EasyChair Preprint

№ 4653

# Creating a Traceable Product Story in Manufacturing Supply Chains Using IPFS

Peter Altmann, Abdul Ghafoor Abbasi, Olov Schelén, Karl Andersson and Morteza Alizadeh



November 26, 2020

# Creating a Traceable Product Story in Manufacturing Supply Chains Using IPFS

Peter Altmann*, Abdul Ghafoor Abbasi*, Olov Schelén†, Karl Andersson†, and Morteza Alizadeh†

* *Department of Industrial Systems*
*RISE Research Institutes of Sweden AB*
SE-16440 Kista, Sweden
Email: peter.altmann@ri.se

† *Department of Computer Science, Electrical and Space Engineering*
*Luleå University of Technology*
SE-93187 Skellefteå, Sweden
Email: olov.schelen@ltu.se

***Abstract*—Evolving traceability requirements increasingly challenge manufacturing supply chain actors to collect tamper-proof and auditable evidence about what inputs they process, in what way these inputs are used, and what the resulting process outputs are. Traceability solutions based on blockchain technology have shown ways to satisfy the requirements of creating a tamper-proof and auditable trail of traceability data. However, the existing solutions struggle to meet the increasing storage requirements necessary to create an evidence trail using manufacturing data. In this paper, we show a way to create a tamper-proof and auditable evolving product story that uses a decentralized file system called the InterPlanetary File System (IPFS). We also show how using linked data can help auditors derive a traceable product story from such an accumulating evidence trail. The solution proposed herein can supplement existing blockchain-based traceability solutions and enable traceability in global manufacturing supply chains where forming a consortium incurs prohibitive costs and where storage requirements are high.**

***Index Terms*—Blockchain, Decentralized Storage, Manufacturing Supply Chain, Traceability**

## I. Introduction

Customer awareness, regulators, and international standardization efforts increasingly challenge downstream manufacturing companies to responsibly source their raw materials. In the case of global manufacturing supply chains, responsible sourcing necessitates a traceability system [1]. Traceability involves capturing data about a complex web of interactions between highly heterogeneous actors; actors that both adapt their operational practices to their local contexts and subject material flows to destructive refinement processes that make durable tagging difficult.

The cobalt manufacturing supply chain (CMSC) shown in Fig. 1 is a salient example of a context where traceability, from raw material to end product, remains challenging. A CMSC can: 1) include thousands of actors from different geopolitical areas, 2) involve activities that are hard to track (e.g., aggregation of inputs, refinement, processing), and 3) consist of several manufacturing steps that produce batches of output with limited distinguishability. Consequently, traceability in a CMSC is hampered by disparate repositories, missing data, poor interoperability, a lack of trust, and high variability in actors' ability to digitally track their manufacturing activities.

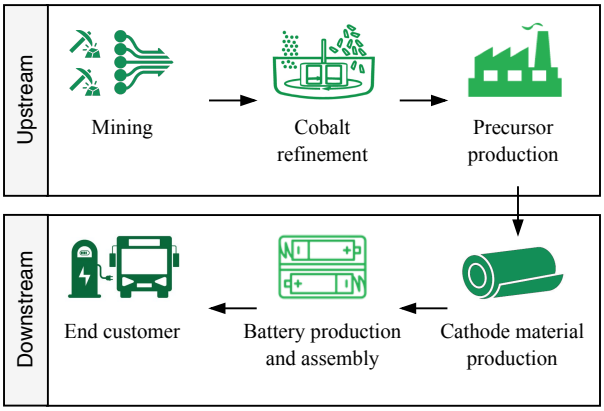


Fig. 1. The Cobalt supply chain from mine to electric vehicle.

Researchers interested in supply chain traceability have carefully examined blockchain attributes like durability, transparency, immutability, decentralization, and verifiability in various industry contexts [2]–[6]. To realize the benefits of these attributes, efforts have been made to promote a shared understanding of data using vocabularies [3] and ontologies [7], durably tag objects and secure reliable data inputs [8], safeguard confidentiality [3], [4], [9], [10], and create digital representations of assets and liabilities in manufacturing supply chains. Particularly relevant for this paper is the work on digital representations of manufacturing chains where raw materials and goods are transformed [6], [9] and where aggregation, mixing, and disaggregation is common [5]. In such manufacturing contexts, traceability solutions either:

1) add records on internal procedures to regular transactions [9],
2) use smart contracts that represent transformation processes as token recipes [6], or
3) model transformation processes as state changes using DAG-based tokens [5]

These traceability solutions leverage the possibility to include a limited amount of arbitrary data in blockchain transactions. In so doing, these solutions aptly ensure *transactional* traceability and represent an important step toward responsible raw materials sourcing.

Because blockchain transactions must adhere to protocol rules (storage of arbitrary data, validation, block structure, etc.), transactional traceability necessitates a level of agreement among supply chain actors. For instance, actors are often required to establish a consortium and, among other things, define roles and membership control [2], [3], [8], [9], [11], decide on a set of rules to automatically enforce [6], [11], and specify data entry types [2], [8]. These decision outcomes can be conceptualized as elements in an agreement set. The alignment process required to form such a set, and then enforce these agreements, can incur prohibitive economic costs [12], [13]. As a consequence of these costs, transactional traceability solutions may not work in global manufacturing supply chains like the CMSC where the agreement set is costly. In such contexts, the design of a traceability solution must emphasize reducing agreement costs among participating actors alongside other blockchain attributes.

To reduce these agreement costs, the solution proposed herein adopts a traceability approach that is *information-centric* (as opposed to transactional). An information-centric approach emphasizes the accumulative creation of a product story derived from manufacturing-related data [3], [7], [8]. This approach derives traceability information from the relationships between process inputs, process activities, and process outputs [6] (as opposed to using transaction data as a proxy for a chain of custody). This emphasis on manufacturing-related data means that the requirements for an information-centric traceability solution differ from a transactional solution in two important ways.

Firstly, a transactional approach stores traceability data primarily on a blockchain. In contrast, the data storage requirements of an information-centric approach far exceed the current storage capabilities of all popular blockchain solutions. Because of blockchains' poor storage scalability and high storage costs, the traceability solution proposed herein aims to store manufacturing data on a decentralized peer-to-peer storage network.

Secondly, a transactional approach relies on an immutable ordered record of balance transfers between participants as a proxy for traceability. Here, a consensus protocol determines the precise ordering of balance transfers or state updates, the rules for including transfers or state updates into a distributed ledger, and the global state of the ledger. In contrast, a product story is created using data elements where each element, and their relationships, is immutable and content addressed in isolation. Supply chain actors can store and retrieve a set of manufacturing data in a decentralized peer-to-peer storage network by locating peers using a distributed hash table (DHT) and exchange these data using a data exchange protocol. Each actor can then organize these data to evidence claims of manufacturing events and manufacturing processes that are relevant to creating an auditable product story.

Note that information-centric traceability does not replace the need for transactional traceability. The two are supplemental insofar that a former emphasizes the ability to trace information related to the creation of a product; the latter tracks ownership transfers. Their respective uses are contextual and their benefits differ. The solution presented herein is designed for contexts where the agreement set and its enforcement incur prohibitive costs and where storage requirements are high.

The key features of the work presented herein are:

1) a way to format evidenced claims related to products into a document,
2) a way to organize these evidenced claim documents into a directed acyclic graph (DAG) so that an accumulative evidence trail related to a product story emerges, and
3) using the IPFS peer-to-peer storage network to store the emerging product story in a way that enables traceability audits by path traversing the evidenced claims.

Point one and two relate to the need to use manufacturing data to evidence traceability claims instead of using transactions as a traceability proxy. Point three relates to the storage options that IPFS affords.

The rest of the paper is organized as follows. Section II introduces the CMSC context and presents some challenges with a transactional traceability approach. Section III presents the concept and design of an information-centric traceability solution. Section IV contrasts the proposed solution with a transactional approach to traceability. Section V concludes the paper, suggests ways to validate the proposed approach, and outlines possible directions for future research.

## II. Traceability in manufacturing supply chains

A focus on transactional traceability emphasizes the use of commercial transactions between CMSC actors as a proxy for material flows. This proxy is not suitable in complex and refinement heavy manufacturing supply chains like the CMSC [1]. The CMSC consists of a myriad of actors, who both engage in dynamic exchange relations and refine inputs to oftentimes chemically changed outputs, who operate in a ever changing regulatory local context, and who face increasing pressure from downstream actors seeking transparency in their supply chains. Transactional traceability must therefore be supplemented by data evidencing input material and energy, the transformation process, and the resulting output.

Table I provides an representative overview of upstream CMSC actors, their numbers, and steps involved in the CMSC along with examples of data that can be used to evidence a product story. Note that the activity set, and therefore the data available for evidencing a claim, varies (especially at the mine level). Forming claims based on activity sets and evidencing

TABLE I
UPSTREAM ACTORS, COMMON MANUFACTURING ACTIVITIES THAT MODIFY MATERIALS, AND EXAMPLES OF DATA FOR EVIDENCING CLAIMS RELATED TO A PRODUCT STORY.

| Actor [count] | Activity | Data evidencing claim |
|---|---|---|
| Mine [>100] | Crushing | Facial recognition |
| | Washing | Location of site |
| | Separation | Site and merchant license |
| | Screening | Fluorescent spectral lines |
| | Subdivision | Processing authorization |
| | | Weight |
| | | Date |
| Intermediary [>50] | Aggregation | Bag tags (QR code) |
| | | Transport route |
| | | Merchant license |
| | | Mass balance |
| | | Date |
| Depots [<50] | Disaggregation | Bag tags (QR code) |
| | Aggregation | Merchant license |
| | Concentration | Processing authorization |
| | | Material usage |
| | | Mass balance |
| | | Location |
| | | Date |
| | | Input and output materials |
| Processors [<10] | Pyrometallurgy | Chemical process |
| | Hydrometallurgy | Energy usage |
| | | Material usage |
| | | Elapsed time |
| | | Mass balance |
| | | Location |
| | | Input and output materials |

these claims with data, allows mining level actors in the CMSC to create a product story based on activities they know.

Contrast the emphasis on activity in Table I with the emphasis on recipes with known inputs and outputs in related work [5], [6]. An emphasis on recipes and/or tokenized asset representations is difficult for upstream actors, who rely on human activity as an input, lack effective ways to meaningfully describe their process outputs (e.g., limited to metrics like weight when a meaningful description would necessitate chemical composition), and cannot establish clear relationships between inputs and manufacturing outputs required for a transactional solution focused on recipes and/or tokens.

Note that transactional traceability and recipes become increasingly relevant as actors undertake more advanced processing steps, i.e., pyrometallurgy, hydrometallurgy, or electrometallurgy (done downstream). These actors process energy and material inputs into refined outputs in batches according to a well defined chemical process. As such, these actors could potentially rely on tokenized asset representations, provided that they can agree with their downstream counterparts on the specifics of the traceability solution.

The traceability solution proposed in this paper does not assume, but is compatible with, any existing transaction focused solution that is able to record at least 32 bytes of arbitrary data in a transaction. Even when a transaction focused solution is absent, processing actors can still use data related to energy and material flows to evidence manufacturing claims. Therefore, the solution presented herein aims to supplement existing transaction focused traceability solutions and considers the prohibitively high costs related to data storage and social agreement in complex global manufacturing chains. The solution, and its components, are presented next.

## III. CONCEPT AND DESIGN

The following subsections detail the concept and design components. All examples are illustrated with CLI commands executed on a Raspberry Pi 4B with 8GB of ram running Raspberry Pi OS (a Debian based operating system optimized for the Raspberry Pi hardware [14]), and go-ipfs v0.6.0 (the main implementation of IPFS [15]).

### A. Traceability data on the InterPlanetary File System

The paper proposes an information-centric traceability solution based on IPFS for global manufacturing supply chains. An in-depth examination of IPFS is available in [16], [17]); here, we detail only the key design features that make IPFS particularly relevant for a decentralized information-centric traceability solution that requires limited social agreement.

As a decentralized peer-to-peer distributed file system, IPFS provides storage scalability and high-availability by replication (i.e., there is no single point of failure). It adopts a content-addressed block storage model to distribute large amounts of versioned data across a network of nodes. Nodes connected to IPFS use the cryptographic hash of a public key, created with S/Kademlia, to create an identity, `NodeId`. This identity is used when determining what node should store information about the location of a specific data block. Specifically, when providing a block, a node will look for the `NodeId` with the lowest exclusive-or (XOR) distance between the bytes that make up the hash identifier of the data block and the bytes in the `NodeId`. These references between data providers and data are stored globally in a Kademlia-style DHT. Nodes also use the DHT together with a file exchange protocol called Bitswap to advertise what data blocks they can provide, what blocks they want, and what blocks they do not want as well to exchange blocks [16].

Using the DHT and Bitswap, nodes form a decentralized peer-to-peer system for storing and distributing data. On top of this, IPFS stores tamper-proof content (both data elements and hyperlinks) and addresses this content based on content hashes using a type-length-value format called `multihash`. Note that IPFS models all data as part of the same Merkle DAG (a Merkle tree without balance requirements and each node can carry a payload [18]), which makes it easy to append data to an existing branch. The ability to extend the tree supports the concept of accumulating manufacturing data into an evolving product story where each iteration can be referenced in a blockchain transaction for the purposes of event ordering and double spending prevention.

Together, the above mentioned design features allow data—used to for instance evidence responsible sourcing claims—to be immutably stored by, and shared between, CMSC actors with seamless scaling, replication and fail-over across nodes without a central authority or single points of failure.

### B. Auditing a product story

To enable product story audits in the CMSC, all claims objects are formatted as JSON-LD, a JSON based format to serialize linked data [19]. Each JSON-LD object contains a `@context` that maps keys to internationalized resource identifiers in order to avoid ambiguity. The solution presented herein leverages this `@context` part of a JSON-LD object to enable traceability auditors to attribute meaning to keys used in evidenced claims documents. Consequently, actors need not seek agreement around definitions and meaning, only on using JSON-LD to organize manufacturing data and clarify meaning.

A JSON-LD object is then encoded with Concise Binary Object Representation (CBOR) [20] into linked data objects referred to as InterPlanetary Linked Data (IPLD). These objects are, as aforementioned, content addressable and can contain both data and traversable links to other IPLD objects.

Using IPLD objects, it is for instance possible to link a manufacturing process input to its corresponding IPLD object. Information about the manufacturing process itself can also be linked to a corresponding IPLD object. Similarly, process outputs can have corresponding IPLD objects. As IPLD objects are referenced and linked, the DAG grows and these DAG nodes contain evidenced claims (or links pointing to such claims) related to the manufacturing process, its inputs, and/or its resulting outputs.

Note that at each manufacturing step, the manufacturing actor extends the existing Merkle DAG by adding their own evidenced claims. Each addition changes the Merkle DAG and a new Merkle root is formed. It is this Merkle root, i.e., the block that contains links to all other blocks, that acts as a claims root that details how these manufacturing data can be used to derive a product story. The resulting Merkle DAG can provide an evidenced traceability solution in the form of a collection of tamper-proof evidenced claims that can be audited by traversing the Merkle DAG.

### C. Evidenced claims

In the example provided herein, traceability information contains data about the: 1) entities, 2) agents, and 3) activities related to each manufacturing step. The PROV namespace [21] defines an entity as "a physical, digital, conceptual, or other kind of thing with some fixed aspects." Relatedly, an agent "bears some form of responsibility for an activity taking place, for the existence of an entity, or for another agent's activity." Finally, an activity "is something that occurs over a period of time and acts upon or with entities; it may include consuming, processing, transforming, modifying, relocating, using, or generating entities."

Once available, data on entities, agents, and/or activities can be audited to reveal certain properties about a manufactured product (e.g., if a product's raw materials were responsibly sourced). Note that the solution proposed herein does not specify what data to collect or what evidenced claims an actor must make. Rather, actors are encouraged to collect data on a best effort basis with tools available to them in their local contexts. For instance, at a mine site, a smartphone can be used to collect data for facial recognition, perform identity checks, and record location data. Additionally, activity type can be recorded together with elapsed time and activity output.

While actors are encouraged to input data evidencing claims on a best effort basis, each evidenced claim should contain:

1) a description of at least one agent, entity, or activity,
2) at least one claim related to that agent, entity, or activity, and
3) a referenced resolvable resource identifier with supporting data to evidence what is claimed in 2).

If these three requirements are met, it becomes possible to assess if the evidenced claim is within a given tolerance range.

Note that an actor can evidence several claims with the same data (or using a subset thereof) and can combine these evidenced claims in ways that support the creation of an auditable product story. The role of published guidelines in determining what claims are meaningful to evidence is discussed next.

### D. Guidelines

A CMSC actor can refer to guidelines to learn what claims are meaningful and how to evidence them. For instance, in the context of the CMSC, the Responsible Minerals Initiative (RMI) has produced guidelines for how to perform responsibly sourced minerals due diligence [22]. The RMI guidelines contain definitions, data attributes, and categorizations that are useful for detailing the context of an evidenced claims object.

In order to exemplify how the RMI guidelines can be included in the set of rules used for interpreting a JSON-LD document, the definitions and categorization specified in the RMI guidelines were formatted as a JSON-LD object and added to IPFS. The resulting content identifier (CID) is a 46 character string (can be up to 113 characters when not using default parameters), which can be path traversed using `$ ipfs dag get <CID>/<path>`. For instance, the path `/DefinedTerm/6/description` returns the RMI's definition of the activity type "extraction." Relatedly, the path `/identitySystem` returns an array of objects specifying the way RMI classifies what roles CMSC actors can have and the activities associated with these roles.

Using the above, a CMSC actor is now able to self-select into a role, reference the RMI guidelines classification for that role, and evidence it with data on e.g., manufacturing activities corresponding to that role. Other CMSC actors are free to audit these data and assess the classification but need neither agree on a set of roles nor a classification mechanism for these roles.

### E. Evidenced claims on IPFS

In the solution presented herein, an evidenced claim formatted as JSON-LD is encoded with CBOR into a traversable IPLD object. The possible content of such an object, i.e., a claims document, is illustrated in Fig 2. The claim document details:

- the `@context` object that maps keys to schemas.
- a `@graph` object with an array of objects relevant for the claim.

- an entity object for raw material *input 1* with node identifier `ipfs:<CID_input1>`
- an entity object for raw material *input 2* with node identifier `ipfs:<CID_input2>`
- an entity object for manufactured output *output A* with node identifier `ipfs:<CID_outputA>`
- an agent object *miner 1* with node identifier `ex:miner1`
- an activity object *extraction* with node identifier `ipfs:<CID>/<path_extraction>`
- a claim object `Usage` specifying a relationship between an activity and two entities
- a claim object `Derivation` specifying entity relationships
- a claim object `Association` stating that an activity was performed by an agent
- a claim object `Generation` stating that an entity was generated by an activity
- a resolvable resource identifier, accessible with the key `ex:evidence`, for data evidencing the above listed claim object.

The example claim in Fig. 2 is added to IPFS using `ipfs dag put`. The claim then becomes addressable using its unique CID.

Note that it is possible to structure these claims differently than illustrated in Fig. 2. It is possible also that a claim is formatted in something else than CBOR and that a resource identifier in a claims object resolves to an object encoded in another format (e.g., raw data stored in an IPLD block). The example claim in Fig. 2 simply demonstrates how an actor could make an evidenced claim on IPFS and that such a claim can be content addressed, and its objects path traversed, through CIDs.

Each CMSC actor will likely produce more than one evidenced claim document and must, therefore, structure these evidenced claims documents into a traceable product story.

### *F. Creating a traceable product story*

As CMSC actors engage in activities, they capture data related to these activities and bundle these data into evidenced claims. As above shown, these claims are then content addressable on the IPFS network and can be path traversed.

Fig. 3 depicts how actors can use guidelines to inform the construction of evidenced claims. Fig. 3 depicts also how data and claims can be organized into a `claimRoot`, which is the hash root of the Merkle DAG containing all the evidenced claims about a particular product. An actor with access to the Merkle DAG root, which links all evidenced claims, can now attempt to retrieve the linked data to derive the product story.

Note how a traceable product story does not replace the need to trace transactions; the two supplement each other. When the output of a manufacturing process changes custody (e.g., when a depot, $A_{dep}$, sells concentrated ore output to a processor, $A_{pro}$) a digital record of the transaction, $tx_n$, can be made. If $tx_n$ can fit additional arbitrary data, then $tx_n$ can reference the root of the evidenced claim, $claimRoot_{dep}$, which contains the hitherto evidenced claims. A many to one relationship between multiple upstream actors and a single downstream actor is trivial to model using a Merkle DAG simply by creating a new IPLD object referencing each upstream actor's `claimRoot`. The recipient, $A_{pro}$, having received $claimRoot_{dep}$, will further refine the concentrated ore output and extend the associated Merkle DAG with these manufacturing data. Consequently, $A_{pro}$ will collect additional manu-

```
{
"@context": {
  "ipfs": "ipfs://",
  "ex": "http://example.org/",
  "@vocab": "https://www.w3.org/ns/prov#"
},
"@graph": [{
  "@type": "Entity",
  "@id": "ipfs:<CID_input1>"
  },{
  "@type": "Entity",
  "@id": "ipfs:<CID_input2>"
  },{
  "@type": "Entity",
  "@id": "ipfs:<CID_outputA>"
  },{
  "@type": "Agent",
  "@id": "ex:miner1"
  },{
  "@type": "Activity",
  "@id": "ipfs:<CID>/<path_extraction>"
  },{
  "@type": "Usage",
  "activity": ["ipfs:<CID>/<path_extraction>"],
  "entity": [
    "ipfs:<CID_input1>",
    "ipfs:<CID_input2>"
    ],
  "ex:evidence": "ipfs:<CID to Usage evidence>"
  },{
  "@type" : "Derivation",
  "generatedEntity": ["ipfs:<CID_outputA>"],
  "usedEntity": [
    "ipfs:<CID_input1>",
    "ipfs:<CID_input2>"
    ],
  "ex:evidence": "ipfs:<CID to Derivation data>"
  },{
  "@type": "Association",
  "activity": ["ipfs:<CID>/<path_extraction>"],
  "agent": ["ex:miner1"],
  "ex:evidence": "ipfs:<CID to Association data>"
  },{
  "@type": "Generation",
  "entity": ["ipfs:<CID_outputA>"],
  "activity": ["ipfs:<CID>/<path_extraction>"],
  "ex:evidence": "ipfs:<CID to Generation data>"
  }]
}
```

Fig. 2. An illustration of an evidenced claim's structure.

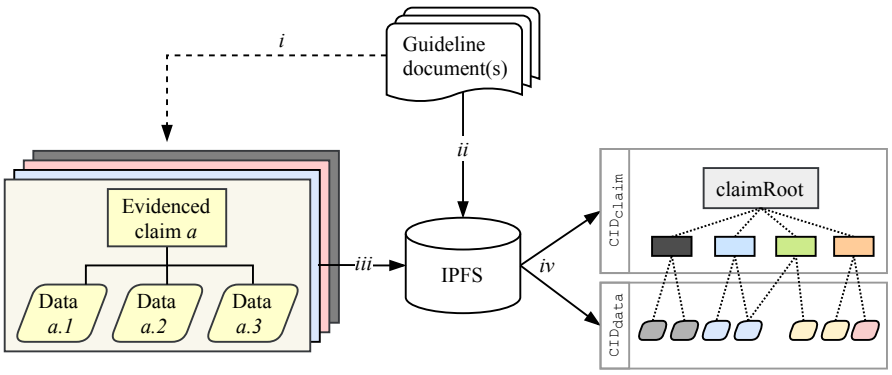


Fig. 3. The structure of evidenced claims is informed by guidelines (e.g., the RMI) (*i*). These guidelines are added to IFPS (*ii*) together with the claims and the supporting evidence (*iii*). Supply chain actors create a claimRoot with their evidenced claims (*iv*). Subsequent actors in the manufacturing supply chain can extend claimRoots with new claims branches to develop a product story.

facturing data to evidence meaningful claims, create evidenced claims documents, and extend the existing $claimRoot_{dep}$ with these documents to form the new $claimRoot_{pro}$. This new Merkle DAG root contains links to an updated product story, which can be audited for traceability purposes on its own, or supplement a transaction based traceability solution where claims verification is computationally treatable. This process can be repeated for each subsequent actor in the entire CMSC.

## IV. Comparative analysis

In this section, the information-centric traceability approach and the transactional traceability approach is contrasted with regards to 1) data storage, 2) the set of social agreement necessitated, and 3) cost factors.

### A. Data storage

Transaction focused blockchain protocols like Bitcoin were not designed with data storage in mind. Nevertheless, several methods for storing arbitrary data on blockchains have emerged; each a trade-off between various properties like integrity, availability, cost, and storage amount [23]. For instance, certain storage methods are only available to a subset of network participants (e.g., the 100 bytes in the `coinbase` that Bitcoin miners can use). Other methods are deprecated (e.g., forcing nodes to store data in the set of unspent transaction outputs (UTXO) by creating transactions that are unspendable) or malleable (e.g., altering the arbitrary data is possible if it does not impact the functional transaction data). Moreover, common methods such as Bitcoin's `OP_RETURN` does not guarantee permanent data storage as these transactions are provably unspent and not part of the UTXO set, which makes it possible to prune them [24]. These facts notwithstanding, storage costs are compared next.

Storage costs on Bitcoin range between 21.26 and 79.25 Satoshi per Byte of arbitrary data depending on the storage method [23]. The only method that is integrity preserving, not possible to prune, and that does not rely on the UTXO costs 23.72 Satoshi per Byte[1] of arbitrary data. Relatedly, the total storage cost on Ethereum depends on the type of activity involved (e.g., contract creation or arbitrary data storage) and the specific implementation [6]. According to Ethereum's yellow paper [25], raw storage of arbitrary data using the `SSTORE` opcode costs 20'000 gas for every 256-bit word, which translates into 640'000 gas for 1 KB of arbitrary data[2].

In contrast, the information-centric approach presented in this paper relies on IPFS where storage costs are negligible. Assuming that the generated data cannot exist in the actor's existing storage environment, the pinning service Pinata offers pinned file storage for a monthly cost of $0.15/GB [26].

It is worth mentioning that many transactional traceability solutions discuss the option to only store a pointer to supplemental data on the blockchain. For instance, [6] mention peer-to-peer storage networks like IPFS and Swarm to store product information outside of the blockchain. We support such work and hope that the solution presented herein furthers the integration between peer-to-peer storage networks and blockchains.

### B. Required agreement set

In a transactional traceability solution, the set of social agreement required is primarily determined by blockchain governance. Specifically, participating actors must assign authority and responsibility among the participating members (available options are permissionless, consortium-based, and private). Furthermore, actors must agree on block creation rules, transaction validation rules, the consensus mechanism, and how decisions regarding protocol updates are made [27].

As aforementioned, transactional traceability solutions commonly adopt a consortium approach. In such a consortium, the members must collectively agree on cost-sharing, execution and maintenance responsibilities, membership control, and

[1]With a BTC/USD exchange rate of $11803 at the time of this writing, storing 1 KB of manufacturing data costs about $2.87.

[2]At the time of this writing, the ETH/USD rate is $443, and the standard gas cost is 99 Gwei (1 Gwei is $10^{-9}$ ETH). This amounts to $26.89 for every 1 KB of manufacturing data stored.

how to update the governance scheme as the consortium changes [3], [27]. Consortium members must agree also on role definitions, settle on a shared classification mechanism, and agree on how to structure traceability claims.

Relatedly, transactional solutions that include smart contracts require participants to agree on a set of relevant events to express in code, how to evaluate these events, and what the resulting consequences should be. Objectively verifiable and deterministic events (e.g., a delayed flight) may be easy to express in a smart contract. Conversely, subjective events (e.g., a reputation system) may be hard to express and can therefore incur high agreement costs.

In contrast, in an information-centric approach:

1) actors generate and control their own identities,
2) actors self-select into roles and reference the classification scheme as part of a JSON-LD `@context`,
3) membership control is handled locally between subsets of participating actors able to reach an agreement,
4) a traceable product story is created in an blockchain agnostic way and actors can traverse a tree of evidenced claims related to a product, and
5) each actor is encouraged to analyze evidenced claims as their specific needs mandate.

Note how the above characteristics minimize the necessary set of costly social agreements between participants. For instance, an actor can self-select into a role without consulting any other participant. Furthermore, there is no need to agree on definitions, data models, and/or schemas as these can be clarified as part of the JSON-LD document. Relatedly, since the solution is blockchain agnostic, participants need not agree on the underlying ledger but are free to use whatever solution that meets their needs.

### C. Social agreement cost drivers

To guide future traceability work, it is helpful to list some of the cost drivers related to social agreement activities. When these drivers are present, an information-centric traceability approach may be favored. In a complex global manufacturing supply chain like the CMSC, the following drivers contribute to high agreement costs when adopting a transactional approach:

- A dynamic turnover rate that calls for frequent changes to membership control.
- The presence of several distinct regulatory and institutional environments; many of which are in flux.
- A high degree of variability in technical competences among participating actors.
- An inability of upstream actors to described or verify the properties of their productive output (e.g., ore miners can lack ways to determine ore grade, which is a crucial parameter for a manufacturing recipe).
- An inability to agree on how to assign weights to parameters that impact subjective evaluations like trust or reputation.

When the above cost drivers are present, an information-centric approach, such as the one presented herein, may be a suitable option.

## V. Conclusions

This paper proposes an information-centric traceability solution for complex manufacturing supply chains. In contrast to transaction-focused traceability solutions that are commonly employed on blockchains, an information-centric approach encourages participants to use data to evidence claims about their manufacturing activities related to a product. These evidenced claims are structured in an expanding product story tree that downstream participants can extend with their own evidenced claims. When these data, the claims, and their relations are stored as IPLD objects on IPFS, the emerging product story becomes traversable and thus open to audits.

The proposed solution requires little agreement among manufacturing supply chain actors and has very low storage costs. As such, the solution is highly suitable for traceability contexts where the necessary agreement required to deploy transactional traceability solutions incurs prohibitive costs. In such contexts, a validation of the proposed approach is an interesting future direction.

The proposed solution is suitable also for traceability contexts where participants can agree on a transactional traceability solution that covers at least parts of the manufacturing supply chain and where a transaction can store a CID. An interesting future direction here is to explore ways to leverage both blockchain technology and decentralized peer-to-peer file storage networks like IPFS for traceability purposes. Here, research is needed to better understand how to optimally substantiate transactional claims.

## References


[1] S. Vetter and P. Scütte, Eds., *Mapping of the artisanal copper-cobalt mining sector in the provinces of Haut-Katanga and Lualaba in the Democratic Republic of the Congo*. Hannover: Bundesanstalt für Geowissenschaften und Rohstoffe, 2019.

[2] S. A. Abeyratne and R. P. Monfared, "Blockchain ready manufacturing supply chain using distributed ledger," *Int. J. Res. Eng. Technol.*, vol. 5, no. 9, pp. 1–10, 2016.

[3] S. Malik, S. S. Kanhere, and R. Jurdak, "Productchain: Scalable blockchain framework to support provenance in supply chains," in *2018 IEEE 17th NCA Symposium*. IEEE, 2018, pp. 1–10.

[4] T. Mitani and A. Otsuka, "Traceability in permissioned blockchain," *IEEE Access*, vol. 8, pp. 21 573–21 588, 2020.

[5] H. Watanabe, T. Ishida, S. Ohashi, S. Fujimura, A. Nakadaira, K. Hidaka, and J. Kishigami, "Enhancing blockchain traceability with dag-based tokens," in *2019 IEEE Int. Conf. Blockchain*. IEEE, 2019, pp. 220–227.

[6] M. Westerkamp, F. Victor, and A. Küpper, "Tracing manufacturing processes using blockchain-based token compositions," *Digit. Commun. Netw.*, vol. 6, no. 2, pp. 167–176, 2020.

[7] H. M. Kim and M. Laskowski, "Toward an ontology-driven blockchain design for supply-chain provenance," *Int. J. Intell. Syst. Account. Finance. Manag.*, vol. 25, no. 1, pp. 18–27, 2018.

[8] F. Tian, "A supply chain traceability system for food safety based on haccp, blockchain & internet of things," in *2017 IEEE 14th Int. Conf. Services Syst. Services Manage. (ICSSSM)*. IEEE, 2017, pp. 1–6.

[9] K. Biswas, V. Muthukkumarasamy, and W. L. Tan, "Blockchain based wine supply chain traceability system," in *2017 Future Technologies Conference (FTC)*, 2017, pp. 56–62.

[10] M. Lohr, J. Hund, J. Jürjens, and S. Staab, "Ensuring genuineness for selectively disclosed confidential data using distributed ledgers: Applications to rail wayside monitoring," in *2019 IEEE Int. Conf. Blockchain*. IEEE, 2019, pp. 477–482.
[11] S. Malik, V. Dedeoglu, S. S. Kanhere, and R. Jurdak, "Trustchain: Trust management in blockchain and iot supported supply chains," in *2019 IEEE Int. Conf. Blockchain*. IEEE, 2019, pp. 184–193.
[12] R. H. Coase, "The nature of the firm," *Economica*, vol. 4, no. 16, pp. 386–405, 1937.
[13] O. E. Williamson, "Transaction cost economics," *Handbook of industrial organization*, vol. 1, pp. 135–182, 1989.
[14] "Raspberry pi os," https://www.raspberrypi.org/documentation/raspbian/, accessed: 2020-08-20.
[15] "Ipfs distributions," https://dist.ipfs.io/#go-ipfs, accessed: 2020-08-20.
[16] J. Benet, "Ipfs - content addressed, versioned, p2p file system," https://arxiv.org/abs/1407.3561, 2014.
[17] S. Henningsen, M. Florian, S. Rust, and B. Scheuermann, "Mapping the interplanetary filesystem," https://arxiv.org/abs/2002.07747, 2020.
[18] "Ipfs distributions," https://docs.ipfs.io/concepts/merkle-dag/, accessed: 2020-08-20.
[19] "Json-ld 1.1," https://www.w3.org/TR/json-ld/, accessed: 2020-08-20.
[20] "Concise binary object representation (cbor)," https://tools.ietf.org/html/rfc7049, accessed: 2020-08-20.
[21] "The prov namespace," https://www.w3.org/ns/prov, accessed: 2020-10-21.
[22] "Responsible minerals initiative blockchain guidelines second edition," https://rb.gy/e1wnfk.
[23] A. Sward, I. Vecna, and F. Stonedahl, "Data insertion in bitcoin's blockchain," *Ledger*, vol. 3, pp. 1–23, 2018.
[24] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Tech. Rep., 2008.
[25] "Ethereum: A secure decentralised generalised transaction ledger," https://ethereum.github.io/yellowpaper/paper.pdf, accessed: 2020-08-21.
[26] "Pinata cloud pricing," https://pinata.cloud/pricing, accessed: 2020-08-21.
[27] O. Dib, K.-L. Brousmiche, A. Durand, E. Thea, and E. B. Hamida, "Consortium blockchains: Overview, applications and challenges," *International Journal On Advances in Telecommunications*, vol. 11, no. 1&2, pp. 51–64, 2018.