Distributed manufacturer services to provide product data on the web

André Hoffmann[1], Anna Wagner[1], Tim Huyeng[1], Meiling Shi[1], Julian Wengzinek[2], Wendelin Sprenger[2], Christoph Maurer[3], Uwe Rüppel[1]

1: Institut für Numerische Methoden und Informatik im Bauwesen, Technische Universität Darmstadt, Darmstadt, Germany
2: Zentrale Technik, Direktion Digitalisierung und Software-Engineering (DS)-BIM.5D, Ed. Züblin AG, Stuttgart, Germany
3: Department Energy Efficient Buildings, Fraunhofer Institute for Solar Energy Systems ISE, Freiburg, Germany
hoffmann@iib.tu-darmstadt.de

Abstract. Digital planning methods are changing the needs of construction stakeholders. It should be as easy as possible to compare product data from different manufacturers and integrate them into the planning model without detours. Web technologies not only offer great potential to enhance existing ways of working, but also open up new business models for product manufacturers in the service and IOT domains. However, manufacturers have a legitimate interest to retain control over the disclosure of their product data. Therefore, it is often only available offline and on demand. Of technical interest are therefore solutions, in which a large part of the data autonomy and access control remains on the side of the manufacturer, but the project planner is allowed to search the product data of all manufacturers like an interconnected system. Frequently, such a searchability is ensured by a uniform data schema and storage of the data in a central storage location, a so-called data warehouse. Nonetheless, this stands in contrast to the natural interest of the manufacturer for data sovereignty. This paper focuses on systems that can replace centralized data storage with distributed data management on manufacturer-owned servers, without losing accessibility for planners. Manufacturers should have as much freedom of implementation as possible when establishing the security concept, modeling their data and integrating the proposed approach into their operating procedure.

1. Introduction

Information silos are a source of inefficiencies (Pala et al., 2016: 897); they result in existing solutions to problems not being found and used. Particularly in the relationship between manufacturer and customer, both sides are affected. On the one hand, the customer cannot identify whether an advertised product is better suited for a purchase than a comparable product based on the provided information. Under certain circumstances, this may lead to no or an insufficient solution being found. The manufacturer, on the other hand, loses a potential buyer.

In the construction industry, dismantling data silos of stakeholders is an important task. Currently, suppliers and specialist engineers from various disciplines exchange geometries in the form of 3D models enriched with semantic information (Pala et al., 2016: 900).

The quality of the data transfer although depends on the import module of the specialist engineer's proprietary software. The software interoperability often stands in contrast to the interest of the software developer, who supports proprietary formats that only allow further processing of the data with software applications from his own product catalog, if at all. One solution to exchange building data in an open and efficient way is the Industry Foundation Classes Standard (IFC). The IFC is supported by about 150 software applications worldwide to enable better work flows for the AEC industry[1]. Additionally, ontologies are examined in

[1] http://www.buildingsmart-tech.org/ (accessed: 01.04.2019)

numerous publications to integrate data models from various areas of engineering (Pauwels *et al.* 2017).

The flexibility and simplicity of the representation in the form of triples allow to model even complex interrelationships in a human- and machine-understandable way. Therefore, this technology presents at least a partial solution to the interoperability problem.

Another problem is the lack of a standardized dissemination process of the BIM objects. Manufacturers often rely on isolated solutions on their own websites or prefer the path of direct distribution. Centralized websites apparently lack manufacturer acceptance, which limits the number of products made available. However on bimobject.com, according to its web presence the "Worlds Leading BIM Content Platform", product families of about 1352 brands can be accessed[2].

In this paper, an approach to distribute product data on the Web without using a central platform and to leave the persistent product data on the side of the manufacturer is discussed. Each manufacturer hosts their own product data service, which is linked to other product data services by message exchange and passes on queries to all other service instances. Queries to the entire system can be directed to any of the manufacturer services, to guarantee independence from a central instance. The global data model is based on a product ontology and serves to compare and link product data.

Decentralization and the freedom to design the implementation of manufacturer services should increase the acceptance of manufacturers. A product ontology reduced to the essentials, which can be linked with domain-specific ontologies if required, should guarantee searchability and comparability.

2. Related Work

Architects and engineers have various central platforms at their disposal for the acquisition of BIM data. When researching the availability of BIM / CAD objects via open-access portals, the highly decentralized distribution of these objects was particularly noticeable. The number of large portals alone, in which objects of numerous manufacturers can be found, amounts to almost 20. If websites with manufacturer-specific content that are linked in the Autodesk BIM blog[3] are included in this estimate, more than 50 portals were identified. However, usually no information about the topicality and completeness of the content on the portals is available.

In addition, platforms often only offer strongly limited possibilities to filter the complete data base in search for specific objects. In most cases, such filtering is restricted to manufacturer of the object, some BIM object types and file formats. Manufacturer-specific pages usually offer a download of their entire library without allowing the user to select singular objects beforehand. Apart of object platforms, platforms to facilitate the exchange of product and other project data between suppliers and other parties involved in the construction process have already been proposed. Vortalway is conceived to be a central cloud solution into which other services can be integrated if required (Grilo and Jardim-Goncalves 2013). Their focus was primarily on mapping the bidding process in order to offer or request a service or a product. These also include ontologies at the conceptual level to solve the interoperability problem (He et al., 2018: 18).

[2] https://www.bimobject.com/en (accessed: 01.04.2019)

[3] https://blogs.autodesk.com/bimblog/revit-content-online-bibliotheken-mit-10-000en-familien-als-download-beitrag-wird-regelmasig-aktualisiert/ (accessed: 01.04.2019)

The networking of manufacturers to offer a service of common benefit is a research idea that already has tradition. A so-called virtual enterprise is “a temporary alliance of enterprises that come together to share skills or core competencies and resources in order to better respond to business opportunities, and whose cooperation is supported by computer networks" (Camarinha-Matos and Afsarmanesh 1999). Under this topic, the OSMOS API was proposed in the construction industry. The OSMOS API pursued a project-related approach of cooperation, but also relied on a centralized platform (Wilson *et al.* 2001).

This paper’s content overlaps also with the field of Enterprise Application Integration, which aims to link applications on an internal or inter-company level in order to preserve resources and knowledge that were invested in existing applications for future purposes. A number of methods, architectural concepts and standards for fulfilling this goal are classified under as such Enterprise Application Integration (Arndt *et al.* 2009).

Furthermore, concepts for describing distributed semantic web architectures have been analyzed in the literature (Vdovjak *et al.* 2006). The hierarchical mediator architecture (see Figure 1) comes closest to the approach underlying this paper (Vdovjak et al., 2006: 44-51). The features provided in the theoretical concept could be simplified with knowledge of the existing data model.

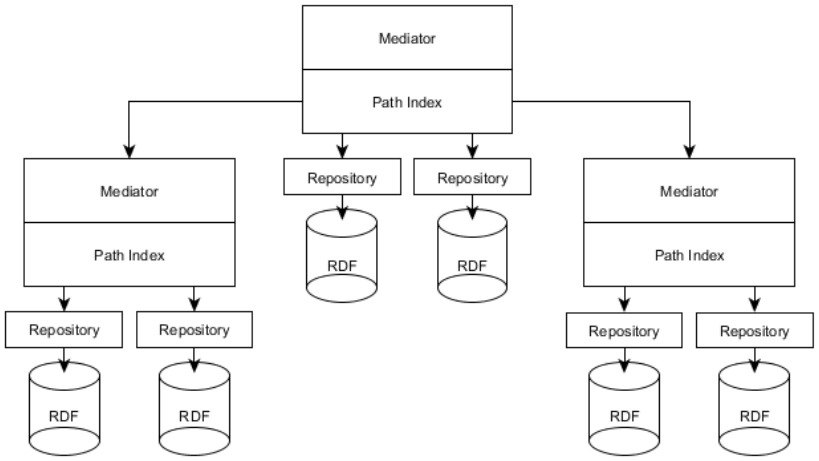

Figure 1: Hierarchical mediator architecture (adapted from Vdovjak *et al.* 2006: 45)

3. Concept

The basic idea of the approach presented in this paper is a network of manufacturer data services. Each of the data services provides product descriptions that contain information, which originated in the design and production phase and are therefore in the manufacturers’ responsibility. Each data service can be integrated into a manufacturer's service infrastructure, for example via an API gateway that orchestrates requests and forwards them to the data service. The data service has two interfaces: a network and a query interface. The network API is used for communication with other manufacturer data services while the query API enables users and third-party systems to send queries to the entire network of manufacturer data services. The manufacturer data services exchange messages to answer queries like a related system. Thus, the same comparison and search possibilities of centralized platforms should be provided. In the context of distributed database systems, this property is referred to as network transparency (Özsu and Valduriez, 2011: 9).

The users and applicatons, that query the network, should not have to deal with the internal architecture and the exact location of the product data. To ensure this transparency, queries to one data service must be forwarded to all other data services relevant to the queries. This must be realized on horizontal and vertical level (see Figure 2). For query distribution on horizontal level, all data services that contain the desired product type must be queried, and the results of the query passed on to the user. On the other hand, parts of the affected queries must also be passed on vertically: Construction products consist of materials and parts from suppliers and should therefore be regarded as assemblies. Suppliers can themselves operate data services in which they provide information on their products. Ergo, queries about specific products can also refer to data stored on the suppliers' data services. In order for the data service to know which data services can be found at which addresses and what type of information they contain, a data catalog must exist that stores such information on each data service and must be synchronized if changes are made. The data service can be integrated into the manufacturer's own infrastructure to offer additional services. For example, the data can be preprocessed by a company's own web services depending on the use case. If, for example, a certain format is required, the web service responsible for processing sends the request to an interface to the overall system.

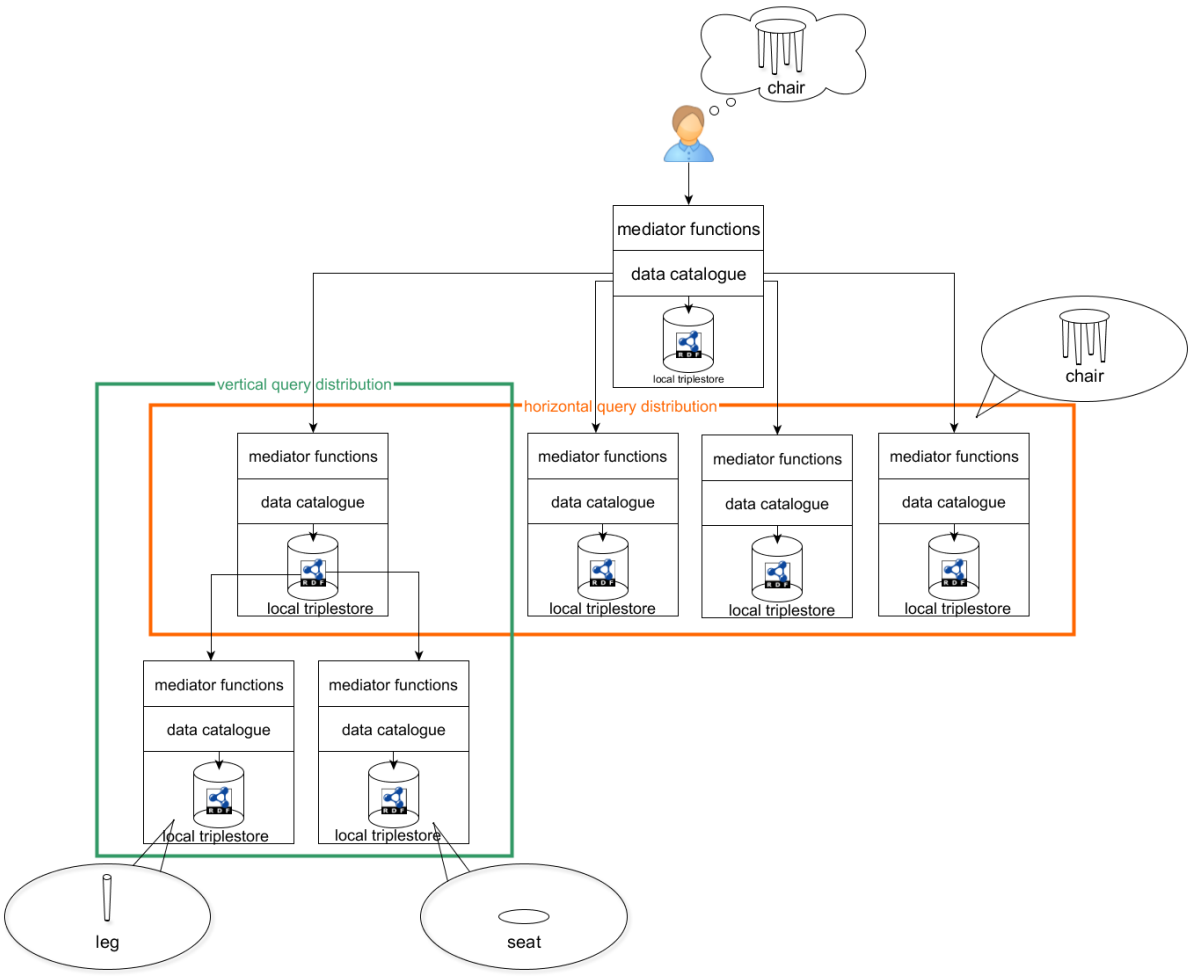

Figure 2: Vertical and horizontal query distribution

4. Data model

To enable communication of distributed and possibly not uniformly designed systems, machine-understandable data schemes are required. This also applies to the cross-platform product searches presented in this paper. It is therefore required for product data to be described using Building Product Ontology[4] (Wagner, A. 2019).

The BPO provides concepts for describing the basic structure of products, while additional information – such as the geometric description or classification of components, products and properties – is connected by other ontologies. For manufacturers, the property of BPO that products can be described in their composition in the form of components and assembly compounds is particularly interesting. Especially the latter enables manufacturers to reuse already existing descriptions of components that are parts of their products. For example, a manufacturer can link directly to the supplier's description of a screw or even complex components such as a door handle or window frame instead of remodeling it based on information received from suppliers in their own data environment.

Such linking functions by the property of the Semantic Web that all objects are occupied by a Unified Resource Identifier (URI) and can be accessed over this also from outside of the original graph to the objects either directly or by performing queries on the graph's database. In the BPO, the assembly connection relevant for this is implemented with the predicate bpo:consistsOf, which connects an assembly with any (also external) component and suggests that the component is installed as part of the product. An exemplary graph for the visualization of such a connection is shown in Figure 3.

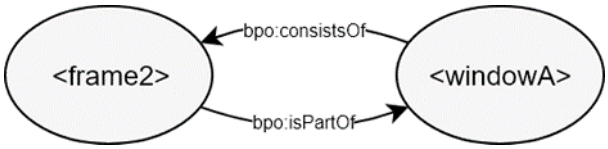

Figure 3: Example Relation between Products and Components

5. Dataservice: Components and processes

An data service consists of two functional components. One component is the so-called *query federator*, that applies the requested product data types to the data catalog to check which data service should be considered for querying. The *query federator* also forwards user queries to the corresponding data services identified in the previous step, and returns the summarized result to the user.

The second component is the *query responder*, which checks requests from other instances to identify whether data from suppliers is required to answer the requests and distributed the part of the query to all instances needed to fully describe the relevant product. The queries addressed to the APIs are formulated in the query language SPARQL. Alternatively, a format can be used that is transformed to a SPARQL query.

[4] https://w3id.org/bpo, accessed 31.05.2019

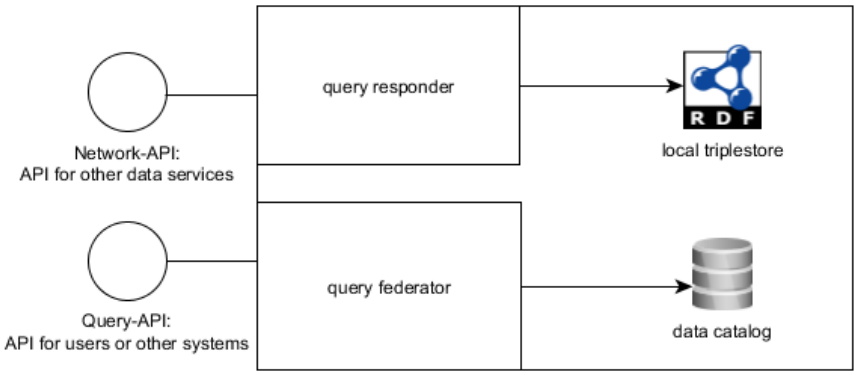

Figure 4: Architecture of a data service

5.1 Data catalog and synchronising

The data catalog is stored locally in each data service. It contains addresses of all data services and the product data types that are stored on them according to the buildingSMART Data Dictionary (bSDD). This metadata can be used to identify and request data services that are relevant for a query. If a new data service is added to the overall system or if a change is made to the product data types provided at a data service, a synchronization process must start. Such a process could be executed either traditionally by sending corresponding messages or - in case additional security is desired, that the data catalog is not manipulated and at every manufacturer the same – by implementing the data catalog synchronization with a blockchain. In essence, a blockchain is a chain of chronological blocks which is maintained by the nodes of a public or permissioned distributed network. By design, it is well suited for systems that require integrity, transparency and immutability (Wüst and Gervais 2018).

In this section the suitability of applying blockchain technology to data catalog synchronization will be analyzed. On implementing the data catalog which exists locally in distributed data services, following requirements must be fulfilled: First, data catalogs that are stored in distinct sites should be in a consistent state. That means changes in data catalog that are made locally must be synchronized in all other data services through the internet. In database management system these changes are documented as transaction logs. The synchronization process through multiple data services can be performed by the broadcast function of blockchain. Transaction logs will be wrapped up in a block and sent to all data services to synchronize; Second, unintended changes, including data altering with malicious intent and hardware breakdown, must be detected and avoided to guarantee data integrity. With suitable consensus mechanism manufacturer can give their approvals or rejections to a transaction log in data catalog so that unintended changes can be obviated; Third, information needs to be available when it is queried from outside of a data service. Blockchain is an art of distributed ledger that is maintained in every data service. If one data service falls down, queries can be sent to other data services. For the above mentioned reasons, blockchain is suitable to store an immutable record of transaction logs for distributed database (Sutton and Samavi 2017) (Aniello *et al.* 2017) that keeps the product data catalogs integral and available.

Apart from the requirements which are mentioned above, following aspects should also be taken into account. If multiple manufacturers update their catalogs simultaneously, all changes must be synchronized without high latency. Besides, executions – including meta data as author and timestamp – should be documented and traceable to avoid allegations in case of hostile

alterations in the database. Furthermore, manufactures should only have writing access to their own product data catalog entries. To fulfill those requirements, a concept for deploying product data catalog synchronization with blockchain is presented (see Figure 5).

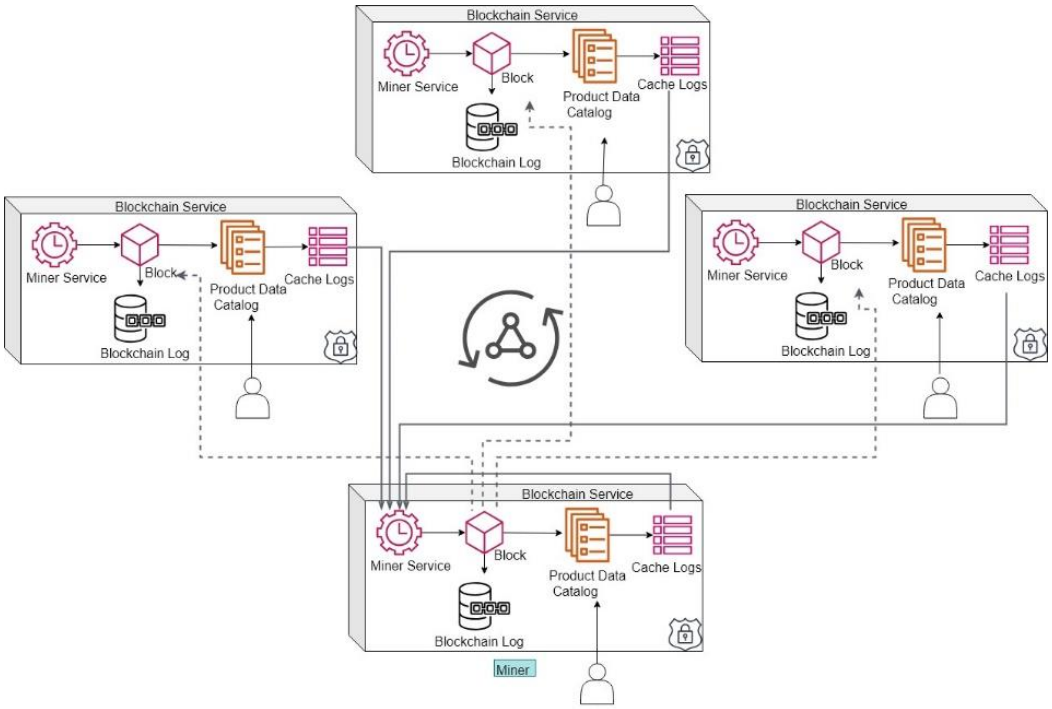

Figure 5: Concept for synchronization with blockchain

Similar to (Gaetani *et al.* 2017), we propose to use the mining rotation consensus to "mine" (generate) a block: First, a timespan will be defined to be round for generating one block. Each round, a miner is chosen by a pseudorandom generator. During the round, all executed transactions that are stored locally in "Cache Logs", including their timestamps, are sent to the "Miner Service" of miner this round. At the end of each round, the miner confirms all received transactions by signing with his digital private key and wrapping them with the cryptographic hash of the previous block as well as the timestamp of the round in a new block. Afterwards, a new hash is generated from this new block and is added to itself. After this mining process, the miner broadcasts the newly generated block to all nodes to verify its correctness. As soon as all nodes have confirmed the block with their individual digital cryptographic signature, the block can be appended to the original local blockchain "Blockchain Log" and execute the transaction logs in local "Product Data Catalog". The blockchain being maintained by a distributed instead of a centralized network also improves database security; if one node crashes or is attacked the transactions will not get lost. To ensure manufactures being restricted to edit their own product data catalog entries only, a smart contract can be applied. Smart contracts are part of the blockchain protocol and define who can update which entries in the product catalog.

5.2 Distributing queries vertically

The following section describes the algorithm for processing requests that require information about components originating from the supplier's data service. For example, queries of products may refer to specific properties that might be part of a subcomponent's description which is provided by the supplier's data service . To address the complete query, the part of the query

that refers to the subproduct must be identified and processed accordingly. The results of both queries can be temporarily merged to answer the original query.

The separation can be realized by using the analogy of data model and data service structure. Products are linked to their subcomponents with the predicate bpo:consistsOf. In order to find queries that relate to several data service, a first step is to check whether the query can be answered completely by the data available on the current data service. If this is not the case, the query is shortened by the triples that use the object of the last bpo:consistsOf relationship as subject until the query can be resolved. The response of the shortened query is checked for references to other data services. Since Linked Data uses URIs as identifier for individual objects, the location of an object's description is stored in the data model. This allows the data service of a product to be identified directly from the result of a query.

The triples that could not be resolved in the original query can now be passed on to the referenced data services as a construct query. SPARQL construct queries allow the extraction of a subgraph that corresponds to the template specified within the query in the form of triples. The resulting subgraph of this construct query can be merged with the triples that match the solvable triples of the local triple store, forming a graph which has all information to answer the original incoming query. The complete process can be seen in Figure 6. The construct query itself is treated as an incoming query by the supplier data service, in which the same process is repeated. In this way, requests of this type are answered recursively by all concerned data services. The process only has to take place if the query is fed by the *query federator* of another data service. The module responsible for this is called *query responder*.

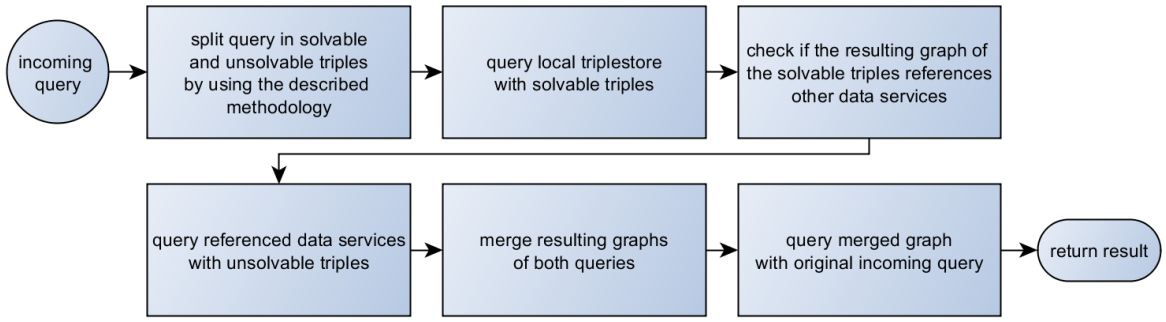

Figure 6: Algorithm to dissolve queries to subcontractors

5.3 Integration into the manufacturer's infrastructure

The presented service can be implemented with the help of a microservice architecture and can also be integrated into existing systems. Microservices are small processes that communicate with other processes via HTTP or a REST API (Dragoni *et al.* 2017). With microservices, a large application can be broken down into small applications that can be combined independently. These smaller applications can be used or replaced flexibly and organized as containers. Overall, the architecture of microservices is more complex than one of a monolithic system, but the scalability and flexibility of this approach outweigh the disadvantage. In the context of the presented work, flexibility is an advantage, since the presented decentralized data storage should be gradually integrated into existing manufacturer systems.

Of importance for the decentralized structure is a clear definition of interfaces between the individual services. Communication could be implemented via HTTP requests or other protocols and must be protected from external attacks.

One way of securing the individual services is an API gateway. Such gateways serve as entry points for a number of microservices, e.g., from manufacturers (see Figure 7). The gateway can authenticate a request with the help of an authentication service and distribute it to the corresponding services. As seen in Figure 7 microservices can also communicate with other services outside of their system. Therefore, it is recommended to use the API gateway from the other manufacture. If requests within one system occur, they can be processed either via the gateway or directly from service to service.

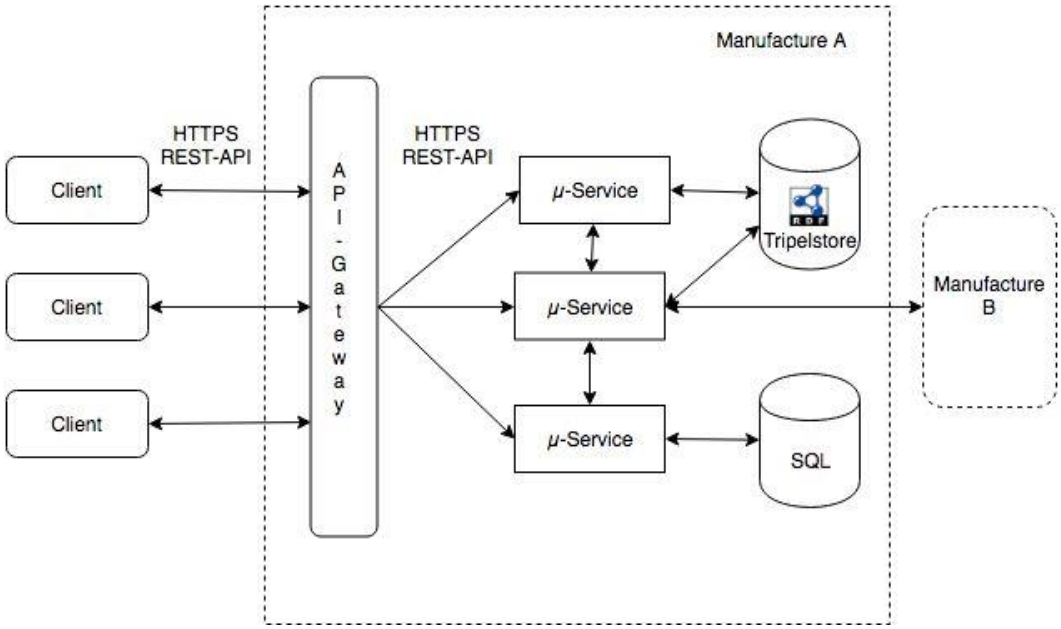

Figure 7: Micro-Service architecture, distribution via API gateways

Thanks to the microservice architecture, the integration of the dataservice into existing systems as well as maintenance and utilization for companies are simplified. Smaller companies, that cannot – or do not want to – provide all necessary services because of their internal structure, can also use and integrate other services. For example, computational services could be outsourced to a specialist who has optimized the specific service. Based on outsourcing processes, a new line of business can emerge, since high-performance services from other manufacturers could be used and the use of the service can be sold. However, data encryption must be ensured for enhancing acceptance and trust between individual stakeholders.

5.4 Authentification

An important component for increasing the confidence of manufacturers and users in micro-services is the authentication of partners. Since the concept proposed in this paper stores the data decentrally, it is essential to ensure unique identification. For this an authentication service is proposed, which can be outsourced to an own microservice if desired by the manufacturers.

This would enable central registration and prevent unwanted multiple logons to the various services and manufacturer instances. There are several possibilities for authentication: In addition to a single sign-on implementation, it is also possible to perform authentication with client certificates, API keys or a public and private keypair. Authorization can be connected to this service or can also be performed on any data service. Here, the individual manufacturers could assign rights to the respective users.

Another advantage of using microservices is the freedom of implementation. A free choice of programming languages as well as packages increases the potential to advance the manufacturers' own developments. In addition, the independent microservices can be deployed more easily and efficiently as containers. Current technologies, such as dockers, are available for this purpose (Stubbs *et al.* 2015).

6. Conclusion and Perspective

So far, parts of the functionalities have been implemented in an experimental manner in order to become familiar with the technologies and to gain an overview of the conceptual requirements for the architecture of the system. As a result of this effort, in the present paper, a concept for a distributed network of manufacturer services was presented, which enables stakeholders to search product data of all manufacturers as if it was a centralized system. One advantage of this decentralized data management is its reproduction of the natural distribution and thus guarantees manufacturers data autonomy. If interested, manufacturers can set up data services and integrate them into their own infrastructure. Thus, the overall system and the internal database of a manufacturer can grow modularly. Here, no single point of failure exists, since the entire network can be accessed via any manufacturer data service and all data services are to be regarded as equivalent. The hierarchical query processing results from the assembly structure of the product data model and, thereby, fits into the decentralized system architecture. Two sources of inefficiencies in the hierarchical mediator architecture should be mentioned (Vdovjak et al., 2006: 49-51), which also apply to the proposed approach: If, for example, a product is a subcomponent of more than one product, the information of the subcomponent is queried several times. Even if subcomponents refer to products that lie further up in the component hierarchy, inefficiencies can occur. The transition to a cooperative mediator architecture , which introduces additional communication channels between the data services, could create remedy here. The architecture of the communication channels could be justified by certain product data types on the data services and functional domains. In addition to the data services themselves, further microservices are to be created in the context of the research project SCOPE. Those services are to be understood as blueprints for various tasks in civil engineering and include, for example, processing routines for data from the BPO data model in planning software formats.

Acknowledgements

This work is part of the research project EnOB: SCOPE, founded by the German Federal Ministry for Economic Affairs and Energy (BMWi).

References

Aniello, L., Baldoni, R., Gaetani, E., Lombardi, F., Margheri, A. and Sassone, V. (2017). A Prototype Evaluation of a Tamper-Resistant High Performance Blockchain-Based Transaction Log for a Distributed Database. *2017 13th European Dependable Computing Conference (EDCC)*. pp. 151–154.

Arndt, C., Hermanns, C., Kuchen, H. and Poldner, M. (2009). *Best Practices in Der Softwareentwicklung*.

Camarinha-Matos, L.M. and Afsarmanesh, H. (eds.) (1999). The Virtual Enterprise Concept. *Infrastructures for Virtual Enterprises*. Boston, MA: Springer US.

Coelho, N. (2018). Security in Microservices Architectures.

Dragoni, N., Giallorenzo, S., Lafuente, A.L., Mazzara, M., Montesi, F., Mustafin, R. and Safina, L. (2017). Microservices: Yesterday, Today, and Tomorrow. In: Mazzara, M. and Meyer, B. (eds.). *Present and Ulterior Software Engineering*. Cham: Springer International Publishing, pp. 195–216.

Gaetani, E., Aniello, L., Baldoni, R., Lombardi, F., Margheri, A. and Sassone, V. (2017). Blockchain-based database to ensure data integrity in cloud computing environments. *Italian Conference on Cybersecurity (20/01/17)*.

Grilo, A. and Jardim-Goncalves, R. (2013). Cloud-Marketplaces: Distributed e-procurement for the AEC sector. *Advanced Engineering Informatics* **27**(2):160–172. doi: https://doi.org/10.1016/j.aei.2012.10.004.

He, D., Li, Z., Wu, C. and Ning, X. (2018). An E-Commerce Platform for Industrialized Construction Procurement Based on BIM and Linked Data. *Sustainability* **10**(8):2613. doi: https://doi.org/10.3390/su10082613.

Özsu, M.T. and Valduriez, P. (2011). *Principles of Distributed Database Systems*. 3rd ed. New York: Springer Science+Business Media.

Pala, M., Edum-Fotwe, F., Ruikar, K., Peters, C. and Doughty, N. (2016). Implementing commercial information exchange: a construction supply chain case study. *Construction Management and Economics* **34**(12):898–918. doi: https://doi.org/10.1080/01446193.2016.1211718.

Pauwels, P., Zhang, S. and Lee, Y.-C. (2017). Semantic web technologies in AEC industry: A literature overview. *Automation in Construction* **73**:145–165. doi: https://doi.org/10.1016/j.autcon.2016.10.003.

Stubbs, J., Moreira, W. and Dooley, R. (2015). Distributed Systems of Microservices Using Docker and Serfnode. *2015 7th International Workshop on Science Gateways*. pp. 34–39.

Sutton, A. and Samavi, R. (2017). Blockchain Enabled Privacy Audit Logs. In: d'Amato, C., Fernandez, M., Tamma, V., Lecue, F., Cudré-Mauroux, P., Sequeda, J., Lange, C., et al. (eds.). *The Semantic Web – ISWC 2017*. Springer International Publishing, pp. 645–660.

Vdovjak, R., Houben, G.-J., Stuckenschmidt, H. and Aerts, A. (2006). RDF and Traditional Query Architectures. In: Staab, S. and Stuckenschmidt, H. (eds.). *Semantic Web and Peer-to-Peer: Decentralized Management and Exchange of Knowledge and Information*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 41–58.

Wagner, A., Rüppel., U. (2019). BPO: The Building Product Ontology for Assembled Products. *7th Linked Data in Architecture and Construction Workshop*. Lisbon: PT. in press.

Wilson, I., Harvey, S., Vankeisbelck, R. and Kazi, A.S. (2001). Enabling The Construction Virtual Enterprise: The Osmos Approach. *Journal of Information Technology in Construction*

6(Special issue Information and Communication Technology Advances in the European Construction Industry):83–110.

Wüst, K. and Gervais, A. (2018). Do you Need a Blockchain? *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*. pp. 45–54.