Design and Implementation of a Decentralized Trusted Issuer Registry for Self-Sovereign Identity

Michael Schmidmaier

15.01.2024, Bachelor's Thesis Final Presentation

Chair of Software Engineering for Business Information Systems (sebis)
Department of Computer Science
School of Computation, Information and Technology (CIT)
Technical University of Munich (TUM)
wwwmatthes.in.tum.de

Outline

1. Motivation
2. Research Questions & Results
3. Prototype Demo
4. Limitations, Future Work & Conclusion

Digital Identity?

Today

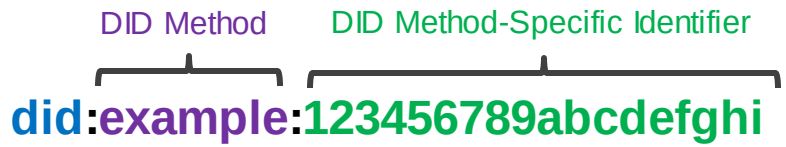
- Centralized identity solutions
- Users have limited control over their identities
- Low interoperability & portability, limited privacy, …

Self-Sovereign Identity (SSI) [1]

- Users have full control over their identity without relying on a third party
- Decentralized, interoperable, privacy preserving, …

Decentralized Identifiers (DIDs) [2]

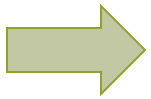
- A W3C standard for identifying subjects without relying on a central organization



did:ethr:0xc530503a148babcaca6…

did:web:tum.de

…



```
// DID-Document
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    "https://w3id.org/security
/suites/ed25519-2020/v1"
  ],
  "id":
"did:example:123456789abcdefghi",
  authentication": [{
    "id":
"did:example:123456789abcdefghi#key
s-1",
    "type":
"Ed25519VerificationKey2020",
    "controller":
"did:example:123456789abcdefghi",
    "publicKeyMultibase":
"zH3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwn
Zn6z3wXmqPV"
  }]
}
```

Verifiable Credentials (VCs) [3]

- A W3C standard for subjects to make verifiable claims about others
- Example: University issues a digital bachelor certificate to a student

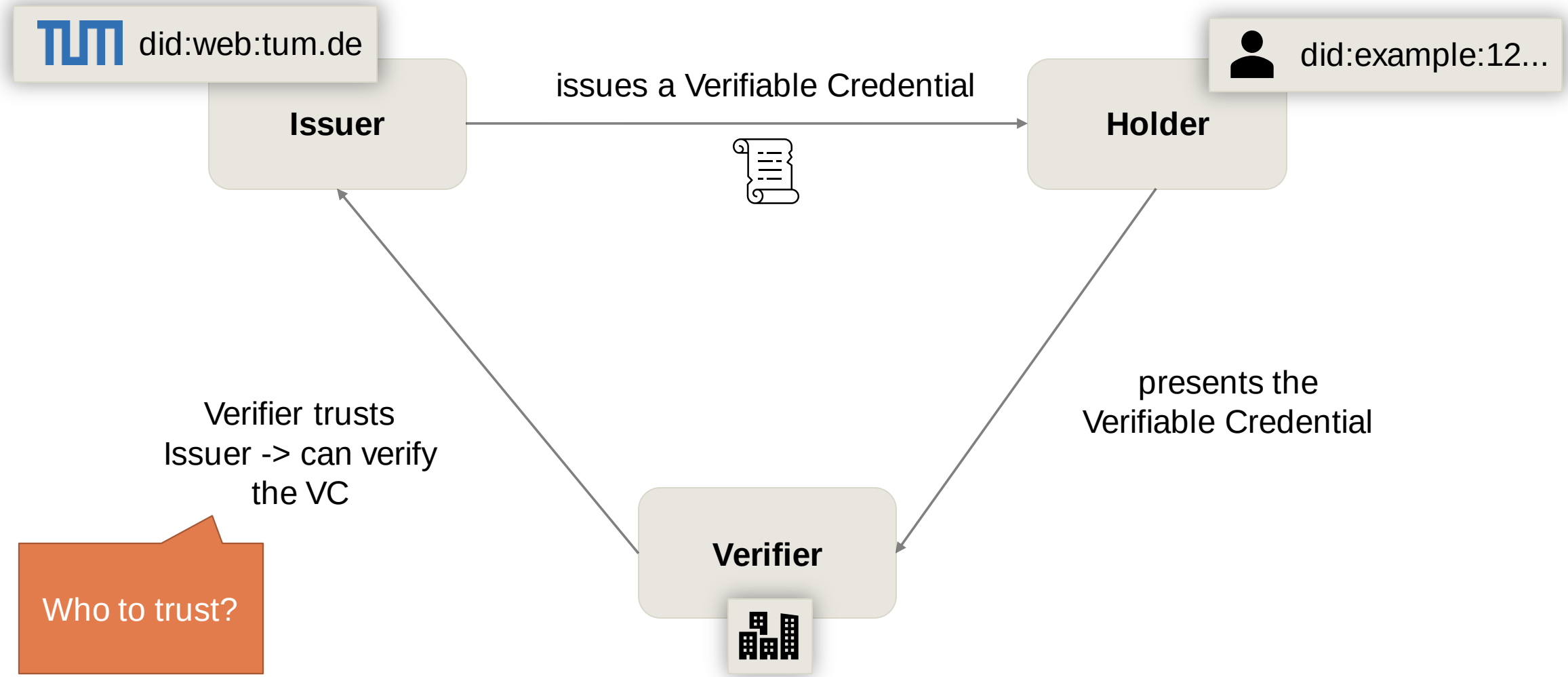
...


```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials
/examples/v1"
  ],
  "id": "http://example.edu/credentials/1872",
  "type": ["VerifiableCredential",
"AlumniCredential"],
  "issuer": "https://example.edu/issuers/565049",
  "issuanceDate": "2023-01-01T19:23:24Z",
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "alumniOf": {
      "id":
"did:example:c276e12ec21ebfeb1f712ebc6f1",
      "name": [{
        "value": "Example University",
        "lang": "en"
      }]
    }
  },
  "proof": {
    "type": "RsaSignature2018",
    "created": "2023-06-18T21:19:10Z",
    "proofPurpose": "assertionMethod",
    "verificationMethod": "https://example.edu
/issuers/565049#key-1",
    "jws": "eyJhbGci...dBBPM"
  }
}
```

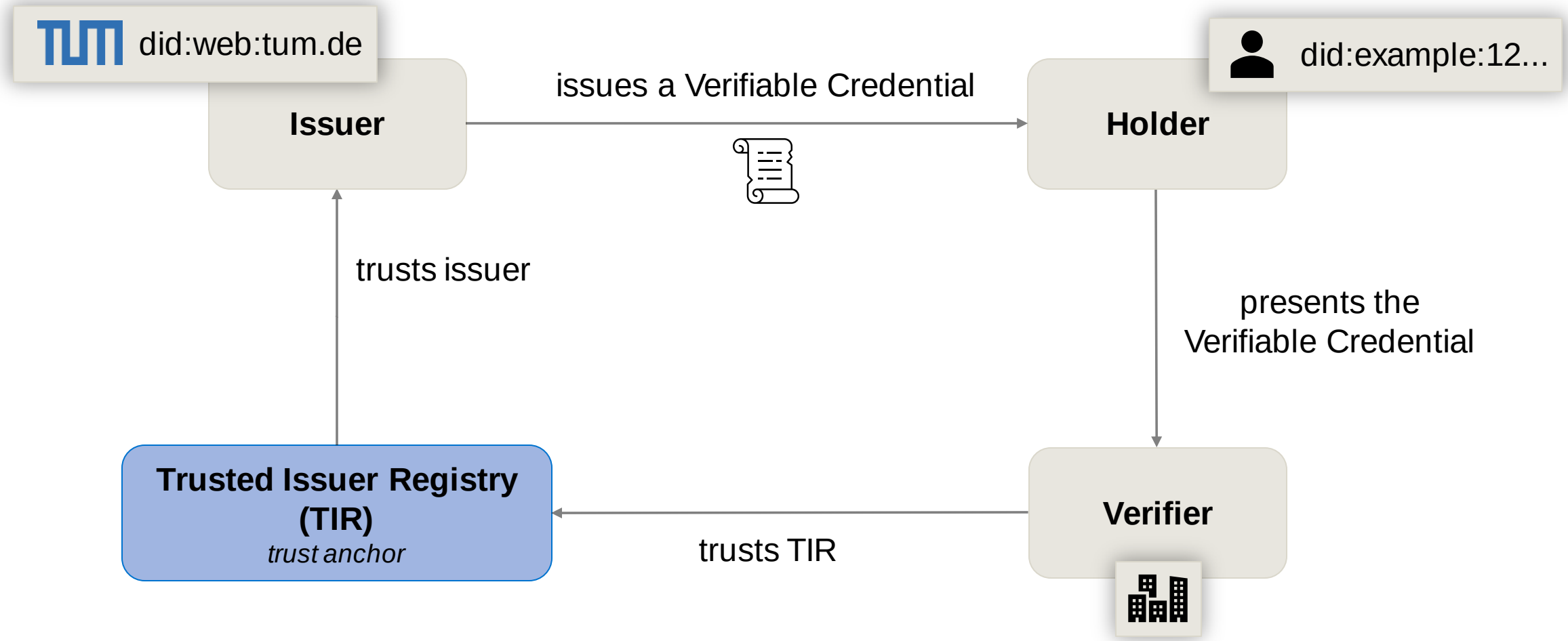

Verifiable Credential

- Metadata (e.g., Issuer DID)
- Claim(s)
- Proof(s)

Problem Statement



Problem Statement



Outline

1. Motivation
2. Research Questions & Results
3. Prototype Demo
4. Limitations, Future Work & Conclusion

Research Questions

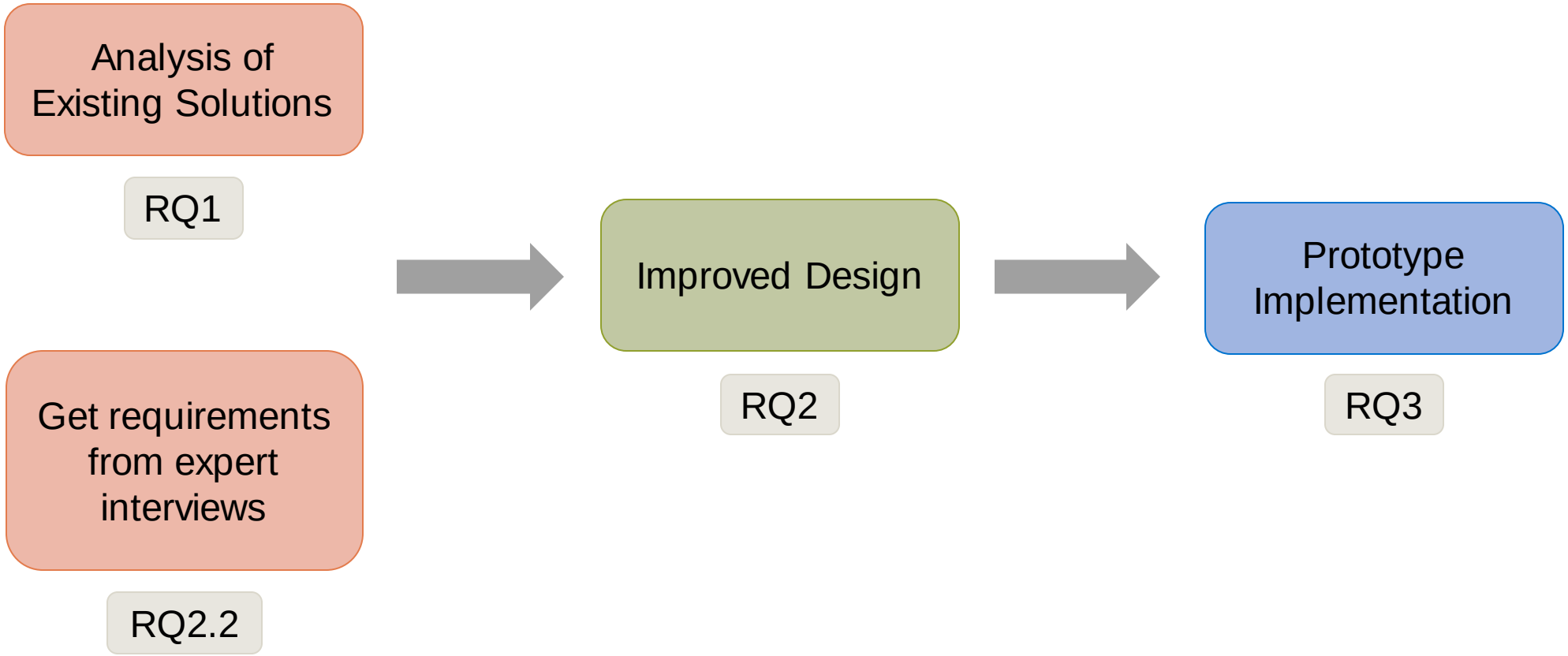
RQ1: What are the advantages and disadvantages of existing centralized and decentralized Trusted Issuer Registry designs?

RQ2: How can a general-purpose Trusted Issuer Registry be designed to meet the needs of Self-Sovereign Identity in Gaia-X ecosystems and address the drawbacks of existing solutions?

- **RQ2.1:** What specific functionalities should a Trusted Issuer Registry provide in Gaia-X ecosystems?
- **RQ2.2:** What are the requirements for a Trusted Issuer Registry in Gaia-X ecosystems?
- **RQ2.3:** What is a suitable technical infrastructure for a Trusted Issuer Registry?
- **RQ2.4:** How can scalable governance be achieved?

RQ3: How can the design be implemented using a concrete technology?

Methodology



Analysis of Existing Trusted Issuer Registry Designs

- Goal: find advantages & disadvantages of current solutions
- Analysis limited to public information, no practical tests
- Structured by:
  Trust Concept, Use Case, Storage, Functionality, Scalability, Performance, Security, Complexity

	X.509 PKI	EBSI	TRAIN	DCC	OCI	ToIP
TIR concept		✓	✓	✓	✓	✓
Chaining concept	✓	✓				
General-purpose	✓		✓		✓	✓
Storage type	**decentral.**	**decentral.**	**combinat.**	**central.**	**decentral.**	n/a
Stored in one place	n/a	✓		✓	✓	n/a
Issuer identification	✓		✓	✓		n/a
Issuer authorization		✓	✓		✓	✓
Hierarchy (sub-registries)	✓	✓	✓			n/a
Subregistry authorization	✓	✓		n/a	n/a	n/a
Caching intended		✓				
Fully integrity-protected	✓	✓[7]		✓		

Analysis of Existing Trusted Issuer Registry Designs

- Many design decisions depend on the use case
- All solutions show some disadvantages
  - Little functionality
  - Security issues
  - Inconvenient management
  - High centralization
  - ...

	X.509 PKI	EBSI	TRAIN	DCC	OCI	ToIP
TIR concept		✓	✓	✓	✓	✓
Chaining concept	✓	✓				
General-purpose	✓		✓		✓	✓
Storage type	**decentral.**	**decentral.**	**combinat.**	**central.**	**decentral.**	n/a
Stored in one place	n/a	✓		✓	✓	n/a
Issuer identification	✓		✓	✓		n/a
Issuer authorization		✓	✓		✓	✓
Hierarchy (sub-registries)	✓	✓	✓			n/a
Subregistry authorization	✓	✓		n/a	n/a	n/a
Caching intended		✓				
Fully integrity-protected	✓	✓[7]		✓		

Requirements Analysis with Expert Interviews

- No established requirements for Trusted Issuer Registries yet
- Semi-structured expert interviews lasting 45 – 60 minutes
- 5 interviewees from the GAIA-X 4 Future Mobility project (future implementers of SSI)

Functional

- **FR1: Register Trusted Issuer:** The TIR must enable registering new trusted issuers.
- **FR2: Update Trusted Issuer:** The TIR must enable updating existing trusted issuers.
- **FR3: Endorse Sub-Registries:** The TIR must allow delegating trust to sub-registries.
- **FR4: Verify Issuer's Trustworthiness:** The TIR must allow verification of an issuer's trustworthiness.

Security

- **NFR1: Issuer Authorization:** The TIR must allow specifying specific issuer qualifications.
- **NFR2: Limit Sub-Registry Trust Delegation:** The TIR should allow limiting the delegated trust for endorsed TIRs.
- **NFR3: Issuer Revocation:** Trust in issuers must be revocable.
- **NFR4: Data Integrity:** The integrity of the TIR's contents must be ensured using state-of-the-art methods.
- **NFR5: Sensitive Data:** Sensitive issuer data must not be public.
- **NFR6: Verification Leak:** Issuer verifications must not leak the issuer's identity.

Performance

- **NFR7: Scalability:** The architecture must scale well to at least 1000 issuers inside a single TIR.
- **NFR8: Verification Latency:** Verifiers must be able to verify an issuer within 5 seconds.
- **NFR9: Write Latency:** Changes made to the TIR must be effective for all verifiers within 24 hours.
- **NFR10: Availability:** Issuers must be verifiable at all times. This may be achieved through caching (see NFR9).

Integration

- **NFR11: Portability:** The TIR must be portable to different base technologies.
- **NFR12: Interoperability:** The TIR should utilize common technologies, standards and interfaces to facilitate interoperability.
- **NFR13: Adaptability:** The TIR architecture must be flexible enough to be used in different use cases.

TIR Design - Overview

TIRs are referenced with DID services. Those can be resolved to the TIR data model. "TIR methods" define the implementation-specific resolution details.

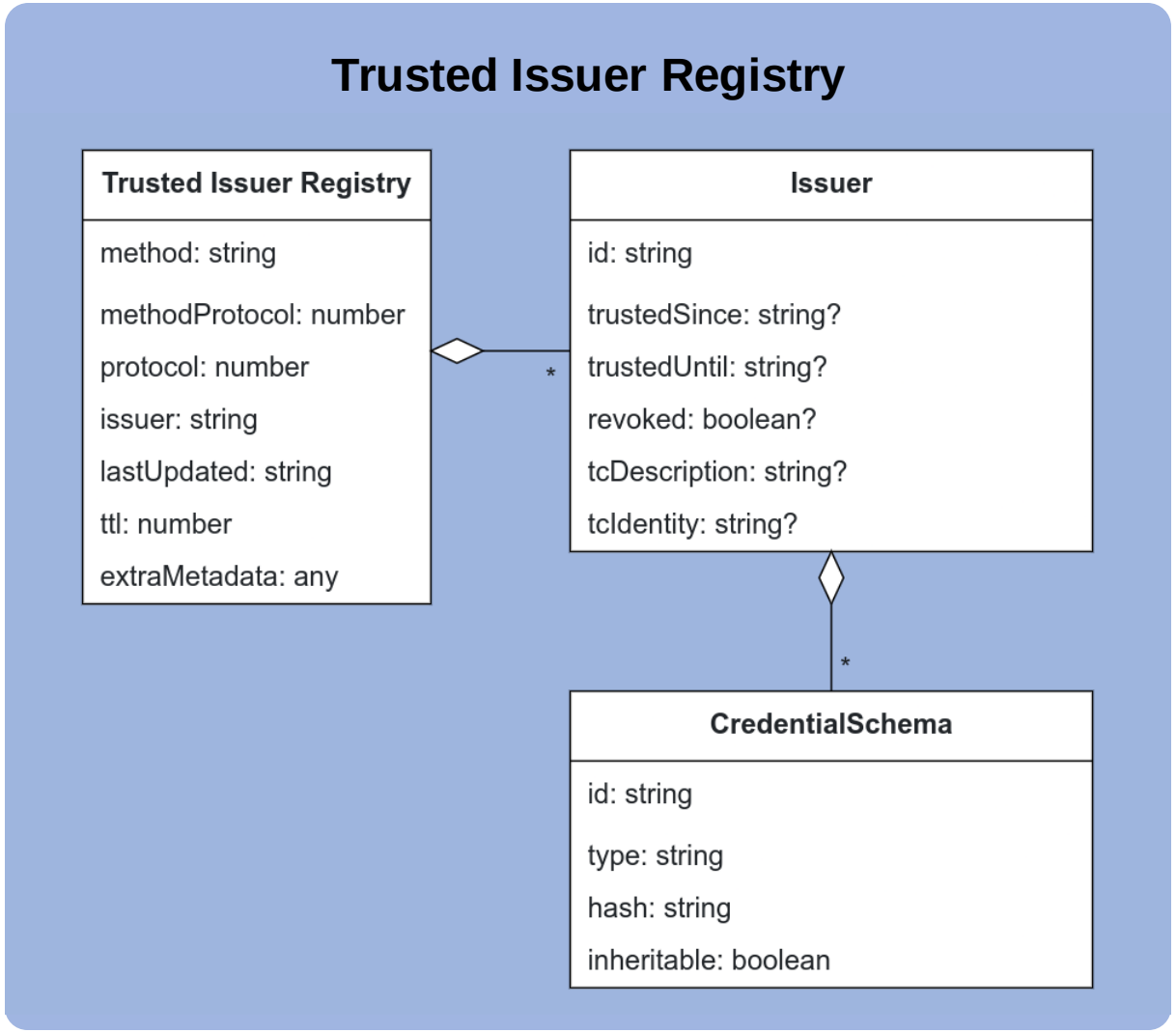
DID: *did:example:123456789*

DID resolution
"DID method"-specific

DID Document

```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    ...
  ]
  "id": "did:example:123456789",
  "verificationMethod": [...],
  "authentication": [...],
  "service": [{
    "id": "did:example:123456789#tir",
    "type": "TrustedIssuerRegistry2023Web",
    "serviceEndpoint":
"https://example.com/tir.json"
  }]
}
```

TIR resolution
"TIR method"-specific



TIR Design - TIR Methods

Specifications defining the actual TIR resolution: service endpoint URI → TIR data model

Web

- Web: Stores the TIR's contents as JSON on a webserver
- Centralized storage

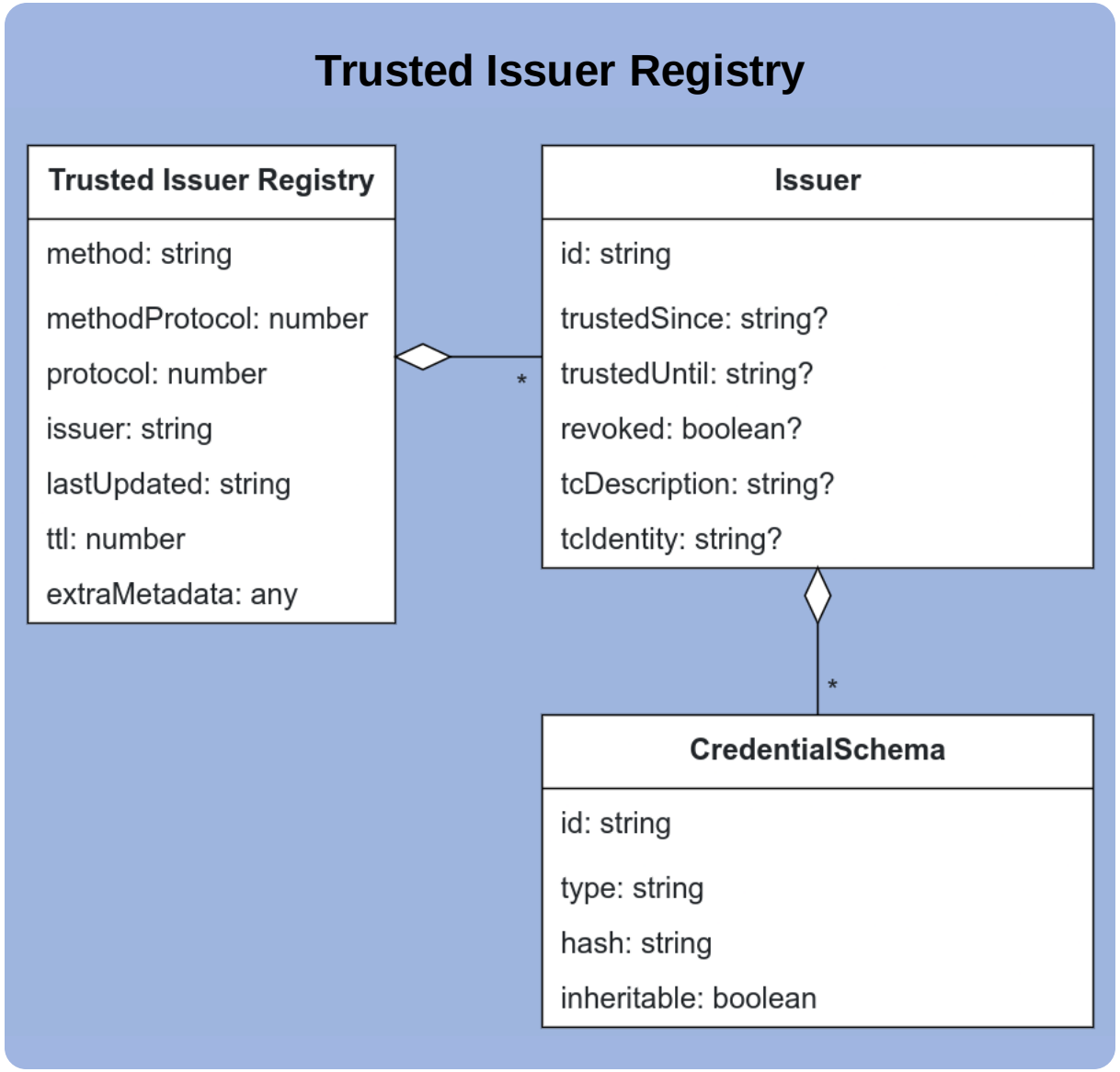
Tezos

- Stores the TIR inside a smart contract on the Tezos blockchain
- Decentralized storage

...

- Different technologies, storage types
- ...

TIR Design - Authorization



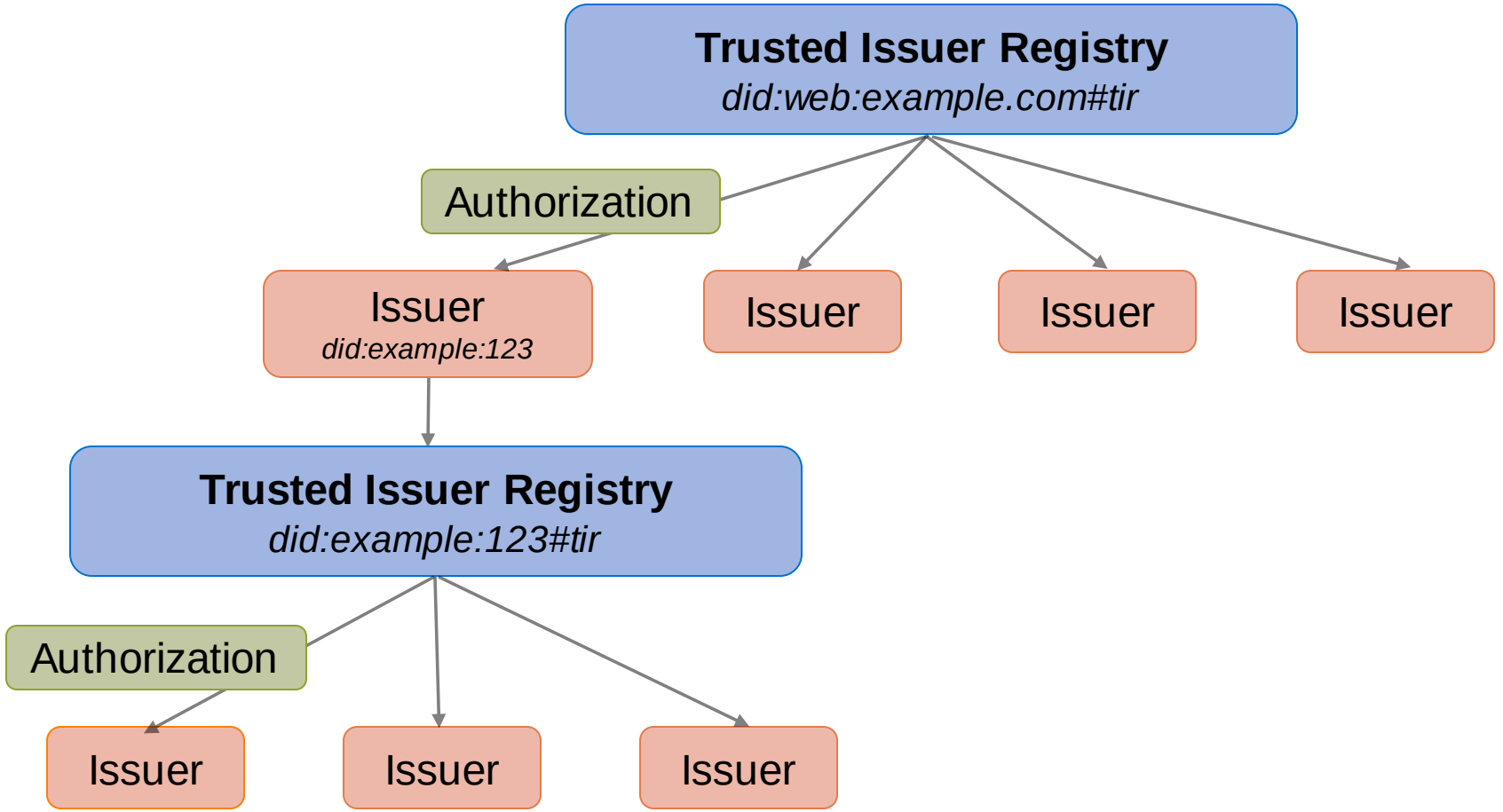
Two ways to limit delegated trust

1. Credential Schemas:
   - Machine-readable information limiting issuable credentials
   - Enables automatic verification

2. Trust Context:
   - Human-readable information on the context in which trust is delegated
   - May connect an issuer to a legal identity
   - Supports manual verification in case automatic verification fails

TIR Design - Hierarchy

Issuers with a DID can reference sub-registers.
Issuers and sub-registries are subject to all authorization and context information on the path to the root.



Prototype Implementation [4]

Smart Contract

- Implementation of the Tezos TIR Method
- JsLigo
- Basic CRUD for contract storage

Core Package

- Shared TS code
- TIR resolution
- Issuer verification

Backend

- Express + Redis cache
- Decouples TIR resolution and issuer verification by caching

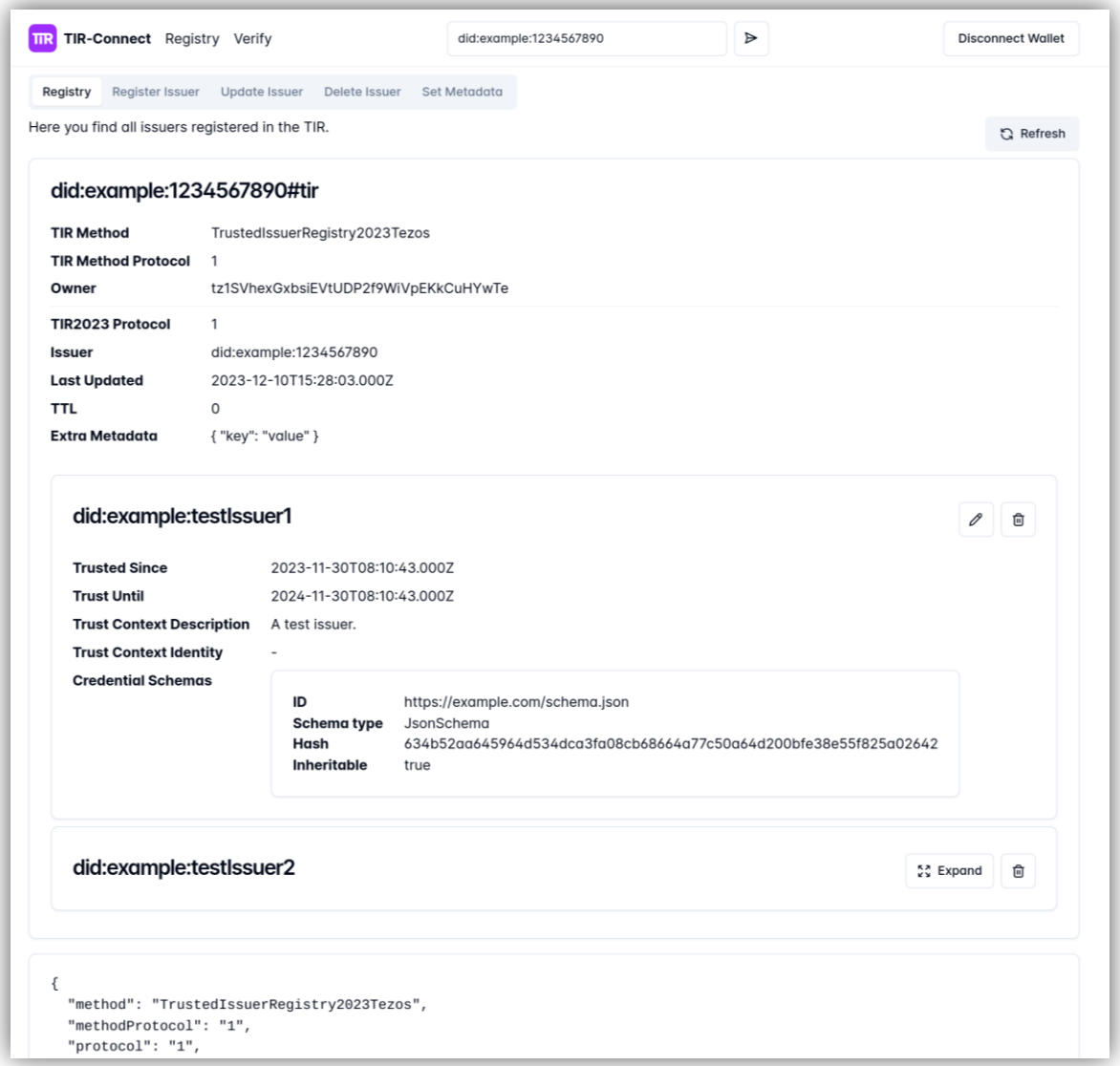
Frontend

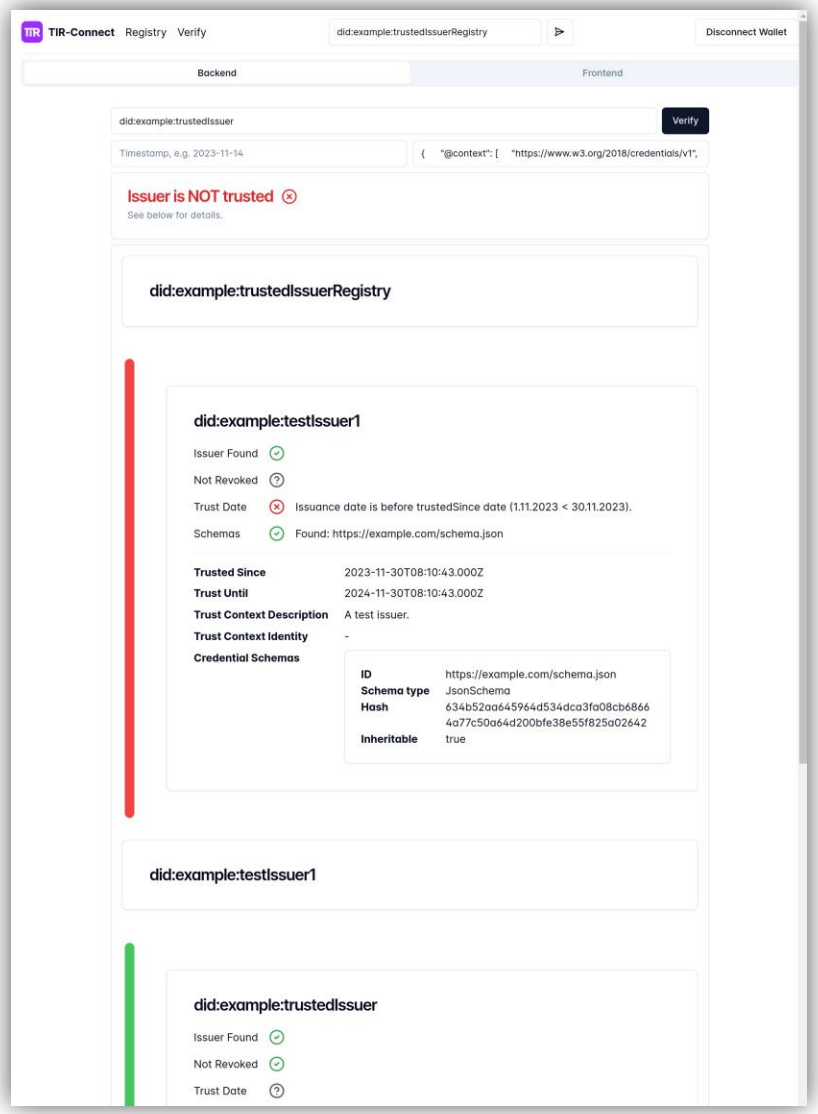
- Vue web app
- UI for Tezos TIR (CRUD)
- Issuer verification (standalone & using backend)

Outline

1. Motivation
2. Research Questions & Results
3. Prototype Demo
4. Limitations, Future Work & Conclusion

Prototype Implementation – Live Demo





Outline

1. Motivation
2. Research Questions & Results
3. Prototype Demo
4. Limitations, Future Work & Conclusion

Limitations & Future Work

RQ1	Analysis of Existing Solutions	• Small set of analyzed solutions • Limited to public documentation • No practical evaluation
RQ2.2	Requirements Analysis with Expert Interviews	• Limited number of participants • All interview participants from same Gaia-X project family
RQ2	TIR Design	• Verification leak • Interoperability - Performance tradeoff
RQ3	Prototype Implementation	• Further refinement needed • No in-depth evaluation

Directions for Future Work

- More extensive interviews
- More detailed, practical evaluations regarding, e.g., scalability
- Further development of TIR design & prototype

Conclusion

RQ1
- Analysis of six existing TIR designs
  - Found several disadvantages
  - Design decisions depend on the requirements

RQ2.2
- Found **17 requirements** for TIRs from a real use case: Gaia-X
  - Basis for design decisions and evaluation

RQ2
- Proposed a decentralized **Trusted Issuer Registry design** fulfilling the requirements and addressing the drawbacks of existing solutions
  - Portable, interoperable, flexible, feature-rich

RQ3
- Demonstrated the design in an extensive **prototype application**
  - Reusable and extendable

Michael Schmidmaier
michael.schmidmaier@tum.de

Technical University of Munich (TUM)
TUM School of CIT
Department of Computer Science (CS)
Chair of Software Engineering for Business Information Systems (sebis)

Boltzmannstraße 3
85748 Garching bei München

References

- Stock photos: pexels.com

- [1] C. Allen, “The Path to Self-Sovereign Identity,” Life With Alacrity. [Online]. Available: http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity.html
- [2] M. Sabadello, M. Sporny, A. Guy, and D. Reed, “Decentralized Identifiers (DIDs) v1.0,” W3C, W3C Recommendation, Jul. 2022. [Online]. Available: https://www.w3.org/TR/2022/REC-did-core-20220719/
- [3] D. Longley, B. Zundel, K. D. Hartog, D. Burnett, G. Noble, and M. Sporny, “Verifiable Credentials Data Model v1.1,” W3C, W3C Recommendation, Mar. 2022. [Online]. Available: https://www.w3.org/TR/2022/REC-vc-data-model-20220303/
- [4] Michael Schmidmaier, “Trusted Issuer Registry 2023 Prototype Implementation.” GitHub, Dec. 14, 2023. [Online]. Available: https://github.com/Trusted-Issuer-Registry-2023/tir2023-prototype

- Michael Schmidmaier, Design and Implementation of a Decentralized Trusted Issuer Registry for Self-Sovereign Identity. 2023. [Online]. Available: https://wwwmatthes.in.tum.de/pages/dss6cww4npqp/Bachelor-s-Thesis-Michael-Schmidmaier

Backup Slides

Gaia-X

- A large European project, currently 377 members
- Goal: a federated, self-sovereign and secure data infrastructure built on common standards and interfaces
- Gaia-X won't run the infrastructure, but build its standards

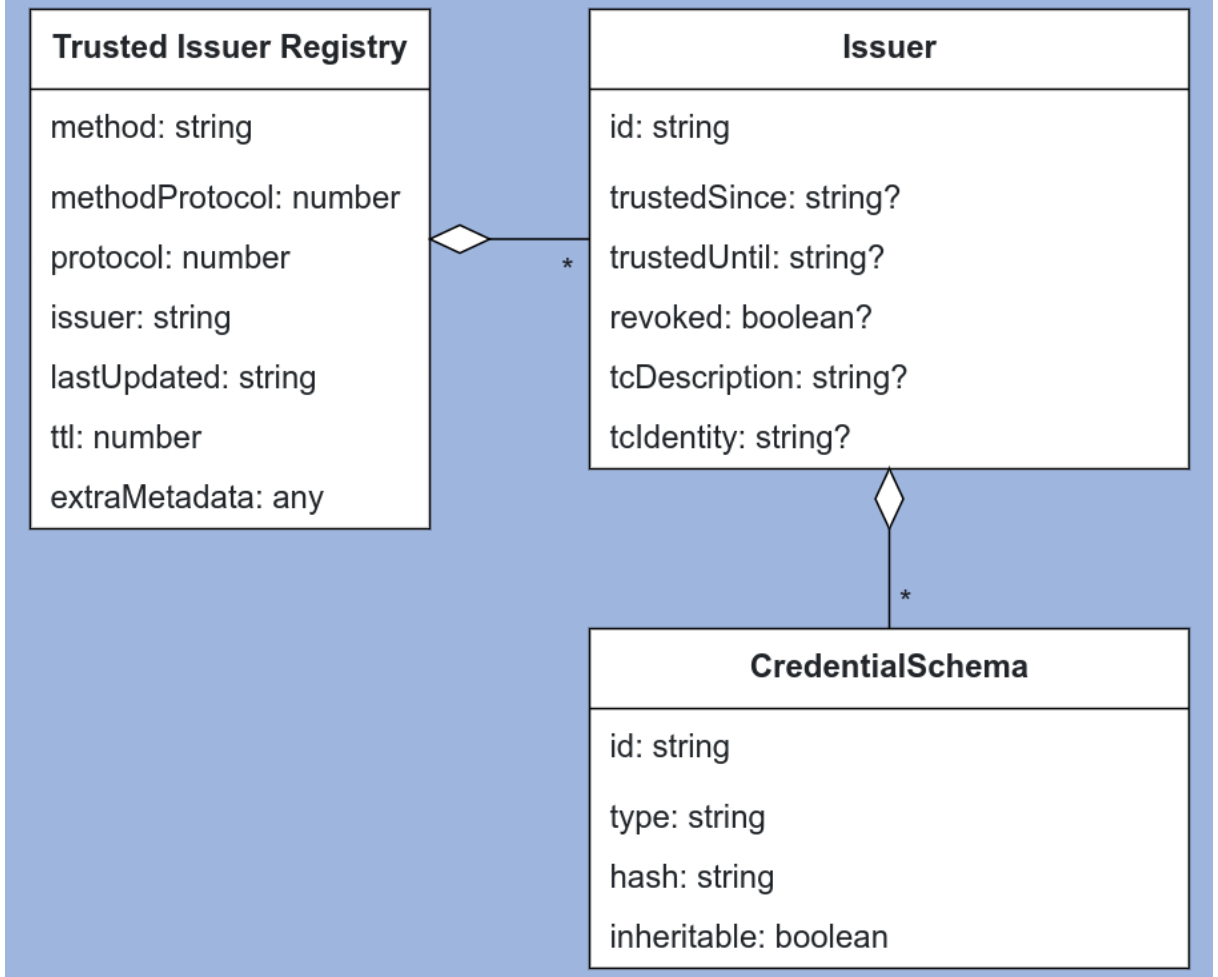
- Source of requirements for my thesis

Interview Guide

1. Introduction: Who are you and what is your position in your company/organization?
2. Please briefly introduce the GAIA-X work package or use case you primarily work on.
3. Is your project / work package already using Verifiable Credentials?
    1. Have you already encountered any problems/weaknesses with the current solution?
4. Please describe the primary use case of Verifiable Credentials in your project.
    1. Who is the Holder, Issuer, Verifier?
    2. What types of credentials are involved? (identity proof, authorization proof, etc.)
    3. How often does a verification process occur?
    4. How many Issuers are involved?
    5. How dynamic is the list of involved Issuers?
    6. Are the Issuers always all known?
5. What specific aspects must the verification check? (technical authenticity, issuer identity, issuer qualification, etc.)
6. Are there specific requirements for the verification? (security, transparency, performance, privacy, costs)
7. Would you prefer to rely on your own Trusted Issuer Registries or third-party ones?
8. Do you have preferences regarding a decentralized or centralized TIR? (decentralized storage and governance)
9. Do you prefer a small, use-case-bound Trusted Issuer Registry over a potentially very large, but well-organized cross-use-case Trusted Issuer Registry?

TIR Design – VC Serialization

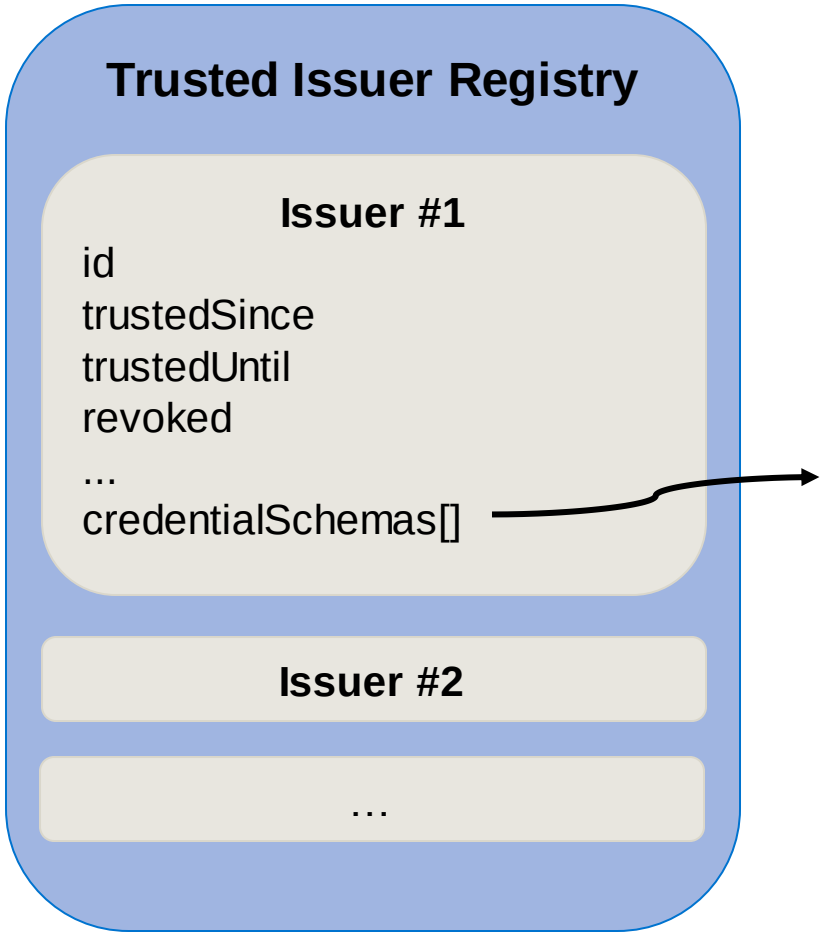
Trusted Issuer Registry



Verifiable Credential Serialization

```
{
  "@context": ["https://www.w3.org/2018/credentials/v1"],
  "id": "http://example.edu/credentials/1872",
  "type": ["VerifiableCredential"],
  "issuer": "did:example:1234567890",
  "issuanceDate": "2023-11-23T11:23:24Z",
  "expirationDate": "2023-11-24T11:23:24Z",
  "credentialSubject": {
    "method": "TrustedIssuerRegistry2023Web",
    "methodProtocol": 1,
    "protocol": 1,
    "issuer": "did:example:1234567890",
    "lastUpdated": "2023-11-23T11:23:24Z",
    "ttl": 86400,
    "extraMetadata": {"key": "value"},
    "issuers": {
      "did:web:example.com": {
        "trustedSince": "2023-11-23T11:23:24Z",
        "trustedUntil": "2024-11-23T11:23:24Z",
        "revoked": false,
        "tcDescription": "An example issuer only used for...",
        "tcIdentity": "John Doe Company, Example Street, ...",
        "credentialSchemas": [{
          "id": "https://example.com/testschema.json",
          "type": "JsonSchema2020",
          "hash": "d14a028c2a3a2bc...8ea62ac5b3e42f",
          "inheritable": true
        }]
      }
    }
  },
  "proof": {...}
}
```

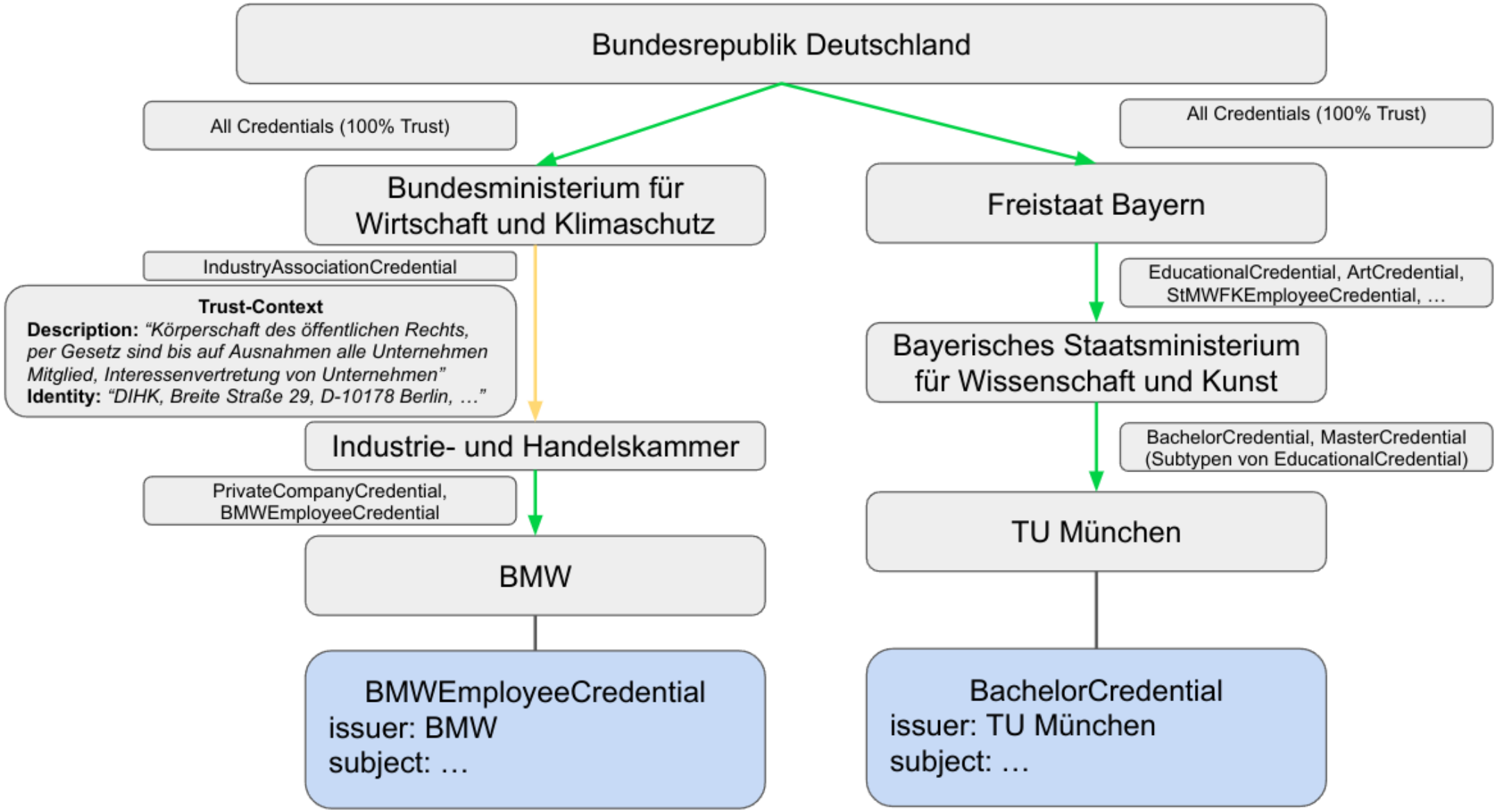
Authorization – Credential Schemas




```
{
  "$id": "https://example.com/schemas/email.json",
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "name": "EmailCredential",
  "description": "EmailCredential using JsonSchema",
  "type": "object",
  "properties": {
    "credentialSubject": {
      "type": "object",
      "properties": {
        "emailAddress": {
          "type": "string",
          "format": "email"
        }
      },
      "required": [
        "emailAddress"
      ]
    }
  }
}
```

[9]

Trust Path Example



Comparison of our design and the analyzed TIRs

	X.509 PKI	EBSI	TRAIN	DCC	OCI	ToIP	Our Design
TIR concept		✓	✓	✓	✓	✓	✓
Chaining concept	✓	✓					
General-purpose	✓		✓		✓	✓	✓
Storage type	**decentral.**	**decentral.**	**combinat.**	**central.**	**decentral.**	n/a	**flexible**
Stored in one place	n/a	✓		✓	✓	n/a	
Issuer identification	✓		✓	✓		n/a	✓
Issuer authorization		✓	✓		✓	✓	✓
Hierarchy (sub-registries)	✓	✓	✓			n/a	✓
Subregistry authorization	✓	✓		n/a	n/a	n/a	✓
Caching intended		✓					✓
Fully integrity-protected	✓	✓		✓			✓

Table 6.1: Comparison of our design and the analyzed TIRs, based on Table 4.1