RESEARCH DATA ALLIANCE

PID Information Types WG

Research Infrastructures meet RDA, Amsterdam, May 26, 2015

Tobias Weigel (DKRZ / University of Hamburg)
Tim DiLauro (Data Conservancy / Johns Hopkins University)

research data sharing without barriers
rd-alliance.org

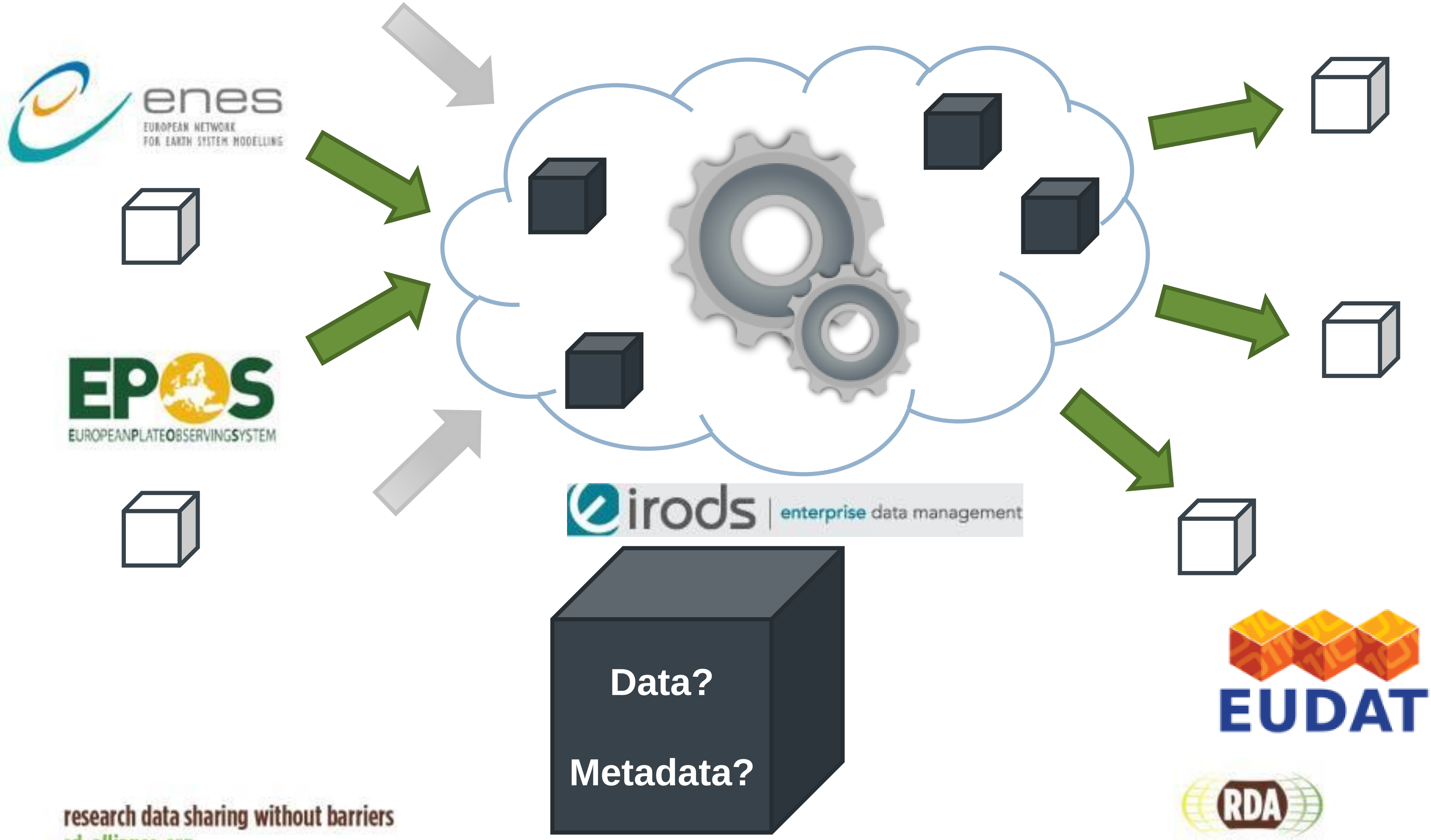
WG PID Information Types

- 18 months, RDA P1 to P4
- co-chairs: Tim DiLauro (JHU); Tobias Weigel (DKRZ)

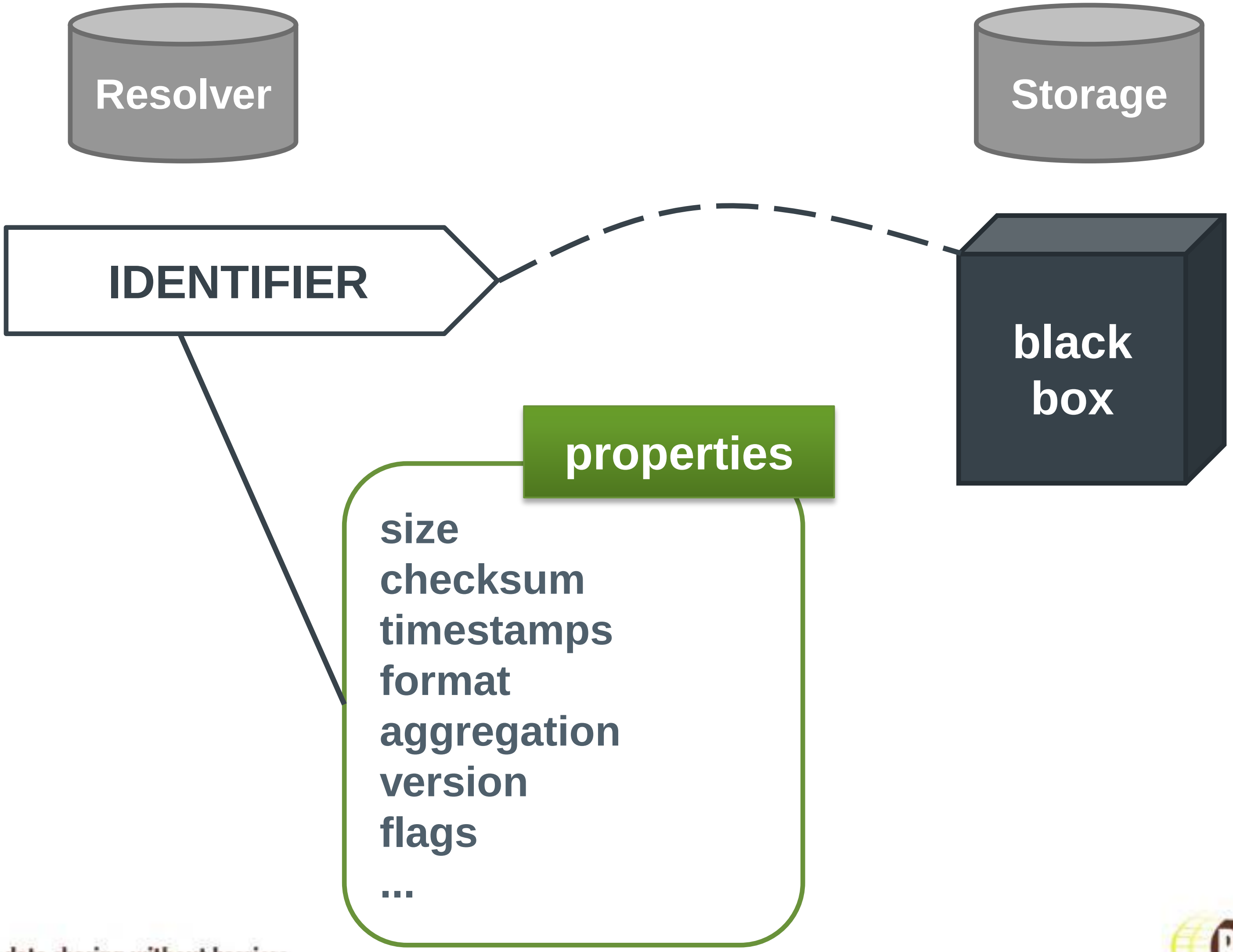
- Goal: Hamonization of basic information types associated with PIDs across disciplines and infrastructures
- Approach: Design an API and type examples to target practical usage

- Strong interaction with Data Type Registries WG

Example scenario


enes
EUROPEAN NETWORK
FOR EARTH SYSTEM MODELLING
EPOS
EUROPEANPLATEOBSERVINGSYSTEM
irods | enterprise data management
Data?
Metadata?
EUDAT

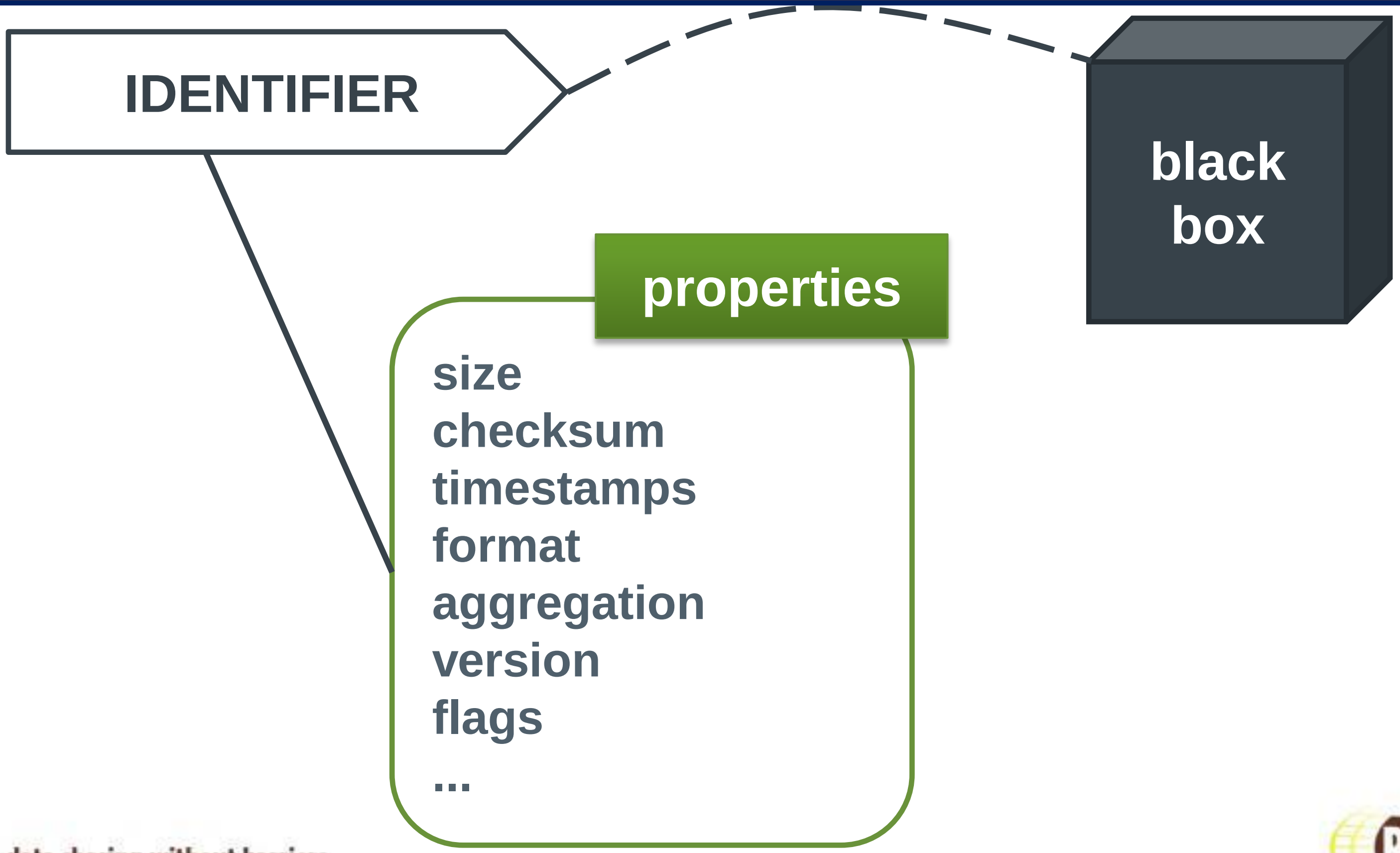
Types for information directly associated with PIDs



size
checksum
timestamps
format
aggregation
version
flags
...

Types for information directly associated with PIDs

A Persistent Identifier is a long-lasting ID represented by a string that uniquely points to a DO and that is intended to be persistently resolvable to access meaningful, current state information about the identified DO. *(from DFT wiki)*



IDENTIFIER

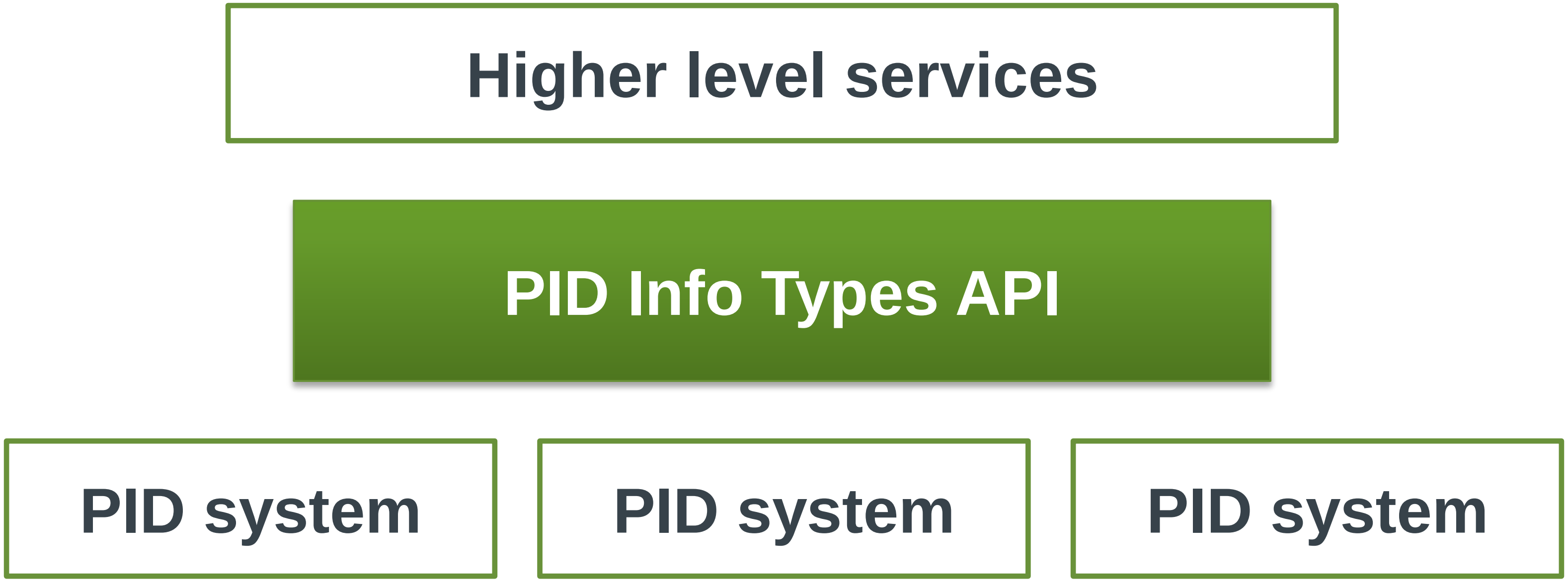
black box

properties

size
checksum
timestamps
format
aggregation
version
flags
...

Something needs to reach across PID systems

The PID Information Types API serves two purposes:
Facilitating **typing** and enabling **interoperability** across PID Systems.



Higher level services

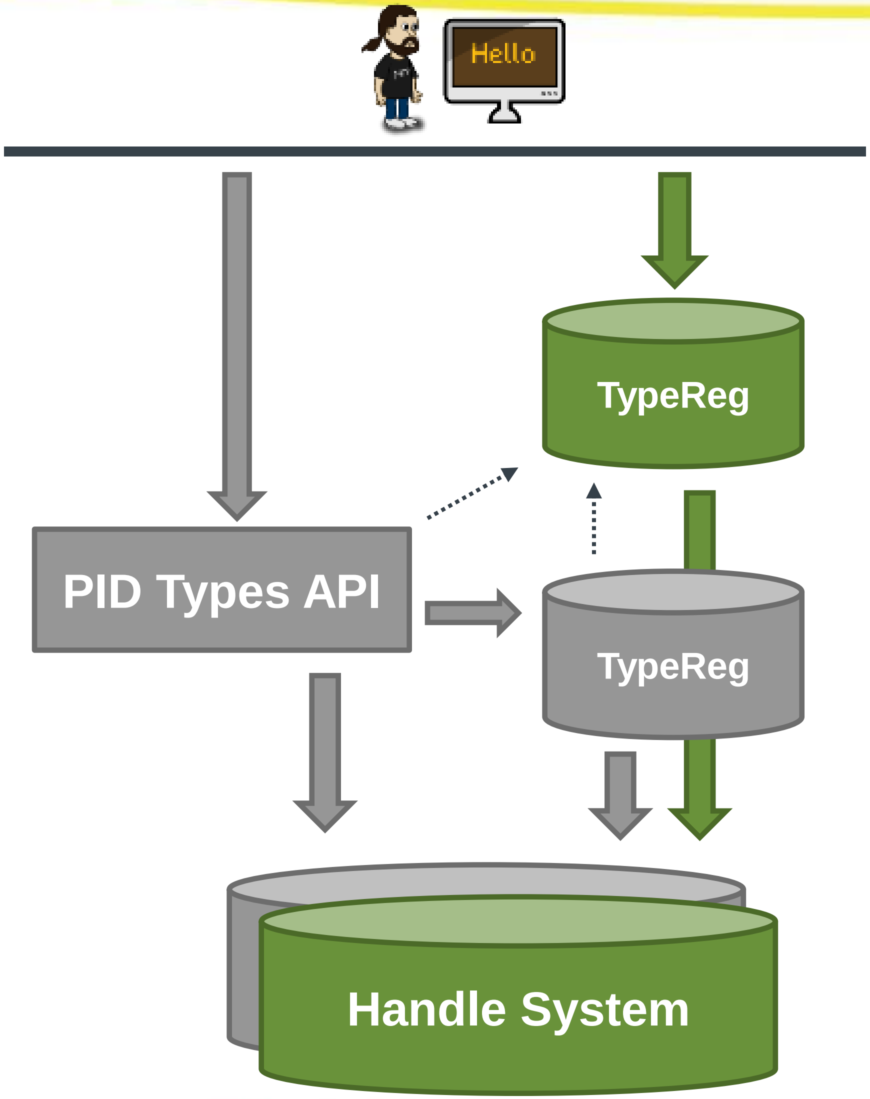
PID Info Types API

PID system

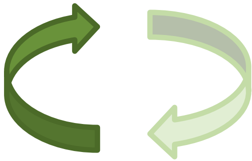
PID system

PID system

Typing and Type Registries



- Two usage scenarios for TypeReg:
  - **Typing of data entities**



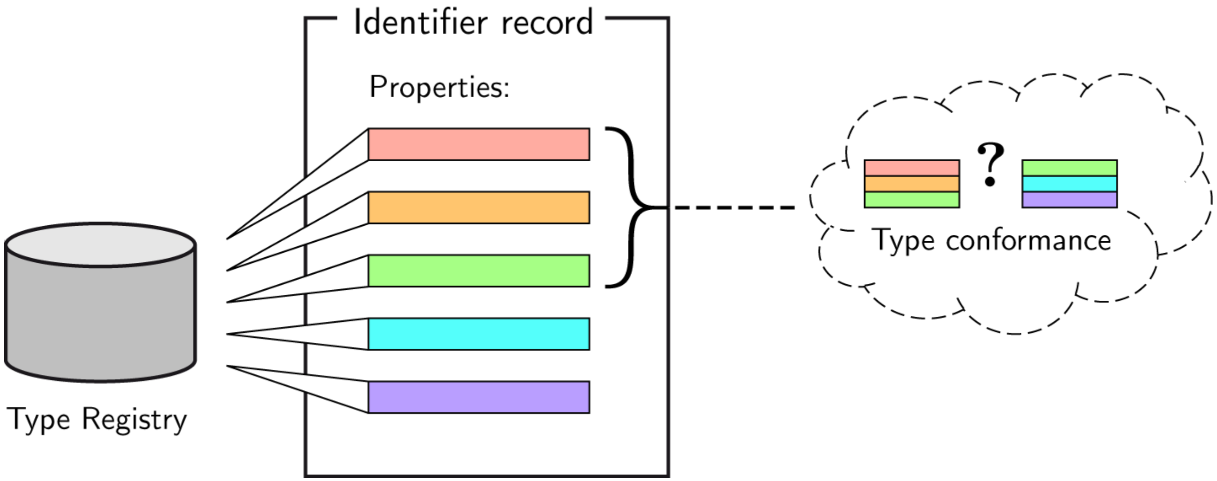
  - **Typing of PID record value fields**
- Reference to data type in properties record

Use cases: Applications for typed PID records

- Replication management
- Version management
- Provenance tracing
- Access control
- Composition
- …

Conformance queries reach beyond single properties

- A PID record may contain various properties, which however make up useful groups
  - Fixity (Format, Checksum, Size, ...)
  - Accessibility (Format, License, Owner, ...)
- A particular service may require a distinct set to be present



Example type list from the final report – to be continued...?

Name	Range	Identifier	Flags
Type: Citation Information (EXAMPLE namespace) 11314.2/d5396a97c316a0eaca055846ba4233ac			
Title	STRING	11314.2/07841c3f84cbe0d4ff8687d0028c2622	
Creator	STRING	11314.2/31810b2c24913929bb5e0d4d949de9f7	
Publication date	DATE	11314.2/daed5901fbbe2570ee95c4009c739de2	
Language	STRING	11314.2/56211d62153b3500ce3b16cf86d6b403	optional
License	STRING	11314.2/2f305c8320611911a9926bb58dfad8c9	optional
Type: System level access information (EXAMPLE namespace) 11314.2/09d35f22e48b60284029ba51c17e2944			
Creation date	DATE	11314.2/6b3e1230d1b68965e290b16a43d2f46d	
Deletion date	DATE	11314.2/7e78be9736ad7f6bb5fb31218821eba5	optional
Permissions	STRING	11314.2/d057258f7b406fd9aad5a3893aba8208	optional
Checksum	STRING	11314.2/56bb4d16b75ae50015b3ed634bbb519f	
Object size (in bytes)	STRING	11314.2/0006e2b8e2f6e1ecce836e593bed38ae	
Type: Aggregation information (EXAMPLE namespace) 11314.2/699d487eff50c2e10982f4b85ed053a9			
Parent object identifier	IDENTIFIER	11314.2/f9e66e5f64ba3179d8f1e64138c69e04	optional
Child object identifier	IDENTIFIER	11314.2/f8db9e3b5f97aa8168fbd59788476375	optional
Type: Versioning information (EXAMPLE namespace) 11314.2/6b507d787dd06e4eb8f23b5bb56ae8bb			
Predecessor identifier	IDENTIFIER	11314.2/467d9ba30e2d9879fd9d483f319e462c	optional
Successor identifier	IDENTIFIER	11314.2/fc78024cb9dac0b0a80ed631ea650d4b	optional
Type: Preliminary example for EUDAT core information (EUDAT namespace) 11314.2/5f45666fc8689e3565728ca512c1b5e7			
Checksum	STRING	see above	
Format	STRING	11314.2/1a4f53a28b72d4bf4f8fdda7a2089595	
Data identifier	IDENTIFIER	11314.2/24dd85c4a3d39fb0d7e83a510a5041c6	
Metadata identifier	IDENTIFIER	11314.2/58a44100d2bcd1a34fb87eb87bc6f701	
Repository of Record	IDENTIFIER	11314.2/5546b0166091d9ae869f081f5548f3fc	
Mutability flag	BOOLEAN	11314.2/7c81e954eaead6a2f772abd83986d3e9	
Landing page address	URL	11314.2/66af2639d388977e81b85f6413df1e2c	
Date of deposition	DATE	11314.2/35837218f18dcc54a2d32e0fb30fa7fb	

Brief overview of the actual API methods

- The API focuses on reading and making sense of typed PID record information.
- There are interfaces via Java and HTTP.

GET /peek/{identifier}

GET /property/{identifier}

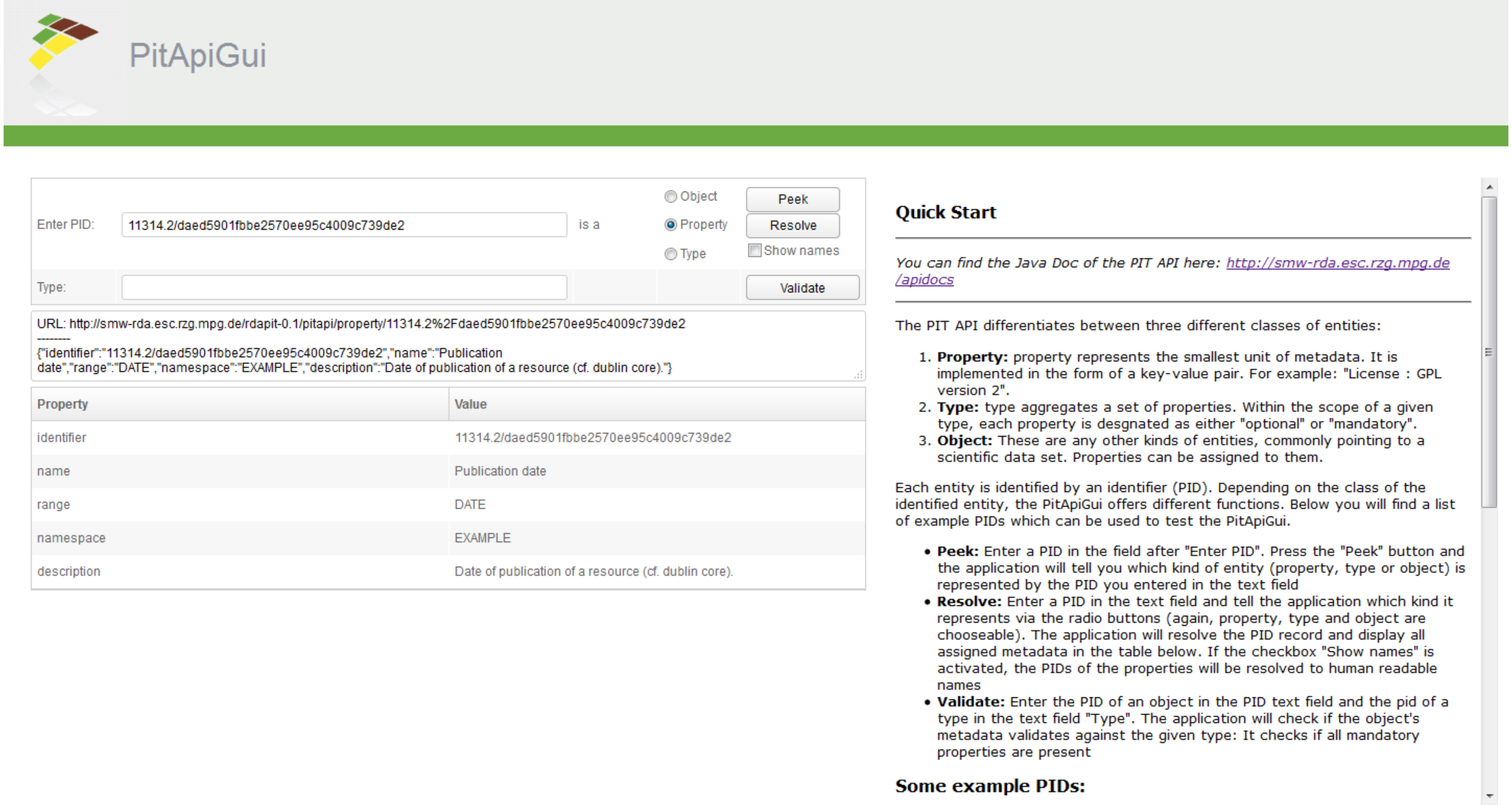
GET /type/{identifier}

GET /pid/{identifier}?...

- Conformance information included if Types are given

The PIT API demonstrator

- Thanks to Tom Zastrow, there is also a small demonstrator running at RZG.



			○ Object	Peek
Enter PID:	11314.2/daed5901fbbe2570ee95c4009c739de2	is a	● Property	Resolve
			○ Type	□ Show names
Type:				Validate

```
URL: http://smw-rda.esc.rzg.mpg.de/rdapit-0.1/pitapi/property/11314.2%2Fdaed5901fbbe2570ee95c4009c739de2
--------
{"identifier":"11314.2/daed5901fbbe2570ee95c4009c739de2","name":"Publication
date","range":"DATE","namespace":"EXAMPLE","description":"Date of publication of a resource (cf. dublin core)."}
```

Property	Value
identifier	11314.2/daed5901fbbe2570ee95c4009c739de2
name	Publication date
range	DATE
namespace	EXAMPLE
description	Date of publication of a resource (cf. dublin core).

Quick Start

You can find the Java Doc of the PIT API here: http://smw-rda.esc.rzg.mpg.de/apidocs

The PIT API differentiates between three different classes of entities:

1. **Property:** property represents the smallest unit of metadata. It is implemented in the form of a key-value pair. For example: "License : GPL version 2".
2. **Type:** type aggregates a set of properties. Within the scope of a given type, each property is desgnated as either "optional" or "mandatory".
3. **Object:** These are any other kinds of entities, commonly pointing to a scientific data set. Properties can be assigned to them.

Each entity is identified by an identifier (PID). Depending on the class of the identified entity, the PitApiGui offers different functions. Below you will find a list of example PIDs which can be used to test the PitApiGui.

- **Peek:** Enter a PID in the field after "Enter PID". Press the "Peek" button and the application will tell you which kind of entity (property, type or object) is represented by the PID you entered in the text field
- **Resolve:** Enter a PID in the text field and tell the application which kind it represents via the radio buttons (again, property, type and object are chooseable). The application will resolve the PID record and display all assigned metadata in the table below. If the checkbox "Show names" is activated, the PIDs of the properties will be resolved to human readable names
- **Validate:** Enter the PID of an object in the PID text field and the pid of a type in the text field "Type". The application will check if the object's metadata validates against the given type: It checks if all mandatory properties are present

Some example PIDs:

How to get the PIT output

- Demonstrator at RZG and documentation:
  http://smw-rda.esc.rzg.mpg.de/PitApiGui/
  http://smw-rda.esc.rzg.mpg.de/apidocs/
- Prototype source code available via git:
  `git clone git://redmine.dkrz.de/rdapit.git`
- Final overview report available from the RDA websites:
  https://rd-alliance.org/groups/pid-information-types-wg.html

- More formal outcome package in the loop.
- Licenses: CC0 / simple BSD

A bit of reflection...

- Even with very simple information, each use case favors a different set of types
- There is no single set of types fitting all cases – we have to live with that in practice and look towards the Type Registries to help us
- Community processes must define types from practical adoption

How does the PID Information Types effort continue?

- The API is a prototype that has to see further refinement further practical adoption
- DKRZ follows through with future plans in the context of an international data infrastructure (ESGF) and EUDAT
  - This will also shed more light on essential types
- Interest was also stated by e.g. Deep Carbon Observatory and the Materials Genome Initiative

Take-home messages

- Work is not over – now comes the clash with practice
- Assigning PIDs is the first step. Typing is the second.
- Political consensus in a community/infrastructure is crucial – challenge too big for single institution
- Keep It Simple & Stupid – also in the future
- Local motivation – automate our workflows at DKRZ

- Continuing efforts in RDA regarding Collections

- Thank you for your attention.