Enrollment (Registration)

In most cases, the registration (aka enrollment) process is initiated when a **System of Record (SOR)** learns about a new person and passes that information on to the Registry. (Some organizations flip this process, where a person cannot be known to the SOR until the Registry tells the SOR about the person.) There are multiple types of SORs, which may vary according to the organization, and may include:

- ERPs, such as Human Resource Management Systems (HRMS), Student Information Systems (SIS), and Alumni Development Systems
- Guest/Visitor Management Systems
- ID Badging Systems (May be authoritative for photos or badge numbers)
- Other organizations' Identity Management Systems (IdMS) (This is particularly true of Virtual Organizations)

In some cases, a Registry can also behave as an SOR repository for a given population.

Requirement

A general Registry solution must be able to support multiple types of SORs, which may not provide the same attributes.

Typically, an SOR is only authoritative for role-level data about a specific population (eg: students or employees). *[OSIdM4HEteam:Not trying to start a terminology battle here... role just means, eg, attributes associated with being a student.]* The Registry is authoritative for person-level data. The line between these two may vary according to a given organization's needs or policies, but typically person-level data includes attributes such as name (which may be selected from an SOR's name for an individual) or netid, while role-level data includes attributes such as title or physical address.

Requirement

A general Registry solution must be flexible enough to handle variations across organizational data models.

Data may be transferred from an SOR to the Registry via one of several mechanisms:

- Batch Extract: Typically a fixed-format, CSV, XML, etc file transferred via SFTP, etc and processed asynchronously at fixed intervals (eg: nightly). There is not typically a human being at the source end of the transaction at the time of the processing.
- Real Time API: Typically a REST, Web Service, etc protocol-based exchange triggered by an event at the SOR and processed synchronously by the Registry. There is typically an employee of the SOR at the SOR's interface at the time of the processing.
- Real Time UI: Typically a web interface providing direct access to the Registry, processing transactions synchronously. This is typically used where the Registry serves as an SOR repository or where duplicate data entry is required because an SOR can not provide sufficiently timely data to the Registry to meet the organization's business needs.

Requirement

A general Registry solution must provide multiple interfaces capable of receiving data from SORs.

Potential API

A new standard for the exchange of data between SORs and Registries would be useful.

When a new record is added to an SOR, it provides information to the Registry to allow the Registry to determine if it already knows about the person being added. The person may previously have been known to the Registry from a different SOR (eg: a student becomes a part time employee) or from the same SOR (eg: a part time employee accepts a second part time position). The latter case is easier (if the same SOR ID, or unique key within the SOR, is provided to the Registry), the former case typically requires some sort of fuzzy matching based on available attributes (such as name, date of birth, email address, national identifier, etc). This process is known as **Reconciliation/Identity Matching**.

Reconciliation/Identity Matching may generate the following results, each of which must be handled appropriately by the SOR:

- Person Identified (may be new or existing)
- Potential Match (of one or more existing people)
- Insufficient Data Provided

Requirement

A general Registry solution must offer a flexible reconciliation mechanism, capable of adapting to the available data and rules at each organization.

Potential API

A new standard for calling out to a potentially external matching system (either a standalone product or as part of a transition off a legacy system) would be useful.

Searching vs Matching

An ERP-centric view of identity may incorporate the notion of "searching before adding", born in part from the assumption that the ERP is authoritative for all person data. *(This wasn't necessarily brought up in any discussion to date, but caused sufficient damage at a major University to be worthy of calling out.)* This causes a number of problems:

- This assumes a human is at one end of the transaction, which only covers a subset of possible data entry paths.
- Data visibility/permissions may become a problem. For example, SIS records protected by FERPA may not be available to HRMS administrators (according to local interpretation of FERPA). In such a scenario, the HRMS administrator will receive incomplete/inaccurate results as part of the search.
- The search algorithm and reconciliation/matching algorithm must be kept in sync.
- Data can change between add and reconcile/match operations, causing unexpected results.
- The UX is very confusing at best, and difficult to explain to administrators.
- From the IdMS perspective, the workflow is not consistent across SORs.
- The ERP (SOR) is only authoritative for role level data, not person level data. It is up to the IdMS to determine if a person already exists.

In short, the SOR should worry about data managed by the SOR, and the IdMS should worry about data managed by the IdMS.

Additional References

- Various OpenRegistry documents, including Jeremy's Youtube video on OpenRegistry, Data Flow, and Add SoR Person Use Case
- COmanage Identity Intake