HSDS Use Case Summaries: Primary Types of Use

This document summarizes Open Referral's findings about the range of users' needs associated with the use of resource directory data via APIs and other methods of exchange.

We've organized a range of use cases into a set of five primary 'types of use' that describe a broad, high-level set of users' needs and desired actions and outcomes. In light of these summaries – which we expect to evolve over time – we can assess proposals for inclusion and revision of features in the Human Service Data Specification and API protocols.

Thanks to Mike Thacker of Porism for drafting the initial version of this document.

Key links:

- Open Referral's personas and user stories – which lays out the primary types of users served by our initiative. Different combinations of these user types are involved in each type of use described below.
- Original document: Por11305 - UK case studies for Open Referral
- Open Referral UK Discourse forum post
- HSDS upgrade – notes
- Github – schema specs, api specs
- Michigan 211's API use case

Table of Contents

1. Service lookup: enabling users to query an HSDS data source to find a service and act on the information

1.1 The use case

Various methods might be deployed to enable users to navigate through a directory to find services that are relevant to them, but at the core there are a common set of functions that should enable any resource navigation tool to use any compliant API feed.

In this context a service lookup function might be used in a public website, or it might involve tooling within an application that enables care coordination – such as a "closed loop referral system" or "social prescribing tool."

This entails enabling users to perform searches on service description and type of service, as well as organization name, address, as well as filtering for specific features like target populations, eligibility criteria, service area, [and other criteria that enable users to reference a user's specific needs in the process of searching for services for which they are eligible]. When such services are identified, social prescribers need information that lets the individual make use of the service (e.g. appropriate contact details, application process, documents required, etc).

In the emerging horizon of this use case, some social service providers might have encrypted endpoints through which they could securely receive (and send) personal information about clients as referred to them. These endpoints might be, for example, a REST server for a FHIR system, or a Direct messaging address – and might be associated with an organization or the specific services it providers.

1.2 Implications for HSDS and the API

API requirements:

- GET a list of services matching these filter criteria: core
    - text (in name, description and possibly other fields according to the local implementation, e.g. via elasticsearch)
    - service type (taxonomy term)

  Less important but useful criteria are: non-core
    - age
    - service eligibility area
    - other eligibility criteria
- GET full details of chosen service(s) such that a referral can be made and full details can be loaded into a local database for more sophisticated searching core
- GET organizations that provide services of a type non-core
- GET services at location non-core

HSDS requirements:

- Attending_type, i.e. how to attend a service (online, phone, in person, ...)
- Attending_access i.e. by referral, booking, drop-in

2. Alignment of multiple datasets: to enable resource data federation, matching/synthesizing/syncing multiple sources of HSDS

2.1 The use case

Allow aggregation (combining) of services data from multiple publishers for a geographical area where directories are compiled by many different public, private and voluntary organisations.

This means supporting federated data whereby multiple organisations publish and maintain their own services data but this data can be harvested and brought together in one place for a particular purpose. For example an online service directory for an English county might take its data from the county local authority, multiple (smaller) district authorities, health organisations and a community/voluntary directory. Once brought together in one place, the combined data must not lose meaning. This implies:

- being in the same data structure or (failing that) being in structures that can be mapped to a common structure
- using the same taxonomies for terms or (failing that) using taxonomies that can be mapped to a common taxonomy
- avoiding duplication of records by using identifiers that are unique across all harvested sources and have fields (e.g. charity number) that help find duplicates from different sources

2.2 Implications for HSDS and the API

API requirements:

- GET a paginated list of services core
- GET services for with a last_assured date/time later than a given date/time non-core
- GET services for which comply with the stated application profile non-core
- GET full details of each service core

UK users have NOT identified a need for a full database as a dump in the Open Referral structure.

UK users do NOT think POST and PUT operations need be standardised. These can be implemented by local implementations that GET data in standard ways.

HSDS requirements:

- Use of UUIDs for all identifiers to enable easy merging of records from different endpoints. This applies to all identifiers apart from external ones (such as taxonomy terms with URIs)
- External identifiers (e.g. organisation zip/postcode, company/charity number) to help identify duplicates

The application profile for one set of feeds from federated publishers must specify:

- Fields required beyond the Open Referral core mandated fields

- Taxonomies and enumerations to be used

3. Analyzing and improving resource data quality.

3.1 The use case

Data quality is an essential factor in meeting users needs, and one of the benefits of standardization and open source tooling is to expand the range of methods to assess, improve, and validate the quality of resource datasets. In this type of use, users (such as data administrators) want to be able to analyze the contents and format of resource datasets, understand its contents and compliance with HSDS, and perform functions that improve data quality with regards to both standards compliance and [cleanliness / granularity / usability of contents].

Open Referral's ecosystem already includes a number of generic tools for use as utilities in multiple circumstances, such as the dashboard, the API query tool, the exporter and the validator developed in the UK, and transformer tools developed in the US. Future such tools can be developed once and shared across all consumers. These tools can become part of a resource data supply chain pipeline, and even components of compliance processes that require validation, etc.

This means having a structure that can be read by such generic tools, so for example, the /services GET method having the same response format wherever the data comes from.

3.2 Implications for HSDS and the API

Common tools will check compliance with standards defined to meet other use cases and will not themselves add extra API or HSDS requirements.

The API needs to be able to read full details of every service - as for an aggregator.

We note that:

- Compliance with the core HSDS specification and with the govern application profile may be recorded as two levels of compliance
- It would be useful to generate compliance and richness scores so hard rules can be imposed during procurement of a directory
- Tools giving the compliance and richness of individual service records would be useful

4. Editing individual records

4.1 The use case

A data administrator needs to submit edits to a record for updating, verification, etc.

Self-updating a register: Allow an organizational representative (i.e. a staffer) to attest to "official" information about their organization's services. They want to establish one authoritative record as the ‘single source of truth.’ As part of this process, they may also want to review other sources' existing information about their services, and propose corrections for review by the respective sources. They want the provenance of their officially attested information to be clearly indicated, with a history of versions that can be consulted. They want to be able to reference this official record via a URI that presents the data as both a human-readable web page and a machine-readable datapackage.

4.2 Implications for HSDS and the API

5. Bulk data transfer

5.1 The use case

Allow transfer of an entire database of services and related information such that it can be interpreted by the recipient.

In one situation in the UK, data is gathered in one database for two local authority areas. It is transferred for one of those areas via standard API requests to a separate database where it is imported into a proprietary tool and merged with other data.

This might involve denormalizing and *re*normalizing the data to be replicated in a new database.

This transfer should include metadata about taxonomies used including their version, custom categories being shared, schema version, and so on. In these bulk transfers, developers might also want fields left out (implying no change to that field, don’t update it) or sent blank (erasing the field contents). This makes it possible to send a change data set, for systems that are capable.

Developers want simple interfaces through which they can verify record status across the set to perform reconciliations. This allows them to manage deletions in systems that have no good way to communicate deletions OR removals from a target set (understanding that a record may not be deleted, but may not meet sharing criteria any longer).

As a developer, the criteria I would like to see in a bulk API are:

1. Deletes need to be included.

Denormalized records so we get all data in one call would be very nice.

5.2 Implications for HSDS and the API

As for aggregation of federated databases.