IRMA and Verifiable Credentials

IRMA and Verifiable Credentials

What is their relation?

Daniel Ostkamp

Student: 851935822
Date: 2020-05-27

IRMA and Verifiable Credentials

What is their relation?

by

Daniel Ostkamp

Student number:	851935822	
Course code:	IM9906	
Thesis committee:	Dr. Greg Alpar, PhD	Open University
	Dr. Fabian van den Broek, PhD	Open University

Faculteit Bètawetenschappen
Open University of the Netherlands
Graduation Assignment

ACKNOWLEDGEMENTS

First, I would like to thank my supervisor Greg Alpar. Without his valuable feedbacks via our regular meetings and mail exchanges I would not be able to deliver this work in such quality.

Second, I would like to thank Fabian van den Broek as the chair of this research to provide me with feedback on my final draft.

Third, I would like to thank the IRMA team for their expert insight into the different IRMA components.

Last but not least, I would like to thank Hylke Foeken for his review of my final draft. Your comments have improved the correctness and readability of this thesis.

Contents

List of Figures

List of Tables

LIST OF SOURCE CODES

ABSTRACT

In this master's thesis, we analyze the relation between the IRMA ("I Reveal My Attributes") system and the Verifiable Credential (VC) standard. By contrasting and comparing IRMA with VC, we identify similarities and differences.

IRMA is a practical, privacy-friendly, and user-centric identity system. It is user-centric as the user, and not the identity provider, is the central role within the system. The goal of the VC standard is to advance interoperability between user-centric identity systems. A deliverable of the VC standard is a data model with related concepts and syntax. Hence, we look into possibilities on how to extend IRMA's interoperability by making IRMA compliant with the VC standard.

Several concepts between IRMA and VC overlap, such as claims and credentials. A credential is a set of claims, for instance, a full name or address of a subject. The credential is produced and signed by an issuing party, such as the government. A credential is verifiable if the claims can be cryptographically verified. Also, IRMA employs Zero-Knowledge Proofs, and VC supports the use of Zero-Knowledge Proofs.

We identify the relevant requirements of the VC data model and real-world use-cases between IRMA and other VC-compliant systems. Based on the requirements and use-cases, we describe modifications needed within IRMA. We suggest introducing a metadata server, which can deliver data from IRMA schemes to external actors via HTTP endpoints. When accepting messages from other VC-compliant systems, IRMA's security properties can be significantly impacted.

We implement a subset of the identified modifications to compute VC-compliant messages within IRMA. The developed prototype can exchange those messages between IRMA components. We publish the modifications of the irmago and irma_mobile components via forks on GitHub. The limitation is that we do not modify the IRMA data model to comply with the VC data model as this is out-of-scope of this research.

We conclude that after modifying IRMA to comply with the VC standard, IRMA extends its interoperability by at least reaching level three of the "levels of conceptual interoperability model" (LCIM). However, it is one thing to make a system compliant with a standard. It is another to exchange messages with other VC-compliant systems as those might use different protocols, APIs, and other technology such as blockchains.

We recommend the Privacy By Design Foundation to modify IRMA to comply with the VC standard. An advantage might be to become a pioneer in the field within the Netherlands or even the EU. This can increase PbDF's visibility and impact. In general, the interoperability of user-centric IMS should get more focus within institutions and industry. Consequently, citizens may see the benefits of such interoperable systems. Additionally, it can reduce the resistance against such systems.

Samenvatting

In deze masterscriptie analyseren we de relatie tussen het IRMA ("I Reveal My Attributes") systeem en de Verifiable Credential (VC) standaard. Door IRMA te contrasteren en te vergelijken met VC identificeren we overeenkomsten en verschillen.

IRMA is een praktisch, privacy-vriendelijk en gebruikersgericht identiteitssysteem. Het is gebruikersgericht want de gebruiker, en niet de identity provider, is de centrale rol binnen het systeem. Het doel van de VC-standaard is om de interoperabiliteit tussen gebruikersgerichte identiteitssystemen te bevorderen. Een product van de VC-standaard is een data model met bijbehorende concepten en syntax. Daarom onderzoeken we de mogelijkheden om de interoperabiliteit van IRMA uit te breiden door IRMA te laten voldoen aan de VC-standaard.

Verschillende concepten tussen IRMA en VC overlappen, zoals claims en credentials. Een credential is een set van claims, bijvoorbeeld een volledige naam of adres. De credentiaal wordt geproduceerd en ondertekend door een uitgevende partij, zoals de regering. Een credentiaal is verifieerbaar als de claims cryptografisch kunnen worden geverifieerd. Ook maakt IRMA gebruik van Zero-Knowledge Proofs en VC ondersteunt het gebruik van Zero-Knowledge Proofs.

We identificeren de relevante vereisten van het VC-data model en praktijkgevallen tussen IRMA en andere VC-compliant systemen. Op basis van de eisen en use-cases beschrijven we de benodigde aanpassingen binnen IRMA. We stellen voor een metadata server in te voeren, die via HTTP-eindpunten gegevens van IRMA-schema's aan externe actoren kan leveren. Bij het accepteren van berichten van andere VC-compatibele systemen kunnen de beveiligingseigenschappen van IRMA aanzienlijk worden beïnvloed.

We implementeren een deelverzameling van de geïdentificeerde wijzigingen om VC-conforme berichten binnen IRMA te berekenen. Het ontwikkelde prototype kan deze berichten uitwisselen tussen IRMA-componenten. We publiceren de modificaties van de irmago en irma_mobile componenten via forks op GitHub. De beperking is dat we het IRMA data model niet aanpassen om te voldoen aan het VC data model, omdat dit buiten scope van dit onderzoek valt.

We concluderen dat na het aanpassen van IRMA om te voldoen aan de VC standaard, IRMA haar interoperabiliteit uitbreidt met tenminste het bereiken van niveau drie van de "levels of conceptual interoperability model" (LCIM). Het is echter één ding om een systeem te laten voldoen aan een standaard. Het is een ander ding om berichten uit te wisselen met andere VC-conforme systemen, omdat die andere protocollen, API's en andere technologie zoals blockchains kunnen toepassen.

We raden de Privacy By Design Foundation aan om IRMA aan te passen aan de VC-standaard. Een voordeel zou kunnen zijn een pionier te worden op dit gebied binnen Nederland of zelfs de EU. Dit kan de zichtbaarheid en impact van PbDF vergroten. In het algemeen zou de interoperabiliteit van gebruikersgerichte IMS meer aandacht moeten krijgen binnen instellingen en het bedrijfsleven. De burger kan dan ook de voordelen van dergelijke interoperabele systemen inzien. Bovendien kan het de weerstand tegen dergelijke systemen verlagen.

GLOSSARY

Attribute An assertion made about a subject, also referred to as *claim*.

Claim An assertion made about a subject, also referred to as *attribute*.

Credential A set of one or more claims made by an issuer about one or more subjects.

Holder A role possessing one or more credentials.

Issuer A role asserting claims about one or more subjects, creating a credential from these claims, and transmitting the credential to a holder.

Linked data Data that is interlinked with other data to get context. It can be queried by software.

Presentation Data derived from one or more verifiable credentials that is shared with a specific verifier.

Proof Same as Signature.

Scheme A data structure within IRMA containing information about issuers, credential types, and attribute names).

Scheme manager A role within IRMA. It maintains and distributes one or more IRMA scheme(s).

Signature A (in our case digital) signature is a technique for verifying the authenticity of digital messages.

Subject A thing about which claims are made.

User-centric IMS A decentral IMS putting the user in control of her data.

Verifiable Credential A tamper-evident credential signed by an issuer that can be cryptographically verified.

Verifiable Presentation A verifiable presentation is a tamper-evident presentation. Authorship of the data can be trusted after a process of cryptographic verification.

Verifier A role checking a subjects identity to authorize subjects to access resources.

1

Introduction

"Study after study has shown that human behavior changes when we know we're being watched. Under observation, we act less free, which means we effectively are less free." (Edward Snowden, from his book *Permanent Record*).

1.1. Problem context and contributions

Traditionally, most operating identity management systems (IMS) have one essential flaw. According to Allen [43], the flaw is that centralized authorities have control over the identities of individuals, for instance, Facebook or Google. This is referred to as "service-centric". Individuals are bound to the central authority and do not have complete control of their identity. For instance, individuals can not transfer their identity from Facebook to Google or vice versa. An identity of an individual consists of attributes, also called claims, about the individual, for example, her birth date or e-mail address.

Also, organizations in both public and private sectors exchange personal information of individuals, and individuals do not know precisely in which ways. In recent years, the online-advertisement industry is a notable example. In particular, the Facebook-Cambridge Analytica data scandal, in which Cambridge Analytica harvested personal data from millions of Facebook users.[1] They used the data mainly for political advertising.

Consequently, a recent survey shows that only 49% of the US and UK population feels in control of their data online.[2] Chaum argues [9] already more than 30 years ago that the collection of personal data can lead to a dossier society, like the one described in Orwell's dystopian novel 1984 [30]. Such a society can result in that the privacy of individuals diminishes.

Already for many years, scientists and the SE-community have researched concepts to enhance the privacy of individuals. Jøsang and Pope [19] introduce the concept of "user-centric identity management". "User-centric" means that the individual becomes the central entity in the IMS. Consequently, individuals can control which parts of their identity

[1]See the coverage by the New York Times: https://www.nytimes.com/2018/03/17/us/politics/cambridge-analytica-trump-campaign.html?module=inline

[2]See following link for more information: https://blog.globalwebindex.com/chart-of-the-week/trust-data-privacy/)

they share with which party. This concept leads to a decentral architecture, in which identity providers (IdPs) and service providers (SPs) do not need to interact directly with each other. Hence, IdPs are not aware of the SPs the individual is visiting.

To be able to exchange information between the different parties, Chaum [9] introduces the notion of credentials. A credential contains a set of claims about a *subject*. In most cases, the individual is the *subject* and also the *holder* of the credential. Usually, an IdP, also called *issuer*, returns the credentials to the individual after the individual authenticated. Subsequently, the credential(s) are stored within a safe environment, which only the individual can access. Then, the individual can release parts of their credentials to any SP, also called *verifier*, to gain access to resources.

Recently, organizations have developed working products based on the idea of a user-centric IMS and its related concepts. As technology advances, it becomes easier and cheaper to develop such products. Two recent projects to mention here are Sovrin and IRMA. The non-profit Sovrin Foundation established Sovrin in 2016.[3] In this research, we focus primarily on the IRMA ("I Reveal My Attributes") project, which the non-profit Privacy by Design Foundation (PbDF) established in 2015.[4] The goal of IRMA is to make user-centric and privacy-friendly identity systems more practical, according to Alpar et al. [2]. The central component within IRMA is the IRMA mobile app. Within the app, individuals can store their credentials received from issuers. Then, via the mobile app, individuals can disclose parts of their credentials to verifiers. Both issuers and verifiers need to deploy an IRMA server to establish a session with any individual using the IRMA app.

However, according to Kim and Kankanhalli [20], individuals usually have resistance against new information systems implementations, referred to as the *status quo bias*. Additionally, if all user-centric systems operate independently, the resistance grows even more due to several reasons. First, individuals need to use for each system another safe environment to store their credentials from that system. Consequently, individuals are not able to receive and disclose credentials across systems. Second, credentials from different systems can contain the same sets of attributes. For instance, a citizen wants to use his "Basisregistratie Personen" attributes both with Sovrin and IRMA. Then, she needs both apps, which store the same attributes. This can be confusing for individuals. Third, participating organizations or institutions need to deploy different servers for each system. For example, if a verifier wants to accept both Sovrin and IRMA attributes, currently, it needs to deploy both a Sovrin and IRMA server. To sum up, several reasons may threaten the public acceptance of the concept of user-centric IMS and related products.

Therefore, the worldwide web consortium (W3C) established a working group. The working group had the goal to address the problem that there is no privacy-enhancing standard to enable individuals to express and transact claims. In 2019, they finished their work, and published a standard titled *Verifiable Credentials* (VCs) [47] as an official recommendation. A VC is defined as "a tamper-evident credential that has authorship that can be cryptographically verified". The standard consists of an unambiguous specification, which defines the syntax and semantics of VCs. By conforming to the specification, user-centric IMSs might extend their interoperability by being able to exchange messages with other systems that conform to the specification.

In this research, we explore the relation between the IRMA system and the VC standard.

[3]See `https://sovrin.org/`

[4]See `https://privacybydesign.foundation/en/`

We thereby make the following contributions:

- We identify similarities and differences between IRMA and VC by comparing and contrasting the two. Key concepts between the two overlap. We describe how to map VC entities to IRMA's data model.
- We analyze which requirements of VC are relevant, as not all requirements within the standard are mandatory.
- We describe four relevant real-world use-cases for IRMA when connecting with external actors. First, an external issuer issues credential to an IRMA user. Second, an IRMA user discloses credentials to an external verifier. Third, an IRMA issuer issues credentials to a non-IRMA user. And fourth, a non-IRMA user discloses credentials to an IRMA verifier.
- Based on the requirements and use-cases, we describe the necessary modifications that are needed within IRMA.
- We conduct an architectural impact analysis based on SACCS. We introduce the metadata server, which delivers scheme information to external actors. If IRMA wants to exchange messages with other VC-compliant systems, it can heavily impact IRMA's security properties.
- We build a prototype implementing a subset of the identified modifications. Subsequently, we prove that the prototype works as intended by exchanging VC-compliant messages between IRMA components.
- We describe goals and deliverables of related and this work.
- We discuss how VC extends IRMA's interoperability. By modifying IRMA to comply with the VC standard, IRMA at least reaches level three of the "levels of conceptual interoperability model".
- We list costs and benefits of making IRMA comply with the VC-standard.
- We specify activities for the PbDF to conduct when wanting to connect with another VC-compliant system.
- For the PbDF, we list ideas for future work. First, researching about decentralizing the IRMA scheme. Second, connect with another VC-compliant system. Third, study VC concepts currently not supported by IRMA. Fourth, study related standards or RFCs. And fifth, study non-technical challenges.
- For the W3C, we give ideas for improving their process and communication.

1.2. RESEARCH OBJECTIVE

As previously stated, the objective of this research is to explore the relation between the IRMA system and the Verifiable Credential standard.

This objective can be decomposed into several Research Questions (RQs), which in turn can be decomposed into several sub-questions:

RQ1 To what extent is it possible to align the data model and concepts of the IRMA system with the data model and concepts of the VC standard?

RQ1.1 What are the concepts and data model of the IRMA system?

RQ1.2 What are the concepts and data model of the VC standard?

RQ1.3 What are the similarities and differences in both IRMA and VC? By contrasting and comparing the two topics, we get a clear picture of how VC might improve IRMA.

RQ2 How to modify the IRMA system to be compliant with the VC standard and to be able to exchange messages with other VC-compliant systems?

RQ2.1 How can we identify relevant requirements? In the VC data model there are several classifications of requirements. We need to identify the relevant ones.

RQ2.2 Which relevant use-cases can be identified? By identifying use-cases, we can analyze further which distinct challenges exist when adapting IRMA to exchange messages with other systems.

RQ2.3 What are the effects on IRMA's architecture? In particular, we want to show what the impact is on security, as IRMA guarantees specific security properties.

RQ2.4 Which modifications can be implemented in a prototype? We can describe which modifications fit within the scope of this research.

RQ2.5 How to validate that those modifications work? After applying modifications, we want to show that those work as intended.

RQ3 How do the modifications help to extend IRMA's interoperability? As an essential goal of VC is to provide us with an interoperable standard, we want to exhibit how the modifications help to extend the interoperability of IRMA.

RQ4 Which recommendations can be provided to the IRMA project based on the experiences of this research? Recommendations can support IRMA stakeholders for making decisions with regards to VC and interoperability.

RQ5 Which recommendations can be provided to the W3C based on the experiences of this project? As we had to analyze the VC standard in detail, possibly we can provide the W3C with recommendations on how to improve.

In chapter 2, *research process*, we explain how we set up the process to get answers to the different questions.

1.3. THESIS OVERVIEW

The paper is organized as follows. In chapter 2, I explain the research process, which I applied to conduct the research. In the third chapter, I give the interested reader more background information on relevant topics, in particular identity management, linked data, and interoperability of software systems. In chapter 4, I describe in detail both IRMA and VC.

Subsequently, I compare and contrast the two. In chapter 5, I identify requirements, modifications, and analyze the impact of the modifications on IRMA's architecture. In chapter 6, I describe the developed prototype and how I validated it. In chapter 7, I list related work and put it into context with this research. In chapter 8, I discuss the findings and limitations of the thesis. Additionally, I provide recommendations for both the PbDF and W3C. In chapter 9, I draw conclusions based on the identified relation of IRMA and VC. In appendix F I evaluate my personal experience carrying out this research.

2 Research Process

Within empirical research, usually, data is collected. Subsequently, researchers use this data to answer knowledge questions; Wieringa [39] calls it the “empirical cycle”. However, within this research, we want to explore a relation between two topics related to software. This can result in some software design and, possibly, some form of working software. Collecting data can help us to explore this relation, but it does not play a central role as it does within typical empirical research. Hence, for our research, it would be beneficial to find a research method that provides us with tools to support our research and its process.

According to Wieringa [39], “design science is the design and investigation of artifacts in context”. In this research, indeed, we want to create one or more artifacts, such as a software design or working software. It can support us in understanding the relationship between IRMA and VC, and improve IRMA. Based on design science, Peffers et al. [31] developed a process model, called *Design Science Research Methodology* (DSRM), providing the IT research community with a process framework. The DSRM process consists of six activities. First, the *problem identification and motivation* activity defines the problem and argues why the proposed solution will be of value. Second, within the *defining the objectives for a solution* activity, as the name already suggests, the objectives are defined for a possible solution. Third, in the *design and development* activity, the actual artifact is created, thereby figuring out the desired functionality and architecture. In the fourth activity, *demonstration*, it needs to be demonstrated that the created artifact contributes to solving the stated problem, which can involve experimentation or simulations. Fifth, in the *evaluation* activity, researchers need to observe or measure how well the artifacts supports a solution to the problem. It can take various forms, such as a comparison of the artifact's functionality with the objective stated in the second activity, or conducting interviews. Within this activity, researchers can decide to jump back to previous activities to try to improve the effectiveness of the artifact. In the last activity, *communication*, researchers communicate the results to relevant audiences.

Based on the process model proposed by Peffers et al. [31], we develop a research process, as shown in 2.1. In the figure, we summarize for each activity, how we approach it, and which research questions (RQ) we aim to answer with it. In the following paragraphs, we describe the activities in more detail.

While my supervisor Greg Alpar provided us with the topic, we still have to discover which problem we want to solve with this research. Consequently, we take a problem-centered initiation. Therefore, in the first activity *identify problem and motivate*, we first

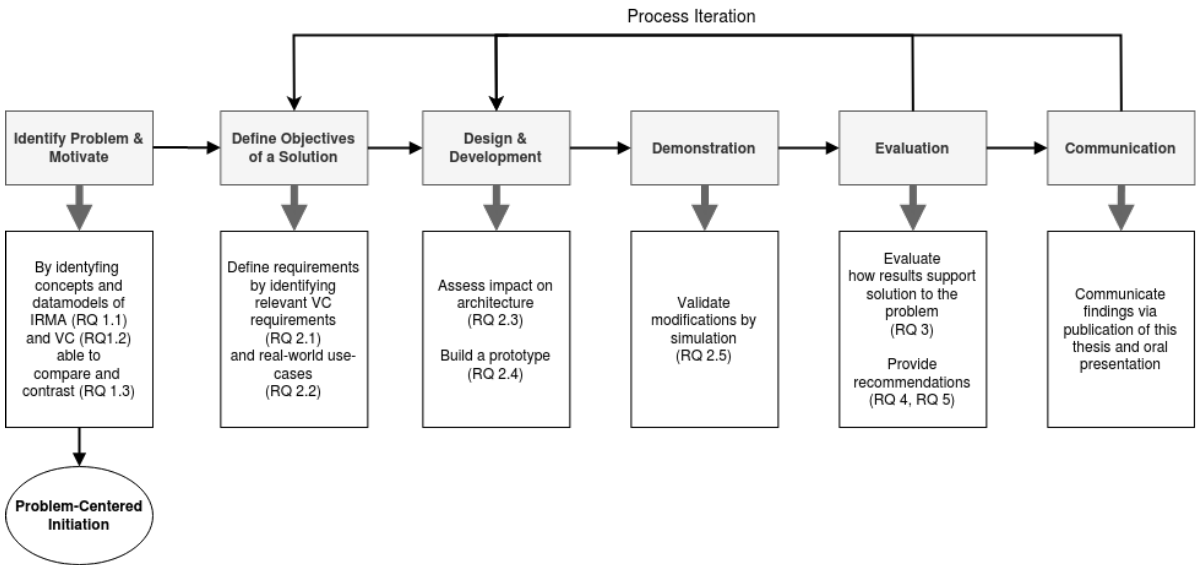

Process Iteration
Identify Problem & Motivate
Define Objectives of a Solution
Design & Development
Demonstration
Evaluation
Communication
By identyfing concepts and datamodels of IRMA (RQ 1.1) and VC (RQ1.2) able to compare and contrast (RQ 1.3)
Define requirements by identifying relevant VC requirements (RQ 2.1) and real-world use-cases (RQ 2.2)
Assess impact on architecture (RQ 2.3) Build a prototype (RQ 2.4)
Validate modifications by simulation (RQ 2.5)
Evaluate how results support solution to the problem (RQ 3) Provide recommendations (RQ 4, RQ 5)
Communicate findings via publication of this thesis and oral presentation
Problem-Centered Initiation

Figure 2.1: Applied DSRM process

need to understand both topics IRMA and VC in greater detail by studying previous work and reading online documentation of both. This information enables us to compare and contrast the two to identify which problems or challenges VC can help to solve within IRMA.

Then, in the following activity, *define objectives of a solution*, we identify the relevant requirements of VC. Also, we investigate which actual real-world use-cases we want to solve by modifying IRMA.

In the third activity, *design and development*, we first conduct an architectural impact analysis. We focus on security aspects, as those are advertised explicitly on IRMA's website. Then, we build a prototype, which implements a subset of the identified modifications.

In the fourth activity, *demonstration*, we demonstrate that the created prototype works as described. We are not able to establish active collaboration with other VC projects or the VC community itself during the time of conducting the research. Hence, we set up IRMA in such a way that the different IRMA components exchange VC-compliant messages with each other.

The *evaluation* activity consists of two steps. First, we evaluate how the results of the previous activities support a solution to the identified problem, that is, extend IRMA's interoperability. We also take related work into account. Second, we provide recommendations to both the PbDF and the W3C.

The *communication* activity involves publishing this master thesis, and after finalizing this thesis, defending the results in an oral form of presentation.

Lastly, based on the *evaluation*, we can jump back to earlier activities, in particular, refining the objectives. Feedback on the final draft version of the thesis and feedback on practice presentations help us to improve this work further.

To summarize, the DSRM process helps us to structure our research process more advantageously than traditional empirical research. The creation of an artifact stays central in our research rather than the collection of data to answer knowledge questions. The provided artifact(s) can help the IRMA project to evaluate if it is feasible to make IRMA compliant with the VC standard in practice.

3 PRELIMINARY

In this chapter, we give the interested reader more information on three topics used later in this research. We start by describing briefly what identity management (IdM) is and how it has evolved over the years. Also, we briefly describe trust relationships between different actors within IdM. Then, we introduce technologies used in practice, in particular attribute-based credentials (ABCs), blockchain, and decentralized identifiers (DID). Additionally, we introduce three current user-centric IMSs projects, Sovrin, uPort, and Verifiable Credentials Ltd. As linked data plays a vital role within VC, we introduce it shortly. We introduce the concept of interoperability as one essential goal of software standards such as VC is to extend interoperability between software systems. We show what its benefits are, and we describe a model that can help to determine the degree of interoperability of software systems.

3.1. IDENTITY MANAGEMENT

Identity management (IdM) is the management of identities of individuals. IdM is also known as identity and access management (IAM). The goal of identity management is to ensure that people with proper identity are authorized to access adequate resources. Pfitzmann et al. [33] define identity as follows: "An identity is any subset of attribute values of a person, which sufficiently identifies this person within any set of persons. So usually there is no such thing as 'the identity', but several of them". As shown in figure 3.1, every individual in the world has specific attributes linked to them, inherent to that individual, for example, her birth date or nationality. SPs request a subset of those attributes to identify an individual. Each identity is identified via an identifier, which can be personally-identifying, such as your name, or not personally-identifying, such as a unique computed id. Identifiers enable a system to distinguish that identity from all other identities within that system.

According to Dhamdhere and Karande [14], we can classify components of IdM into four major categories: authentication, authorization, user management, and central user repository. The authentication category consists of authentication and session management. It means, that after a user is authenticated by, for example, either providing username and password combination or disclosing attributes, a system creates a session. The system uses the session during interactions between the user and the system. The authorization module determines whether a user is granted access to a resource by checking the user's identity against authorization policies. The user management category consists

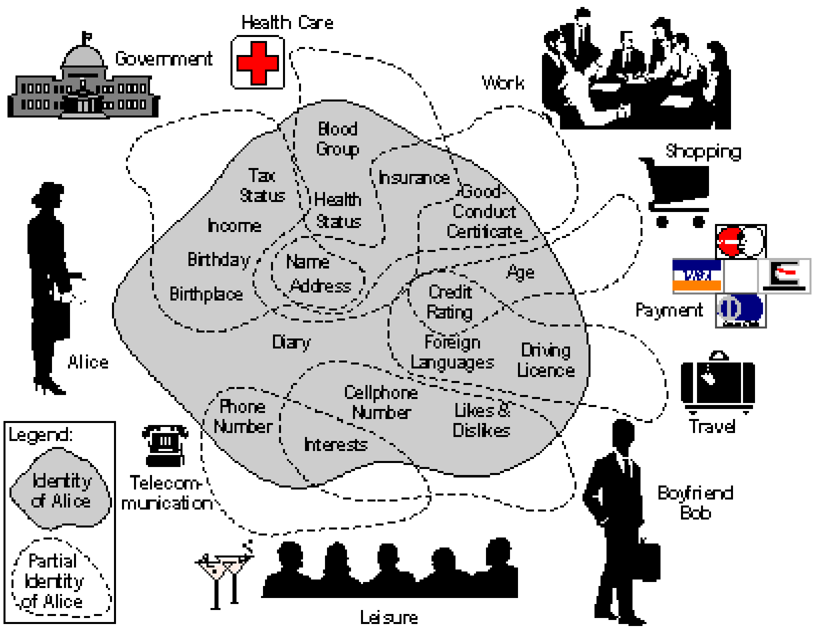


Figure 3.1: Alice's partial identities (from Clauß and Köhntopp [12])

of the set of administrative functions needed to manage the user, for example, the registration process, or password, and role/group management. Another critical concept is self-service, whereby the users take care that their identity data is accurate. Commonly, the central user repository stores identity information and shares it with other systems or services if needed. The most well-known technologies to mention here are LDAP, an open protocol, or Active Directory (AD) from Microsoft, a directory service.

3.1.1. EVOLUTION

Allen [43] describes four phases that IdM has gone through since the existence of the internet, thereby addressing the issue with central authorities.

In the first phase, *centralized identity*, one central authority is responsible for dealing with identities. The holder of the claim is always dependent on this authority, whether some claim about the holder's identity is valid or not. Allen gives the example of certificate authorities (CAs). CAs are responsible for handing out certificates to internet sites. Users trust the CAs to hand out only valid certificates. One big problem with centralized authorities is that users are bound to that one authority, and this authority can deny users identity or confirm a false identity. Consequently, this central authority has all the power.

The second phase, *federated identity*, was established around the year 2000. Microsoft's *Passport* initiative enabled users to use the same identity to authenticate and authorize on different sites. Also, companies or institutions frequently use Single Sign-On (SSO) modules. With SSO, users do not need to log in again when wanting to access another system within the same organization or domain. The problem, however, is still that a central authority manages the identity of the user.

In the third phase, *user-centric identity*, as the term suggests, the idea is to make the users the central actor in the identity process, as already introduced in the introduction. Allen claims that powerful companies made it difficult for researchers and implementers to achieve that goal. And what's left is the idea that a user only authenticates to a credential provider, and this provider manages all identities of the user; Facebooks popular Connect service is an example.

In phase number four, self-sovereign identity (SSI), the centralized nature of the previous stages shall diminish. Users shall completely own their data by storing it locally on, for instance, a mobile device. Then, users can decide which parties get access to parts of their identity.

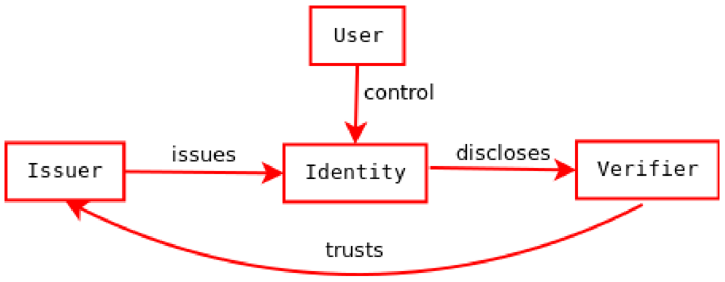


Figure 3.2: SSI actors (inspired by Mühle et al. [27])

Mühle et al. [27] identify the relation between the different actors in an SSI system, as depicted in figure 3.2. As previously mentioned, an issuer issues signed claims to users. A lot of current SSI software systems provide wallet applications to users, which typically are installed on mobile phones to let users store their claims locally. Any relying party, a so-called verifier, requests specific attributes from the user based on authorization policies for the service it offers. Then, the user discloses those attributes after approving the request. On the trust relation, we elaborate further in subsection 3.1.3.

We want to note that the definition of SSI is still under discussion within the industry and research community. Recent research show that the term *user-centric* is still used, for instance in [21] or [42]. Hence, in what follows, we choose to use *user-centric*, because, as the name already suggests, the user stays central.

3.1.2. CREDENTIALS

The fundament of any *user-centric* system are credentials. A credential is a cryptographic container, containing a set of claims of a subject. An issuer cryptographically signs the claims by attaching a signature to the credential. The signature provides relying parties confidence about the validity of the claims made. Commonly, credentials also contain metadata, such as who the actual issuer is, and the validity period of the credential. Chaum [9] extends the concept with the addition of *anonymous credentials*. By utilizing pseudonyms, individuals can make sure that when disclosing a credential, that the individual is "anonymous". This makes it harder for other parties or attackers to identify the individual when only looking at the content of a credential. This concept is also referred to as *unlinkability*. Pfitzmann et al. [33] define unlinkability as an attacker is not sufficiently able to distinguish if messages can be linked to each other to identify the user who sends those messages. Users can increase unlinkability by using different pseudonyms for each transaction. A pseudonym is an identifier, which does not reveal the real identity of an individual.

According to Alpar and Jacobs [3], in recent years, the notion of anonymous credentials is renamed to attribute-based credentials (ABCs). One of the optional features of an ABC is that several attributes can be shown independently of one another. A technique to realize this feature is called Zero-Knowledge Proof (ZKP), described first by Golderwasser et al. [18]. With ZKPs, someone (a prover) can convince someone else (a verifier) that she knows a secret without revealing the secret itself. For example, imagine, a club wants that only people older than 18 years, can enter. With ZKP, the prover only needs to reveal to the bouncer the fact that she is older than 18 years old, without having to reveal her actual birth date, or any other information that is on her ID.[1] Hence, the concept of ZKP is privacy-enhancing. Camenisch and Lysyanskaya [8] developed a digital signature scheme to enable implementers to employ ZKPs in software systems. The scheme became part of IBM's Idemix specification [46].

3.1.3. TRUST RELATIONSHIPS

In figure 3.2, one specific trust relationship exists; in particular, the verifier needs to trust the issuer that the issuer issued valid credentials to the user. Otherwise, a verifier can not trust any credentials disclosed by a user. In the end, this is a social problem and still subject to academic research. However, in the architecture document of ABC4Trust by Bichsel et. al [6], the authors try to list all the trust relationships typically needed in an ABC-system. All parties involved need to trust each other sufficiently to make a user-centric IMS work. In the following, we list only relationships relevant for our research; that are relationships between users, issuers, and verifiers.

- All parties involved have certain trust assumptions in common. First, they need to put trust in the correctness of the underlying cryptographic protocols. Second, into the trustworthiness of the implemented software system.
- From a user's perspective, the user trusts the issuer that it delivers correct credentials. Also, issuers design credentials in such a way to avoid the introduction of privacy risks. And that issuers do not block the use of credentials without a valid reason.
- Users need to trust verifiers that they do not misbehave, by, for example, trying to create a dossier of the user. However, institutions can enforce regulations to make this harder for verifiers.
- As stated above, verifiers need to trust issuers to issue only valid and correct credentials.
- Also, verifiers need to trust users not to share credentials with others if this is not permitted. Possibly, the sharing of credentials can even be made impossible due to the chosen technology.

3.1.4. BLOCKCHAIN AND DID

For registering identifiers and related information, according to Mühle et al. [27], a blockchain plays a crucial role in the architecture of an SSI system, thereby replacing the registration authority as known in centralized identity management systems. Satoshi Nakamoto [28] invented the blockchain technology. It is the underlying technology used by the crypto-currency Bitcoin. A blockchain is a chain of blocks. More specifically, it is an expanding list of records linked together by using specific cryptography. The cryptography makes it hard for attackers to modify content stored on a blockchain. Usually, a peer-to-peer network

[1]Quisquater et al. [34] explain ZKP in a more extensive example

manages a blockchain. Each peer needs to follow a specific protocol to add a block. This concept, according to Christidis and Devetsikiotis [11], enables trustless networks. Trustless means that parties can trust the integrity of the data of others even though they do not trust other parties.

According to the W3C, it is impossible to establish a complete decentral identity system with traditional centralized authorities, as each of those authorities serves as its root of trust. Hence, the W3C is currently developing a specification, called Decentralized Identifiers (DID), standardizing the usage of decentralized digital identities.[2] While DIDs do not require blockchain technology, they are often associated with it nowadays. We can think of a DID infrastructure as a globally available key-value database. The key is the DID itself, which is a string consisting of several parts. The value refers to a DID document, which a blockchain usually stores. A DID document contains identifying information about a party, e.g., a public key required in cryptographic protocols. Then, any relying party can use this publicly available information.

All in all, DIDs based on blockchain technology can play an essential role in truly decentralizing identity management.

3.1.5. USER-CENTRIC IMS PROJECTS

In the following two subsections, we introduce three recent user-centric IMS projects, in particular Sovrin, uPort, and Verifiable Credentials Ltd.

Sovrin The Sovrin Foundation is a private-sector, non-profit organization that established the Sovrin project. The project aims to provide a solution based on SSI principles. They claim that they employ the first public permissioned blockchain, which differentiates it from other SSI systems.[3] Permissioned blockchains can run at much higher throughput than permissionless blockchains, while still achieving a high level of trust. With permissionless blockchains, every user can add entries to the blockchain. With permissioned blockchains, only users with permission can add entries. Consequently, within Sovrin, the *Board of Trustees* ultimately governs Sovrin, which approves the policies to select stewards. Stewards are trusted institutions that operate the nodes of the blockchain.

The software stack of Sovrin consists of three layers: the blockchain, agents, and clients. Anyone can develop agents and clients, and hence are open for the competitive market. Both agents and clients act as a client to the blockchain. The difference between an agent and a client is that the agent operates via a P2P network layered over the blockchain. Hence the agent is also acting as a server, addressable via a network endpoint.

uPort uPort is a platform for SSI, built on the Ethereum blockchain. It consists of three main components: smart contracts, developer libraries, and a mobile app. Christidis and Devetsikiotis explain in [11] that smart contracts in the context of a blockchain are scripts stored on the blockchain. They are only executed by addressing a transaction towards it. uPort developed a smart contract architecture that enables uPort to attach attributes to identifiers and selectively disclose attributes. The mobile app holds the user's claims, and consequently, it is possible to disclose those claims to other SPs. The developer libraries

[2]See the DID specification here: https://w3c.github.io/did-core/

[3]The official whitepaper about Sovrin: https://sovrin.org/wp-content/uploads/2017/04/The-Technical-Foundations-of-Sovrin.pdf.

enable third-party app developers to integrate uPort support in their apps. After the first implementation, the project integrated the DID and VC standard based on JSON Web Tokens.[4]. Interestingly, uPort provides libraries for an HTTPS DID method. It means that it stores the DID document not on a blockchain, but under a web-resource reachable via the HTTP protocol. Consequently, by applying the HTTPS method, a DID has the following format: “did:https:example.com”. However, a centralized authority needs to host the DID documents, thereby undermining the feature of trustless networks.

Verifiable Credentials Ltd David Chadwick is CEO of the company “Verifiable Credentials Ltd”, which he established in 2019. He is an expert in the IdM field and is co-author of the VC standard. He claims that by combining the VC and “Web Authentication protocol” standard, they create a secure and privacy-preserving IMS infrastructure. The “Web Authentication protocol” defines an API to enable the creation and sharing of credentials by applications to authenticate users.[5] Contrary to the other two projects, they do not make use of blockchain technology.

3.2. LINKED DATA

The VC specification refers extensively to the concept of *Linked Data*. Therefore, we explain it here in more detail, together with related ideas. Everybody familiar with the web knows how to browse from one website to another by using hyperlinks; websites are just documents made for humans to consume. Also, humans can understand the meaning of text. The underlying question is how we can enable computer systems to find, link, and understand data semantically. This is where Linked Data comes in as it provides a set of principles for generating semantic data. Tim-Berners Lee, the founder of the world-wide-web, was one of the first persons talking about the notion of linked data and introducing those principles listed under https://www.w3.org/DesignIssues/LinkedData.html. He stated that “the Semantic Web isn’t just about putting data on the web. It is about making links so that a person or machine can explore the web of data. With linked data, when you have some of it, you can find other, related, data”.

A standard format used for linked data is RDF, the Resource Description Framework.[6] The core structure of the abstract syntax is a set of triples. Each triple called an RDF graph consists of a subject, a predicate, and an object. For computer systems, it is essential to have an ontology for predicates to “understand” the relationship between a subject and an object. Ezike [15] introduced such an ontology, downloadable under http://dig.csail.mit.edu/2018/svc, which he developed for SolidVC.

JSON-LD [48] is a JSON-based format to serialize linked data with multiple intentions. First, to build interoperable web services. Second, to store linked data in JSON-based storage engines. Third, use linked data in programming environments.[7] One goal of JSON-LD is to require as little effort as possible from developers to transform their existing JSON to

[4]See also uPort Credentials - create verification example: https://developer.uport.me/credentials/createverification

[5]See https://www.w3.org/TR/webauthn/

[6]See https://www.w3.org/TR/rdf11-concepts/ for concepts and abstract syntax

[7]Google announced in 2015 that it recognizes JSON-LD as a structured data type to have the ability to understand better the content of the data it parses for their search engine. See https://webmasters.googleblog.com/2015/03/easier-website-development-with-web.html for more information.

JSON-LD.[8] In figure 1 we use an example of a valid JSON-LD message that describes a *Person*.[9] The *@context* is *http://www.schema.org*, resulting in that all keys and @type values are defined under *http://www.schema.org*, for example *alternateName* can be looked up via `https://schema.org/alternateName.jsonld`.

To sum up, linked data and the underlying principles enable computer systems to query and understand data on a semantic level. With JSON-LD, a JSON-based format was developed supporting linked data.

Listing 1 JSON-LD person

```
{
"@context": "http://www.schema.org",
"@type": "Person",
"@id": "https://jay.holtslander.ca/#person",
"name": "Jay Holtslander",
"alternateName": "Jason Holtslander",
"nationality": "Canadian",
"birthPlace" : {
    "@type": "Place",
    "address": {
          "@type": "PostalAddress",
        "addressLocality": "Vancouver",
        "addressRegion": "BC",
        "addressCountry": "Canada"
    }
},
"gender": "Male",
"Description": "Technologist",
"disambiguatingDescription": "Co-founder of CodeCore Bootcamp",
"jobTitle": "Technical Director"
}
```

3.3. INTEROPERABILITY

Within *ISO/IEC 2382-01*, interoperability is defined as follows: “The capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units.”[10] Phones are a good example of successful interoperability. No matter which company produced the phone, and no matter which company set up and maintains the cellular

[8]See design goals of JSON-LD: `https://json-ld.org/spec/latest/json-ld/#design-goals-and-rationale`

[9]Partly copied from `https://github.com/JayHoltslander/Structured-Data-JSON-LD/blob/master/Person.json`

[10]The PDF version is available via archive.org: `https://web.archive.org/web/20071129034634/http://old.jtc1sc36.org/doc/36N0646.pdf`

network, and no matter if the call is placed wirelessly or via landline, from any phone, you can call any other phone.

There are several benefits of interoperability, and we describe two important in what follows. First, according to Choi and Whinston [10], it plays an essential role in lowering costs and prices in the economy. It plays an even more crucial role in our digital economy as we already use there different interoperable parts, such as in hardware or the world wide web. In contrast, according to research from the NIST [17], inadequate interoperability in the US capital facilities industry leads to costs of around 15.8 billion dollars annually. Second, Opara-Martins and Feng Tian [29] argue that interoperability helps to tackle the problem of vendor lock-in. Vendor lock-in means that once a company switched to a particular vendor, it is difficult to change to another vendor due to the use of proprietary software. In particular, organizations fear the movement to a cloud-computing-based infrastructure due to the risk of vendor lock-in.

Organizations can reach a certain degree of interoperability by defining and providing standards for IT systems, thereby increasing the capability of interoperation between systems. However, as Lewis et al. [23] state, expectations of what can be achieved by a standard are often unrealistic. In particular, expectations are that after systems comply with specific standards, that it guarantees seamless interoperability. But it does not take into account organizational and cultural context. Hence, organizations need to be aware of such limitations.

According to Brownsword et al. [7], achieving and maintaining interoperability between systems is difficult due to several reasons. First, it is difficult due to the complexity of the systems and their potential interactions with other systems. Then, the lack of conformity between participating organizations. The third reason is the evolvement of the systems itself. And finally, due to the lack of visibility of all the details within and between the interoperating systems. Hence, it can be useful to have some model or framework available that enables us to categorize or measure the level of interoperability. Consequently, it might enable us to reason how to increase the capability of interoperation.

Tolk and Muguira [37], amongst others, developed a model called “the Levels of Conceptual Interoperability” (LCIM). The need for this model emerged as, according to Tolk and Muguira existing models were not sufficient, and “establishment of metadata standards allows a much more open use of data within the systems, as not the data itself has to be standardized, but the interpretation of the data in the given context”. The model was revised several times until the community agreed upon a version with seven hierarchical levels [38], see figure 3.3.

Each new level in the model is targeting a different perspective for the exchange of data. In level zero, systems are not interoperable. On the first level, technical interoperability unambiguously protocols for communication exists used by infrastructure. In level two, syntactic interoperability, a standard structure to exchange information exists. In level three, semantic interoperability, systems share the meaning of data. On the fourth level, pragmatic interoperability, systems know each other’s methods and procedures, hence the context is understood. In level five, dynamic interoperability, after exchanging data, the effects of processing the information is understood by participating systems. On the last level, conceptual interoperability, all involved concepts need to be fully documented and publicly available in a way that other engineers can understand them; and the model needs to be implementation independent.

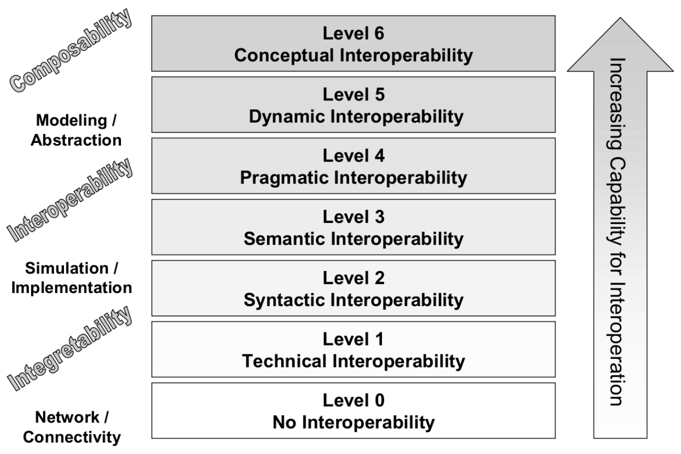

Composability
Modeling / Abstraction
Interoperability
Simulation / Implementation
Integretability
Network / Connectivity
Level 6 Conceptual Interoperability
Level 5 Dynamic Interoperability
Level 4 Pragmatic Interoperability
Level 3 Semantic Interoperability
Level 2 Syntactic Interoperability
Level 1 Technical Interoperability
Level 0 No Interoperability
Increasing Capability for Interoperation

Figure 3.3: LCIM levels (copy from Wang et al. [38])

To conclude, interoperability is a characteristic of a system to make it easier to exchange information with other systems. Interoperability has several advantages, such as lowering costs or tackling the issue of vendor lock-in. Standards can help to make systems more interoperable. However, often expectations of standards are too high. Additionally, several reasons make it difficult to achieve and maintain interoperability. Researchers provide us with models such as LCIM to measure the capability of interoperation. It facilitates discussions between parties on how to increase this capability.

4

IRMA, VC AND THEIR RELATION

In this chapter, we introduce the two topics that are part of the title of this thesis in greater detail, “IRMA” and “Verifiable Credential”. In section 4.1, we outline the history of the IRMA project and what its mission is. Then, we define the roles, architecture, data model, and concepts. In section 4.2, we elaborate on the motivation and goals of the VC working group, and the use-cases they foresee. Also, we zoom into the VC specification, describing the roles available, data model, and related concepts. At the end of this chapter, we compare and contrast the two concepts to identify similarities and differences.

We obtain technical documentation of IRMA from the official documentation: `https://irma.app/docs/`.

The VC data model is obtainable via `https://www.w3.org/TR/vc-data-model/`.

4.1. IRMA

In 2016 the Digital Security research group of the Radboud University and the Privacy and Identity Lab founded the Privacy by Design Foundation (PbDF). The mission of the foundation is to promote the development use of open, secure, and privacy-friendly technologies. They established the IRMA project. Within IRMA, the mobile app stays central. By using the app, the user has full control of her identity. Credentials are only stored locally on the user’s phone, and no one but them has access to it. In every disclosing session, a user authenticates with another pseudonym. This results in the unlinkability of messages and, consequently, makes it harder for parties to identify individuals. However, in many use-cases, individuals use personal identifying attributes such as the BSN or email address. Those attributes make individuals identifiable.

If we look at the four different categories in which we can classify IdM, we can argue that IRMA falls into three of those categories. *Authentication* and *authorization*, as organizations can employ IRMA to let users gain access to a resource by checking disclosed attributes against policies. And also, *user management*, as issuers calculate the validity period of credentials, and users have control of which attributes they store about themselves within the IRMA app.

4.1.1. USE-CASES

In the following list some of the use-cases are described briefly that the IRMA project currently realizes:

- The IRMA organization helps to realize age verification in liquor stores in the city of Almere.[1]
- nuts.nl is a consortium consisting of several e-health companies. The consortium tries to join forces to set up an open standard for identifying and authenticating individuals with the help of IRMA.
- Also, IRMA sees itself as a technology that can help against fake news and deepfakes by using digital signatures according to Jacobs.[2]

4.1.2. Roles

The following roles participate in the IRMA ecosystem:

User The user using the IRMA app, acts as the client in the IRMA protocol;

Verifier or service provider A party offering a service to users, and usually wants to verify a user's attributes for identifying purposes;

Identity provider or issuer A party wanting to issue attributes to a user;

Issuer Issues credentials to a client when instructed by an identity provider using its private key;

Requestor A party that either wants to verify a users attributes or issues attributes to a user;

Scheme manager Maintaining one or more schemes. Via scheme distributing issuer information, such as public keys, and credential types. Decides which issuers may join a scheme.

4.1.3. Architecture

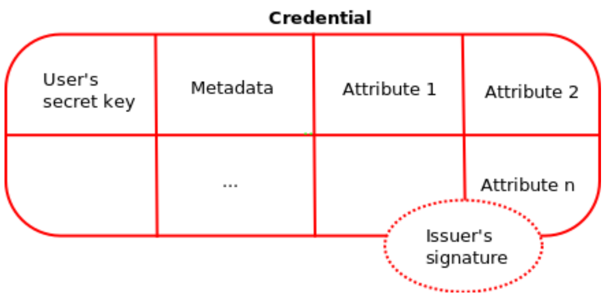


Figure 4.1: IRMA credential (inspired by Alpar and Jacobs [3])

In figure 4.1 we see an IRMA credential. It contains the attributes of the user, the issuer's signature, the metadata attribute, and the user's secret key. The first attribute of any IRMA credential is always the user's secret key. The IRMA-app generates the secret key when it runs for the first time. However, IRMA never discloses the key. Still, the key is of relevance in the cryptographic protocol. It proves that the credentials belong to the same user if the user disclosed attributes from multiple credentials. The issuer's signature is essential within any user-centric system. It signs the attributes proving that those attributes are indeed from

[1]See a tweet from the IRMA project: https://twitter.com/IRMA_privacy/status/1111384252761796608.

[2]See the IRMA Manifest by Jacobs: https://privacybydesign.foundation/pdf/IRMA-manifest-2019.pdf, in Dutch.

that issuer and valid. The metadata attribute contains information about which credential type this credential is an instance of, the issuing date of the credential, and the expiry date of the credential. The credential type determines which issuer signs the credential, its validity period, and names of the contained attributes.[3] Credential types and other information are stored within a scheme. IRMA uses centrally maintained schemes, whereby the schemes are updated periodically by the different components, to tell all parties which credential types are valid, and which issuer may issue them

IRMAs cryptographic protocol is partly implementing IBM's Idemix specification. In particular, IRMA uses ZKPs for two reasons. First, to hide attributes in a credential that a user does not want to disclose, referred to as "selective disclosure". Second, to prove knowledge of the secret key to the issuer without actually disclosing it.[4]

A public-private-key-pair is needed to enable a verifier to verify a user's credentials. The private key is used by the issuer to sign a credential. The public key is used by a verifier to verify that the disclosed attributes are signed using the corresponding private key.

In IRMA, three distinct session types exist. First, in an issuance session, the user wants to obtain claims from an issuer to be loaded in the IRMA app. Subsequently, users can use those claims in the other session types. Second, in a disclosure session, a verifier asks the user to disclose attributes. It depends on the use-case what kind of attributes, and the user needs to confirm the disclosure. Third, similar to a disclosure session, in a signing session, the attributes are attached to a message. However, in this research, signing sessions are left-out-of-scope as it is a variant of a disclosing session. In appendix B we describe the issuance and disclosure processes of IRMA in detail.

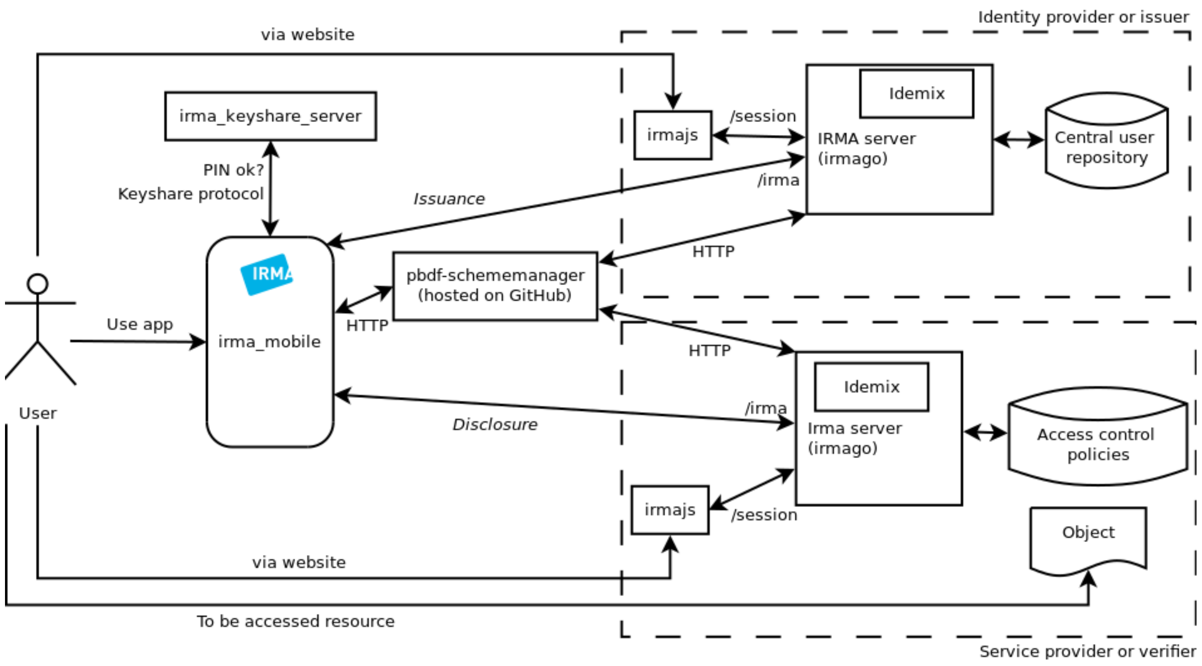


Figure 4.2: IRMA system overview

In figure 4.2, we show how the different components in an issuance and disclosure ses-

[3]See for an example https://github.com/privacybydesign/pbdf-schememanager/blob/master/pbdf/Issues/irmatube/description.xml.

[4]More information about how IRMA implements ZKP, see https://irma.app/docs/zkp/.

sion interact with each other. The user needs to use the IRMA app to receive and store attributes and, subsequently, use them in a disclosing session. IRMA uses the keyshare server to verify the PIN, and in the keyshare protocol. The issuer or verifier embeds the “irmajs” library into its website to provide the user with the QR code. Initially, the party in the role of requestor needs to create a session via the */session* endpoint on the IRMA server. Based on the session information, the website computes the QR code. After the user scans the QR code or using a deep-link to open the IRMA app, the connection between the app and the IRMA server is established. Then, the app uses */irma* endpoint to exchange messages with the IRMA server. In an issuance session, the issuer needs to employ a central user repository, where the attributes of a user are stored to be able to pass the correct attributes to the authenticated user. In a disclosing session, the verifier needs to check the disclosed attributes against its access control policies to either grant or deny the user access to the desired resource. Periodically, the servers and the mobile app update the used schemes by contacting the scheme manager, who maintains the scheme.

In the following paragraphs, we explain the different components in more detail.

irma_mobile Users need to install the IRMA app on her mobile device, which users can download from the official Android and iOS app stores. irma_mobile depends on *irmago* as the business logic is handled entirely within *irmago*. By implementing the data model and supporting code within *irmago*, the IRMA project only needs to maintain those in one central place.

irma_keyshare_server The keyshare server has multiple responsibilities. First, it can validate the PIN entered by the user. Only then, the KSS allows the session to continue. Second, it can block a user for a certain amount of time if the user enters the PIN wrong too many times. Fourth, for users, it is also possible to block their account via the KSS website. One disadvantage of the KSS is that if it is not available, no one using the scheme linked to the KSS can use that scheme’s credentials anymore. Within an IRMA scheme, the use of the keyshare server can be enabled or disabled.

irmago Irmago is an IRMA implementation in Golang. It contains the data model, business logic to handle requests, generate, and store credentials. Also, it provides the command-line tool *irma*, which amongst others, can start an IRMA server instance. When running an IRMA server, two endpoints are exposed: */irma*, that is used by the IRMA app during IRMA sessions. And */sessions*, that is used by requestors, enabling them to initiate a session, monitor them, and retrieve session results.

irmajs The irmajs component is essentially a JavaScript client that consumes the RESTful JSON API from the *irmago* component. When interacting with either the issuer or verifier, the user browses to a website from that party. Then, the irmajs component requests the session from the IRMA server. Subsequently, it generates the QR-code or deep-link to communicate the session to the client. Consequently, the client can directly communicate with the IRMA server via the */irma* endpoint.

pbdf-schememanager The responsibility of the scheme manager is to maintain the information of one ore more IRMA schemes and distribute it to relevant parties. Anyone

```
SchemeName
+-- IssuerName
|   +-- Issues
|   |   +-- CredentialTypeName
|   |       +--- description.xml
|   |       +--- logo.png
|   +-- PublicKeys
|   |   +-- 0.xml
|   |   +-- 1.xml
|   +-- PrivateKeys (need not be present)
|   |   +-- 0.xml
|   |   +-- 1.xml
|   +-- description.xml
|   +-- logo.png
+-- description.xml
+-- index
+-- index.sig
+-- timestamp
+-- pk.pem
+-- kss-0.pem
```

Figure 4.3: IRMA scheme structure (copy from https://irma.app/docs/schemes/)

can start a new scheme, and consequently, become the scheme manager for that scheme. The PbDF maintains its scheme called “pbdf-schememanager”, which the IRMA app uses by default. The “pbdf-schememanager” scheme is hosted on GitHub. [5] Organizations can configure the app and IRMA servers to use different schemes.

A scheme contains essential information such as which parties issue credentials, their public keys, and which credential types to use. From a trust angle, this means that there is always a central party. The party is responsible for maintaining the scheme, for instance, to update issuer information or to add new credential types. Also, scheme managers often delegate the hosting to a third party, for instance, GitHub, in case of the “pbdf-schememanager”.

Each scheme is signed within IRMA. In the file *index* IRMA stores all hashes of all files within a scheme. Then, IRMA stores a signature of the *index* file in *index.sig*. This enables any party to verify that no one tampered with the scheme.

A scheme is set up as a directory structure, as visible in figure 4.3. Within a scheme, each issuer gets a directory (IssuerName). Within the “Issues” subdirectory, the scheme contains credential types that the issuer can issue; listing 2 shows partly the official Facebook type belonging to issuer “pbdf”. As a consequence, it means that issuers can not share types, and one issuer owns at least one credential type. Within the “PublicKeys” subdirectory, the public keys are stored, which the verifier uses in a disclosing session to verify that the disclosed credential is valid. As certificates expire, issuers need to renew their certificates periodically. A new public key does not overwrite the old one. This enables users to disclose credentials signed with an expired certificate. Within *irmago*, a so-called “public key counter” is used to refer to the different public keys of an issuer.

[5] See https://github.com/privacybydesign/pbdf-schememanager

Listing 2 IRMA Facebook Credential Type

```
<IssueSpecification version="4">
    <Name>
        <en>Facebook</en>
    </Name>
    <ShortName>
        <en>Facebook</en>
    </ShortName>
    <SchemeManager>pbdf</SchemeManager>
    <IssuerID>pbdf</IssuerID>
    <CredentialID>facebook</CredentialID>
    <Description>
        <en>Facebook attributes obtained using Facebook...</en>
    </Description>
    <ShouldBeSingleton>false</ShouldBeSingleton>
    <IssueURL>
        <en>https://pbdf/issuance/social/facebook/</en>
    </IssueURL>
    <Attributes>
        <Attribute id="fullname">
            <Name>
                <en>Full name</en>
            </Name>
            <Description>
                <en>Your full name</en>
            </Description>
        </Attribute>
        ...
        <Attribute id="dateofbirth" optional="true">
            <Name>
                <en>Date of birth</en>
            </Name>
        </Attribute>
    </Attributes>
</IssueSpecification>
```

Data model

Based on the *client* and *storage* package within *irmago* we can visualize the relationships between the entities of the data model as shown in figure 4.4. The *client* object references different *CredentialTypeIdentifier*s depending on which credentials the user retrieved from an issuer. Each *CredentialTypeIdentifier* refers to a *credential*. A *credential* includes one *AttributeList* which contains one or more attributes. Each attribute has a key-value pair organized via a map. The key is an instance of *AttributeTypeIdentifier*, and the value is an instance of a *TranslatedString*. The *TranslatedString* contains the actual rawValue, the

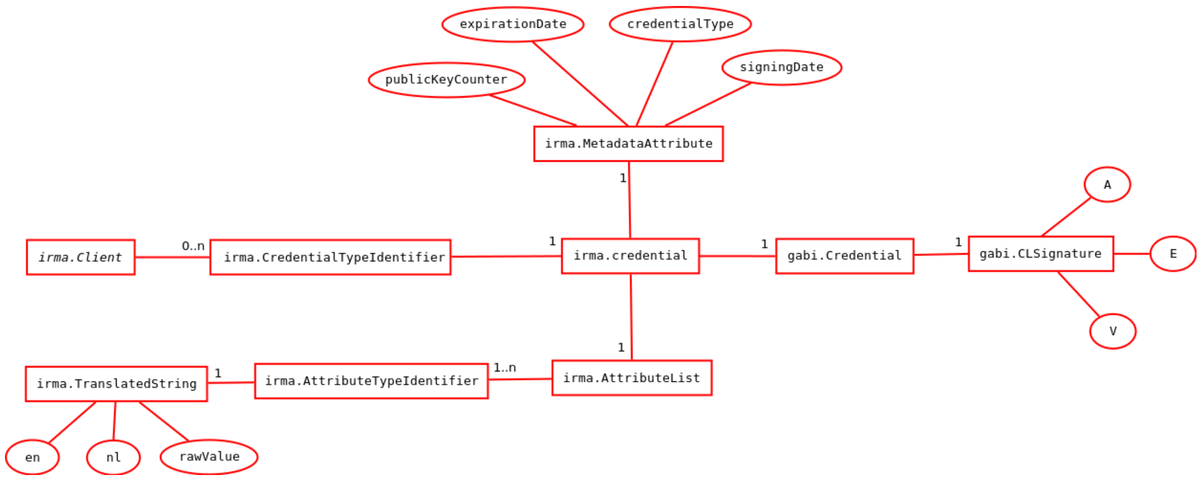


Figure 4.4: IRMA data model

English value, and the Dutch value of the attribute. In addition, each *credential* refers to one *gabi.Credential*. It holds the *gabi.CLSignature* which represents the signature received from the issuer. IRMA stores the actual relevant integer values of the signature in the variables A, E, and V. Also, as said earlier, each *credential* contains relevant metadata. Subsequently, the metadata is an attribute on its own via the *MetadataAttribute*. It contains the credential type, signing date, validity via the expirationDate, and the public key counter.

Security properties

IRMA guarantees several security properties due to the chosen architecture, protocols and technologies:[6]

Credential unforgeability Only the issuer, which holds the Idemix private key, can issue credentials. Hence, a verifier may assume safely they were issued by that issuer.

Multi-show unlinkability A verifier cannot tell if the same user performed two IRMA sessions or not, even if the user discloses the same set of attributes twice.

Issuer unlinkability An issuer of attributes cannot trace disclosures of attributes to identify a user, even if the issuer is also the verifier.

Grouping credentials using the private key The IRMA app can prove that different credentials share the same secret key. Subsequently, the attributes must come from the same user.

Eavesdroppers cannot perform replay attacks A verifier can detect if a nonce is used more than once and can reject the request. The verifier sends in each session the so-called nonce to the IRMA app, and the IRMA app uses this nonce in a certain fashion,

Verifiers cannot perform replay attacks The IRMA app never sends a complete copy of the credential's signature to the verifier as the app hides parts of it using ZKP. Consequently, verifiers are not able to reuse the credential to disclose them to other verifiers.

[6]See also https://irma.app/docs/overview/#irma-security-properties

No impersonation attacks Due to its decentral nature, the IRMA architecture guarantees that no other party than the user and the requestor are involved when exchanging the contents of credentials.

No privacy hotspots The user sends attributes directly to the verifier without passing through a central party.

Selective disclosure Users have full control of which attribute they disclose.

4.2. Verifiable Credentials

In 2017 the W3C established a working group, which was tasked to develop a data model and syntax for the expression of verifiable credentials.[7] Out-of-scope was defining an attribute exchange protocol and browser APIs. The working group states that "there is currently no widely used self-sovereign and privacy-enhancing standard for expressing and transacting verifiable claims (aka credentials, attestations) via the Web". A digital verifiable credential is more tamper-evident due to the use of technology than, for instance, a physical passport. The employed technology shall make it easier and more secure to express and exchange credentials.

Initially, the working group identified three distinct problems. First, there is no standard available for users to prove claims to a verifying party, resulting in error-prone manual input and also fraud on the web. Second, users and their claims do not independently exist from service providers, which leads amongst others to vendor lock-in and reduced privacy for all stakeholders; Third, only industry-specific solutions are available and no interoperable standard of expressing credentials.

Based on these problems, the working group identified three main goals it tries to achieve:[8] [9]

- Create a standard way that users can assess their credentials to a service provider, which consequently can lower fraud on the internet;
- Ensure that users and their associated claims can be independent of service providers to enable users to change their SP, which prevents vendor lock-in;
- Ensure that there is an interoperable standard for expressing verifiable credentials, leading to that users can manage their digital identities uniformly.

The main objective of the VC standard is to provide implementers with an unambiguous specification that supports them in building interoperable user-centric and privacy-friendly identity systems.

The data model document [47] is structured in different sections. In the introduction, the reader gets general information about VC and what its use-cases are. Then, in the *core data model* section, the crucial concepts are described. In particular, claim, credential, and presentation, which we introduce in more detail in later subsections. In the fourth and fifth sections, the document lists concepts, whereby some are normative, and others are non-normative. Normative means that those parts are describing how to comply with the standard, whereby it does not define whether the part is a MUST-have or only RECOMMENDED

[7] See `https://www.w3.org/2017/vc/charter.html` for more details

[8] See the VC working group FAQ: `https://w3c.github.io/webpayments-ig/VCTF/charter/faq.html`

[9] A survey`https://w3c.github.io/webpayments-ig/VCTF/support` among 91 organizations shows that out of the 56 organizations participating 93 % of them support the problem statement and 96 % support the goals and that those were reasonable goals to pursue.

to implement.[10] Non-normative, or also called informative, supports the conceptual understanding. In the next section, syntax, the authors explain how the data model is realized with JSON or JSON-LD. We leave the following sections out-of-scope for this thesis.

4.2.1. USE-CASES

The working group identified several problem domains where VCs can be applied, listed on `https://w3c.github.io/vc-use-cases/`. We picked per domain one example to give the reader an idea which implementations are imaginable:

Finance With VCs, a user can prove parts of her identity relevant for a money transfer or even opening a new bank account via the internet.

Education A system can identify students for online classes via an 'I am a student at university x'-claim.

Healthcare A doctor could use his doctors' certificate to write prescriptions, which improved accountability and verifiability.

Retail A retail company could request address-attributes from customers to verify their address.

Professional A system can use certificates to prove that one can give certain training. With an expiry date added to the certificate, participants can ensure that the person giving the training is still officially qualified.

Legal identity Long lines due to airport security checks could be avoided in the future if VCs are employed to verify a person's legal identity.

4.2.2. ROLES

In the following, we describe several roles participating in the VC ecosystem:

Holder Receives credentials from an issuer and usually stores credentials in some kind of repository;

Issuer Issues credentials to holders;

Subject Usually having one or more VCs asserted about it, and in most cases, the holder is also the subject, but it is not always the case, e.g., a mother could hold the VCs of her child;

Verifier Requests claims from holders to authenticate them;

Verifiable Data Registry Mediate creation and verification of identifiers, keys, VC schemas, and revocation registries.

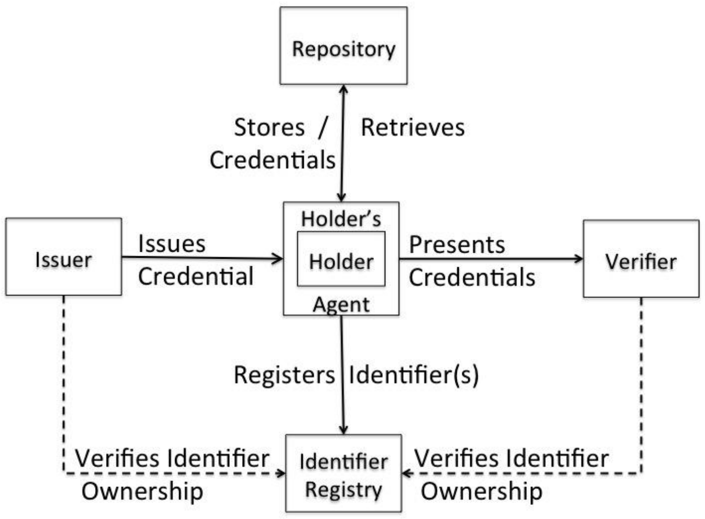


Figure 4.5: VC architecture (copy from https://w3c-ccg.github.io/vc-lifecycle/)

4.2.3. Architecture

In figure 4.5, the interactions between the different roles of a basic verifiable credential system are depicted. Initially, the issuer issues credentials to the holder's agent (e.g., a smartphone) before a holder can present attributes to any verifier. To store the credentials, the holder puts them into a repository owned by herself, which gives the holder full control of her credentials. The holder can ask her agent to compute and send a verifiable presentation to the verifier to present credentials. VC systems may use a verifiable data registry, or previously called identifier registry, to create and verify identifiers, schemes, issuer public keys, and any other relevant data needed to run sessions within a VC ecosystem. The verifiable data registry can either be centralized, for instance, a centralized database, or decentralized, for instance, a blockchain.

In the next sections, we introduce essential concepts of the VC data model, in particular claims, credentials, presentations, and extensibility.

Claim

A claim (or attribute) is a statement about a subject, e.g., a person is older than 18 years. A subject does not necessarily need to be a human being but can be any other entity about which claims can be made. Claims are expressed by using *subject-property-value* relationships, referred to as *graphs*, as shown in figure 4.6. "Mr. Pat" (subject) is an alumni of (property) "Open Universiteit" (value). As described in secion 3.2 implementers can make use of JSON-LD to model those kinds of relationships and more complex ones.

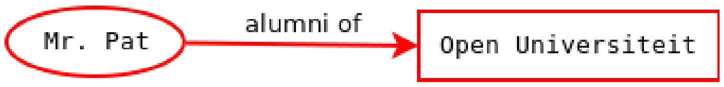


Figure 4.6: Claim example

[10]Inspired by Wikipedia: https://en.wikipedia.org/wiki/Normative#Standards_documents

CREDENTIAL

According to the specification, “a verifiable credential (VC) is a tamper-evident credential that has authorship that can be cryptographically verified”. “Tamper-evident” means that for parties, it shall be easy to gather evidence if tampering has occurred within a credential. “Cryptographically verified” means that a software system can verify the contents of a VC cryptographically without human interaction. A VC can contain claims about different subjects. However, for the sake of simplicity, we state that a holder of a VC is also the subject.

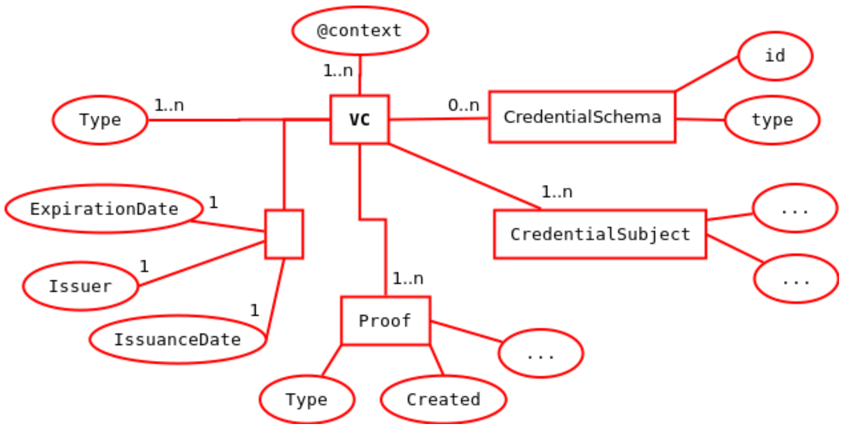


Figure 4.7: Verifiable Credential data model

Since we express the concepts of a VC in subject-property-value relationships, we can easily translate this into an Entity Relationship Diagram (ERD). In figure 4.7, we show that a verifiable credential consists of several required entities. The *@context* property tells systems which terminology both systems must understand to be able to communicate with each other. Each credential contains *credential metadata*, in particular, which issuer issued the credential, when it was issued, and optionally, an expiration date. A *type* determines if a VC (or verifiable presentation) is appropriate in the current situation. For example, if a holder expects a UniversityDegreeCredential type, but receives a HigeschoolDegreeCredential, it can reject the received credential and inform the issuer about it. The VC specification supports mechanisms to have other parties dereference the type property. That can result in machine-readable information about the type. A credential has at least one claim included about a subject, which a VC expresses via the *credentialSubject* property. The *credentialSchema* property is used in two different ways, either for data verification purposes for verifying that a VC conforms to a published schema or for data encoding purposes for mapping the contents of a VC to another representation format, such as ZKP. Also, the credential contains one or more proof objects depending on the proof mechanism utilized. A proof in a credential is a signature generated by the issuer to verify the authorship of the credential.

PRESENTATION

When identifying oneself, a person usually only wants to show the claims of herself that she needs for the intended purpose. The expression of a subset of a person’s claims to a verifying party is called a verifiable presentation. As depicted in fig 4.8 a verifiable presentation consists of several elements. The most significant difference with a VC is that a presentation contains derived verifiable credentials. A derived VC is a VC generated from an earlier

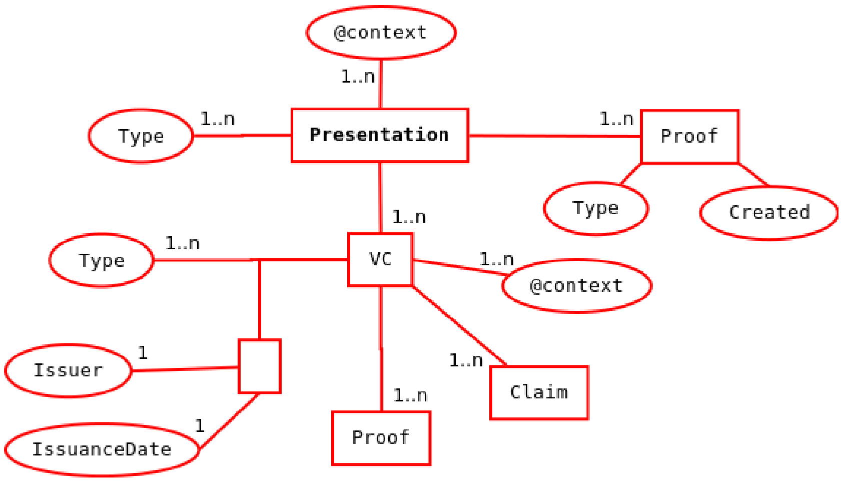


Figure 4.8: Verifiable Presentation data model

received VC from an issuer. The derived VC may contain only a subset of claims needed by the use-case. For instance, by utilizing ZKP, users can selectively disclose claims. The VC specification explicitly supports the use of *Zero-Knowledge Proofs*. Two other concepts are needed to implement ZKP within a verifiable presentation. First, each derived credential needs to contain a credential definition via the "credentialSchema" property to perform cryptographic operations. And second, the verifier uses the "proof" property to check that all derived credentials come from the same user.

Figure 4.9 shows a visual example of the relationship between VCs and the derived credentials in a ZKP presentation. Only some of the claims of the original VCs are part of the final presentation via selective disclosure. In this example, each derived credentials contain a proof property to prove the "knowledge of signature". In the presentation, the proof property contains the "common link secret", which proves that the credentials are all from the same holder.

Extensibility

Another goal of the data model is to enable *permissionless innovation*. Permissionless innovation has its roots in the discussion if the web must remain open. For instance, Cerf [45] argues in favor of it, one of the fathers of the internet. *Permissionless* means that everyone shall be able to *innovate* without the need for prior approval. To enable everyone to *innovate* within the scope of the specification, the data model implements several mechanisms to make it *extensible*.

First, through the use of a graph-based data model, it becomes possible to model complex relationships. For instance, a subject retrieves a driving license credential, which we visualize in figure 4.10 Within the credentialSubject, the driving school where the subject took his driving lessons is added as an object. The driving school has a contact person who we add as a person object. Also, the driving school chooses the instructor, who we also add as a person object.

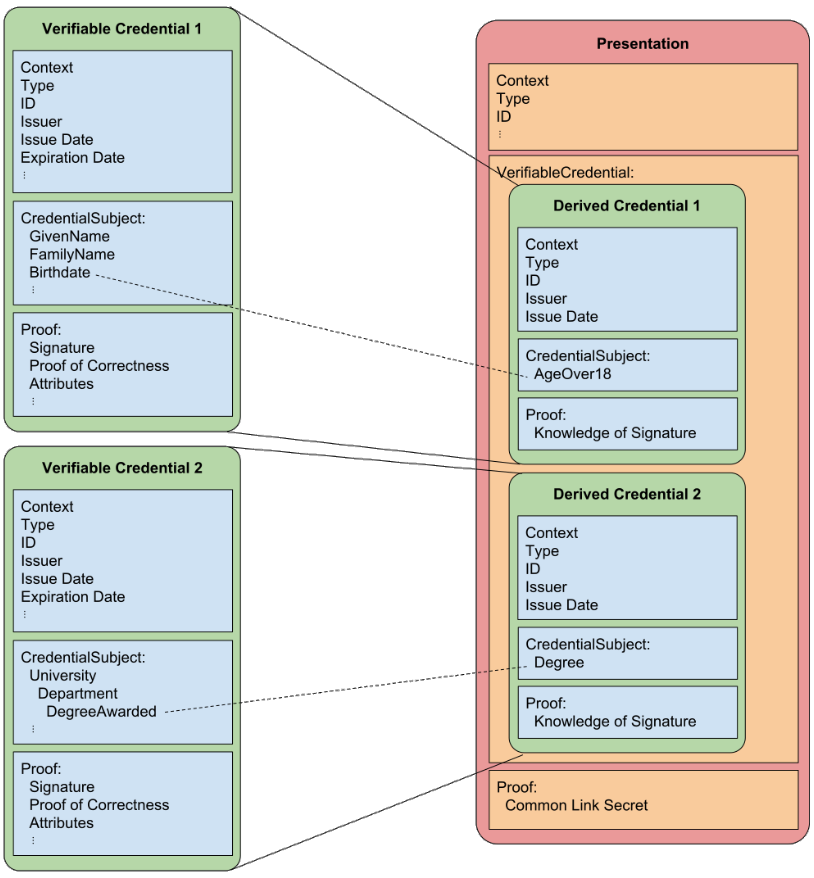


Figure 4.9: ZKP presentation (copy from VC data model [47])

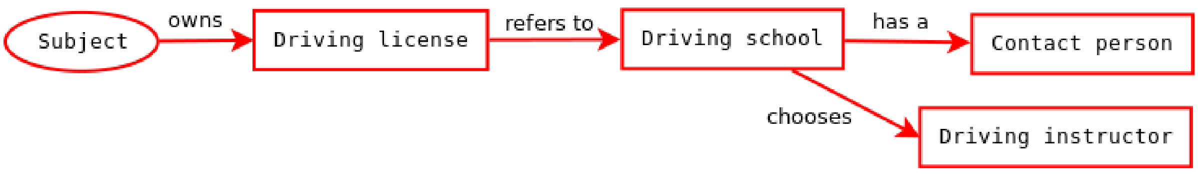


Figure 4.10: Driving license graph example

Second, through the support of multiple types of proof formats. The working group felt the need to document two proof formats in the VC data model, in particular *JSON Web Token*, and *Linked Data Proofs*. However, other systems, such as IRMA, can employ different ones.

Third, extend the machine-readable vocabulary to describe information by using linked data without the need for a centralized system. In theory, any system shall be able to add new types dynamically in a decentralized manner. The data model describes an example by adding a corporate reference number and favorite food to a subject.

However, the more extensible a software becomes, the more complex the software can become. Hence implementers need to weigh how flexible their system should or even can be, for instance, due to legal obligations.

4.3. COMPARE AND CONTRAST

The goal of this research is to explore the relation between the concepts of IRMA and VC. Hence we compare and contrast the two based on goals and concepts. As visible in venn diagram 4.11, we can identify specific similarities and differences. In the following paragraphs, we elaborate on each of those in more detail. First, we describe the differences, and afterward, the similarities. At last, based on the findings, we provide a conclusion.

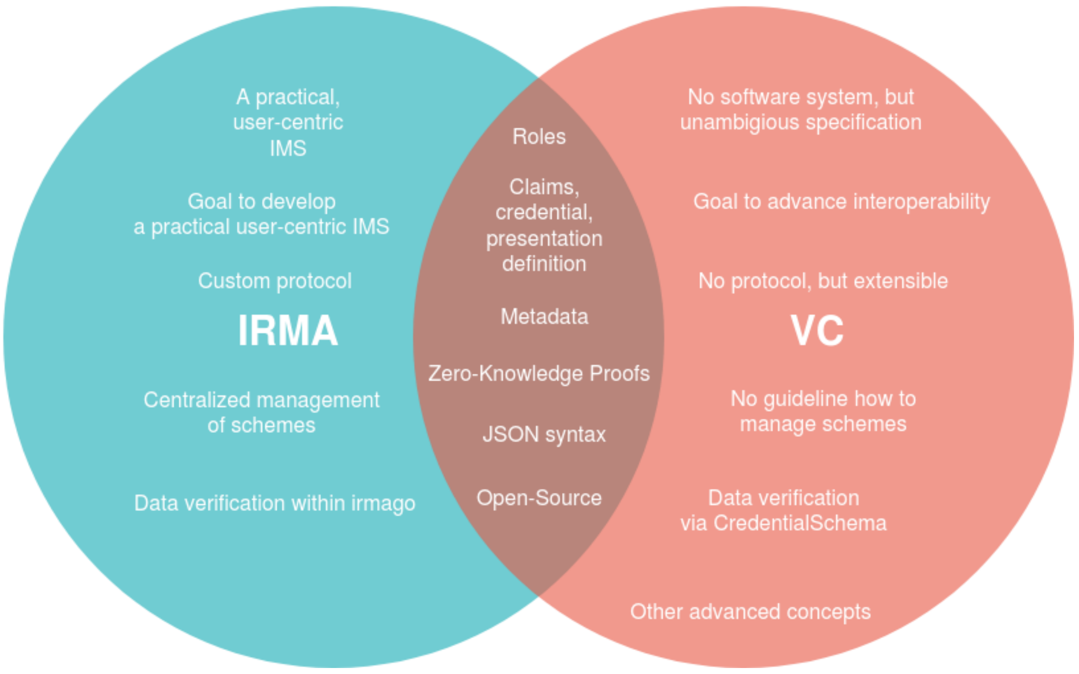

A practical, user-centric IMS
Goal to develop a practical user-centric IMS
Custom protocol
IRMA
Centralized management of schemes
Data verification within irmago
Roles
Claims, credential, presentation definition
Metadata
Zero-Knowledge Proofs
JSON syntax
Open-Source
No software system, but unambigious specification
Goal to advance interoperability
No protocol, but extensible
VC
No guideline how to manage schemes
Data verification via CredentialSchema
Other advanced concepts

Figure 4.11: Goals and concepts in IRMA and VC

IRMA is a **practical and user-centric IMS**, providing us with components for the front-end (irmajs and the IRMA app) and backend (irmago). In contrast, the VC standard does not provide us with a software system but a **specification**. However, the user-centric identity model plays a crucial role in the specification.

One goal of the VC standard is to extend **interoperability** of user-centric identity systems. For users, one benefit is that they can more easily transfer credentials from one system to another, thereby preventing vendor lock-in. Also, users might be able to receive credentials from other systems and disclose credentials across systems. The IRMA project focuses on promoting and improving its system. In the following chapters, we focus extensively on how to adapt IRMA to extend its interoperability.

IRMA implements a **custom protocol**, as described in appendix B. On the other hand, the VC standard does not provide us with a protocol. However, the VC working group produced an unambiguous specification providing us with a data model, related concepts, and syntax. Therefore, we take the data model of IRMA and enrich the entities with the name of entities of the VC data model via red text next to the IRMA entities, as visible in figure 4.12. Within IRMA, each credential is uniquely identifiable via the CredentialTypeIdentifier. Subsequently, we can use this identifier to extend the *type* property of a VC. IRMA's *publicKeyCounter* and *credentialType* are both needed when resolving issuer information. The CLSignature within IRMA provides us with *proof* information. Within the current IRMA data model, a credential holder is always the subject of the credential, and hence one credential contains one and only one subject. Therefore, we can link the *AttributeList* within IRMA to the *credentialSubject* property of VC. Each *AttributeTypeIdentifier* within an *AttributeList* translates to a claim about the subject.

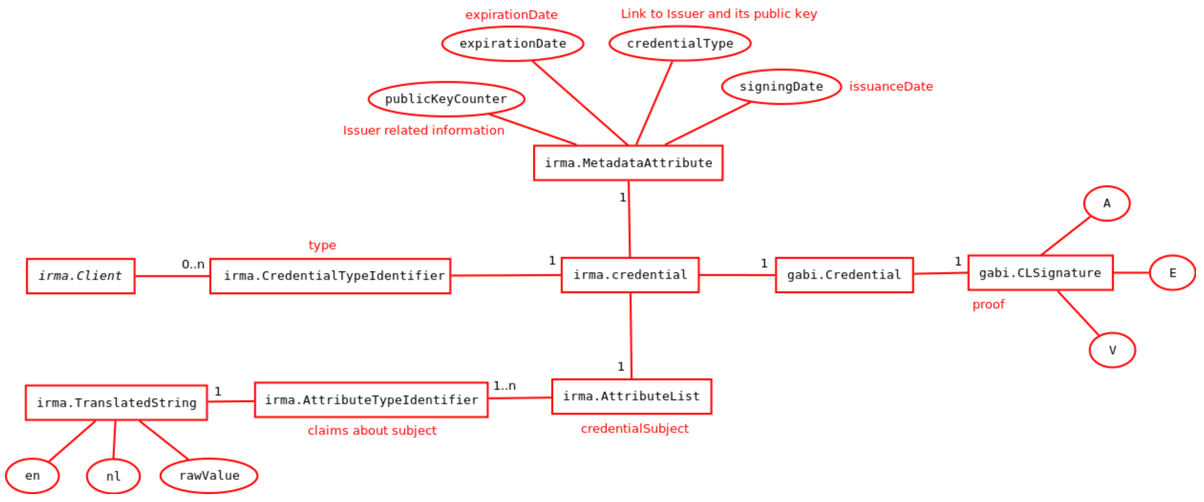


Figure 4.12: IRMA data model with related VC entities

Making the VC data model **extensible** gives implementers some freedom about how to implement their protocol within the VC ecosystem. For instance, IRMA can embed signatures within the proof object of a VC. Then, IRMA needs to transparently communicate how to implement the cryptography to enable other systems to verify that signature.

Within IRMA, maintaining and distributing the scheme is a **centralized** responsibility. The role associated with this responsibility is called *scheme manager*. Hence, within IRMA for each scheme, there is a centralized *root of trust*. All parties need to trust the scheme manager to maintain the scheme correctly. IRMA solves this trust issue by signing the content of a scheme. Then, each party can verify the integrity of the scheme. In VC the *scheme manager* role is called *verifiable data registry*. VC does not provide guidelines on how a system shall maintain and distribute scheme information.

IRMA enforces **data verification** by verifiying messages within *irmago* based on information within a *scheme*. VC introduces the concept of *CredentialSchema* to enable parties to share data verification information by resolving URIs, for instance, via JSON schemas. To ensure the integrity of the data obtained via CredentialSchema, most recent literature, such as Muehle et al. [27], or projects, like Sovrin, emphasize or employ blockchain technologies in combination with DIDs.

The VC specification provides us with other **advanced concepts**, which IRMA currently does not support:

Status The current status of a credential, e.g., indicate whether a credential is suspended or revoked;

Refreshing Automatically refresh an expired credential;

Terms of Use Parties can use this to communicate terms under which VCs can be used;

Evidence Includes additional evidence supporting the verifier to establish sufficient confidence in the claims within a credential;

Disputes For example, if a holder disputed a claim made by an issuer, e.g., the address is incorrect.

They are out-of-scope for this research.

All **roles** defined within the VC specification are also available within the IRMA system.

The **definition** of claims and also credentials are the same on a conceptual level. Also, the concept of a *verifiable presentation* is similar within IRMA and called a “disclosure proof”.

Certain **metadata** is essential in any user-centric IMS. Therefore, both IRMA and VC have metadata in common. In particular, the issuer that issued the credential, when it was issued, and when it expires.

IRMA uses **zero-knowledge proofs** in both issuing and disclosure sessions. Since the VC specification explicitly supports ZKP to be used in a disclosing session, we can generate a ZKP presentation based on the IRMA data model. Later in this research, we elaborate on this again when creating a prototype in chapter 6.

According to the VC specification, services or applications can use any data representation **syntax**, such as XML or JSON, capable of expressing the data model. Yet, all serialization syntaxes need to translate to the data model deterministically to enable other parties to process and validate messages in the same way. The data model can be encoded entirely by utilizing JSON, and IRMA uses JSON to exchange messages between the different components. Also, in theory, it might be easy to modify IRMA to support JSON-LD.

All source codes of IRMA and VC are **open-source** and stored in different repositories on GitHub.

We conclude that it might be possible to modify IRMA to comply to the VC standard with acceptable effort. We identify overlap in roles, definitions of key concepts, and metadata. Also, VC supports the use of ZKP. IRMA uses JSON, and implementers can express the VC data model in JSON. We show how we can align the IRMA data model with entities of the VC data model. Also, VC leaves implementers freedom on how to implement a protocol due to the data model's extensibility.

In the following chapter, we identify requirements to describe which modifications within IRMA are needed to comply with the VC standard. It might result in extending IRMA's interoperability.

5
Requirements, modifications, and impact

In this chapter, we first elaborate on the requirements of the VC data model. Then, since we concluded in the previous chapter that one of the essential goals of VC is to advance interoperability between user-centric IMSs, we explore which use-cases exist between IRMA and other systems. Those use-cases help us to analyze further which distinct challenges exist to achieve interoperability between IRMA and other systems. Then, we can identify modifications within IRMA to handle those challenges. Afterward, to show the impact of the modifications on the IRMA architecture, we conduct an architectural impact analysis. At last, we provide other systems with a list of modifications they need to take into account if they want to connect with IRMA.

5.1. VC Requirements

The VC data model categorizes its requirements based on words defined in RFC 2119 [44]. Commonly, standards use RFC 2119 to indicate requirements. We describe in the following list briefly the words used in the VC data model:

MUST The given definition is an absolute requirement of the data model;

MUST NOT The given definition is an absolute prohibition of the data model;

MAY The given definition is truly optional;

SHOULD / RECOMMENDED The given definition may be ignored due to valid reasons in particular circumstances.

Any conforming system at least needs to meet the absolute requirements and prohibitions to become compliant with the VC standard. Via the *vc-test-suite* any implementation can validate if it conforms to the VC data model.[1] we list the absolute requirements and prohibitions of the VC Data model in appendix C.

A VC system can use a verifiable presentation in a disclosing session. However, it is not mandatory to implement the verifiable presentation concept. For instance, a system

[1]Official repository of the vc-test-suite: `https://github.com/w3c/vc-test-suite`

could only implement the issuing part, and still wants to conform with the VC standard. But within IRMA, credentials are disclosed. Consequently, IRMA shall employ a verifiable presentation. When a verifiable presentation is used in combination with ZKPs, additional requirements and prohibitions apply. We list those also in appendix C.

5.2. USE-CASES

Malan et al. state in [25] that "use cases have quickly become a widespread practice for capturing functional requirements". A use-case describes how an actor can achieve a particular goal when using a system. To visualize different goals from different actors in one figure, often a use-case diagram is used.

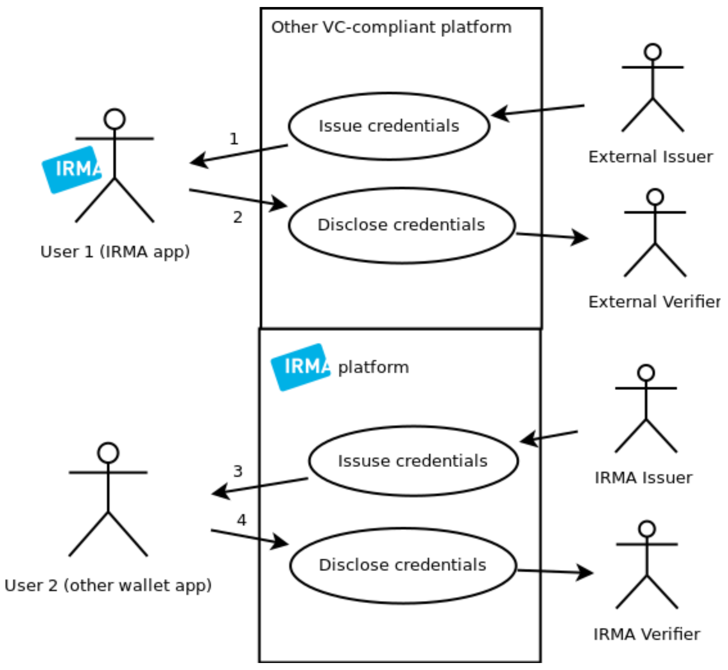


Figure 5.1: IRMA and VC use-cases

To extend IRMA's interoperability, we are interested in use-cases where either an IRMA user interacts with actors from other systems or a non-IRMA user interacts with IRMA actors. Hence, we can identify the four use-cases in use-case diagram 5.1. In theory, it is possible to identify eight use-cases between the actors from left to right, which we show for completeness reasons in table 5.1. Use-cases, in which only IRMA actors participate, are already implemented, and not relevant for this research. Additionally, we are not interested in use-cases between only external actors, as IRMA is not involved. For example, a non-IRMA user receives credentials from an external issuer. Subsequently, she uses those credentials in a disclosure session with an external verifier.

Use-case 1 shows a user using the IRMA app. The user wants to request credentials from an issuer, which is not employing an IRMA server. Afterward, the user wants to store those credentials within the IRMA app. Challenging here is that the app needs to store the credential with all its properties.

Use-case 2 shows an end-user using the IRMA app. The user has credentials stored on her phone. Then, she wants to authenticate to a verifier, which is not employing an IRMA server to disclose credentials to an external verifier. Interesting in this use-case is that the

Table 5.1: Complete use-cases

Use-case Id	Actor IN	Actor OUT	Action
1	External issuer	User 1	Issue Credentials
2	User 1	External Verifier	Disclose credentials
3	IRMA Issuer	User 2	Issue Credentials
4	User 2	IRMA Verifier	Disclose credentials
5	External Issuer	User 2	Issue Credentials
6	User 2	External Verifier	Disclose credentials
7	IRMA Issuer	User 1	Issue Credentials
8	User 1	IRMA Verifier	Disclose credentials

IRMA app needs to scan and process a non-IRMA QRCode. Also, the app needs to compute a verifiable presentation that external verifiers can process.

Use-case 3 shows an end-user using a non-IRMA wallet app. The end-user requests credentials from the IRMA issuer. The issuer returns the credentials. The challenge in this use-case is to compute VC-compliant credentials.

Use-case 4 shows an end-user using a non-IRMA wallet app. The user has credentials stored on her phone. Then, she wants to authenticate to an IRMA verifier by disclosing credentials via a verifiable presentation. In this use-case, the exciting part is that the IRMA server must be able to process a verifiable presentation. Subsequently, the server needs to obtain relevant metadata from an external system.

In the following paragraphs, for each use-case, we describe which interactions between actors are needed to realize the use-case. We choose to use sequence diagrams as we already describe the IRMA protocol via sequence diagrams in appendix B. Interactions between actors or actions within an actor leading to modifications within IRMA are marked bold in the diagrams and elaborated later on. Additionally, we limit ourselves by focusing on happy flows, and thus, we leave unhappy flows, in particular error handling, out-of-scope for this work.

Within each use-case, we mention that a user needs to scan a QRCode. However, within IRMA, it is also possible that instead of a QRCode, the requesting party embeds an IRMA deep-link in case the user uses a mobile browser. Then, the IRMA app opens directly instead of showing a QRCode, thereby establishing the session between the user and IRMA server.[2] Also, it is possible to employ other technologies such as Bluetooth or NFC. But we leave them out-of-scope for this research.

External issuer issues credentials to IRMA user As shown in figure 5.2, the user first needs to open the website from the external issuer. The issuer asks the user to authenticate to verify the user's authenticity before establishing an issuing session and showing her the QRCode. Then, the user scans the QRCode via the IRMA app. Subsequently, the IRMA app sends a commitment, and the external issuer returns the generated VC. Finally, the IRMA app needs to store the received VC to be able to use it in a disclosing session.

[2] See `https://irma.app/docs/api-irmajs/#handlesession` for the related technical documentation.

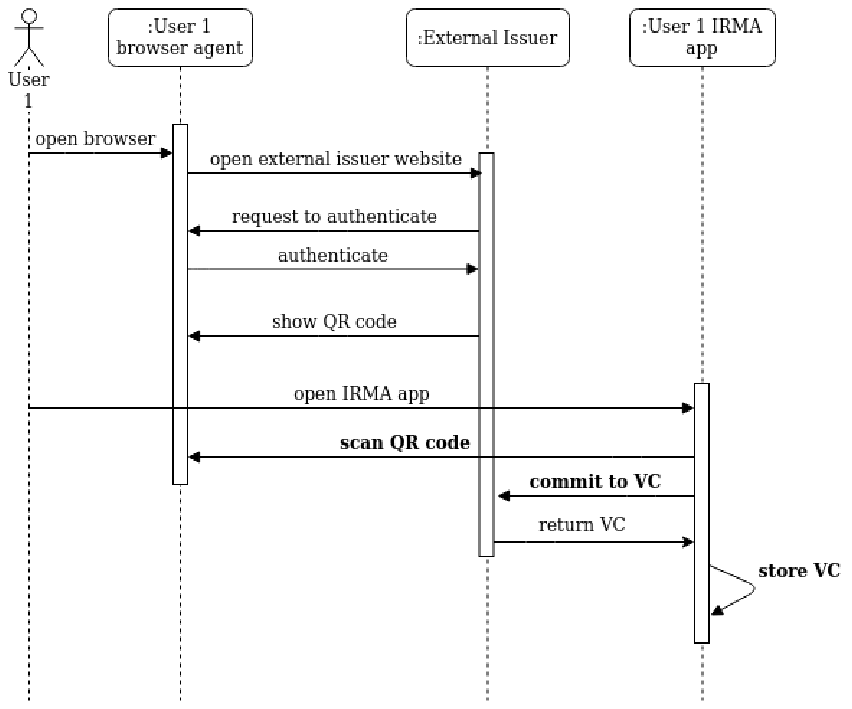


Figure 5.2: Use-case 1 - External issuer issues to IRMA user

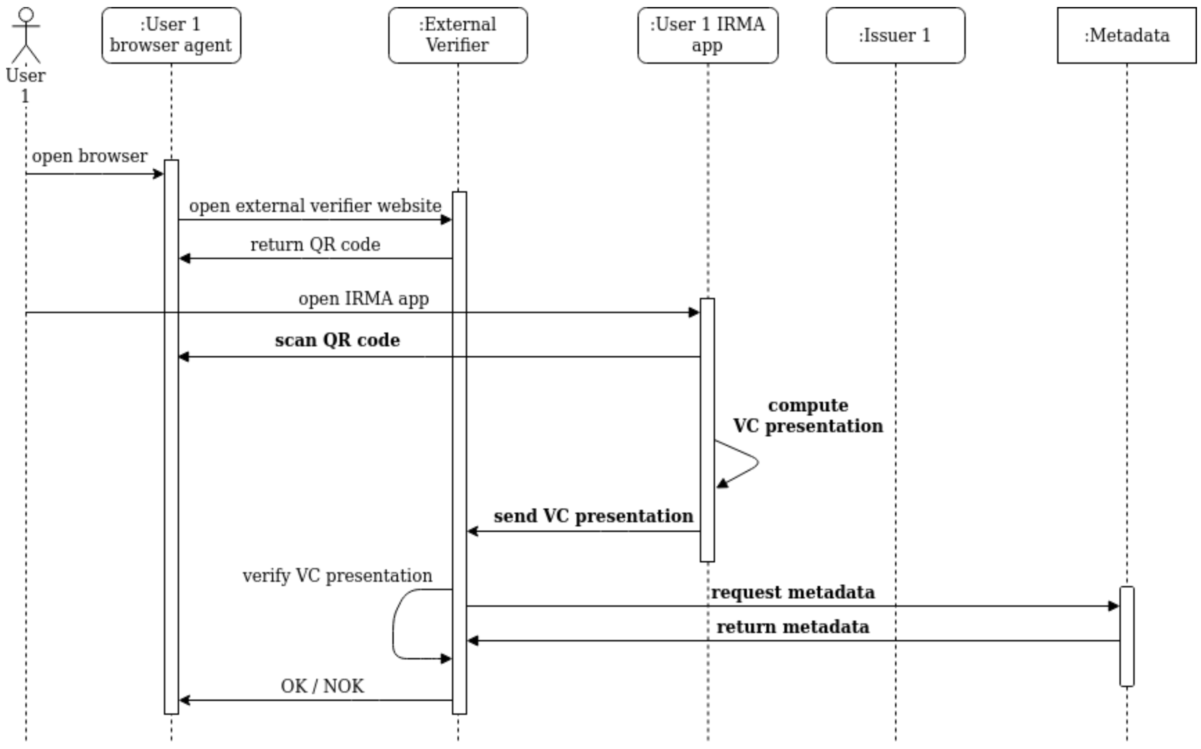


Figure 5.3: Use-case 2 - IRMA user discloses to external verifier

IRMA user discloses credentials to external verifier A prerequisite for this use-case, as depicted in figure 5.3 is that the user already has credentials stored in the IRMA app. The user opens the website from the verifying party, and the browser agent shows QRCode. In the next step, the user scans the QRCode with the IRMA app, and consequently, the external verifier requests to disclose attributes. After the user gave consent, the IRMA app computes a verifiable presentation and send it to the external verifier. The external verifier then verifies the retrieved presentation, wherefore it has to obtain specific metadata.

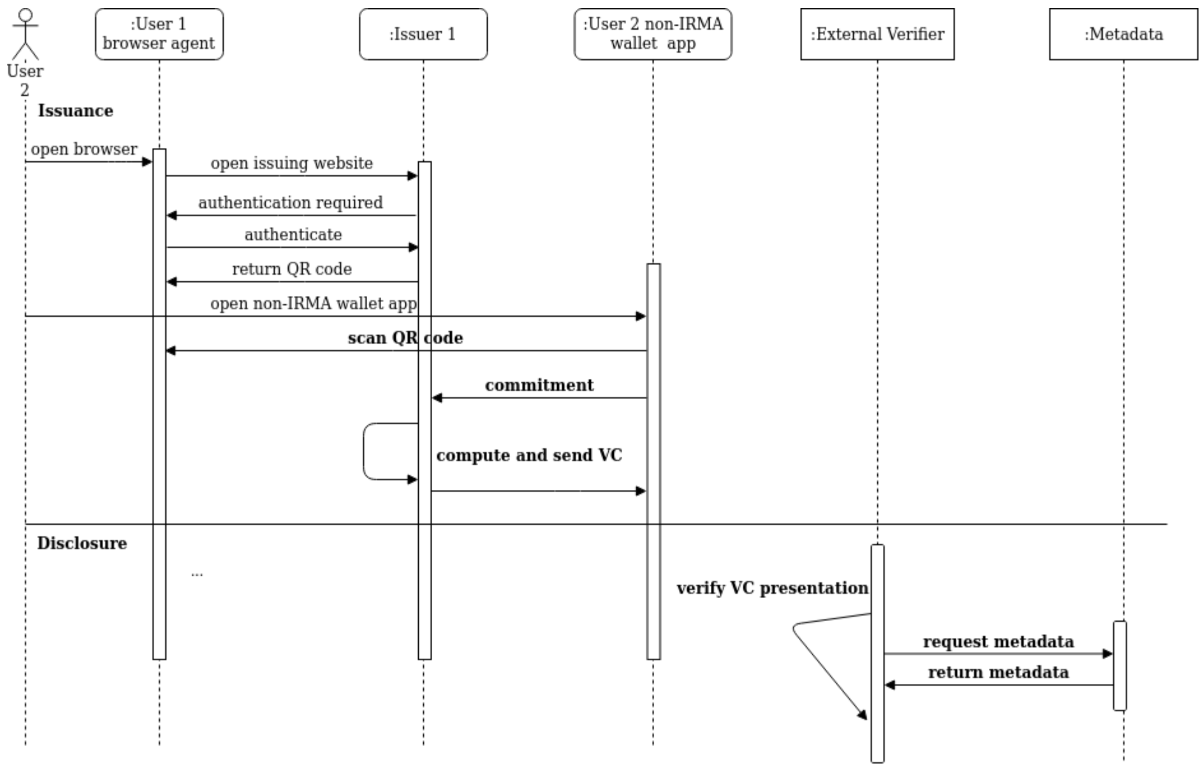


Figure 5.4: Use-case 3 - IRMA issuer issues to non-IRMA user

IRMA issuer issues credentials to non-IRMA user As shown in figure 5.4, a user opens, as usual, the website of the issuer. The issuer asks her to authenticate, and after doing so, the issuer, which runs an IRMA issuing server instance, returns the QRCode to the users' browser agent. The user scans the QRCode with her non-IRMA wallet app to establish a connection with the issuer. Then, the app computes and sends the commitment to the IRMA issuer. After verifying the commitment, the issuing server creates a VC and returns it to the users' wallet app. If the user uses those credentials in a disclosing session, the verifier has to obtain metadata.

Non-IRMA user discloses credentials to IRMA verifier As shown in figure 5.5, first, the user visits the verifiers' website and then scans the QRCode with her non-IRMA wallet app to establish the disclosing session between the app and the IRMA server of the verifier. Then, the app asks the verifier to send the disclosure request. After the user approves the request, the app sends a verifiable presentation to the verifier. The verifier must be able to parse the verifiable presentation and request and process metadata to be able to verify the presentation.

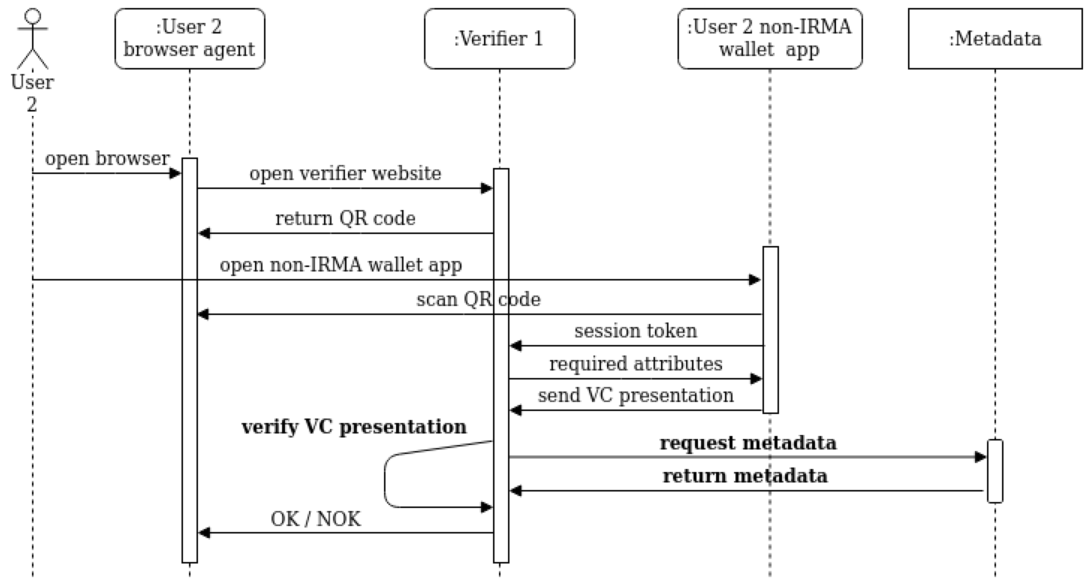


Figure 5.5: Use-case 4 - Non-IRMA user authenticates to IRMA verifier

5.2.1. IRMA Modifications

Based on the interactions within the four use-cases, we can identify modifications within IRMA to realize the use-cases, which we list in table 5.2. Each row in the table leads to one modification based on one or more related use-cases, the involved actor, and the (inter)action. In the following paragraphs, we describe each modification in more detail.

1: Scan QRCode Currently, there is no standard available that specifies how to compute a QRCode within a VC system. Hence, for each new system that IRMA wants to interact with, we need to implement the parsing of the QRCode manually. Within IRMA, the QRCode consists of session information, which the app uses to establish a connection with the requestor. For other systems, the QRCode might contain similar information, but the structure of the QRCode itself can be very different.

2: Compute Presentation If other parties deploy a non-IRMA server for verifying purposes in a disclosing session, and a user wants to disclose credentials stored in the IRMA app, the IRMA app needs to be able to compute a verifiable presentation after the session is initialized.

3: Send Presentation Subsequently, the IRMA app sends the verifiable presentation to the verifying party via an interface. The QRCode contains information about the interface, or the QRCode contains information on how to get the relevant interface information.

4: Obtain IRMA Metadata After the computed verifiable presentation is send to the non-IRMA verifying party, the verifier wants to verify the disclosed attributes. Therefore, the verifier requests metadata that is referred to within a verifiable presentation. Most importantly, the verifiers need to retrieve information about the issuer, including the public key.

We can argue why a relying party should not download the IRMA scheme as IRMA parties also do. However, for external parties, it shall be more convenient to request the specific

Table 5.2: Modifications identified via use-cases

Id	Use-case	Actor	Interaction	Modification
1	1 / 2	IRMA app	Scan QRCode	The IRMA app must be able to parse QRCode of external actors
2	2	IRMA app	Compute verifiable presentation	The IRMA app must be able to compute a verifiable presentation
3	2	IRMA app	Send verifiable presentation	The IRMA app must be able to send the verifiable presentation
4	2 / 3	External verifier	Obtain IRMA metadata	External actors must be able to obtain relevant IRMA metadata
5	1	IRMA app	Compute commitment	The IRMA app must be able to compute a commitment
6	1	IRMA app	Store VC	The IRMA app must be able to store a VC
7	3	IRMA server	Receive commitment	The IRMA server must be able to receive a committed VC
8	3	IRMA server	Compute VC	The IRMA server must be able to compute a VC
9	4	IRMA server	Verify verifiable presentation	The IRMA server must be able to verify a verifiable presentation
10	4	IRMA server	Obtain external metadata	The IRMA server must be able to obtain required metadata to verify the presentation

information they need via the URIs in VC messages. It would be inconvenient for external parties to download the whole scheme and then to filter out the relevant information.

5: Compute Commitment The IRMA app shall request credentials from an external issuer after the user has scanned the QRCode in an issuance session. In most ABC systems, such as in IRMA, the client needs to commit to the VC via a committed credential. Since we expect that other systems use a similar feature, the IRMA-app shall compute a VC-compliant commitment message.

6: Store VC The IRMA app must be able to store a VC permanently. If other VC systems implement concepts, such as "terms of use", and those systems require any connecting system to support those concepts, IRMA must implement those concepts as well.

7: Receive committed VC The non-IRMA wallet app needs to be able to compute a commitment message based on the earlier received nonce in an issuance session (see 6 for reference) as otherwise, the IRMA server is not able to compute the CL signature within the VC.

8: Compute VC The IRMA server needs to compute a signed VC. Here we need to take the MUST-have requirements of the VC data model into consideration. As the messaging is synchronous, the server can just return the signed VC as the response.

9: Verify Presentation If a user with a non-IRMA wallet app sends a verifiable presentation to the IRMA server in a disclosing session, the IRMA server wants to verify the verifiable presentation. As cryptographic protocols can differ between systems, we need to extend IRMA in such a way to support those protocols. Subsequently, IRMA can verify the signatures within the verifiable presentation.

10: Obtain external Metadata To be able to verify the received presentation, the IRMA server needs to obtain the relevant metadata from external systems. In the VC data model, the examples mostly employ DIDs. As a lot of new user-centric systems employ blockchains and DIDs, IRMA might need to resolve DIDs. Then, IRMA can parse the related DID documents.

5.3. Architectural Impact

Lehmans' [22] second Law, "Increasing Complexity", states that, if we do not adequately handle changes, the complexity of the software system increases, making it more challenging to maintain the system in the future. Hence, before implementing a software change, it is useful to analyze which impact changes have on the architecture of the software. Williams and Carver [40] developed a classification scheme called *Software Architecture Change Characterization Scheme* (SACCS) to conduct change impact analysis. After conducting three empirical studies, Williams and Carver conclude [41] that "SACCS provides insight into the difficulty of a change request, SACCS helps to facilitate discussion among developers, and SACCS is a useful tool for supporting change implementation". Hence, we decide to use this scheme, as it uses findings of previous research to classify changes. Also, it is easy to apply the scheme, and it helps to facilitate discussions. I decided to group all the modifications described in the previous section together as one change request (CR). Analyzing each change one-by-one is a cumbersome and too time-consuming task for the scope of this research.

By applying the method, we identify general characteristics of the change, i.E., motivation and type of the change, size, impact on static and dynamic system properties, and the features and quality attributes affected. In appendix D, we provide the curious reader with a summary of descriptions of characteristics used hereafter if it is unclear what is meant by it. Also, SACCS provides us with the *specific characterization*, which allows us to provide more detail about the effects of the CR on the logical and runtime structures. However, we leave the specific characterization out-of-scope of this research. Nevertheless, the result of applying this method is to facilitate a discussion with stakeholders of IRMA to help them decide whether it is feasible to implement the CR. Subsequently, it helps to identify architectural areas to focus on when implementing the CR.

The **motivation** of the CR is an *enhancement* as we want to enhance IRMA by complying with the VC data model and extend the interoperability of IRMA. The **source** of the CR is *stakeholder request* initially based on the topic listed in A provided by my supervisor Greg Alpar, and consequently the critically/importance is *requested*. The **developer experience**, which is myself, I would consider *localized* as I am familiar with the architecture of IRMA. But not extensively as I only started studying it about a year ago. The **category** attribute, in my opinion, might be *perfective*, because it also helps to better meet user needs in the future when VC-compliant systems become more popular. Also, we could include *adaptive*, as our goal is to make IRMA compliant with the VC standard. The **granular effect** is *architectural* as we do not change system functions visible to the user. The **property** of the CR is in my opinion both *static* and *dynamic*. Static, as there is the need to update the underlying data model of IRMA to comply to the VC specification. Dynamic, as we need to introduce a new interface that during runtime responds to external actor's requests to deliver metadata.

In the following tables, we score characteristics, based on an impact scale, and also motivate the score. The impact scale determines the impact the change has on the given attribute. It starts with score 0, meaning *no impact*, and ends with 5, meaning *major focus of change*.

In the first table 5.3, we list and score the related logical and runtime characteristics. Unfortunately, SACCS focuses on object-oriented software characteristics, but we try to align it with the concepts of *go* as well as we can. In the second table, we determine how the CR affects functional requirements, also called **features**, by listing the different areas in table 5.4. In the third table 5.5, we list the different quality attributes and score them. From an architectural point-of-view it is helpful to assess how **quality attributes**, based on ISO Standard 25010, are affected by the CR.[3] We could conduct a more in-depth analysis of each attribute, but this beyond the scope of this thesis. Also, it is more useful to conduct when connecting to other systems. The IRMA project guarantees specific security properties due to cryptographic protocol choices, as described in section 4.1.3. Hence, we elaborate later on security aspects in more detail.

In sum, we can argue that the CR might impact many characteristics to a great extent. Yet, this depends mostly on the system, with which IRMA wants to exchange messages. Hence, we advise the IRMA project to conduct a new change impact analysis for each new system. However, we think the above analysis gives a good overview of how IRMA's architecture is affected. If initially, IRMA "only" wants to comply with the VC data model, we can reason that the *Repository Access* and *Source Structure* attributes are most impacted, as the underlying data model of IRMA needs to be changed.

5.3.1. COMPONENTS AND INTERFACES

Based on the CR, we show in figure 5.6 all components, their interfaces, and the interaction between components. The CR impacts components and interfaces printed in bold type. In the following, we describe these impacts.

First, we introduce a new component, called *metadata server*, which exposes two new HTTP endpoints. The */issuer* endpoint returns information about an issuer, including the public key. The */schema* endpoint returns a JSON-schema, with which an actor can verify the content of a VC or verifiable presentation. In the next chapter, we describe how to

[3]See the official ISO website: `https://www.iso.org/standard/35733.html`.

Table 5.3: Architecture characteristics

Architecture characteristic	**Score**	**Motivation**
Logical		
Layers	2	The existing layers of the IRMA system do not need to change. Eventually, the business logic layer needs to be extended for each new system IRMA wants to exchange messages with.
Inheritance Structure	2	A flexible design supporting the easy addition of new systems would be beneficial, for instance, by utilizing the facade design pattern.
Module Decomposition	0	Not impacted.
Source Structure	2	Moderately impacted due to the addition of VC related structs.
Dependency Relationship	2	Due to the addition of VC related structs, some dependencies are impacted.
Runtime		
Control Flow Processing	2	The flow of processing is primarily impacted by the additional logic of external VC systems, but existing flows are not much impacted.
Repository Access	4	The IRMA app needs to store VCs. Hence the storage functionality needs to be refactored.
Concurrent Processes	0	Not affected as requests are still handled one-by-one.
Component Interaction	0	The interactions of components within IRMA is not affected.
Distributed Components	4	IRMA needs to interact with external components via different interfaces. We elaborate on this later in more detail.
Component Deployment	2	The new metadata component needs to be deployed separately with its own configuration.

implement this new component in more detail.

Second, two endpoints under the *irma* interface have to accept VC or verifiable presentations. In the case of a disclosing session, the */proofs* endpoint needs to accept verifiable presentation. In the case of an issuing session, the */commitments* endpoint needs to accept a committed VC, and return a signed VC.

Third, each external system implements different protocols, possibly different technologies, and employs different APIs. For each of such a system, IRMA needs to be adapted.

Table 5.4: Features

Feature	Score	Motivation
Devices	0	Not impacted.
Data Access	3	The IRMA server needs to handle committed VC messages and verifiable presentations. Also, the IRMA app needs to be able to parse other QRCodes not generated by IRMA to establish sessions with other VC systems.
Data Transfer	4	VCs, verifiable presentations, and related metadata need to be transferred to external components.
System Interface	4	Existing interfaces need to accept VC-compliant messages. One new interface is needed for communicating the metadata. We elaborate on this later in more detail.
User Interface	0	Not impacted.
Communication	4	If other systems utilize, for instance, DIDs, this characteristic can be profoundly impacted.
Computation	4	As each newly added system may use a different cryptographic protocol, the computation characteristic might be highly impacted.
Input/Output	4	Since the VC specification does not dictate which data representation syntax to use, different systems can employ different formats.

5.3.2. Security

McGraw states [26] that the more systems are connected, the more extensible and complex they become, the amount of security vulnerabilities increases. If security vulnerabilities exist in a system, according to Adkins et al. [1], it has the potential to break the trust of users quickly. Once the trust is broken, it undermines the usefulness of the system. Hence, when adapting IRMA to become more connected (due to the metadata server), extensible (due to permissionless innovation of VC), and complex (by additional code to support other systems), we need to analyze the impact on security aspects of IRMA in detail.

According to Scarfone et al. [36], there exist three objectives in the classical model for IT-security: maintaining confidentiality, integrity, and availability. Confidentiality ensures that only an authorized person can access resources. Integrity assures that the data or system can be trusted. Availability means that the systems or data are available when required.

We elaborate on each objective in what follows. Additionally, we elaborate on the impact on IRMA's security properties.

Confidentiality

For confidentiality, the trust relationships in an IMS are essential. If we look at the trust relationships in section 3.1.3, we see that issuers have a role in ensuring that only legitimate users receive their credentials. Other parties need to trust the issuers that they enforce it. On the other hand, verifiers must ensure that only users get access to resources whose attributes match the policies. Consequently, systems need to be transparent about

Table 5.5: Quality Attributes

Quality attribute	**Score**	**Motivation**
Functional suitability	0	The functionality is not impacted by the CR.
Reliability	3	Since external systems are involved, this can be heavily impacted. But specifics can only be determined after connecting to another system.
Performance efficiency	3	Can be impacted as VC related messages can differ in size to the original IRMA messages computed.
Usability	0	Not impacted by this CR.
Security	4	Since the goal is to exchange messages with other systems, the security attribute might be impacted heavily. We elaborate on this later in this chapter.
Compatibility	4	The goal by complying with the VC standard is to make IRMA more interoperable. This impacts the attribute massively.
Maintainability	2	To be able to execute integrated tests with other VC systems, those systems must be available.
Transferability	0	Not affected by this CR.

the management of their trust relationships. If systems decide to exchange messages with each other, it seems reasonable to set up agreements between the two.

INTEGRITY

IRMA employs a mechanism to protect the integrity of a scheme, as described in section 4.1.3. However, we introduce a metadata server. The server exposes scheme information via different endpoints, as described in section 5.3.1. Then, external actors can request specific scheme information via the URLs available in VC messages. Therefore, the current hashes stored in the *index* file of a scheme are useless when obtaining information via the metadata server. Hence, we need to identify new mechanisms to enforce integrity of the information when requesting it via URLs.

One option is to compute hashlinks for each link exposed by the scheme server. Sporny started to develop the hashlink standard in 2018.[4] It is currently a draft version. A hashlink is a modified hyperlink. It enables consuming parties to determine if someone tampered with the resource associated with the link. An advantage is that no separate system is needed to add this level of content integrity protection next to URLs.

Another option is, similar to IRMAs current approach, to compute hashes for all possible URLs that can obtain metadata. Then, IRMA can store those hashes in a file, and subsequently, sign that file. An advantage is that IRMA does not need to introduce any new technology. A disadvantage is that external parties need to manually download the scheme and check the hashes to check the integrity of the content.

[4]More information on hashlink under: `https://w3c-ccg.github.io/hashlink/`

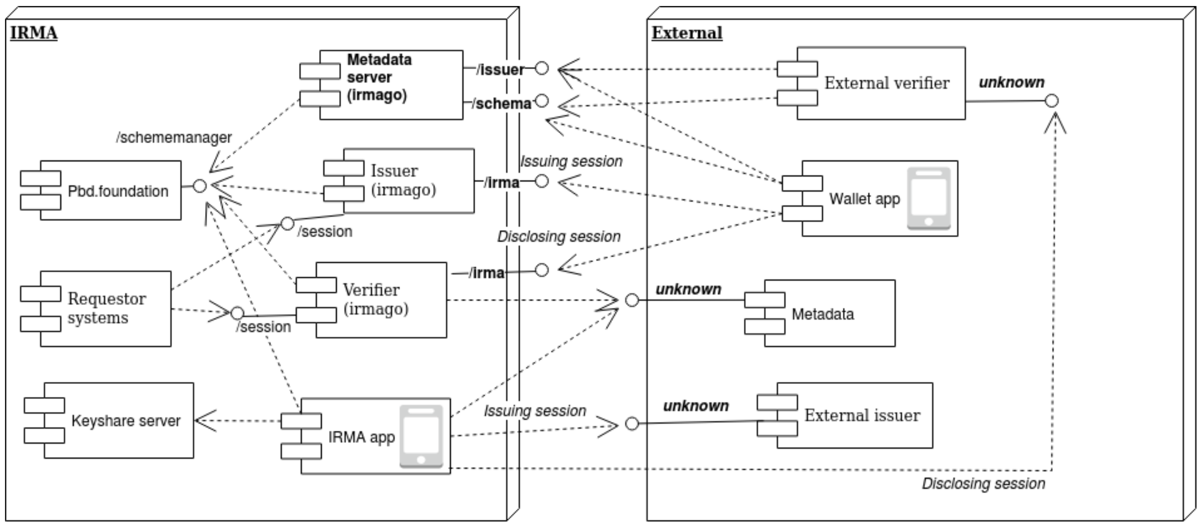


Figure 5.6: IRMA and externl system components and interfaces

On the other hand, if IRMA accepts credentials and presentations from other systems, other systems must enforce integrity of the content. Also, the systems need to communicate transparently about how external parties can check integrity.

AVAILABILITY

If the metadata server is not available, no external actors can verify a message. Hence, the metadata server must be highly available. Loukas and Öke [24] argue that denial of service (Dos) attacks are a significant threat as they are easy to launch. Defending the network, on the other hand, is disproportionately difficult. Consequently, IRMA needs to take measures to detect and defend against Dos attacks. Additionally, the server should be fault-tolerant. This means that at least two instances are running in different locations. If one location has issues and the server is not responding, the other instance can respond to requests.

IRMA SECURITY PROPERTIES

In what follows, for each security property, we describe how the proposed modifications impact them.

Credential unforgeability We may assume safely that other systems ensure that the property holds. Otherwise, credentials can not be trusted.

Multi-show and issuer unlinkability Within the VC data model, identifiers are optional. In theory, however, other systems could enforce the use of identifiers to identify individuals. But this is rather unlikely, as it would reduce the privacy of the individual.

Grouping credentials using the private key This depends on the cryptographic protocol the system employs, from which the IRMA app receives credentials. A challenge is if a user wants to use both IRMA and non-IRMA credentials in one disclosing session.

Eavesdroppers cannot perform replay attacks This depends on the cryptographic protocol the system employs, which the verifier is using.

Verifiers cannot perform replay attacks This depends on which credentials a user uses in a disclosing session. If a user discloses only IRMA credentials, the property holds as IRMA can apply ZKP on its credentials. If a user discloses non-IRMA credentials, it depends if those credentials support ZKP or other mechanisms to hide parts of the signature of the credential.

No impersonation attacks This property is inherent to the user-centric IdM concept.

No privacy hotspots This property is inherent to the user-centric IdM concept.

Selective disclosure This depends on the cryptography employed by systems that issue credentials.

Additionally, due to the introduction of the metadata server, we can argue that *scheme mamanger* can create profiles of parties requesting metadata. We might link this to the *no privacy hotspots* property. If another wallet app obtains metadata when receiving credentials from the IRMA system to verify the content, the scheme manager can log which IP requests the metadata. However, the scheme manager does not know the value of the attributes. Also, the scheme manager can not trace a user disclosing credentials, as the verifier requests the metadata. The scheme manager does not see any content of the verifiable presentation.

5.4. EXTERNAL SYSTEMS MODIFICATIONS

If external systems wish to exchange messages with the IRMA system, they need to apply certain modifications as well. First, wallet apps must be able to scan the IRMA QRCode to establish the session with the IRMA server. Second, the system needs to incorporate the (cryptographic) protocol employed by IRMA. In case of an issuing session initiated by a non-IRMA client, the system must be able to compute a commitment message in case of an issuing session. Additionally, if a user using the IRMA app sends a verifiable presentation, it must be able to verify the disclosed credentials via the signatures. Third, the system needs to obtain the relevant metadata via the metadata server.

6
Prototype

In the previous chapter, we identify modifications within IRMA. In this chapter, first, we explain how we identify the modifications we want to implement. Then, we show how to implement those, thereby delivering a prototype. Afterward, we describe how we alter IRMA to demonstrate that the modifications work as intended. Finally, we provide an overview of the names and URLs to the source-code repositories containing the modifications for the proof of concept.

6.1. Identify modifications to implement

We decided to extend IRMA with the modifications without affecting the core data model and business logic of IRMA. This would be too complex within the scope of this research. Consequently, we do not implement modification *6*. Also, even if VC promotes interoperability, it is still a tedious task to connect to another VC-compliant system. Modifications *1*, *3*, *7*, *9* and *10* only make sense to implement when connecting to another system. But as we want to demonstrate that the modifications work, we choose to build a custom QRCode. Then, after the user scans that QRCode with the IRMA app, the app concludes that it connects to another system.

Consequently, we implement the following set of modifications:

1 The IRMA app must be able to parse the QRCode of external actors;
2 The IRMA app must be able to compute a verifiable presentation;
4 External actors must be able to obtain relevant metadata from the IRMA system;
5 The IRMA app must be able to compute a committed VC;
8 The IRMA server must be able to compute and return a VC.

In the next sections, we describe for each of the modifications, how currently IRMA implements comparable concepts. Afterward, we describe how we need to adapt IRMA to implement the modification. Before describing modification *2* and *4*, we first describe how to compute and return a VC.

6.2. SCAN QRCODE

6.2.1. CURRENT IRMA IMPLEMENTATION

Within IRMA the QRCode contains the URL to establish the session between the IRMA app and the IRMA server, plus the action to be executed, as visible in the following example: *{"u":"http://localhost:8088/irma/f6JOn5NZqheTspDFISzW","irmaqr":"issuing"}*
The scanning of the QRCode is implemented within the *irma_mobile* component in the file *components/QRScanner/QRScannerContainer.js*, event *readQRCode*. After scanning the QRCode, the app validates the QRCode. Afterward, within the app, a new session is initialized via the *NewSession* method of the textitActionHandler struc type. Within that method, the app calls the *client.NewSession* method belonging to the *irmago* library. There, the *sessionrequest* string is parsed to a QRCode object via the *UnmarshalValidate* method.

6.2.2. MODIFIED IRMA IMPLEMENTATION

For each individual QRCode, we define a new struc type. Then, via the *UnmarshalValidate* method within *NewSession*, IRMA tries to create a valid QRCode until there is a match. If there is no QRCode match, it continues with *schemeRequest :=* .... Then, for each external system, within the *client* package, we need to implement a new method similar to *newQRSession*.

Within the *readQRCode* method within *irma_mobile*, we need to change the validation of the QRCode to be less restrictive to allow the IRMA app to parse non-IRMA QRCodes. Therefore, we remove the check *typeof sessionPointer.irmaqr !== 'string'*.

6.3. COMPUTE VC

6.3.1. CURRENT IRMA IMPLEMENTATION

The method *handlePostCommitments* belonging to the *session* struc type computes the message that contains the corresponding CL-signatures of the requested credentials. Then, the method returns the credential to the method *handleProtocolMessage* of the *server* struc type. The method *handleProtocolMessage* generates the to-be-returned JSON response together with the status code. Subsequently, it returns the message as a byte array.

6.3.2. MODIFIED IRMA IMPLEMENTATION

As the *handleProtocolMessage* returns a message as a byte array, within this method, we can call another method that computes and returns a VC. Based on whether the client sends a "legacy" IRMA commitment message or a VC-compliant commitment message, we either invoke the original *handlePostCommitments* method or we call a new method called *handlePostCommitmentsVC*.

In the following paragraphs, we explain how we map IRMA properties to required properties within a VC.

@context As of IRMA server version 0.30 [1], the string-value of the *@context* property identifies what kind of session is initialized. In the VC data model, the *@context* property is an ordered set. The first value always must be *https://www.w3.org/2018/credentials/v1* to let software systems know that the messages exchanged are about VCs. The following items

[1]See `https://irma.app/docs/next/session-requests/` for full details

can express context information. Currently, the IRMA URL does not resolve to anything, but this is not obligatory.

type A software system determines if a VC (or verifiable presentation) is appropriate based on the *type* property. In IRMA, each credential type has a unique name. Via the method call *cred.CredentialTypeID.String()* we can obtain the name. Additionally, the name of the scheme is also part of the string to uniquely identify the credential type. Thus, we can use this unique name to identify a credential, and add it to the *type* set.

credentialSubject The *credentialSubject* property contains the claims made about the subject, e.g. a fullName element that contains the *familyname* and *firstname* attributes. Within IRMA this information is stored in the *AttributeList* of the *irma.credential*. IRMA links each credential to one subject. Hence, within a VC, we only refer to one subject within the *credentialSubject* property.

issuer Each VC needs to contain an *issuer* property . As the property at least needs to contain a URI, the URI deferences into information about the issuer. For instance, the information can contain the public key of the issuer to enable verifiers to verify disclosed credentials. We can obtain the issuer information via the CredentialType. Additionally, via the issuerID we can get the public key counter, which refers to the current public key version: *session.conf.IrmaConfiguration.PublicKeyIndices(issuerID)*.

issuanceDate We can calculate the issuanceDate based on the current date-time in the ISO8601 standard format.

proof A credential needs to contain proof information to make the credential verifiable. The *type* property within the proof object needs to specify which proof method the VC uses, and how a system needs to interpret the containing signature. The CL-signature computed in method *handlePostCommitment* is included in the proof property.

credentialSchema According to the VC data model, each credential used within a ZKP presentation must include a credentialSchema. Since IRMA employs ZKP, we consequently want to compute ZKP presentations later. The credentialSchema consists of one or more data schemas. Each data schema needs to specify an *id* and *type* property. The *id* points to the schema file via a URI. The *type* indicates if we want to verify the structure and contents of a VC or encode the contents of a VC. Unfortunately, we can not figure out how we need to implement the encoding schema.

Hence, we decided to implement a data verification scheme. Since IRMA uses the JSON-format to exchange messages, we employ JSON schemas. JSON schema was developed as an attempt to provide a schema language for JSON. It is comparable to XSD for XML. [2] We choose to add the *aries-framework-go* project as a dependency to the *irmago* project as it automatically verifies a provided VC via the schemas. [3] The credentialSchema needs

[2]Pezoa et al. [32] provided in 2016 a formal definition of syntax and semantics for JSON Schema.
[3]The aries-framework-go repository on GitHub: `https://github.com/hyperledger/aries-framework-go`

to have *JsonSchemaValidator2018* as type. The aries-framework aims to implement DID standards and the VC standard. We implement the *Validator* interface within the VC. Then, within the *Validate* method, we invoke the *NewCredential* method of the aries-framework-go, as we show in listing 3.

Listing 3 Verify the content of a VC within *irmago*

```go
func (vc *VerifiableCredential) Validate() error {
    vcByte, _ := json.Marshal(vc)
    _, _, err := ariesvc.NewCredential(vcByte)
    if err != nil {
        return err
    }
    return nil
}
```

6.4. COMPUTE PRESENTATION

6.4.1. CURRENT IRMA IMPLEMENTATION

Currently, within IRMA the verifier can send a so-called *condiscon* session request to the user.[4] *Condiscon* stands for the conjunction of disjunctions of conjunctions. This request can give the user a choice of which set of attributes to disclose; for example, either the BSN (A1) or full name (A2), together with the postal address of the user (A3). Formally expressed, it looks like: ((A1 AND A3) OR (A2 AND A3)). After the user made a choice, the IRMA app computes the corresponding disclosure proof. Then, the app sends the proof to the verifying party, as we describe in more detail in appendix B.2. In a disclosure session, if a verifier requests attributes from at least two different credentials, the proof message contains for each credential one object within the *proofs* element. As we see in 6.1, the proof is computed in method *getProof* belonging to the *session* struc type. Then, the disclosure proof is send to the verifying party within the *sendResponse* method via the *Post* method of the *transport* type.

6.4.2. MODIFIED IRMA IMPLEMENTATION

Within the *compare and contrast* 4.3 section, we conclude that IRMA employs ZKP, and VC explicitly supports the use of ZKP. Hence, we can produce a mapping from the IRMA code towards a ZKP verifiable presentation, as we show in figure 6.2.

We create a derived VC for each credential type within a disclosing session. In IRMA, the metadata property contains the issuanceDate, which we can obtain via *SigningDate()*. For the expirationDate, the same holds via the *Expiry()* method, it is just not visible on the right-hand side within the presentation. We can compute the credentialSubject for a derived credential by looping over the DisclosedAttributes.

The presentation itself contains the *@context* property as VCs do as well. The second item in the set can be *https://irma.app/ld/request/disclosure/v2* as IRMA currently uses to

[4]More information on the official IRMA doc: https://irma.app/docs/condiscon/.

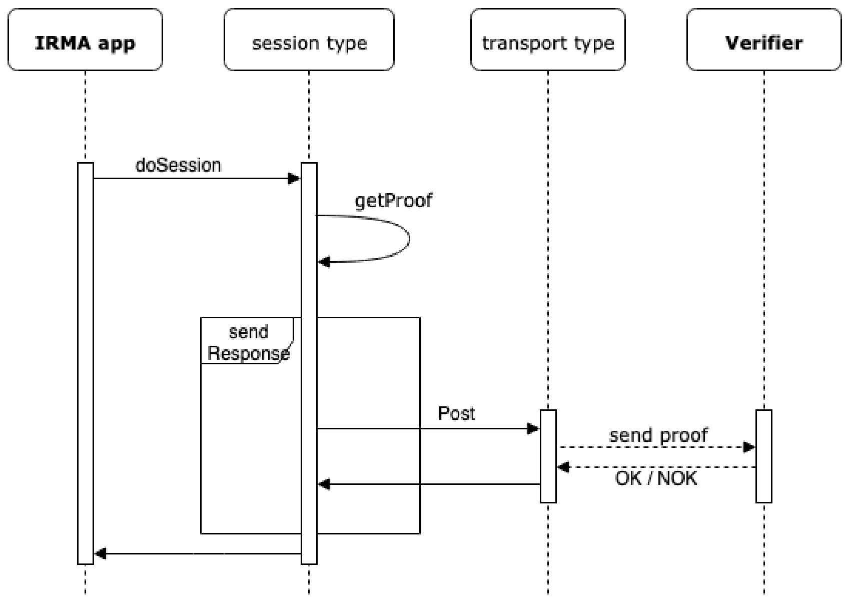


Figure 6.1: IRMA code: compute and send proof flow

indicate an IRMA message within a disclosing session. The *type* property will consist of *VerifiablePresentation* only. The ProofList object can be mapped to the proof property of the verifiable presentation as it is the proof that proves the knowledge of the hidden attributes, the validity of the credentials, and the fact that all the credentials in this proofs were issued to the same user.

6.5. OBTAIN METADATA

6.5.1. CURRENT IRMA IMPLEMENTATION

In section 4.1.3, we describe how the schememanager works and how IRMA organizes schemes. IRMA components regularly check if schemes are updated, and consequently, download updates if a newer version if available.

6.5.2. MODIFIED IRMA IMPLEMENTATION

In the previous chapter, we introduce a *metadata* server to make relevant scheme information available to external parties by dereferencing URIs.

As *irmago* already implements a framework for running a HTTP daemon and subsequently, handling HTTP requests, we decide to add the *metadata* server to the *irmago* component. We can extend the *irma server* command to run a *metadata* daemon, which is started via the command *irma server metadata*. Hence, we can deploy the metadata server independently of other components. To achieve this, we create a new *cobra.Command* and add it to the *RootCommand* via *AddCommand*. Within the *Server* struct type, we add a new method, *startMetadataServer*, that starts the server instance, whereby this server uses another port than the default IRMA server. The default port is 8089; however, administrators can set in the server configuration another port via the property *metadataport*.

A new HTTP Handler, created within *VCHandler()* by a new Chi router instance, can handle the requests. We mount an HTTP GET endpoint for each metadata type (issuer and schema) within the handler. Subsequently, the method *s.irmaserv.VCHandler* is invoked,

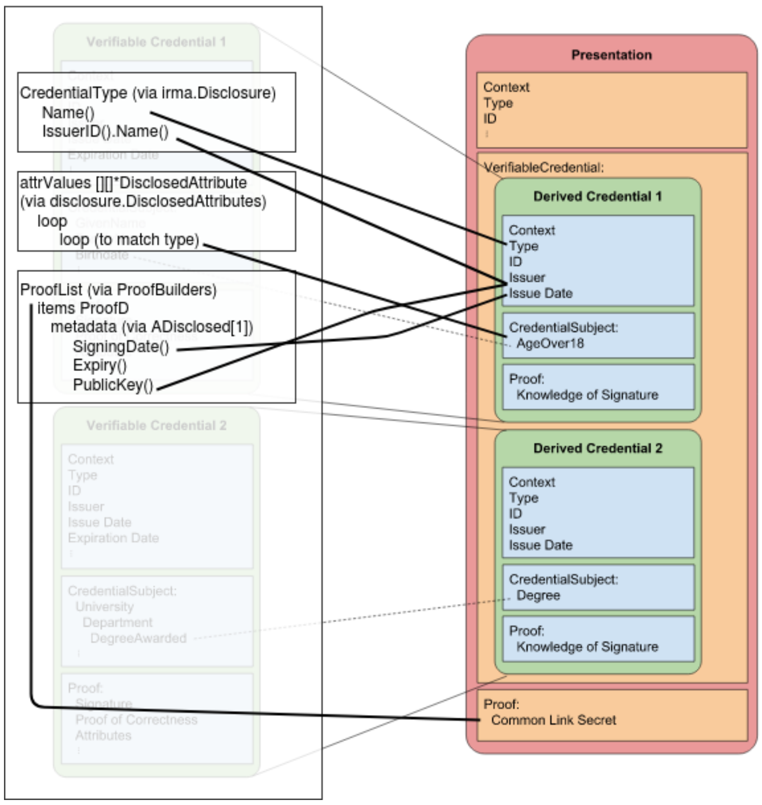


Figure 6.2: Mapping from IRMA data model to ZKP presentation

which can handle both types of requests.

Both the issuer and scheme URLs contain the name of the schema. Also, it contains the name of the credential type, as the credential type refers to one issuer. The issuer URL ends with the public key counter to indicate which version of the public key the issuer used to sign the credential.

In the following list, we show examples on how to retrieve information from the two endpoints:

- Issuer: *http://localhost:8089/issuer/irma-demo/MijnOverheid/2*
- Schema: *http://localhost:8089/schema/irma-demo/MijnOverheid*

As the PbdF is maintaining the *pbdf* production scheme, the PbdF shall also be responsible for resolving metadata requests related to the *pdbf* scheme. Each other party maintaining a schema must run a metadata server.

6.6. COMPUTE COMMITMENT

6.6.1. CURRENT IRMA IMPLEMENTATION

Within the *getProof* method within the *session* stuc type, *irmago* returns the commitment, which is called “secretkey-knowledge proof” within IRMA. The message itself is computed within the method *IssueCommitments* within the client package.

6.6.2. MODIFIED IRMA IMPLEMENTATION

After the app determines that a user established a session with a non-IRMA issuer, it computes a VC-compliant commitment message. We can use the *getProof* method for adding the logic. Then, a new method is added, called *IssueCommitmentsVC*, that returns the VC-compliant commitment.

6.7. DEMONSTRATION

After implementing the modifications, we demonstrate that the modifications work as intended. Since we were not able to connect to another system, we decide to adapt IRMA to run in a simulation mode. Simulation mode means that IRMA components exchange VC-compliant messages with each other. In what follows, we describe how we adapt IRMA to enable the simulation mode for both the issuing and disclosing session types.

First, for modifications concerning issuing, we introduce a flag, *IssueVC*, to switch to "VC-mode" within *irmago*. The VC-mode entails that in an issuing session established with an IRMA server, the IRMA app computes a VC-commitment message and sends it to the server. The *handleProtocolMessage* method can determine if an IRMA or VC commitment is sent. Then, the server extracts the *legacy* IRMA commitment signature embedded within the VC commitment, and consequently continues as usual.

Second, to validate the modifications concerning disclosing, we add a new QRCode struct. We call it *QRSovrin* for demonstration purposes. An example of that QRCode is as follows: *{"url":"http://192.168.2.100:8088/irma/f6JOn5NZqheTspDFISzW","action":"disclosing", "system":"sovrin"}*. When *irmago* maps the QRCode to the QRSovrin struct, it subsequently maps the *URL* to *qr.URL*, and the *type* to *qr.Type*. Then, we extend the *newQRSession* with a flag to indicate if the request is external or not. If it is an external request, *irmago* computes and sends a verifiable presentation to the verifier. Consequently, the server can unmarshal the message to a verifiable presentation, extract the proof, and verify it.

In appendix E, we show step-by-step how to set up and run IRMA in "VC-simulation" mode. In the issuing session, it results in that our local IRMA server in the role of issuer computes an *ageLower* VC. Via the JSON-schema, we can verify the content of the VC. In the disclosing session, it results in a verifiable presentation. The presentation contains a disclosed ageLower.over21 attribute.

6.8. SOURCE CODE

We forked two repositories of IRMA in which we pushed the modifications described in this chapter:

irmago `https://github.com/Iso5786/irmago/tree/daniel`

irma_mobile `https://github.com/Iso5786/irma_mobile/tree/daniel`

7 Related work

This chapter surveys previous work related to identity management, focusing on the topic of interoperability between IMSs. Therefore, we first summarize the projects ABC4Trust and FIDIS, both heavily funded by the EU. They both aim to advance identity management. The W3C working group published the final recommendation version of the VC specification at the end of 2019. Hence, it is hard to find work that addresses the same central problem as this research does. At least, we introduce a thesis about SolidVC, which is a decentralized framework based on VC. Then, we take these three works and relate the goals and deliverables with IRMA, VC, and our research. Lastly, based on the requirements identified by the FIDIS project, we analyze how the other related work projects contribute to them.

The FIDIS project, established in the early 2000s, had the vision to explore "the relationship between identification and identity in a high-tech environment; and implications for the workings of democracy and rule of law in the European Area of Freedom, Security and Justice (Art. 29 Treaty of the European Union)".[1] As a result, it delivered "integrated approaches to research", "legal, socio-economic, usability and application requirements", and "public architecture & specifications".

Interestingly, one of the research topics is "interoperability of identities and identity management systems". Researchers published, amongst others, documents about approaches on interoperability. Subsequently, they published a set of requirements for interoperability of IMSs [5] based on interviews with experts from the private and public sectors. One of the results is figure 7.1, which visualizes how they group interoperability of IMSs into three themes. They conclude that developing a standard is but a first step in the process of interoperation. Additional work lies in harmonizing legislative and policy rules. Those rules describe how personal information and identity management are defined, processed, and exchanged. For instance, there needs to be legal clarification on the reuse of issued credentials outside of the country in which they were issued. Also, at the social and cultural level, we need to understand the different contexts in which IMSs are built and used. They argue that all those points need to be understood and addressed before we can progress towards full interoperability.

Bakchouse [5] states that "in some contexts interoperability is seen as the enemy of privacy. Indeed, lack of interoperability may be seen as a bulwark against intrusion into the

[1] FIDIS official website: `http://www.fidis.net/about/`

privacy of personal information. Privacy activists take comfort from the fact that different IMS may not be able to exchange identity information”. Hence, the SE-community needs to communicate the benefits of the decentral nature of user-centric systems to citizens. The above quote does not hold for such systems, as users become the data controller. Only then citizens realize that interoperability is not the enemy, but their friend in this context.

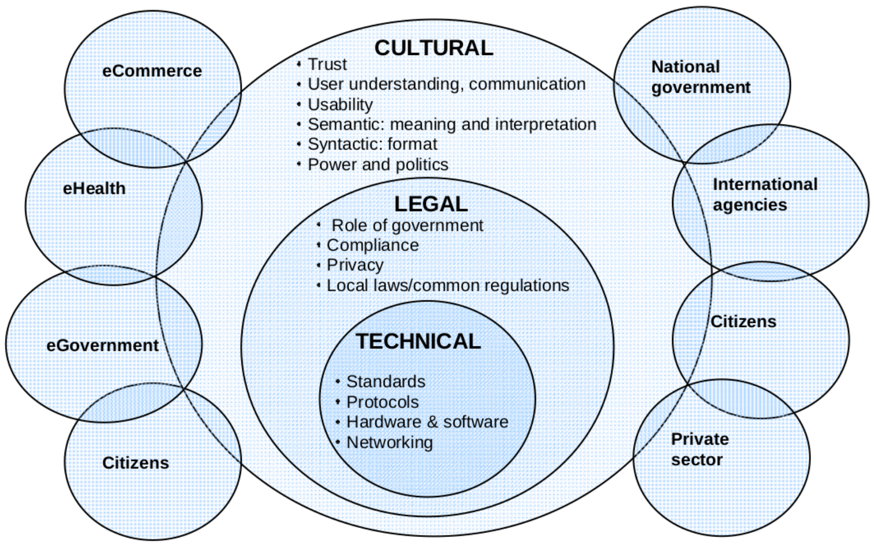


Figure 7.1: FIDIS interoperability themes grouped by perspectives (copy from [5])

In 2010, the EU largely funded a research and development project, ABC4Trust [35]. It intends to advance ABC-technologies. One of the deliverables is an extensive architectural document, containing features and concepts of Privacy-ABCs, architecture, a protocol specification, an API, and a crypto architecture. They define Privacy-ABCs as allowing holders to derive new tokens. Consequently, it protects the privacy of the user. The goal of the architecture is the definition of a common unified architecture to enable different ABC systems to interchange messages. It results in that users can receive credentials from different systems. Subsequently, users can use them on the same hardware and software without noticing differences. Additionally, requesting parties, such as issuers and verifiers, can choose which system they want to adopt if those systems implement the protocol.

The architecture itself is using a layered approach to separate responsibilities. The advantage is that the API and protocol can exist independently from algorithms or cryptography used, and are technology agnostic. Developers can utilize the API to connect their applications to different ABC engines. The protocol provides a specification for data artifacts, similar to what the VC specification provides. The specification itself is described in XML, but it is possible to utilize different syntaxes such as JSON. The cryptographic engine is delivered as a separate library. Implementers can use them without the engine due to the layered approach. To sum up, in contrast to the VC specification, it also delivers a protocol and an API specification.

We can only assume that both IRMA and VC got inspired by several concepts described in the architecture document. In particular, IRMA also implements features such as key binding, commitment scheme, and blind signature schemes (also called CLSignature).

SolidVC promotes itself as "a decentralized framework for Verifiable Credentials on the web" [15]. SolidVC is built on top of Solid (standing for Social Linked Data), which Tim Berners-Lee initiated. He has the vision to decentralize the web, whereby Solid can contribute to it. Solid promises that people regain power over their data. It is a web-based framework of decentralized applications. It provides users with services to manage their data, which can be everything from e-mails to videos. As the name already suggests, Solid uses the concept of linked data.

SolidVC uses the openness of Solid, employing its technology stack, and combine it with VC. Consequently, it delivers a decentralized framework for VCs. The framework does not rely on blockchain technology. SolidVC utilizes JSON-LD signatures to implement digital signatures. [2] According to Ezike [15], due to the usage of open web protocols and specifications, SolidVC is interoperable. However, as SolidVC is dependent on the Solid framework, other VC systems need to incorporate parts of Solid to connect with SolidVC.

In table 7.1, we relate the goals and deliverables of the three projects, with the goals and deliverables IRMA, VC, and this research. Both FIDIS and ABC4Trust elaborate on the interoperability of IdMs. The significant difference is that FIDIS describes on a higher level which challenges exist and how to tackle them. ABC4Trust proposes solutions to challenges, in particular, by providing an open protocol and API specification. As the name already suggests, ABC4Trust uses the concept of ABC. Subsequently, it tries to advance ABC-technology by defining an architecture and delivering open reference implementation. IRMA uses some of the concepts defined by ABC4Trust. The main goal of the VC standard is to advance interoperability between user-centric systems by delivering an unambiguous specification. SolidVC uses the VC specification to build a decentralized VC-platform. Within our work, we research on the relation between IRMA and VC by comparing and contrasting the goals and concepts of both. Consequently, we build an "IRMA VC prototype" to show how to extend IRMAs interoperability. Additionally, we provide recommendations for both the PbDF and W3C.

Additionally, in table 7.2, we take the main results from expert interviews identified by the FIDIS project [4]. Then, we explain how the other related works, including this one, can contribute to those results.

To sum up, we see that the FIDIS project investigated which challenges exist to advance the interoperability of IMSs. Also, it provides us with a set of requirements to help to tackle those challenges. The ABC4Trust project delivers a unified architecture for ABC systems, including a specification and protocol. SolidVC shows how to implement the VC specification to build a decentralized user-centric system without having to employ blockchain technology. We show in this research how we can modify an existing identity system, IRMA, to extend its interoperability. Hence from a technological perspective, we can argue that many issues are tackled. However, the industry needs to discuss and apply solutions more broadly. Still, from the legal and cultural perspectives, many challenges exist to enable interoperability, which needs more research in the future. At least new legislation within the EU, in particular the GDPR, might indicate that legislators actively work on the legal and cultural issues. Also, if we as a society want to see improvements in the field of identities and identity management, the society itself needs to invest more time and resources to foster the discussion, and subsequently, advance the field.

[2] See following repo for the implementation: `https://github.com/digitalbazaar/jsonld-signatures`

Table 7.1: Goals and deliverables of related and this work

Project	**Start year**	**Objectives**	**Deliverables**
FIDIS	2004	Explore relationship between identity and identification; research about interoperability of identity and identification concepts.	Legal, socio-economic, usability and application requirements; public architecture & specifications.
ABC4Trust	2010	Define a common, unified architecture for ABC systems, Delivering open reference implementations.	ABC architecture, Open reference implementation.
IRMA	2015	Make user-centric IMS more practical.	Practical, user-centric and privact-enhancing IMS.
VC	2017	Promote interoperability to prevent vendor lock-in, and seamless integration of different systems.	Unambiguous specification including a data model and syntax.
SolidVC	2019	Research about decentralized VC platform.	Decentralized VC platform built with the open protocols of the Web, master thesis.
IRMA VC	2019	Research about relation between IRMA and VC.	Compare and contrast the two, build an “IRMA VC-prototype” to extend IRMA’s interoperability, provide recommendations for PbDF and W3C.

Table 7.2: FIDIS themes and related work contributions

Topic	**Contribution**
Semantic issues of meaning and interpretation must be clear and unambiguous.	Both ABC4Trust and VC contribute to this point by delivering a specification.
Ability to communicate to the population about how the system works and what its benefits are.	Both private and public sectors need to contribute to this point. IRMA tries to show via different use-cases what benefits there are by using such a system. SolidVC shows how other decentralized VC platforms can be build.
One of the main requirements is the privacy of personal information and compliance with data protection legislation.	Several legislations within the EU in recent years, such as GDPR, have the objective to improve the privacy of its citizens. As one crucial point within GDPR is the “right to data portability”, according to [13], “it can be the opportunity to foster interoperability of services”.
Governments need to play a crucial role in establishing interoperability technology standards and laws.	The EU funded FIDIS and ABC4Trust primarily. Also, the EU funds the W3C; see the W3C funding website: `https://www.w3.org/Consortium/nmfunds/`.
Usability is a vital factor.	Projects such as IRMA can play a crucial role, as they focus a lot on usability.
More investigation needed for the importance of technology.	IRMA and SolidVC can contribute due to its open nature. In our research we show how we can modify IRMA to extend its interoperability.

8

Discussion, limitations, and recommendations

In this chapter, we discuss the findings of our research. Then, we elaborate on the limitations of this research. Also, for both the PbDF and the W3C, we list recommendations. For the PbDF, we give recommendations for possible future work. For the W3C, we give ideas on how to improve their process and communication.

8.1. Discussion

In what follows, we place different discussion aspects in separate subsections. First, we elaborate on how the results extend IRMA's interoperability. Second, we discuss what it means to make IRMA compliant with the VC standard and its impact. Third, we elaborate on the activities needed when connecting with another VC-compliant system. Fourth, we debate about the non-technical challenges. There, we introduce the idea of a chicken-egg problem between the industry, legislators, and the public. Fifth, we list the costs and benefits for the PbDF when making IRMA compliant with the VC standard. At last, we advise the PbDF on how to move forward.

8.1.1. Extending IRMA's interoperability

The goal of a standard such as the VC is to advance interoperability; in this particular case, interoperability between user-centric, decentralized IMS. After conforming to the VC standard, the results indicate that IRMA extends its interoperability. The same structure for data is used (syntax), and the meaning of data is unambiguously defined (semantics). It results in reaching level three of LCIM 3.3.

We can even reason about why IRMA does not reach level four of LCIM. According to Wang et al. [38], to reach the fourth level of LCIM, "a method for sharing meaning of terms and methods for anticipating context are required". Taxonomies, ontology, and UML artifacts are approaches for defining such a method. The VC data model defines taxonomies by defining the different concepts and describing the relationship between them, for instance, by describing the relationship between a verifiable presentation, containing one or more credentials, which each contain one or more credential subjects. Recently, Feilmayr and Wöß [16] refined the definition of an ontology as follows: "An ontology is a formal, explicit specification of a shared conceptualization that is characterized by high semantic expressiveness required for increased complexity." We can argue that the VC specification is precisely this, as it defines concepts that are shared between systems conforming to the VC standard. IRMA itself offers extensive documentation about its protocol and APIs supported by UML diagrams.

8.1.2. Make IRMA compliant and its impact

After making IRMA compliant with the VC standard, the IRMA system does not improve other features, for instance, increasing the privacy of its users or security of the IRMA system. Both IRMA and VC base their concepts on user-centric, decentralized IdM. In particular, we see that IRMA already employs the core concepts of VC. The flow of information between roles in VC supports the flow of information in IRMA.

The result of developing the prototype may indicate that modifying the IRMA data model to comply with the VC data model might be manageable. The main concepts of IRMA and VC are similar, as we show in figure 4.12. Also, IRMA already employs the JSON format, which makes it easy to express VC-compliant messages with IRMA.

Therefore, the prototype might suggest that it is straightforward to exchange and process VC-compliant messages from other systems. However, the VC specification does not provide us with a definition of protocols and APIs. Also, other IMS can use different technologies. This does it not make trivial to implement the modifications that are needed when connecting with another IMS that comply with the VC standard.

Nevertheless, the VC standard introduces the goal of *permissionless innovation*. With *permissionless innovation*, systems can extend the data model in various ways. The benefit of being extensible is that systems independently can innovate and improve their solutions. Then, they can share those solutions transparently with the industry. Consequently, companies might learn from each other. Learning from each other might lead to identifying best practices, thereby improving together along the way. Additionally, extensibility allows systems such as IRMA to comply with the VC standard later without the need to change their employed protocol and technologies.

The results of the SACCS method show that the impact on the architecture of IRMA is significant when implementing the modifications. If the PbDF decides to employ a *metadata server* for their official PbdF scheme, the PbDF needs to take additional measures to protect integrity (for instance, via hashlinks) and availability (i.E., protection against DoS attacks) of the metadata. Additionally, the PbDF should communicate to other parties what best practices are when employing a metadata server.

With IRMA, the security properties guarantee a high level of privacy for individuals. Also, it protects against abuse. After making IRMA compliant with VC, we are most confident that all security properties of IRMA still hold. This is primarily due to VC's decentral architecture, making identifiers optional, and supporting ZKP. However, when the PbDF decides to connect with another VC-compliant system, it depends on that system, how well the security properties hold. In particular, it depends on which protocol and technology the system employs, as VC leaves implementers much freedom. Hence, the PbDF needs to analyze the impact on the architecture on a case-by-case basis again.

8.1.3. Connecting to a VC-compliant system

If IRMA decides to connect to a VC-compliant system, we list in what follows the different activities the PbDF should conduct:

Describe required modifications Modifications 1, 3, 7, 9, and 10 need to be described in detail.

Conduct architectural impact analysis The PbDF should conduct a more comprehensive architectural impact analysis based on the modifications.

Decide The PbDF needs to decide if it accepts the modifications and their impact on the architecture.

Modify IRMA The PbDF has to implement the modifications within IRMA.

Set up agreement Also, the PbDF shall set up an agreement with the organization responsible for the other system. Such an agreement can contain information on how to handle performance aspects or security properties affected.

As we can see in the implementation report of VC, other systems already comply with the VC standard.[1] However, we found no evidence that systems use the extended interoperability to exchange messages with each other.

8.1.4. NON-TECHNICAL CHALLENGES

As identified in the previous chapter, challenges also exist in non-technical areas. Legislators and society need to discuss the legal and cultural challenges more extensively.

One of the more significant issues is the management of trust between parties and systems. Trust is essential for each IMS, as without the trust of its users, no system has a future. IRMA introduces the role of a *scheme manager*, which all other parties need to trust that participate in the related scheme, for managing the public keys of issuers and definitions of credentials correctly. Hence, it is a centralized, trusted third party. New systems, for instance, Sovrin, employ blockchain technology to decentralize this responsibility. The advantage is that issuers can manage their metadata, credential definitions, and public keys without having to trust a centralized party. However, blockchain-based IMSs still need a governance structure that makes the system trustworthy to users. Hence, decentralizing responsibilities does not magically solve trust issues.

Also, as stated in the previous chapter, legal issues exist when a user wants to use her credentials in environments with different laws and regulations. For instance, imagine a user travels abroad to the US and wants to use her IRMA ageLower credential to prove in an American store that she is above 21 years old to buy beer. Is the store allowed to accept the IRMA ageLower credential?

It is also possible to see the goal of extending user-centric, decentralized IMS's interoperability as a chicken-egg problem. The industry waits for legislators to enact policies to drive adoption of the identity model, and also waits for the society to discuss opportunities and challenges. However, policymakers and society wait for the industry to show what is possible from a technology perspective. Consequently, both sides wait for each other to act first. If the PbDF takes the opportunity to act first, it can become a pioneer in this field within the Netherlands or even the EU. Becoming a pioneer might increase their visibility and impact.

[1]See the implementation report on `https://w3c.github.io/vc-test-suite/implementations/`

8.1.5. Costs and benefits for the PbDF

In table 8.1, we list costs and benefits for the PbDF if it makes IRMA compliant with the VC standard. The benefits are that we can extend IRMA's interoperability. Also, the PbDF can actively contribute to the discussion between industry, legislators, and citizens. By contributing, the PbDF can become a pioneer in the field. Consequently, it can build up additional knowledge from the field. The costs are that it makes IRMA more complex (e.g., due to the introduction of the metadata server), and thus, harder to maintain. Also, initially, the PbDF needs to invest in additional resources.

Table 8.1: Costs and benefits when IRMA adapts the VC standard

Benefits	**Costs**
Extend IRMA's interoperability	Makes IRMA more complex
Contribute to discussion	Makes IRMA more complex
Become pioneer in the field	Additional investment
Increase knowledge	

8.1.6. Advice for the PbDF

We consider our findings as valuable for the PbDF. The findings give insight into how to modify IRMA to comply with the VC standard, and which consequences it has. We advise the PbDF to invest resources to modify IRMA to become compliant with the VC standard. As software engineers, we learned that the later we *fix* something, the more *expensive* it becomes in the end. Also, we encourage the PbDF to take a leading role by explaining to legislators and citizens what the opportunities and challenges are by advancing interoperability. In the far future, however, this might result in fewer people and organizations using IRMA as there are interoperable alternatives. Those might work well together with other VC-compliant systems.

8.2. Limitations

We can not for sure say what the exact impact is on the IRMA codebase and architecture of the modifications. It is beyond the scope of this project to implement the VC data model fully. Also, due to time constraints, we were not able to exchange messages with another VC-compliant system. As it is not a trivial change, we cannot be precise about the impact as well.

As mentioned before, there are non-technical challenges to tackle if the industry wants to make VC a success. As those challenges need to involve legislators and society as a whole, more work needs to be conducted by non-technical experts in the future.

We tried to get in touch again with people from the VC working group after having an initial interview, but this was not successful. Also, asking on other communication channels such as GitHub for support was not fruitful after the working group finalized the specification. Hence, due to a lack of knowledge, for instance, we did not succeed in implementing a data encoding scheme.

Again, due to time constraints, we did not discuss our results with members of the PbDF. However, in the following section, we provide recommendations to both the PbDF and W3C.

8.3. RECOMMENDATIONS

8.3.1. POSSIBLE FUTURE WORK FOR THE PBDF

RESEARCH ABOUT DECENTRALIZING THE IRMA SCHEME

Currently, other systems employ blockchain technology to store metadata information.[2] Blockchain technology protects the integrity of the content.

Other technologies such as InterPlanetary File System (IPFS) might also be an alternative.[3] IPFS is a protocol for storing and sharing data in a distributed file system. It makes use of peer-to-peer networks.

Future research could examine the costs and benefits of those and other alternatives.

CONNECT WITH ANOTHER VC-COMPLIANT IMS

Processing messages from other systems is not trivial. Hence, we recommend taking this research as a foundation for future research aiming to connect to another VC-compliant IMS. Important to mention is that the organization developing and maintaining the VC-compliant IMS is willing to help. Otherwise, the risk of failure and frustration is high.

STUDY VC CONCEPTS NOT SUPPORTED BY IRMA

In section 4.3, we list advanced concepts of VC, which IRMA currently does not support. In particular, those are *status*, *refreshing*, *terms of use*, *evidence*, and *disputes*. Also, we did not succeed in implementing data encoding schemes. Hence, investigating in future work how those concepts fit into IRMA, can be valuable for the PbDF.

STUDY RELATED STANDARDS OR RFCS

The W3C and other organizations continue working on other standards or RFCs related to user-centric IdM, for instance, the previously mentioned "Web Authentication API". Additionally, the W3C is working on a standard, called "Credential Handler API".[4] It defines an API that can handle credential requests and credential storage.

Aries RFC 0036 "Issue Credential Protocol 1.0" describes a possible protocol to be used in an issuing session.[5] Aries RFC 0037 "Present Proof Protocol 1.0" describes a possible protocol to be used in a disclosing session.[6]

It might be insightful to compare and contrast those protocols and APIs with the protocols and APIs of IRMA.

STUDY NON-TECHNICAL CHALLENGES

A further research direction can be to identify the non-technical challenges further. Then, different advice or solutions can be identified, possibly together with other organizations or legislators. For instance, how GDPR can contribute to advance the concept of user-centric

[2]For IRMA, Timen Olthof developed a proof-of-concept to decentralize the scheme via the Ethereum blockchain, see `https://github.com/timenolthof/irmaethereumscheme`
[3]See official website under `https://ipfs.io/`
[4]See `https://w3c-ccg.github.io/credential-handler-api/`
[5]See repository on GitHub: `https://github.com/hyperledger/aries-rfcs/tree/master/features/0036-issue-credential`
[6]See repository on GitHub: `https://github.com/hyperledger/aries-rfcs/blob/master/features/0037-present-proof/`

IMS. Alternatively, one could also think about developing a governance framework, which can be employed across user-centric IMS.

8.3.2. IMPROVEMENT IDEAS FOR THE W3C

GIVE INTEGRATED EXAMPLES

To understand concepts more clearly, it helps to have good examples. Hence, a more integrated example of how to employ VC can be helpful for future implementers. Currently, most examples within the VC data model are independent. For instance, it could help to understand the concept of the *data encoding schema* better.

ELABORATE MORE ON PERMISSIONLESS INNOVATION

The goal of *permissionless innovation* is described very briefly in the specification. If a system becomes more extensible, the more complicated it might become. Thus, it is harder to maintain. Hence, more elaboration on extensibility and permissionless innovation might help future implementers.

RELATE WITH OTHER WORK

For others, it might be useful to have an overview that brings different standards and initiatives from the W3C together, which may make it more accessible for researchers and engineers. For identity-related work, for instance, the "Credentials Community Group" might be a good place.[7]

[7] See https://www.w3.org/community/credentials/

9 Conclusion

In this research, we explored the relation between IRMA and VC. We revealed that the goal of the VC standard is to advance interoperability between user-centric, decentral IMSs. We conclude that by making IRMA comply with the VC standard, IRMA at least reaches level three of LCIM, possibly even level four. Consequently, IRMA extends its interoperability. Then, if other IMSs comply with the VC standard and implement IRMAs protocol, users of the IRMA app can use their attributes outside of the IRMA system. Also, IRMA issuers can issue attributes to non-IRMA users. By making IRMA comply with the VC standard based on the modifications we described, we can also conclude that IRMA does not enhance other features, such as increase the privacy of its users.

After comparing and contrasting IRMA and VC, we discovered that roles, key concepts such as claims, credentials, and presentations overlap between IRMA and VC. Also, VC promotes the concept of ZKPs, and IRMA employs it. VC supports the JSON syntax, and IRMA uses it. The developed prototype shows that it might be manageable to adapt IRMA to comply with the VC-standard.

It is one thing to modify IRMA to comply with the VC standard. It is another, however, to achieve interoperability between different VC-conforming IMSs due to several challenges. One is that other VC-compliant systems can employ different (cryptographic) protocols, APIs, and technologies. This can have a profound impact on IRMA's security properties. Another challenge is legal and cultural issues. We need to identify and discuss them more broadly with legislators and society as a whole.

We advised the PbDF to modify IRMA to become compliant with the VC standard. If the PbDF invests now in interoperability for decentral, user-centric IMSs, it might become a pioneer in the field. Consequently, it might help to reduce the resistance of individuals to use those systems. Also, as the public understands the advantages better, it can even increase the acceptance of the public for those systems. This might lead to more investments. However, the costs for the PbDF are initial investments and increased complexity of IRMA.

We decided within the scope of this research to not modify the IRMA data model. Also, we did not succeed in establishing an active collaboration with another project. This could have enabled us to show how to exchange messages with another VC-compliant system.

Therefore, we proposed several recommendations for future work for the PbDF. First, examine alternatives to decentralize the IRMA scheme. Second, research about other VC-compliant IMS, with which IRMA can connect. Third, study concepts of VC that IRMA currently does not support. Fourth, study related standards and RFCs. At last, study legal and social challenges, and subsequently, provide advice or solutions.

Also, we shared ideas for improvement with the W3C. In particular, W3C can give integrated examples to understand certain concepts better. Also, elaborate more on the impact of permissionless innovation. Lastly, it can help implementers and researchers to relate work, such as VC, with other initiatives in the same field.

In general, within the field of identity management, and in particular, within decentral, user-centric IdM, we are missing a unified approach to interoperability by the industry and legislators. In the end, it would benefit all citizens if the industry builds interoperable, decentralized, and user-centric solutions. Hence, institutions (especially the EU) and the industry together should invest more resources to advance the field. They can take existing work (for instance, FIDIS and ABC4Trust) and new technological developments to build up new knowledge subsequently. At least, due to standards such as VC, new concepts such as SSI, and legislations such as GDPR, we see some progress within the field.

BIBLIOGRAPHY

LITERATURE

[1] Heather Adkins, Betsy Beyer, Paul Blankinship, Piotr Lewandowski, Ana Oprea, and Adam Stubblefield. *Building Secure and Reliable Systems: Best Practices for Designing, Implementing, and Maintaining Systems.* 1 edition. O'Reilly Media, Mar. 31, 2020. 558 pp. ISBN: 978-1-4920-8312-2.

[2] Greg Alpar, Fabian van den Broek, Brinda Hampiholi, Bart Jacobs, Wouter Lueks, and Sietse Ringers. "IRMA: practical, decentralized and privacy-friendly identity management using smartphones". In: HotPETs. 2017.

[3] Greg Alpar and Bart Jacobs. "Credential design in attribute-based identity management". In: *R. Leenes.* TILTing Perspectives (2013).

[4] James Backhouse. *D4. 2: Set of requirements for interoperability of Identity Management Systems.* Publisher: Citeseer. 2005.

[5] James Backhouse. "Interoperability of identity and identity management systems". In: *Datenschutz und Datensicherheit-DuD* 30.9 (2006). Publisher: Springer, pp. 568–570.

[6] Patrik Bichsel, Jan Camenisch, Maria Dubovitskaya, R Enderlein, Stephan Krenn, Ioannis Krontiris, Anja Lehmann, Gregory Neven, Janus Dam Nielsen, Christian Paquin, et al. "D2. 2 Architecture for attribute-based credential technologies-final version". In: *ABC4TRUST project deliverable* (2014).

[7] Lisa L Brownsword, David J Carney, David Fisher, Grace Lewis, and Craig Meyers. *Current perspectives on interoperability.* CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 2004.

[8] Jan Camenisch and Anna Lysyanskaya. "A signature scheme with efficient protocols". In: *International Conference on Security in Communication Networks.* Springer, 2002, pp. 268–289.

[9] David Chaum. "Security without identification: Transaction systems to make big brother obsolete". In: *Communications of the ACM* 28.10 (1985), pp. 1030–1044.

[10] Soon-Yong Choi and Andrew B Whinston. "Benefits and requirements for interoperability in the electronic marketplace". In: *Technology in society* 22.1 (2000). Publisher: Elsevier, pp. 33–44.

[11] Konstantinos Christidis and Michael Devetsikiotis. "Blockchains and smart contracts for the internet of things". In: *IEEE Access* 4 (2016). Publisher: IEEE, pp. 2292–2303.

[12] Sebastian Clauß and Marit Köhntopp. "Identity management and its support of multilateral security". In: *Computer Networks* 37.2 (2001). Publisher: Elsevier B.V, pp. 205–219.

[13] Paul De Hert, Vagelis Papakonstantinou, Gianclaudio Malgieri, Laurent Beslay, and Ignacio Sanchez. "The right to data portability in the GDPR: Towards user-centric interoperability of digital services". In: *Computer Law & Security Review* 34.2 (Apr. 1, 2018), pp. 193–203.

[14] Mayuri Dhamdhere and Sridevi Karande. "Identity and access management: concept, challenges, solutions". In: *International Journal of Latest Trends in Engineering and Technology* Volume 8 (Issue 1).

[15] Kayode Yadilichi Ezike. "SolidVC: a decentralized framework for Verifiable Credentials on the web". Master Thesis. Massachusetts Institute of Technology, 2019.

[16] Christina Feilmayr and Wolfram Wöß. "An analysis of ontologies and their success factors for application to business". In: *Data & Knowledge Engineering* 101 (Jan. 1, 2016), pp. 1–23.

[17] NIST GCR. "Cost analysis of inadequate interoperability in the US capital facilities industry". In: *National Institute of Standards and Technology (NIST)* (2004), pp. 223–253.

[18] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. "The knowledge complexity of interactive proof systems". In: *SIAM Journal on computing* 18.1 (1989). Publisher: SIAM, pp. 186–208.

[19] Audun Jøsang and Simon Pope. "User centric identity management". In: *AusCERT Asia Pacific information technology security conference*. Citeseer, 2005, p. 77.

[20] Hee-Woong Kim and Atreyi Kankanhalli. "Investigating User Resistance to Information Systems Implementation: A Status Quo Bias Perspective". In: *MIS Quarterly* 33.3 (2009), pp. 567–582.

[21] Romain Laborde, Arnaud Oglaza, Samer Wazan, François Barrere, Abdelmalek Benzekri, David W Chadwick, and Rémi Venant. "A User-Centric Identity Management Framework based on the W3C Verifiable Credentials and the FIDO Universal Authentication Framework". In: *2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2020, pp. 1–8.

[22] Meir M Lehman. "Programs, life cycles, and laws of software evolution". In: *Proceedings of the IEEE* 68.9 (1980), pp. 1060–1076.

[23] Grace A Lewis, Edwin Morris, Soumya Simanta, and Lutz Wrage. "Why standards are not enough to guarantee end-to-end interoperability". In: *Seventh International Conference on Composition-Based Software Systems (ICCBSS 2008)*. IEEE, 2008, pp. 164–173.

[24] Georgios Loukas and Gülay Öke. "Protection against denial of service attacks: A survey". In: *The Computer Journal* 53.7 (2010), pp. 1020–1037.

[25] Ruth Malan, Dana Bredemeyer, et al. "Functional requirements and use cases". In: *Bredemeyer Consulting* (2001).

[26] Gary McGraw. *Software Security: Building Security In*. Addison-Wesley Professional, 2006. ISBN: 0-321-35670-5.

[27] Alexander Mühle, Andreas Grüner, Tatiana Gayvoronskaya, and Christoph Meinel. "A survey on essential components of a self-sovereign identity". In: *Computer Science Review* 30 (2018), pp. 80–86.

[28] Satoshi Nakamoto et al. *Bitcoin: A peer-to-peer electronic cash system*. 2008.

[29] Justice Opara-Martins, Reza Sahandi, and Feng Tian. "Critical review of vendor lock-in and its impact on adoption of cloud computing". In: *International Conference on Information Society (i-Society 2014)*. IEEE, 2014, pp. 92–97.

[30] George Orwell and AM Heath. *Animal farm and 1984*. Houghton Mifflin Harcourt, 2003.

[31] Ken Peffers, Tuure Tuunanen, Marcus A Rothenberger, and Samir Chatterjee. "A design science research methodology for information systems research". In: *Journal of management information systems* 24.3 (2007), pp. 45–77.

[32] Felipe Pezoa, Juan L Reutter, Fernando Suarez, Martín Ugarte, and Domagoj Vrgoč. "Foundations of JSON schema". In: *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016, pp. 263–273.

[33] Andreas Pfitzmann and Marit Hansen. *A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management*. 2010.

[34] Jean-Jacques Quisquater, Myriam Quisquater, Muriel Quisquater, Michaël Quisquater, Louis Guillou, Marie Annick Guillou, Gaïd Guillou, Anna Guillou, Gwenolé Guillou, and Soazig Guillou. "How to Explain Zero-Knowledge Protocols to Your Children". In: *Advances in Cryptology — CRYPTO' 89 Proceedings*. Ed. by Gilles Brassard. Lecture Notes in Computer Science. Springer New York, 1990, pp. 628–631.

[35] Kai Rannenberg, Jan Camenisch, and Ahmad Sabouri. *Attribute-based Credentials for Trust: Identity in the Information Society*. Springer Publishing Company, Incorporated, 2014. ISBN: 3-319-14438-3 978-3-319-14438-2.

[36] Karen Scarfone, Wayne Jansen, and Miles Tracy. "Guide to general server security". In: *NIST Special Publication* 800 (s 123 2008).

[37] Andreas Tolk and James A Muguira. "The Levels of Conceptual Interoperability Model: Applying Systems Engineering Principles to M&S". In: *Proceedings of the 2003 fall simulation interoperability workshop*. Vol. 7. Citeseer, 2003, pp. 1–11.

[38] Wenguang Wang, Andreas Tolk, and Weiping Wang. "The Levels of Conceptual Interoperability Model: Applying Systems Engineering Principles to M&S". In: *CoRR* abs/0908.0191 (2009).

[39] Roel J Wieringa. *Design science methodology for information systems and software engineering*. Springer, 2014. ISBN: 978-3-662-52446-6.

[40] Byron J Williams and Jeffrey C Carver. "Characterizing software architecture changes: A systematic review". In: *Information and Software Technology* 52.1 (2010), pp. 31–51.

[41] Byron J Williams and Jeffrey C Carver. "Examination of the software architecture change characterization scheme using three empirical studies". In: *Empirical Software Engineering* 19.3 (2014), pp. 419–464.

[42] Yvonne Wilson and Abhishek Hingnikar. “Evolution of Identity”. In: *Solving Identity Management in Modern Applications*. Springer, 2019, pp. 19–28.

WEB RESOURCES

[43] Christopher Allen. *The Path to Self-Sovereign Identity*. The Path to Self-Sovereign Identity. Apr. 27, 2016. URL: https://github.com/ChristopherA/self-sovereign-identity/blob/master/ThePathToSelf-SovereignIdentity.md.

[44] Scott Bradner. *Key words for use in RFCs to Indicate Requirement Levels*. Library Catalog: tools.ietf.org. Mar. 1997. URL: https://tools.ietf.org/html/rfc2119.

[45] Vinton G. Cerf. *Opinion | Keep the Internet Open*. The New York Times. May 24, 2012. URL: https://www.nytimes.com/2012/05/25/opinion/keep-the-internet-open.html.

[46] *Identity Mixer, IBM Research Zurich*. May 9, 2018. URL: https://www.zurich.ibm.com/identity_mixer/.

[47] M Sporny, D Longley, and D Chadwick. *Verifiable Credentials Data Model 1.0*. Verifiable Credentials Data Model 1.0. Nov. 19, 2019. URL: https://www.w3.org/TR/vc-data-model/.

[48] Manu Sporny, Dave Longley, Gregg Kellogg, Markus Lanthaler, and Niklas Lindström. *JSON-LD 1.1*. W3C Candidate Recommendation. Apr. 17, 2020. URL: https://www.w3.org/TR/json-ld/.

A
Master thesis topic

Provided by Greg Alpar: “IRMA is a new kind of authentication method. It provides a new way to prove information about your identity to a website. Traditionally, authentication is a proof of a unique identifier. For instance, if a username is an identifier, the corresponding password is the proof that the username belongs to you. Similarly, if a citizen number (such as a BSN) is an identifier, a driving license, an identity card or a successful DigID authentication can be its proof.

In IRMA one uses attributes instead of identifiers. Attributes are properties or qualifications of a person. The fact that you are over 18 years of age or you are a resident of Belgium are attributes of yours. But because your name and your citizen number are also examples for attributes, attributes can be considered as a generalisation of identifiers. Attributes in IRMA are issued by authoritative parties and protected by advanced cryptographic techniques. As an example, each field in your passport is an attribute (name, passport number, nationality, etc.) which can be issued by some Dutch authority. You can read much more about the IRMA technology on the Privacy by Design Foundation’s website.

Since IRMA provides many powerful applications, it is ready to be rolled out in many contexts. The World Wide Web Consortium (W3C) works to develop Web standards. One of its recent proposals is the ‘Verifiable Claims Data Model and Representations’. Its aim is to define a “a standard way to express these sorts of claims on the Web in a way that is cryptographically secure, privacy respecting, and automatically verifiable”.

The goal of this project is to create IRMA Verifiable Claims. This entails a common understanding of the IRMA and the Verifiable Claims infrastructure and to develop a working IRMA implementation that complies with the W3C infrastructure.

(Remark: As the W3C proposal is in a draft phase, it accepts recommendations, which implies that there is also a possibility to influence the Consortium’s final output.) ”

B
IRMA PROTOCOL

To give the reader of a better understanding of the protocol implemented within IRMA, we describe for the issuance and disclosure process the interaction between actors and components.

B.1. ISSUANCE

In figure B.1 a sequence diagram shows how the IRMA issuance process works. The following text describes each step in more detail.

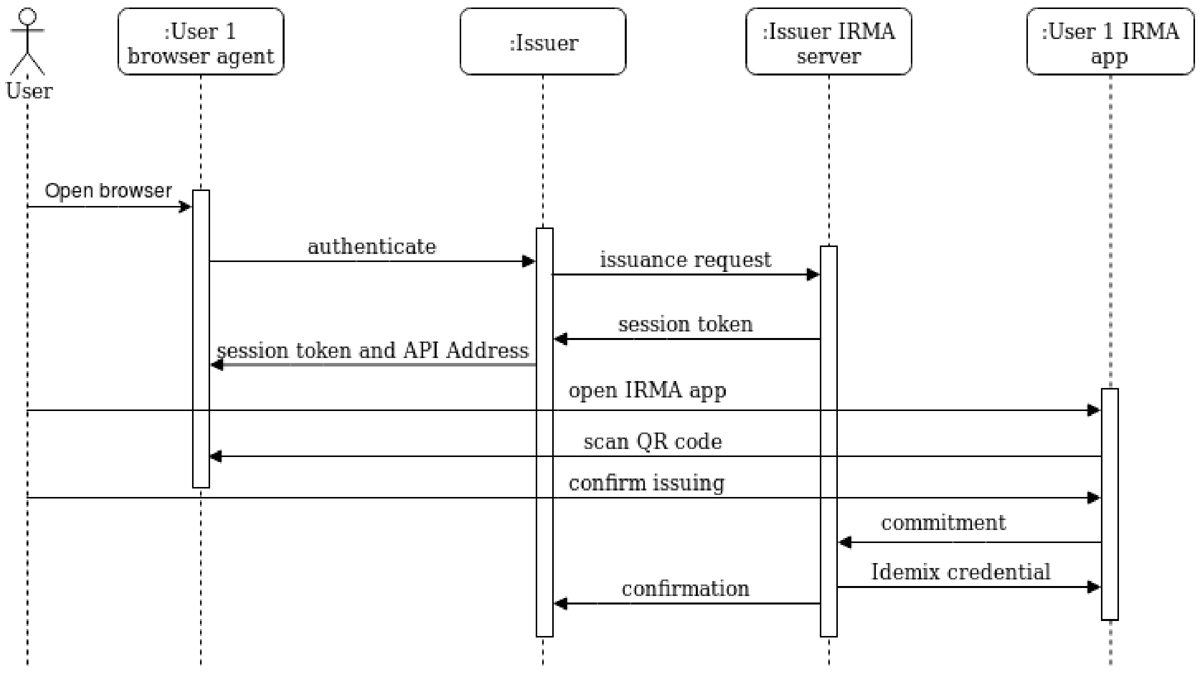


Figure B.1: IRMA issuance process

1. The user can choose to load attributes by opening IRMA's *attribute* issuance website;
2. After selecting an *issuer* the user will be redirected to the issuer's website;

3. The issuer informs the irma server that it wants to issue certain attributes (optionally, with certain values) to the user (see listing 4 for the related JSON request);

4. If the irma server validates the request successfully, it will return a session token (see listing 5 for the related response) which is then translated into an QRCode that the user needs to scan;

5. After the user scans the QRCode, the app sends the session token to the irma server (from IRMA server console: *method=GET type=client url=/irma/eFTj0ar7vEIagyrOLyGx/*) and the server sends the unsigned attributes to the IRMA app along with a nonce;

6. If the user agrees to receive this *credential*, first the client needs to compute a commitment which uses the earlier received nonce, and then sending the commitment to the server, see 6 .

7. If the server verifies the proofs, it returns a message back to the app containing the signature which proofs that the credentials are valid and can be used by the user in later disclosure sessions (see listing 7 for the message);

8. At last, a confirmation message is send to the user (see listing 8 for the actual JSON message).

B.2. DISCLOSURE

In figure B.2 a sequence diagram shows how the IRMA disclosure process works. The following text describes each step in more detail.

1. When the user decided to identify herself via IRMA to a *verifier*, the *verifier* sends a request to the IRMA server to start a disclosure session;

2. If the IRMA server accepts the request, it assigns a session token to it and returns the content of the QRCode that the verifying party shall display to the user. The QRCode contains the URL to itself and the session token.

3. Then, the user scans the QRCode which results in a GET request from the app to the */irma/* endpoint. The response contains the attributes that need to be disclosed (see listing 9);

4. The app let the user choose which *attributes* she wants to *disclose* ("selective disclosure");

5. The app generates the response based on the selected *attributes* by attaching a valid proof of knowledg (see listing 10 for the related response);

6. The *verifiers* IRMA server can verify the *attributes* by verifying the proof of knowledge and consequently, either grants or denies the access to the requested object.

The initiation of the session between verifier, irma server and the end user is omitted here as it is already explained in the above issuance section.

Listing 4 IRMA issuing request from issuer to IRMA server

```json
{
  "request": {
    "context": "AQ==",
    "nonce": "PPGKfVbsDZ/1ZST/qyGH0A==",
    "type": "issuing",
    "credentials": [
      {
        "validity": 1576949693,
        "keyCounter": 2,
        "credential": "irma-demo.MijnOverheid.ageLower",
        "attributes": {
          "over12": "yes",
          "over16": "yes",
          "over18": "yes",
          "over21": "no"
        }
      }
    ],
    "disclose": [
      {
        "label": "email",
        "attributes": [
          "pbdf.pbdf.mijnirma.email"
        ]
      }
    ]
  }
}
```

Listing 5 IRMA server issuing response to issuer

```json
{
  "sessionPtr": {
    "u": "http://192.168.2.100:8088/irma/hmHR4xe0g8eYtz0AnefV",
    "irmaqr": "issuing"
  },
  "token": "Z0cSKiq5MwvyvKdH0ftI"
}
```

Listing 6 IRMA Idemix client commitment message

```json
{
    "U": null,
    "n_2": "ol7Sus84sZWVafTQlSmphQ==",
    "combinedProofs": [
        {
            "U": "HKT53VzhcY7We6v7ltke0e0YWa",
            "c": "nOZv/vdR17+7/vmlGCLELwCYkS",
            "v_prime_response": "v62uVkohkTKkuVc",
            "s_response": "PvgCM7bVMW+Cp/5rnN",
        }
    ],
    "proofPJwt": "",
    "proofPJwts": null,
    "indices": []
}
```

Listing 7 IRMA Idemix issuance response containing signature

```json
[
  {
    "proof": {
      "c": "NKhxc5rxdgFbHJ3ZcfPVNn38wDfNej8i1G5fB7qQY/s=",
      "e_response": "IXP+3YquJFLg3hRD9vXQ6KzvQa6G9WODW9kV1HGE4...="
    },
    "signature": {
      "A": "kQg7Fd8cmALbejgPzvpL+hsGkd+9+uWqH3600euI1ib1x...",
      "e": "EAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA...",
      "v": "CWobkV5ecYptSkWJs63mSJPTfsc2XfS/EUDJdf3KngjmWU...",
      "KeyshareP": null
    }
  }
]
```

Listing 8 IRMA issuance confirmation

```
{
  "token": "ZOcSKiq5MwvyvKdHOftI",
  "status": "DONE",
  "type": "issuing",
  "proofStatus": "VALID",
  "disclosed": [
    {
      "rawvalue": "GGoFw1nWtSM",
      "value": {
        "": "GGoFw1nWtSM",
        "en": "GGoFw1nWtSM",
        "nl": "GGoFw1nWtSM"
      },
      "id": "pbdf.pbdf.mijnirma.email",
      "status": "PRESENT"
    }
  ]
}
```

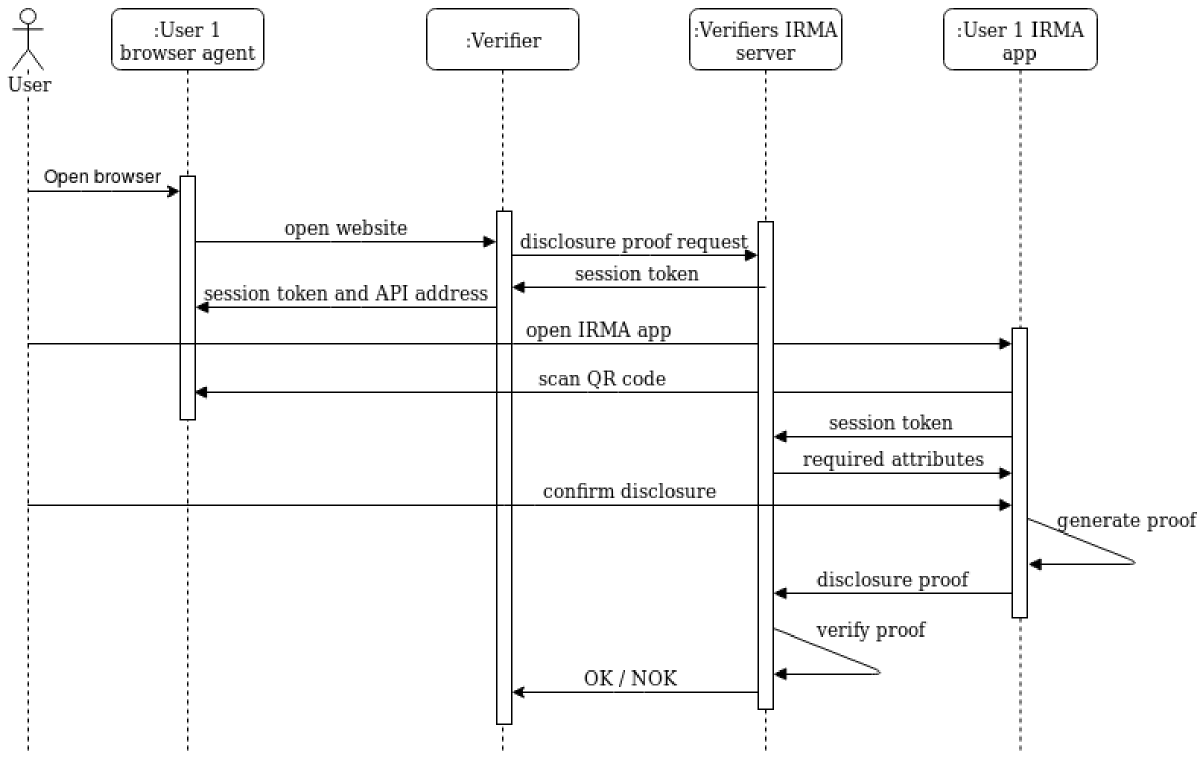


Figure B.2: IRMA disclosure process

Listing 9 IRMA disclosure request

```
{
  "request": {
    "context": "AQ==",
    "nonce": "V2TMsHgomvYhMifg1bijVA==",
    "type": "disclosing",
    "content": [
      {
        "label": "email",
        "attributes": [
          "pbdf.pbdf.mijnirma.email"
        ]
      }
    ]
  }
}
```

Listing 10 IRMA disclosure proof response

```
{
  "proofs": [
    {
      "c": "ZFNuGr3hrIa93jPITpP/CN0Tz5+MHDW1joQK60GQyN4=",
      "A": "LSACS3IP2Whx4ARk76wAMCtHtfG7ky7k0oku8bxyszmjExG...",
      "e_response": "X2Kk1vLkQi+scytAyIQtBMv7tm3BHWSgpy3as....",
      "v_response":
       ↪  "BH5iVXojM8bvb4lbIkUqm4z08yVkYDHRhvwsPMyBeTTVm...",
      "a_responses": {
        "0": "47CAsQMT+ahKzUFqYrHYVmpkVVsGxCevgzU7pBemg4DPC..."
      },
      "a_disclosed": {
        "1": "AwAJ+gIKAA0wAvzh3LqriPdt4p/zejln",
        "2": "jo7ej05i3K7opps="
      }
    }
  ],
  "indices": [
    [
      {
        "cred": 0,
        "attr": 2
      }
    ]
  ]
}
```

C
VC REQUIREMENTS

In this appendix, we list the absolute requirements and prohibitons of the VC data model.

In the following list we describe absolute requirements of a verifiable credential:

- *@context* property must be present and must be one or more URIs;
- *@context* first value must be *https://www.w3.org/2018/credentials/v1*;
- *@context* subsequent items can be objects that express context information;
- *type* property must be present and must be one or more URIs;
- *type* properties must contain at least *VerifiableCredential*;
- *credentialSubject* property must be present;
- *credentialSubject* is defined as a set of objects that contain one or more properties that are related to the subject of a VC;
- *issuer* property must be present;
- *issuer* property must be a single URI;
- *issuanceDate* property must be present;
- *issuanceDate* must be an ISO8601 datetime;
- *proof* property must be present;
- *proof* property must include specific method using the type property.

In the following list we describe absolute requirements of a verifiable presentation:

- *Presentations* must be of type VerifiablePresentation;
- *type* property must be present and must be one or more URIs;
- *type* properties must be at least *VerifiablePresentation*;
- *Presentations* must include verifiableCredential and proof.

In the following list we describe absolute requirements of a verifiable if that credential is used in ZKP systems:

- *credentialSchema* must be present;
- *proof* must be present.

In the following list we describe absolute requirements and prohibitons of a verifiable presentation when used in combination with ZKP:

- each *derived VC* must have a credentialSchema present;
- a *verifiable presentation* must not leak information to enable verifiers to correlate the holder across presentations;
- *proof* must be present to enable verifiers to assess that all derived VCs are from the same holder. be present.

D

SACCS CHARACTERISTICS

In this appendix chapter we describe certain SACCS charateristics. Without a description it is hard to understand them, and the given motivation.

D.1. Architecture characteristics

D.1.1. Logical characteristics

Layers Indicates the hierarchical layers of a system.

Inheritance Structure Parent-child-sibling relationship between modules. In *go* this would either be realized by implementing an interface, or by adding a struct within a struct.[1]

Module Decomposition The system at varying levels of abstraction, for example having modules for the UI, business logic and database operations.

Source Structure Related to the structure of source code, i.E. files and imports of files in other files.

Dependency Relationship Describes the system modules and relations between them.

D.1.2. Runtime characteristics

Control Flow Processing Interaction of system processes through a pipeline representation of the architecture.

Repository Access Database and accessor relationship of the system.

Concurrent Processes Interaction of processes as system threads.

Component Interaction Sharing of information between components.

Distributed Components Shows how remote processes interact via interfaces.

Component Deployment Shows components and their location on system hardware.

[1]See for a good explanation: `https://hackthology.com/object-oriented-inheritance-in-go.html`

D.2. Features

The descriptions are copied from Williams and Carver [40].

Devices Hardware devices used by the system.

Data Access Receipt of data from external systems or repositories.

Data Transfer Flow of data from system to external systems.

System Interface Software interfaces with external systems.

User Interface Human-computer interaction interfaces.

Communication Protocols used to interface other systems or data.

Computation Algorithm function and modification of data.

Input/Output Format of information processed by the system.

D.3. Quality attributes

The quality attributes are based on ISO 25010. Since ISO 25010 did not exist yet, when Williams and Carver [40] published their paper, we decided to use the quality attributes listed in ISO 25010.[2]

Functionality suitability Degree to which a system provides functions that meet needs of stakeholders. Sub-attributes are: completeness, correctness and appropriateness.

Performance efficiency Degree to which a system provides performance relative to amount of resources under certain conditions. Sub-attributes are time behaviour, resource utilization and capactiy.

Reliability Degree to which a system performs certain functions under specificed conddi-tions. Sub-attributes are maturity, availability, fault tolerance and recoverability.

Usability Degree to which users can recognize if a system is appropriate for their needs. Sub-attributes are learnability, operability, user error protection, UI aesthetics and accessibility.

Security Degree to which a system protects information and data. Sub-attributes are confidentiality, integrity, non-repudiation, accountability, confidentiality, authenticity.

Compatibility Degree to which a system can exchange information with other products. Sub-attributes are co-existence and interoperability.

Maintainability Degree of effectiveness and efficiency with which a system can be modified. Sub-attributes are modularity, reusability, analysability, modifiability, testability.

Transferability Degree to which a system can be transferred from one environment, or be replaced. Sub-attributes are portability, adaptability, installability, replaceability.

[2]Information gathered from the official ISO website: `https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:ed-1:v1:en`

E

IRMA VC-simulation

In this appendix we describe how to set up IRMA to run the prototype, and results of running the prototype. First, we explain how to prepare the irmago and irma_mobile repositories, and run the different components. Then, we show the results of the modifications in both an issuing and disclosing session.

E.1. Preparation

1. If you already have IRMA repositories in your *$GOPATH*, move them to a different location;
2. If not, you need to set up both IRMA components as described in the corresponding READMEs of IRMA, and then move them to a different location;
3. Clone both GitHub repositories listed in section 6.8 into your *$GOPATH*, and make sure to have the *daniel* branches checked out;
4. Install the irma binary by executing *go install ./irma* within the *irmago* project;
5. To start the metadata server, execute *irma server metadata -vv* (-vv foor additional logging);
6. Connect an Android phone via USB, and verify that the device is visible with the command *adb devices*;[1]
7. Go to the irma_mobile directory, and execute the commands *yarn install*, *dep ensure*, and *./irmago_copy_and_run* in that order.[2]

E.2. Issuing session

We already set the flag *IssueVC* to true within *irmago*, hence when set up correctly, the app shall communicate a commitment-VC.

Execute the following steps to retrieve a VC via an issuing session:

[1]As I only have an Android phone, I only validated the changes on the Android platform.
[2]Unfortunately, if you already have a newer IRMA app version installed, you need to remove it manually.

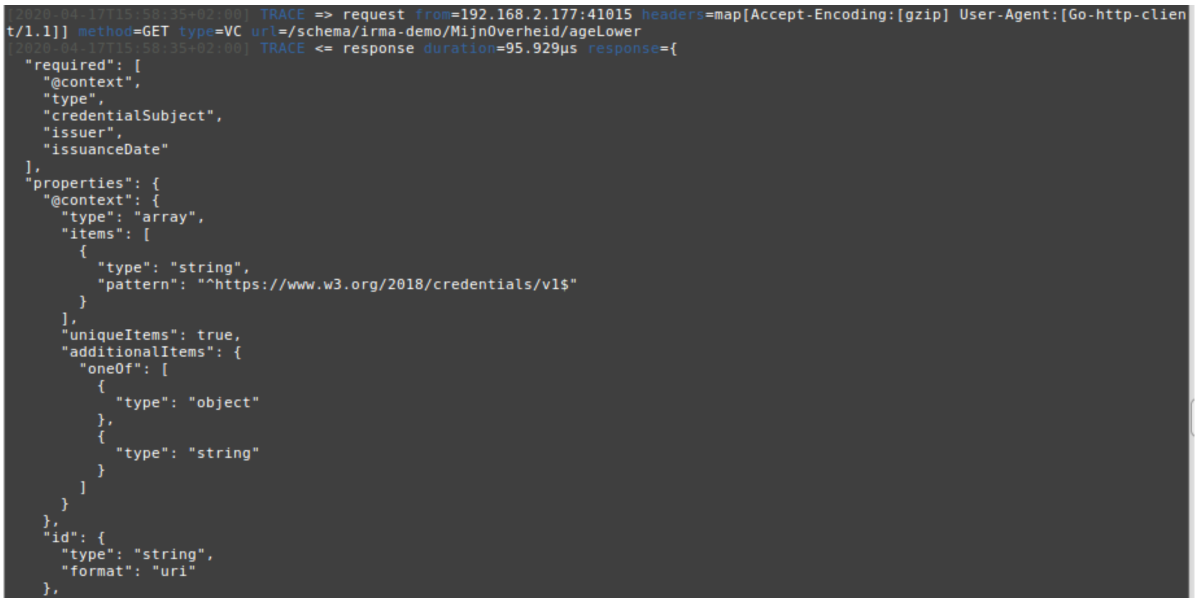
1. Start an IRMA server via *irma server -vv*;
2. Initiate an issuing session via *irma session –server http://localhost:8088 –issue irma-demo.MijnOverheid.ageLower=yes,yes,yes,yes*;
3. Scan the QRCode with the IRMA app;
4. Check that you received the *ageLower* credential on your phone;
5. Check that the session status is *DONE*;
6. If everythings works well, within the terminal of the running IRMA server, you see the messages printed.

One message is the actual VC, as visibile in listing 11. The *@context* contains the mandatory first item and appended the IRMA issuance URI currently used within IRMA. The first type element is *VerifiableCredential* as this message is a VC. The second type element refers to the type information, in this case from the *MijnOverheid* issuer belonging to the *irma-demo* scheme with as type *fullName*. The issuer information, when dereferenced via the given URL resolves to, amongst others, the public key of the issuer, see listing 12. The credentialSubject contains the attributes in plain text. The proof section contains the signature of the credentials, whereby the default IRMA proof message is added under the *gabi* element. Also, due to the implementation of the *Validator* interface within the VC struct, the IRMA app requests the schema and validates the VC, as shown in screenshot E.1 Additionally, for completeness reasons, we print the whole ageLower schema in listing 13.

```
{
    "@context": [
        "https://www.w3.org/2018/credentials/v1",
        "https://irma.app/ld/request/issuance/v2"
    ],
    "credentialSchema": [
        {
            "type": "JsonSchemaValidator2018",
            "id": "http://localhost/schema/irma-demo/MijnOverheid/ageLower"
        }
    ],
    "type": [
        "VerifiableCredential",
        "ageLower"
    ],
    "issuer": "http://localhost/issuer/irma-demo/MijnOverheid/2",
    "issuanceDate": "2020-04-17T15:58:35Z",
    "credentialSubject": [
        {
            "ageLower": {
                "over12": {
                    "en": "yes",
                    "nl": "yes",
                        "rawValue": "yes"
                    },
                    "over16": {
                        "en": "yes",
                        "nl": "yes",
                        "rawValue": "yes"
                    },
                    "over18": {
                        "en": "yes",
                        "nl": "yes",
                        "rawValue": "yes"
                    },
                    "over21": {
                        "en": "yes",
                        "nl": "yes",
                        "rawValue": "yes"
                    }
                }
            }
        ],
        "proof": {
            "type": "IdemixZKP",
            "created": "2020-04-17T15:58:35+02:00",
            "proofMsg": [
                {
                    "proof": {
                        "c": "VEEgc00wmtqli9hYC...",
                        "e_response": "Wn/WqIl6A..."
                    },
                    "signature": {
                        "A": "HtRKdN8/0IYfuo0I...",
                        "e": "EAAAAAAAAAAAAAAAAAAAAAAAAAA...",
                        "v": "CRWK944MdURZBh6okesXRO5xYvF...",
                        "KeyshareP": null
                    }
                }
            ]
        }
}
```

Listing 11: IRMA ageLower VC

```
{
  "required": [
    "@context",
    "type",
    "credentialSubject",
```



```
[2020-04-17T15:58:35+02:00] TRACE => request from=192.168.2.177:41015 headers=map[Accept-Encoding:[gzip] User-Agent:[Go-http-clien
t/1.1]] method=GET type=VC url=/schema/irma-demo/MijnOverheid/ageLower
[2020-04-17T15:58:35+02:00] TRACE <= response duration=95.929µs response={
  "required": [
    "@context",
    "type",
    "credentialSubject",
    "issuer",
    "issuanceDate"
  ],
  "properties": {
    "@context": {
      "type": "array",
      "items": [
        {
          "type": "string",
          "pattern": "^https://www.w3.org/2018/credentials/v1$"
        }
      ],
      "uniqueItems": true,
      "additionalItems": {
        "oneOf": [
          {
            "type": "object"
          },
          {
            "type": "string"
          }
        ]
      }
    },
    "id": {
      "type": "string",
      "format": "uri"
    },
```

Figure E.1: ageLower schema request from the IRMA app

```
    "issuer",
    "issuanceDate"
  ],
  "properties": {
    "@context": {
      "type": "array",
      "items": [
        {
          "type": "string",
          "pattern": "^https://www.w3.org/2018/credentials/v1\$"
        }
      ],
      "uniqueItems": true,
      "additionalItems": {
        "oneOf": [
          {
            "type": "object"
          },
          {
            "type": "string"
          }
        ]
      }
    },
    "id": {
      "type": "string",
      "format": "uri"
    },
    "type": {
      "oneOf": [
        {
          "type": "array",
          "items": [
            {
              "type": "string",
              "pattern": "^VerifiableCredential\$"
            }
          ]
        },
        {
          "type": "string",
          "pattern": "^VerifiableCredential\$"
        }
      ],
      "additionalItems": {
        "type": "string"
      },
      "minItems": 2
    },
    "credentialSubject":
            {
        "type": "array",
        "items": [
                {
                        "additionalProperties": false,
                        "properties": {
                                "ageLower": {
                                        "\$ref": "#/definitions/ageLower"
                                }
                        },
                        "required": [
                                "ageLower"
                        ]
                }
        ]
}
        ,
    "issuer": {
      "anyOf": [
        {
          "type": "string",
          "format": "uri"
        },
        {
            "type": "object",
            "required": [
              "id"
            ],
            "properties": {
              "id": {
                "type": "string"
              }
            }
          }
        ]
      },
      "issuanceDate": {
        "\$ref": "#/definitions/timestamp"
      },
      "proof": {
        "type": "object",
        "required": [
          "type"
        ],
        "properties": {
          "type": {
            "type": "string"
          }
        }
      },
      "expirationDate": {
        "\$ref": "#/definitions/timestamp"
      },
      "credentialStatus": {
        "\$ref": "#/definitions/typedID"
      },
      "credentialSchema": {
        "\$ref": "#/definitions/typedIDs"
      },
      "evidence": {
        "\$ref": "#/definitions/typedIDs"
      },
      "refreshService": {
        "\$ref": "#/definitions/typedID"
      }
    },
    "definitions": {
      "timestamp": {
        "type": "string",
        "pattern": "\\d{4}-[01]\\d-[0-3]\\dT[0-2]\\d:[0-5]\\d:[0-5]\\dZ"
    },
    "typedID": {
      "type": "object",
      "required": [
        "id",
        "type"
      ],
      "properties": {
        "id": {
          "type": "string",
          "format": "uri"
        },
        "type": {
          "anyOf": [
            {
              "type": "string"
            },
            {
              "type": "array",
              "items": {
                "type": "string"
              }
            }
          ]
        }
      }
    },
    "typedIDs": {
      "anyOf": [
        {
          "$ref": "#/definitions/typedID"
        },
        {
          "type": "array",
          "items": {
            "$ref": "#/definitions/typedID"
          }
        }
      ]
    },

        "ageLower": {
                "additionalProperties": false,
                "anyOf": [
                        {
                                "required": [
                                        "over12"
                                ]
                        },
                        {
                                "required": [
                                        "over16"
                                ]
                        },
                        {
                                "required": [
                                        "over18"
                                ]
                        },
                        {
                                "required": [
                                        "over21"
                                ]
                        }
                ],
                "description": "\u003eYour junior age limits ...",
                "properties": {
                        "over12": {
                                "$ref": "#/definitions/translatedString"
                        },
                        "over16": {
                                "$ref": "#/definitions/translatedString"
                        },
                        "over18": {
                                "$ref": "#/definitions/translatedString"
                        },
                        "over21": {
                                "$ref": "#/definitions/translatedString"
                        }
                },
                "type": "object"
        },
        "translatedString": {
                "properties": {
                        "en": {
                                "type": "string"
                        },
                        "nl": {
                                "type": "string"
                        },
                        "rawValue": {
                                "type": "string"
                        }
                }, 
                "required": [
                        "rawValue",
                        "en",
                        "nl"
                ],
                "type": "object"
        }

  }
}
```

Listing 13: ageLower JSON schema

E.3. DISCLOSING SESSION

1. Start an IRMA disclosing session via the IRMA command: *irma session –server http://localhost:8088 –disclose irma-demo.MijnOverheid.ageLower.over21*;

2. From the console of the IRMA server, copy the URL from the session response to be able to create the content of the QRCode;

3. Apply the following template for generating the content of the QRCode: {"url":"«URL»","action":"disclosing", "system":"sovrin"};

4. Via for example the website `https://www.the-qrcode-generator.com/` generate the QR (an example is shown in figure E.2);

5. After scanning the QRCode with the IRMA app, and approving the request, check that the verifiable presentation is printed in the IRMA server console; see listing 14 for an example;

6. Check that the *proofStatus* of the session is *VALID*; it proves that the disclosing is successful.

Figure E.2: A QRCode from another system

```
{
    "@context": [
        "https://www.w3.org/2018/credentials/v1",
        "https://irma.app/ld/request/disclosure/v2"
    ],
    "type": [
        "VerifiablePresentation"
    ],
    "verifiableCredential": [
        {
            "@context": [
                "https://www.w3.org/2018/credentials/v1",
                "https://irma.app/ld/request/disclosure/v2"
            ],
            "credentialSchema": [
                {
                    "type": "JsonSchemaValidator2018",
                    "id": "http://localhost/schema/.../ageLower"
                }
            ],
            "type": [
                "VerifiableCredential",
                "ageLower"
            ],
            "issuer": "http://localhost/issuer/MijnOverheid/2",
            "expirationDate": "2020-10-15T00:00:00Z",
            "credentialSubject": [
                {
                    "ageLower": {
                        "over21": {
                            "en": "yes",
                            "nl": "yes",
                            "rawValue": "yes"
                        }
                    }
                }
            ]
        }
    ],
    "proof": {
        "type": "IRMAZKPPresentationProofv1",
        "created": "2020-04-17T14:46:25Z",
        "proofMsg": {
            "proofs": [
                {
                    "c": "ModFF2P/to0fVG5QFqbaTkx5rzuLeDwoThTCg...",
                    "A": "Yw7pxNVozEHpUQUkNj1iCKTtXJmzlaAXqS0/G...",
                    "e_response": "Syp3ZKpPlgEi4ezL0hZxPym+4...",
                    "v_response": "CZkptQkTV+3q2KVmAkniSKz2kZGeR3w...",
                    "a_responses": {
                        "0": "tAQ5G/xb1fdQ0vhdl02AcJesjL8DrE5AvEP0...",
                        "2": "s5wmS0U7JrsU+8QqhjP...",
                        "3": "cGLHNh6vHEE4utHvwZXV7Qa...",
                        "4": "93Zl4nXYaQkEe6ljnPjEh..."
                    },
                    "a_disclosed": {
                        "1": "AwAKQAAaAALXKWEdEtj9YcHv3rGAKSfq",
                        "5": "8srn"
                    }
                }
            ],
            "indices": [
                [
                    {
                        "cred": 0,
                        "attr": 5
                    }
                ]
            ]
        }
    }
}
```

Listing 14: Disclosed ageLower credential in a verifiable presentation

Listing 12 MijnOverheid issuer metadata

```
{
    "email": "demoverheid@example.com",
    "name": "DemoVerheid.nl",
    "pk": {
        "XMLName": {
            "Space": "http://www.zurich.ibm.com/security/idemix",
            "Local": "IssuerPublicKey"
        },
        "Counter": 2,
        "ExpiryDate": 1701158935,
        "N": "u0n0juLyFLTH...",
        "Z": "BePLF2fghAzyeglJ...",
        "S": "MFSq1yx6Zy6yi...",
        "R": [...],
        "EpochLength": 432000,
        "Params": {
            "LePrime": 120,
            "Lh": 256,
            "Lm": 256,
            "Ln": 1024,
            "Lstatzk": 80,
            "Le": 597,
            "LeCommit": 456,
            "LmCommit": 592,
            "LRA": 1104,
            "LsCommit": 593,
            "Lv": 1700,
            "LvCommit": 2036,
            "LvPrime": 1104,
            "LvPrimeCommit": 1440
        },
        "Issuer": "irma-demo.MijnOverheid"
    }
}
```

F
Personal experience

In this chapter, I write about my experiences conducting this research.

Initially, the literature study was extensive, as I had to understand both IRMA and VC in great detail. To be able to modify the IRMA software, I had to learn the *go* programming language. Then, I had to understand the IRMA source code.

During the project, I had some issues with figuring out which research method to follow. This work was more about creating some software related artifacts. It was less about gathering data, and reason about it, as it is common with empirical research. Hence, it took some time until I figured that the design science approach could fit my needs. Only quite late, I revised the introduction and research questions extensively. This resulted in a better guideline to follow for rest of the research.[1] Additionally, finalizing the thesis took more time than anticipated.

All in all, I can state that after multiple iterations, and additional study of material with regards to design science, I am satisfied with the structure and content of the research.

I advise the Open Universiteit to lecture students about "Design Science" as it might be helpful during the graduation project.

I used the book "Style: Lessons in Clarity and Grace" (from Williams and Bizup) to improve my English writing skills.

In my opinion, students can use my thesis as a template when conducting a *compare and contrast* research, and additionally, developing a software artifact. Initially, I explain in detail how to conduct the research based on DSRM, thereby describing which artifacts to create. Then, I describe the topics the research is about extensively. Afterward, I identify requirements, modifications, and the impact of the modifications. Based on a subset of the modifications, I show how to develop a prototype. Finally, the related work and following discussion help to put everything into context.

[1]I used the presentation from Wieringa for improving my research questions: `https://wwwhome.ewi.utwente.nl\/~roelw\/DSM90minutes.pdf`

Acronyms

ABC	Attribute Based Credential.
API	Application Programming Interface.
CA	Certificate Authority.
eIDAS	electronic IDentification, Authentication and trust Services.
ERD	Entity Relationship Diagram.
EU	European Union.
GDPR	General Data Protection Regulation.
IAM	Identity and Access Management.
Idemix	IBM Identity Mixer.
IdM	Identity Management.
IMS	Identity Management System.
IRMA	I Reveal My Attributes.
LCIM	Levels of Conceptual Interoperability.
NIST	National Institute of Standards and Technology.
PbDF	Privacy by Design Foundation.
SACCS	Software Architecture Change Characterization Scheme.
SP	Service Provider.
SSI	Self-Sovereign Identity.
SSO	Single-Sign On.
VC	Verifiable Credential.
W3C	World Wide Web Consortium.
ZKP	Zero-Knowledge Proofs.