TraIT TTP

Technical Specification

Draft

Revision 3.0

9/01/2014

CONFIDENTIAL

KORTRIJKSESTEENWEG 214 B3
B-9830 SINT-MARTENS-LATEM (BELGIUM)

PHONE : +32 (0) 9 210 78 90
FAX : +32 (0) 9 211 09 99

E-MAIL : INFO@CUSTODIX.COM
WEBSITE : WWW.CUSTODIX.COM

Document description

Document Id	20131209 - TraIT TTP Technical Specifications v3.docx	Status	Draft
Document Type	Technical Specifications	Security	CONFIDENTIAL
Revision	3.0	Date	9/01/2014
Title	TraIT TTP		
Subject	Technical Specification		

Document history

Date	Revision	Author	Changes
20131205	1.0	David Voets	Initial version
20131209	2.0	David Voets	Complete PIMS WS specifications
20140108	3.0	David Voets Sam Vanloocke Jelle Van Den Driessche	Incorporated corrections and remarks.

Keywords

TraIT RFP TTP CTMM

Abstract

This document contains the TraIT TTP technical specifications. Please note that this version is still in a draft state and that the new and existing interfaces subject to TraIT customisation may be subject to changes.

Notes & remarks

[Comments]

Table of contents

1 Introduction

This document contains the TraIT TTP technical specifications. Please note that this version is still in a draft state and that the new and existing interfaces subject to TraIT customisation may be subject to changes.

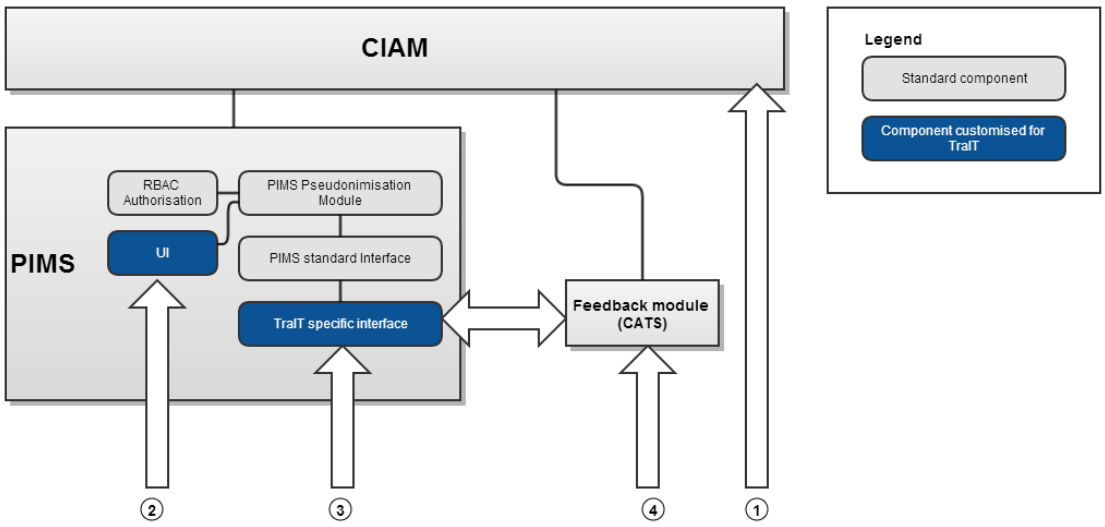
2 References

This specification document refers to the following documents:

- TraIT TTP Service – Request for Proposals (Annex 4_RFP TTP TraIT_v10 20130515.docx) by Andre Dekker (MAASTRO Clinic)
- Custodix offer TraIT TTP service v1.1 (20130930-TraIT RFP Custodix Proposal – v1.1.pdf) by David Voets and Brecht Claerhout (Custodix)

3 Overview

The TraIT TTP is composed from the following components:



- CIAM: Identity and Access Management
  - IdM: Identity management (including management of credentials and attributes)
  - IdP: Identity Provider for end-user authentication and single-sign-on (SSO)
  - STS: Security Token Service for Web Service client authentication
- PIMS: pseudonym issuance and record linkage
- CATS: security and privacy-related data transformations. In the context of TraIT this service is used to transpose “additional data requests” and to make them available to the submitting site(s).

For a more detailed overview, we refer to the TraIT TTP Proposal document mentioned under *References*.

4 Standards and interfaces

4.1 Web interfaces (GUI)

CIAM (IdM and IdP), PIMS and CATS offer a graphical user interface in the form of a web (HTTP(S)) interface. The presentation technology used is based on HTML, CSS, Javascript and Ajax. For more details on how to operate these web applications, we refer to the respective user manuals.

4.1.1 Browser compatibility

The web interfaces have been designed for the actively supported versions of the following browsers: Microsoft Internet Explorer, Google Chrome and Mozilla Firefox.

4.1.2 Authentication

PIMS and CATS act as Service Providers (SPs) according to the SAML v2.0 model and will redirect the user to the CIAM IdP if an authenticated session is required. This occurs in accordance to the SAML v2.0 Web Browser SSO profile[1].

4.1.3 Session management

CIAM IdP relies on cookies to enable SSO (i.e. to avoid that the user has to authenticate each time (s)he is redirected to the IdP. CIAM IdM, PIMS and CATS rely on URL rewriting to pass the session id to the client.

4.2 Web Service interfaces (API)

CIAM, PIMS and CATS expose a SOAP v.1.1[2] Web Service interface. The exposed Web Services advertise their interface in the form of a WSDL v1.1[3] document.

These Web Services advertise their security requirements in the form of Policy annotations (WS-Policy[4]) annotated to the appropriate WSDL elements (WS-PolicyAttachment[13]). The security policies are expressed according to the WS-SecurityPolicy v1.2[5] standard.

For authentication (and authorization) PIMS and CATS rely on a WS-Security[6] token to be included in each request requiring authentication. This security token must be a SAML[7] v.2.0 issued by the CIAM STS according to the WS-Trust v1.4[8] specification (as specified by the “IssuedToken“ security policy).

[1] http://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf
[2] http://www.w3.org/TR/2000/NOTE-SOAP-20000508/
[3] http://www.w3.org/TR/wsdl
[4] http://schemas.xmlsoap.org/ws/2004/09/policy/
[5] http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702/ws-securitypolicy-1.2-spec-os.html
[6] https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss
[7] http://saml.xml.org/
[8] http://docs.oasis-open.org/ws-sx/ws-trust/v1.4/ws-trust.html

5 Environments

5.1 Staging environment

Application / service	URL	Notes
CIAM IdM web portal	https://ciam-dev-trait.custodix.com/idm/	Used for user administration purposes. An account (login) is required in order to use this application.
CIAM IdM Web Services	https://ciam-dev-trait.custodix.com/idm/services	Used for automated user administration.
CIAM Identity Provider (IdP)	https://ciam-dev-trait.custodix.com/idp/login.jsp	Does not need to be accessed directly by end-users. PIMS and CATS will automatically redirect the user's browser to this service when login is required.
CIAM Security Token Service (STS)	https://ciam-dev-trait.custodix.com/sts/	STS with X.509-based authentication[9] .
PIMS web portal	https://pims-dev-trait.custodix.com/pims/	Used for manual interaction with PIMS.
PIMS Web Services	https://pims-dev-trait.custodix.com/pims/services/	The relevant service for requesting target identifiers is PseudonymizationService[10]
CATS web portal	https://pims-dev-trait.custodix.com/cats/	Used for manual interaction with CATS (i.e. uploading or downloading additional data requests).
CATS Web Services	https://pims-dev-trait.custodix.com/cats/services	Can be used for automatically uploading[11] or downloading[12] additional data requests.

[9] WSDL location for X.509-based authentication is https://ciam-dev-trait.custodix.com/sts/services/X509STS?wsdl.
WS-MetaDataExchange endpoint is https://ciam-dev-trait.custodix.com/sts/services/X509STS/mex
[10] WSDL accessible at https://pims-dev-trait.custodix.com/pims/services/PseudonymizationService?wsdl
[11] WSDL accessible at https://pims-dev-trait.custodix.com/cats/services/upload?wsdl
[12] WSDL accessible at https://pims-dev-trait.custodix.com/cats/services/messages?wsdl

6 Terminology

PIMS relies on a somewhat different terminology than is used in the TraIT TTP RFP. Although the PIMS graphical user interface (web site) will be customized to use the TraIT terminology, native PIMS concepts will still be exposed through the API (web service interface). In order to clarify what is meant with the different PIMS concepts, we provide the following mapping table explaining the correspondence between PIMS and TraIT concepts:

PIMS concept	TraIT concept	Remarks
Catalogue	(Single de-identified) target collection	A catalogue defines the scope for issuing target identifiers. Sites can submit identifiers to multiple *catalogues*(target collections), yielding multiple different target identifiers (even if based on the same subject).
Enrichment	Additional site subject identifier type	Enrichment means changing the (set of) subject identifier(s) assigned to a given target identifier with the purpose of improved matching (e.g. by adding additional identifiers) or for correcting submitted data (e.g. by changing a previously submitted identifier value).
Linked Pseudonym	Target Identifier	PIMS allows managing two types of pseudonyms: *source pseudonyms* act as aliases for a set of submitted data (demographics and identifiers) belonging to the same physical subject. Linked pseudonyms act as aliases for a set of *source pseudonyms* that are found to be corresponding to the same physical subject (outcome of record linkage). *Source pseudonyms* are only used internally in the context of TraIT.
Realm	Target Collection	PIMS supports two types of realms: catalogues and research projects. The former is used in TraIT to implement the “request target identifier” use case. The latter is used in TraIT to implement the “double de-identification” use case.
Research Project	Double de-identified target collection	A target collection owner can define a research project on his collection and resubmit a target identifier to the defined research project. The target identifier (linked pseudonym) obtained for this research project is the double de-identified target identifier.
Subject identifier	Site identifier(s)	A collection of characteristics (including demographics, partial identifiers (e.g. phone numbers) and identifiers) that can be used to identify a given subject. TraIT uses only identifiers.
Source	Submitting site	

7 Security

PIMS (and CATS) Web Services advertise their security requirements in the form of Policy annotations (WS-Policy[13]) annotated to the appropriate WSDL elements (WS-PolicyAttachment[13]).

7.1 Identity Management

In order to be able to use the PIMS and CATS services described in this document, a CIAM user account is required. CIAM users are grouped under organizations which in turn are organized under domains. Within TraIT, all organizations will be registered under the same domain.

Each organization in CIAM will be appointed an administrator user who will be able to manage all users within that organization. Next to this there is a general domain administrator account (managed by Custodix) which is able to perform all administration operations within the CIAM TraIT domain.

7.2 Affiliation, roles, privileges and other attributes

As explained above, each CIAM user is affiliated with an organization. Both submitting sites and target collections must be registered as organizations in CIAM. The CIAM concept of "organization" is used by PIMS and CATS to determine the scope (realm) under which a certain operation is performed. Users affiliated with these respective organizations will be able to perform certain operations based on their role and privileges. Users can only be affiliated with one organization (i.e. two distinct accounts will be needed if the same physical person is acting both as a target collection owner and a submitting site user).

The following roles will be assessed and/or used by PIMS and CATS:

Role	Meaning
ROLE_TECHNICIAN	Target collection owner
ROLE_SOURCE	Submitting site
ROLE_ADMINISTRATOR	Global administrator

The following privileges will be assessed and/or used by PIMS and CATS:

Privilege	Meaning
RIGHT_REIDENTIFICATION	Allows to perform re-identification (go back to the submitted identifiers for a given target identifier).
RIGHT_PROJECTMANAGEMENT	Double de-identified collection management
RIGHT_SOURCEMANAGEMENT	Submitting site management
RIGHT_CONFIGURATIONMANAGEMENT	Target collection configuration management

[13] http://schemas.xmlsoap.org/ws/2004/09/policy/

Typical authorization rules require a combination of affiliation, role and privileges to be present in the user's authentication token. We refer to section 7.5 on how these roles and privileges will be used to authorize PIMS and CATS service requests.

Each CIAM user has a registered email address. This address will be used to send CIAM notifications (e.g. to handle password change requests) and to notify users that a request for additional data is available for download (see section 8.3.6).

7.3 Confidentiality and integrity

PIMS and CATS Web Services support confidentiality and integrity of the exchanged Web Service messages by relying on transport layer security (SSL/TLS) as shown in the following WS-Policy fragment:

```
<sp:TransportBinding>
  <wsp:Policy>
    <sp:TransportToken>
      <wsp:Policy>
        <sp:HttpsToken RequireClientCertificate="false">
          <wsp:Policy/>
        </sp:HttpsToken>
      </wsp:Policy>
    </sp:TransportToken>
    <sp:AlgorithmSuite>
      <wsp:Policy>
        <sp:Basic128/>
      </wsp:Policy>
    </sp:AlgorithmSuite>
    <sp:Layout>
      <wsp:Policy>
        <sp:Lax/>
      </wsp:Policy>
    </sp:Layout>
    <sp:IncludeTimestamp/>
  </wsp:Policy>
</sp:TransportBinding>
```

In principle if supported by the Web Service client run-time the use of the binding does not require any specific configuration (the certificate used by the server is issued by a public CA that is contained by default in most trusted CA stores (e.g. Windows Trust Store, Java default trust store, etc…).

7.4 Authentication

In order to authorize requests PIMS relies on a WS-Security token issued by the CIAM STS which is to be included in the request. This is advertised through the following WS-Policy fragment included in the WSDL:

```
<sp:SupportingTokens>
  <wsp:Policy>
    <sp:IssuedToken sp:IncludeToken="http://docs.oasis-open.org/ws-sx/ws-
securitypolicy/200702/IncludeToken/AlwaysToRecipient">
      <sp:RequestSecurityTokenTemplate>
        <t:TokenType>urn:oasis:names:tc:SAML:2.0:assertion</t:TokenType>
        <t:KeyType>http://docs.oasis-open.org/ws-sx/ws-
trust/200512/Bearer</t:KeyType>
        <t:KeySize>128</t:KeySize>
```

```
      </sp:RequestSecurityTokenTemplate>
      <wsp:Policy>
        <sp:RequireInternalReference/>
      </wsp:Policy>
    </sp:IssuedToken>
  </wsp:Policy>
</sp:SupportingTokens>
```

7.4.1 Web Service client example (Apache CXF using Spring configuration)

This example assumes that stub code was generated, e.g. using CXF's wsdl2java tool[14]. It also assumes that the web service client is instantiated and invoked as follows:

```
PseudonymizationService service = new PseudonymizationService();
PseudonymizationServicePort port = service.getPseudonymizationServicePort();
PseudonymizationResponse response = port.requestPseudonyms(myRequest);
```

This example assumes usage of the JAX-WS specification. Note however that next to using the JAX-WS API, CXF also offers a proprietary API and can deal with both static (compiled) and dynamic (generated at run-time using dynamic proxies) web service stubs. Configuration of web service clients and endpoints can be performed using the Spring framework or can be done directly in code (relying on the CXF API). For more details we refer to the CXF documentation.

Securing the invocation by this web service client is very simple as the client will discover the applicable security policies dynamically. You only need to configure the wsdl location and keystore, truststore and password handling. This can be done quite easily by placing a spring configuration file named `cxf.xml` on the root of your classpath. This file contains a prototype configuration (as stated by `createdFromApi="true" below`) for your web service client:

```
<beans xmlns="http://www.springframework.org/schema/beans"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:jaxws="http://cxf.apache.org/jaxws"
   xmlns:cxf="http://cxf.apache.org/core"
   xmlns:tns="http://ws.custodix.com/"
   xmlns:aop="http://www.springframework.org/schema/aop"
   xsi:schemaLocation="
http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-3.0.xsd
http://cxf.apache.org/core http://cxf.apache.org/schemas/core.xsd
http://cxf.apache.org/jaxws http://cxf.apache.org/schemas/jaxws.xsd">

  <jaxws:client name="{http://www.custodix.com/MPI/1.0}PseudonymizationServicePort"
createdFromAPI="true"
      address="http://localhost:8080/pims/services/PseudonymizationService">
    <jaxws:properties>
      <entry key="ws-security.sts.client">
        <bean class="org.apache.cxf.ws.security.trust.STSClient">
          <constructor-arg ref="cxf"/>
          <property name="wsdlLocation" value="http://localhost:8080/sts/STSService?wsdl"/>
          <property name="serviceName" value="{http://sts.custodix.com/}STSService"/>
          <property name="endpointName" value="{http://sts.custodix.com/}ISTSService_Port"/>
          <property name="properties">
            <map>
              <entry key="ws-security.signature.username" value="myKey"/>
              <entry key="ws-security.callback-handler" value="ClientCallbackHandler"/>
```

[14] See http://cxf.apache.org/docs/wsdl-to-java.html

```
            <entry key="ws-security.signature.properties" value="clientKeystore.properties"/>
            <entry key="ws-security.encryption.properties" value="stsKeystore.properties"/>
            <entry key="ws-security.encryption.username" value="stsKey"/>
          </map>
        </property>
      </bean>
    </entry>
  </jaxws:properties>
</jaxws:client>

</beans>
```

In the above configuration, the name refers to the qualified name of the port being called and address the location of the service to be called (in this example it is the PIMS `PseudonymizationService`). Support for obtaining a security token from the CIAM STS using the WS-Trust specification is configured using the `ws-security.sts.client` property. The bean specified under this property will be used as a client to request a security token from the CIAM STS. As such its wsdlLocation, serviceName and endpointName properties must be set so that they point to the CIAM STS wsdl location, qualified service name and qualified port name respectively.

For this example, we assume that the STS client will authenticate itself to the STS based on the WS-Security X.509 Token profile[15]. This requires that the service/application running this Web Service client has been previously registered and activated in CIAM using an X.509 certificate issued on the dedicated TraIT PKI.

The callback handler is used to obtain the correct password to unlock keys from the specified key stores. A simple implementation example assuming that the password used to protect a given key is equal to the key identifier (alias) is given below:

```
import java.io.IOException;
import javax.security.auth.callback.Callback;
import javax.security.auth.callback.CallbackHandler;
import javax.security.auth.callback.UnsupportedCallbackException;

import org.apache.ws.security.WSPasswordCallback;

public class ClientCallbackHandler implements CallbackHandler {

    public void handle(Callback[] callbacks) throws IOException,
            UnsupportedCallbackException {
        for (int i = 0; i < callbacks.length; i++) {
            if (callbacks[i] instanceof WSPasswordCallback) {
                WSPasswordCallback pc = (WSPasswordCallback) callbacks[i];
                pc.setPassword(pc.getIdentifier());
            }
        }
    }
}
```

The signature properties file specified above is used to configure how the signature key used for signing the STS security token request can be retrieved and unlocked:

```
org.apache.ws.security.crypto.merlin.keystore.type=jks
org.apache.ws.security.crypto.merlin.keystore.password=myKey
org.apache.ws.security.crypto.merlin.keystore.alias=myKey
org.apache.ws.security.crypto.merlin.file=myKeyStore.jks
```

[15] https://www.oasis-open.org/committees/download.php/16785/wss-v1.1-spec-os-x509TokenProfile.pdf

Equivalently, the encryption properties file is used to configure how the encryption key used for encrypting the STS security token request can be retrieved and unlocked:

```
org.apache.ws.security.crypto.merlin.keystore.type=jks
org.apache.ws.security.crypto.merlin.keystore.password=stsKey
org.apache.ws.security.crypto.merlin.keystore.alias=stsKey
org.apache.ws.security.crypto.merlin.file=stsKeyStore.jks
```

When correctly configured, the CXF run-time will try to satisfy the ws-policy requirements as annotated in the WSDL of the Web Service being called (PIMS `PseudonymizationService` in the case of this example).

The following CXF modules are required in order to allow running this example (Maven[16] dependencies extract):

```
<dependency>
        <groupId>org.apache.cxf</groupId>
        <artifactId>cxf-rt-frontend-jaxws</artifactId>
        <version>${cxf.version}</version>
    </dependency>
    <dependency>
        <groupId>org.apache.cxf</groupId>
        <artifactId>cxf-rt-ws-security</artifactId>
        <version>${cxf.version}</version>
    </dependency>
    <dependency>
        <groupId>org.apache.cxf</groupId>
        <artifactId>cxf-rt-ws-policy</artifactId>
        <version>${cxf.version}</version>
    </dependency>
    <dependency>
        <groupId>org.apache.cxf</groupId>
        <artifactId>cxf-rt-ws-mex</artifactId>
        <version>${cxf.version}</version>
    </dependency>
```

For other configuration and usage options, please refer to the CXF documentation available at http://cxf.apache.org/docs/index.html.

7.5 Authorization

PIMS authorizes a request based on the attributes contained in the security token included with the request. The following table shows the access control policies for invoking the various PIMS operations used within TraIT:

Operation[17]	Policy	Remarks

[16] http://maven.apache.org/. Not that the use of Maven for dependency management is not required to run this example.

Operation[17]	Policy	Remarks
PseudonymizationService .requestPseudonyms ([realm], subjects)	Permit if the requesting user is affiliated with an organization that is configured as one of the submitting sites for the target collection identified by the "realm" argument. Deny otherwise.	Realm argument refers to the target Collection (by name) for which a target identifier is requested. The Subjects argument refers to the list of submitting site subject identifiers for which target identifiers are requested.
PseudonymizationService .enrich (pseudonym, [realm], subject)	Permit if the requesting user is affiliated with an organization that is configured as one of the submitting sites for the target collection identified by the "realm" argument. Deny otherwise.	Realm argument refers to the target collection (by name) for which the specified pseudonym (target identifier) was previously issued. The subject argument refers to the additional site subject identifier to be registered.
PseudonymizationService .linkPseudonyms (pseudonym, researchProject)	Permit if the requesting user is affiliated with the target collection (organization) for which the specified pseudonym was issued. Deny otherwise.	The researchProject argument refers to the double de-identified collection (by name) for which the double de-identified target identifier is requested. The pseudonym(s) argument refers to the (list of) target identifier(s) for which (a) double de-identified identifier(s) is/are requested.
RegistrationService .registerResearchProject (researchProject)	Permit if the requesting user is affiliated with the target collection (organization) on which the research project is being defined.	The researchProject argument refers to the double de-identified collection being set up by this call. This information includes double de-identified collection name, description, and reference to the target collection on which it is based and other optional items (see further).

If a request does not meet the applicable security policy requirements, a SOAP fault will be returned specifying that authorization failed.

[17] Optional arguments are placed between square brackets.

8 Pseudonymisation service operations

8.1 Correspondence to TraIT TTP RFP Use Cases

The following table specifies how to interact with the TTP services for each of the relevant use cases as specified in the TraIT TTP RFP document.

RFP Section	TTP Operation[18]	Remarks
3.1.1 First data submission, step 1: requesting a Target Subject Identifier	`PseudonymizationService .requestPseudonyms`	Also supports requesting target subject identifiers under 3.1.2 Additional data submissions
3.1.1 First data submission, step 2: requesting a Target Data Object Identifier	`PseudonymizationService .requestDataObjectPseudonyms`	Also supports requesting target data object identifiers under 3.1.2 Additional data submissions
3.1.3 Additional Site Subject Identifier Type	`PseudonymizationService .enrich`	
3.2 Query & Retrieval	No direct interaction with PIMS required	The target identifiers for a public (double de-identified) collection will have been obtained during a previous interaction with PIMS
3.3 Overview of data objects for a given subject	No PIMS web service counterpart.	Only available through the service desk.
3.4 Overview of existing target identifiers	`PseudonymizationService .requestOverview`	
3.5 Double de-identification	`PseudonymizationService .linkPseudonyms`	The double de-identified collection must be set up first using `RegistrationService .registerResearchProject`.
3.6 Request for additional data to a submitting site for a specific subject	No direct interaction with PIMS required. CATS operation to send the request: `UploadService.upload` CATS operation to retrieve the re-identified request: `MessageSOAPWebServiceService .getMessage`	The required PIMS functionality to support this use case is called by CATS.
3.7 Request for additional data to a non-submitting site for a specific subject	Same as above	Same as above

[18] PIMS Web Service operation (unless indicated otherwise).

RFP Section	TTP Operation[18]	Remarks
3.8 Conversion of a collection	`PseudonymizationService .requestPseudonyms` `PseudonymizationService .requestDataObjectPseudonyms` `PseudonymizationService .linkPseudonyms`	Can be achieved by: • Resubmitting the subject and data object identifiers to a newly defined collection. • Conversion to a double de-identification is handled by `linkPseudonyms` (see UC 3.5)
3.9 Request for pseudonymization service for new study with one or more submitting sites	`RegistrationService .registerCatalogue` `RegistrationService .registerResearchProject`	Configuring a new target collection is done using `registerCatalogue`. Configuring a new double de-identified collection is done using `registerResearchProject`.
3.10 Request for removal of entire study or data from specific sites	`PseudonymizationService .requestRemoval`	

8.2 General concepts

8.2.1 Web Service invocation

All PIMS and CATS services are exposed as SOAP-based Web Services. Each successful PIMS response contains a unique UUID that can be used as a reference. Upon encountering an error condition, PIMS and CATS will return a SOAP fault. If the error concerned is an unexpected error, the SOAP fault will include a unique ticket number that can be used as a reference for troubleshooting (when contacting the service desk).

All Web Services are secured using WS-Security and require interaction with the CIAM STS according to the WS-Trust protocol.

8.2.2 Validation of submitted identifier(s)

In order to allow target collection owners to specify what kind of identifiers can be used to obtain target identifiers, target collections can be configured with a subject and data object validator. E.g. a target collection may only accept the *Burger Service Nummer* (BSN) as a valid submitting site subject identifier to obtain a target identifier.

8.2.2.1 Occurrence validator

This validator can be used to verify whether an expected number of values is provided for a given subject attribute.

E.g. A target identifier must be requested based on exactly one *Burger Service Number* (BSN) and zero or more HIS numbers, this can be specified by the following definition:

```
nationalIdentificationNumber 1
patientNumber *
```

See also 8.3.1.1 Mapping Site subject identifiers to the PIMS subject model.

8.2.3 Target identifier generators

It is possible to configure the scheme to be used for generating target identifiers (pseudonyms in the PIMS terminology). This is done by selecting and configuring a pseudonym generator. A pseudonym generator is identified by its type (called class) and configured by a set of name/value pairs (called properties). The following subsections describe the pseudonym generators available in PIMS and how these can be configured.

8.2.3.1 Type 4 (pseudo-random generated) UUID

This is the default pseudonym generator used by PIMS when no pseudonym generator has been specified by the user. This generator issues type 4 (pseudo-random generated) UUIDs as specified in ISO/IEC 11578:1996.

8.2.3.2 Incremental identifier

This pseudonym generator simply increments the previously issued target identifier (pseudonym). An optional property `prefix` can be set to specify the string with which the issued target identifier should start. An incremental integer number starting from 1 will be appended to this prefix. If the `prefix` property is omitted, the generated target identifiers will simply be integer values (starting from 1).

8.2.3.3 HMAC-based (irreversible)

This pseudonym generator calculates the keyed-hash from a specified subject identifier using a specified key in hexadecimal format.

Property	Value
key	A 160 bit key in hexadecimal format (40 characters)
Identifier	XPath expression specifying the subject attribute (in the PIMS subject model) to be used for the HMAC calculation, e.g. `nationalIdentificationNumber`

The output is a hexadecimal string (40 characters).

8.2.3.4 AES-based (reversible)

This pseudonym generator encrypts a given subject identifier using a specified AES key in hexadecimal format.

Property	Value
key	A 128 bit key in hexadecimal format (32 characters)
Identifier	XPath expression specyfing the subject attribute (in the PIMS subject model) to be used for the HMAC calculation, e.g. `nationalIdentificationNumber`

The output is a hexadecimal string (32 characters).

8.2.3.5 Scriptable generator

This generator uses the Groovy[19] script specified by the `script` property to calculate the target identifier. The script will be able to access the submitted identifier(s) through the `subject` variable and must yield a string value as a result.

Example:

```
import java.security.MessageDigest

def id = subject.nationalIdentificationNumbers[0]
def type = id.domain == null ? "BSN" : id.domain.trim()
def value = type + "_" + id.value.replaceAll("[^0-9]", "")
def digest = MessageDigest.getInstance("SHA1")
digest.update(value.getBytes())
return new BigInteger(1,digest.digest()).toString(16).padLeft( 40,'0')
```

This script will yield the target identifier `2d1369817c021ed4f9f72a6fa85182e6fac4b76f` for a submitted identifier of type `nationalIdentificationNumber` with value "`9413.31.490 `" (see also 8.3.1.1 Mapping Site subject identifiers to the PIMS subject model).

8.3 Web Service interaction details

8.3.1 Requesting a Target Subject Identifier

In general PIMS allows requesting pseudonyms (target identifiers) based on a combination of subject identifiers (name, patient number, etc.…) and demographics (gender, address, ethnic origin, etc…). In TraIT pseudonyms will be requested based on identifiers only. Therefore most of the elements and attributes of the PIMS subject element[20] will not be used.

Although the interface for requesting pseudonyms (target identifiers) is generic, target collection owners will be able to place constraints on the information that is passed in the PIMS subject included in a pseudonymisation request. These constraints are enforced in a validation step performed at the start of the request handling. If validation fails, the requester will be notified of this by receiving a SOAP Fault in response to the request. The SOAP fault will provide details on which subject elements or attributes are in violation of the target collection's site subject constraints.

Possible constraints:

- A given PIMS subject property must not occur, must occur exactly one time or at least one time.
    - E.g. a national identification number may be provided and exactly one patient number must be provided. All other PIMS subject properties must not occur.

[19] See http://groovy.codehaus.org/

[20] The PIMS subject element is specified in the schema section of the PseudonymizationService WSDL file. Section 0

Environments specifies where the WSDL for the PseudonymizationService can be found.

- A given PIMS subject identifier must have a domain value (see further) occurring in a preconfigured list of allowed values.
    - E.g. a patient number must have its domain set to one of the following values: MAASTRA_MRN, IT_CodiceSanitario or MUMC_Animal_ID.

8.3.1.1 Mapping Site subject identifiers to the PIMS subject model

The following table shows which properties from the PIMS subject model can be used to pass site subject identifiers in target identifier requests:

PIMS Identifier	TraIT site subject identifier examples mentioned in the RFP	Remarks
insuranceNumber	N/A	
nationalIdentificationNumber	NL_BSN	
patientNumber	MAASTRO_MRN, IT_CodiceSanitario, MUMC_Animal_ID, TraIT_Study_ID, TraIT_Intermediate_ID, TraIT_Public_ID	

The above PIMS identifiers are strings and have the following (optional) attributes:

- **domain**: type identifier that specifies the scope in which it is issued . This attribute corresponds to the TraIT "site subject identifier type". Examples from the RFP are shown in the above table.
- **timestamp**: date and time at which the identifier was recorded or assigned to the given subject at the submitting site's information system.

8.3.1.2 Example

Requesting a Target Identifier based on the subject's BSN (SOAP Header content omitted for reasons of brevity):

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:requestPseudonyms>
         <ns:realm>MyTargetCollection</ns:realm>
         <ns:subject>
            <ns:nationalIdentificationNumber
ns:domain="NL_BSN">941331490</ns:nationalIdentificationNumber>
         </ns:subject>
      </ns:requestPseudonyms>
   </soapenv:Body>
</soapenv:Envelope>
```

The response will contain the issued target identifier:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:requestPseudonymsResponse ns:responseId="cc7bbd5a-c878-4294-
ac66-22b61e5207c5">
         <ns:linkedPseudonym>10239841.342.2.11234</ns:linkedPseudonym>
      </ns:requestPseudonymsResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

The issued target identifier is the string specified between the `linkedPseudonym` tags. If a single request contains multiple subjects, the response will contain the same number of linked pseudonyms and in the same order used in the request i.e. the first linked pseudonym returned corresponds to the first subject in the request, the second linked pseudonym returned corresponds to the second subject in the request, etc…).

8.3.2 Requesting a Target Data Object Identifier

8.3.2.1 Mapping Site data object identifiers to the PIMS data object model

The following table shows which properties from the PIMS data model can be used to pass site subject identifiers in target identifier requests:

PIMS Identifier (XPath)	TraIT site subject identifier examples mentioned in the RFP	Remarks
dataObject/ @type	DICOM file Metadata File (e.g. XML)	PIMS does not interpret this attribute, but merely keeps it as an informative reference.
dataObject/ subjectReference	AIRFORCE123 DEC_Subject1 10239841.342.2.123483 10239841.342.2.11234 10239841.342.2.34255	PIMS will look up the matching site subject by using this value and the target collection reference (by name) that is passed as part of the target data object identifier request.
dataObject/ identifier	2.16.840.1.114337.5848532277 3.1378.1142522548.134 UMCG_2012_145	An identifier used at the site to identify the data object, e.g. a DICOM OID or tissue sample id.
dataObject/ identifier/ @domain	MAASTRO_DICOM StudyInstanceUID UMCG_Sample ID	The `domain` attribute of the `identifier` element is used to denote the administrative domain governing the identifier.

8.3.2.2 Example

Requesting a Target Data Object Identifier based on the subject's DICOM file identifier (SOAP Header content omitted for reasons of brevity):

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:requestDataObjectPseudonyms>
         <ns:realm>MyTargetCollection</ns:realm>
         <ns:dataObject type="DICOM file">
           <ns:subjectReference>
             <ns:linkedPseudonym>10239841.342.2.11234</ns:linkedPseudonym>
           </ns:subjectReference>
           <ns:identifier ns:domain="MAASTRO_DICOM">
            2.16.840.1.114337.58485322773.1378.1142522548.134
           </ns:identifier>
          </ns:dataObject>
      </ns:requestDataObjectPseudonyms>
   </soapenv:Body>
</soapenv:Envelope>
```

The realm element contains the reference (by name) to the target collection. Each data element passed in the request (one in this example) includes a reference to the subject it belongs to (by passing its target identifier for the given target collection as previously obtained in the *Example* for *General concepts*

Web Service invocation

All PIMS and CATS services are exposed as SOAP-based Web Services. Each successful PIMS response contains a unique UUID that can be used as a reference. Upon encountering an error condition, PIMS and CATS will return a SOAP fault. If the error concerned is an unexpected error, the SOAP fault will include a unique ticket number that can be used as a reference for troubleshooting (when contacting the service desk).

All Web Services are secured using WS-Security and require interaction with the CIAM STS according to the WS-Trust protocol.

8.3.3 Validation of submitted identifier(s)

In order to allow target collection owners to specify what kind of identifiers can be used to obtain target identifiers, target collections can be configured with a subject and data object validator. E.g. a target collection may only accept the *Burger Service Nummer* (BSN) as a valid submitting site subject identifier to obtain a target identifier.

8.3.3.1 Occurrence validator

This validator can be used to verify whether an expected number of values is provided for a given subject attribute.

E.g. A target identifier must be requested based on exactly one *Burger Service Number* (BSN) and zero or more HIS numbers, this can be specified by the following definition:

```
nationalIdentificationNumber 1
patientNumber *
```

See also 8.3.1.1 Mapping Site subject identifiers to the PIMS subject model.

8.3.4 Target identifier generators

It is possible to configure the scheme to be used for generating target identifiers (pseudonyms in the PIMS terminology). This is done by selecting and configuring a pseudonym generator. A pseudonym generator is identified by its type (called class) and configured by a set of name/value pairs (called properties). The following subsections describe the pseudonym generators available in PIMS and how these can be configured.

8.3.4.1 Type 4 (pseudo-random generated) UUID

This is the default pseudonym generator used by PIMS when no pseudonym generator has been specified by the user. This generator issues type 4 (pseudo-random generated) UUIDs as specified in ISO/IEC 11578:1996.

8.3.4.2 Incremental identifier

This pseudonym generator simply increments the previously issued target identifier (pseudonym). An optional property `prefix` can be set to specify the string with which the issued target identifier should start. An incremental integer number starting from 1 will be appended to this prefix. If the `prefix` property is omitted, the generated target identifiers will simply be integer values (starting from 1).

8.3.4.3 HMAC-based (irreversible)

This pseudonym generator calculates the keyed-hash from a specified subject identifier using a specified key in hexadecimal format.

Property	Value
key	A 160 bit key in hexadecimal format (40 characters)
Identifier	XPath expression specifying the subject attribute (in the PIMS subject model) to be used for the HMAC calculation, e.g. `nationalIdentificationNumber`

The output is a hexadecimal string (40 characters).

8.3.4.4 AES-based (reversible)

This pseudonym generator encrypts a given subject identifier using a specified AES key in hexadecimal format.

Property	Value
key	A 128 bit key in hexadecimal format (32 characters)
Identifier	XPath expression specyfing the subject attribute (in the PIMS subject model) to be used for the HMAC calculation, e.g. `nationalIdentificationNumber`

The output is a hexadecimal string (32 characters).

8.3.4.5 Scriptable generator

This generator uses the Groovy script specified by the `script` property to calculate the target identifier. The script will be able to access the submitted identifier(s) through the `subject` variable and must yield a string value as a result.

Example:

```
import java.security.MessageDigest

def id = subject.nationalIdentificationNumbers[0]
def type = id.domain == null ? "BSN" : id.domain.trim()
def value = type + "_" + id.value.replaceAll("[^0-9]", "")
def digest = MessageDigest.getInstance("SHA1")
digest.update(value.getBytes())
return new BigInteger(1,digest.digest()).toString(16).padLeft( 40,'0')
```

This script will yield the target identifier `2d1369817c021ed4f9f72a6fa85182e6fac4b76f` for a submitted identifier of type `nationalIdentificationNumber` with value "`9413.31.490 `" (see also 8.3.1.1 Mapping Site subject identifiers to the PIMS subject model).

8.4 Web Service interaction details

Requesting a Target Subject Identifier).

The response will contain the issued target data object identifier:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:requestDataObjectPseudonymsResponse ns:responseId="699d98de-
3dff-47b0-95c8-de8581824165">
         <ns:linkedPseudonym>
           b35a991f-8cb3-4f87-ada9-4c1c1587b668
         </ns:linkedPseudonym>
      </ns: requestDataObjectPseudonymsResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

The issued target data object identifier is the string specified between the `linkedPseudonym` tags. If a single request contains multiple data objects, the response will contain the same number of linked pseudonyms and in the same order used in the request i.e. the first linked pseudonym returned corresponds to the first data object in the request, the second linked pseudonym returned corresponds to the second data object in the request, etc…).

8.4.1 Additional Site Subject identifier type

Sometimes a submitting site may wish to record additional identifiers for a target identifier that was previously issued on another site subject identifier (type or value).

PIMS includes a feature called "enrich pseudonym" which allows submitting sites to update the information on the basis of which a pseudonym was previously issued. This feature supports two update modes: "add" for adding a new (set of) subject identifiers to the previously submitted (set of)

subject identifiers and “replace” for replacing the (set of) previously submitted subject identifiers with the newly provided (set of) subject identifiers.

8.4.1.1 Example

Enriching a Target Identifier with the subject’s BSN (SOAP Header content omitted for reasons of brevity):

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:enrich action="add">
         <ns:realm>MyTargetCollection</ns:realm>
         <ns:pseudonym>10239841.342.2.11234</ns:pseudonym>
         <ns:subject>
            <ns:nationalIdentificationNumber
ns:domain="NL_BSN">941331490</ns:nationalIdentificationNumber>
         </ns:subject>
      </ns:enrich>
   </soapenv:Body>
</soapenv:Envelope>
```

The response:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:enrichResponse
                    ns:responseId="cc7bbd5a-c878-4294-ac66-22b61e5207c5"/>
   </soapenv:Body>
</soapenv:Envelope>
```

8.4.2 Overview of existing target identifiers

This Web Service allows to retrieve an overview of all issued target identifiers (both for subject and data objects) for a given subject or data object identifier.

8.4.2.1 Example

The following example illustrates how to request an overview of issued target identifiers for a given subject (with BSN “941331490”) within a given target collection (“MyTargetCollection”):

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns: requestOverview>
         <ns:realm>MyTargetCollection</ns:realm>
         <ns:subject>
            <ns:nationalIdentificationNumber
ns:domain="NL_BSN">941331490</ns:nationalIdentificationNumber>
         </ns:subject>
      </ns: requestOverview >
   </soapenv:Body>
</soapenv:Envelope>
```

The response shows the target identifier under which this subject has been registered in the target collection and the target identifiers of the data objects belonging to that subject that have been registered in the target collection:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:requestOverviewResponse
                    ns:responseId="fc2d8927-c14f-4b17-84a9-5245424b7d1f">
        <ns:subject>
          <ns:linkedPseudonym>
            10239841.342.2.11234
          </ns:linkedPseudonym>
        </ns:subject>
        <ns:dataObject>
          <ns:linkedPseudonym>
            1.2.826.0.1.3680043.8.1084.10.1234
          </ns:linkedPseudonym>
        </ns:dataObject >
        <ns:dataObject >
          <ns:linkedPseudonym>
            1.2.826.0.1.3680043.8.1084.10.1235
          </ns:linkedPseudonym>
        </ns:dataObject >
      </ns:requestOverviewResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

8.4.3 Double de-identification

The concept of double de-identification is supported through the PIMS “research project” mechanism: it is possible to define a “research project” on an existing catalogue (target collection). This allows an authorized user (the target collection owner) to request a new linked pseudonym (double de-identified target identifier) based on an existing linked pseudonym (target identifier) that was previously issued for the given target collection.

The research project must be set up first using the `RegistrationService.registerResearchProject` operation as described in section 8.4.5.2.

8.4.3.1 Example

The following example shows how the target identifier issued in Example for a single de-identified collection can be resubmitted in order to obtain a target identifier for a double de-identified collection. Only users who are affiliated with the target collection are authorized to perform a request to obtain a double de-identified target identifier (SOAP Header content omitted for reasons of brevity):

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:linkPseudonyms>
         <ns:pseudonym realm="MyTargetCollection">
           10239841.342.2.11234
         </ns:pseudonym>
         <ns:researchProject>
           MyDoubleDeidentifiedTargetCollection
         </ns:researchProject>
      </ns:linkPseudonyms>
   </soapenv:Body>
</soapenv:Envelope>
```

The response will contain the issued target identifier for the double de-identified collection:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:requestPseudonymsResponse ns:responseId="cc7bbd5a-c878-4294-
ac66-22b61e5207c5">
         <ns:linkedPseudonym>MY_DOUBLE_DEID_123</ns:linkedPseudonym>
      </ns:requestPseudonymsResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

8.4.4 Request for additional data to a (non-)submitting site for a specific subject

This use case is supported as follows:

- The target collection owner submits an XML request to CATS (either using the GUI web interface of the Web Service interface). The subject (patient) is identified in this request by means of the target identifier. The target collection name and target identifier are included in the XML request.
- CATS inspects the XML request and uses the target identifier to issue a re-identification request to PIMS.
- PIMS returns the site identifiers matching the target identifier grouped per submitting site.

- CATS splits up the processed request in multiple copies containing the site identifiers originating from the same submitting site.
- CATS looks up the responsible user for dealing with additional data requests for each submitting site (via CIAM) and makes the processed request available for download. The responsible user for dealing with additional data requests for a given submitting site is identified in CIAM by a dedicated role.
- CATS sends an email notification to each responsible user stating that pending requests are available (a link is included which allows the user to access a page on the CATS web interface displaying the list of all pending requests available for the user).
- The responsible user(s) download(s) and review(s) the request(s) from CATS.

Requirements:

- The requesting user (e.g. target collection owner) must be authorized to perform re-identification requests. This is achieved by assigning the necessary privilege in CIAM to this user (by the organization administrator) and by enabling CATS to perform re-identification requests on behalf of that user.
- The receiving user responsible for dealing with additional data requests for a given submitting site must be configured in CIAM by assigning the appropriate role (to be specified at the time of writing) (done by the organization administrator).
- The request content is valid (well-formed) XML starting with the <?xml> directive.
- It contains an xml element containing the subject's target collection identifier as text or CDATA contents and an attribute containing the target collection name as registered in PIMS (TargetIdentifier element in the example below).

CATS will transform this request so that it contains the subject identifier(s) corresponding to the specified target identifier as originally sent by each of the submitting sites. The request is made available for download to each of the submitting sites.

8.4.4.1 Example

```
<AdditionalDataRequest xmlns="urn:ctmm:trait:schemas:ttp:1.0">
  <Subject>
    <TargetIdentifier targetCollection="MyTargetCollection”>
10239841.342.2.11234</TargetIdentifier>
  </Subject>
  <!-- other relevant information to specify what kind of additional data
is requested. Not modified by CATS. -->
</AdditionalDataRequest>
```

After processing by CATS, the request destined for a single submitting site will look like this:

```
<AdditionalDataRequest xmlns="urn:ctmm:trait:schemas:ttp:1.0">
  <Subject>
    <ns:subject xmlns:ns="http://www.custodix.com/MPI/1.0">
      <ns:nationalIdentificationNumber ns:domain="NL_BSN">
        941331490
      </ns:nationalIdentificationNumber>
    </ns:subject>
  </Subject>
  <!-- other relevant information to specify what kind of additional data
is requested. Not modified by CATS. -->
</AdditionalDataRequest>
```

For visualization purposes, it is advisable that the request be provided with an XSL stylesheet reference so that it can be presented in an appealing and user-friendly way in the receiving user's browser.

8.4.5 Request for pseudonymization service for new study with one or more submitting sites

Depending on the type of target collection, a new target collection is defined by invoking either the `registerCatalogue` (for normal target collections) or the `registerResearchProject` (for double de-identified target collections) operation of the `RegistrationService` service.

The newly created target collection will be under the control of the organization with whom the requesting user is affiliated: only users from that organization with the correct role and privileges will be able to perform management operations on the target collection.

8.4.5.1 Defining a new target collection

The TraIT concept of *target collection* is equivalent to the PIMS *catalogue* concept. A single PIMS instance can serve multiple catalogues. It is thus not necessary to set up a new instance for each target collection.

A catalogue is defined using the `RegistrationService.registerCatalogue` operation:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:registerCatalogue>
         <ns:catalogue name="CTMM_AIRFORCE">
            <ns:description>Optional description of the de-identified
collection</ns:description>
            <ns:pseudonymGenerator
class="com.custodix.pims.generators.IncrementalPseudonymGenerator">
               <ns:property name="prefix">AIRFORCE</ns:property>
            </ns:pseudonymGenerator>
         </ns:catalogue>
      </ns:registerCatalogue>
   </soapenv:Body>
</soapenv:Envelope>
```

In the above request, only the `name` attribute is required and it must be unique (not already in use). If no pseudonym generator is specified, the default implementation will be used (type 4 pseudo randomly generated UUID[21]). For an overview of all available pseudonym generators and their configuration, please refer to section 8.2.

If all went well, the response will be similar to the following:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:registerCatalogueResponse
                    ns:responseId="40fa799c-c2df-4ab6-abcf-73eae6e9bbdf"/>
   </soapenv:Body>
</soapenv:Envelope>
```

8.4.5.2 Defining a new double de-identified target collection

The TraIT concept of *double de-identified target collection* is equivalent to the PIMS *research project* concept. A research project is defined on an existing catalogue (target collection) and allows issuing target identifiers that are distinct from the target identifiers issued for the underlying target collection.

Only users affiliated with the organization "owning" the target collection that have the correct role and privileges will be allowed to define a research project (double de-identified collection) on the given catalogue (target collection).

A research project is defined using the `RegistrationService.registerResearchProject` operation:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:registerResearchProject>
         <ns:researchProject name="MyDoubleDeidentifiedTargetCollection">
            <ns:description>Optional description of the double de-
identified collection</ns:description>
            <ns:pseudonymGenerator
class="com.custodix.pims.generators.IncrementalPseudonymGenerator">
              <ns:property name="prefix">MY_DOUBLE_DEID_</ns:property>
            </ns:pseudonymGenerator>
         </ns:researchProject>
      </ns:registerResearchProject >
   </soapenv:Body>
</soapenv:Envelope>
```

[21] See ISO/IEC 11578:1996

In the above request, only the `name` attribute is required and it must be unique (not already in use). If no pseudonym generator is specified, the default implementation will be used (type 4 pseudo randomly generated UUID[22]). For an overview of all available pseudonym generators and their configuration, please refer to section 8.2.

If not explicitly specified, the catalogue (target collection) on which the research project (double de-identified collection) is set to be the (unique) catalogue with whom the requesting user is affiliated.

If all went well, the response will be similar to the following:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:registerResearchProjectResponse
                    ns:responseId="38378dc6-4abe-4603-896f-e32c8e41bb70"/>
   </soapenv:Body>
</soapenv:Envelope>
```

From this moment on, target identifiers for the newly created research project (double de-identified collection) can be requested.

8.4.6 Request for removal of entire study or data from specific sites

PIMS supports removal of issued target subject and data object identifiers for a given target collection on the following levels:

- The entire target collection.
- All target identifiers issued based on a site subject or data object originating from a given submitting site.
- All target (subject and data object) identifiers issued for a particular subject.

In the first case, all issued target subject and data object identifiers will be removed for a given target collection. In the second case, all issued target subject and data object identifiers based on data originating from the given submitting site will be deleted. In the third case, only the specified target subject identifier and all its assigned data object identifiers will be deleted.

8.4.6.1 Example: removal of an entire target collection

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:requestRemoval>
         <ns:realm>MyTargetCollection</ns:realm>
      </ns:requestRemoval>
   </soapenv:Body>
</soapenv:Envelope>
```

[22] See ISO/IEC 11578:1996

If all went well, the corresponding response looks as follows:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:requestRemovalResponse
                    ns:responseId=" a4a20284-f76b-41f5-b7bc-d35ef72aa712"/>
   </soapenv:Body>
</soapenv:Envelope>
```

8.4.6.2 Example: removal of data originating from a submitting site

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:requestRemoval>
         <ns:realm>MyTargetCollection</ns:realm>
         <ns:source>MySubmittingSite</ns:source>
      </ns:requestRemoval>
   </soapenv:Body>
</soapenv:Envelope>
```

If all went well, the corresponding response looks as follows:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:requestRemovalResponse
                    ns:responseId="514b1f38-4fc2-40d7-85b8-dd8f08e84090"/>
   </soapenv:Body>
</soapenv:Envelope>
```

8.4.6.3 Example: removal of data related to a given subject

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:requestRemoval>
         <ns:realm>MyTargetCollection</ns:realm>
         <ns:linkedPseudonym>10239841.342.2.11234</ns:linkedPseudonym>
      </ns:requestRemoval>
   </soapenv:Body>
</soapenv:Envelope>
```

If all went well, the corresponding response looks as follows:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.custodix.com/MPI/1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <ns:requestRemovalResponse
                  ns:responseId="885d1a94-c21c-4739-adb8-91689e06dea1"/>
   </soapenv:Body>
</soapenv:Envelope>
```

9 More information

For questions and remarks related to this document, please contact support@custodix.com and mention “TraIT technical specifications” as a reference.