Didmos
Technical Introduction

Version 1

DAASI International

Same Same but Different

If IdM projects always address the same problems, why does identity management always present a new challenge?

This is precisely the question which led to the development of didmos:

- A standardized system that is flexible enough to fulfil all customer requirements
- Proven for at least 9 customers

DAASI International

didmos

DAASI International

Identity

Management with

Open

Source

The Idea Behind didmos

- There may be common similar requirements for identity management, yet they are rarely exactly the same
- It is very difficult to develop a software which meets all requirements. Hence, didmos consists of several modules for different tasks.
- Each module further consists of sub-modules, which are assigned specific tasks, and which can be configured
- didmos also allows for plugin interfaces in many places to allow for customer specific logic
- Didmos relies on well-established open source products such as OpenLDAP, RabbitMQ, SATOSA, and many more
- Available as docker images
- Download at https://gitlab.daasi.de/didmos2

DAASI International

didmos Processes

Data Exchange (XML and/or JSON-based)

- Synchronize with didmos ETL Flow (Extract Transform Load)
- Provision with didmos Provisioner
- AD-Password synchronization with didmos Pwd Synchroniser

Management Processes

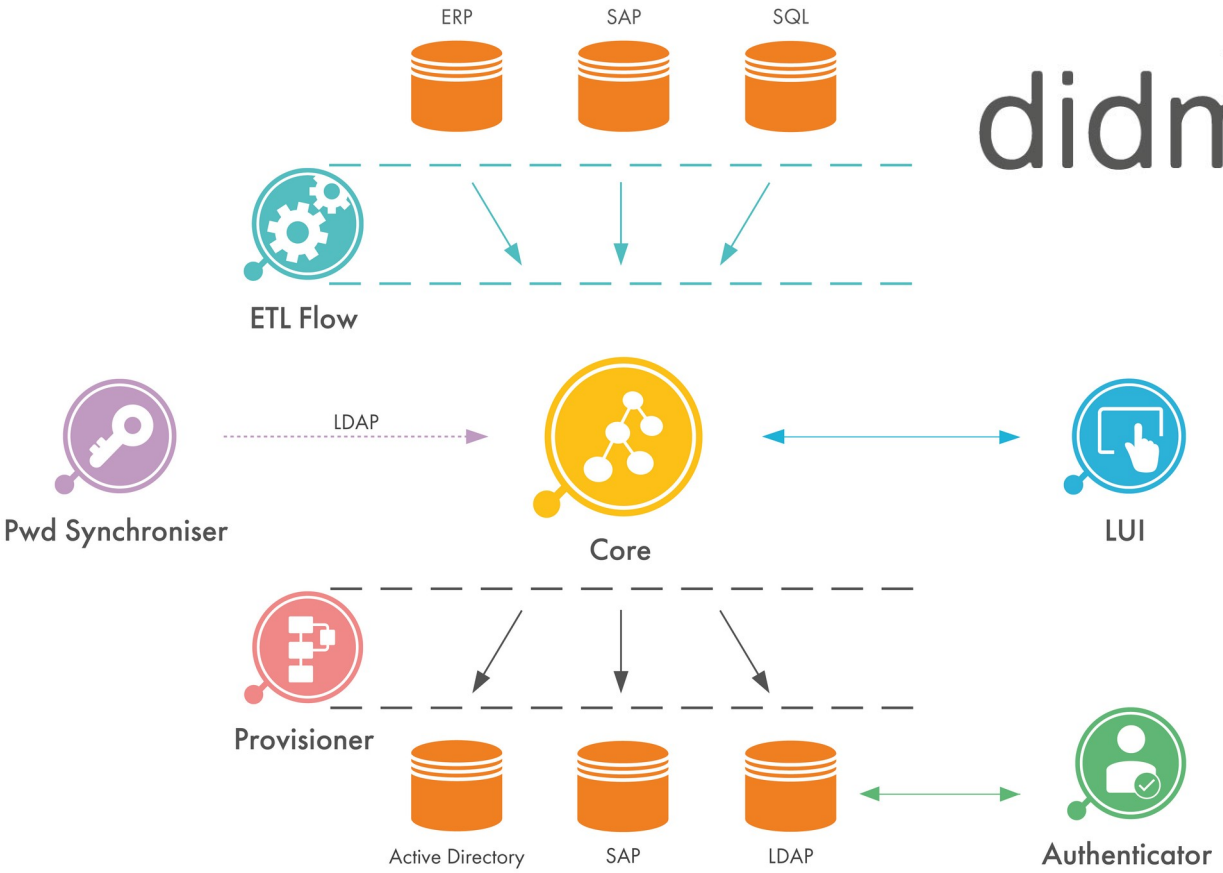
- Admin and self-service interfaces with LUI (LDAP User Interface)
- Propagation and other background processes

Access Management

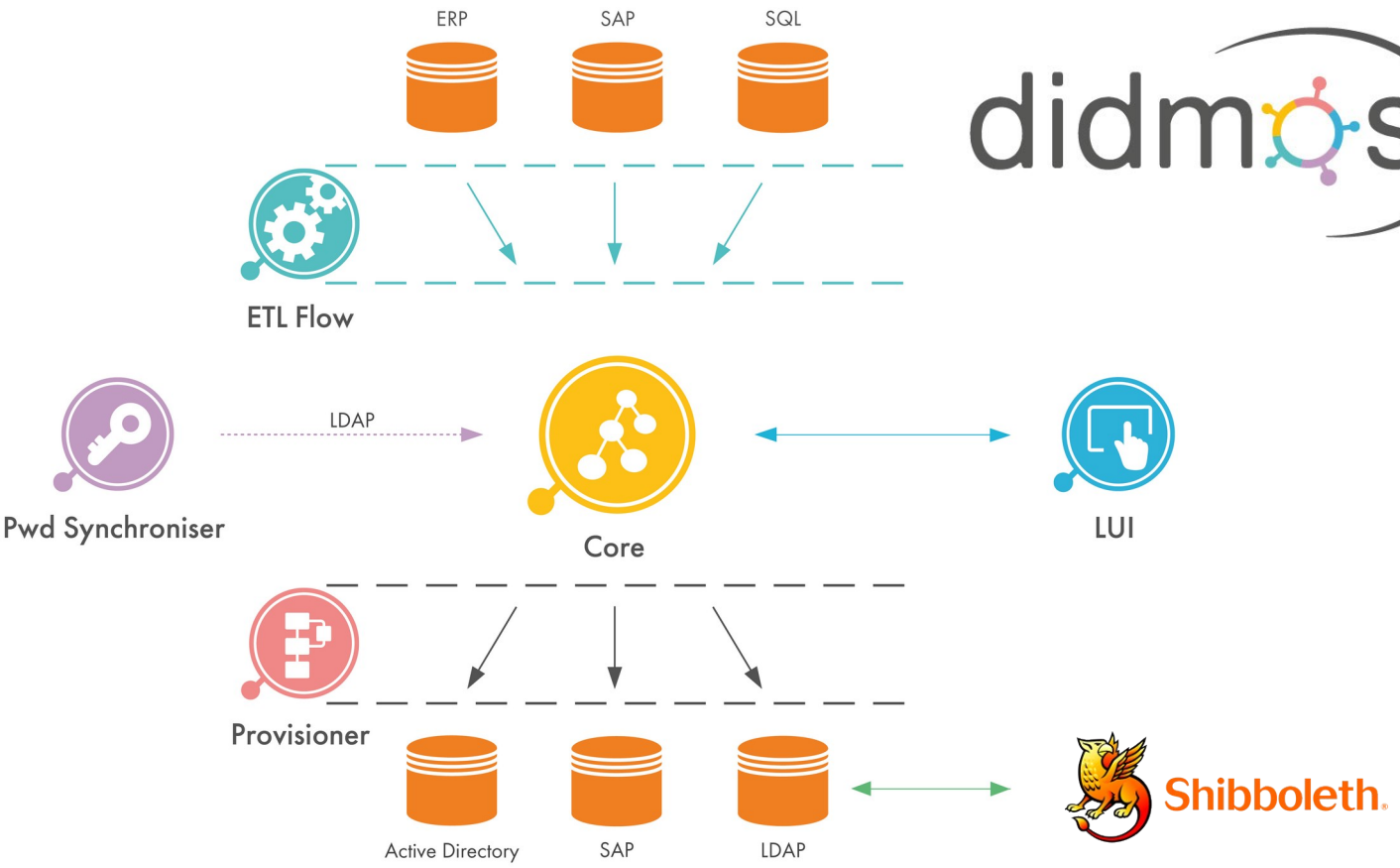
- Decision Point in didmos Core
- SSO and federation protocols with didmos Authenticator

DAASI International

didmos - Overview



didmos - Overview



didmos - Module

Core

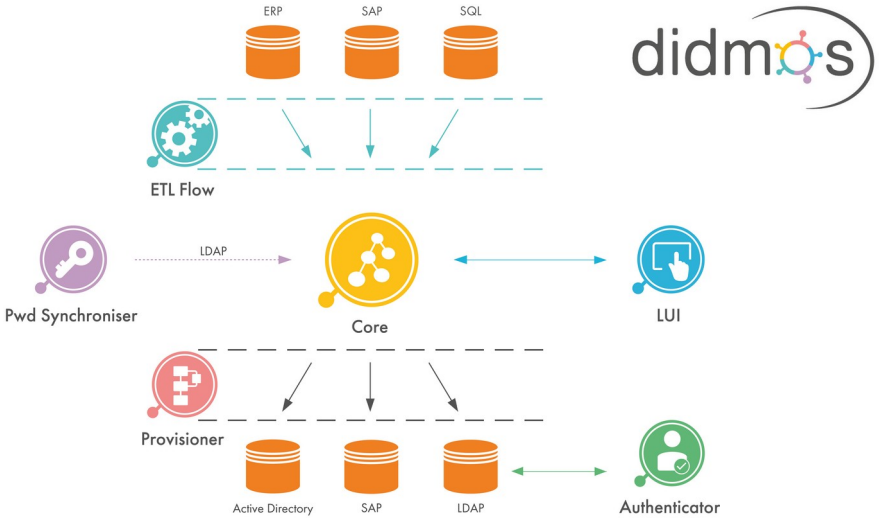
Heart of the didmos suite with administration functions (SCIM), processes and data store (LDAP)

LUI

Self-Service and Admin-Interface

Authenticator

Authentication component

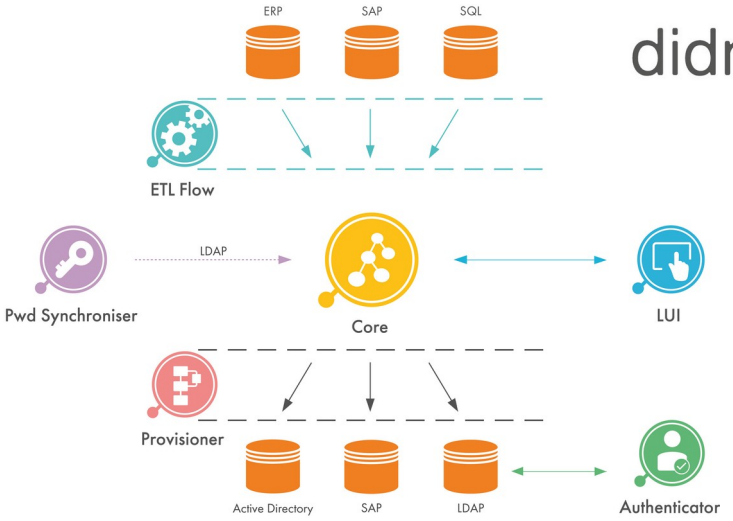


didmos - Modules

ETL Flow
Data transfer into the IdM

Pwd Synchroniser
Password synchronisation

Provisioner
Data transfer into target systems



didmos Core is the IdM Heart of didmos

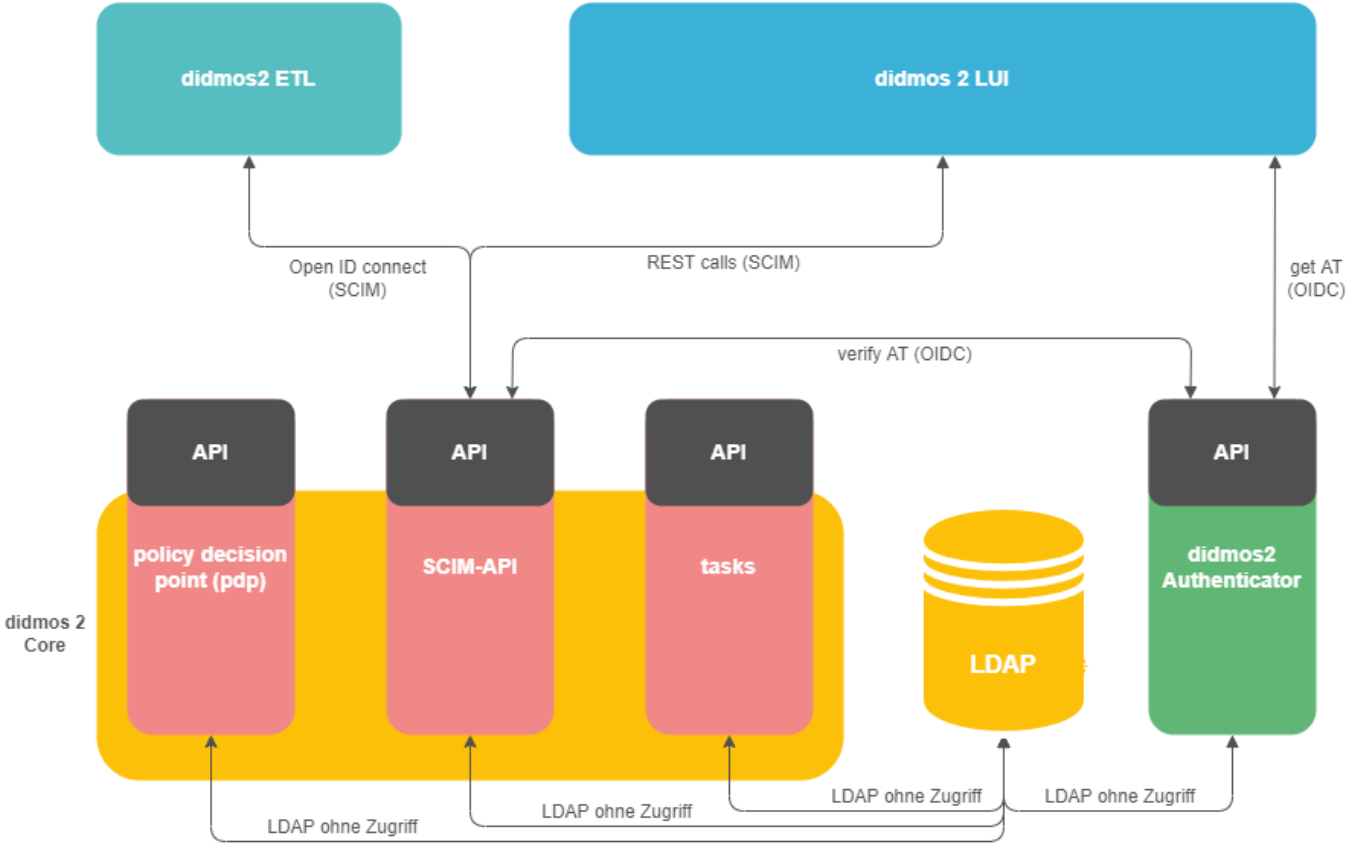
- Core consists of two components:
  - OpenLDAP server as metadirectory
  - Backend/API server for data access and background processes (based on Django/Python)
- Support for modern standards such as
  - SCIM for data access
  - OIDC/OAuth2 for authentication
- The API server consists of several modules (called “apps” in Django), which respectively provide a modern REST API via which data can be shared as JSON

Modules in Core

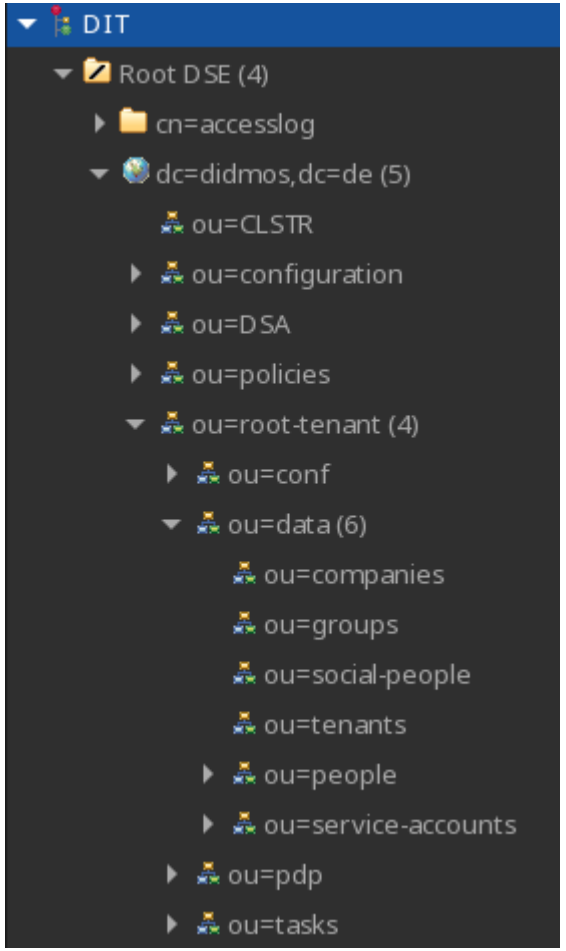
- Module for the administration of identities, groups and roles
  - Is addressed by the clients ETL Flow, and LUI
- Module to maintain workflows
  - Request and authorisation workflows
  - Time-controlled internal tasks
- Module to administrate permissions
  - Central policy decision point
  - Implementation of the RBAC standard
- Module for MFA token management
  - Currently connects to PrivacyIDEA
- Customer specific modules can be added
  - Any specialised function can be integrated
  - Customised code is maintained separately from the Core code

DAASI International

didmos Core data flow

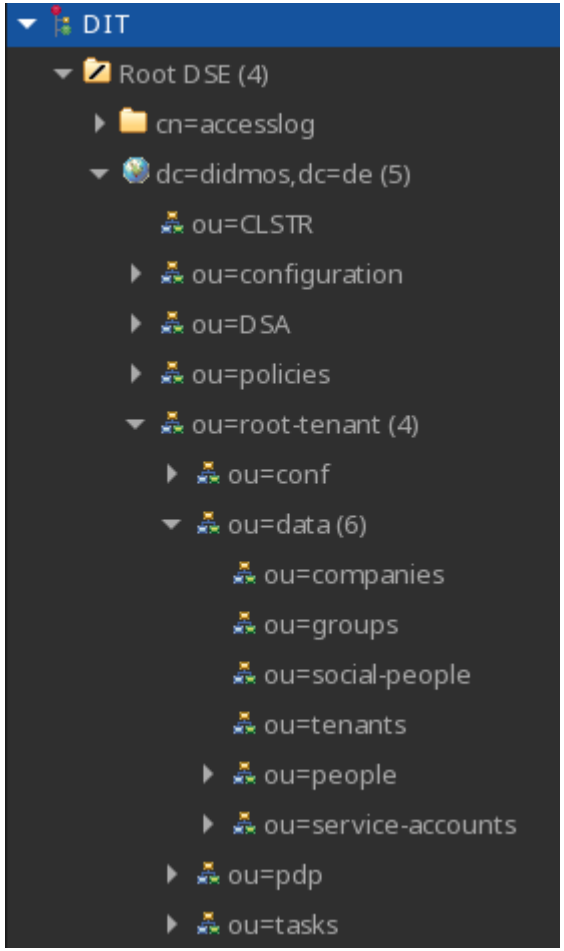


didmos Core LDAP data model



- **Multi tenancy**
  - Root-tenant can work stand-alone for simple (i.e. single tenant) use cases
- **Internal and external accounts**
  - ou=people for didmos-managed users
  - ou=social-people for “shadow accounts” which are created by didmos Auth on login
- **HA capable**
  - LDAP as multi master setup
  - Each backend instance uses a dedicated LDAP

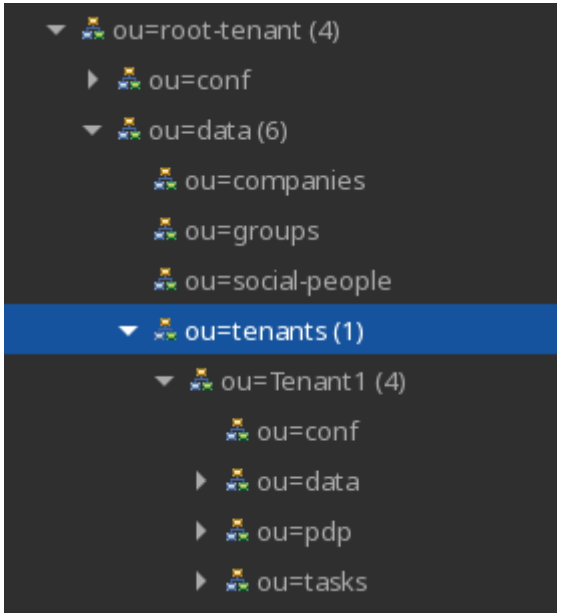
didmos Core LDAP data model cont'd



- **Groups**
  - Group owner functionality (to manage memberships via Self-Service)
  - Different access levels: open, requestable, closed
- **Tasks**
  - Requests
    - Types: Group-Requests, Role-Requests, Account-Requests
  - Internal periodic Tasks
    - E.g. Clean up, notification mails
    - Scriptable (Python) via LDAP objects

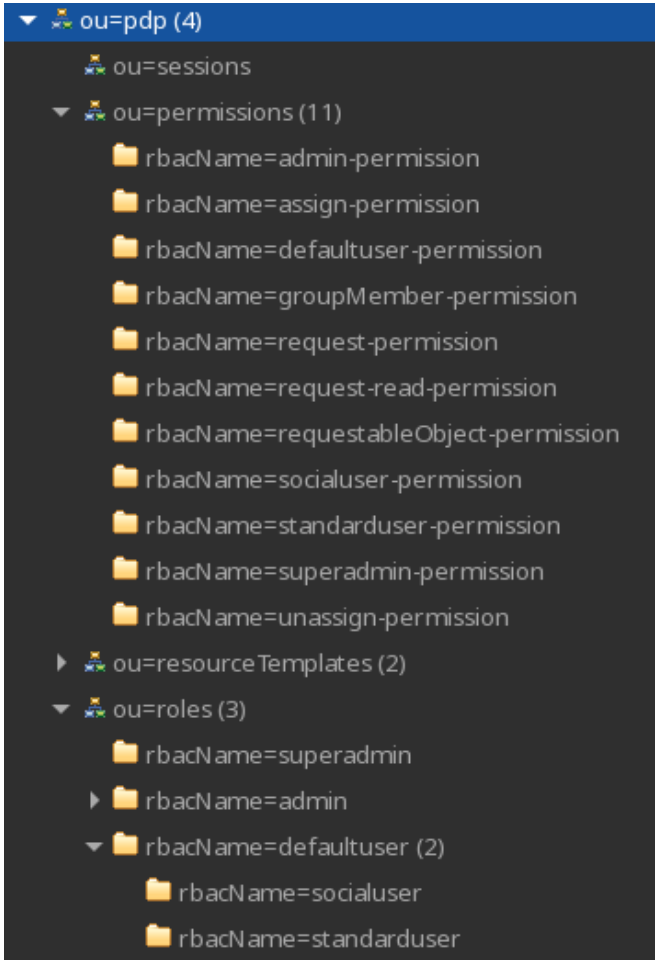
DAASI International

didmos Core LDAP data model



- **Optional sub-tenants**
  - Isolated data (users, groups, tasks etc.)
  - Tenant-specific permissions (PDP)

didmos Core LDAP data model



- **RBAC in ou=pdp**
  - *Permissions* that allow *roles* to perform *operations* (e.g. read, create, delete, ...) on *resources* (e.g. all users, all groups, self, …)
  - *Roles* that have users from the same tenant (*) as members
  - didmos ships with a set of predefined permissions and roles, but can be dynamically managed via PDP API module

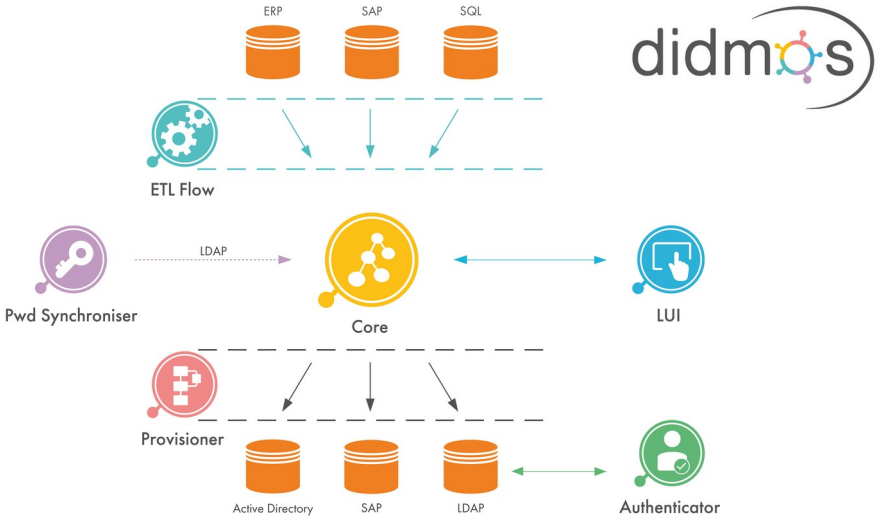
  * except global roles for helpdesk etc

didmos LUI

LUI
LDAP **U**ser **I**nterface

generic web portal framework for

- Administrative functions
- Self-service functions



didmos LUI

Frontend entirely based on JavaScript (Angular)

- Single-page application
- Responsive
- Mobile app (PWA) also available
- Authentication with OIDC
- Entire layout can be changed on build-time (CSS, Templates), some elements on run-time (logo, themes), due to Angular

Extensive usage of SCIM

- Only Core is responsible for the manipulation of LDAP objects
- SCIM calls protected by OAuth2/Bearer Tokens
- Menu items can be controlled via permission assignment from Core to LUI

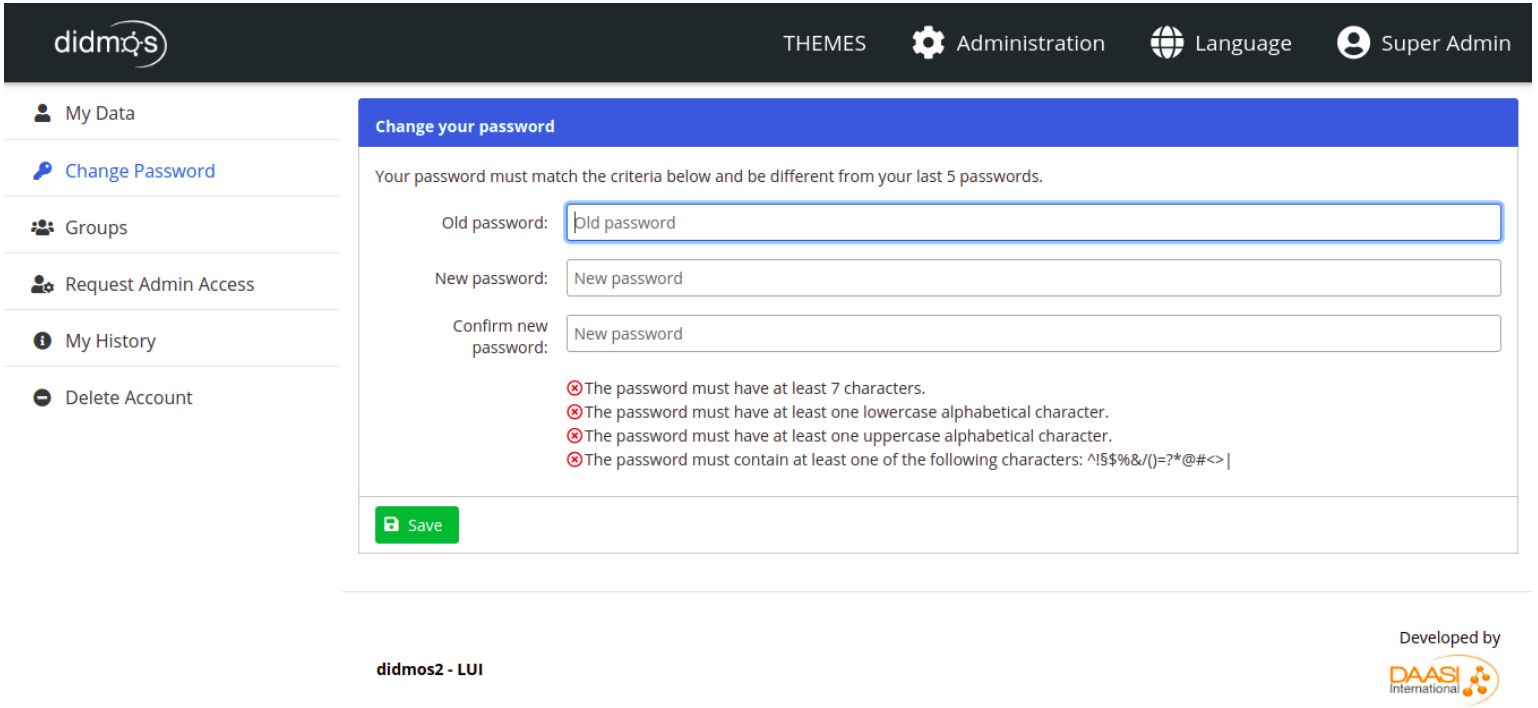
didmos LUI modules

- Each project uses a dedicated frontend, but common functionality is developed in a shared library with features such as:
  - Search and browse function for users, groups, other objects
  - Administration of group memberships
  - Assignment of role memberships
  - Password (re-)set function
    - e.g. an email with one-time valid link/token
  - Self-disclosure (GDPR)
  - Request processing (Account, Group, Role, Delete requests)
  - Bulk import (CSV file)

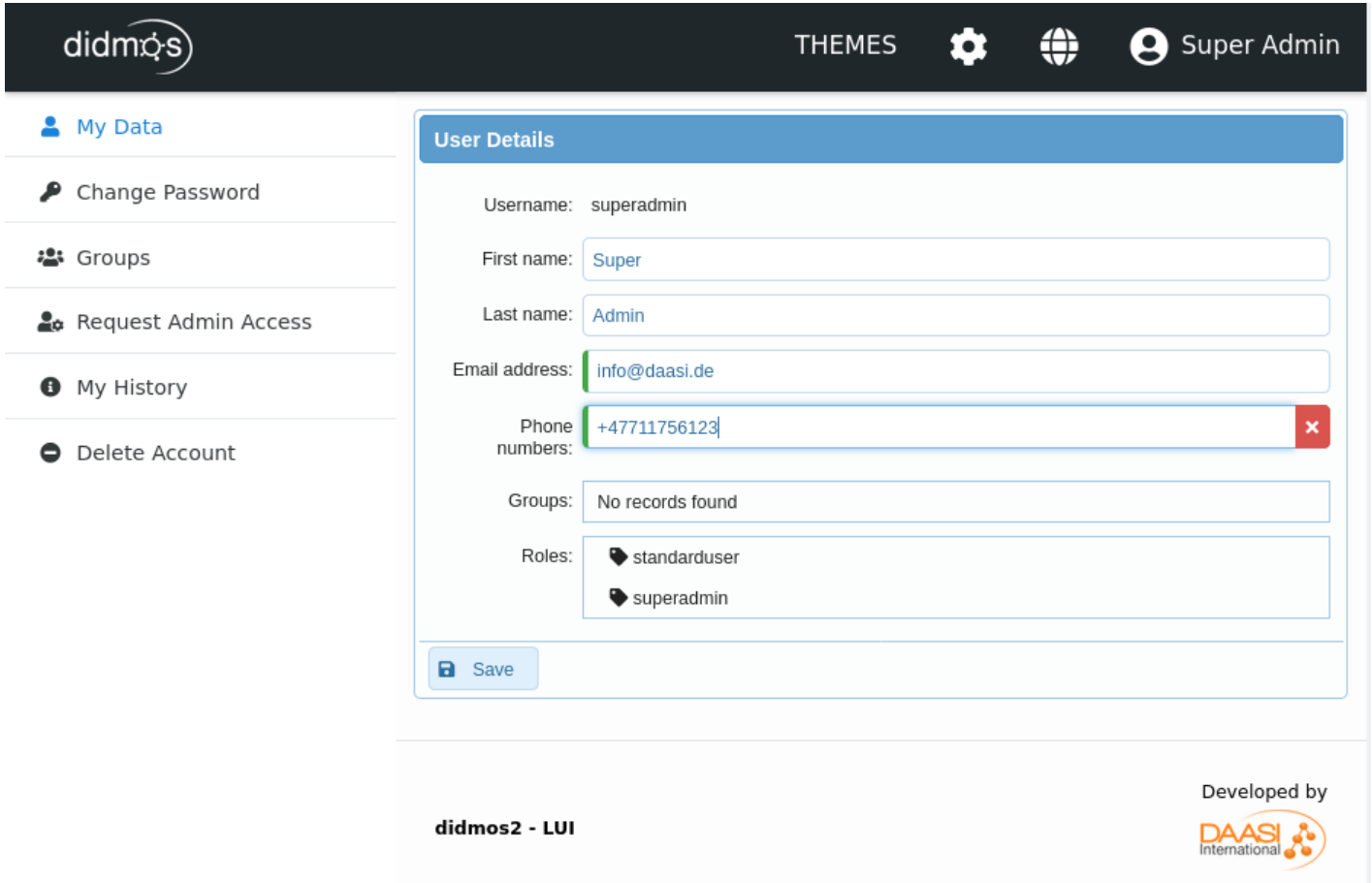
didmos LUI modules

- … more common functionality in shared library:
  - Activity-Log for compliance
  - Multi-tenancy capable
  - Multi-step delete process
  - MFA token management (with PrivacyIDEA)
- Customised extensions for any desired function and process possible, e.g.:
  - Portal functionality
  - Other data objects with requests and managements (e.g. companies, affiliation, services)
  - Multiple accounts per identity
  - Etc.

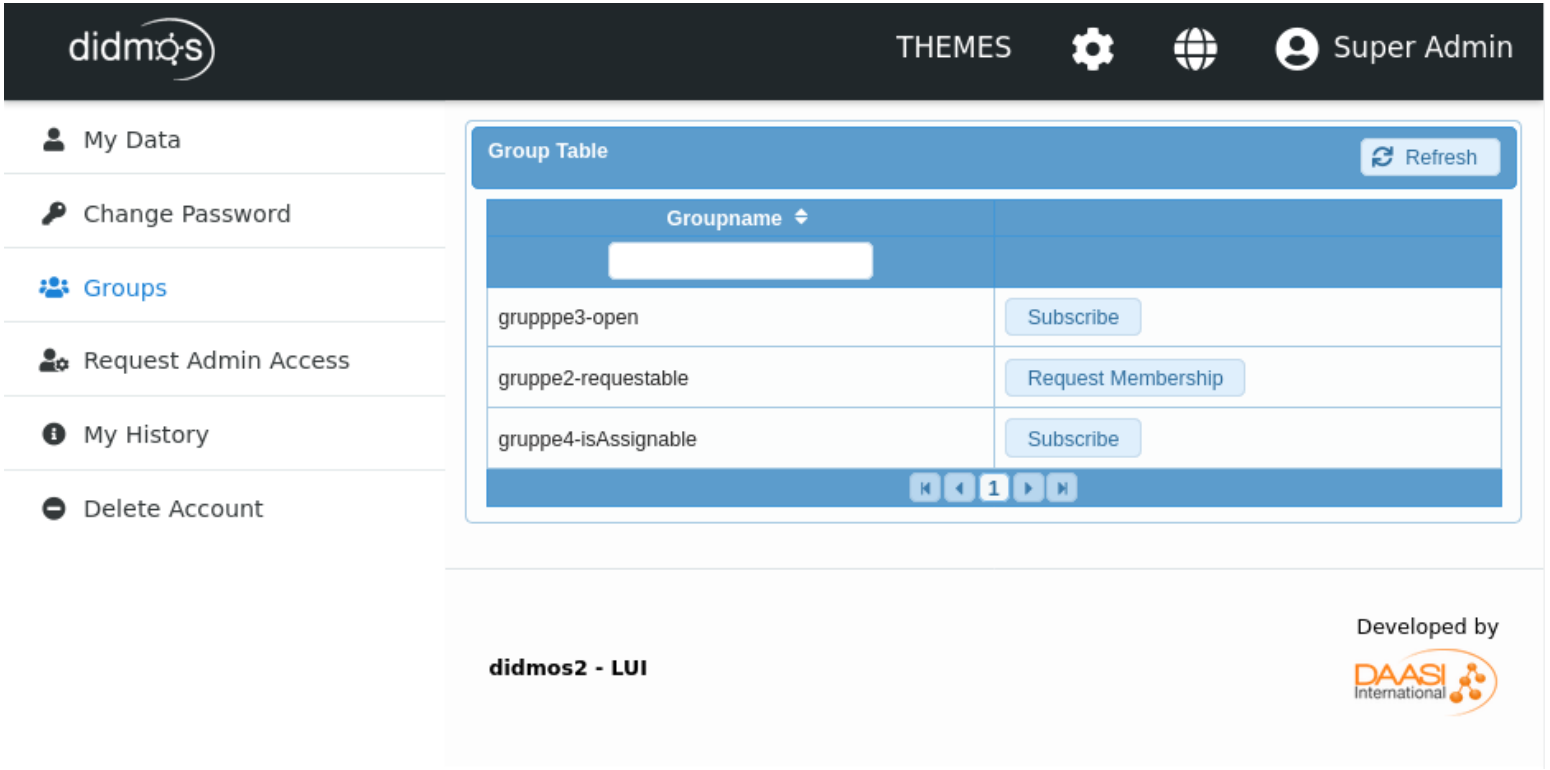
didmos LUI
Self-Service password change



didmos LUI
Self-Service profile



didmos LUI
Group management

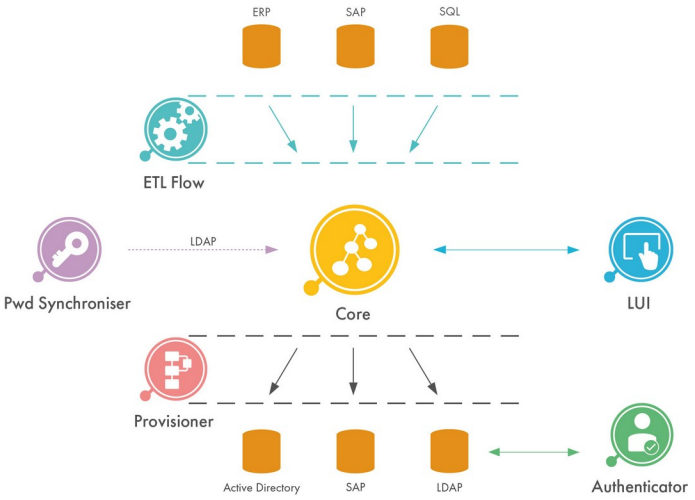


didmos Auth

Authenticator

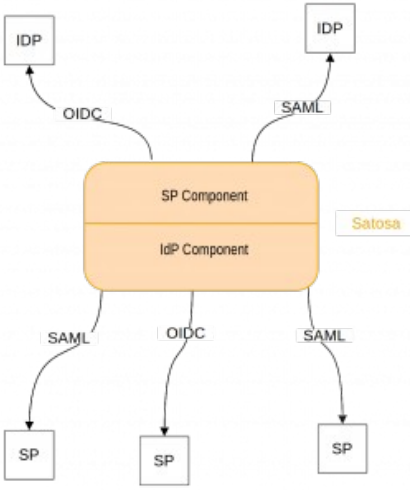
SSO proxy based on Satosa

- Protocols: SAML2, OIDC, Social IDPs
- Extensions for communication with other didmos modules



didmos Auth - Satosa

- For base functionality Auth uses Satosa and the Python-libraries of the IdentityPython project (pysaml, pyoidc, etc.)
- **Frontends/IdPs:** Those connect to “downstream” SPs
  - SAML2
  - OIDC
- **Backends/SPs**: Those connect to “upstream” IdPs
  - SAML2 (e.g. eduGAIN)
  - OIDC
  - Social IdPs (which mostly are OIDC flavors)
- **Microservices**: Do *something* during requests or responses
- Architecture is thus a proxy compliant to AARC BPA



didmos Auth - Extensions

- We have extended Satosa with
  - Several Satosa modules (backends and microservices)
  - A Django-based application for user interaction (with i18n/templates)

- **didmos Backends for Satosa**
  - **LDAP:** Authentication against LDAP server
    - This is mostly used for “local” users managed in didmos
  - **SQL**: Authentication against SQL

didmos Auth – Extensions cont'd

- **didmos Microservices for Satosa**
  - **Attribute resolver:** Retrieve attributes from different sources
    - SCIM - connects to didmos Core to query user attributes, group memberships etc. for a user
    - SQL
  - **"Shadow account" registration:**
    - Based on configuration, Satosa can create a unique ID for a user from proxy backends (e.g. SAML2)
    - That unique ID is used to create/update a "shadow account" in didmos Core to allow management of that user in didmos (e. g. assign roles, groups, Self-Service access etc.)
    - Typically all user attributes received via SAML2 are mapped and stored in didmos Core, but attributes could also be proxied transparently

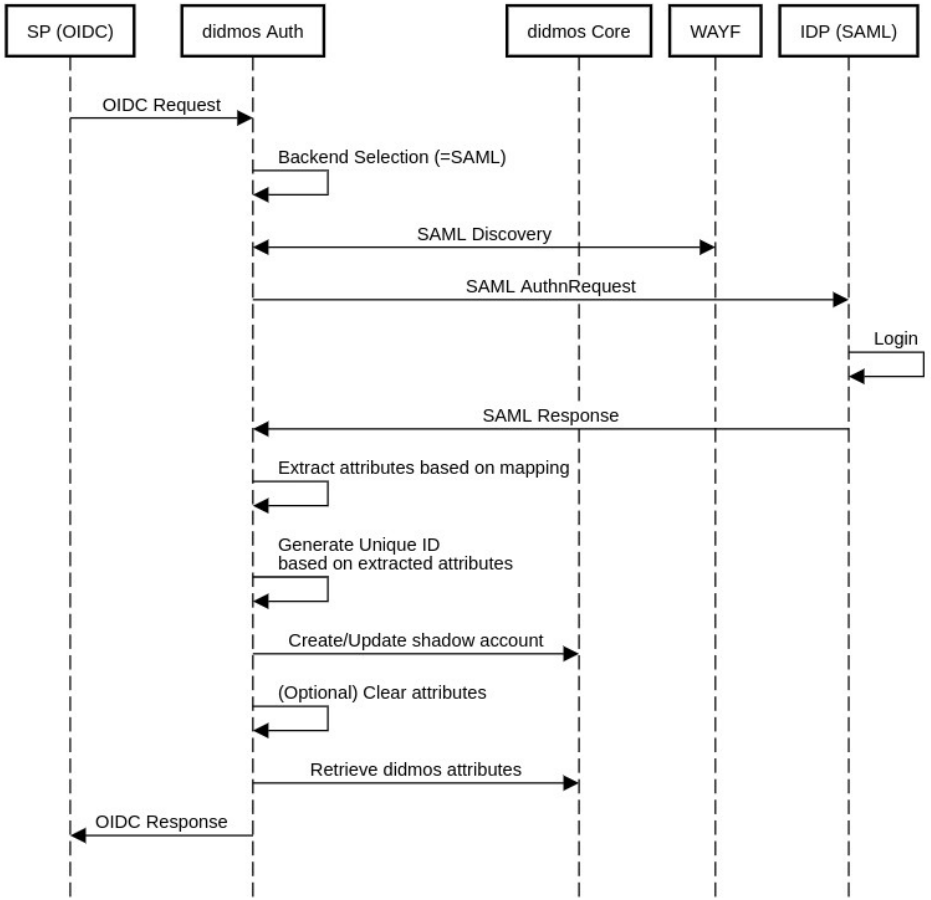
DAASI International

didmos Auth – Extensions cont'd

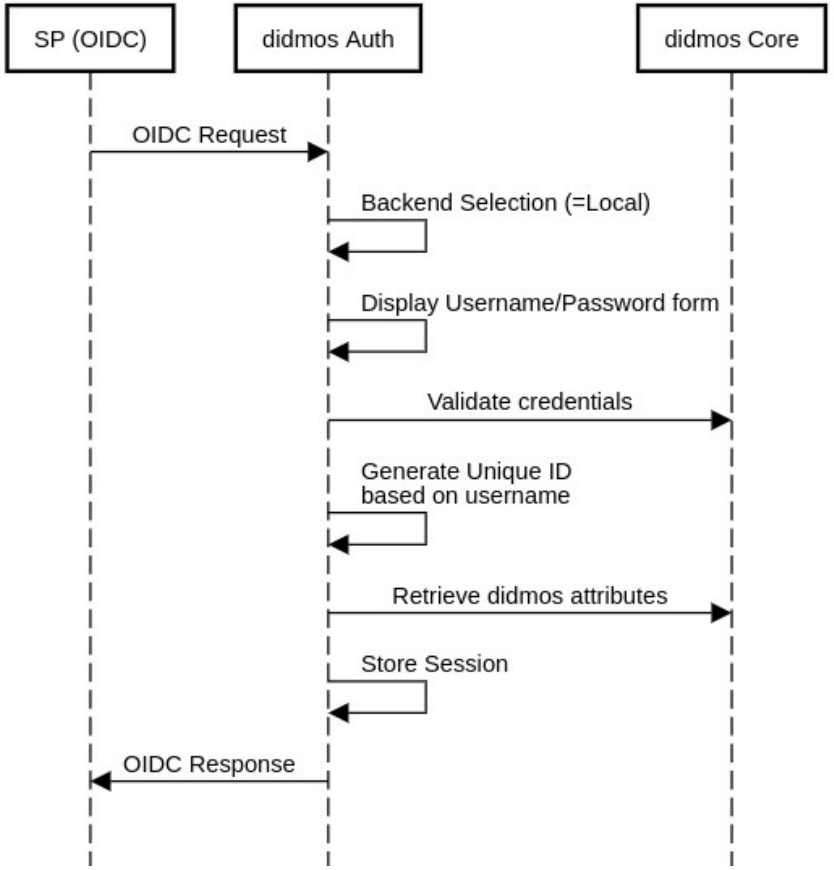
- **… didmos Microservices for Satosa**
  - **Backend selection:** Select authentication backend based on different methods
    - Selection UI
    - Domain-based
  - **Session management:** Store backend selection and LDAP/SQL authentication results
    - Note: Proxied authentication (e.g. at an upstream SAML2 IdP) is not stored in the Satosa session (but could be)
  - **MFA**: Perform MFA against didmos Core and PrivacyIDEA

DAASI International

Typical login flow (Proxied)



Typical login flow (Local)



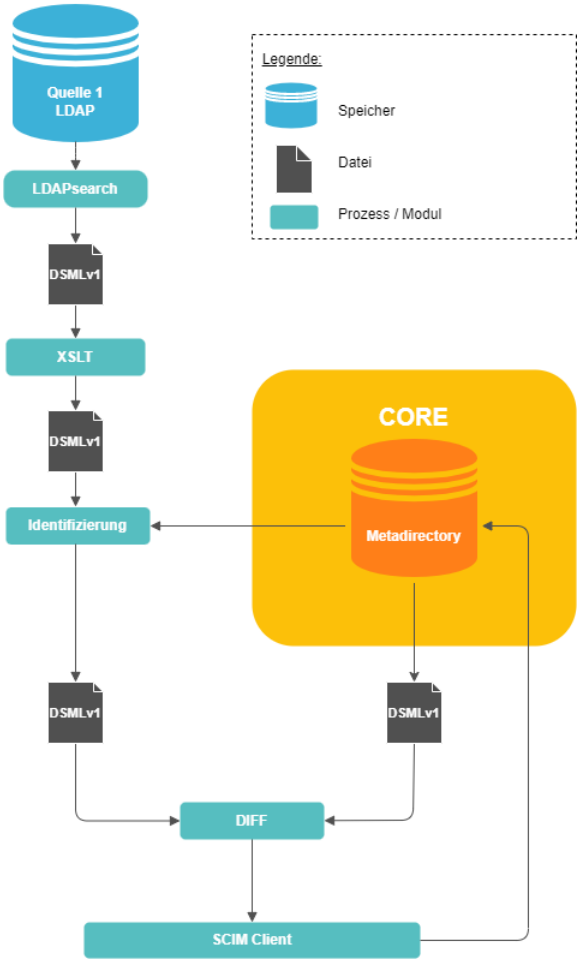
didmos ETL Flow

- **E**xtract **T**ransform **L**oad
- The workflow is configured via LDAP objects (=configuration)
- Allows parallel processing of tasks
- Values derived from tasks can be saved in storage and/or be extracted as file
- Conditional jumps to other tasks are possible (on error)
- When executing tasks it is possible to use values derived from other tasks as parameters
- Typically used for either one-time data migration or regular synchronization from source systems

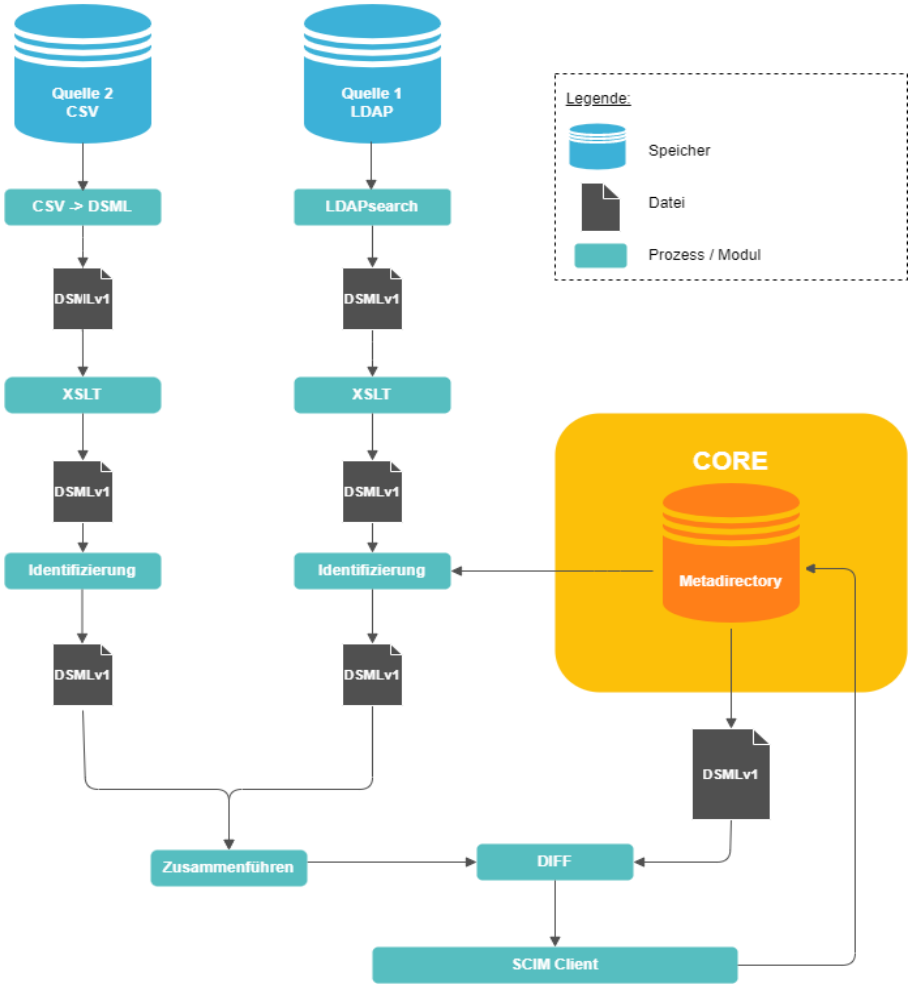
didmos ETL Flow – Standard Flow

- Reading data sources (e.g. ldapsearch)
- Plausibility check before conversion
- Conversion to DSMLv1 as shared format
- Preparation via XSLT to data sets comparable with the IdM
- Identifying against the IdM
- Merging identical data sets within one data source if necessary
- Merging multiple data sources
- Reading the IdM
- Calculating the difference
- Installing changes

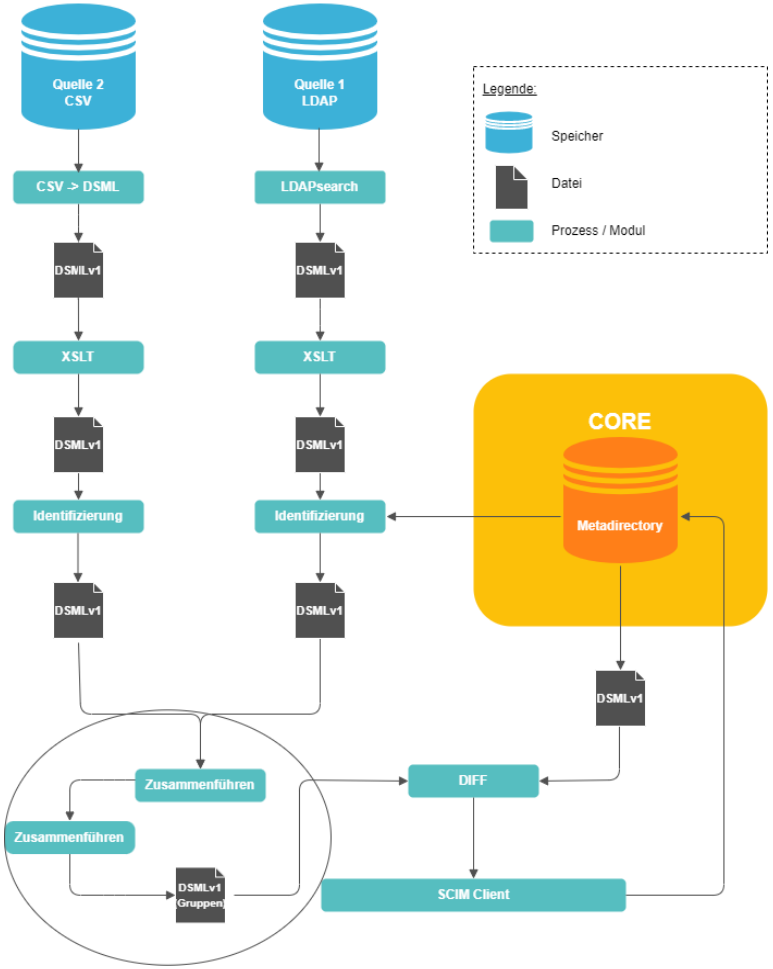
didmos ETL Flow



didmos ETL Flow



didmos ETL Flow



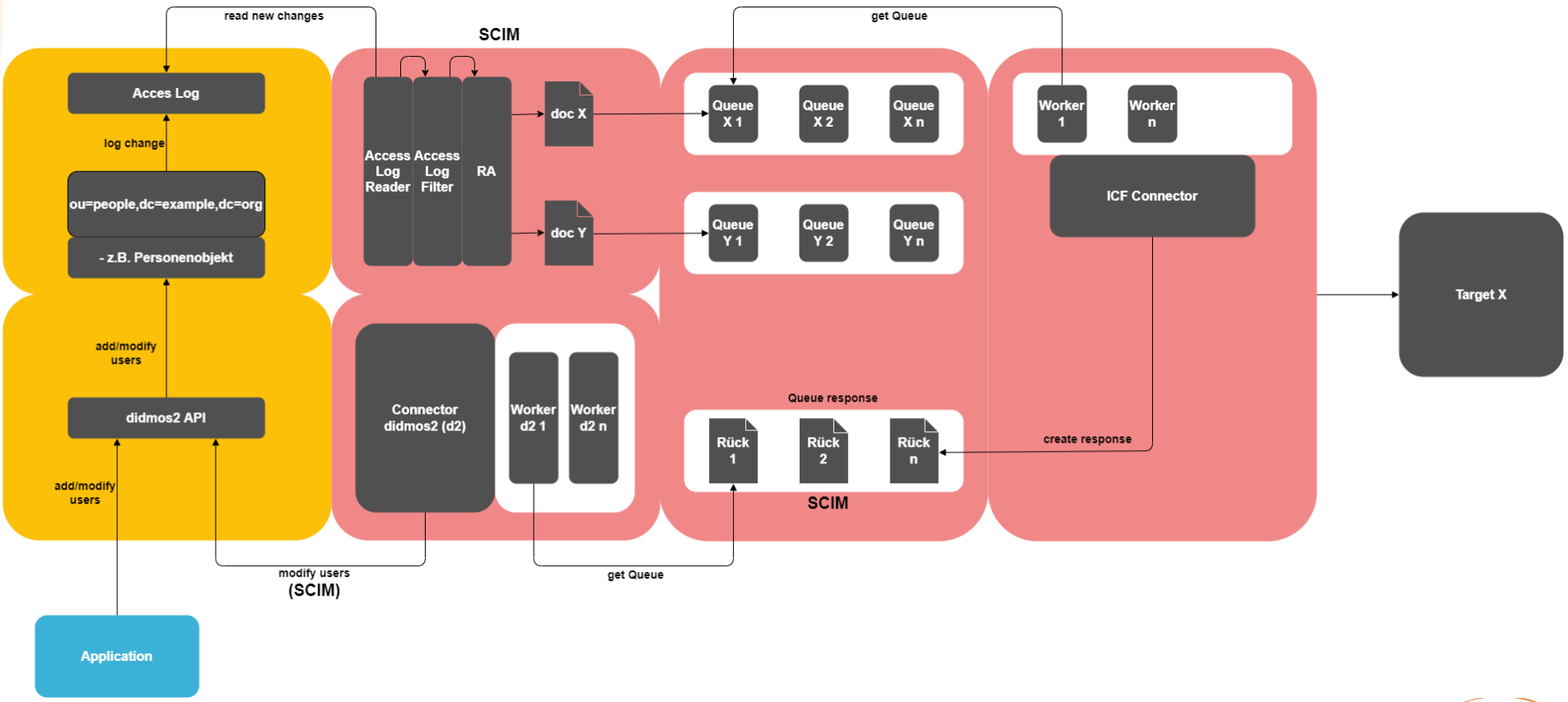
didmos Provisioner

- Real-time transfer of identity information to connected target systems
  - Reads change information in the OpenLDAP accesslog
  - Architecture is inspired by SPML, however it now implements SCIM as well
- Changes are recorded in JSON documents which are inserted into the queuing system RabbitMQ
- A dedicated worker monitors new documents in the queue and then writes the changes to the target system

didmos Provisioner - Worker

- Generally all ICF compatible connectors can be integrated in workers
- We have workers for
  - LDAP
  - Active Directory
  - SCIM2
  - Gluu
  - Logging
  - E-Mail
  - Other proprietary systems of customers

didmos Provisioner



What we are currently working on

- Several active didmos customer projects that lead to new features in all modules
- Common configuration store for all components in LDAP (“Configserver”)
  - Challenging for components that require custom config structure, such as Satosa, that by default only works with yaml files
  - Approach: dynamically generate needed config format for component from LDAP objects
  - Config management in LUI admin interface
- Extend didmos for eduID use cases
  - Bonafid: eduID and IAMaaS for NRENs in Africa
  - Evaluated using didmos for eduID systems in Germany

DAASI International

Thank you for listening.

DAASI International

Phone: +49 7071 407109-0

email: info@daasi.de

Web: www.daasi.de