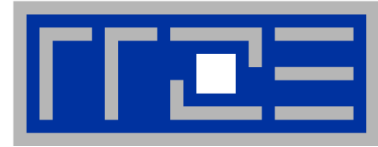


# The web front end WAID

and other developments of IdM  
project Erlangen  
(IDMone)





- **Web front end WAID**
  - **Web Administration for IDentity management**
- **and other developments**
  - **Matching – matching people**
    - **Integration in Novell's Identity Manager**
  - **First-Aid IdM-Toolkit**
    - **jpwgen**
    - **jidgen**
    - **idmsec**
    - **m.a.r.v.**



# WAID – Web Administration for Identity Management



# Why develop a new interface?

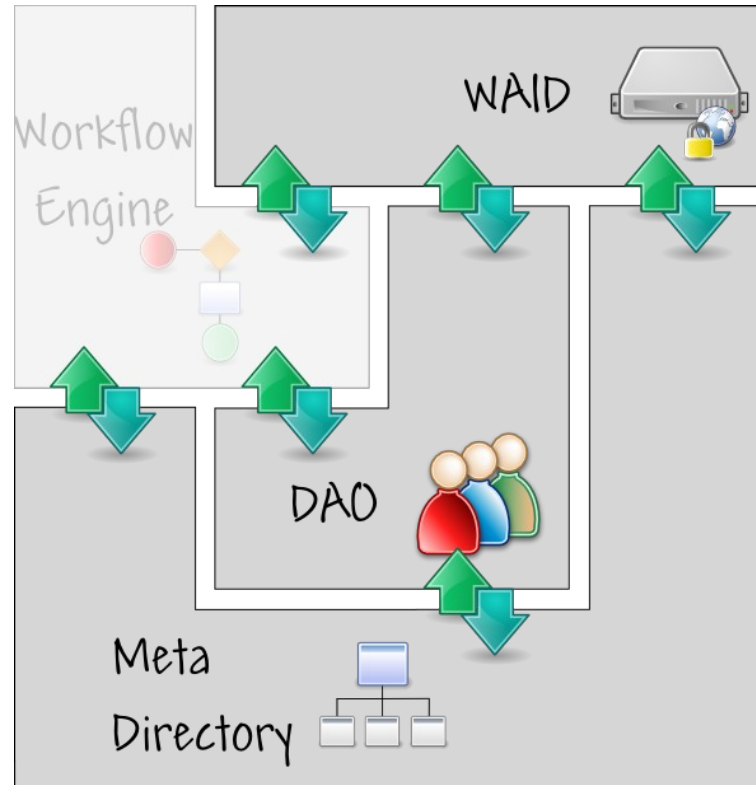


- **Original web interface:**
  - **Novell's User Application**
- **Blog- Entry March 11th 2008 (Peter Rygus)**
  - **[...] With all its requirements, IDMone overburdened User Application in such a way that we have decided to create our own interface. While this delays IDMone's production schedule noticeably, it promises a better, more flexible and accessible web interface in the long run.[...]**
- **Main reasons**
  - **Customizations are usually very time-consuming**
  - **results are only a compromise**

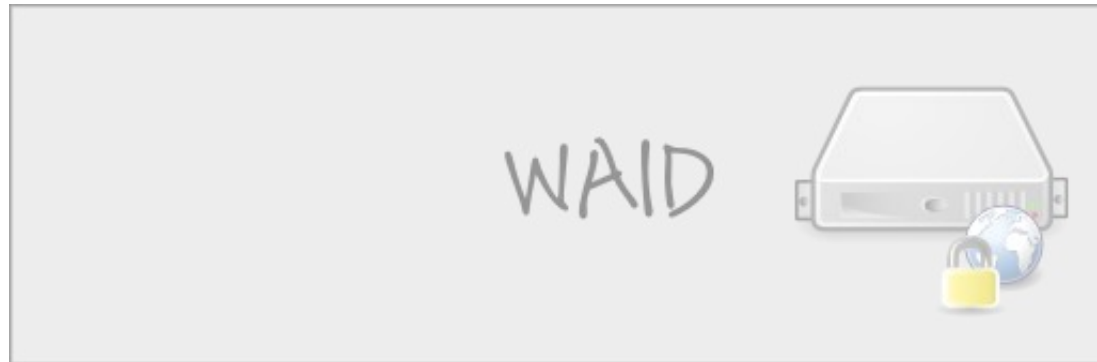


- **the product behind the FAU's Identity Management Self Service**
  - <https://www.idm.uni-erlangen.de/>
- **based on popular Java Frameworks**
  - Tapestry as Web Framework
  - Spring-Framework incl.
    - Spring Security und Spring LDAP
  - <http://www.blogs.uni-erlangen.de/IDM/stories/2375/>
- **based on the FAU's construction kit**
  - <http://www.vorlagen.uni-erlangen.de/>
  - accessible websites
  - FAU Corporate Design
  - clear separation between content and visual design

# Big Picture



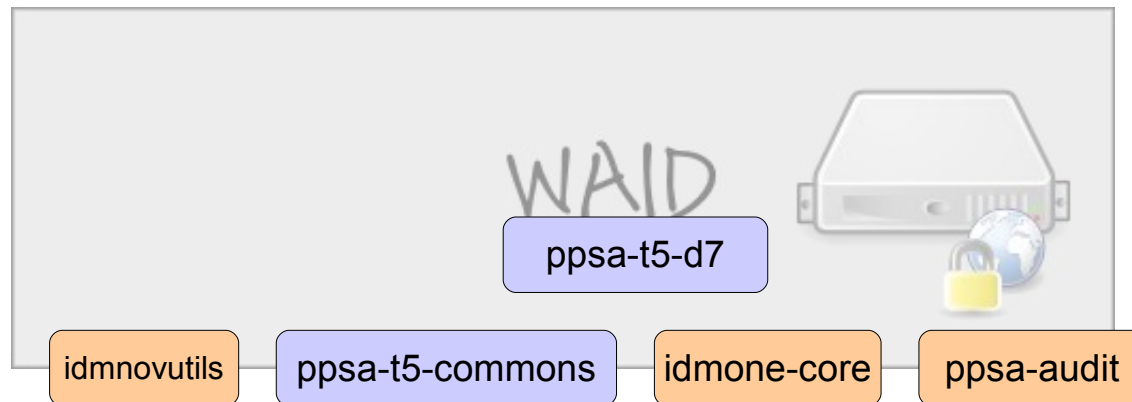
- Reusable modules



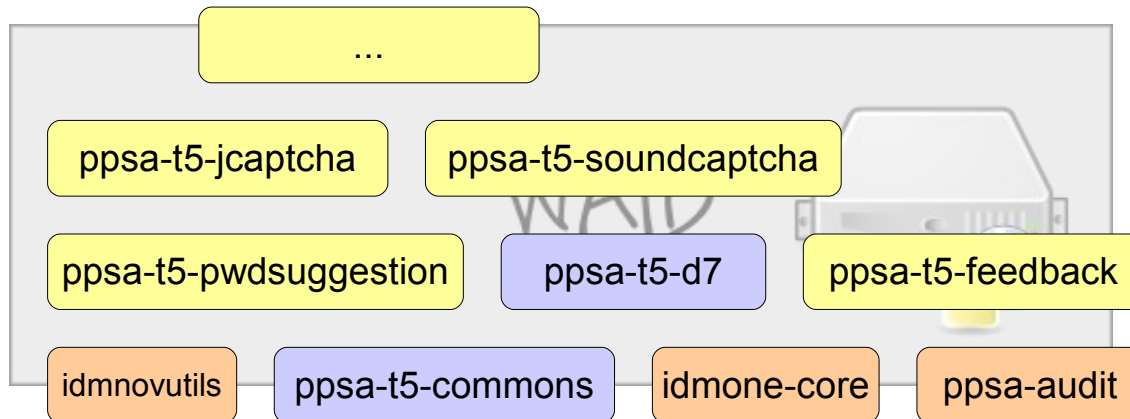
- **Reusable modules**
  - e.g. FAU web construction kit for Tapestry
  - Basic system is already in use for other applications



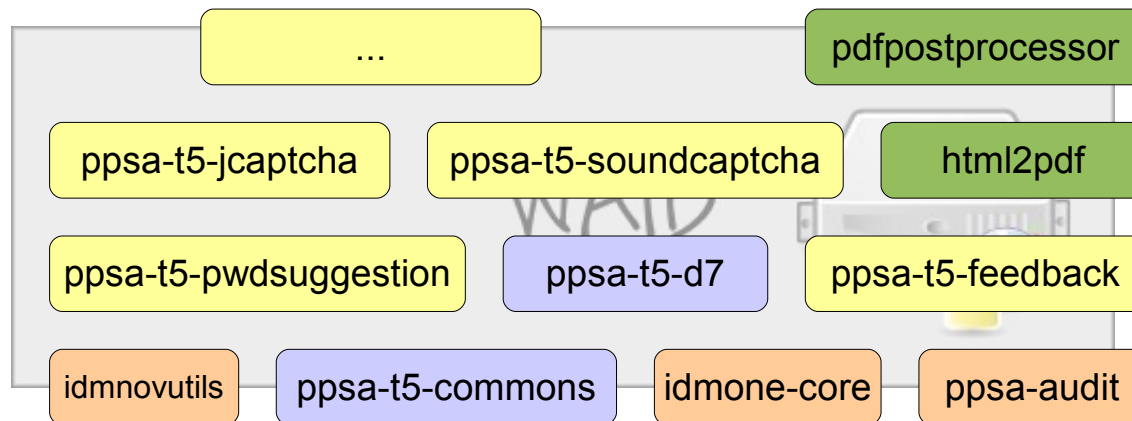
- **Reusable modules**
  - e.g. Novell utilities
  - Library is also used by console tools



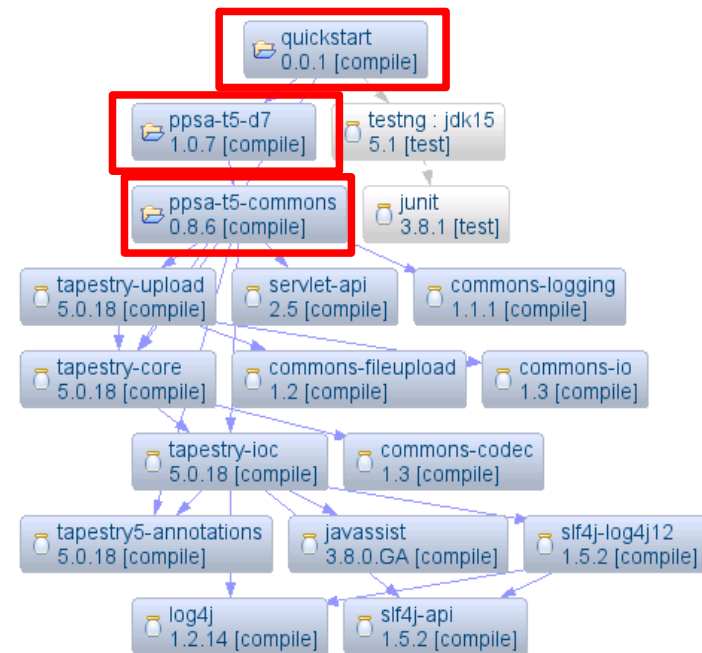
- **Reusable modules**
  - e.g. password generator service
  - **Generates passwords compliant with Novell's password policy (without configuration!)**



- **Reusable modules**
  - e.g. PDF export
  - any XHTML pages incl. pictures (watermarks, ...)



- **Quickstart in 2 minutes via Maven-Archetype**
  - incl. corporate design
  - incl. dynamic menu
  - Eclipse Demo
- **Dependency tree:**





- **completely localized web front end**
  - currently in German and English
  - expandable at will
- **Accessible websites (98 of 100 points)**
- **Corporate design**
- **no compromises**
- **reduced expenditure of time for expansions**



- **Features**
  - **data privacy self information**
  - **show service(s)**
  - **show affiliations**
  - **change password**
  - **security questions (Challenge & Response)**
  - **feedback page**
  - **PDF printing**
    - **User information letter**
    - **Service information letter**
  - **Functions for admins**
    - **Search for users**
    - **Set (initial) password**
    - **Print user information letter**
    - **Affiliation display**
    - **Service display**
  - ...



- **Components are to be published**
  - **Conditions**
    - **Product maturity**
    - **Documentation**
  - **Open Source**
    - **Licence: GPL or LGPL**
- **Your own IdM-Project?**
  - **join us**
  - **or ask for assistance**
- **if you are interested**
  - **contact us**
    - **[ids@rrze.uni-erlangen.de](mailto:ids@rrze.uni-erlangen.de)**



# Matching – matching people Integration in Novell's Identity Manager



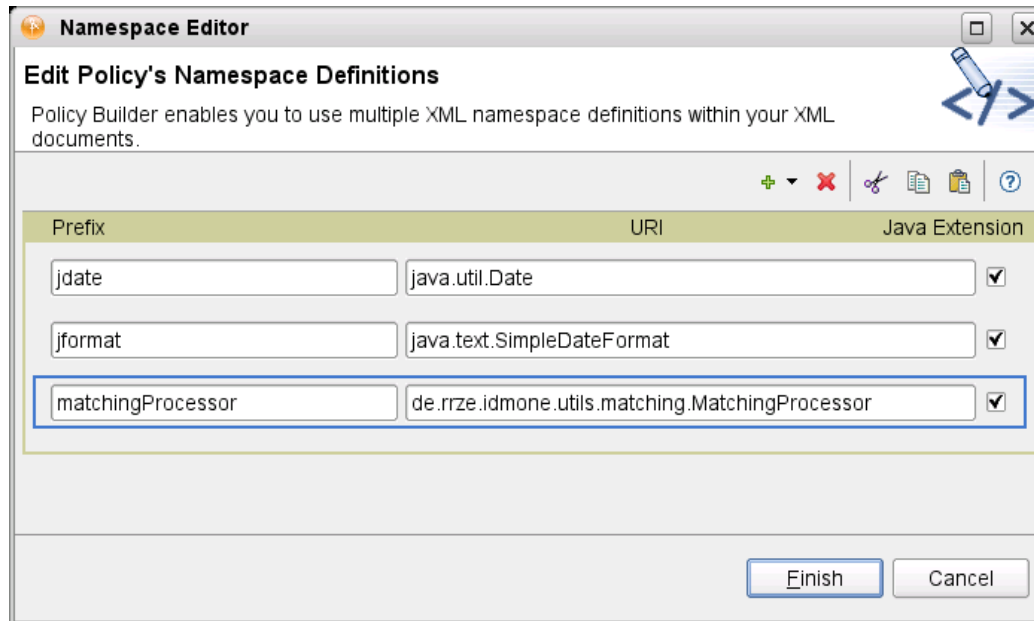


- **Matching basics:**
  - see presentation: **Data Linkage in IDM Systems**  
Krasimir Zhelev, 20.02.08 BRZL AK MetaDir
  - **Rules engine (Drools)**
  - **Similarity functions**  
(phonetical, **Pattern Matching**, combined)
- **Result**
  - a Java library
- **Simple usage**
  - **Input:**
    - a new person
    - possible candidates (= existing people)
  - **Output:**
    - best hit incl. probability
    - second-best hit incl. probability
    - ...

# Preparation



- copy JAR files to
  - `/opt/novell/eDirectory/lib/dirxml/classes/`
- **DirXML-Policy: Extending the namespace**





- Variables for given name, surname, date of birth and place of birth
- Initialising the MatchingProcessor with a “new person”
- Adding possible candidates

```
Actions
✓ ⚡ set local variable("base", scope="policy", Parse DN("ldap", "dest-dn", Global Configuration Value("PEOPLE_CONTAINER")))
✓ ⚡ set local variable("givenname", scope="policy", Operation Attribute("Given Name"))
✓ ⚡ set local variable("surname", scope="policy", Operation Attribute("Surname"))
✓ ⚡ set local variable("dateofbirth", scope="policy", Operation Attribute("schacDateOfBirth"))
✓ ⚡ set local variable("placeofbirth", scope="policy", Operation Attribute("schacPlaceOfBirth"))
✓ ⚡ set local variable("matchingProcessor", scope="policy", object(XPath("matchingProcessor:new($givenname.$surname.$dateofbirth.$placeofbirth)"))
✓ ⚡ set local variable("candidates", scope="driver", nodeset(Query(class name="User", max-result-count="200", dn(Parse DN("ldap", "dest-dn", Global Configuration Value("PEOPLE_CONTAINER")), match("Given Name", Substring(length="2", Local Variable("givenname"))+***), match("Surname", Substring(length="1"
—*), "Given Name", "Surname", "schacDateOfBirth", "schacPlaceOfBirth)))
✓ ⚡ if
    if local variable 'candidates' not equal ""
then
    set local variable("match_result", scope="policy", object(XPath("matchingProcessor:addNodeSet($matchingProcessor.$candidates)"))
else
```



- **Start analysis**
- **Variables for first DN, first coefficient, second DN, first coefficient, second coefficient**
- **Actual decision via DirXML-Script**

```
✓ ↗ set local variable("match_result", scope="policy", object(XPath("matchingProcessor:nodeSetMatching($matchingProcessor))))
✓ ↗ set local variable("firstDn", scope="policy", object(XPath("matchingProcessor:getFirstDn($matchingProcessor))))
✓ ↗ set local variable("firstCoeff", scope="policy", object(XPath("matchingProcessor:getFirstCoeff($matchingProcessor))))
✓ ↗ set local variable("secondDn", scope="policy", object(XPath("matchingProcessor:getSecondDn($matchingProcessor))))
✓ ↗ set local variable("secondCoeff", scope="policy", object(XPath("matchingProcessor:getSecondCoeff($matchingProcessor))))
✓ ↗ if
    if local variable 'firstCoeff' greater than "0.93"
    then
        set operation destination DN(dn(Local Variable("firstDn")))
        if
            if local variable 'secondCoeff' greater than "0.80"
            then
                trace message(level="3", "second coeff very very high")
            else
                trace message("good match")
        else
            if
                if local variable 'firstCoeff' greater than "0.70"
                then
                    trace message(level="3", "possible match")
                else
                    trace message("no match")
```



- **Expandable Matching**
  - due to Rules-Engine, Java, ...
- **Full integration in Novell's Identity Manager**
  - no external LDAP connection necessary
  - Input and
  - Analysis are done in DirXML
- **important parameters can be changed at any time**
  - no programming skills necessary!
- **everything in one library**
  - usable from any source driver
  - with a simple “Link a policy”



- **Planned upgrade for generic use**
  - **Looking for a project partner for further development**
- **Planned publications based on GPL**
- **Acquisition so far is only possible within the scope of joined projects**

## First-Aid IdM-Toolkit

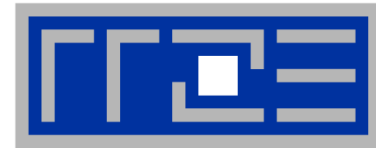
jpwgen

jidgen

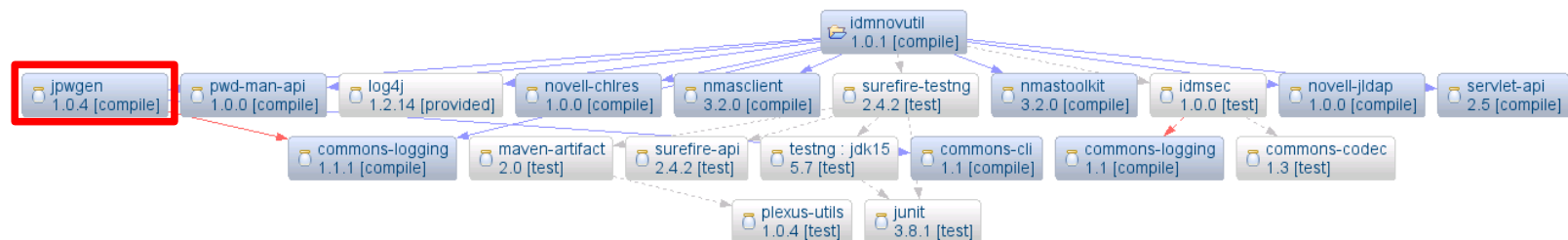
idmsec

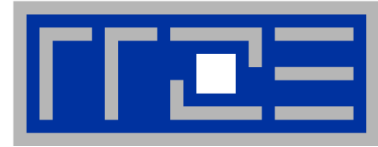
FAQ

MARV



- <http://jpwgen.berlios.de/>
  - Open Source (LGPL)
- Java-based password generator
  - Java-Version of the well-known pwgen program
- Usage
  - via command line or
  - as library
- Application in ppsa-t5-pwdsuggestion
  - generates password suggestions in WAID
  - compliant with Novell's password policy!



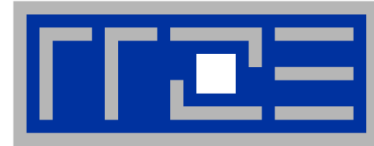


- <http://jidgen.berlios.de/>
  - Open Source (LGPL)
  - Version 0.8
- Java-based ID-Generator
- Usage
  - via command line or
  - as library
- Configurable via template language
  - Examples:
    - `java -jar jidgen.jar -Ts i -Ty 2008 -TI Doe -Tf John -T s:y1:2l:2f`
      - i8dojo
    - `java -jar jidgen.jar -B -Bf blacklist_file -T C+:V+:N2+:C+:V+:C+`
      - no35dax

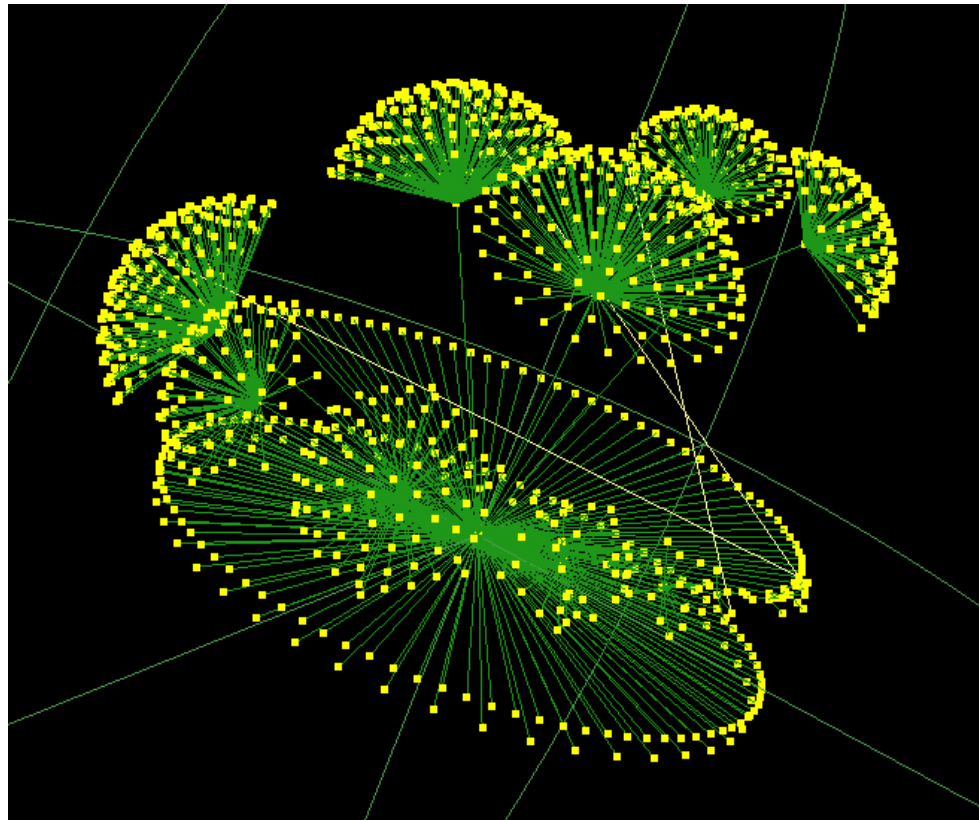


- **Provides a password service**
  - creation of password hashes for all systems
- **Supports algorithms and encoding**
  - MD5, SMD5
  - SHA, SSHA
  - UNIX CRYPT, UNIX SMD5 CRYPT
  - Apache MD5, HIS/SOS Apache MD5
  - OpenBSD-style Blowfish
  - SMB, LANMAN, NTUNICODE
  - APHELION
  - ...
- **Call**

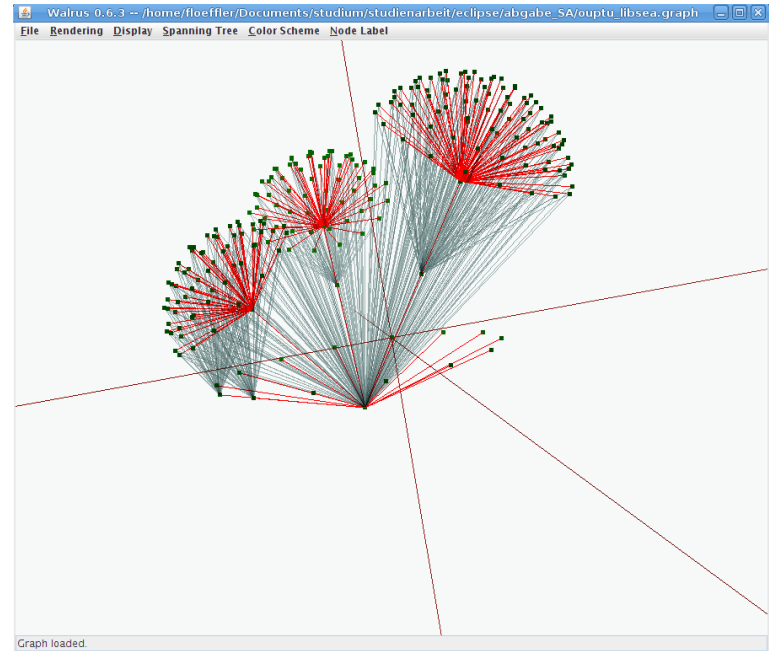
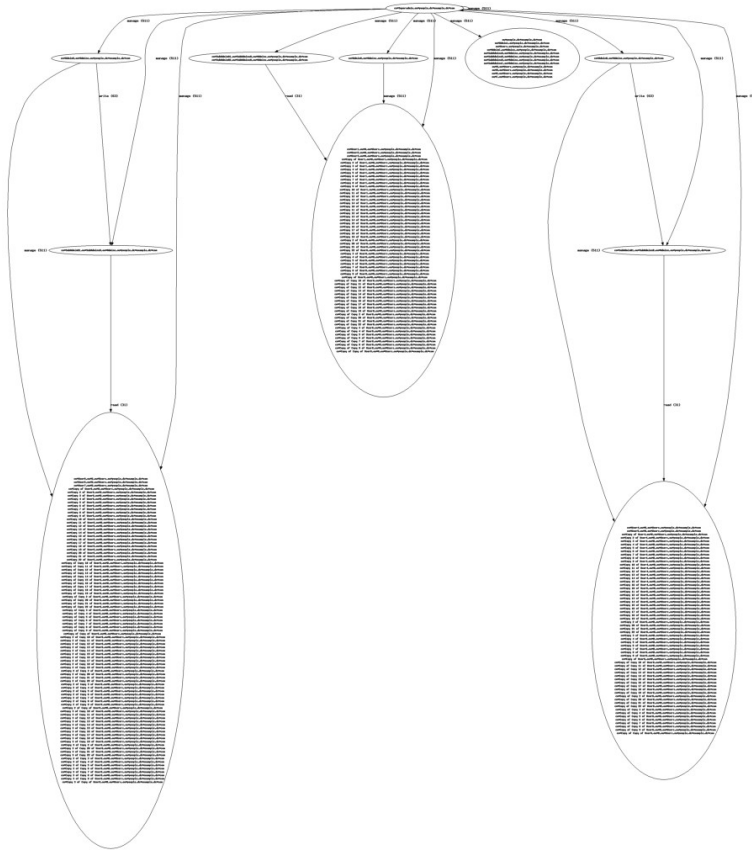
```
✓ ↗ set local variable("pwdService", scope="policy", object(xpath("pwdService:getInstance()")))
✓ ↗ set destination attribute value("fauSMD5Unix", XPath("pwdService:crypt($pwdService, add-attr[@attr-name='nspmDistributionPassword']/value, 'UMD5')"))
✓ ↗ set destination attribute value("fauSMD5Password", XPath("pwdService:crypt($pwdService, add-attr[@attr-name='nspmDistributionPassword']/value, 'HIS')"))
✓ ↗ set destination attribute value("unixPassword", XPath("pwdService:crypt($pwdService, add-attr[@attr-name='nspmDistributionPassword']/value, 'UNIX_CRYPT')"))
✓ ↗ set destination attribute value("fauSSHAPassword", XPath("pwdService:crypt($pwdService, add-attr[@attr-name='nspmDistributionPassword']/value, 'SSHA')"))
✓ ↗ set destination attribute value("ntPassword", XPath("pwdService:crypt($pwdService, add-attr[@attr-name='nspmDistributionPassword']/value, 'SMB_NTUNICODE')"))
✓ ↗ set destination attribute value("fauBFPassword", XPath("pwdService:crypt($pwdService, add-attr[@attr-name='nspmDistributionPassword']/value, 'BCRYPT')"))
```



- **Modular Access Rights Visualization**
  - Visualising solution for LDAP Access Controls
  - Status quo: OpenLDAP-ACLs (not complete)



# M.A.R.V. - further examples





Any questions?





Thank you!

