Identity Connector Framework

Open Source IAM (Un)conference
Vienna, 18.02.2020
Jonathan Gietz, David Hübner, DAASI International
Short Intro to ICF

- Provision and Sync from IAM-Systems into target systems (bi-directional)
- Connectors deal with target system and can be developed independently
- Once a Connector is developed (at best) every instance of the target system can be used.
- A Connector can be used in different API implementations
More Connector Properties (SPI)

• Multiple Interfaces
  - Connector
  - CreateOp
  - DeleteOp
  - SearchOp
  - TestOp
  - SchemaOp
  - ...

• Each Connector can have different capabilities
• Stateless by design
• Configuration is provided from outside via API
• Exceptions to throw errors to API
  - ConnectorException
  - AlreadyExistsException
  - ...

Example Connector Code Snipet

```java
@ConnectorClass(displayNameKey = "ExampleConnector.connector.display",
    configurationClass = ExampleConnectorConfiguration.class)
public class ExampleConnector implements Connector, SchemaOp, CreateOp, TestOp {

    @Override
    public Configuration getConfiguration() {
        // TODO
        return null;
    }

    @Override
    public void init(Configuration cfg) {
        // TODO
    }

    @Override
    public void dispose() {
        // TODO
    }

    @Override
    public Schema schema() {
        // TODO
        return null;
    }

    @Override
    public void test() {
        // TODO
    }

    @Override
    public Uid create(ObjectClass objectClass, Set<Attribute> createAttributes,
                      OperationOptions options) {
        // TODO Auto-generated method stub
        return null;
    }

}
```
Who uses ICF

- DAASI International in didmos2 (using ConnId)
- Evolveum in midPoint (using ConnId)
- Oracle in Oracle Identity Manager (Using Sun ICF?)
- Forgerock (Developing OpenICF)
- Tirasa (Developing ConnId)
ICF in didmos2

- **didmos2 Backend (Python)**
  - Writes
  - Reads periodically

- **didmos2 Response Connector**
  - Writes Data back to Backend

- **Worker Response**

- **Queue Responses**

- **RabbitMQ**
  - Writes in Queues
  - Queue 1
  - Queue 2

- **Connector A**

- **Worker (Spring Boot) (API Implementation)**
  - Worker A
  - Worker A
  - Worker B

- **Connector B**
Pros & Cons (DAASI Point of View)

Pros:

- Open Source
- A lot of different vendors use it
- Thus a kind of de facto standard in Open Source Ecosystem
- Many IdM sync problems are solved in a standardized way
- High flexibility
- Lots of connectors already developed

Cons:

- Mapping between connector schema and target can be difficult when it comes to complex data structures
- Sometimes not enough guidelines (especially in terms of API).
- More a tool set than a framework