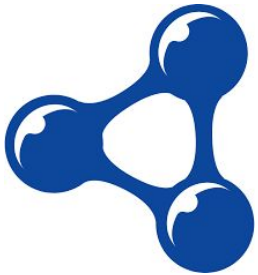
TrustFlows

Pieter Colpaert and Ben De Meester

We established a vision on **publishing** and **reading** data

Data Transfer



RDF, IRIs
Vocabularies

SHACL
or ShEx

Application
profiles

Rules
and
ontologies

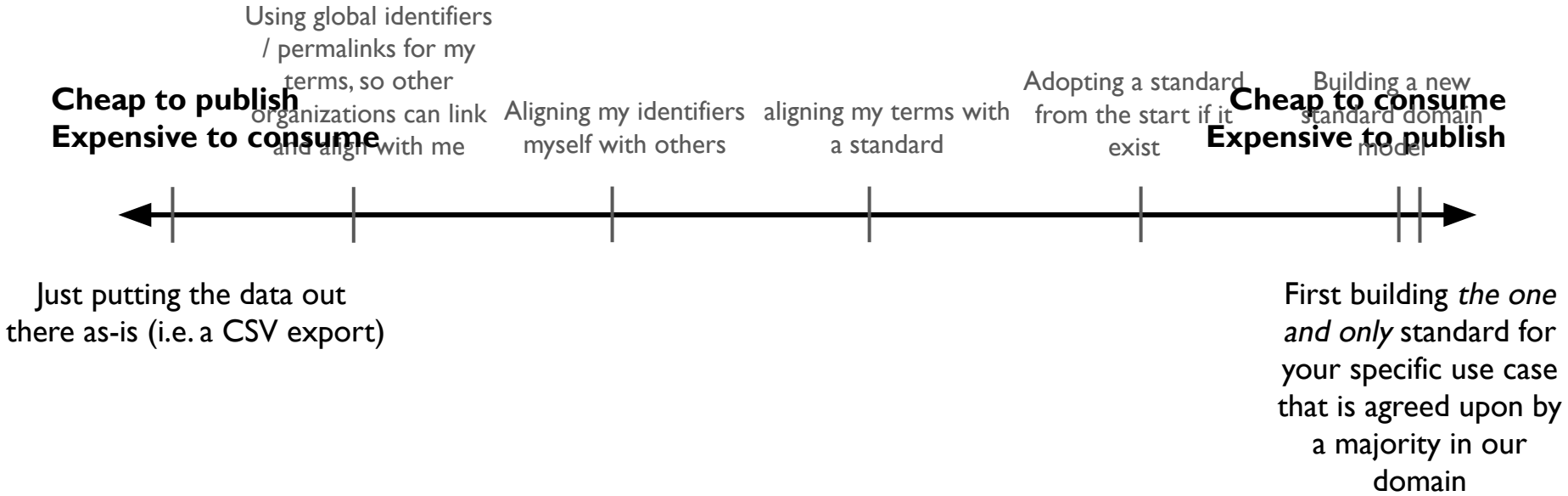
Logic

Enabling **semantic interoperability** on web-scale

Data engineering is making trade-offs

Linked Data enables choice

We established a vision on **publishing** and **reading** data

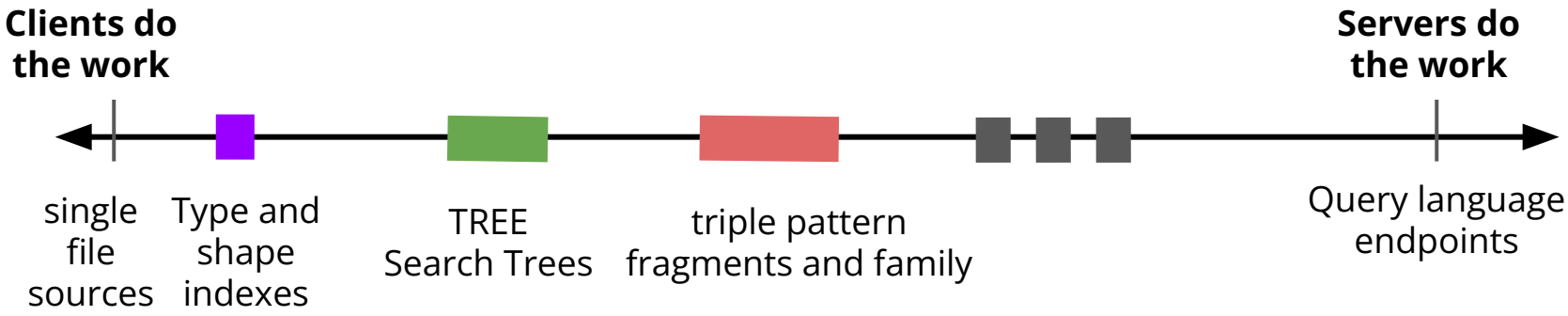


Choice in how we build consensus

We established a vision on **publishing** and **reading** data

Choice in what kind of question answering we want to stimulate

We established a vision on **publishing** and **reading** data



enabled thanks to **Comunica** and our **hypermedia specs**

We established a vision on **reading** data

And thanks to Solid, this works as well beyond open data

But how do we **write**?

Ruben Verborgh 16:02

Agree or disagree?

Data is just a number.
Triples are data.
Triples are just numbers.

Hence, all triples already exist.

But…
Why would we then
write anything anywhere at all?

We **don’t** write **data**

We write **trust**

And we should be explicit about it

DON’T

describe how one particular resource needs to change

Using HTTP PATCH with a N3 Patch body.

```
_:rename a solid:InsertDeletePatch;
  solid:where   { ?person ex:familyName "Garcia". };
  solid:inserts { ?person ex:givenName "Alex". };
  solid:deletes { ?person ex:givenName "Claudia". }.
```

This N3 Patch instructs to rename *Claudia Garcia* into *Alex Garcia*, on the condition that no other Garcia family members are present in the target RDF document.

https://solidproject.org/TR/protocol#n3-patch

LET'S

describe the change that happened in the real-world

Using HTTP POST to an inbox

```
_:rename a ex:EidReading ;
         ex:signature _:signature ;
         ex:payload _:payload .

_:payload {
    <#me> ex:givenName "Alex" ;
          …
}

_:signature {
   …
}
```

Welke stappen zijn er?

In Gent moet je hiervoor **persoonlijk** langskomen bij **Burgerzaken - Geboorte**, met **afspraak ↗**.

Je vult aan het loket de verklaring van voornaamswijziging in.

De ambtenaar van de burgerlijke stand beoordeelt je verzoek. Geeft de nieuwe voornaam bijvoorbeeld geen aanleiding tot verwarring, is de nieuwe voornaam niet belachelijk? Wat zijn je gerechtelijke antecedenten?

Bij twijfel kan de ambtenaar van de burgerlijke stand **advies** inwinnen via de procureur des Konings. Dat advies is niet bindend, maar moet wel binnen de 3 maanden worden gegeven.

De ambtenaar van de burgerlijke stand neemt binnen de 3 maanden een beslissing.

- Bij een **positieve** beslissing past de burgerlijke stand
  - alle akten aan waarin je naam wordt vermeld, bijvoorbeeld je geboorteakte, huwelijksakte, geboorteakte van je kinderen
  - je persoonsgegevens in het rijksregister aan

Je krijgt een uitnodiging om een nieuwe identiteitskaart aan te vragen.

- Bij een **negatieve** beslissing krijg je een gemotiveerde kennisgeving thuis

Write processes already exist

https://stad.gent/nl/burgerzaken/identiteit/persoonsgegevens-bekijken-wijzigen/voornaam-wijzigen

Welke stappen zijn er?

In Gent moet je hiervoor **persoonlijk** langskomen bij **Burgerzaken - Geboorte**, met **afspraak ↗**.

Je vult aan het loket de verklaring van voornaamswijziging in.

De ambtenaar van de burgerlijke stand beoordeelt je verzoek. Geeft de nieuwe voornaam bijvoorbeeld geen aanleiding tot verwarring, is de nieuwe voornaam niet belachelijk? Wat zijn je gerechtelijke antecedenten?

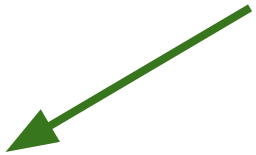
Bij twijfel kan de ambtenaar van de burgerlijke stand **advies** inwinnen via de procureur des Konings. Dat advies is niet bindend, maar moet wel binnen de 3 maanden worden gegeven.

De ambtenaar van de burgerlijke stand neemt binnen de 3 maanden een beslissing.

- Bij een **positieve** beslissing past de burgerlijke stand
  - alle akten aan waarin je naam wordt vermeld, bijvoorbeeld je geboorteakte, huwelijksakte, geboorteakte van je kinderen
  - je persoonsgegevens in het rijksregister aan

Je krijgt een uitnodiging om een nieuwe identiteitskaart aan te vragen.

- Bij een **negatieve** beslissing krijg je een gemotiveerde kennisgeving thuis



And they specify how multiple *read* documents will change as a consequence

https://stad.gent/nl/burgerzaken/identiteit/persoonsgegevens-bekijken-wijzigen/voornaam-wijzigen

LET'S

make those trust processes explicit

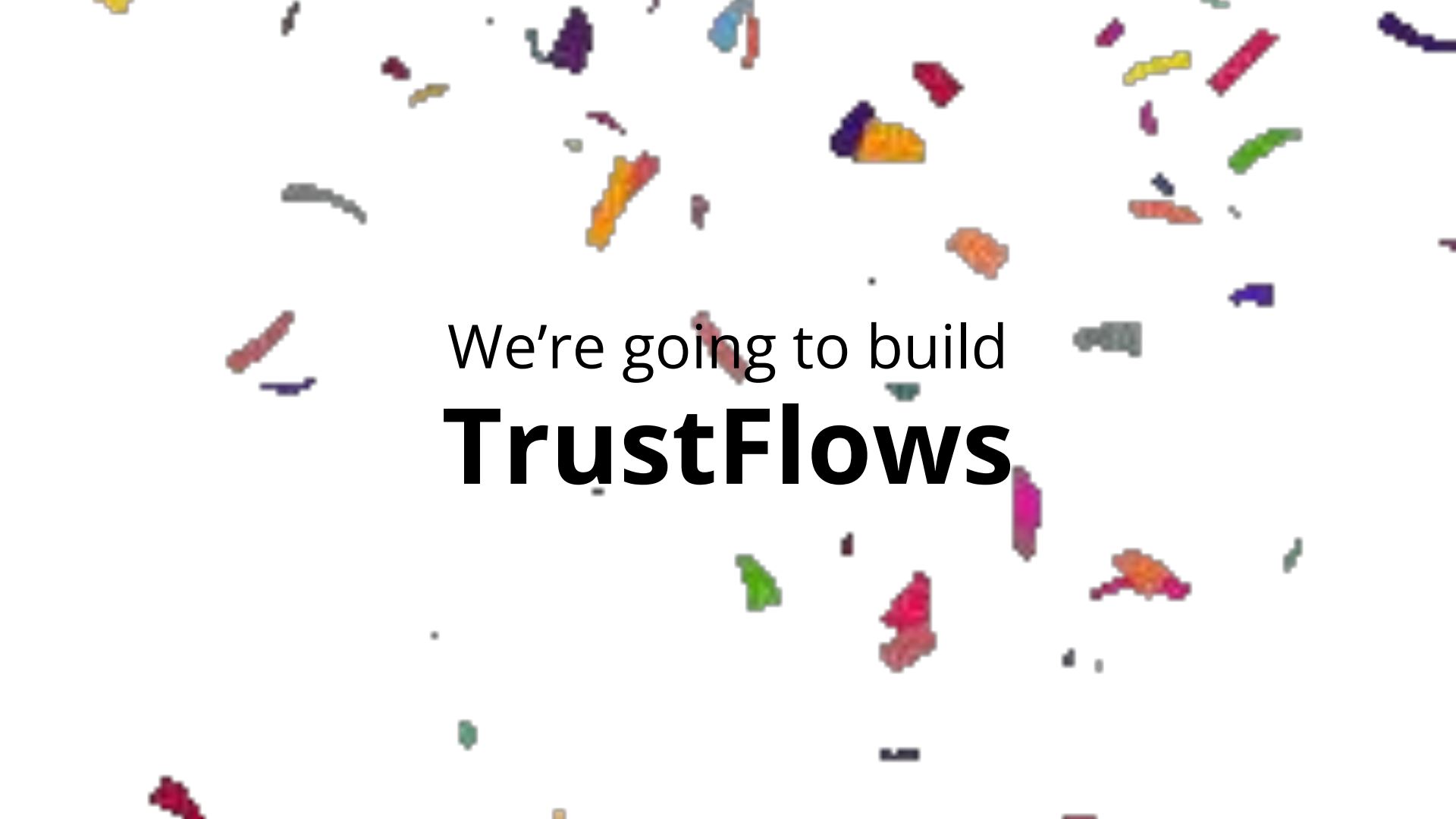
```
_:rename a ex:AcceptanceOfNameChange ;
         ex:acceptedBy <https://stad.gent/…> ;
         ex:acceptedAt "2024-10-22T12:34:00Z"^^xsd:dateTime ;
         ex:payload _:payload .

_:payload {
    <#me> ex:givenName "Alex" ;
          …
}

<https://stad.gent/…> ex:hasProofOfMandate …
```

Read and Write

Write to Read


We’re going to build
TrustFlows

TrustFlows

Is a **method**
resulting in **specifications**
influencing **architectures**

TrustFlows

Is a **method** for building data exchange projects

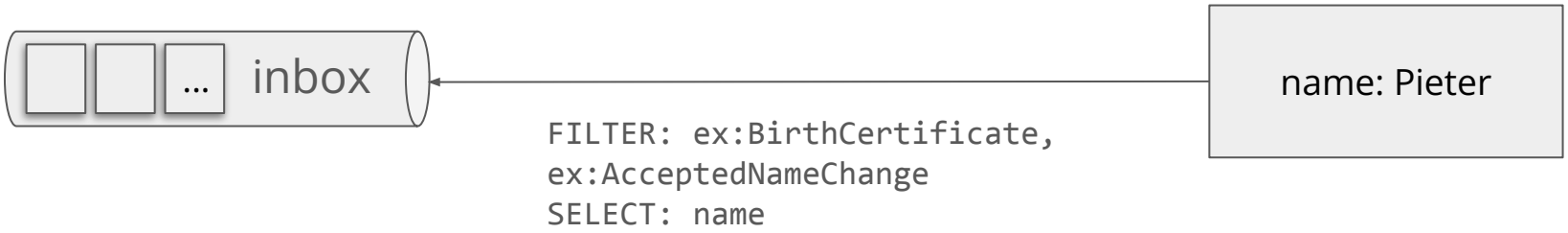
Specifying the domain model we use to **write**

For example, what words and schemas do we need to…

- as a person, request a name change
- as a government official, accept or reject a name change

Specifying how these **writes lead to the read**

For example, this read interface only *trusts* a name based on either a **birth certificate**, or **an accepted name change**.



TrustFlows

specifications
reflect business processes and conditions

There’s not one way to build a

TrustFlows

architecture

But those architectures will share a couple of ideas

We need to understand **identity** across multiple servers

Separation of concerns between **data processors** and **identity providers**

⇒ Can be achieved using WebID, Solid-OIDC, … and LWS

But those architectures will share a couple of ideas

We need to understand **policies** across multiple servers

Separation of concerns between **data processors** and **authorization servers**

white paper published at solidlab.be/white-papers

From Resource Control to Digital Trust with User-Managed Access

Author: WouterTermont, Ruben Dedecker, Wout Slabbinck, Beatriz Esteves, Ben De Meester, and Ruben Verborgh, SolidLab, IDLab, Ghent University – imec

The User-Managed Access (UMA) extension to OAuth 2.0 is a promising candidate for increasing Digital Trust in personal data ecosystems like Solid. With minor modifications, it can achieve many requirements regarding usage control and transaction contextualization, even though additional specification is needed to address delegation of control and retraction of usage policies.

Read the full paper here.

But those architectures will share a couple of ideas

We need to understand **policies** across multiple servers

⇒ **ODRL** is gaining traction as a way to express complex usage control policies

The authorization server will use a **policy engine** that can evaluate ODRL policies based on formal semantics of ODRL

https://w3c.github.io/odrl/formal-semantics/

We will need interoperable Trust Envelopes

```
_:rename a ex:EidReading ;
         ex:signature _:signature ;
         ex:payload _:payload .

_:payload {
    <#Ruben> ex:givenName "Ruben" ;
             ex:birthDate "1987-02-28";
          …
}

_:signature {
   …
}
```

We are applying this in our projects already

Ben De Meester

PACSOI

Personal health data in a **safe**, **trustworthy** and **scalable** manner

Solid to realize **decentralized patient-centric data storage** to break through healthcare and monitoring tools barriers and improve patient care and secondary use of data.

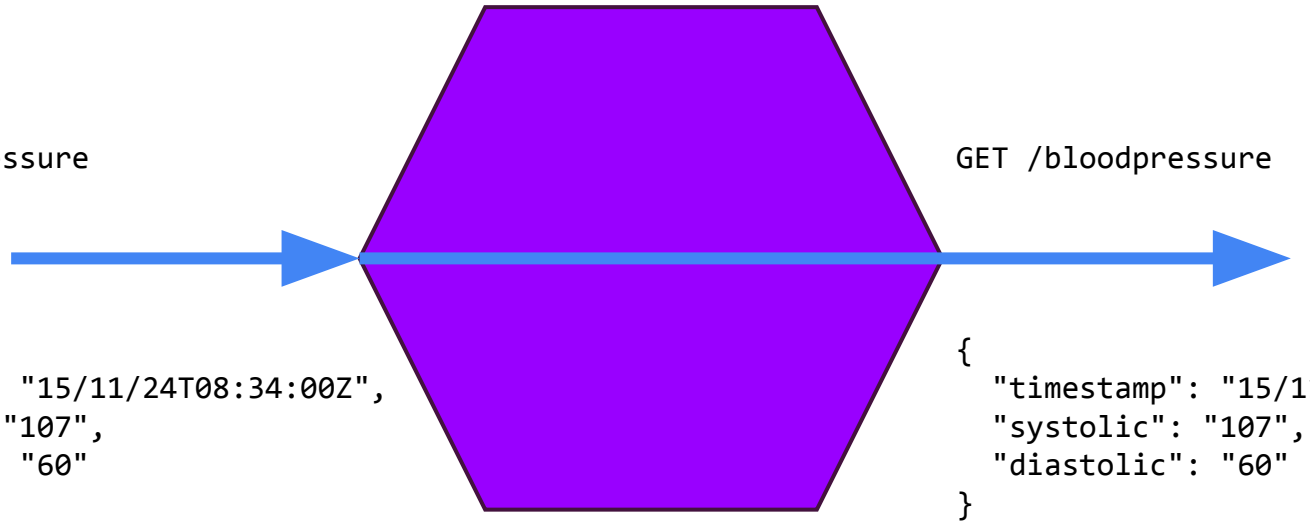
https://www.imec-int.com/en/research-portfolio/pacsoi

PACSOI
A Simple Use Case

I want to share
my blood pressure

Solid pod?

POST /bloodpressure



GET /bloodpressure

```
{
  "timestamp": "15/11/24T08:34:00Z",
  "systolic": "107",
  "diastolic": "60"
}
```

```
{
  "timestamp": "15/11/24T08:34:00Z",
  "systolic": "107",
  "diastolic": "60"
}
```

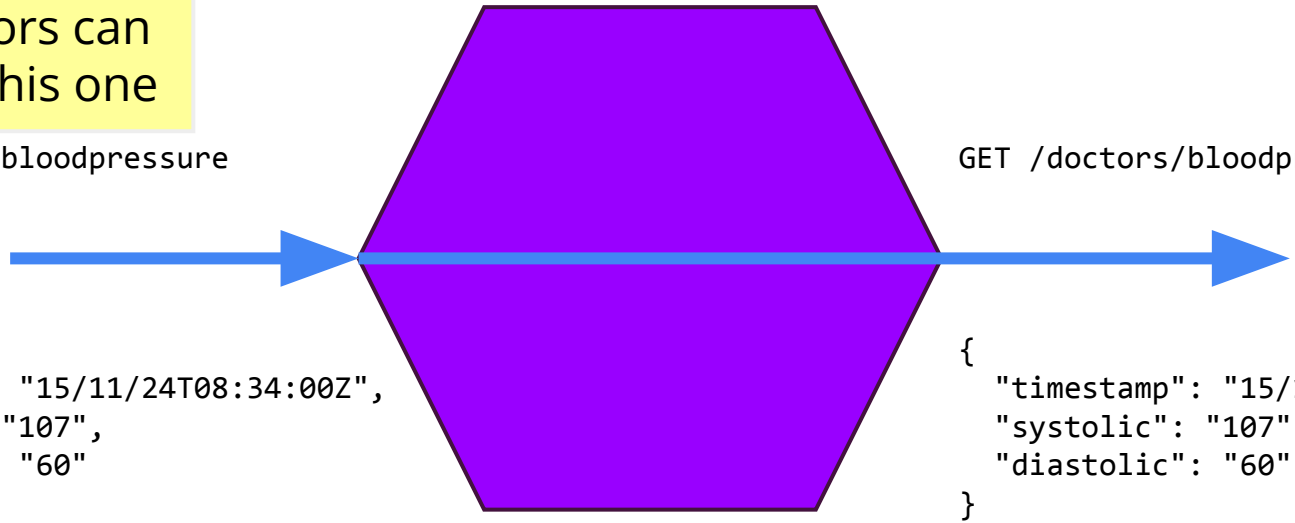
PACSOI
A Use Case

I want to share
my blood pressure
as measured by a doctor

Solid pod?

Only doctors can WRITE to this one

POST /doctors/bloodpressure



GET /doctors/bloodpressure

```
{
  "timestamp": "15/11/24T08:34:00Z",
  "systolic": "107",
  "diastolic": "60"
}
```

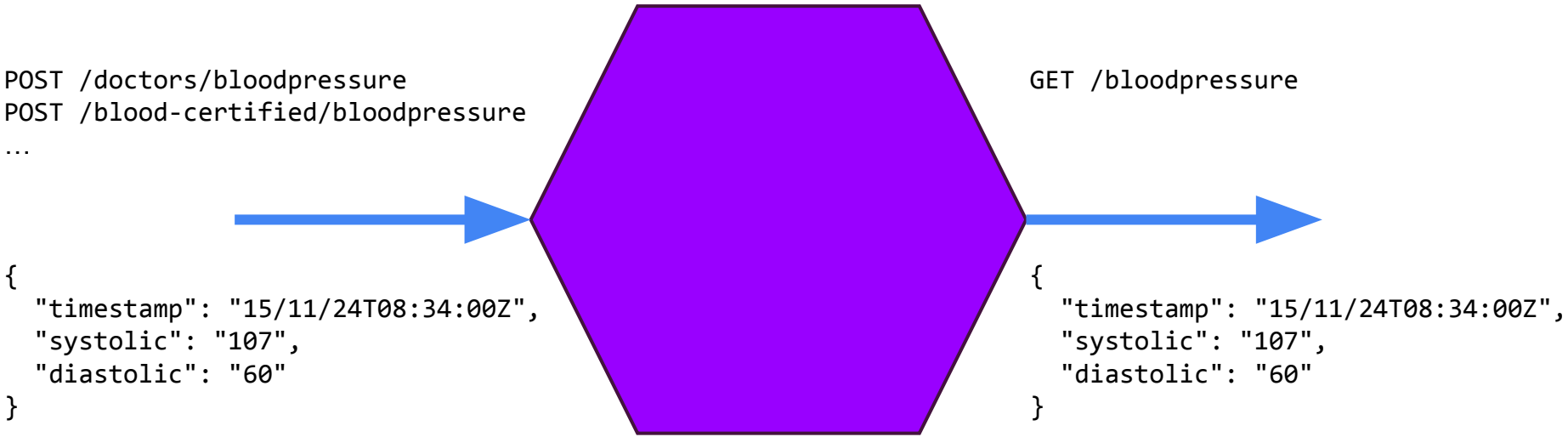
```
{
  "timestamp": "15/11/24T08:34:00Z",
  "systolic": "107",
  "diastolic": "60"
}
```

PACSOI
A Real Use Case

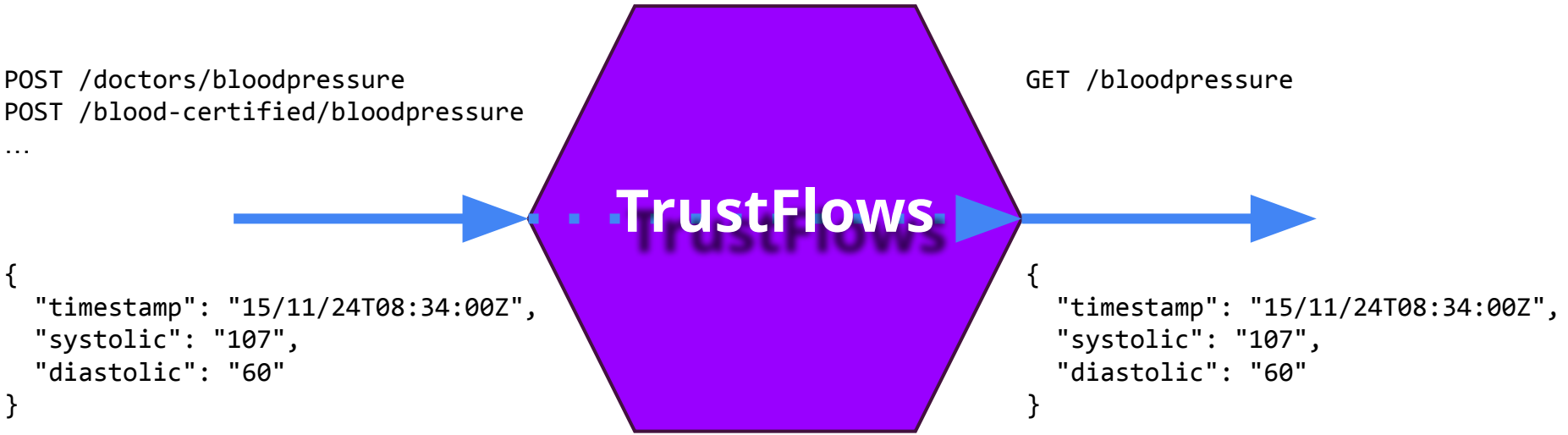
I want to share
my blood pressure
as measured by certified machines with doctors

Solid pod?

There's a discrepancy between
the trust during write, and
the trust during read



Solid pod?



Make trust explicit
in every write and read,
with Solid-compliant API calls

TrustFlows principles

Decouple between storage and authorization server

We can assume all actions are properly authorized, with sufficient context

Model trust processes, and create an RDF model

Besides the actual data model, we also need an activity model:
what actions can we perform on the data?

Establish pipelines from write to read

Transform and filter write actions into the corresponding read endpoints

TrustFlows principles

Decouple between storage and authorization server

- We can assume all actions are properly authorized, with sufficient context

Model trust processes, and create an RDF model

- **Besides the actual data model, we also need an activity model: what actions can we perform on the data?**

Establish pipelines from write to read

- Transform and filter write actions into the corresponding read endpoints

Model trust process
(for blood pressure measurements)

Data model (think: SHACL shapes)

- sensor values

Activity model

- create new sensor value

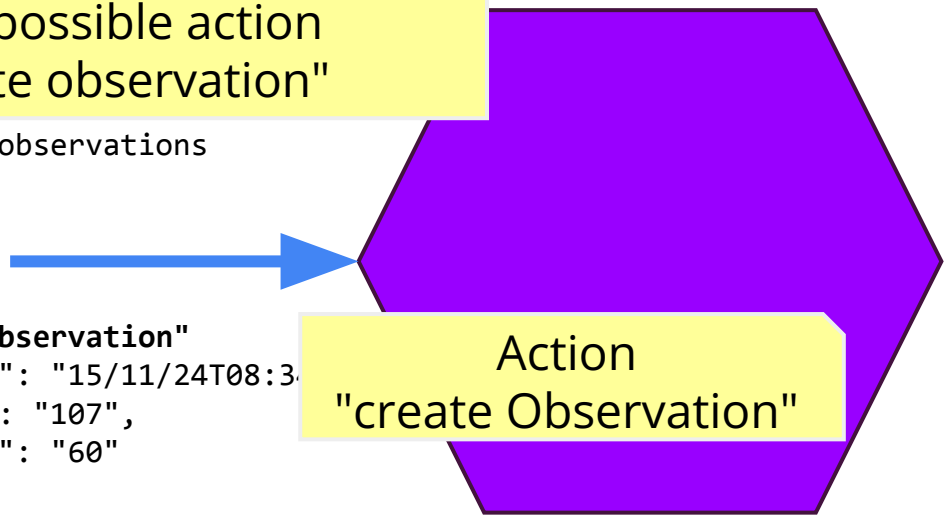
Solid pod with TrustFlows – writing

A WRITE inbox that covers the data model "observations", with possible action "create observation"

```
POST /inbox/observations
…


{
  "type": "Observation"
  "timestamp": "15/11/24T08:3
  "systolic": "107",
  "diastolic": "60"
}

Authorization context:
User: Doctor House
Scope: weekly-blood-pressure-
```



Only specific WRITEs are authorized

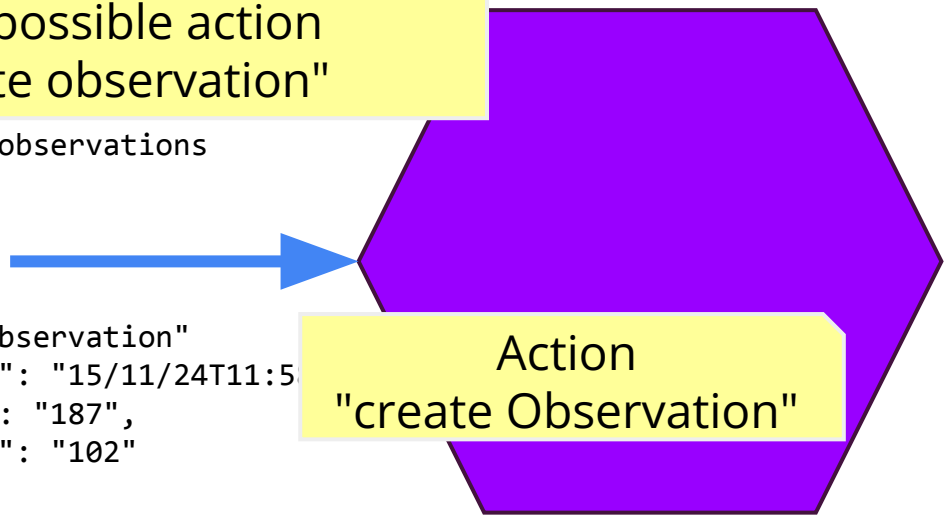
Solid pod with TrustFlows – writing (2)

A WRITE inbox that covers the data model "observations", with possible action "create observation"

```
POST /inbox/observations
…


{
  "type": "Observation"
  "timestamp": "15/11/24T11:5
  "systolic": "187",
  "diastolic": "102"
}

Authorization context:
User: John Doe
Scope: fitbit-blood-pressure
```



Only specific WRITEs are authorized

TrustFlows principles

Decouple between storage and authorization server

We can assume all actions are properly authorized, with sufficient context\

Model trust processes, and create an RDF model

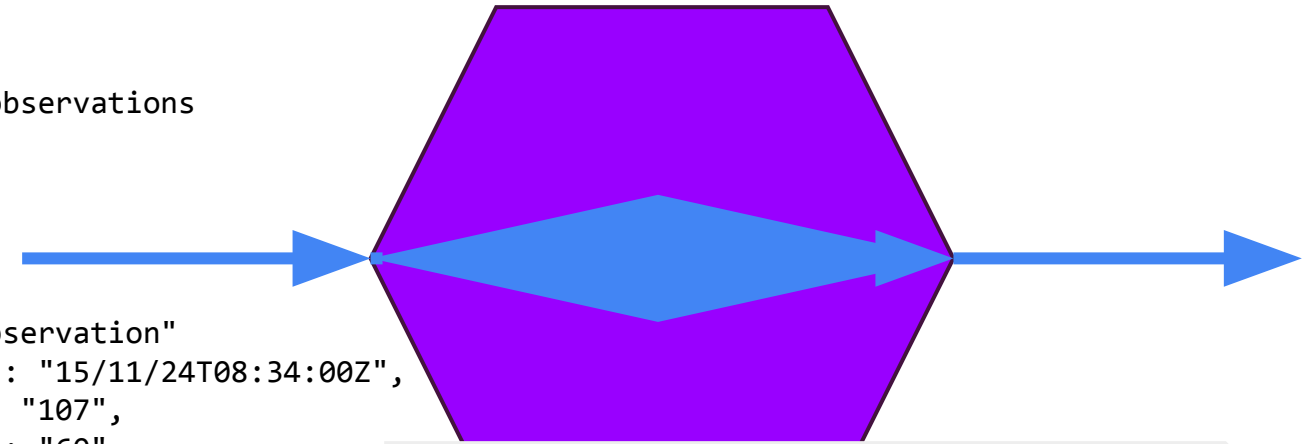
Besides the actual data model, we also need an activity model:
what actions can we perform on the data?

Establish pipelines from write to read

Transform and filter write actions into the corresponding read endpoints

Solid pod with TrustFlows - pipeline

```
POST /inbox/observations
...
```

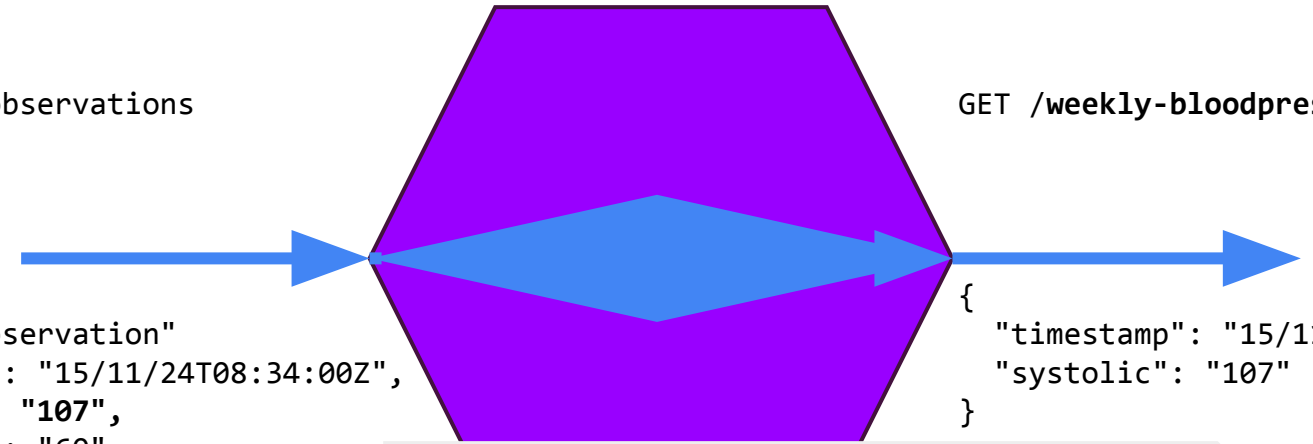


```
{
  "type": "Observation"
  "timestamp": "15/11/24T08:34:00Z",
  "systolic": "107",
  "diastolic": "60"
}

Authorization context:
User: Doctor House
Scope: weekly-blood-pressure-test
```

That input is made readable in multiple slices

Solid pod with TrustFlows - reading



```
POST /inbox/observations
...
```

```
{
  "type": "Observation"
  "timestamp": "15/11/24T08:34:00Z",
  "systolic": "107",
  "diastolic": "60"
}
```

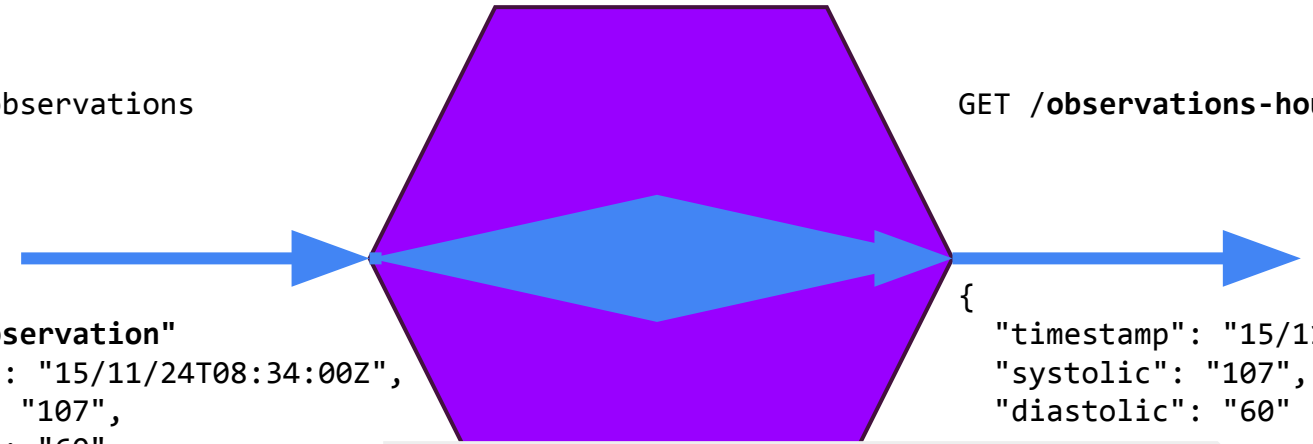
```
Authorization context:
User: Doctor House
Scope: weekly-blood-pressure-test
```

```
GET /weekly-bloodpressure
```

```
{
  "timestamp": "15/11/24T08:34:00Z",
  "systolic": "107"
}
```

That input is made readable in multiple slices, e.g.
‘**all systolic Observations of scope weekly-blood-pressure-test**’

Solid pod with TrustFlows - reading (2)



POST /inbox/observations
…

```
{
  "type": "Observation"
  "timestamp": "15/11/24T08:34:00Z",
  "systolic": "107",
  "diastolic": "60"
}
```

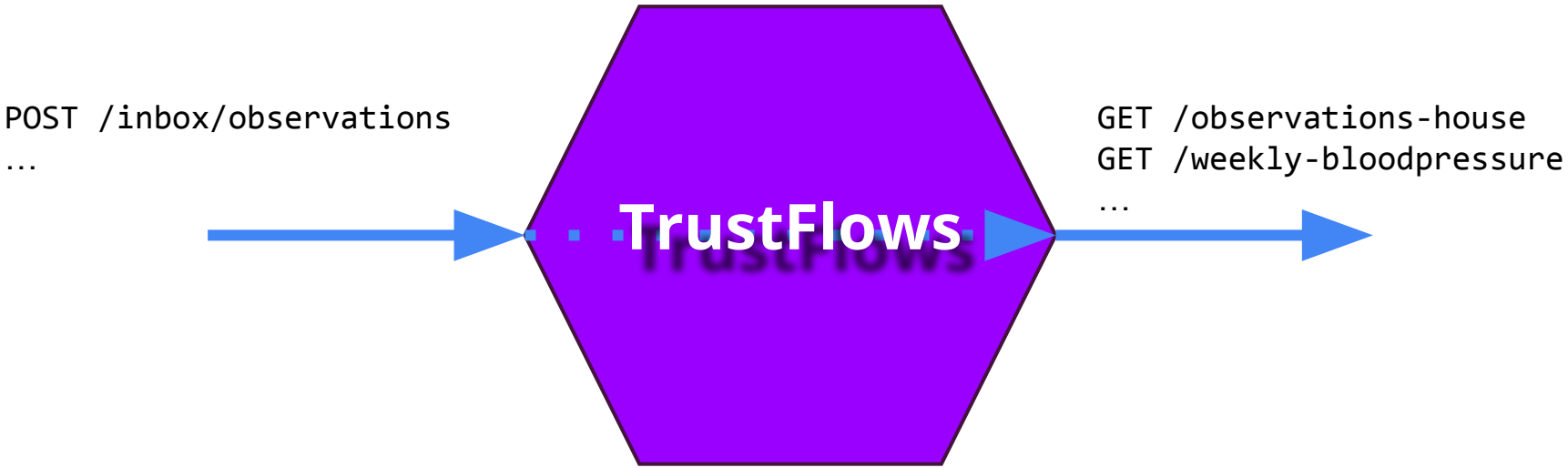
Authorization context:
User: Doctor House
Scope: weekly-blood-pressure-test

GET /**observations-house**

```
{
  "timestamp": "15/11/24T08:34:00Z",
  "systolic": "107",
  "diastolic": "60"
```

That input is made readable in multiple slices, e.g.
‘**all Observations of my current Medical Doctor**’

What changes for Solid applications?


POST /inbox/observations
…
TrustFlows
GET /observations-house
GET /weekly-bloodpressure
…

What changes for Solid servers?

Option 1: complement existing server implementations with satellite services that enrich the read endpoints based on the write inboxes [1]

Option 2: extend existing server implementations with built-in functionality [2]

[1] Jeroen Werbrouck (UGent) , Pieter Pauwels, Jakob Beetz, Ruben Verborgh (UGent) and Erik Mannens (UGent) (2024) SEMANTIC WEB. 15(2). p.429-460
[2] Early prototyping: Kvasir

Questions?
Feedback?

“This is not a question but more of a comment” responses?