Design of a software architecture supporting business-to-government information sharing to improve public safety and security

van Engelenburg, Sélinde; Janssen, Marijn; Klievink, Bram

DOI
10.1007/s10844-017-0478-z

Publication date
2017

Document Version
Final published version

Published in
Journal of Intelligent Information Systems

J Intell Inf Syst
DOI 10.1007/s10844-017-0478-z

Design of a software architecture supporting business-to-government information sharing to improve public safety and security

Combining business rules, Events and blockchain technology

Sélinde van Engelenburg[1] · **Marijn Janssen**[1] ·
Bram Klievink[1]

Received: 20 December 2016 / Revised: 26 April 2017 / Accepted: 4 July 2017

Abstract To ensure public safety and security, it is vitally important for governments to collect information from businesses and analyse it. Such information can be used to determine whether transported goods might be suspicious and therefore require physical inspection. Although businesses are obliged to report some information, they are reluctant to share additional information for fear of sharing competitively sensitive information, becoming liable and not being compliant with the law. These reasons are often overlooked in the design of software architectures for information sharing. In the present research, we followed a design science approach to develop a software architecture for business-to-government information sharing. Based on literature and a case study, we elicited the requirements an architecture that provides for the sharing of information should meet to make it acceptable to businesses. We then developed the architecture and evaluated it against the requirements. The architecture consists of a blockchain that stores events and rules for information sharing that are controlled by businesses. For each event, two parties use their private keys to encrypt its Merkle root to confirm that they know the data are correct. This makes it easy to check whether information is reliable and whether an event should be accepted. Access control, metadata and context information enable the context-based sharing of information. This is combined with the encryption and decryption of data to provide access to certain data within an organisation.

✉ Sélinde van Engelenburg
S.H.vanEngelenburg@tudelft.nl

Marijn Janssen
M.F.W.H.A.Janssen@tudelft.nl

Bram Klievink
A.J.Klievink@tudelft.nl

[1] Faculty of Technology, Policy and Management, Delft University of Technology,
Delft, The Netherlands

Published online: 27 July 2017

Keywords Software architecture · ICT architecture · Information sharing · Safety · Security · Willingness · Business rules · Access control · Encryption · Supply chain · Customs · Blockchain

1 Introduction

Easy and seamless information sharing can benefit both businesses and government agencies (Bharosa et al. 2013; Fawcett et al. 2007; Klievink et al. 2012a). Better information may, for instance, help businesses to improve supply chains (Klievink et al. 2012b). Government organisations in particular need information to ensure safety and security. For example, based on such information, a customs organisation can decide to inspect goods in order to identify smuggling. Information can be used to judge whether there is something wrong and thus a need for physical inspection. For this, governments need additional high-quality information on which to base their risk assessments.

There are, however, some factors that might make businesses unwilling to share their information (Fawcett et al. 2007). A perceived increase in vulnerabilities due to revealing information, combined with a lack of incentives to share, seriously hampers new information sharing initiatives (Klievink and Lucassen 2013). Therefore, even in cases in which information sharing may improve public safety and security, businesses might be unwilling to share. An architecture should thus meet the goals of both companies and governments.

We found that businesses require the ability to keep information confidential and assurance that information sharing is lawful. Furthermore, they want to avoid liability. In addition to meeting the requirements of businesses, customs should be able to rely on the information in order to ensure safety and security. This means that the information that is shared should be reliable. It also means that the information sharing process must be secure and that it should not be possible to intercept or manipulate information.

We developed a software architecture of a business-to-government (B2G) information sharing architecture to provide for an information sharing process that is acceptable to businesses. The Software Engineering Standards Committee of IEEE Computer Society (2000, p. 3) defines an architecture as the *"fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution."* According to Bass et al. (2003, p. 21), a software architecture is the structure or structures of a system and it comprises *"software elements, the externally visible properties of those elements and the relationships among them."* By externally visible properties, they mean *"those assumptions other elements can make of an element."* However, there is a lack of knowledge about how an architecture that meets the requirements of both companies and governments should look like. The objective of this research is to design a B2G information sharing architecture that will protect safety and security and meets the requirements of businesses. We used a design science approach to achieve this objective (Peffers et al. 2007).

In the next section, we describe our research approach, which was based on the steps in a design cycle described by Peffers et al. (2007). The structure of this paper mirrors these design science steps. In Section 3, we discuss the requirements for the architecture and in Section 4 we present its design. The architecture is then demonstrated and evaluated in Section 5.

2 Research approach

We employed a design science approach (Peffers et al. 2007) to design a software architecture supporting B2G information sharing. Central to design science is the cyclical development and evaluation of IT artefacts with the intention to solve organisational problems (Hevner et al. 2004). We followed the six steps of design science research described in the work of Peffers et al. 2007, viz.: 1) problem identification and motivation, 2) define the objectives for a solution, 3) design and development, 4) demonstration, 5) evaluation and 6) communication.

Steps 1 and 2 were presented in the introduction of this paper. In addition, we studied the requirements for the architecture, whilst focusing on willingness. These requirements provided the basis for the design of the artefact. We derived the requirements based on primary and secondary data.

For the literature we focused on the reasons why businesses are reluctant to share their information. The case material we used were the minutes and a transcript of two workshops with several staff members who have expertise in the juridical domain and the domain of technical innovation at a carrier. Furthermore, we used a transcript of an interview with a researcher with experience in implementing new information sharing processes in real-life supply chains as part of her research. In addition, we used a transcript of an interview with a staff member with a technical background and a staff member with managerial background at a third-party logistics provider. In all cases, various scenarios for information sharing were presented, and concerns about sharing and willingness to share information in those scenarios were discussed in detail.

In this paper we first describe the case of information sharing between businesses in a supply chain and customs. It is important to emphasise, however, that the studying of the case study material was in parallel with the studying of literature; we present it in this order so that we can explain a clear application domain first to the benefit of making the problem more specific to the reader. The research activities were independent of each other, but both contributed to an understanding of the requirements that the developed artefact had to meet. In practice, we found that the information obtained from the case material confirmed the conclusions based on the literature, and vice versa.

We identified factors that influence the willingness of businesses to share information. Three important requirements were found: 1) in some cases businesses need to be able to keep information confidential to, for example, prevent misuse; 2) businesses want to avoid liability and therefore in some cases are not willing to have information and consequently cannot share it; and 3) information sharing should be lawful. The requirements are described in more detail in Section 3.

Step 3 is the design and development of the artefact itself (Peffers et al. 2007). As described by Gregor and Hevner (2013), in the first design cycle the design is usually based on inspired creativity and trial and error. The design is the result of various cycles in which the case and literature were used to develop and improve the architecture (rigor), and in which stakeholders were involved (relevance) through workshops. In this way we could verify and refine the fit of a constructed artefact in a specific case.

Our initial design (see van Engelenburg et al. 2015) shared an important principle with the blockchain, viz. a distributed ledger and regulating access using encryption and keys. Based on this finding, we further investigated the literature on blockchains. We describe the architecture we developed in Section 4.

Step 4 is the demonstration of the artefact (Peffers et al. 2007). We demonstrate the architecture for a typical user activity in Section 5.1. The subsequent step is the evaluation of the artefact (Peffers et al. 2007). We performed an evaluation per requirement, as described in the remainder of Section 5. The evaluation was based on the same workshops as the requirements. In these workshops several elements of the architecture were presented and discussed with participants. The last step in design science research is communication (Peffers et al. 2007), which we do in this paper.

3 Information sharing requirements

In this section, we provide an outline of the case study. We also provide arguments for the appropriateness of this specific case study for our research. Next, we determine and discuss the requirements for the architecture.

3.1 Businesses sharing information with customs

The case study that is the basis of our requirements is that of information sharing between businesses in a supply chain and customs. A supply chain can be described as a complex network that consists of many different stakeholders, including shippers, deep-sea carriers, port operators and customs organisations (Van Baalen et al. 2009). Mentzer et al. (2001) argue that supply chains can be defined in various ways. According to Tsay et al. (1999, p. 301), modern usage of the term supply chain is consistent with the following: *"a supply chain is two or more parties linked by a flow of goods, information and funds."*. In concurrence with this description, when we refer to 'businesses in a supply chain', we are referring to businesses that are linked by a flow of goods, information and funds. The role of customs is that of gate-keeper and collector of excise and other duties and taxes. Customs is responsible for monitoring the flow of goods and interfering with it if there are safety, security or other public policy reasons to do so (NWO 2013).

We selected this case because of its complexity, which is due to the inclusion of various types of actors, from both government and business. These parties have different roles and interests. Adding further to the complexity is the international character, due to which visibility across the whole chain is limited and various laws may play a role. Furthermore, there was enough material available on this case for use in this study.

In this research, we focused on businesses in a supply chain transporting goods in containers. One of the main problems in container transport is that it is impossible to view the contents of containers without opening them. The number of containers is so high, that it is not even possible to come close to inspecting each container (Levinson 2010). This poses a serious problem for customs, which needs to monitor the flow of goods (Hesketh 2010) and prevent the entry of illegal goods.

Customs cannot open all containers, but officials can perform a risk assessment and target high-risk containers for inspection (Tan et al. 2011). To perform risk assessment, customs needs to analyse information on the shipments it will receive before the goods arrive at the border (Tan et al. 2011). Businesses could provide customs with this information. Businesses are obliged to share information with customs in the form of documents. For instance, a carrier that transports containers over the sea is required by European law to provide customs with an Entry Summary Declaration (ENS) describing the goods it is carrying and to do so 24 hours before departure (The Commission Of The European Communities 2006). The ENS is the main source for the risk assessment done by customs

(Zomer 2011). However, the information in documents such as the ENS is often not timely, or has been altered, is inaccurate or is vague (Klievink and Lucassen 2013; Lee and Whang 2000; Hesketh 2010). This poses safety and security risks (Hesketh 2010).

The need for high-quality information is reflected in the possible risks. For instance, according to the (European Economic and Social Committee 2003), ships and maritime transport are vulnerable to terrorist risks. More specifically, terrorists or weapons of mass destruction could be smuggled in containers (European Economic and Social Committee 2003). Disasters such as the explosion on the ship MSC Flaminia show the possible consequences if safety is not sufficiently ensured when dangerous goods are involved (see e.g. http://www.bbc.com/news/uk-england-cornwall-19448577).

To reduce safety and security risks, customs needs additional high-quality information on which to base its risk assessments. Fortunately, businesses already gather such information for their own business processes and for safety and security checks. For instance, the manufacturer of goods that are transported has a lot of details about them, such as their weight (Hesketh 2010). If customs finds that the weight of the container is unexpected based on the weight of the goods, this might be a reason for physical inspection. Another example is that of a shipper that packed the container and thus has first-hand knowledge of its contents (Hesketh 2010).

The information that businesses gather is of high quality since their own commercial operations depend on it (Bharosa et al. 2013). This information could be reused for purposes other than the ones it was gathered for, according to the piggy-backing principle (Tan et al. 2011; Bharosa et al. 2013). In this case, customs could reuse it to monitor the flow of goods.

Customs organisations can be expected to contribute to the competitiveness of their countries (Customs Administration of the Netherlands 2014). To protect competitiveness they need to keep the administrative burden on businesses low. They will therefore not simply require businesses to share additional high-quality information with them. Whether businesses voluntarily share additional information depends on whether their requirements are met. Since willingness was important in the case study, it made sense to study literature on it to determine factors that influence it.

Business are obliged to share certain information with customs, as shown in Fig. 1. The goods flow in the supply chain is shown using thick arrows. All businesses involved might have some information that can help customs decide whether physical inspections, such as the opening of containers, are necessary. The arrows on the right show information sharing between businesses and customs. The solid arrows show the obligatory sharing of the documents by a few parties. The information that they contain is of low quality. The dotted arrows show the potential sharing of additional high-quality information by all parties in the supply chain. This information is shared only when businesses are willing to do so.

3.2 Information sharing requirements

We focused on improving the willingness of businesses to share information by identifying requirements that should be met for the architecture to be acceptable to businesses. We identified three key requirements, which are discussed below.

3.2.1 Requirement 1: Information should be kept confidential according to the needs of businesses

Urcioli et al. (2013) identified confidentiality as a barrier to using information sharing platforms in e-Customs. For competitive reasons (e.g. fear of being bypassed in the chain) or

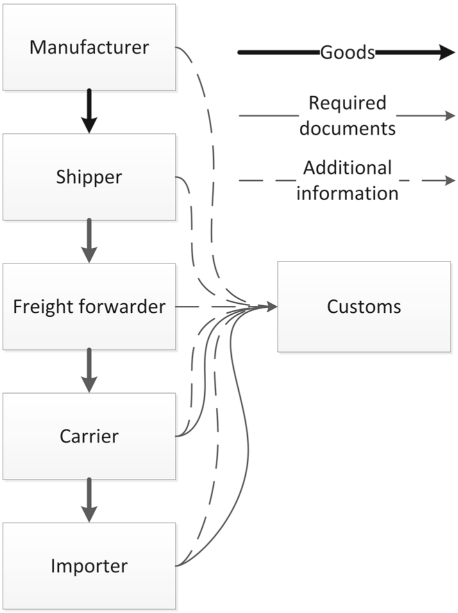

Fig. 1 Sharing of obligatory and additional information in supply chains

security reasons (e.g. information on high-value goods), businesses may be hesitant to share information with others (Klievink et al. 2012a; Fawcett et al. 2007). Furthermore, parties may perceive a higher vulnerability to misuse or opportunism by the partners they share data with Hart and Saunders (1997).

A factor that might improve the willingness to share information is trust. By trust we mean that parties need to be confident that the parties they share information with will not misuse the information they acquire or are otherwise victims of opportunistic behaviour by others (Hart and Saunders 1997; Klievink and Lucassen 2013). However, when sharing information, parties often have little information on the competence, care in use and reliability of others (Mishra 1996). This is especially the case in the competitive and global context of international trade, where information sharing is typically limited to the information needed or required for core processes and interactions with authorities. Therefore, trust alone is not sufficient, and parties want to be able to shield data from others. For the architecture to be acceptable to businesses, it should thus keep information confidential according to the needs of businesses.

3.2.2 Requirement 2: Information should be shared while allowing businesses to avoid liability

The sharing of more information can in some cases result in liability for businesses. For instance, the security analysis of goods that are shipped to the European Union (EU) from outside the EU is performed by the national customs authority of the first port of call in the EU (Zomer 2011). Customs organisations base this security analysis mainly on the Entry Summary Declaration (ENS). This declaration has to be filed by the carrier 24 hours before departure from the port of origin (Zomer 2011; Jensen et al. 2014; Jensen and Vatrapu 2015). According to EU Regulation 1875/2006 of the European Customs code (The Commission

Of The European Communities 2006, p. 114), the ENS should contain a description of the goods that *"is precise enough for Customs services to be able to identify the goods."*

The information in the ENS is based on ship manifests and data from the bill of lading (Zomer 2011; Klievink et al. 2014). The ship manifest is a list of all cargo on a ship and is based on the bills of lading associated with the cargo (Hesketh 2010; Veenstra et al. 2013; Klievink et al. 2014). A bill of lading is the receipt that the carrier gives to the shipper stating that it has received the goods and will transport them, and it shows the details of these goods (Hesketh 2010; Levi 2005).

According to the Hague-Visby Rules, the liability of the carrier is limited to a certain amount per package, unless the value and a full description of the goods are declared by the shipper and included in the bill of lading (Hesketh 2010). Were the carrier liable for the full costs, the shipping rates would increase (Hesketh 2010). To prevent this, the shipper omits the value of the goods and makes the goods description vague (Hesketh 2010). Since the information in the ENS is based on the information in the bill of lading, the goods description in the ENS is vague as well, even though it was originally intended to be precise enough to identify the goods.

In practice, the carrier thus avoids having the detailed goods description because it might increase its liability. Of course, it cannot share information it does not have. We thus identified a new requirement for the willingness to share information, namely businesses' preference to avoid increased liability. To increase the willingness to share information, the architecture should therefore make it possible for businesses to share information while allowing them to avoid liability.

3.2.3 Requirement 3: Information sharing using the architecture should be lawful

Legal considerations make information sharing between businesses in a supply chain and customs a highly complex process (Karampetsou 2016). Different legal frameworks are applicable to different categories of data, for example personal or impersonal, or confidential or public data (Karampetsou 2016). With whom data can be legally shared depends on the country in which goods are moving (Van Stijn et al. 2011), and different sources of law, such as national and European law, might be applicable at the same time. Moreover, legislation may change frequently (Gong and Janssen 2014).

Data protection law is one of the sources of law important for regulating information sharing between businesses in a supply chain and customs. According to article 8 of the European Convention on Human Rights, everyone has the right to respect for their private life (European Court of Human Rights 2010). According to jurisprudence, 'everyone', in this case, also includes legal entities such as businesses (Karampetsou 2016). The collecting, storing or using of personal data regarding a legal entity such as a business, especially by government organisations such as customs, is thus protected (Karampetsou 2016). Data concerning the professional reputation of a business are also protected (Karampetsou 2016).

It is not easy for businesses to determine whether information sharing is lawful according to data protection law. This might become clear only after looking at jurisprudence. Data protection law is applicable to the phases of gathering, storing and sharing personal data (Karampetsou 2016). The legal status of the sharing of data depends on the lawfulness of the gathering and storing of the data. Thus, to determine whether information sharing is lawful, the phases of gathering and storing the data also have to be considered. This makes matters even more complex.

Several restrictions are put on the information sharing process by data protection law. For instance, data cannot be stored for an unlimited length of time and cannot be shared for a purpose other than the one they were gathered and stored for (goal-binding principle) (European Union Agency for Fundamental Rights and The Council of Europe 2014). In certain circumstances, exceptions can be made when information sharing is vital to public safety and security (NWO 2013). Such exceptions thus also have to be taken into account, especially when one wants to support information sharing to protect safety and security.

All in all, laws and regulations make information sharing between businesses in a supply chain and customs quite complex. This discourages businesses from sharing information with customs. Uncertainty about the legal status of information, as well as the legality of the methods used to obtain it, may lead to, for instance, carriers shielding their data from other parties (NWO 2013). Another factor is thus that businesses need to know that information sharing is lawful. One way to do this is by designing the architecture in such a way that information sharing is lawful when the architecture is used.

4 Designing a software architecture

The architecture we developed consists of five main components; 1) blockchain for recording events, 2) business rules for setting the conditions to share information, 3) access control to ensure only authorised access, 4) metadata and context information to understand whether the context enables information to be shared, and 5) encryption and decryption.

Blockchain technology is used to create a general ledger of events that are accessible to customs and that enable the secure sharing of reliable information. Business rules are used to set parameters about under what conditions to share information. Metadata and context information about the requester of access to data, together with the business rules, are used as input for the decision component to reach a decision about whether to share data. In this way, data sharing is context-dependent and is controlled by businesses. Access to data is prevented or granted by respectively encrypting and decrypting data elements using a key.

In each of the following subsections, we discuss an element of the architecture and its relationship to other elements. Section 5 provides an illustration and evaluation of the architecture.

4.1 Information sharing using blockchain technology

According to Pilkington (2016, p. 11), a blockchain is a *"secure public ledger platform shared by all parties through the Internet or an alternative distributed network of computers"*. Although blockchain technologies are usually associated with applications in the domain of cryptocurrencies such as Bitcoin, they can also be used outside the monetary domain (Pilkington and Zhegu 2016; Walport 2015). For instance, ledgers can be used to track the origin and transformations of goods in a supply chain (Pilkington and Zhegu 2016).

The architecture we developed is based on a distributed ledger model in which events, such as the stuffing, departure or arrival of a container are recorded. The idea of using events in information sharing is not new; event-driven architectures for information sharing in the supply chain domain already exist (see e.g. Overbeek et al. 2012). The combination with a blockchain is new, however. We discuss the elements required to combine blockchain technology and events in the following subsections.

Blockchain is based on the six steps described by Nakamoto (2008) in the paper in which Bitcoin was introduced. Since cryptocurrency is not our domain, some changes needed to be made to these steps. Most apparent is the skipping of the first two steps described by

Nakamoto (2008, p. 3), viz. *“New transactions are broadcast to all nodes”* and *“Each node collects new transactions into a block”*.

In the case of Bitcoin, nodes can gain bitcoins by adding blocks to the chain (Nakamoto 2008). To make this difficult, they therefore have to provide a proof of work first (Nakamoto 2008). The proof of work involves finding a value that satisfies certain properties (Nakamoto 2008). The finding of this value requires a lot of work, but once found it is easy to check whether the required properties are satisfied (Nakamoto 2008). We do not need proof of work to add a new event. Instead of making it difficult to put a block on the chain because currency can be earned by it, it should be easy for parties to add a block to share information.

The remaining steps for the proposed architecture are the following:

Step 1. A node adds a new event to the ledger as a new block
Step 2. The new block is broadcast to all other nodes
Step 3. Nodes accept the block according to a consensus mechanism (see Section 4.1.5)
Step 4. If the block is accepted, a new block that is added contains a hash of its header.

The blockchain in our architecture consists of a network in which a ledger containing information is distributed. The information is captured in events that have a body and a header. The events are accepted by nodes in the network according to a consensus mechanism. In the following subsections, we discuss these elements based on the steps above. We also worked the steps out in detailed sub-steps to provide a procedural overview of the use of the blockchain technology in our architecture. These are presented in the last subsection.

4.1.1 The network in which the ledger is distributed

In our architecture, events are shared in a network of systems of businesses and government organisations. For distributed ledgers several variants are possible, namely permissioned or permissionless ledgers, public or private ledgers, and variants in between (Mainelli and Smith 2015; Pilkington and Zhegu 2016; Walport 2015). Whether a ledger is public or private determines who can use copies of the ledger (Walport 2015; Buterin 2015). Whether a ledger is permissioned or permissionless determines who maintains the ledger (Walport 2015; Buterin 2015).

All parties in the supply chain should be in the network, as they all might have information to share. However, there is no clear reason for letting parties other than those businesses and government organisations be part of the network. In addition, there is no reason to restrict the right to determine consensus to a single party, such as in a fully private and permissioned ledger. Having multiple parties maintain the ledger and determine consensus improves reliability. All parties in the network should thus be able to maintain the ledger.

4.1.2 The information that is recorded in the ledger

In the monetary domain, blocks with sets of transactions are added to the ledger. In our architecture, events correspond to such blocks, while the associated data elements roughly correspond to the transactions. When an event is added to the ledger, it contains its associated data elements and their metadata. The data elements should at least contain the type of event, the time the event happened and an ID for the object that the event happened to (e.g. a container number). This is needed to make it possible for customs and others to link the event chronologically to other events concerning the same object.

An example of an event that could be added to the ledger is the stuffing of a container. The event should contain at least the following data elements (with example values):

- event type: container stuffed
- event time: 2017-04-14 09:22:00
- event object: KZDU3401208.

Additional data elements that could be associated with this event are, for instance, descriptions of the goods the container was stuffed with and the weight of the container. These additional data elements can be added to the event as well.

Metadata of the data elements are added to enable context-based access control. Furthermore, certain data elements will be encrypted to keep them confidential. When they are encrypted, metadata are needed to know what the data elements are and whether it would be interesting for parties to try and get access to them.

The metadata that could be stored together with the data element of the weight of the container in our example is the owner of this information, the method of measuring and the type of data. The content of the metadata stored with the data elements is further discussed in Section 4.4. In the following subsection, we discuss the format in which the data elements and their metadata are stored in the events.

4.1.3 The data elements in events and their Merkle trees

Events consist of a header and a body. The header contains information required to link the event to the chain and information required for the consensus mechanism. The body contains the data elements associated with the event. To add a new event to the ledger in step 1, first the data to be shared should be in the appropriate format. Each data element has metadata associated with it. The metadata of each data element could be stored together with it in a structure similar to that of a JSON object (Crockford 2006). To protect confidentiality, each data element associated with the event and elements of its metadata could be encrypted within the structure. We discuss the encryption and decryption of data elements in Section 4.5.

In the case of Bitcoin, transactions are hashed in a Merkle tree and then the Merkle root is stored in the header of a block to allow efficient and secure verification (Nakamoto 2008). A Merkle tree in our case allows all hashes of all data elements in an event to be stored in a single hash value (the Merkle root) (Luu et al. 2015). Furthermore, it allows for a simple proof of the existence of a data element in the tree (Luu et al. 2015; Merkle 1987). This makes it possible to check whether all data elements in a Merkle root are in an event, or whether they have been tampered with.

The Merkle tree based on the hashes is built as described by Merkle (1987) and Massias et al. (1999). The exact hash-function to be used in our architecture is outside of the scope of this paper. However, the avoidance of collisions is important to be able to check whether a data element is in the tree.

Figure 2 provides an example of what the Merkle tree containing the elements could look like. The data elements (in a structure together with their metadata), here denoted by $d1, ..., d4$, are hashed in to the hashes $H(d1), ..., H(d4)$, respectively. These are the leaves of the Merkle tree. The leaves are concatenated by two, denoted by $H(d1)|H(d2)$ and $H(d3)|H(d4)$. These concatenations are then hashed to a single values, viz. $H(H(d1)|H(d2))$ and $H(H(d3)|H(d4))$. The parent hashes obtained in that manner are then again concatenated by two and hashed again. This goes on until there is only a single hash left, the Merkle root.

In the case of Bitcoin, the Merkle tree is pruned to save disk space (Nakamoto 2008). In our case, we do not see the need for pruning parts of the tree. If disk space is a concern, the complete tree containing the data elements can be deleted to free disk space, while the

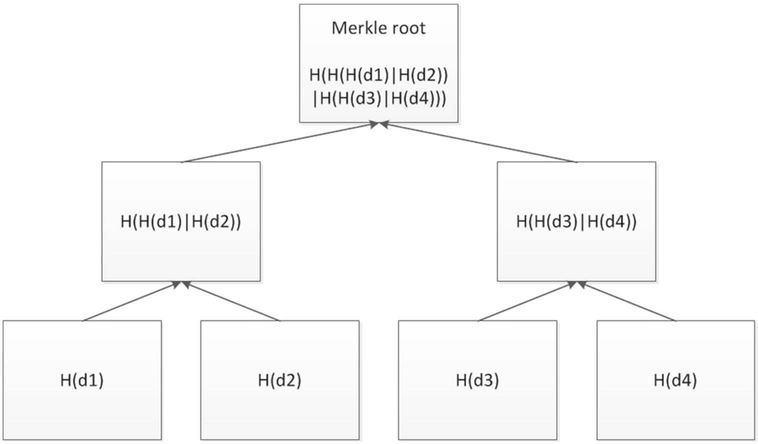

Fig. 2 Example of a Merkle tree for four data elements

header of the event is still stored. This can be done when the data on an event is no longer relevant, for instance when the goods have arrived at their destination. The Merkle root stored in the header is enough to be able to check whether a data element is part of an event.

4.1.4 The headers of events

In addition to data elements, events have a header. The header has seven fields:

Field 1. The Merkle root of the Merkle tree of the data elements associated with the event
Field 2. A hash of the header of the previous event in the blockchain
Field 3. A timestamp for when the event was added
Field 4. A unique ID for the party adding the event
Field 5. The Merkle root encrypted using the private key of the party adding the event
Field 6. A unique ID for the party confirming the event
Field 7. The Merkle root encrypted using the private key of the confirming party.

The inclusion of a hash of the header of the previous event links the events together in the blockchain, just as in the case of Bitcoin (Nakamoto 2008). The events are ordered in the chain chronologically; therefore each header should also contain a field with a timestamp for when it was created. When a party wants to add a new event, it adds the hash of the header of the event in the chain that it accepts with the most recent time stamp.

When there is a fork in the chain, the chain with the block with the lowest time stamp directly after the fork that should be accepted according to the consensus mechanism should be considered the actual blockchain. The party that added the other event should try again and add its event after that. In the rare case when time stamps are exactly the same, it should be possible to order the events in another way, for instance based on the value of the Merkle root.

Fields 1, 2 and 3 can be used to add a new event to the chain and broadcast it, completing steps 1 and 2. Since we do not need proof of work in our architecture, events can now be simply added and broadcast. However, we do need a consensus mechanism for the acceptance of events that are added to the blockchain (step 4). Fields 4, 5, 6 and 7 are needed for this consensus mechanism.

4.1.5 *The consensus mechanism*

Since we are concerned with sharing information that will be used to ensure safety and security, the correctness of the information is of vital importance. The consensus mechanism should help to ensure such correctness. To do so, for all events two parties must confirm that the data are correct. To fulfil step 3, only events for which such confirmation has been provided should be accepted by nodes.

Consider again the example of an event for the stuffing of a container. There could be a party that arranged the transport, such as a freight forwarder, which knows when the container was stuffed and also has a description of the goods that should be in the container. In that case, there is also a party that actually stuffed the container, such as the consolidator. When the event is added to the ledger by the freight forwarder, the forwarder can ask the consolidator for confirmation that the container was in fact stuffed and that the data elements in the event are correct.

To get consensus, there should be an easy way for nodes to check whether two parties in the network have confirmed the correctness of the information in the event. Businesses and government agencies in the network should be assigned unique IDs. For a party to confirm the information in an event, a link should be made between the party and the data elements in the event. Such a link is made when a party uses its private key to encrypt the Merkle root of a Merkle tree of the data elements in the event. Both the IDs of the parties that confirmed the information (fields 4 and 6) and the Merkle root that they encrypted (fields 5 and 7) should then be added to the header of the event.

Nodes can now easily check whether the information in the event was confirmed by two parties. For each confirming party, they first look up their public key using the ID of the party in the header. Next, they use this public key to decrypt the Merkle root. If the decryption succeeds, they know that the confirmation did in fact come from this specific party, since they used their private key to encrypt the Merkle root. This also makes it possible to hold them accountable for incorrect information. In addition, they know that the information in the event was in fact the information that they confirmed if the Merkle root is a Merkle root of a Merkle tree of the data elements in the event. When the nodes in the network accept the new event, we can continue to step 4, in which a party adds a new event to the chain on top of the accepted event.

4.1.6 *Procedural overview*

Based on the discussion above, we can add details to the steps mentioned in the introduction of this subsection. Step 1 (node adds a new event to the ledger as a new block) is fulfilled when the following steps are performed by the party that wants to add a new event:

Step 1.1 Gather (possibly encrypted) data elements associated with the event
Step 1.2 Build the Merkle tree of the data elements and add the Merkle root to the header
Step 1.3 Encrypt the Merkle root using private key
Step 1.4 Add the encrypted Merkle root and own ID to the header
Step 1.5 Send data elements and the Merkle tree to confirming party
Step 1.6 Receive back the Merkle root encrypted with the private key of the confirming party
Step 1.7 Add the encrypted Merkle root and the ID of confirming party to header
Step 1.8 Add hash of previous event in the chain to header
Step 1.9 Add timestamp to header.

The next step (step 2) is broadcasting the new event to other nodes. This step does not need to be divided into smaller steps. However, the third step in which the nodes accept the event according to a consensus mechanism, can be divided into smaller steps. To check whether a new event is an acceptable addition to the chain, nodes perform the following steps:

Step 3.1 Check whether the Merkle root in the header conforms with the data elements of the event
Step 3.2 Check whether the hash is a hash of the header of the previous event
Step 3.3 Decrypt encrypted Merkle roots using the public keys of the adding party and confirming party
Step 3.4 Check whether these Merkle roots are the same as the Merkle root in the header that was not encrypted.

In the fourth step, parties express acceptance by adding new events to the chain on top of the accepted event. This step thus refers back to the first one in which a new event is added. The following subsections concern obtaining access to the information that is distributed in the manner described in this subsection.

4.2 Business rules

Business rules in the architecture are used to specify who should have access to information. A business rule can be defined in various ways. Graham (2006, p. 7) defines business rules with an emphasis on the form and expressive power as follows: *"A business rule is a compact, atomic, well-formed, declarative statement about an aspect of a business that can be expressed in terms that can be directly related to the business and its collaborators, using simple unambiguous language that is accessible to all interested parties: business owner, business analyst, technical architect, customer, and so on."* While the form of business rules is of course important, here we are more concerned with their function. Work that focuses more on the function of the rules exists as well. For instance, Lee et al. (1999) view workflows as sets of business rules. However, the definition of Ross (2003) fits our purposes best, as he defines business rules as a directive intended to influence or guide business process behaviour. This conforms with the use of business rules in our research as an input for decision making on which parties get access to data. In this way, they influence and guide the information sharing process.

In the proposed architecture, the business rules are used as input for the decision made by the decision component. The specific business rules are provided and specified by the owners of the data and by the parties that own any data that the owners' data are based upon. By specifying these rules, they have control over who gets access to their data and they can keep information confidential according to their company's policy on access control. Each company might have different policies and therefore a company should be able to specify these rules itself. Since the business rules are requested each time a decision needs to be made, the owners of the data can change their business rules and thereby influence new decisions when their needs change.

Business rules can be used to specify precisely who does and who does not get access to a data element or even its metadata. For example, a business could specify that only a client that pays them extra gets access to the estimated arrival time of a container that contains their goods. Businesses could also use business rules to express more general policies on access. They could, for example, specify that in the case of an emergency, all data elements should be accessible to customs.

Generic businesses rules are not specific to a certain part of the data. They can be used to incorporate some general common sense in the decision making process. Furthermore, they could capture legal knowledge and be used to ensure that access to data is lawful. Capturing legal knowledge using business rules is not new; for instance, it is discussed in the work of Gong and Janssen (2014) and Palmirani et al. (2011).

Specifying exactly the way in which access to data is controlled and what the form of the business rules should be, is outside of the scope of this paper. Our focus is on the design of the architecture. However, to be able to provide some examples of business rules that could be used, we need a form to express them in.

Access control is discussed extensively in the literature. Karp et al. (2010) provide an overview of the different methods of access control, such as role-based access control (RBAC) and attribute-based access control (ABAC). According to Lee et al. (1999), first-order logic (FOL) is often proposed in the literature as a general framework that is suitable to formalise access control policies. It thus makes sense to use FOL to express the business rules in our examples as well. We emphasise, however, that further research is needed to determine whether FOL is in fact the appropriate format for the business rules in the architecture.

The format of the business rules used in the proposed architecture is highly dependent on the reasoning mechanism used in its decision component. There might be different kinds of possibly complex business rules, considering the large variety of businesses, the large variety of information to be shared, and the complexity of laws and legislation. Furthermore, the business rules of several businesses might be involved in making a decision on a specific part of data. It is therefore to be expected that contradictions will arise during the reasoning process. Moreover, it might be easier and more natural to express rules as being an exception to other rules, especially for generic rules based on the law. It is therefore highly likely that in the end there will be a need for a defeasible logic that is able to solve contradictions in a sensible manner. One could, for instance, think of some kind of argumentation logic, especially since such logics are quite often used in legal reasoning (Prakken and Vreeswijk 2002).

As discussed in Section 3.2.1, parties might want to keep information confidential for various reasons. Consider a scenario in which high-value goods are shipped. The freight forwarder might want to keep the goods description confidential from parties other than customs for fear of theft. They could encrypt the goods description before adding it to an event and then specify a specific business rule to control access to this information. The FOL sentence below shows what such a business rule might look like. The symbols have their usual meaning and the capitalised letters denote variables.

$$dataType(Data, goods\_description) \land \neg hasRole(Req, customs) \rightarrow \neg access(Req, Data) \quad (1)$$

When a party wants access to this information, the freight forwarder can send the rule to the decision component and thereby control access to the goods description.

As mentioned in Section 3.2.3, taking into account the goal-binding of information can be important for ensuring the lawfulness of information sharing. A generic business rule that takes into account the goal-binding principle could look as shown in example 2.

$$goalGathered(Data, G1) \land goalUse(Req, Data, G2) \land G1 \neq G2 \rightarrow \neg access(Req, Data) \quad (2)$$

When access to information is requested, the requester will be denied access according to this rule in cases in which the goal for gathering the information is not the same as the goal for its intended use by a party. Of course, this depends on the other rules that play a role as well.

4.3 The architecture components for access control

The architecture has two components that are necessary for controlling access, viz. a decision component and a component that allows access to data elements according to the decision. The decision component can be used by businesses or government agencies that want access to request a key. Based on the decision of the decision component, the second component provides the requester with a key that can be used to decrypt the data elements if the requester is allowed to access according to the decision.

The decision made in the decision component is based on several types of input from different information sources. There are two types of business rules (see Section 4.2) and two types of information (see Section 4.4). The first type of business rules are specific to the data to which access is requested. Each time the decision component needs to make a decision, it sends a request for specific business rules to the owners of the data that it needs to make a decision on and to the owners of the data that their data are based upon. The other type of business rules are generic business rules. They are the same for all decisions and are stored in the decision component itself.

The first type of information required by the decision component is context information on the requesters themselves and their intention to use the data. This information is send by the requester together with their request for access. The other type is metadata on the data elements they request access to. The metadata provides information about who is the owner of the data element and the party that is the owner of any data that the data element is based upon. Using this metadata the specific business rules can be requested from the appropriate parties. The decision component has access to its dedicated copy of the ledger to obtain metadata.

There are two possible outcomes of the decision process by the decision component, leading to two kinds of output of the component. The first is that the requester is not allowed access, in which case the requester is informed of this. The second possibility is that the requester is allowed access, in which case the decision and other information needed is sent to the other necessary component. This component generates a decryption key that can be used only by the requester of the information to decrypt the data elements that the requester is allowed to access according to the decision. This key is then sent to the requester. Figure 3 gives an overview of the components and their interfaces.

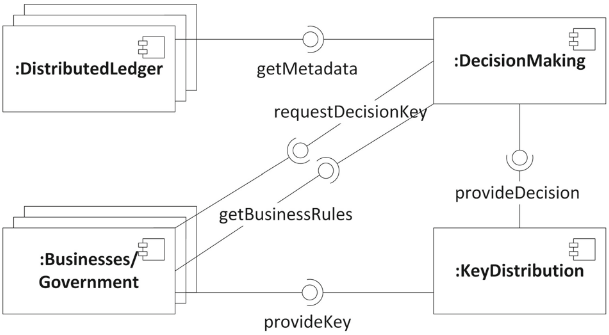

Fig. 3 UML component diagram for access control (Fowler and Scott 1999)

4.4 Metadata and context information

Decisions made by the decision component are based on metadata and context information. Metadata are often defined in the literature as data about data (Schuurman et al. 2008; Greenberg 2005). This simple definition is sufficient for this paper.

The data elements in events are all associated with the event. In some cases they are clearly data about the event, such as the event type. In other cases, such as the data element of container weight in a ‘container stuffed’ event, this association is more loose. All the data elements in the event themselves can also have metadata, such as the owner of the data element or when it was created.

In our case, the metadata will have to be rich enough for the decision component to be able to make a decision. It would make sense for the parties that add the data elements to an event, to generate all metadata as well. They will usually, if not always, be the same parties that control access to the data and deliver the specific business rules to the decision system. They will thus know what metadata are needed in order to make a decision. Furthermore, some metadata should always be provided, such as metadata containing information on the owners of data elements and the parties that are the owners of any data that the metadata are based upon, since otherwise it will not be possible to specify the parties the specific business rules should be requested from. Furthermore, metadata might be needed to decide whether a party should be allowed access according to the generic rules as well.

An additional function of the metadata is to let parties that want to access data elements in an event know what information is in there. They might already have some idea in some cases because of the events mentioned, but since the data elements themselves can be encrypted, there is no other way for them to know this directly and to determine whether or not they would like to have access. In the events, data elements are stored together with their metadata in the same structure. It might often be the case within this structure that the data element itself (e.g. goods description) is encrypted, while its metadata (e.g. the type of data element) are not.

To have access to the metadata, the decision component needs access to a copy of the distributed ledger. There might be cases in which metadata should be kept confidential, for instance because otherwise the volumes of goods transported by a business could be derived, which is competitively sensitive information. There is no clear reason in such cases not to allow the encryption of metadata.

In the scenario in which business rule (1) was specified, the metadata of data shared by the freight forwarder should include the data type of the goods description and the metadata identifying them as a party whose business rules have to be requested by the decision component. For example, for business rule (2) the metadata should include the goal for gathering the data elements.

In addition to the metadata, the decision component also bases its decision on context information. Context information is data generated by the requester of access to the information and concerns the context in which access to information is requested. The requester can send this context information together with the request for access to the decision component.

In the scenario of business rule (1), the context information required to make a decision would be the role of the requester. If this is anybody but customs, access to the information might be denied. In the scenario of business rule (2), the context information that is required is the goal for which the requester wants to use the information. If this is not the same as the goal for which the data were gathered, access might be denied by the decision component.

4.5 Regulating access via encryption and decryption of parts of data

Access to data is usually regulated by sending or not sending data to others or by allowing or not allowing others access to a database. This is different for the architecture we propose. In this architecture, access is regulated by encrypting and decrypting parts of data. This means that it is possible for businesses to encrypt data elements or their metadata and distribute them to others, without automatically granting them access. As a result, the access to data and the location where the data are saved are not linked. In other words, physical access to information does not imply logical access.

The fact that access and location are no longer linked allows organisations to enrich data or combine data in the ledger in useful ways that they might not have even known about otherwise. In order for this to work, the enriched data or combined data should also be encrypted before adding it to events, and the rules of the owners of the original data should also be applicable to the new data that are based on them. An example of the enrichment of data is a company adding the data element of the weight of containers, based on information received about the weight of goods and its own information about in which containers goods are. If they share this data element, the business rules of the owner of the data elements containing the weight of the goods, and those of the owner of the information that this data element is based upon, would be applicable, as would of course the business rules of the business itself.

5 Evaluation of the architecture

In this section, we first demonstrate the way in which information is shared using the architecture. We then present an evaluation of the architecture based on each of the requirements we specified in Section 3.2. We used the same workshops for the evaluation as for the requirements.

5.1 A demonstration

In design science, artefacts need to be demonstrated. To evaluate the way in which the proposed architecture works and how the components and users relate to each other, a demonstration was developed of a typical user activity. We used an example of B2G information sharing in the container-shipping domain to illustrate how information can be shared using the architecture. The example is shown in Fig. 4.

The example in this illustration is the same as that in Section 4. So, the freight forwarder arranges the transport of goods and the consolidator stuffs the goods in a container. The freight forwarder wants to add the stuffing of the container as an event that contains a detailed description of the goods in the container. The consolidator will be the party confirming the correctness of the information in the event. Since the freight forwarder is adding the goods description to the event, the forwarder adds itself in its metadata as a party that should be consulted for access control.

It is important to note that if data elements, such as the goods description, are based on data from others, these parties should be added to the metadata of the data element as parties that should also be consulted for access control. This possibility was left out of the example to protect the clarity and intelligibility of the activity diagram in Fig. 4. To include it would

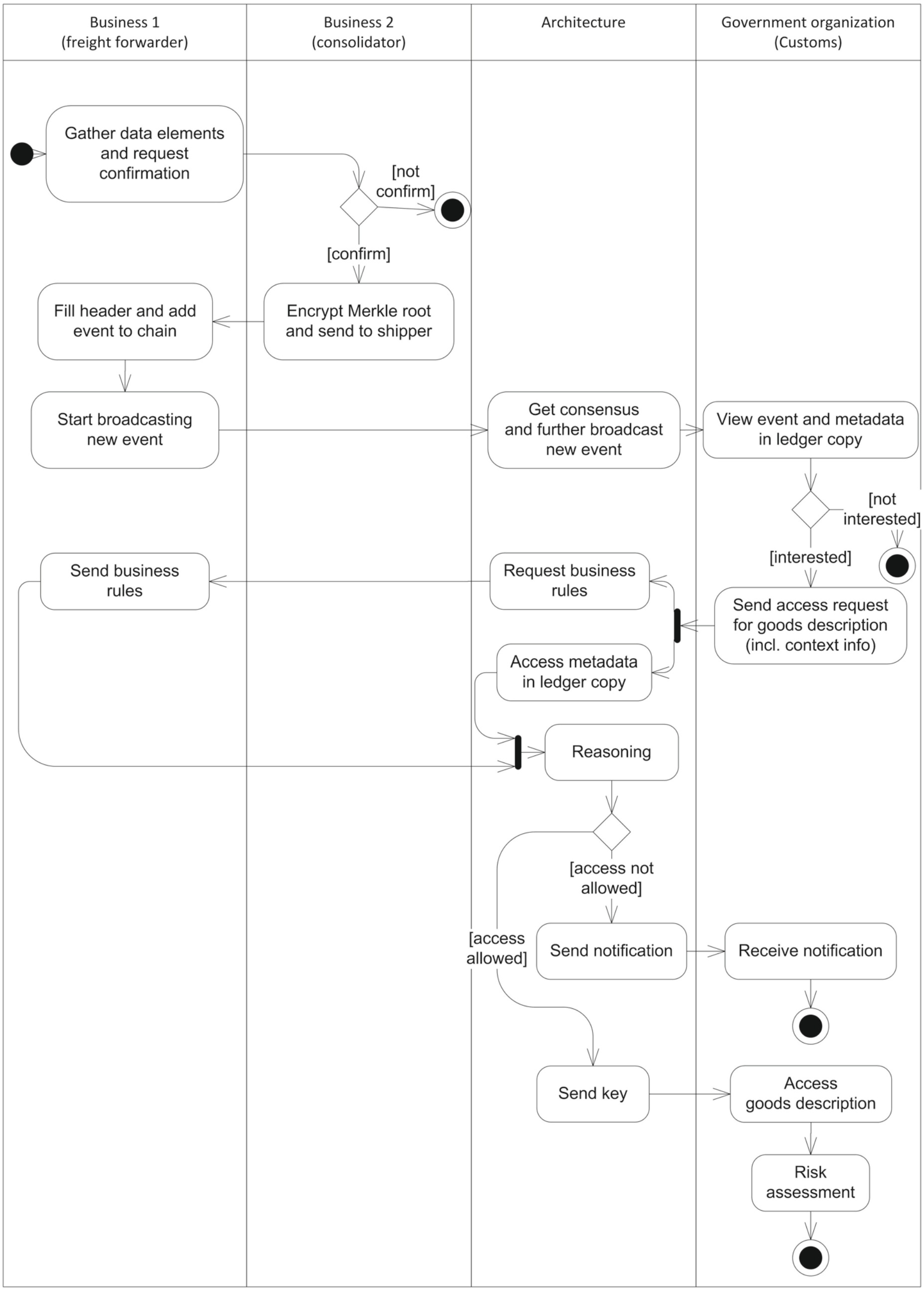
Business 1
(freight forwarder)
Business 2
(consolidator)
Architecture
Government organization
(Customs)
Gather data elements and request confirmation
[not confirm]
[confirm]
Fill header and add event to chain
Encrypt Merkle root and send to shipper
Start broadcasting new event
Get consensus and further broadcast new event
View event and metadata in ledger copy
[not interested]
[interested]
Send business rules
Request business rules
Send access request for goods description (incl. context info)
Access metadata in ledger copy
Reasoning
[access not allowed]
[access allowed]
Send notification
Receive notification
Send key
Access goods description
Risk assessment

Fig. 4 UML activity diagram demonstration of typical user activity (Fowler and Scott 1999)

require the request for business rules to go to multiple parties and the addition of lanes for new parties from which business rules should also be obtained.

If the goods that are stuffed in the container are valuable, the freight forwarder might want to keep the goods description confidential for fear of theft. The forwarded thus encrypts the goods description itself and specifies business rules for who should get access. The forwarder does not encrypt the metadata of the goods description, including its data type. This allows parties who need access to know that the event contains a description of the goods in the container. In this example, customs would like access to the detailed goods description in the event to determine whether they should perform a physical inspection.

To add the 'container stuffed' event and distribute it, the freight forwarder and the consolidator first follow the steps described in detail in Section 4.1.6. When the event is distributed, customs will have the event and associated data elements in its copy of the ledger as well. Customs can use the metadata of the encrypted goods description to determine whether it is interesting for them to obtain access.

If customs does want to have access to the goods description, it sends a request to the decision component of the architecture, together with information on the context in which it requests access. Next, the decision component gets the appropriate metadata from its own copy of the ledger. It then requests the freight forwarder, which should be consulted according to the metadata, to supply its business rules.

The decision component uses these business rules, together with the generic business rules, metadata and context information, to reach a decision on whether customs should be allowed access. If customs cannot have access according to the rules, it is sent a notification of this. If customs is allowed access, it is sent a key to decrypt and access the goods description in the event. Customs can now use it for risk assessment.

5.2 Evaluation of requirement 1: Information should be kept confidential according to the needs of businesses

For the first requirement to be met, the architecture should allow access to confidential information if and only if this is according to the needs of businesses. This makes three scenarios important for the evaluation, viz. scenarios in which a party requesting access 1) should not be allowed access according to the needs of businesses, 2) should be allowed access according to the needs of businesses, and 3) should be allowed access to the needs of some businesses, but not according to the needs of others.

In the first scenario, businesses can encrypt data elements and formulate business rules to deny access. When information is confidential to multiple businesses, all these businesses could be added to the metadata as parties that should be consulted by the decision component. In this way, information can be kept confidential according to the needs of multiple businesses. This also obviates the need for businesses to make arrangements with others when sharing information from others or based on information from others. Instead, they can simply add the other businesses to the metadata as parties to be consulted and provide them with control in that way.

There is the possibility that businesses add data elements without encryption when needed, or without mentioning the appropriate businesses to consult. In such cases, confidentiality might be compromised. The risk of doing this by accident might be lower than in other architectures, since another party has to confirm the data and might notice the mistake. The risk of deliberate sharing of illicit information might also be lower, since an accomplice first needs to be found. In addition, if the data are not encrypted, there is a high risk of being found out. While the risks might be low, the consequences of compromised confidentiality can be high, since information could be shared unprotected with all parties in the network.

Businesses themselves are responsible for formulating business rules according to their needs. Making a refined specification might make them realise more clearly what information actually needs to be kept confidential and from whom. A consequence of this could be a more refined distinction between information that other parties can and cannot have access to. This might result in an increased willingness to share information.

For businesses, a disadvantage of having to formulate business rules is that when their needs are complex and there is a large variety of data elements, some expertise and resources could be required. In addition, a high complexity of business rules means more complex reasoning by the decision component. This increases the risk of errors and unpredictable behaviour of the decision component, which might eventually threaten trust in confidentiality.

For the second scenario, in which a party requesting access should be allowed access according to the needs of businesses, businesses can simply choose to not encrypt data elements, so everyone can have access. However, if they foresee that their needs will change in the future, they can encrypt data elements and allow everybody access according to their business rules. Since business rules are requested from parties for each decision, they can simply change their business rules when they want information to be kept confidential at a later stage. This is also possible the other way round, if information no longer needs to be confidential after a certain period of time.

In the third scenario, in which information access to a data element is allowed according to the business rules of one party (e.g. a freight forwarder) but not according to the business rules of another (e.g. a consolidator), access should not be allowed. One could argue that this is not in accordance with the needs of the freight forwarder in this example. It could, for instance, benefit from allowing access and this might increase its willingness to share information. However, the consolidator would no longer be ensured that information is kept confidential. It is likely that in that case, the consolidator would not find the architecture acceptable at all, which is directly contrary to our objective.

In all scenarios, it might be important for parties to keep track of which parties have had access to their data. This could be arranged by making the decision component also send its decision to the parties it has requested business rules from. This only works for encrypted data elements, so if parties want to keep track of access, they should encrypt their information.

5.3 Evaluation of requirement 2: Information should be shared while allowing businesses to avoid liability

For this requirement to be met, information that could increase a business's liability should be shared in the architecture without increasing liability. For our evaluation, a relevant scenario is thus one in which information is shared in the architecture that would have increased a business's liability if it were shared directly. An example of such a information is described in Section 3.2.2.

Whether a business's liability can be increased by sharing information using the architecture is unclear. However, we can say something about what would be reasonable. When a business is part of the network, it has a copy of the ledger. When information is in the ledger, the business thus has a copy of the information in its system. In our architecture, however, physical access does not imply logical access. It seems reasonable that liability should not be increased when a business has a copy of the ledger, but does not have and cannot get access to the information.

For a business not to have access to information in the architecture, the information needs to be encrypted. Furthermore, for a business not to be able to obtain access, it should not be allowed access according to a decision made by the decision component. Such a decision could also function as 'proof' that the business did not have access.

For a business not to have access to information, another party should provide business rules denying it access. In the situation without the architecture, parties make an effort not to share information for fear of increasing others' liability, such as the shipper in the example in Section 3.2.2. Similarly, they might make an effort to add business rules and encrypt information in the proposed architecture as well.

5.4 Evaluation of requirement 3: Information sharing using the architecture should be lawful

The third requirement is met when information is shared via the architecture only when it is lawful to do so. For our evaluation, two scenarios are thus important. In the first, information sharing is lawful; in the second, it is not.

In the first scenario, access to the information can be prohibited when it is unlawful by specifying the appropriate generic rules. For the second scenario, the generic rules should be specified such that access is allowed when information sharing is lawful. Whether the third requirement is met thus depends on the quality of the generic rules and the quality of the decision component.

Businesses using the architecture do not need to formulate these generic rules themselves, since they are part of the decision component. Furthermore, they do not need insight into the law, nor to track and adapt to changes to the law. The latter can be taken care of by changing the generic rules in the decision component.

While at first sight this would be an ideal situation for businesses in the architecture, there might be some problems. If businesses are not going to formulate the rules themselves and keep track of changes in the law, who is going to do that? The responsibility to be compliant in a sense shifts from the users of the architecture to the organisation that specifies the generic business rules. This might result in some ethical and legal difficulties. Such an organisation would have a lot of power, since it would have an influence on all requests for access. A solution could be to let the decision component be governed by the parties in the network themselves, solving some of the ethical difficulties with responsibility and distribute between the users the power over the contents of the generic business rules.

It is unclear whether the legislation that is applicable to information that is directly accessible is applicable to information that is encrypted and is not directly accessible. However, for the same reasons as in the situation with the increase in liability, it might be reasonable to say that this should not be the case. If access is only granted to data elements when this is lawful, it would be reasonable for users of the architecture to be confident that they are complying with the law in the first scenario. This in turn might lead to an increase in their willingness to share information.

To summarise, we found the proposed architecture to have several advantages. Due to the use of blockchain technology, information sharing is reliable and secure. Furthermore, multiple businesses can exercise flexible control over data elements to keep them confidential. Moreover, there are indications that it is possible to protect businesses from increased liability. Another advantage is that if generic rules are specified and kept up to date with the law, businesses might not need to be concerned with making sure that information sharing is compliant.

6 Conclusion and suggestions for further research

Using a design science approach, we developed a software architecture for the sharing of information with government organisations that can be acceptable to businesses. This enables government organisations to receive more information that can be used for better and more focussed inspections. This, in turn, contributes to public safety and security. Based on a case, we derived the requirements for the architecture, viz.: 1) information should be kept confidential according to the needs of businesses, 2) information should be shared while allowing businesses to avoid liability, and 3) information sharing using the architecture should be lawful.

The architecture meeting these requirements consists of five main components; 1) blockchain for recording events, 2) business rules for setting the conditions to share information, 3) access control to ensure only authorised access, 4) metadata and context information to understand whether the context enables information to be shared, and 5) encryption and decryption.

Blockchains enable the sharing of events and ensure the trustworthiness of information in these events. Business rules can be specified by the businesses. This enables them to keep their data confidential whenever they need to. In addition, in the architecture the sharing of data that are received by others, that are enriched or combined, is made easier by making it possible for the business rules of multiple parties to be applicable. Furthermore, it seems that the use of generic business rules to ensure that data access complies with legislation, is also a means to increase willingness to share information. The way the generic business rules and the decision component are governed is important for this.

Although there are architectures in each of the domains, to our knowledge this is the first to combine events, a blockchain and business rules. Only the combination satisfies the three main requirements. This suggests that the combination of technologies can also provide advantages in other domains.

Future research should focus on further evaluating the architecture by requesting a review of the architecture by industry experts and experts in the juridical domain. In addition, the components of the architecture, such as the decision component and the encryption and decryption mechanisms, should be further developed.

Acknowledgements The work in this paper is part of the project JUridical and context-aware Sharing of informaTion for ensuring compliance (JUST). This project is funded by the Netherlands Organisation for Scientific Research (NWO) as part of the ISCOM programme (Innovation in Supply Chain Compliance and Border management) under grant number 438-13-601.

References

Bass, L., Clements, P., Kazman, R., & Northrop, L. (2003). What is software architecture? In *Addison-Wesley Professional* (pp. 19–46).

Bharosa, N., Janssen, M., Van Wijk, R., De Winne, N., Van der Voort, H., Hulstijn, J., & Tan, Y.H. (2013). Tapping into existing information flows: The transformation to compliance by design in business-to-government information exchange. *Government Information Quarterly*, *30*, S9–S18. doi:10.1016/j.giq.2012.08.006.

Buterin, V. (2015). On Public and Private Blockchains. https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/.

Crockford, D. (2006). The application/json media type for JavaScript object notation. RFC-4627 (July):1–10, doi:10.17487/rfc4627.

Customs Administration of the Netherlands (2014). Pushing boundaries: The Customs Administration of The Netherlands' Point on the Horizon for the Enforcement on Continuously Increasing Flows of Goods.

European Court of Human Rights (2010). European Convention on Human Rights.

European Economic and Social Committee (2003). Opinion of the european economic and social committee on the 'security of transports'. *Official Journal of the European Union*, *46*, 174–183.

European Union Agency for Fundamental Rights and The Council of Europe (2014). *Handbook on European data protection law*. Publications Office of the European Union. doi:10.2811/69915.

Fawcett, S.E., Osterhaus, P., Magnan, G.M., Brau, J.C., & McCarter, M.W. (2007). Information sharing and supply chain performance: the role of connectivity and willingness. *Supply Chain Management: An International Journal*, *12*(5), 358–368. doi:10.1108/13598540710776935.

Fowler, M., & Scott, K. (1999). UML distilled: *a brief guide to the standard object modeling language*. Addison-Wesley.

Gong, Y., & Janssen, M. (2014). A framework for translating legal knowledge into administrative processes: dynamic adaption of business processes. In *Information technology and open source: applications for education, innovation, and sustainability* (204–211). Springer.

Graham, I. (2006). *Business rules management and service oriented architecture*. Wiley.

Greenberg, J. (2005). Understanding metadata and metadata schemes. *Cataloging & Classification Quarterly*, *40*(3–4), 17–36. doi:10.1300/J104v40n03.

Gregor, S., & Hevner, A.R. (2013). Positioning and presenting design science research for maximum impact. *MIS Quarterly*, *37*(2), 337–355. doi:10.2753/MIS0742-1222240302.

Hart, P., & Saunders, C. (1997). Power and trust: Critical factors in the adoption and use of electronic data interchange. *Organization Science*, *8*(1), 23–42. doi:10.1287/orsc.8.1.23.

Hesketh, D. (2010). Weaknesses in the supply chain: Who packed the box. *World Customs Journal*, *4*(2), 3–20.

Hevner, A.R., March, S.T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly*, *28*(1), 75–105. doi:10.2307/25148625, /dl.acm.org/citation.cfm?id=2017212.2017217.

Jensen, T., & Vatrapu, R. (2015). Ships & roses: A revelatory case study of affordances in international trade. In *Proceedings ECIS 2015* (pp. 1–18), doi:10.18151/7217370.

Jensen, T., Bjørn-andersen, N., & Vatrapu, R. (2014). Avocados crossing borders: The missing common information infrastructure for international trade. In *Culture in international context* (pp. 15–24), doi:10.1145/2631488.2631500.

Karampetsou, A. (2016). Container information & privacy concerns: opening the "Pandora's" box? Legal challenges of a Business-to-Customs Information Sharing with regard to Containerized Cargo. In *Current issues in maritime & transport law* (pp. 1–17). Bologna: Bonomo Editore.

Karp, A.H., Haury, H., & Davis, M.H. (2010). From ABAC to ZBAC: the evolution of access control models. In *International conference on cyber warfare and security.* Academic Conferences International Limited.

Klievink, B., & Lucassen, I. (2013). Facilitating adoption of international information infrastructures: a Living Labs approach. In *International conference on electronic government* (pp. 250–261). Springer Berlin Heidelberg. doi:10.1007/978-3-642-40358-3_21.

Klievink, B., Janssen, M., & Tan, Y.H. (2012a). A stakeholder analysis of business-to-government information sharing. *International Journal of Electronic Government Research*, *8*(4), 54–64. doi:10.4018/jegr.2012100104.

Klievink, B., Van Stijn, E., Hesketh, D., Aldewereld, H., Overbeek, S., Heijmann, F., & Tan, Y.H. (2012b). Enhancing visibility in international supply chains: The data pipeline concept. *International Journal of Electronic Government Research (IJEGR)*, *8*(4), 14–33. doi:10.4018/jegr.2012100102.

Klievink, B., Aldewereld, H., & Tan, Y.H. (2014). Establishing information infrastructures for international trade: discussing the role and governance of port-community systems. In *5th international conference on information systems, logistics and supply chain (ILS2014)* (pp. 1–10). Dinalog.

Lee, H.B., Kim, J.W., & Park, S.J. (1999). KWM: Knowledge-based workflow model for agile organization. *Journal of Intelligent Information Systems*, *13*(3), 261–278. doi:10.1023/A:1008773617579.

Lee, H.L., & Whang, S. (2000). Information sharing in a supply chain. *International Journal of Manufacturing Technology*, *1*(1), 79–93.

Levi, M.D. (2005). *International finance*, 4th Edn. Routledge.

Levinson, M. (2010). The world the box made. In *The Box: how the shipping container made the world smaller and the world economy bigger* (pp. 1–15). Princeton University Press.

Luu, L., Narayanan, V., Baweja, K., Zheng, C., Gilbert, S., & Saxena, P. (2015). SCP: a computationally-scalable byzantine consensus protocol for blockchains. *IACR Cryptology ePrint Archive* (pp. 1–16).

Mainelli, M., & Smith, M. (2015). Sharing ledgers for sharing economies: an exploration of mutual distributed ledgers (aka blockchain technology). *The Journal of Financial Perspectives*, *3*(3), 38–69.

Massias, H., Serret Avila, X., & Quisquater, J.J. (1999). Design of a secure timestamping service with minimal trust requirement. In *20th symposium on information theory in the Benelux*.

Mentzer, J.T., Dewitt, W., Keebler, J.S., Min, S., Nix, N.W., Smith, C.D., & Zacharia, Z.G. (2001). Defining supply chain management. *Journal of Business Logistics*, *22*(2), 1–25. doi:10.1002/j.2158-1592.2001.tb00001.x.

Merkle, R.C. (1987). A digital signature based on a conventional encryption function. In Pomerance, C. (Ed.) *Advances in cryptology - CRYPT0 87, lecture notes in computer science* (Vol. 369–378). Springer Berlin Heidelberg. doi:10.1007/3-540-48184-2_32.

Mishra, A.K. (1996). Organizational responses to crisis: the centrality of trust. In Kramer, R.M., & Tyler, T. (Eds.) *Trust in organizations* (pp. 261–287). Sage: Newbury Park, CA.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. doi:10.1007/s10838-008-9062-0, 43543534534v343453.

NWO (2013). Innovation in Supply Chain Compliance and Border Management (ISCOM). https://www.nwo.nl/onderzoek-en-resultaten/onderzoeksprojecten/i/95/11895.html.

Overbeek, S., Janssen, M., & Tan, Y.H. (2012). An event-driven architecture for integrating information, processes and services in a plastic toys supply chain. *International Journal of Cooperative Information Systems*, *21*(04), 343–381. doi:10.1142/S0218843012500062.

Palmirani, M., Governatori, G., Rotolo, A., Tabet, S., Boley, H., & Paschke, A. (2011). LegalRuleML: XML-Based rules and norms. In *RuleML 2011 - America* (Vol. 7018, pp. 298–312). Berlin Heidelberg: Springer-Verlag. doi:10.1007/978-3-642-24908-2_30.

Peffers, K., Tuunanen, T., Rothenberger, M.A., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems*, *24*(3), 45–77. doi:10.2753/MIS0742-1222240302.

Pilkington, M., & Zhegu, M. (2016). Blockchain technology: Principles and applications. In Olleros, F.X. (Ed.) *Research handbook on digital transformations* (227–253). Edward Elgar Publishing.

Prakken, H., & Vreeswijk, G. (2002). Logics for defeasible argumentation. In Gabbay, D., & Guenthner, F. (Eds.) *Handbook of philosophical logic* (Vol. 4, pp. 218–319). Kluwer Academic Publishers. doi:10.1007/978-94-017-0456-4_3.

Ross, R.G. (2003). *Principles of the business rule approach*. Addison-Wesley Professional.

Schuurman, N., Deshpande, A., & Allen, D.M. (2008). Data integration across borders: a case study of the Abbotsford-Sumas aquifer (British Columbia/Washington State). *Journal of the American Water Resources Association*, *44*(4), 921–934.

Tan, Y.H., Bjørn-Andersen, N., Klein, S., & Rukanova, B. (2011). Accelerating Global Supply Chains with IT-Innovation. doi:10.1007/978-3-642-15669-4.

The Commission Of The European Communities (2006). EU Regulation 1875/2006.

The Software Engineering Standards Committee of IEEE Computer Society (2000). 1471-2000 - IEEE Recommended Practice for Architectural Description for Software-Intensive Systems. Technical Report. 42010, IEEE Computer Society. doi:10.1109/IEEESTD.2000.91944.

Tsay, A.A., Nahmias, S., & Agrawal, N. (1999). Modeling supply chain contracts: A review. In Tayur, S., Ganeshan, R., & Magazine, M. (Eds.) *Quantitative models for supply chain management* (pp. 299–336). US: Springer. doi:10.1007/978-1-4615-4949-9_10.

Urciuoli, L., Hintsa, J., & Ahokas, J. (2013). Drivers and barriers affecting usage of e-Customs — A global survey with customs administrations using multivariate analysis techniques. *Government Information Quarterly*, *30*(4), 473–485. doi:10.1016/j.giq.2013.06.001.

van Engelenburg, S., Janssen, M., & Klievink, B. (2015). Design of a business-to-government information sharing architecture using business rules. In *Software engineering and formal methods* (Vol. 9509, pp. 124–138). Springer. doi:10.1007/978-3-319-15201-1.

Van Baalen, P., Zuidwijk, R., & Van Nunen, J. (2009). Port inter-organizational information systems: Capabilities to service global supply chains. *Foundations and Trends in Technology Information and Operations Management*, *2*(2-3), 81–241. doi:10.1561/0200000008.

Van Stijn, E., Hesketh, D., Tan, Y.H., Klievink, B., Overbeek, S., Heijmann, F., Pikart, M., & Butterly, T. (2011). Annex 3: The data pipeline. In *Connecting international trade: single windows and supply chains in the next decade, united nations economic commission for Europe* (pp. 158–183).

Veenstra, A.W., Hulstijn, J., Christiaanse, R., & Tan, Y.H. (2013). Information exchange in global logistics chains: an application for model-based auditing. In *BNAIC 2013: Proceedings of the 25th Benelux conference on artificial intelligence*. Delft, Netherlands.

Walport, M. (2015). *Distributed ledger technology: Beyond block chain.* Technical report, UK: Government Office for Science.

Zomer, G.R. (2011). Smart trade logistics - compliance as an opportunity. In *IT innovations enabling seamless and secure supply chains witness 2011* (Vol. 769, pp. 9–19).