universitätbonn

Semantic Blockchain

Firas Kassawat, Ernane Luis, S. Matthew English

Supervisor:
Prof. (Univ. Simón Bolívar) Dr. Maria Esther Vidal

Outline

- Introduction
- Motivation
- Vocabulary
- Semantic Blockchain
  - Knowledge Graph
  - Query Processing
  - Data Analytics
  - Demo
- Obstacles Faced
- Future Work
- Demo & Architecture

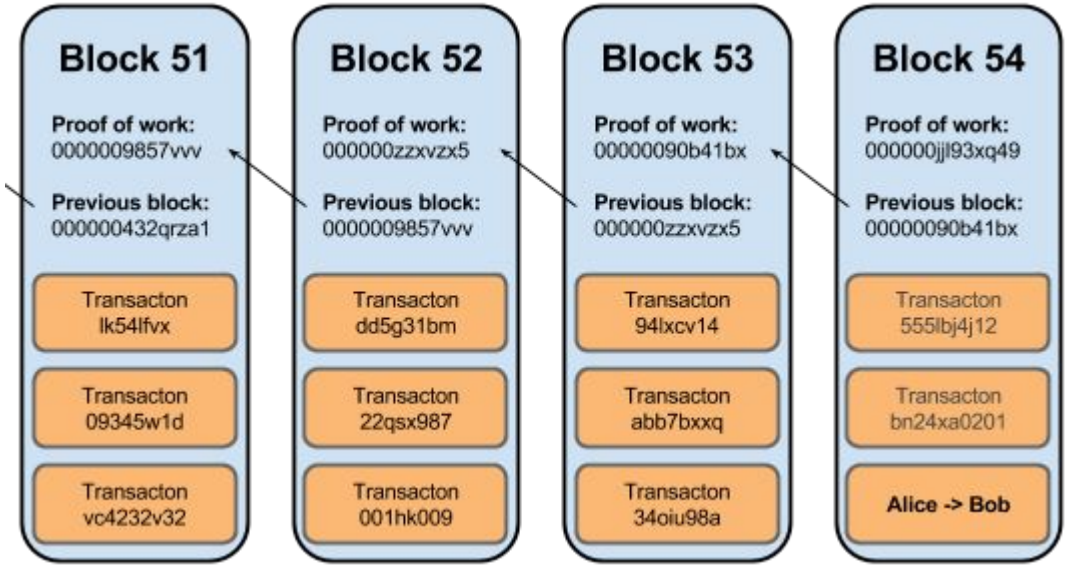
universitätbonn

1

Introduction

Creating a Bitcoin Knowledge Graph

The Bitcoin Ecosystem



Bitcoin

Bitcoin is a digital asset and value transmission system:

→ Open Data

→ Distributed

→ Pseudo-Anonymous

2

Motivation

Sophisticated Network Analytics
Through Data Integration

Problem with Blockchain Explorers

BLOCKCHAIN

Blockchain Explorers are Insufficient:

→ **No** issuing of queries

→ **No** clustering of addresses

→ **No** association of pseudonyms with *"reality"*

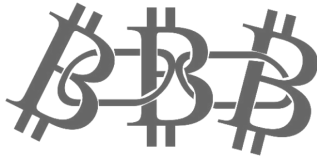
→ **No** voluntary association with online identity

onename

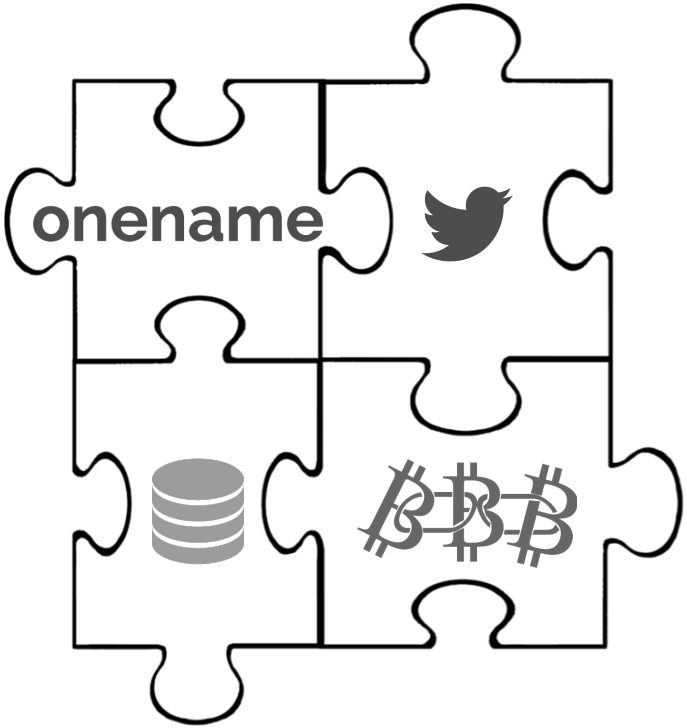
Regulatory Issues:

→ Anti-money laundering (AML)

→ Know your customer (KYC)



Unification of Different Resources



Queries that can aggregate this info:

→ Seamless integration of data

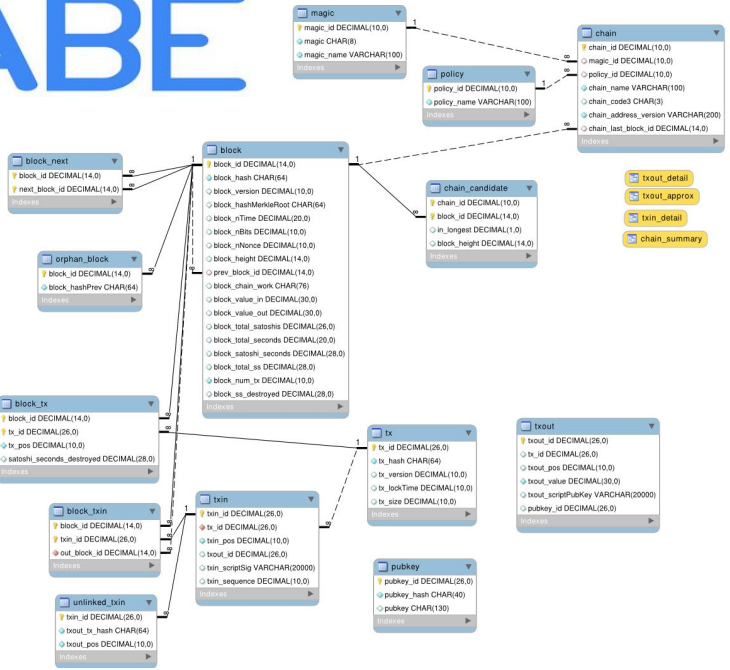
→ Creating a more complete picture of network participants and interactions

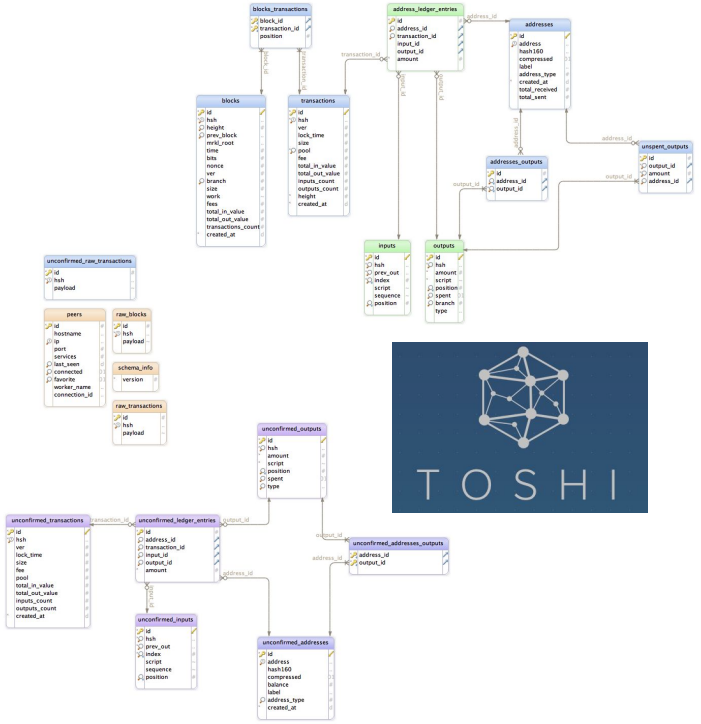
How to store this information?

universitätbonn

Relational Blockchain Database Model

ABE

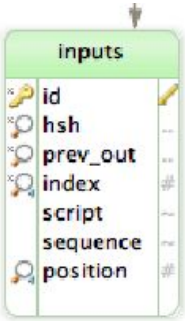




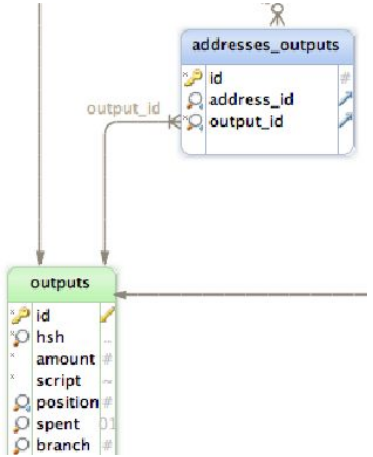
Deficiencies of Relational Database Model

universitätbonn

“Hi, my name is Alice,
I’ve sent Bob some BTC”



“No, I didn’t receive
anything from Alice”



```
SELECT tx_id, tx_version, tx_lockTime, tx_size
    FROM tx
 WHERE tx_hash =
"9bb5071f1b1c3ef3bb3fcf2497712c86178bdb0bd047dc4103233979
a55ec2e7"
join    SELECT
        txout.txout_pos,
        txout.txout_scriptPubKey,
        txout.txout_value,
        nexttx.tx_hash,
        nexttx.tx_id,
        txin.txin_pos,
        pubkey.pubkey_hash
      FROM txout
      LEFT JOIN txin ON (txin.txout_id = txout.txout_id)
      LEFT JOIN pubkey ON (pubkey.pubkey_id = txout.pubkey_id)
      LEFT JOIN tx nexttx ON (txin.tx_id = nexttx.tx_id)
     WHERE txout.tx_id = ?
     ORDER BY txout.txout_pos
join
```

```
SELECT
        txin.txin_pos,
        txin.txin_scriptSig,
        txout.txout_value,
        COALESCE(prevtx.tx_hash, u.txout_tx_hash),
        prevtx.tx_id,
        COALESCE(txout.txout_pos, u.txout_pos),
        pubkey.pubkey_hash
      FROM txin
      LEFT JOIN txout ON (txout.txout_id = txin.txout_id)
      LEFT JOIN pubkey ON (pubkey.pubkey_id = txout.pubkey_id)
      LEFT JOIN tx prevtx ON (txout.tx_id = prevtx.tx_id)
      LEFT JOIN unlinked_txin u ON (u.txin_id = txin.txin_id)
     WHERE txin.tx_id = ?
     ORDER BY txin.txin_pos
```

Match
(b:BTC_Address)-[:recieve]->(p:Transaction
{hsh:{hsh}])-[:recieve]->(b:BTC_Address)

Why Graph Database?

Natural modeling of highly connected data.

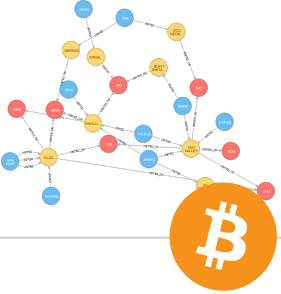
Special graph storage structure

Efficient schemaless graph algorithms.

Support for query languages.

Operators to query the graph structure.

Best way to represent transactional data similar to Blochain data..



Transactional Graph Database

Representing Transactions: Graph Data vs. Triple Store

For traversal look up and querying is faster:

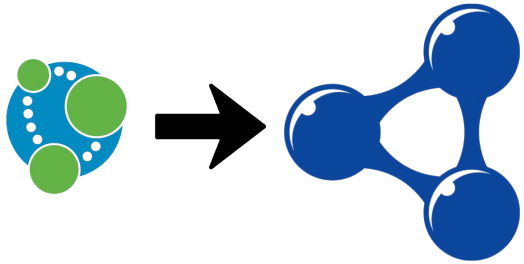
→ Issue queries of high complexity

→ Efficient query processing throughout graph

***Traversal* is Possible in Graph DB but *not* in Relational DB**

The goal is to facilitate query processing & data analytics

→ Global identifier to establish the integration



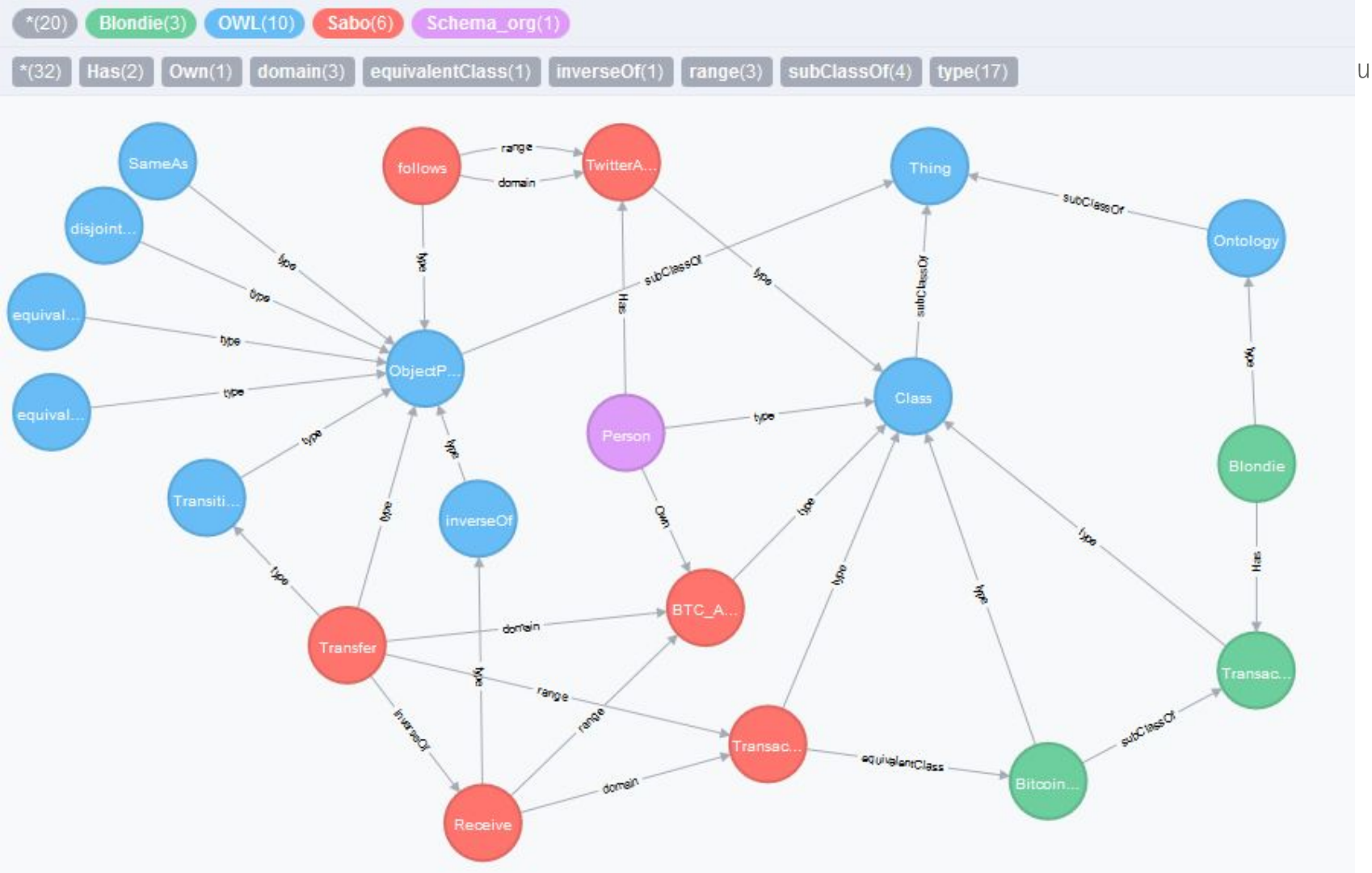
Mapping

IS=<O,S,M>

3 Vocabulary

- RDF
- RDFs
- Schema.org
- OWL
- Blondie
- Sabo

universitätbonn



universitätbonn

4

Semantic Blockchain

Let's start with the first set of slides

Why Neo4j?

Intuitive

Using a graph model for data representation.

Massively scalable

Native graph storage

Up to several billion nodes/relationships/properties

Expressive

With a powerful, human readable graph query language (Cypher)

Fast

With a powerful traversal framework for high-speed graph queries

Simple

Accessible by a convenient REST and Web interface

Why Neo4j?

Intuitive

Using a graph model for data representation.

Massively scalable

Native graph storage

Up to several billion nodes/relationships/properties

Expressive

With a powerful, human readable graph query language (Cypher)

Fast

With a powerful traversal framework for high-speed graph queries

Simple

Accessible by a convenient REST and Web interface

universitätbonn

5 Future Work

Let's start with the first set of slides

universitätbonn

Blockchain Ontology Survey

melvincarvalho/crypto-currency-ontologies

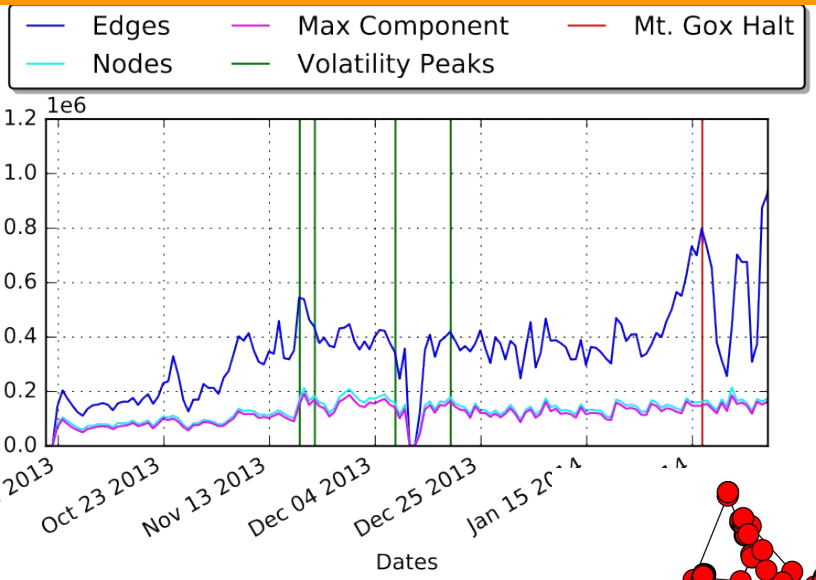
Go to the ***File*** menu and select ***Make a copy***.

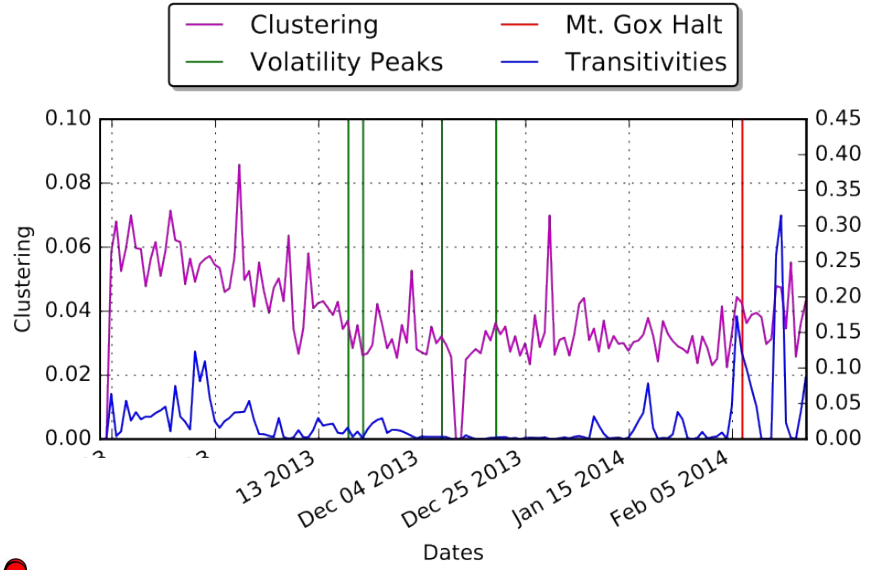
You will get a copy of this document on your Google Drive and will be able to edit, add or delete slides.

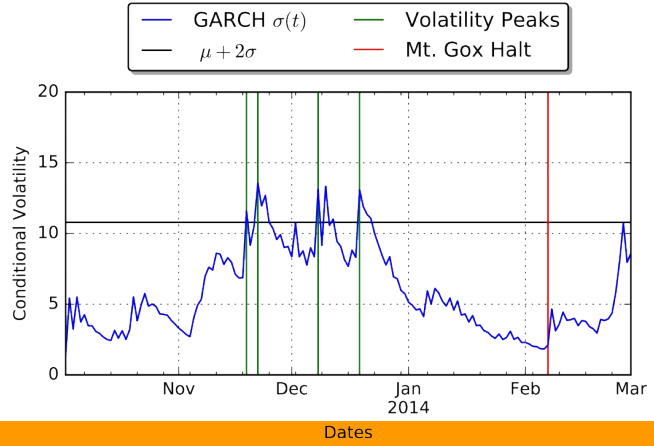
BLONDiE
BLOCKCHAIN ONTOLOGY WITH DYNAMIC EXTENSIBILITY

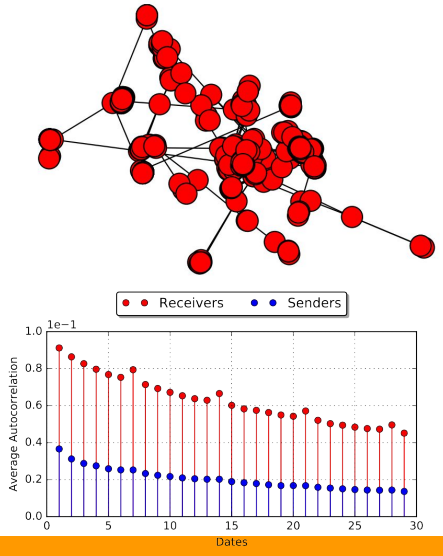
Blockchain Ontology with Dynamic Extensibility (BLONDiE)

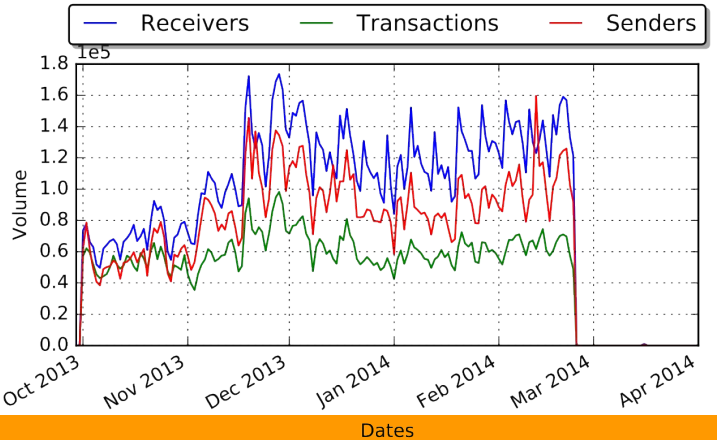
Go to the ***File*** menu and select ***Download as Microsoft PowerPoint***. You will get a .pptx file that you can edit in PowerPoint.


Edges
Nodes
Max Component
Volatility Peaks
Mt. Gox Halt
1e6
Count
Dates
Oct 02 2013
Oct 23 2013
Nov 13 2013
Dec 04 2013
Dec 25 2013

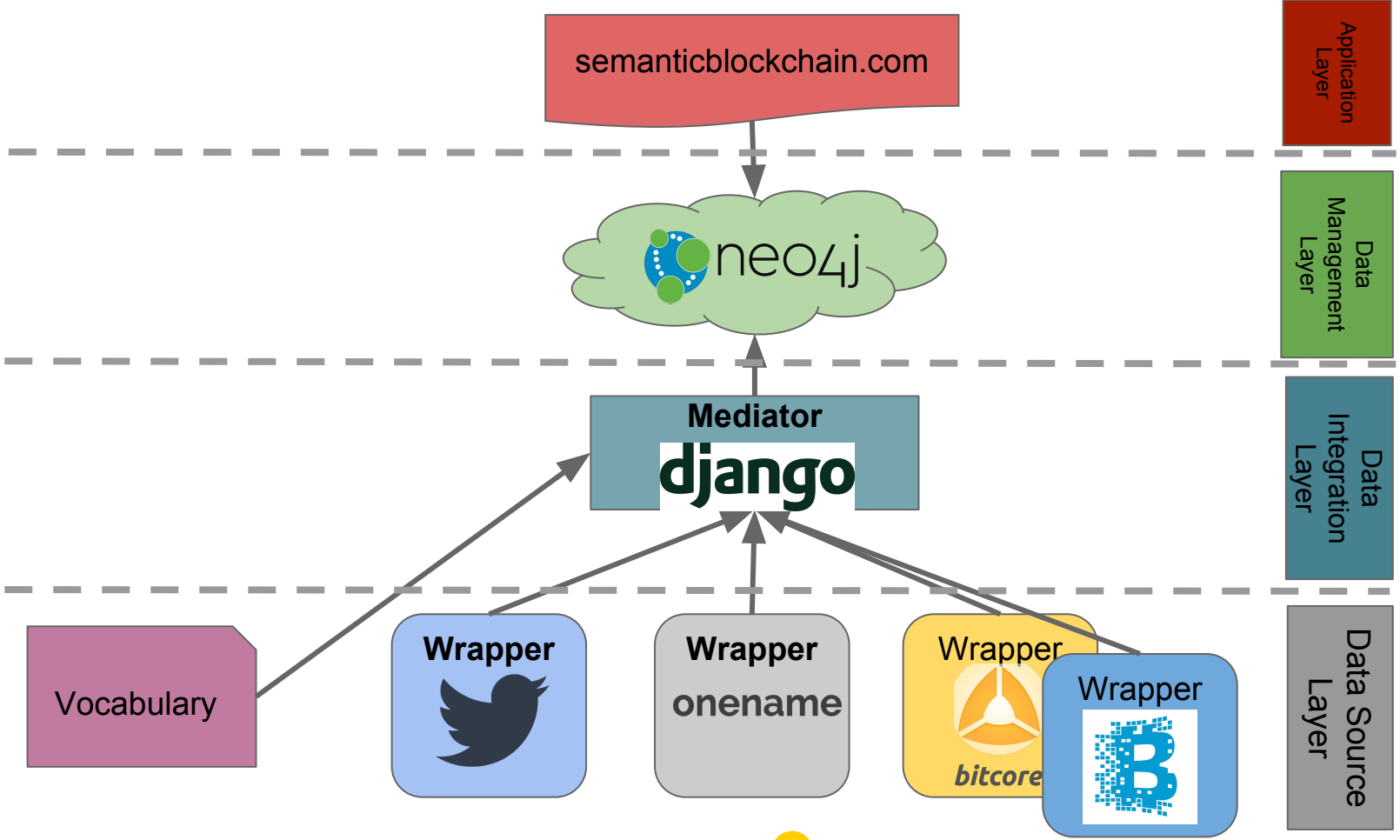

Clustering
Volatility Peaks
Mt. Gox Halt
Transitivities
Clustering
Dates
Dec 04 2013
Dec 25 2013
Jan 15 2014
Feb 05 2014


GARCH $\sigma(t)$
$\mu + 2\sigma$
Volatility Peaks
Mt. Gox Halt
Conditional Volatility
Nov
Dec
Jan
2014
Feb
Mar
Dates


Receivers
Senders
1e−1
Average Autocorrelation
Dates

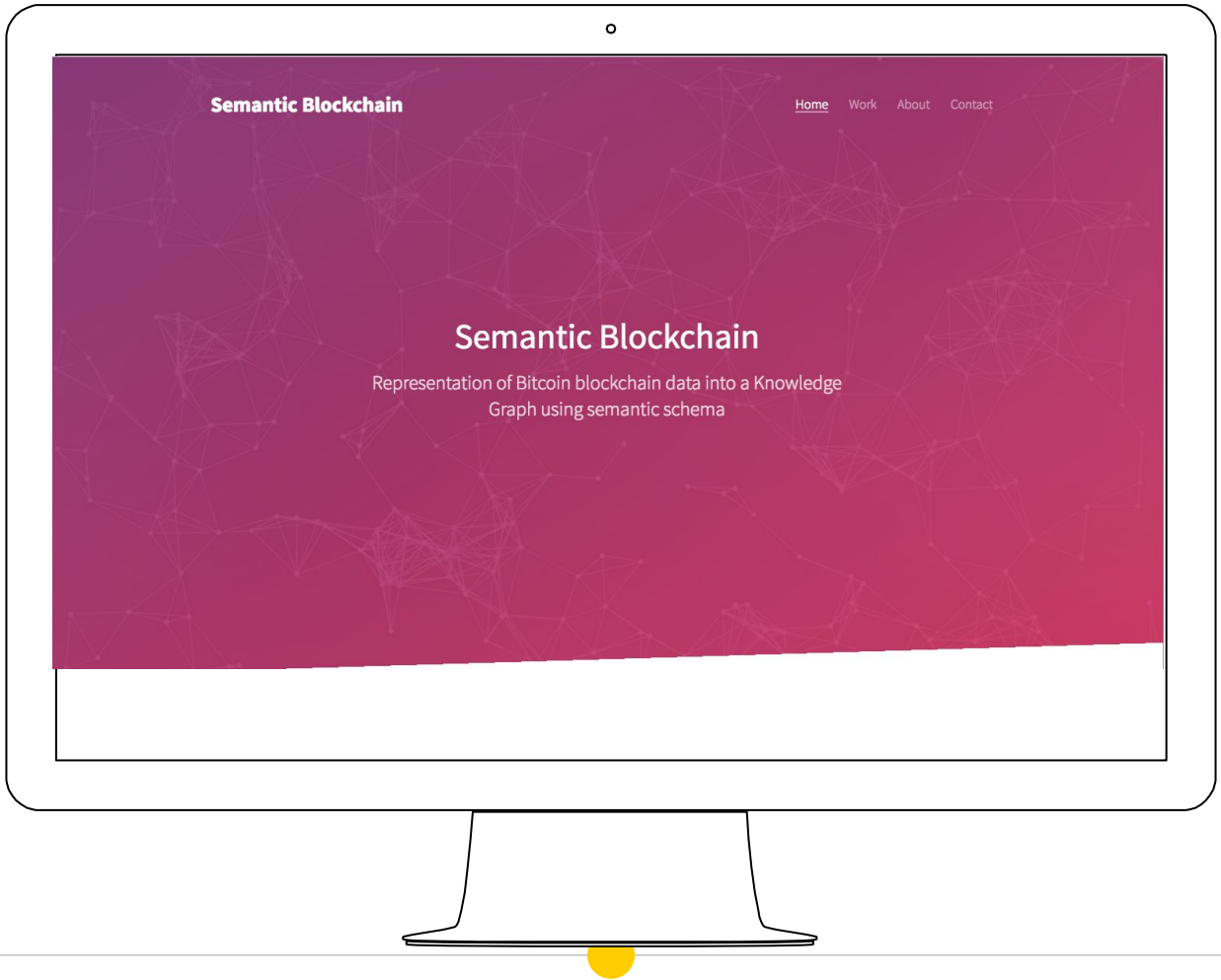

Receivers
Transactions
Senders
1e5
Volume
Oct 2013
Nov 2013
Dec 2013
Jan 2014
Feb 2014
Mar 2014
Apr 2014
Dates

6 Demo & Architecture



Software Architecture Diagram

universitätbonn

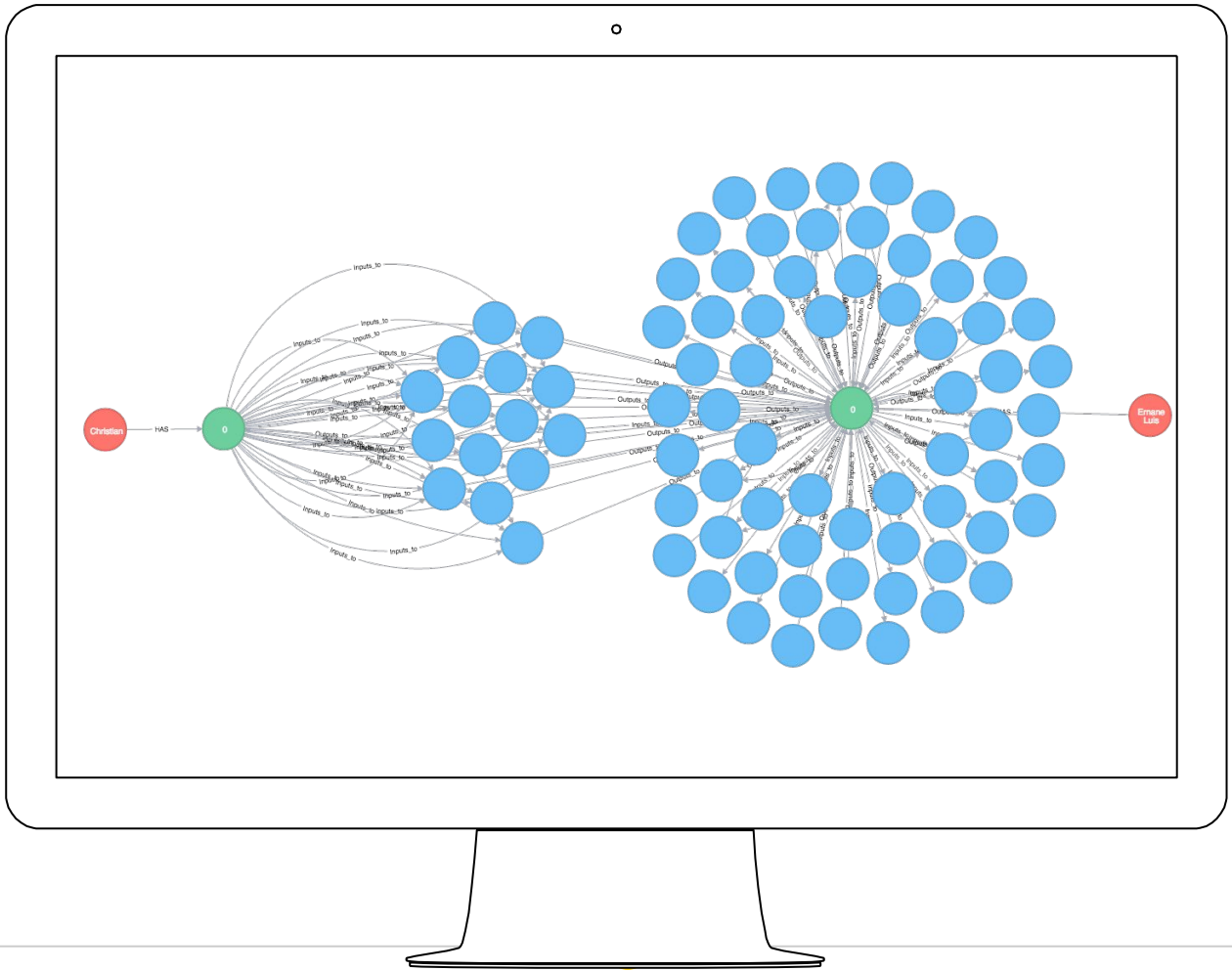


Website: *www.semanticblockchain.com*

universitätbonn

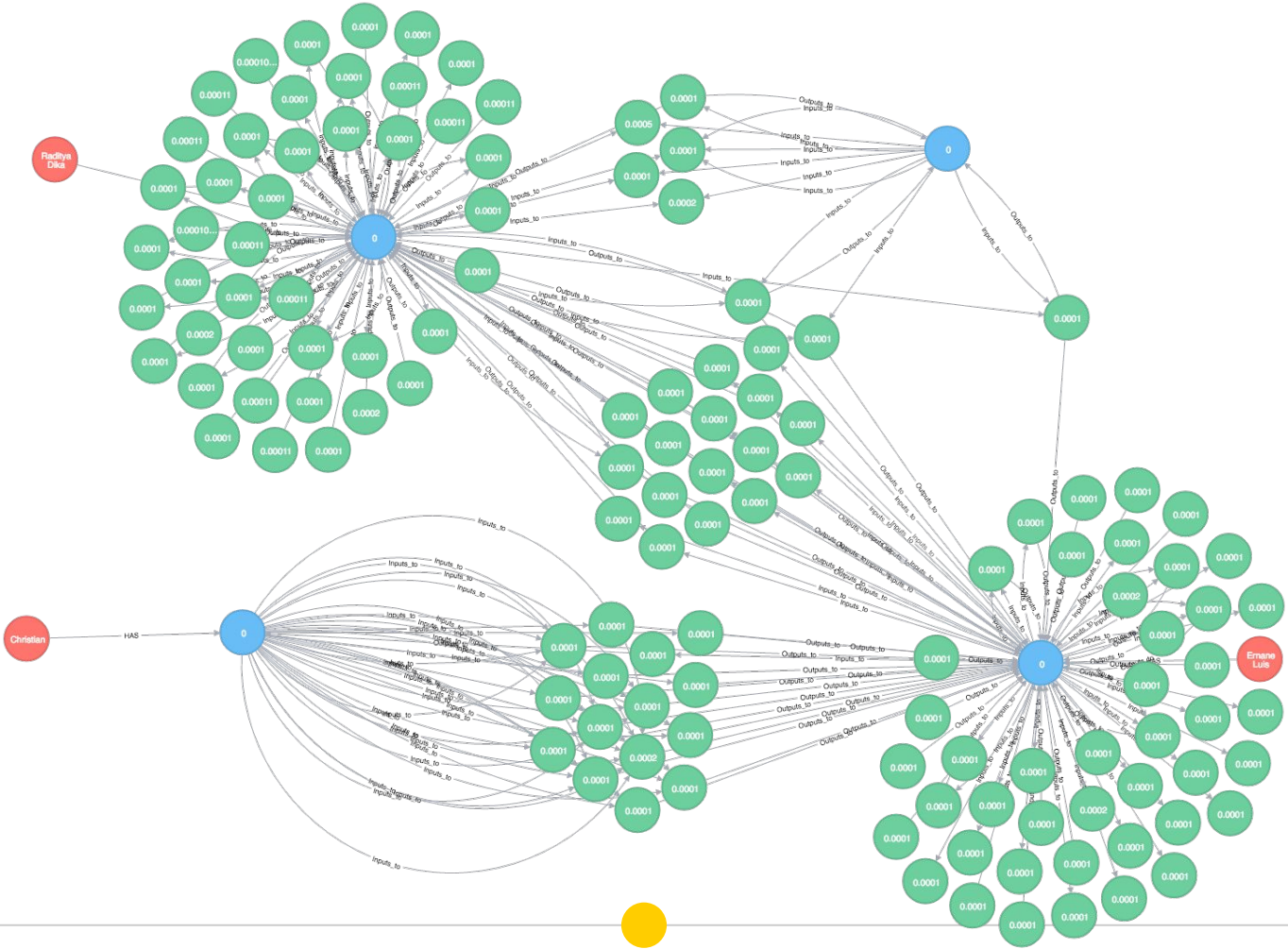
```
MATCH (n:BTC_Address)-[r:Inputs_to]->(m:Transaction)
MATCH (m:Transaction)-[x:Outputs_to]->(p:BTC_Address)
where p<>n AND m.hsh='2e9f40d4af66964a526d51233f87c401eaf66e
Optional MATCH (a:Person)-[h:HAS]->(p)
Optional MATCH (a:Person)-[has:HAS]->(n)
RETURN n,r,m,x,p,a,h,has
LIMIT 50;
```

Demo of a Transversal query



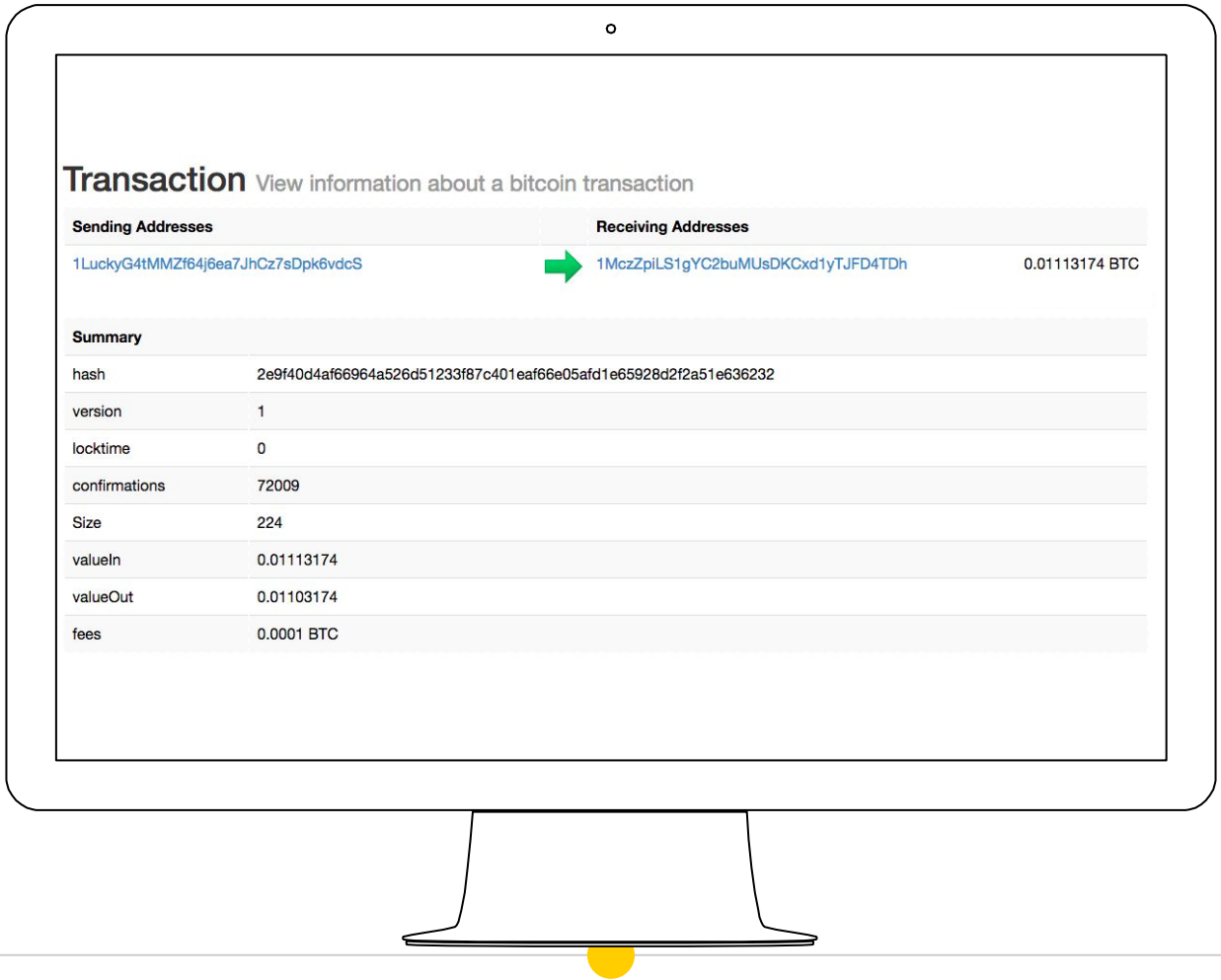
Demo of a Transversal query *Green: BTC Address* *Blue: Transaction* *Red: Person*

universitätbonn

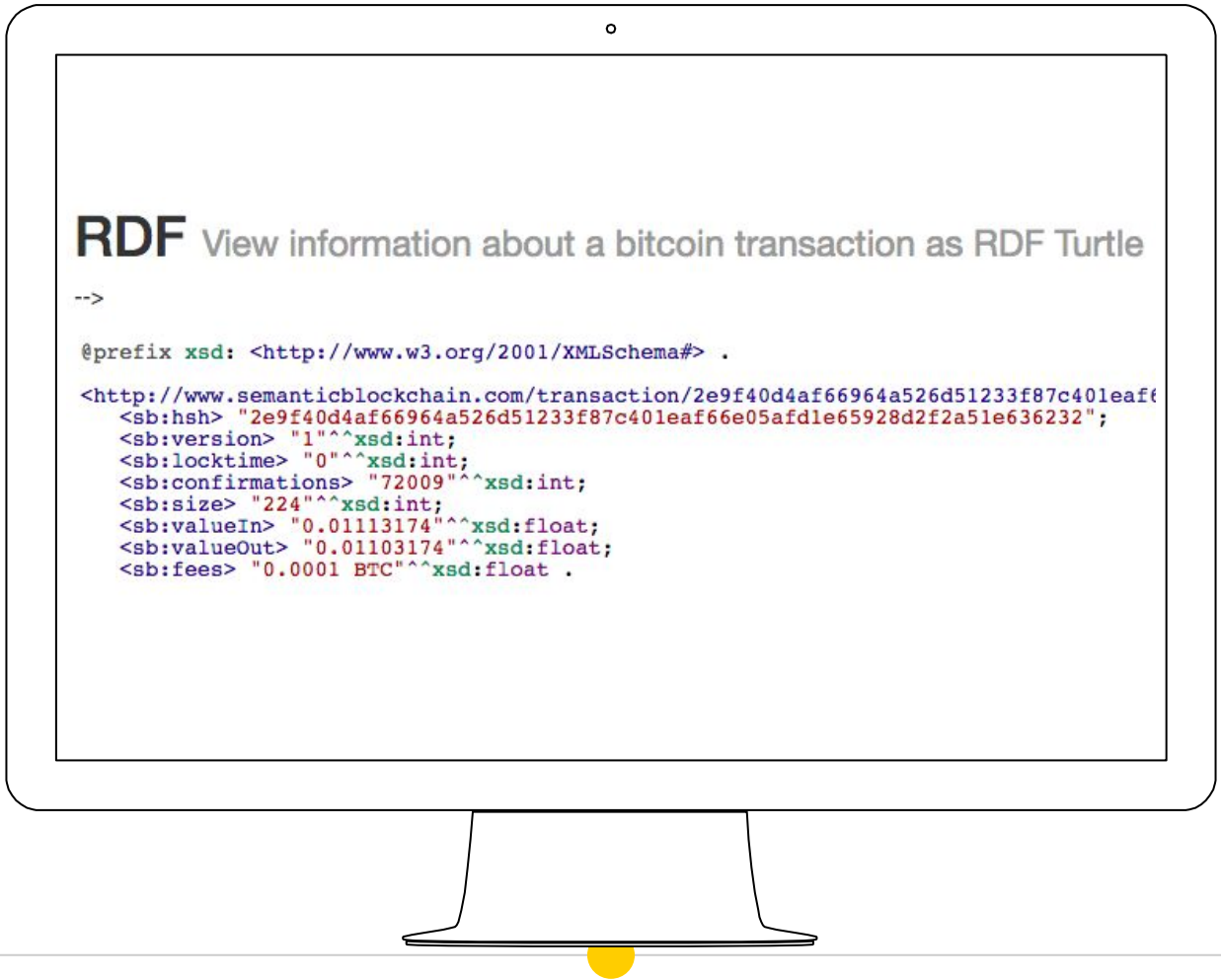


Demo of a Transversal query **Blue**: *BTC Address* **Green**: *Transaction* **Red**: *Person*

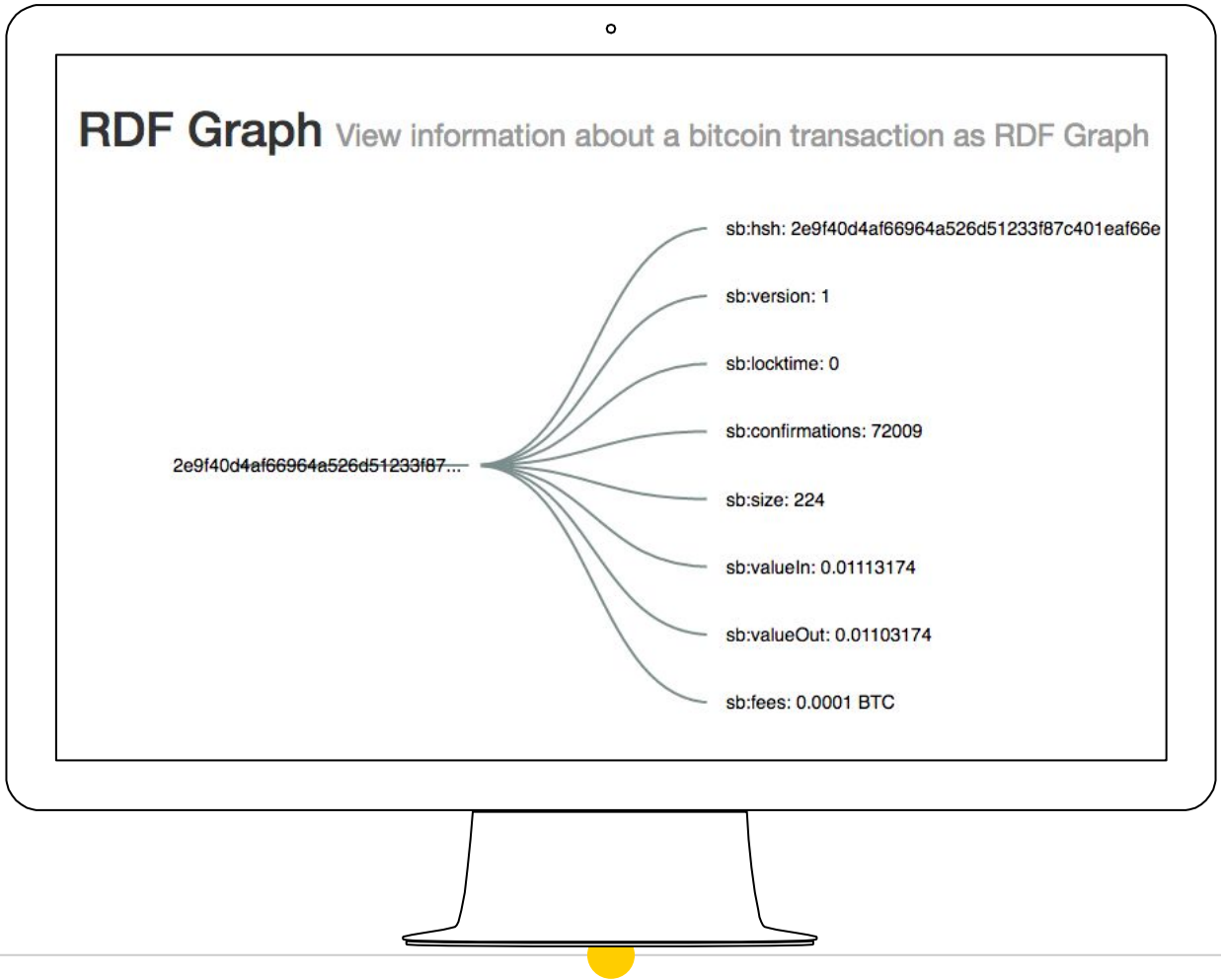
universitätbonn



Demo of a Transaction Data described by RDFa



Demo of a Transaction Data described by RDF Turtle



Demo of a Transaction Data described by RDF Graph

universitätbonn

Thanks!

Any questions ?