viewpoints for a full product lifecycle viewpoint, a manufacturing viewpoint, and a materials viewpoint.

artifact [34]. For example, computing the downstream impact of changes in a requirement, or querying upstream requirements and analyses done on a part when it fails during operation.

Further, Feng et al. [10] developed a method for managing knowledge in the context of smart manufacturing. The authors provided three contributions: (1) context for data, information, understanding, and autonomy in knowledge generation, (2) knowledge constructs decomposed into basic, composable units, and (3) a reference application to smart manufacturing. However, Feng et al. found further advances in knowledge-base architectures is required to better enable integration of information across a product lifecycle [10]. We have shown through the case study in Section 4 that distributed and/or federated information can be effectively linked across several enterprises and information can quickly be curated, discovered, and retrieved. Feng et al. defined necessary knowledge contructs for the product lifecycle, while the work described here provides the necessary information structuring, object representation, and communication mechanisms. Together, the two bodies of work provide a viable solution to industry for enabling smart manufacturing digital threads.

5.3 Further Research

Communicating with a Diverse Set of Systems. Implementing the architecture we propose in this paper requires addressing how different types of systems that use several generally accepted authentication and communication protocols can be connected in a way that universal access is possible. We believe a "standard interface" approach is best, but what exactly is a standard interface is not completely understood yet and topic of future work for us. Further, we recognize that connecting all systems to each other is not easy or even necessary. Instead, we recommend a holistic analysis be conducted to determine what systems must communicate with each other. Then, using the concept of an agent-based adapter, the enterprises should be able to simplify the connection of a specific system into a larger network of systems. The best approach would be to harmonize all the enterprise systems around a standards-based API, but we recognize that total harmonization is not feasible. Therefore, at a higher level of abstraction we propose the agent-based adapter be connected to a system's API as the recommended method for communicating with any enterprise system. Further research is needed to provide specific recommendations for the best available authentication and communication protocols to understand how the Query Controller and Object Controller, shown in Figure 3, could be best wrapped in an agent-based adapter and implemented widely at line ten of Algorithm 1.

Dynamically Generating Connections. The LHS must provide capabilities to generate and register artifacts in the digital thread and link them using connections. For example, the connections are needed when generating design models from requirements (e.g., design synthesis), or generating simulation models and manufacturing process plans from design models, or registering new machines and machine configurations on a factory floor. Further, the LHS should enable automated generation of connections between artifacts when one is generated from the other. For instance, connections between design and manufacturing models are automatically generated when manufacturing models are generated from design models. This would overcome the manual creation of connections between artifacts that is laborious. Further research is required to enable the autonomous linking capabilities. Specifically, in near-real-time, how are all the links across enterprises tracked? Or how can inference systems be used to facilitate tracking of links?

Frequently Asked Queries. We presented example queries through the case study described in Section 4. However, a complete and concise set of recommended "Frequently Asked Queries" must be research furthered to provide industry with a reference library of graph-based queries that can be deployed to answer key questions across the product lifecycle. Each role in the product lifecycle has typical questions he/she asks about a product while executing tasks in the context of his/her domain expertise. Combining a library of common queries with the methods described in this paper could significantly reduce the effort of human capital in making decisions by leveraging the capabilities of generating contextual graph-based viewpoints and quickly extracting actionable intelligence through knowledge generation.

6 Conclusion

This paper provides a method for using graphs to link data across the product lifecycle enabling smart manufacturing digital threads. It allows the possibility of quickly locating artifacts across distributed and/or federated enterprises without making any presumptions about the objects, the artifacts, or their locations. The expected outcome of the major contributions presented this work are a standards-based, linked-data approach, providing seamless traceability across the product lifecycle, enabling high-quality manufacturing contextualization of information, and supporting enterprise-wide knowledge reuse.

The method presented here leverages several established and trusted approaches and technologies. The first is the Handle System, which is the backbone to the widely popular Distributed Object Identifier (DOI) system that persistently identifies media and publication objects – for instance, the publisher of this paper provided a DOI for pointing universally to this paper. Second, we leverage the foundations of graph-theory, which provides the ability to quickly create, track, and query connections between artifacts in support of contextually generating knowledge about the product lifecycle. Last, we extended generally accepted linked-data approaches to manufacturing contexts and provided additional capabilities to overcome architectural and IP-related challenges that are specific to the manufacturing section.

The prototype implementation and case study presented through this work is an incremental step toward providing industry with connected systems and linked-data federated across entire enterprises. We recognize more work is required and identified several areas of further research needed to achieve success. Given that the problem is significantly large, an easily implemented solution will not be realized overnight. However, starting with deploying the persistent addressing and methods for connecting systems as we proposed here should provide industry with some much needed relief in its struggle with managing all its data.

The next steps in this work is to enhance the reliability of data available by introducing more rigor in how links are stored, configured, and where the links are stored. Further, a more comprehensive metadata schema is in development. This includes leveraging work to extract the minimum information requirements to complete one loop of the product lifecycle [9, 52, 7]. Making these enhancements, coupled with continuous research and development to close all identified industrial gaps, puts the LHS in a good position to deliver significant impact through enabling cost-effective deployment of digital threads.

Acknowledgements

The authors wish to thank Allison Barnard Feeney, Timothy Sprock, Brian A. Weiss, and Vijay Srinivasan from NIST and the peer-reviewers for their comments and input to this paper.

Disclaimers

 Certain commercial systems are identified in this paper. Such identification does not imply recommendation or endorsement by NIST. Nor does it imply that the products identified are necessarily the best available for the purpose.

References

[1] MAPI Foundation, 2015. Facts about modern manufacturing. url: https://mapifoundation.org/manufacturing-facts/2015/8/9/facts-about-modern-manufacturing.

[2] Hedberg Jr, T., Lubell, J., Fischer, L., Maggiano, L., and 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), mar, p. 021001. doi: 10.1115/1.4032697.

[3] Anderson, G., 2016. The economic impact of technology infrastructure for smart manufacturing. Tech. rep., National Institute of Standards and Technology, Gaithersburg MD. doi: 10.6028/NIST.EAB.4.

[4] Prasad, B., Morenc, R. S., and Rangan, R. M., 1993. "Information management for concurrent engineering: Research issues". Concurrent Engineering, **1**(1), mar, pp. 3–20. doi: 10.1177/1063293X9300100102.

[5] Rangan, R. M., Rohde, S. M., Peak, R., Chadha, B., and Bliznakov, P., 2005. "Streamlining product lifecycle processes: A survey of product lifecycle management implementations, directions, and challenges". Journal of Computing and Information Science in Engineering, **5**(3), sep, p. 227. doi: 10.1115/1.2031270.

[6] Waurzyniak, P., 2012. "Manufacturing factory data". Manufacturing Engineering, jul, pp. 71–82.

[7] Hedberg Jr, T. D., Hartman, N. W., Rosche, P., and Fischer, K., 2017. "Identified research directions for using manufacturing knowledge earlier in the product life cycle". International Journal of Production Research, **55**(3), pp. 819–827. doi: 10.1080/00207543.2016.1213453.

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

[9] Ruemler, S. P., Zimmerman, K. E., Hartman, N. W., Hedberg Jr, T., and Barnard Feeney, A., 2016. "Promoting model-based definition to establish a complete product definition". Journal of Manufacturing Science and Engineering, **139**(5), nov, p. 051008. doi: 10.1115/1.4034625.

[10] Feng, S. C., Bernstein, W. Z., Hedberg Jr, T., and Barnard Feeney, A., 2017. "Toward knowledge management for smart manufacturing". Journal of Computing and Information Science in Engineering, **17**(3), jul, p. 031016. doi: 10.1115/1.4037178.

[11] Regli, W., Rossignac, J., Shapiro, V., and Srinivasan, V., 2016. "The new frontiers in computational modeling of material structures". Computer-Aided Design, **77**, pp. 73–85. doi: 10.1016/j.cad.2016.03.002.

[12] Hedberg Jr, T. D., Barnard Feeney, A., Helu, M. M., and Camelio, J. A., 2017. "Toward a lifecycle information framework and technology in manufacturing". Journal of Computing and Information Science in Engineering,

17(2), feb, p. 021010. doi: 10.1115/1.4034132.

[13] Sprock, T., Murrenhoff, A., and McGinnis, L. F., 2017. "A hierarchical approach to warehouse design". International Journal of Production Research, **55**(21), nov, pp. 6331–6343. doi: 10.1080/00207543.2016.1241447.

[14] Sprock, T., and Bock, C., 2017. "Incorporating abstraction methods into system-analysis integration methodology for discrete event logistics systems". In 2017 Winter Simulation Conference (WSC), IEEE, pp. 966–976. doi: 10.1109/WSC.2017.8247847.

[15] Hedberg, T. D., Krima, S., and Camelio, J. A., 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), pp. 11008–11011. doi: 10.1115/1.4034131.

[16] Hedberg, T. D., Krima, S., and Camelio, J. A., 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). doi: 10.1115/1.4042839.

[17] Object Management Group (OMG), 2015. Unified modeling language (UML) v2.5. url: http://www.omg.org/spec/UML/2.5/.

[18] Object Management Group (OMG), 2015. Systems modeling language (SysML) v1.4. url: http://www.omg.org/spec/SysML/.

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

[20] International Standards Organization, 2012. Industrial automation systems and integration – JT file format specification for 3D visualization.

[21] Johnson, D., and Speicher, S., 2013. Open Services for Lifecycle Collaboration Core Specification Version 2.0. url: http://open-services.net/bin/view/Main/OslcCoreSpecification.

[22] International Standards Organization, 2012. Industrial automation systems and integration – product data representation and exchange – part 239: Application protocol: Product life cycle support.

[23] West, T. D., and Blackburn, M., 2017. "Is digital thread/digital twin affordable? a systemic assessment of the cost of DoD's latest manhattan project". In Procedia Computer Science, Vol. 114, Elsevier Science Publishers B. V., pp. 47–56. doi: 10.1016/j.procs.2017.09.003.

[24] Kraft, E. M., 2016. "The Air Force Digital Thread/Digital Twin - Life Cycle Integration and Use of Computational and Experimental Knowledge". In 54th AIAA Aerospace Sciences Meeting, 2016, January 4, 2016 - January 8, 2016. American Institute of Aeronautics and Astronautics Inc, AIAA, San Diego, CA, United states.

[25] Wardhani, R., and Xu, X., 2016. "Model-based manufacturing based on STEP AP242". In 12th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA), pp. 1–5. doi: 10.1109/MESA.2016.7587187.

[26] Bajaj, M., and Hedberg Jr, T., 2018. “System lifecycle handler – spinning a digital thread for manufacturing”. In 28th Annual INCOSE International Symposium, Wiley, pp. 1636–1650. doi: 10.1002/j.2334-5837.2018.00573.x.

[27] Li, J., Tao, F., Cheng, Y., and Zhao, L., 2015. “Big data in product lifecycle management”. International Journal of Advanced Manufacturing Technology, **81**(1-4), oct, pp. 667–684. doi: 10.1007/s00170-015-7151-x.

[28] Shilovitsky, O., 2013. “PLM and data management in 21st century”. In TechSoft3D Tech Talk, SlideShare.net.

[29] Hedberg Jr, T. D., Barnard Feeney, A., and Camelio, J. A., 2017. “Towards a diagnostic and prognostic method for knowledge-driven decision making in smart manufacturing technologies”. In Disciplinary Convergence in Systems Engineering Research, A. M. Madni, B. Boehm, R. G. Ghanem, D. Erwin, and M. J. Wheaton, eds. Springer International Publishing, ch. 60.

[30] Simons, R., 2013. Levers of Control: How Managers Use Innovative Control Systems to Drive Strategic Renewal. Harvard Business Press.

[31] Rossberg, J., 2009. “Application lifecycle management”. In Pro Visual Studio Team System Application Lifecycle Management. Apress, Berkeley, CA, ch. 2, pp. 1–319.

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

[33] Bajaj, M., Zwemer, D., Yntema, R., Phung, A., Kumar, A., Dwivedi, A., and Waikar, M., 2016. “MBSE++ – foundations for extended model-based systems engineering across system lifecycle”. In INCOSE International Symposium, Vol. 26, Wiley-Blackwell, pp. 2429–2445. doi: 10.1002/j.2334-5837.2016.00304.x.

[34] Bajaj, M., Backhaus, J., Walden, T., Waikar, M., Zwemer, D., Schreiber, C., Issa, G., and Martin, L., 2017. “Graph-based digital blueprint for model based engineering of complex systems”. In INCOSE International Symposium, Vol. 27, Wiley-Blackwell, pp. 151–169. doi: 10.1002/j.2334-5837.2017.00351.x.

[35] Kahn, R., and Wilensky, R., 2006. “A framework for distributed digital object services”. International Journal on Digital Libraries, **6**(2), apr, pp. 115–123. doi: 10.1007/s00799-005-0128-x.

[36] Sun, S., Lannom, L., and Boesch, B., 2003. Handle system overview. Tech. rep., The Internet Society, Network Working Group, nov. doi: 10.17487/rfc3650.

[37] Sun, S., Reilly, S., and Lannom, L., 2003. Handle system namespace and service definition. Tech. rep., The Internet Society, Network Working Group, nov. doi: 10.17487/rfc3651.

[38] Sun, S., Reilly, S., Lannom, L., and Petrone, J., 2003. Handle system protocol (ver 2.1) specification. Tech. rep., The Internet Society, Network Working Group, nov. doi: 10.17487/rfc3652.

[39] International Standards Organization, 2012. Information and documentation – digital object identifier system.

[40] International DOI Foundation, 2018. Key facts on digital object identifier system. url: http://www.doi.org/factsheets/DOIKeyFacts.html.

[41] Fischer, A., and Arthurs, G., 2015. INCOSE 2015 MBSE workshop breakout session.

url: http://www.omgwiki.org/MBSE/doku.php?id=mbse:incose_mbse_iw_2015:breakout_out_session_model_lifecylce_mgmt.

[42] Hedberg Jr, T., Sharp, M., Maw, T., Rahman, M., Jadhav, S., Whicker, J., Feeney, A. B., and Helu, M., 2019. "Design, manufacturing, and inspection data for a three-component assembly". Journal of Research of National Institute of Standards and Technology, **124**(124004). doi: 10.6028/jres.124.004.

[43] American Society of Mechanical Engineers, 2012. Digital Product Definition Data Practices.

[44] 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.

[45] MTConnect Institute, 2014. MTConnect Standard. url: http://www.mtconnect.org/media/39542/mtc_part_1_overview_v1.3.pdf.

[46] Dimensional Metrology Standards Consortium, 2014. Part 1: Overview and Fundamental Principles in Quality Information Framework (QIF) – An Integrated Model for Manufacturing Quality Information. url: http://qifstandards.org/.

[47] Francis, N., Taylor, A., Green, A., Guagliardo, P., Libkin, L., Lindaaker, T., Marsault, V., Plantikow, S., Rydberg, M., and Selmer, P., 2018. "Cypher: An Evolving Query Language for Property Graphs". In Proceedings of the 2018 International Conference on Management of Data - SIGMOD '18, ACM Press, pp. 1433–1445. doi: 10.1145/3183713.3190657.

[48] Adolphy, S., Grosser, H., Kirsch, L., and Stark, R., 2015. "Method for automated structuring of product data and its applications". In Procedia CIRP, Vol. 38, Elsevier, pp. 153–158. doi: 10.1016/j.procir.2015.07.063.

[49] Hedberg Jr, T. D., Helu, M. M., and Sprock, T., 2018. "A standards and technology roadmap for scalable distributed manufacturing systems". In Proceedings of the 2018 Manufacturing Science and Engineering Conference, American Society of Mechanical Engineers (ASME). doi: 10.1115/MSEC2018-6550.

[50] Brandl, D., 2013. "Drowning in data, starved for information". Control Engineering.

[51] Deo, N., 2016. Graph theory with applications to engineering & computer science. Dover Publications, Inc., Mineola, New York.

[52] Miller, A. M., Hartman, N., Hedberg Jr, T., Barnard Feeney, A., and Zahner, J., 2017. "Towards identifying the elements of a minimum information model for use in a model-based definition". In ASME 2017 12th International Manufacturing Science and Engineering Conference collocated with the JSME/ASME 2017 6th International Conference on Materials and Processing – Volume 3: Manufacturing Equipment and Systems, Vol. 3, ASME, p. V003T04A017. doi: 10.1115/MSEC2017-2979.

List of Tables

List of Figures

List of Algorithms

List of Listings