Digital Object Architecture Primer

Digital Objects - from RDA Results towards Implementation

5 Apr 2019

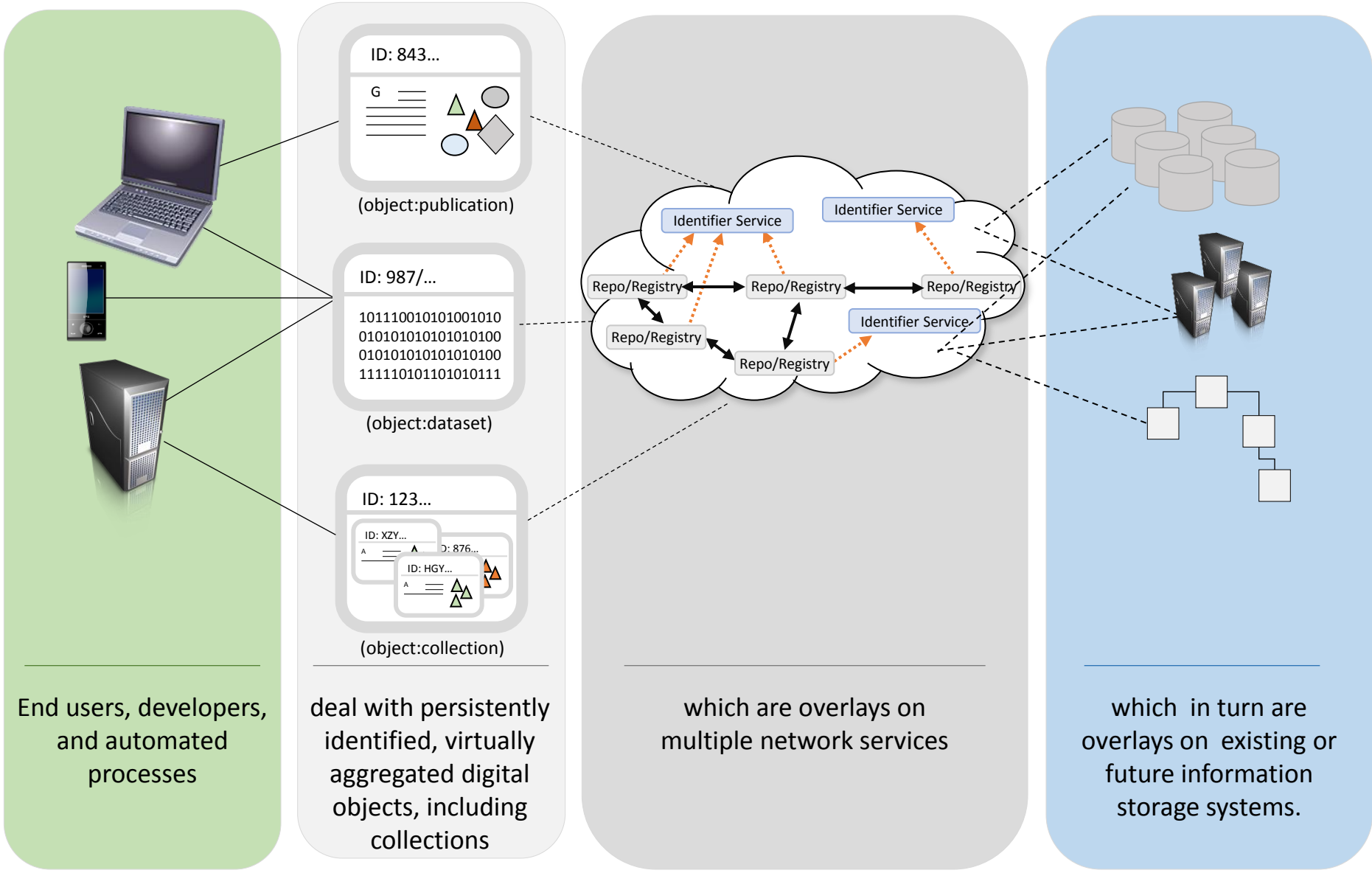
Larry Lannom

Corporation for National Research Initiatives

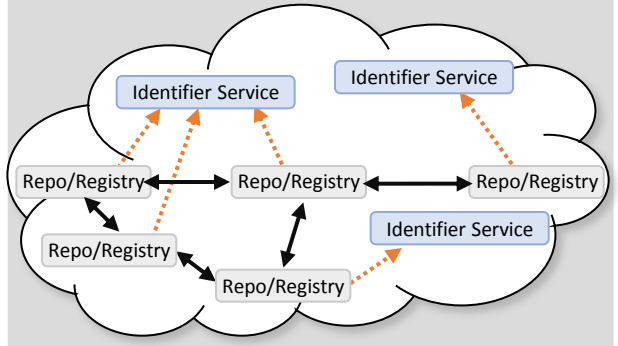
What Is the Problem?

- The world is awash in data - volume/velocity/variety
- Science has gone from articles to data – the 4th paradigm
- Commercial/Industrial data has gone from accounting to tracking all activities, e.g., real-time supply chains
- All of this data should yield better science and engineering and more commercial efficiency, but data is hard to understand, share, or reuse across the silos in which it starts
  - Numbers are not self-explanatory
  - Both human-level description and numbers come with a specific context
  - How can I find the data I need?
  - What can I do with the data? Can I trust the data?
  - Science has a reproducibility problem and scientists spend more time on data than science
- Need to turn the challenge into an opportunity, change the problem of too much hard to use/find/understand data to the advantage of lots of accessible and understandable data
- Common infrastructure at the data management level cannot solve all problems but is the necessary first step

Global Digital Object Cloud (GDOC)

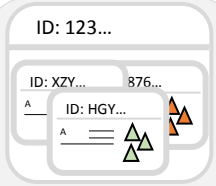


Global Digital Object Cloud (GDOC)



which are overlays on multiple network services

End users, developers, and automated processes

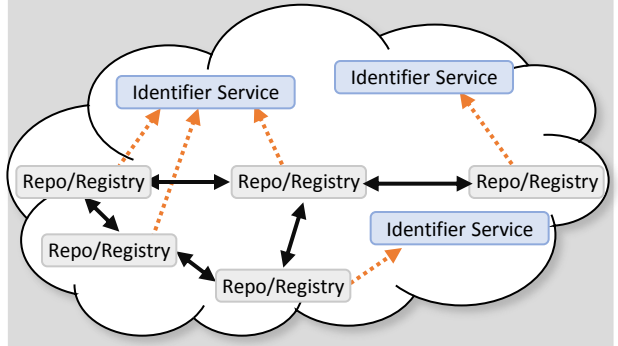


deal with persistently identified, virtually aggregated digital objects, including collections

which in turn are overlays on existing or future information storage systems.

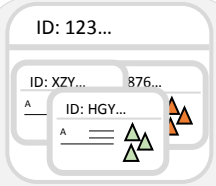
Global Digital Object Cloud (GDOC)

These services can be orchestrated to provide an object view of underlying storage, e.g., file systems, or basic data management systems, e.g., databases.



which are overlays on multiple network services

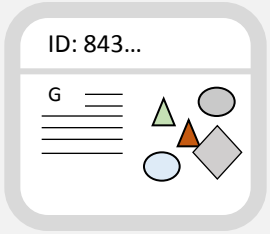
End users, developers, and automated processes



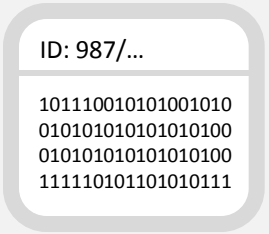
deal with persistently identified, virtually aggregated digital objects, including collections

which in turn are overlays on existing or future information storage systems.

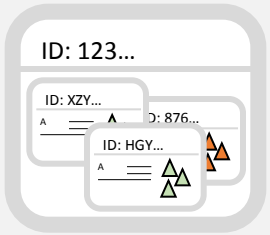
Global Digital Object Cloud (GDOC)



(object:publication)



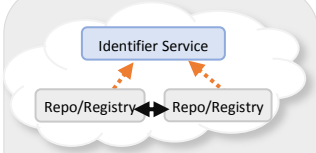
(object:dataset)



(object:collection)

deal with persistently identified, virtually aggregated digital objects, including collections

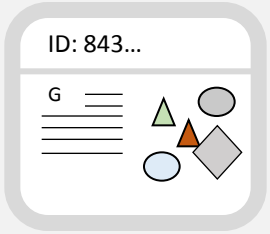
End users, developers, and automated processes



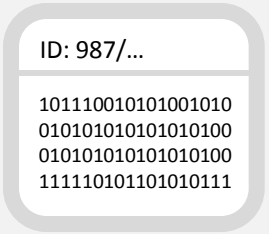
which are overlays on multiple network services

which in turn are overlays on existing or future information storage systems.

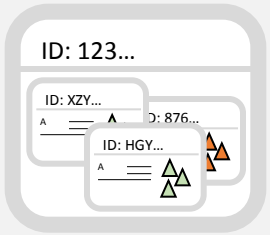
Global Digital Object Cloud (GDOC)



(object:publication)



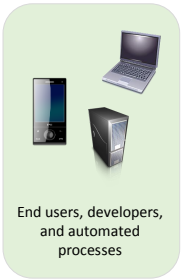
(object:dataset)



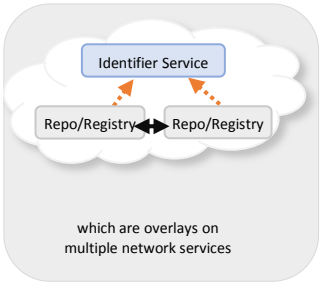
(object:collection)

deal with persistently identified, virtually aggregated digital objects, including collections

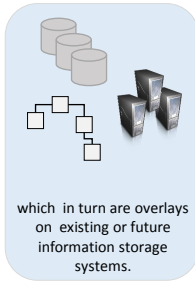
The resulting set of identified and well-structured objects provide a common, and constant, view and 'remote control' management of data distributed in various locations and systems, which can change without changing the virtualized object.



End users, developers, and automated processes



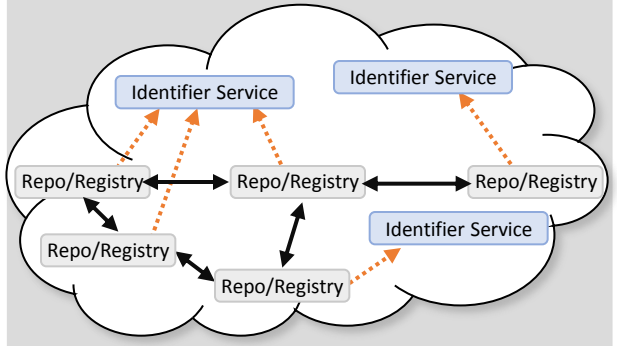
which are overlays on multiple network services



which in turn are overlays on existing or future information storage systems.

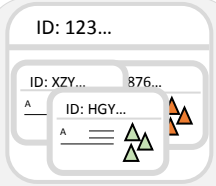
Global Digital Object Cloud (GDOC)

All of these services exist today in one form or another, but some are not yet widely used and few are tightly coordinated and orchestrated in the way that is needed.



which are overlays on multiple network services

End users, developers, and automated processes



deal with persistently identified, virtually aggregated digital objects, including collections

which in turn are overlays on existing or future information storage systems.

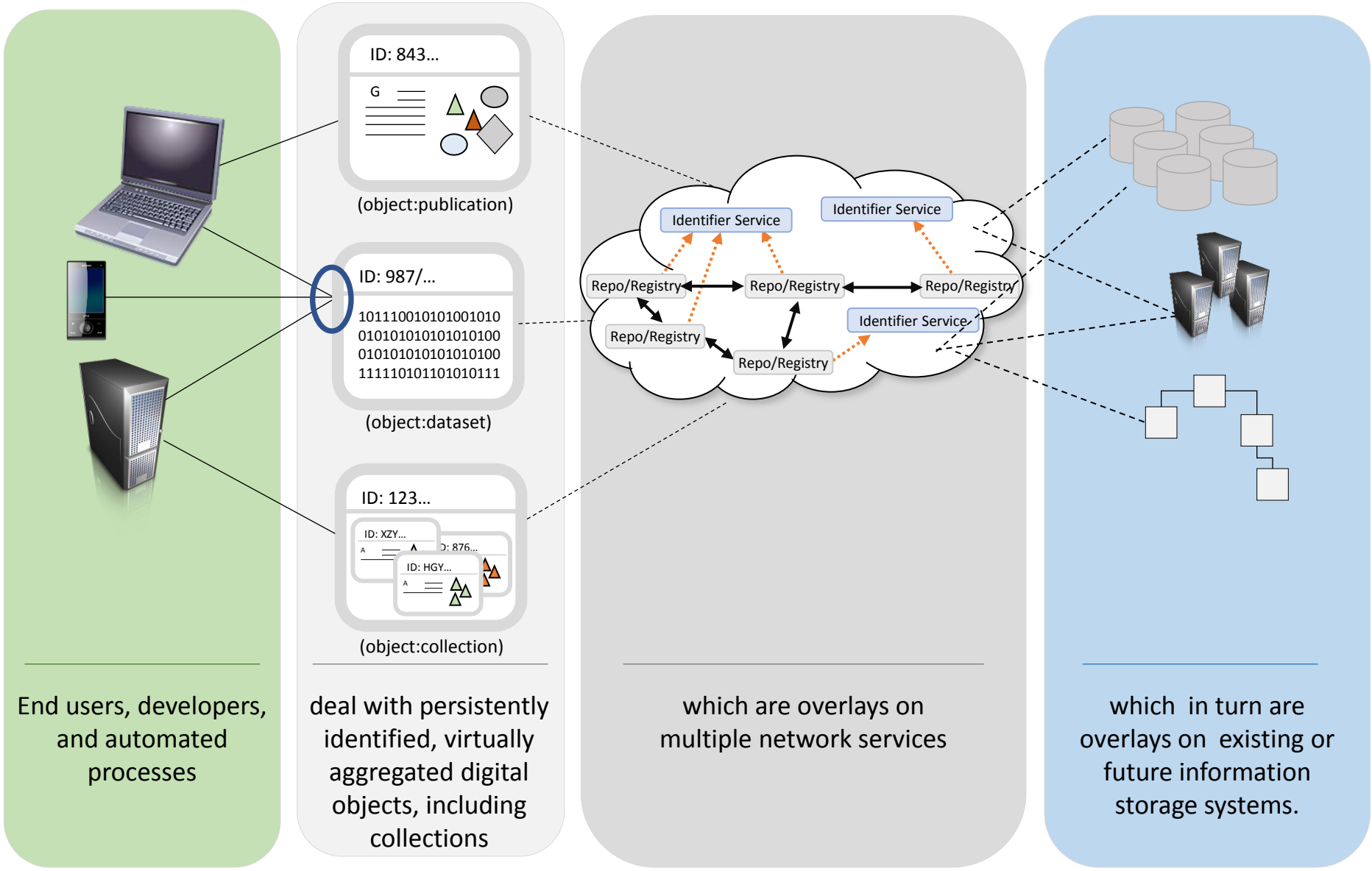
What Exactly are we Proposing to Do?

- Collectively build a distributed environment based on the digital object model
  - Everything in the environment is a digital object
  - For basic information management tasks every object can be treated the same, regardless of information content
    - Every object has a globally unique and actionable identifier
    - Every object is typed
    - Every object has tightly associated metadata
    - Every object has a queryable set of operations that can be performed on it
- Start with the minimal set of components and services that enable the DO model
  - Identifiers + Resolution System
  - Types + Type Registries
  - DO Repositories, including repositories of metadata, aka, registries
  - Mapping/brokering software & services to map existing data storage and management systems to DOs
  - Digital Object Interface Protocol, implemented by DO Repositories
- Open the environment to as many use cases as possible to hone the core infrastructural pieces

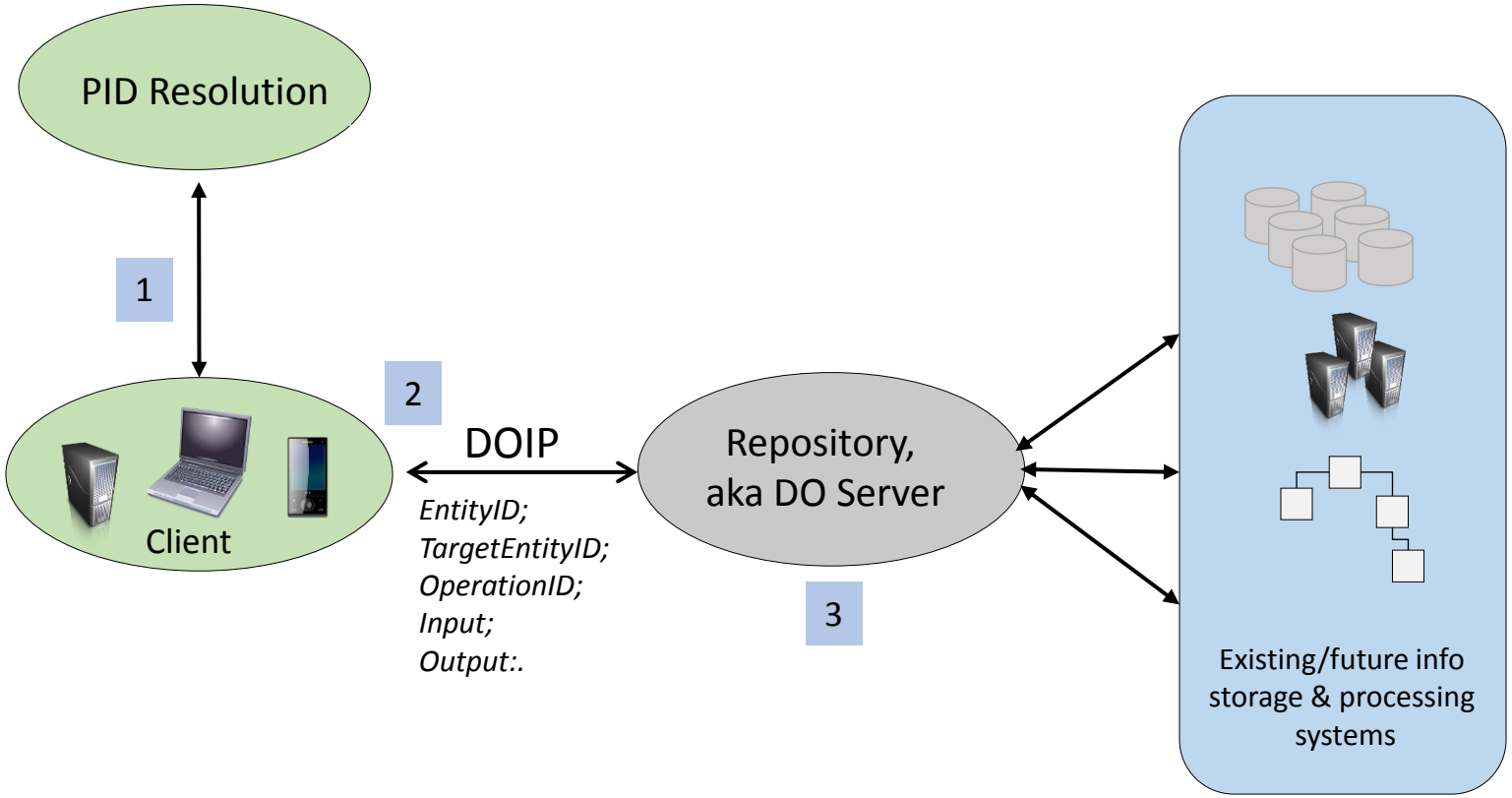
Why is this a Good Idea?

- The Digital Object Model Simplifies the Solution Space
  - Treat every information object the same until you have to differentiate among them to accomplish your purpose
  - Push the current cacophony of information management and storage systems down a level of abstraction
  - Objects are self-describing in that they carry their type information independent of their current system location
- The environment will be based on open standards and proven technology
- This approach has already gathered significant support, including groups within RDA and the Go FAIR initiative

Global Digital Object Cloud (GDOC)



Generic DO Access Flow

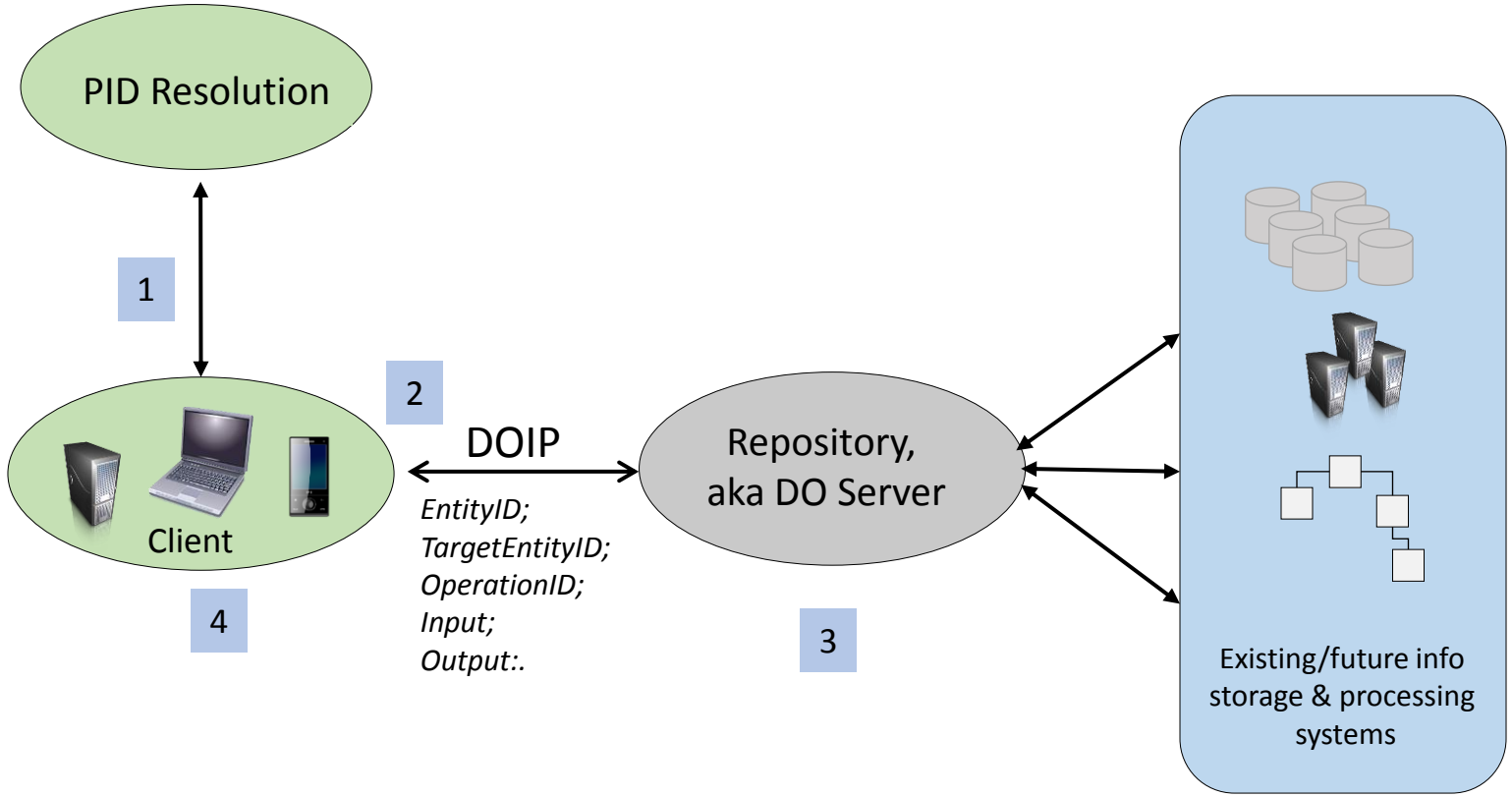


1 Client resolves PID to get current state data, minimally incl. network location.

2 Client sends DOIP request to relevant repository.

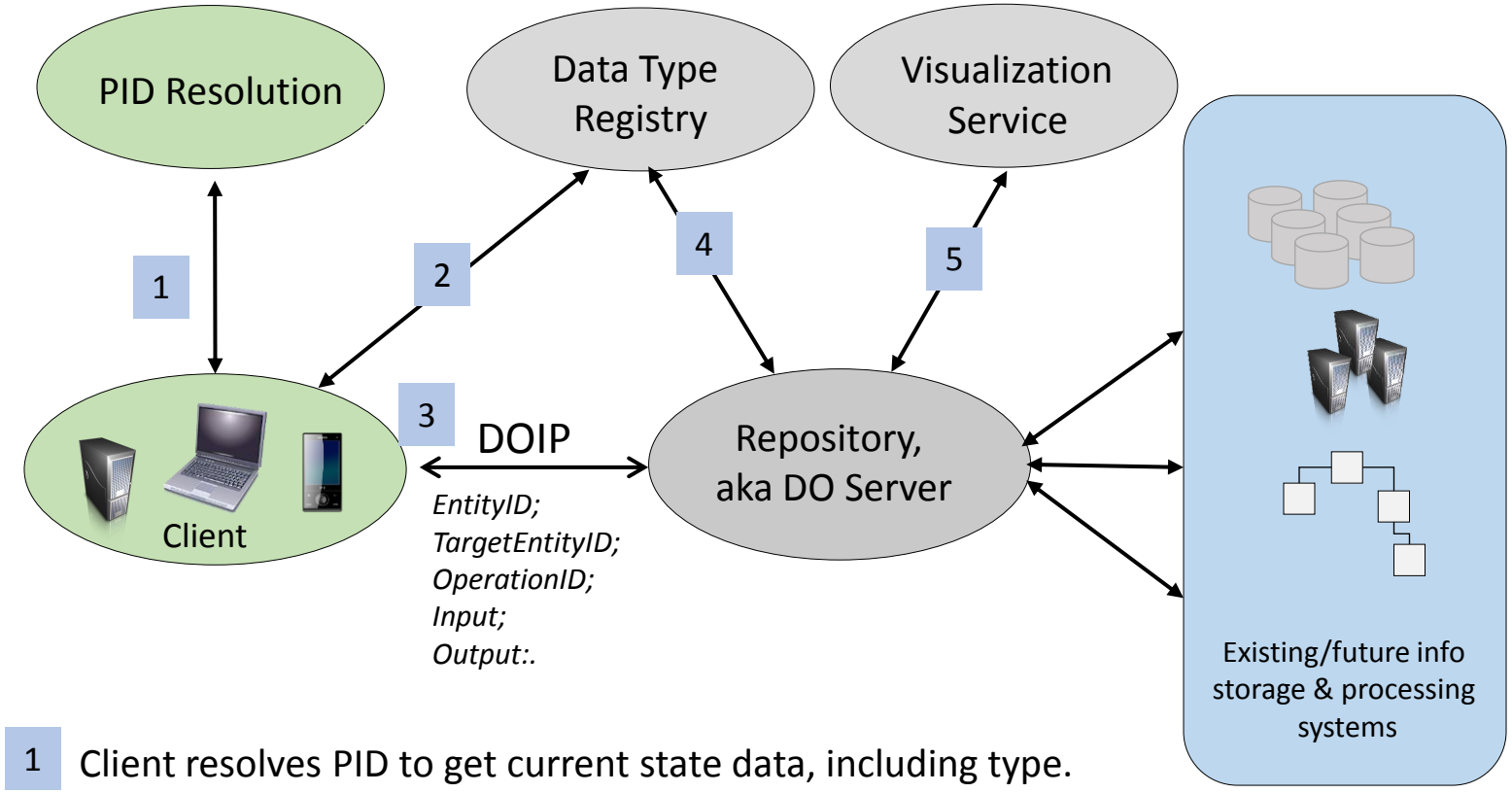
3 Repository finds or computes data to respond to client request.

Verifying a DO Using Checksum



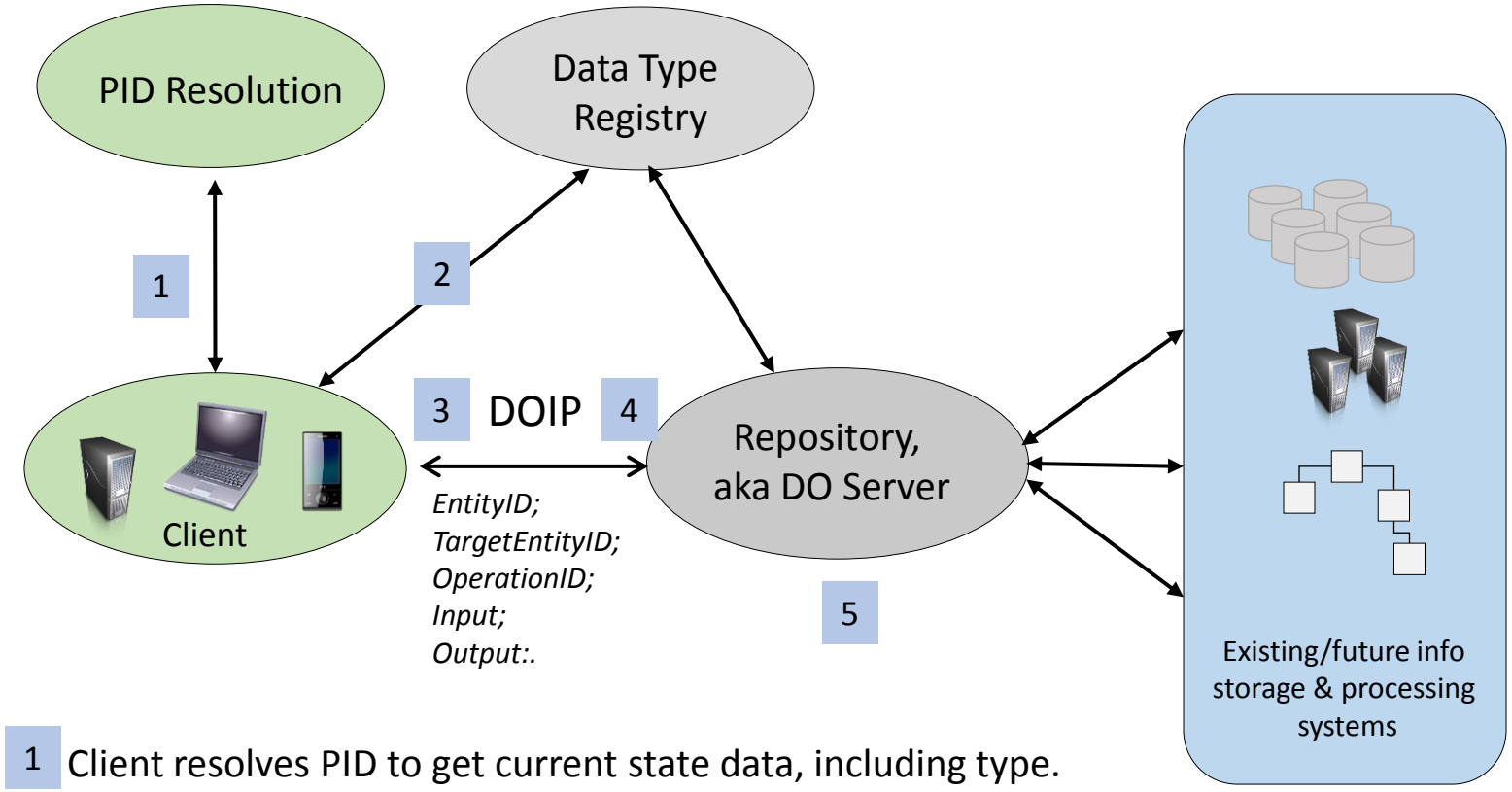
1. Client resolves PID to get current state data, including checksum.
2. Client sends DOIP request to relevant repository, requesting the object itself as the return.
3. Repository finds or computes data to respond to client request.
4. Client computes the checksum of the returned object and compares it to the value in the PID.

Visualization Request



1. Client resolves PID to get current state data, including type.
2. Client resolves type & evaluates potential ops, finds visualization.
3. Client sends DOIP request for visualization to relevant repository.
4. Repository resolves type to find network visualization services.
5. Repository requests visualization (sends data or gets routine) and responds to client with image or location (many configurations possible).

Computation of Survey Crosstabs



1. Client resolves PID to get current state data, including type.
2. Client resolves type, finds survey type, including a crosstabs operation.
3. Client sends DOIP request for survey template (assuming a human client)
4. Client decides on crosstabs for a given set of questions and, using type info on the request language, sends request to repository.
5. Repository computes crosstabs according to the request and returns values to client.