RAIDAR Architecture

Eric Scace
scace@mit.edu

1 Design principles

See *RAIDAR: an Open Network of Rights & Asset Information in Decentralized, Authoritative Repositories* for a brief discussion of the guiding design principles.

1.1 DatumPack [1]

A DatumPack [DP] refers to a package containing:

- A datum;
- Provenance: a chain of records regarding:
  - Sources: of the original value, and of each change thereafter, referencing identity (including proofs) and authorization (including proofs) for the person, her system, and her software.
  - Indices of expertise and reliability for each source. [2]
  - Custody: proofs of correctness and degree of confidence in proofs through the chain of custody. [3]

RAIDAR operates on DPS representing four types of information:

- Asset, such as a recording or a composition.
- Asset metadata [4]
- Right
- Right metadata.

A clever-term-yet-to-be-defined refers to the collection of the above pertaining to a single asset. RAIDAR may operate on license and license metadata DPS, and potentially royalty-related DPS, in future versions.

1.2 Initial version

Version 1 will not implement all functions and interface aspects described below. Footnotes identify reduced capabilities, stubs, etc for version 1 as well as potential extensions for future releases.

2 Functional groupings

2.1 User

2.2 Core

3 External services

3.2.1 Storage Mediation

Core employs Storage Mediation to retain DPS.

[1] Is there a better term than this invented word?
[2] See *Raider, an Open Network of Rights & Asset Information...* for definitions of expertise and reliability.
[3] Ledger services provide these proofs of correctness for a DP.
[4] Enumerate asset metadata, right, and right metadata further, based on white board discussion with George and white paper.

3.2.2 Ledger Mediation

3.2.3 Identity Authentication & Access Management

3.2.4 Other potential external services

4 Interfaces

Aspects common to all interfaces:

- All interfaces shall be RESTful.
- Every message on each interface shall contain version control features designed to maximize forward and backward compatibility. A single method meeting this requirement shall be applied to all interfaces. [5]

4.1 User-Core

4.2 Core-Core

4.3 Core-Storage Mediation

4.4 Core-Ledger Mediation

4.5 Core-Identity Authentication & Access Management

[5] Link to description of chosen method.