

ADDITIVE OS: AN OPEN-SOURCE PLATFORM FOR ADDITIVE MANUFACTURING DATA MANAGEMENT & IP PROTECTION

Evan P. Diewald*

*Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213

Abstract

The additive manufacturing (AM) digital thread presents unique challenges for data management and security. While proprietary software packages solve many issues, they can be expensive and lacking in customization. Additive OS is an open-source platform for importing, sharing, organizing, and querying AM data. Man-in-the-middle attacks, secure print licensing, and IP theft are addressed using custom smart contracts, ontology is preserved with a NoSQL database and directed acyclic graph (DAG) representations, and peer-to-peer content delivery facilitates low-latency file transfer. The application includes a browser-based graphical user interface, but developers can access the underlying API to invoke sophisticated queries, add functionality, or run the lightweight client on low-resource hardware.

Introduction

Cybersecurity Threats in AM

One of the features that distinguishes additive manufacturing (AM) from other forms of manufacturing is its virtualizable workflow. In this *digital thread*, the fabrication cycle is accompanied by a *digital twin*, which takes the form of design files, build planning parameters, process monitoring data, invoices, and other media. By nature, the process is highly iterative, as part geometries and machine inputs may be optimized in response to simulations or mechanical testing. Given that 3D printed components are particularly common in mission critical applications such as aerospace and defense, it is essential that regulators can confidently trace a part's entire lifecycle. However, the digital thread is vulnerable to a number of attack vectors that challenge conventional audit trails. Existing research has shown that saboteurs can relatively easily embed micro-defects in CAD files [1]–[3], alter machine input files and firmware [4], [5], or manipulate in situ monitoring data, which is increasingly used as a means of process qualification [6]. These subtle changes are non-trivial to detect but can have a drastic impact on a part's mechanical properties. Cybersecurity is further confounded by the decentralization of the AM supply chain, which includes feedstock suppliers, engineering firms, and machine shops.

As a rapid prototyping tool, AM is often used to fabricate sensitive or pre-production designs, making intellectual property (IP) theft a pressing concern for many users. In a survey of experts from industry, academia, and governmental organizations, Kurpjuweit, et. al. found that IP digital rights management and counterfeit prevention were considered top issues in AM cybersecurity, after threats related to sabotage and traceability [7]. One of the more eye-opening results from the survey was an anecdote that, when sharing confidential designs with an external firm, some designers prefer to mail a physical copy of the part rather than digitally share the CAD file. Such examples are indicative of the real-world challenges in establishing trust among distributed stakeholders in the AM supply chain. Typically, proving ownership of novel material

requires legal protection in the form of copyrights or patents. In addition to the lengthy application process and significant legal expenses, this conventional approach often requires public disclosure of early-stage prototypes.

AM and Blockchain

“Blockchain”, or distributed ledger technology (DLT), is a broad term used to describe decentralized peer-to-peer networks of individual nodes that maintain an immutable, append-only record. Proposed state changes, or transactions, are bundled into the atomic units of the data structure, called blocks. Because each block contains the cryptographic hash of the previous block, the integrity of the ledger is validated by calculating each block’s hash in sequence. Blockchains vary in consensus protocols, which are the algorithms that provide an economic incentive for each individual node to keep an accurate and honest version of the current world state. “Proof of Work”, which was popularized by Nakamoto with the introduction of Bitcoin in 2009 [8], requires nodes to compete to “mine” new blocks of transactions by finding the solution to computationally-intense cryptographic puzzles¹. The Ethereum blockchain is currently in transition from Proof of Work to a more efficient protocol known as Proof of Stake². The upcoming launch of “ETH 2.0” aims to increase network capacity and reduce transaction fees without sacrificing security [9]. Other distinctions among DLT frameworks include the privacy of the network and the types of transactions that they support. Blockchains can typically be classified as *private*, meaning they have restricted access (e.g. Hyperledger Fabric), or *public*, meaning they are fully open (e.g. Bitcoin). Consortium chains, maintained by several firms in the same industry, are a compromise between the two extremes. An important innovation established by Ethereum was the introduction of *smart contracts*, which are computer programs that interact with a blockchain through pre-defined functions and business logic. Each transaction with a smart contract is recorded on the shared ledger, providing a resilient audit trace.

Cybersecurity experts in academia and industry recognize that blockchains can be applied to the AM supply chain to increase accountability. Research articles have presented theoretical frameworks illustrating how DLT could secure various aspects of the digital thread. In [10], the authors propose a private ledger to track the lifecycle of parts through a use case for the aerospace industry. A conceptual approach for a blockchain-based AM data management and exchange system was outlined by Papakostas, et. al. [11]. Alkhader et. al. produced a smart contract with functions where designers and printers trace an example development sequence through a trustless interface³ [12]. A sample scenario was demonstrated by calling the functions within the Remix IDE. The “Secure Additive Manufacturing Platform” (SAMPL) defines a general smart contract in which part integrity and copyright protections could be validated from manufacturer to customer using RFID tags [13]. Kennedy, et. al. used lanthanide nanomaterial chemical signatures to print

¹ More specifically, miners attempt to find a “nonce” value that, when appended to the block’s existing data, results in a block hash with a certain number of leading zeros. The number of leading zeros is known as the “difficulty”, and it is adjusted in response to the computing capacity of the network to ensure consistent block times.

² In Proof of Stake, nodes “stake” a large amount of capital in exchange for the rewards associated with mining blocks. Dishonest or unreliable nodes forfeit all or a portion of their initial investment.

³ The smart contract required a third-party “Certificate Authority” to sign off on each transaction.

unique QR codes in the fused deposition modeling process. The codes could be scanned with a smart phone to log processing steps as Ethereum transactions [14].

Commercial solutions are more complete, but they involve proprietary technology and costly licensing agreements. General Electric Additive filed a patent application in 2018 for a distributed ledger system that verifies that the build file and material lot have not been altered before reaching the machine [15]. GE and Microsoft joined efforts on their “TRUEngine” platform, which uses a private fork of the Ethereum network to track components in aircraft engines throughout their lifecycle [16]. Start-ups like Cubichain Technologies have demonstrated a proof of concept to “protect the digital data stream for additively manufactured aerospace titanium parts” [17]. While the details of this confidential technology are limited at this time, Cubichain is built on the MultiChain private blockchain platform (based on Bitcoin Core), whose commercial license is currently priced at \$5,000/year per node [18].

At this time, the authors are not aware of an open-source application that facilitates AM data management in conjunction with smart contracts for independent traceability and IP protection. Commercial platforms lean toward private architectures which are fast and secure but require more expensive infrastructure. Furthermore, as ultimately centralized solutions (nodes are maintained by a single organization), they do not take advantage of the distributed nature of public blockchains. By design, *trusted* networks cannot be independently audited by external parties, such as regulators. For large firms with vertically integrated AM supply chains, this may be acceptable, but many design shops and laboratories outsource powder supply, manufacturing, post-processing, and/or characterization work. One of the most significant benefits of 3D printing is the ability to manufacture *at the point of need*, which requires a reliable source of truth between geographically or organizationally disparate entities. Additive OS balances privacy and performance concerns with the benefits of a fully trustless and objectively certifiable digital thread.

AM Data Management

Under guidance from experts in industry, academia, and national standards institutions, America Makes published a Strategic Guide for AM Data Management and Schema in 2019 [19]. The committee voted “the need for unique, unified data identifiers for AM data” as one the most significant gaps related to AM data management. This convention would allow for more useful analytics, e.g. drawing “cross comparisons” between a given part’s process conditions, characterization data, and mechanical properties [20], [21]. AM facilities often store massive amounts of data in an ad hoc manner, limiting the usefulness of Big Data analytics and machine learning [22]. For users that opt for a unified database, they must choose between costly commercial packages, limited open-access platforms, or home-grown solutions. GRANTA:MI [23] is a proprietary application for importing, storing, and querying AM data. Users choose between a selection of closed-source, pre-defined schemas that include process parameters, mechanical testing data, and URLs to images stored externally. Senvol Database [24] is a compilation of 3D printing machines and materials, meant to aid in selection of, for instance, polymer filament that will meet certain strength requirements. To access the underlying data and application programming interface (API), a license is required. Startups Link3D [25] and Authentise [26] integrate business operations, logistics, and AM workflow tracking within their proprietary manufacturing execution software.

In the academic sector, efforts are mainly focused on establishing an appropriate representation for AM data [27]–[31]. Computational ontologies formalize the inheritance structure of a dataset [32]; they have been successfully applied to domains like medicine and biology to extract meaningful relationships and maintain consistency [33]. This representation - allows data scientists to use advanced deep learning techniques such as graph neural networks, which utilize the adjacency between nodes as an intuitive feature set for regression and classification models [34], [35]. Establishing a common ontology makes the dataset easier to query, and new entries can be imported using existing templates. Some databases and querying languages, including GraphDB [36] and SPARQL [37] are specifically designed to capture semantic structure, but ontologies can also be represented in general-purpose NoSQL platforms such as MongoDB [38], [39].

In terms of web applications and platforms that integrate AM data ontology and workflow management, open-source solutions are significantly more limited than commercial competitors. The most polished application was developed by NIST [28], but new users must apply for special permission, and all data is completely public. As highlighted in the America Makes guide, while sharing insights can enrich collective knowledge, security and privacy remain paramount in most cases.

Methods

Smart Contracts

As opposed to their legal counterparts, smart contracts do not require third-party authentication to establish an immutable and objective audit trace. While smart contracts are most common in financial applications, such as decentralized exchanges [40], [41], they are also used to govern supply chains [42], manage healthcare records, and provide distributed voting mechanisms [43]. Most modern blockchains have support for smart contracts, but a comparison of developer platforms is beyond the scope of this article. Additive OS processes transactions on the Ethereum network [44], which interprets smart contracts written in Solidity, an object-oriented programming language that compiles high-level scripts into machine-readable bytecode [45]. Ethereum has relatively low transaction fees and adequate speeds, and it is by far the most popular blockchain for decentralized applications due to broad developer support and extensive tooling. Importantly, Additive OS decouples transaction signing from the application itself, meaning users do not have to hard-code their private keys into the system. Transactions are signed using Metamask, a browser extension-based Ethereum wallet with over 1M users [46].

Additive OS deploys two smart contracts to address the most pressing AM cybersecurity threats of man-in-the-middle (MITM) attacks and secure design licensing. The primary role of the AMProject smart contract is to provide methods for associating authenticated files with a project. Each Project object is defined by a unique index, its author's ETH address, and an array of File objects corresponding to verified content. Each File contains the SHA3 checksum of an added file and the address of the user who signed the transaction. Thus, the legitimacy of a shared file can be confirmed by comparing its hash to what is stored in the smart contract (Figure 1). The inputs to one-way hashing algorithms like SHA3 cannot be reverse-engineered, ensuring that the

confidential contents of the file are not exposed to the public ledger [47]⁴. Even if Additive OS's database cluster is compromised or deleted, the permanent contract and its associated transactions are not lost. Similar contracts have been demonstrated to secure the AM supply chain against tampering, but the applications were implemented in a toy setting and not integrated in a user interface [12]. In the first use case, Additive OS is used to expose a MITM attack on a turbine blade, similar to the dr0wned threat demonstrated by Belikovetsky, et. al. [1]. However, the same smart contract can be generalized to preserve the entire digital thread through substantiation of process monitoring data, invoices, and test results.

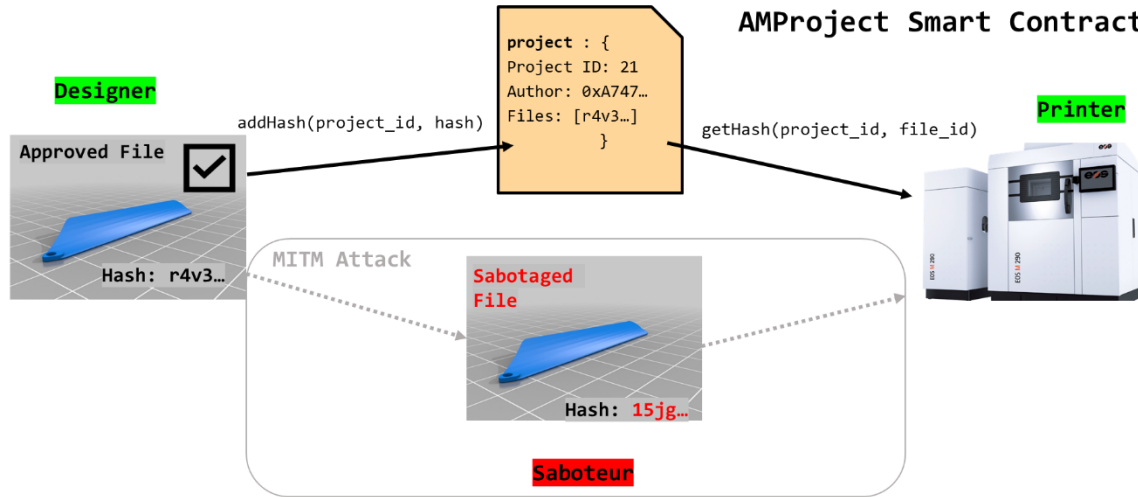


Figure 1. Schematic of the AMProject smart contract, which protects against man-in-the-middle attacks.

The second smart contract, **AMLICENSE**, addresses more subtle threats related to IP rights and print licensing. A new license instance must be defined by the ETH wallet addresses of the licensor and licensee, the checksum of the part's design file, and the number of licensed prints. Like the AMProject smart contract, by storing only the file's hash, the sensitive file contents are not exposed to the public blockchain. Again, this can help protect against MITM attacks, but in this scenario, the intention is to establish ownership of the IP without public disclosure. If the licensee (e.g. a service bureau or fabrication shop) were to copy the design file for themselves or share it with a competitor, the smart contract transaction persists as a timestamped attestation of the original deal. As a trustless interaction, arbiters could verify the transaction contents without relying on the either party's centralized and editable records. While scenarios like this were once a theoretical exercise, blockchains' growing acceptance as a tamperproof audit trail is starting to hold real legal weight; several states have begun to pass legislation accepting smart contract transactions as admissible evidence [48]–[50].

⁴ Moreover, concise checksums keep gas fees low.

When a print is executed by the licensee⁵, they log another transaction that includes an operator ID and the hash of the build report. This produces a link to a digital “certificate of authenticity” that can be carried with the physical part as a QR code or RFID tag. Further down the supply chain, the underlying transaction details can be independently audited by end users, and unauthorized duplicates (i.e. parts without a unique and valid certificate) can be easily identified. The AMLicense smart contract (**Error! Reference source not found.**) demonstrates the benefits of using a public ledger like the Ethereum Mainnet, as opposed to private architectures implemented by some firms [16], [17]. By the time a 3D printed component is installed by a customer, its digital fingerprint may have crossed between several different organizations, from feedstock suppliers and designers to simulation experts and machinists. By logging transactions in a public blockchain, distributed stakeholders (including regulators) have access to a unified audit trail. However, to outside observers, the anonymized hashes are not interpretable. As AM scales, the expenses of drafting legal paperwork for each license add up, at the disadvantage of small shops who do not retain dedicated counsel. Smart contracts provide a cost-effective and

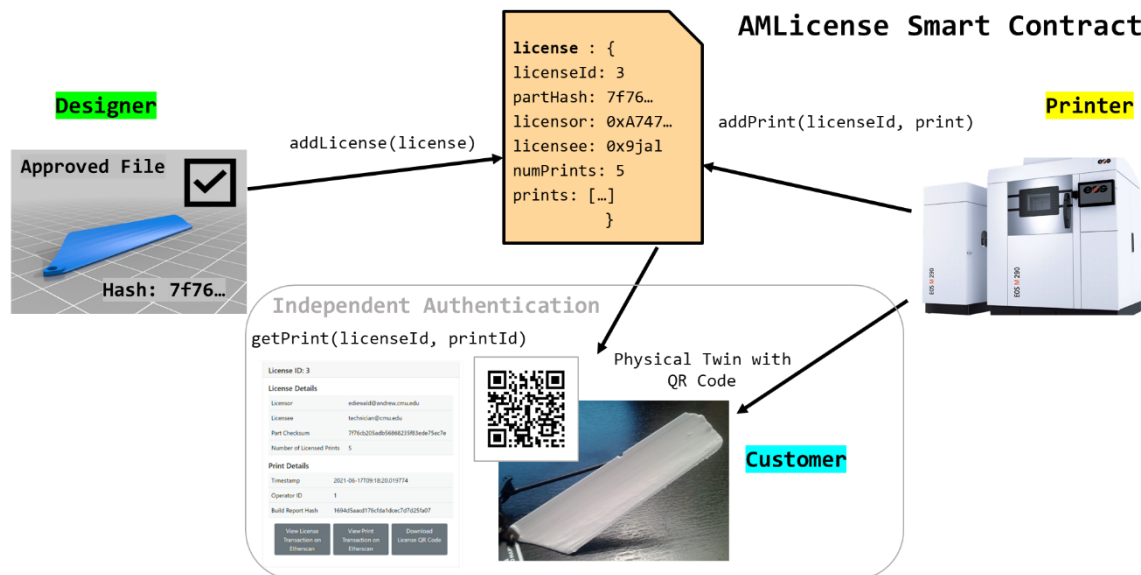


Figure 2. Schematic of the AMLicense contract, which establishes digital IP rights and licensing terms among distributed stakeholders. End users benefit from independently certifiable data provenance.

interpretable alternative.

Additive OS Data Management and Directed Acyclic Graphs

As discussed in the Introduction, AM presents unique data management challenges. The schema must capture a complex and evolving ontology while enforcing a consistent identification structure. To encourage analytics, the database engine should support sophisticated queries and be accessible through popular programming languages like Python, C++, and MATLAB. Finally, the data importation process should be as automated as possible to minimize human error and ensure usability.

⁵ An exception will be raised if the number of prints does exceeds the allotted amount or the transaction is signed by someone other than the licensee or licensor.

As opposed to relational (SQL) databases, which are relatively inflexible, or RDF-based platforms, which have a steep learning curve, Additive OS is driven by MongoDB, a popular open-source framework for NoSQL (i.e. non-relational) databases [38]. Other than the requirement that each entry has a unique identifier, MongoDB imposes little structure on how documents are stored. Complex objects can be inserted and fetched as human-readable JSON files or language-native representations, like Python dictionaries. While the Additive OS API is written in Python, data scientists can execute queries in their language of choice.

Ontology is captured through an interpretable naming convention that allows data to be encoded as directed acyclic graphs (DAGs). *Graphs* express adjacency between vertices through edges, and *directed graphs* specify that edges have a direction from parent to child. Thus, *directed acyclic graphs* are a certain type of directed graph that does not allow for closed inheritance loops [51]. DAGs are not only an intuitive way to represent AM data, but they introduce certain properties that are useful for mathematical and statistical models, like topological ordering, which can be used to find the shortest path between nodes [52]. In clinical studies, DAGs have been used to help identify biases, confounding factors, and causal relationships between risk factors and diseases [53]. Analogous challenges exist in AM, where process-structure-property relationships can be confounded by complex physics.

In Additive OS, the root node of an AM data DAG is a build, which is represented by a unique identifier (UID), e.g. CMU01. This node may include build-specific attributes such as the preheat temperature or links to layerwise images. Leaf nodes that directly inherit the characteristics of the build root (for instance, an individual part) have a UID that includes the build UID, followed by a period delimiter, followed by a secondary index, e.g. CMU01.01. Part objects store parameters such as mechanical testing results or a link to the CAD file. Similarly, to encode direction between, for instance, a part and its infill parameters, the infill parameter object for part CMU01.01 would be CMU01.01.IF, and could contain attributes like laser power or hatch spacing. Build ontologies of arbitrary complexity (including heat treatments, machining, and other post-processing steps) can be represented in similar fashion (Figure 3).

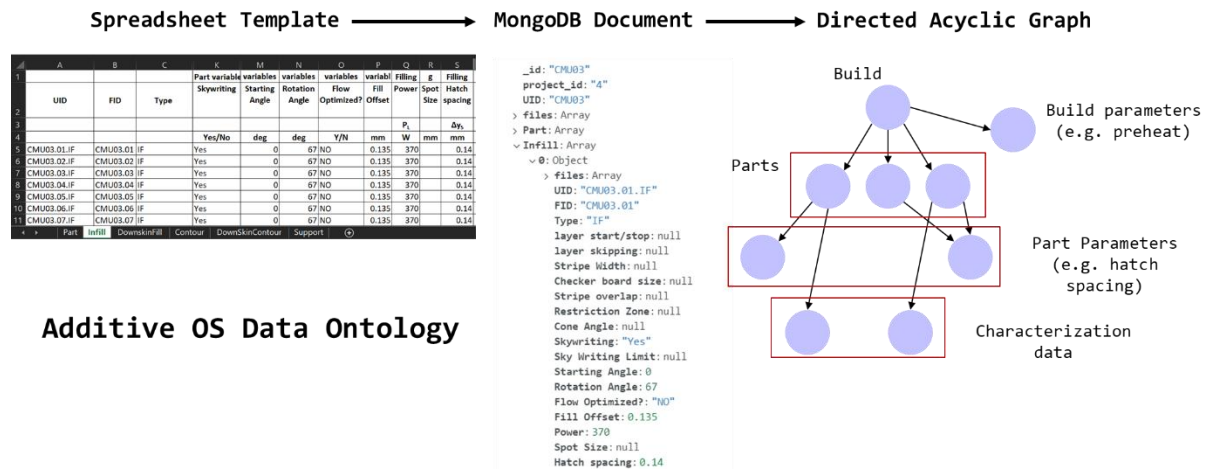


Figure 3. Data ontology in Additive OS. Build data is imported through flexible spreadsheet templates, stored in MongoDB as JSON documents, and retrieved/visualized as DAGs.

Ideally, build data could be directly imported from build preparation files. However, closed-source machines limit the amount of useful information that can be extracted from these documents. Thus, to ensure broad support, Additive OS automatically imports data into MongoDB from flexible, spreadsheet-based templates which were developed as part of a multi-university research project. Once a build tree is imported to a project, process monitoring files, characterization data, and post-processing steps can be attached to the UID that they are associated with. For example, micrographs or yield strength measurements can be attached to individual parts, while passive acoustic data may be attributed to the entire build. Additive OS includes an interactive tool for visualizing the build DAGs, which are generated using the PyVis library [54], and they can also be exported in a format compatible with popular graph neural network packages such as PyTorch and Keras. Under the Data Explorer tab, users can quickly test MongoDB queries on the database before writing code (Figure 4).

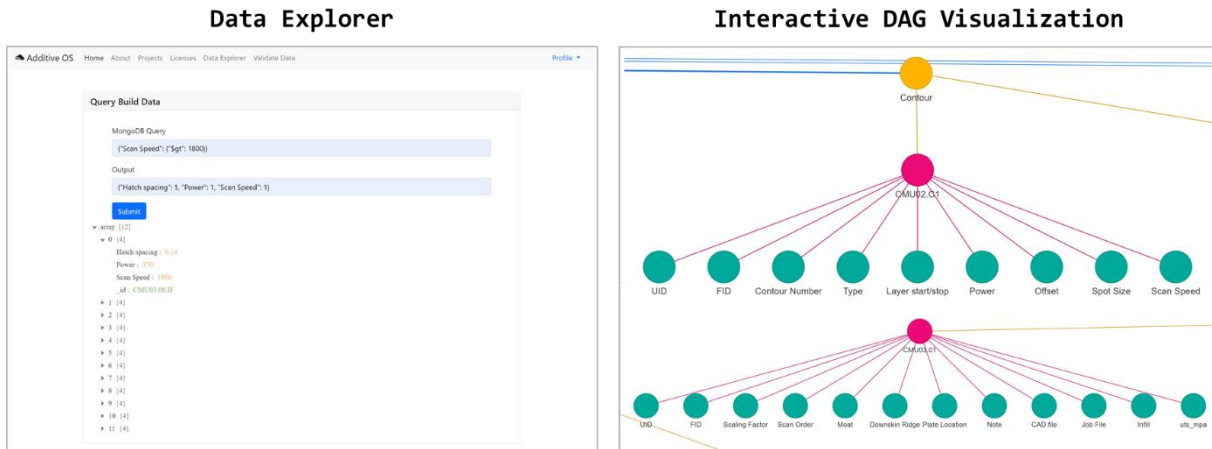


Figure 4. In addition to the Python API, Additive OS provides tools for inspecting AM data and its structure. The Data Explorer (left) provides a testbed for MongoDB queries before writing code. The DAG visualizer (right) is an interactive and tangible representation of build structure.

Peer-to-Peer File Storage

Given the collaborative nature of the AM digital thread, files must have high availability and multiple replications, but security remains paramount. In the cloud computing era, there are many options for reliable, centralized file storage. User-friendly platforms such as Box [55] and Google Drive [56] are popular and intuitive, but difficult to integrate into applications programmatically. Amazon Web Services' Simple Storage Service (AWS S3) buckets [57] expose this functionality, but coverage is regional. If files are not replicated in geographically diverse locations via expensive content distribution networks (CDNs), users can experience high latency when downloading content from monolithic data centers. Most importantly, the aforementioned platforms use *name-based addressing* schemes, meaning a file is retrieved based on its hierarchically-structured location, e.g. `/images/01.png`. While this is a human-readable format, it is imprecise. Files can be updated and saved to the same path, either intentionally or maliciously, leading to unnecessary confusion and security risks.

The InterPlanetary File System (IPFS) is a decentralized, peer-to-peer CDN that resolves many of the issues with centralized cloud storage platforms [58]. Rather than retrieving content

from remote data centers, clients request file objects from nodes (computers running the client software) that choose to host them, giving preference to those with the lowest latency. Popular files may be locally cached by multiple nodes in different regions, resulting in a scalable, resilient data storage platform capable of hosting photo albums, large scientific datasets, and a copy of Wikipedia [59]. Furthermore, IPFS employs a *content-based addressing* scheme, where each new object is given a unique identifier that is related to the hash of the content it represents. This serves as a form of version control à la Git [60]. Every file that is added to IPFS through Additive OS undergoes symmetric, client-side encryption using keys managed by AWS [61]. While this is adequate for many situations (and takes full advantage of the scalability of the public IPFS network), it is also possible to create private IPFS networks for additional privacy. As discussed in the next subsection, using existing Additive OS functions, content can easily be replicated among a cluster of dedicated servers for improved availability.

Full Customization with Minimal Hardware Requirements

In contrast to many proprietary platforms, developers can fully customize the Additive OS ecosystem through the open-source API (Figure 5). While the browser-based application exposes the API's methods through a graphical user interface, the lightweight client can run on low-resource hardware such as ARM devices. For example, inexpensive Raspberry Pi computers can upload process monitoring data directly to the platform using a simple Python script. Data scientists can run sophisticated queries of the MongoDB database to produce curated training sets for machine learning models. Files can be replicated among multiple remote servers to ensure access when primary nodes are offline. We hope Additive OS becomes a collaborative project where contributors continue to improve the platform and add functionality for the benefit of the AM community at large.

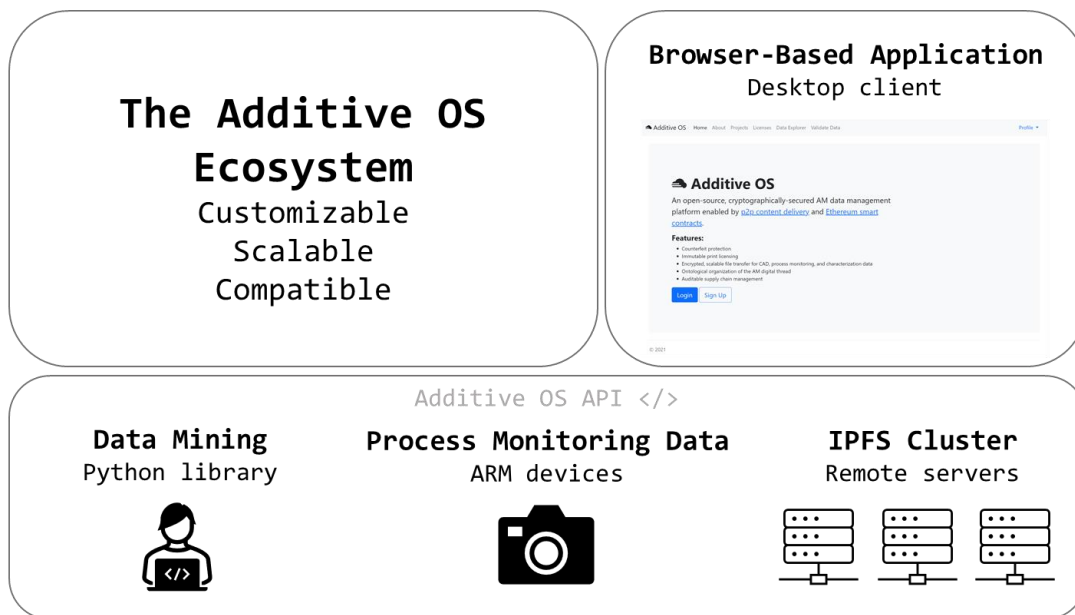


Figure 5. In addition to the browser-based application, users can use the Additive OS Python library to network their AM digital ecosystem, from data acquisition and organization to file storage and analytics.

Results: Example Use Cases

In this section, two example use cases are presented to show how smart contract functions are implemented in Additive OS.

Use Case #1: Sabotage

The first use case is inspired by the threat described in [1]. In this scenario, after a thorough design review, a rotor blade design is approved for manufacture. However, while in transit from the designer to the printing technician, the CAD file is intercepted, and a small defect is embedded in the part. Background studies have shown that even small voids, while difficult to detect upon manual inspection, can be detrimental to part properties [4], [5]. The filenames and sizes of the authentic and sabotaged parts are identical, as only a few bytes were changed.

In Additive OS, creating a new project executes the `addProject` function in the `AMProject` smart contract (Figure 6), which initiates a project object and assigns a unique project ID. The designer can share the content with the technician by adding their email address to the access list on the project page. When the designer browses for and submits the approved CAD file, the application computes the MD5 checksum, encrypts, and pins the file to the local IPFS node⁶. It calls the smart contract's `addHash` function, which associates the checksum with this project. Relevant metadata is inserted in the MongoDB database, including the IPFS CID pointing to the file and a URL to view the smart contract transaction on Etherscan [62], which is an independent block explorer for the Ethereum network⁷. The technician can then download the CAD file from IPFS, decrypt it locally, and quickly compute the checksum using the `Validate Data` tab on Additive OS or using their own MD5 solver. Even if the MongoDB cluster is compromised or accidentally deleted, the hash of the authentic file will always persist in the smart contract, which has no methods for removing or updating project or hash data. In fact, all transactions⁸ will be recorded with the smart contract, ensuring fault-tolerant data provenance.

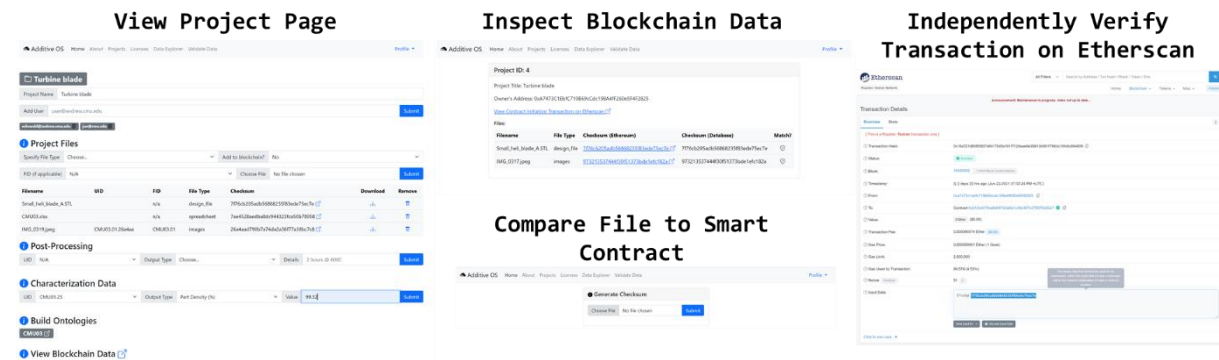


Figure 6. Screenshots of the Additive OS browser interface showing how the application can be used for project organization and tamper detection (use case #1).

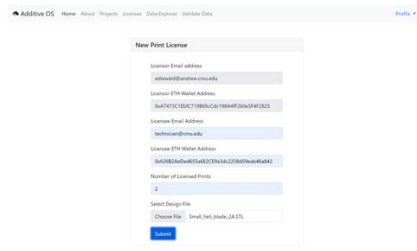
⁶ Additive OS does have support for remote pinning through an external service called Pinata [63], but this negates many of the main benefits of operating a node independently.

⁷ Of course, anyone also has the option to run a their own Ethereum node and view the blockchain data directly for themselves.

⁸ Read-only functions like `getHash` are not considered transactions, as they do not alter the Ethereum world state.

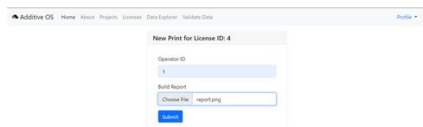
Use Case #2: Licensing and IP Theft

The second use case addresses licensing and counterfeiting issues that are commonly discussed by AM practitioners as outlined in [7]. As opposed to the first scenario, this case assumes an adversarial role between a part's designer and the service bureau that was subcontracted to fabricate it. The designer would like to ensure that the printing shop can only create 5 copies of an unpatented widget, which will be sent to an end customer. In Additive OS, the designer uses a form to create a license object by calling the `addLicense` function in the `AMLICENSE` smart contract (Figure 7). The object stores the ETH wallet addresses of the licensee and licensor, the maximum number of licensed prints, and the checksum of the CAD file. Each time the service bureau prints a part, they use another form to log the operator ID and a hash of the build report. This `addPrint` transaction creates a tamperproof digital certificate of authenticity, which Additive OS generates a printable QR code for. The self-governing smart contract cannot create more licenses than the designer originally specified, so unauthorized copies can be easily detected. Moreover, if the service bureau stole the design for themselves or a competitor, the contract would provide attestation of IP ownership, as the designer could produce the file associated with the checksum supplied in the `addLicense` transaction.



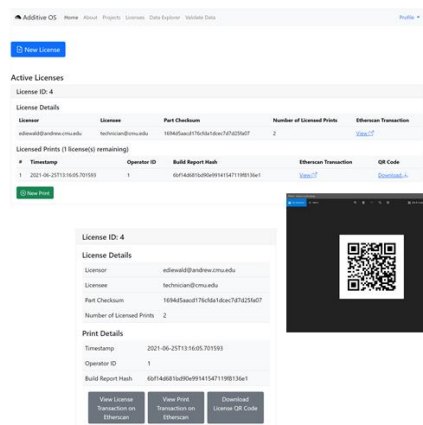
Create a New License

Specify number of prints, design file



Log a Licensed Print

Include operator ID, build report



Inspect License Details

View transactions on Etherscan, print QR code



View Digital Certificate of Authenticity

QR code remains with physical twin

Figure 7. An example workflow for secure print licensing. Each stage invokes a transaction in the `AMLICENSE` smart contract to read or write data.

Conclusions

Additive OS fills a need for an intuitive, open-source AM data and file management platform. Through Ethereum smart contracts built into the backend of the software, our solution integrates sabotage detection and IP theft prevention directly into the user interface. For most users, the browser-based application is sufficient for uploading, downloading, sharing, and organizing project files. However, developers and data scientists can enjoy open access to the underlying Python-based API. Data ontology is preserved through a unique ID-based naming convention, which allows users to import build parameters automatically from spreadsheet templates. A flexible MongoDB instance captures the hierarchy of complex DAG structures and provides support for queries in dozens of programming languages. Given the minimal dependencies, the Additive OS client can be run on low-resource hardware, allowing laboratories to connect in situ monitoring equipment or server clusters directly to the network with a few lines of code. Encrypted peer-to-peer file storage through IPFS enables low-latency, highly available content delivery without compromising privacy.

Like many software projects, Additive OS is an ongoing effort that will continuously improve through new features, security updates, and efficiency enhancements. The author is committed to maintaining the repository, updating documentation, and handling issues, but other developers are wholeheartedly encouraged to contribute to this open-source initiative. To try Additive OS in your own facility, you will need a MongoDB cluster (available with a free trial), an AWS account for encryption key management, an Ethereum wallet to cover transaction fees⁹, and some technical expertise. Documentation is provided on the project's Github page.

References

- [1] S. Belikovetsky, M. Yampolskiy, J. Toh, J. Gatlin, and Y. Elovici, "Dr0wned – Cyber-physical attack with additive manufacturing," in *11th USENIX Workshop on Offensive Technologies, WOOT 2017, co-located with USENIX Security 2017*, 2017.
- [2] L. D. Sturm, C. B. Williams, J. A. Camelio, J. White, and R. Parker, "CYBER-PHYSICAL VULNERABILITIES IN ADDITIVE MANUFACTURING SYSTEMS."
- [3] S. E. Zeltmann, N. Gupta, N. G. Tsoutsos, M. Maniatakos, J. Rajendran, and R. Karri, "Manufacturing and Security Challenges in 3D Printing," *Springer*, vol. 68, no. 7, pp. 1872–1881, Jul. 2016.
- [4] B. Ranabhat, J. Clements, J. Gatlin, K.-T. Hsiao, and M. Yampolskiy, "Optimal Sabotage Attack on Composite Material Parts."
- [5] S. B. Moore, W. B. Glisson, and M. Yampolskiy, *Implications of Malicious 3D Printer Firmware*. 2017.
- [6] A. Slaughter, M. Yampolskiy, M. Maahews, W. E. King, G. Guss, and Y. Elovici, "How to Ensure Bad dality in Metal Additive Manufacturing: In-Situ Infrared Thermography from the Security Perspective," in *Proceedings of the 12th International Conference on Availability, Reliability and Security*, vol. 11.

⁹ The smart contracts used in the examples were deployed to the Ropsten Testnet, which is free to use and suitable for development, but is significantly less stable than the Ethereum Mainnet.

- [7] S. Kurpjuweit, C. G. Schmidt, M. Klöckner, and S. M. Wagner, "Blockchain in Additive Manufacturing and its Impact on Supply Chains," in *Journal of Business Logistics*, 2021, vol. 42, no. 1, pp. 46–70.
- [8] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System."
- [9] "The Eth2 upgrades | ethereum.org." [Online]. Available: <https://ethereum.org/en/eth2/>. [Accessed: 25-Jun-2021].
- [10] C. Mandolla, A. M. Petruzzelli, G. Percoco, and A. Urbinati, "Building a digital twin for additive manufacturing through the exploitation of blockchain: A case analysis of the aircraft industry," *Comput. Ind.*, vol. 109, pp. 134–152, Aug. 2019.
- [11] N. Papakostas, A. Newell, and V. Hargaden, "A novel paradigm for managing the product development process utilising blockchain technology principles," *CIRP Ann.*, vol. 68, no. 1, pp. 137–140, Jan. 2019.
- [12] W. Alkhader, N. Alkaabi, K. Salah, R. Jayaraman, J. Arshad, and M. Omar, "Blockchain-based traceability and management for additive manufacturing," *IEEE Access*, vol. 8, pp. 188363–188377, 2020.
- [13] J. Stjepandic, "Copyright Protection in Additive Manufacturing with Blockchain Approach," 2017.
- [14] Z. C. Kennedy *et al.*, "Enhanced anti-counterfeiting measures for additive manufacturing: Coupling lanthanide nanomaterial chemical signatures with blockchain technology," *J. Mater. Chem. C*, vol. 5, no. 37, pp. 9570–9578, Sep. 2017.
- [15] "United States Patent Application: 0180173203." [Online]. Available: <http://appft.uspto.gov/netacgi/nph-Parser?Sect1=PTO2&Sect2=HITOFF&u=%2Fnetacgi/nph-adv.html&r=1&p=1&f=G&l=50&d=PG01&S1=20180173203.PGNR.&OS=dn/20180173203&RS=DN/20180173203>. [Accessed: 03-Jun-2020].
- [16] "Codename 'TRUEngine': GE Aviation, Microsoft Reveal Aircraft Parts Blockchain - CoinDesk." [Online]. Available: <https://www.coindesk.com/codename-truengine-ge-aviation-and-microsoft-reveal-aircraft-parts-certification-blockchain>. [Accessed: 03-Jun-2020].
- [17] "Cubichain Technologies." [Online]. Available: <http://www.cubichain.com/index.jsp>. [Accessed: 18-Jun-2021].
- [18] "MultiChain Pricing | MultiChain." [Online]. Available: <https://www.multichain.com/pricing/>. [Accessed: 18-Jun-2021].
- [19] Shomiz, "Strategic Guide: Additive Manufacturing Data Management and Schema Findings and Path Forward."
- [20] D. Eddy, S. Krishnamurty, I. Grosse, M. Perham, J. Wileden, and F. Ameri, "Knowledge management with an intelligent tool for additive manufacturing," in *Proceedings of the ASME Design Engineering Technical Conference*, 2015, vol. 1A-2015.
- [21] F. Belkadi, L. M. Vidal, A. Bernard, E. Pei, and E. M. Sanfilippo, "Towards an Unified Additive Manufacturing Product-Process Model for Digital Chain Management Purpose," in *Procedia CIRP*, 2018, vol. 70, pp. 428–433.
- [22] E. M. Sanfilippo, F. Belkadi, and A. Bernard, "Ontology-based knowledge representation for additive manufacturing," *Comput. Ind.*, vol. 109, pp. 182–194, Aug. 2019.
- [23] "Ansys GRANTA MI for Additive Manufacturing," 2020.
- [24] "Senvol | Data to help companies implement additive manufacturing." [Online]. Available: <http://senvol.com/>. [Accessed: 22-Nov-2020].

- [25] “Link3D Additive Manufacturing Workflow & MES.” [Online]. Available: <https://am.link3d.co/>. [Accessed: 23-Jun-2021].
- [26] “aMES OEM | Authentise.” [Online]. Available: <https://www.authentise.com/mes-oem>. [Accessed: 23-Jun-2021].
- [27] B. Roh, ... S. K.-I., and undefined 2016, “Ontology-based laser and thermal metamodels for metal-based additive manufacturing,” *asmedigitalcollection.asme.org*.
- [28] Y. Lu, P. Witherell, and A. Donmez, “A collaborative data management system for additive manufacturing,” in *Proceedings of the ASME Design Engineering Technical Conference*, 2017, vol. 1.
- [29] M. Mohd Ali, R. Rai, J. N. Otte, and B. Smith, “A product life cycle ontology for additive manufacturing,” *Comput. Ind.*, vol. 105, pp. 191–203, Feb. 2019.
- [30] J. S. Liang, “An ontology-oriented knowledge methodology for process planning in additive layer manufacturing,” *Robot. Comput. Integr. Manuf.*, vol. 53, pp. 28–44, Oct. 2018.
- [31] M. Dinar and D. W. Rosen, “A design for additive manufacturing ontology,” *J. Comput. Inf. Sci. Eng.*, vol. 17, no. 2, Jun. 2017.
- [32] N. Guarino, D. Oberle, and S. Staab, “What Is an Ontology?,” in *Handbook on Ontologies*, Springer Berlin Heidelberg, 2009, pp. 1–17.
- [33] B. Calabrese, “Standards and models for biological data: Common formats,” in *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics*, vol. 1–3, Elsevier, 2018, pp. 130–136.
- [34] J. Zhou *et al.*, “Graph neural networks: A review of methods and applications,” 2021.
- [35] “The Role of Ontology and Information Architecture in AI.” [Online]. Available: <https://www.earley.com/insights/role-ontology-and-information-architecture-ai>. [Accessed: 23-Jun-2021].
- [36] “GraphDB™ - Ontotext.” [Online]. Available: <https://www.ontotext.com/products/graphdb/>. [Accessed: 23-Jun-2021].
- [37] “SPARQL Query Language for RDF.” [Online]. Available: <https://www.w3.org/TR/rdf-sparql-query/>. [Accessed: 23-Jun-2021].
- [38] “The most popular database for modern apps | MongoDB.” [Online]. Available: <https://www.mongodb.com/>. [Accessed: 20-Nov-2020].
- [39] H. Abbes and F. Gargouri, “ScienceDirect Big Data Integration: a MongoDB Database and Modular Ontologies based Approach,” *Procedia Comput. Sci.*, vol. 96, pp. 446–455, 2016.
- [40] “PancakeSwap Intro - PancakeSwap.” [Online]. Available: <https://docs.pancakeswap.finance/>. [Accessed: 23-Jun-2021].
- [41] “Uniswap | Smart contracts.” [Online]. Available: <https://uniswap.org/docs/v2/protocol-overview/smart-contracts/>. [Accessed: 23-Jun-2021].
- [42] M. A. Law, A. Sanchez, and J. Rubin, “Smart Contracts and their Application in Supply Chain Management Signature redacted Signature of Author System Design and Management Program Signature redacted Certified by Signature redacted,” Massachusetts Institute of Technology, 2017.
- [43] “Smart Contracts - Overview, Uses, Benefits, Limitations.” [Online]. Available: <https://corporatefinanceinstitute.com/resources/knowledge/deals/smart-contracts/>. [Accessed: 23-Jun-2021].
- [44] “Ethereum Whitepaper | ethereum.org.” [Online]. Available: <https://ethereum.org/en/whitepaper/>. [Accessed: 23-Jun-2021].

- [45] “Solidity — Solidity 0.8.6 documentation.” [Online]. Available: <https://docs.soliditylang.org/en/v0.8.6/>. [Accessed: 23-Jun-2021].
- [46] “MetaMask.” [Online]. Available: <https://metamask.io/index.html>. [Accessed: 12-Jul-2021].
- [47] *Security for Microsoft Windows System Administrators*. Elsevier, 2011.
- [48] “Chapter 1306 - Ohio Revised Code | Ohio Laws.” [Online]. Available: <https://codes.ohio.gov/ohio-revised-code/chapter-1306>. [Accessed: 24-Jun-2021].
- [49] “Vermont Laws.” [Online]. Available: <https://legislature.vermont.gov/statutes/section/12/081/01913>. [Accessed: 24-Jun-2021].
- [50] “View Document.” [Online]. Available: <https://www.azleg.gov/viewdocument/?docName=https://www.azleg.gov/ars/44/07061.htm>. [Accessed: 24-Jun-2021].
- [51] “Directed Acyclic Graphs (DAGs).” [Online]. Available: https://ericsink.com/vcbe/html/directed_acyclic_graphs.html. [Accessed: 24-Jun-2021].
- [52] “Introducing Directed Acyclic Graphs and their use cases.” [Online]. Available: <https://www.capgemini.com/gb-en/2020/10/introducing-directed-acyclic-graphs-and-their-use-cases/>. [Accessed: 24-Jun-2021].
- [53] B. Sauer and T. J. VanderWeele, “Use of Directed Acyclic Graphs,” 2013.
- [54] “Interactive network visualizations — pyvis 0.1.3.1 documentation.” [Online]. Available: <https://pyvis.readthedocs.io/en/latest/>. [Accessed: 24-Jun-2021].
- [55] “Box — Secure Cloud Content Management, Workflow, and Collaboration.” [Online]. Available: <https://www.box.com/home>. [Accessed: 24-Jun-2021].
- [56] “Cloud Storage for Work and Home - Google Drive.” [Online]. Available: <https://www.google.com/drive/>. [Accessed: 24-Jun-2021].
- [57] “Cloud Object Storage | Store & Retrieve Data Anywhere | Amazon Simple Storage Service (S3).” [Online]. Available: <https://aws.amazon.com/s3/>. [Accessed: 20-Nov-2020].
- [58] “What is IPFS? | IPFS Docs.” [Online]. Available: <https://docs.ipfs.io/concepts/what-is-ipfs/#decentralization>. [Accessed: 24-Jun-2021].
- [59] “Swapping bits and distributing hashes on the decentralized web | by Carson Farmer | Textile | Medium.” [Online]. Available: <https://medium.com/textileio/swapping-bits-and-distributing-hashes-on-the-decentralized-web-5da98a3507>. [Accessed: 24-Jun-2021].
- [60] “Git.” [Online]. Available: <https://git-scm.com/>. [Accessed: 24-Jun-2021].
- [61] “Key Management Service - Amazon Web Services (AWS).” [Online]. Available: <https://aws.amazon.com/kms/>. [Accessed: 24-Jun-2021].
- [62] “Ethereum (ETH) Blockchain Explorer.” [Online]. Available: <https://etherscan.io/>. [Accessed: 24-Jun-2021].
- [63] “Pinata | Effortless IPFS File Management.” [Online]. Available: <https://pinata.cloud/>. [Accessed: 24-Jun-2021].