

**Interface concept for the
TMF Patient List Server / PID Generator**

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

Revision History

Date	Version	Description	Author
2005-01-04	0.1	Initial Version	Andrew Fowler
2005-01-13	0.2	Revised Version	Philippe Verplancke
2005-01-19	0.3	Revised Version	Andrew Fowler
2005-01-20	0.4	Approved Version, including Mr. Debold's variable list	Philippe Verplancke

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

Table of Contents

1. Introduction.....	4
2. Terminology.....	4
3. General interface concept.....	5
3.1 Underlying protocols and specifications.....	5
3.2 TMF Server based model:.....	6
3.3 CDISC based model:.....	6
3.3.1 Mapping the CDISC-ODM model to the current TMF Server.....	6
3.3.2 CDISC Field/Attribute usage.....	8
3.4 Model comparison.....	9
3.4.1 So which is the 'best' model ?.....	9
4. Commands.....	10
4.1 Get subject data (CDISC model).....	11
4.1.1 Request.....	11
4.1.2 Response – Data retrieved.....	11
4.1.3 Response – Invalid SubjectID.....	12
4.1.4 Response – Unknown StudyOID/MDV OID (Fault).....	12
4.2 Get (and Create) PID.....	13
4.2.1 Request.....	13
4.2.2 Response – No Match.....	13
4.2.3 Response – Single match (Retrieve).....	14
4.2.4 Response – Multiple match (Retrieve).....	14
4.2.5 Response – Created.....	15
4.3 Is Subject ID Valid.....	16
4.3.1 Request.....	16
4.3.2 Response – valid.....	16
4.3.3 Response – invalid.....	16
5. Security.....	17
6. WSDL.....	17
7. Appendix.....	18
7.1 Get subject data (TMF Model).....	18
7.1.1 Request.....	18
7.1.2 Response.....	18

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

1. Introduction

TMF has developed the Patient List server / PID generator for handling patient identifying information and PIDs. In order to give access to this system to external client and server software, a new interface is planned. The goal of this document is to describe a concept for such an interface.

This document does not represent the final status of the discussion between all parties involved nor can it be a complete specification. Certain points remain to be clarified in a later version.

2. Terminology

TMF Client – Client or external server application wishing to access the services provided by the TMF's Patient List server/PID generator

TMF Server – TMF's Patient List Server/PID generator pair

CDISC-ODM – Clinical Data Interchange Consortium-Operational Data Model

Important Note:

Since the interface message payload is based on CDISC-ODM, the latter's terminology will also be used in this document and the prototype interface examples.

For example, the CDISC terminology speaks of "Study Definition". In the context of this document, this is never referring to the definition of medical content of a clinical trial, but merely to the metadata describing which patient identifying variables are transferred to/from the PID Server through the CDISC protocol.

*Also according to CDISC terminology, **Patients** are called **Subjects** and **PIDs** are called **SubjectIDs**. This naming is also used throughout the interface's XML elements and leads to a consistent naming within the interface.*

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

3. General interface concept

3.1 Underlying protocols and specifications

The following short list of requirements is placed on the interfacing method:

- a) The interaction between TMF clients and server is to be performed over the internet. Furthermore TMF clients will typically be in secured LANs with limited internet access. The underlying protocol should therefore preferably lend itself to being used with the standard HTTP port as destination.
- b) Since the interface will be used over the internet, it is imperative that a secure means of communication (authentication and encryption) be available.
- c) The interface must support a request-response style of service since this is the primary service type available from the TMF Server.
- d) The payload of messages between TMF clients and server should preferably make use of existing standards common in the field of medical trials (CDISC, HL7vXML ADT, HL7V3). As a result, the underlying protocol will need to carry these 'complex' message structures and responses.
- e) The interface should be platform and programming language neutral since no assumptions can be made about the TMF Client's environment.

SOAP has been chosen as messaging format for the interface as it satisfies all of the above requirements:

- 1) SOAP defines a binding to the HTTP protocol allowing it to be transported over existing HTTP mechanisms - (a) and (c). The HTTP protocol can be secured via well know means (HTTPS/SSL), thus providing (b).
- 2) Since SOAP messages are in XML format, payloads such as CDISC documents can be naturally embedded into SOAP messages without the need to define new encoding mechanisms - (d).
- 3) SOAP makes use of the XML specification for the message format. SOAP and XML libraries are available today and in wide use on many different platforms and in many different programming languages – (e)

Two possibilities are considered for formatting of message payload: a TMF Server derived model; and a CDISC based model. The message format is described in more detail in chapter 4 (Commands).

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

3.2 TMF Server based model:

In this model, the message payload format is derived directly from the TMF Server's capability i.e. the required input/output parameters. The result is a relatively flat structure where data can be mapped directly to input/output fields of the TMF Server. It is left to the client to do mapping from it's own internal system to the TMF service fields.

3.3 CDISC based model:

In this model, CDISC-ODM is used as a basis for formatting of subject data in the interface message payloads. CDISC-ODM lends itself well to the transfer of patient data thus providing an extensible mechanism by which to add patient data fields as the system grows over time.

In combination, SOAP and CDISC-ODM can provide a large part of the protocol and message payload specification required for achieving the required interface. In order to make use of such model, the following three problems must be overcome:

- 1) The CDISC-ODM data model must be mapped to TMF server input/output data

The use of object identifiers (StudyOID, SubjectKey, etc.) used in the CDISC-ODM model must be clarified within the context of SOAP messages.

- 2) Remaining elements must be defined in order to carry request/response parameters which are not part of the CDISC-ODM data model (e.g. sureness, force PID generation, subject PID).

A possible solution to the above problems is suggested in the following sub-sections.

3.3.1 Mapping the CDISC-ODM model to the current TMF Server

In short, the CDISC-ODM defines an XML based document format for the interchange of Clinical Data. A CDISC-ODM document can contain, amongst other things:

- 1) study definition elements: elements which carry the definition of data items recorded in a clinical trial.
- 2) clinical data elements: elements which carry the recorded clinical data (Clinical data in CDISC-ODM terminology includes both patient identifying data (IDAT) and medical data (MDAT) in TMF terms).
- 3) administrative elements: e.g. investigator and site information.

For the purpose of the interface being considered in this document, only clinical data elements and administrative elements are of interest and want to be exchanged. However, since the structure of clinical data elements generated by the TMF Server must be in accordance with the study definition in use, the server must be aware of this study definition. Furthermore, the server must know which CDISC ItemData fields correspond to which TMF Server fields.

Two alternatives are suggested to the last two requirements:

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

- 1) Restrict TMF Server to a single study definition. All clients must be able to understand and submit data in the specified study definition. This enforces strong coordination between studies. It may also increase the complexity for a study which is already underway to join the TMF service at a later date since the latter may already be based on an "incompatible" study definition.
- 2) Study definition/mapping information is registered with the TMF Server service before the latter can be used with a given study. Allows each client to retrieve and submit data in the study definition which it uses. Studies gain independence from one another in terms of the study definition but at the cost of greater complexity on the server side since the latter must be able to store multiple study definitions and generate/interpret data accordingly. The coordination overhead involved in the study registration process could further be reduced if the 'registration' process can also be provided on-line.

In both cases, the study definition allows the TMF service to generate clinical data in a form and structure expected by the client (fixed in case 1); and map clinical data items to the TMF's own internal fields.

Note: In implementing this TMF interface, the forces that be may well wish to start with the 'easy' solution (1) and then progress onto (2) as and when required. This progression could be made transparent to existing clients.

In the remainder of this document is based on the following CDISC study definition snippet, i.e. the metadata structure describing the patient identifying variables, upon which all ODM ClinicalData elements are based. It shows a means by which registration of a study definition could be used to configure a CDISC-to-TMF field mapping via the <tmf:Field> elements inside the CDISC ItemDef elements. The variable names are taken from the list prepared by Mr. Debold for KN AHF.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<ODM xmlns="http://www.cdisc.org/ns/odm/v1.2"
      xmlns:tmf="http://www.tmf.org/services/types"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Study OID="Study.TMF">
    <GlobalVariables>[...]</GlobalVariables>
    <MetaDataVersion OID="Version.001" Name="Version 001">
      <Protocol>
        <StudyEventRef StudyEventOID="E.01" Mandatory="Yes" OrderNumber="1"/>
      </Protocol>
      <StudyEventDef OID="E.01" Name="Base data" Repeating="No" Type="Unscheduled">
        <FormRef FormOID="F.01" Mandatory="Yes" OrderNumber="1"/>
      </StudyEventDef>
      <FormDef OID="F.01" Name="Subject data" Repeating="No">
        <ItemGroupRef ItemGroupOID="IG.01" Mandatory="Yes" OrderNumber="1"/>
      </FormDef>
      <ItemGroupDef OID="IG.01" Name="Subject data" Repeating="No" IsReferenceData="No">
        <ItemRef ItemOID="Krankenkassennummer" Mandatory="No" OrderNumber="1"/>
        <ItemRef ItemOID="Versichertenummer" Mandatory="No" OrderNumber="2"/>
        <ItemRef ItemOID="Versichertenummer2" Mandatory="No" OrderNumber="3"/>
        <ItemRef ItemOID="Titel" Mandatory="No" OrderNumber="4"/>
      </ItemGroupDef>
    </MetaDataVersion>
  </Study>
</ODM>
```

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

```

<ItemRef ItemOID="Vorname" Mandatory="Yes" OrderNumber="5" />
<ItemRef ItemOID="Namenszusatz" Mandatory="No" OrderNumber="6" />
<ItemRef ItemOID="Familiennamen" Mandatory="Yes" OrderNumber="7" />
<ItemRef ItemOID="Altname" Mandatory="No" OrderNumber="8" />
<ItemRef ItemOID="Geburtsdatum" Mandatory="Yes" OrderNumber="9" />
<ItemRef ItemOID="Geschlecht" Mandatory="Yes" OrderNumber="10" />
<ItemRef ItemOID="Postleitzahl" Mandatory="No" OrderNumber="11" />
<ItemRef ItemOID="Strassennummer" Mandatory="No" OrderNumber="12" />
<ItemRef ItemOID="Ortsname" Mandatory="No" OrderNumber="13" />
<ItemRef ItemOID="Sicherheit" Mandatory="Yes" OrderNumber="14" />
<ItemRef ItemOID="PruefarztID" Mandatory="No" OrderNumber="15" />
<ItemRef ItemOID="KlinikID" Mandatory="No" OrderNumber="16" />
<ItemRef ItemOID="DatumUhrzeitMeldung" Mandatory="No" OrderNumber="17" />
</ItemGroupDef>
<ItemDef OID="Krankenkassennummer" Name="Krankenkassennummer"
      DataType="integer" Length="30">
  <tmf:Field Name="FLD_Krankenkassennummer" />
</ItemDef>
<ItemDef OID="Versichertennummer" Name="Versichertennummer"
      DataType="integer" Length="30">
  <tmf:Field Name="FLD_Versichertennummer" />
</ItemDef>
<ItemDef OID="Versichertennummer2" Name="Versichertennummer2"
      DataType="integer" Length="30">
  <tmf:Field Name="FLD_Versichertennummer2" />
</ItemDef>
<ItemDef OID="Titel" Name="Titel" DataType="text" Length="20">
  <tmf:Field Name="FLD_Titel" />
</ItemDef>
<ItemDef OID="Vorname" Name="Vorname" DataType="text" Length="20">
  <tmf:Field Name="FLD_Vorname" />
</ItemDef>
<ItemDef OID="Namenszusatz" Name="Namenszusatz" DataType="text" Length="20">
  <tmf:Field Name="FLD_Namenszusatz" />
</ItemDef>
<ItemDef OID="Familiennamen" Name="Namenszusatz" DataType="text" Length="50">
  <tmf:Field Name="FLD_Familiennamen" />
</ItemDef>
etc. etc. ...
{Weitere ItemDefs entsprechend den ItemRefs sowie CDISC CodeList Elemente hinzufügen!}
</MetaDataVersion>
</Study>
</ODM>

```

3.3.2 CDISC Field/Attribute usage.

The CDISC-ODM model defines some fields whose use needs to be more closely specified/agreed upon for application within the interface under discussion.

OIDs of study definition elements are always in accordance with the active study definition (see previous section).

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

- 1) The SubjectKey of SubjectData elements have no meaning outside the context of the transaction taking place.
- 2) The OIDs of Users and Locations may have a meaning in the context of this interface if these entities are managed in the PID server database next to the patient identifying data. To date, it is not clear yet whether the PID server will also be responsible of keeping a list of users and locations (doctor's office or hospital).

3.4 Model comparison

<i>Item</i>	<i>TMF Server model</i>	<i>CDISC model</i>
Ease of server-side implementation	+ Direct mapping to TMF Server input/output.	- Server needs understanding/store of Study Definition(s).
Ease of client-side implementation	+ Easier for non-CDISC based clients.	+ Easier for CDISC-ODM based clients
"Proprietaryness"	- 'Very' proprietary	+ Standard approach
Extensibility		+ Field extensibility based on CDISC-ODM model.

3.4.1 So which is the 'best' model ?

The complications resulting from the use of the CDISC-ODM model described raise the question why the CDISC-ODM should be used. Interfacing a TMF Client with the functionality provided by the TMF Server today would indeed be simpler if the 'TMF server based model' were chosen.

However, when looking at the TMF Server's service from a generic point of view as a repository of subject data, it does make sense to store and provide (or at least appear to store and provide) data in the form requested by a client.

The decision as to which model to choose is not so much a technical one, as a strategic one. TMF promotes CDISC as a future standard for competence networks. And TMF wants to make the PID server more generally useful for every competence network.

In the long term, if TMF decides to expand the service with, for example, the ability to store repeating elements of a CDISC-ODM model, a flat data structure will no longer suffice. Providing a service based on the CDISC-ODM model from version 1 will allow for a clean upgrade path to full CDISC storage capability should this become required in the future.

The two interface possibilities, 'TMF Server based' and 'CDISC-ODM based', are not mutually exclusive. There is no technical reason why two versions of the service could not run in parallel, allowing the client to pick the view it wishes. Obviously though, developing and supporting two services is more work than for a single one, especially when future enhancements turn out not to fit in of the two models.

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

4. Commands

The commands in this section are introduced by means of example in an informal syntax to describe the XML grammar of the SOAP body:

- The syntax is shown as an XML instance. Values are only examples unless stated otherwise.
- The following characters are appended to elements and attributes to indicate cardinality: '?' (0 or 1), '*' (0 or more), '+' (1 or more)

The following namespaces and associated prefixes are used:

Prefix	Namespace	Description
cdisc	http://www.cdisc.org/ns/odm/v1.2	CDISC ODM version 1.2
tmf	t.b.d. http://www.tmf.org/services	TMF service namespace. Must be finalised in accordance with TMF guidelines.
tmf-t	t.b.d. http://www.tmf.org/services/types	TMF datatypes namespace. Must be finalised in accordance with TMF guidelines.

SOAP Message headers are shown in later sections of this document. The formal description is provided in the form of a WSDL document in a later section.

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

4.1 Get subject data (CDISC model)

4.1.1 Request

```
<tmf:getSubjectData> {1}
  <tmf-t:RequestSubjectID Value="KW4M2LY8"/> {2}
  <cdisc:ClinicalData MetaDataVersionOID="Version.001" StudyOID="Study.TMF"/> {3}
</tmf:getSubjectData>
```

- 1) Command to be performed.
- 2) SubjectID information. Value attribute is the SubjectID itself.
- 3) Empty ClinicalData element. The StudyOID and MetaDataVersionOID attributes indicate to the recipient in which the data is to be retrieved.

4.1.2 Response – Data retrieved

```
<tmf:getSubjectDataResponse>
  <tmf-t:ResponseSubjectID ResponseCode="Retrieved" Value="KW4M2LY8"> {1}
    <tmf-t:Message>SubbjectData restrieved sucefully.</tmf-t:Message>
    <tmf-t:SubjectDataRef SubjectKey="Subj.AAA"/> {2}
  </tmf-t:ResponseSubjectID>
  <cdisc:ClinicalData MetaDataVersionOID="Version.001" StudyOID="Study.TMF">
    <cdisc:SubjectData SubjectKey="Subj.AAA"> {3}
      <cdisc:StudyEventData StudyEventOID="E.01">
        <cdisc:FormData FormOID="F.01">
          <cdisc:ItemGroupData ItemGroupOID="IG.01">
            <cdisc:ItemData ItemOID="Name" Value="M&#xC3;&#x152;ller"/>
            <cdisc:ItemData ItemOID="Vorname" Value="Peter"/>
            <cdisc:ItemData ItemOID="Geburtsdatum" Value="1954-11-23"/>
            <cdisc:ItemData ItemOID="Geschlecht" Value="1"/>
            <cdisc:ItemData ItemOID="Sicherheit" Value="1"/>
          </cdisc:ItemGroupData>
        </cdisc:FormData>
      </cdisc:StudyEventData>
    </cdisc:SubjectData>
  </cdisc:ClinicalData>
</tmf:getSubjectDataResponse>
```

- 1) The ResponseSubjectID indicates that the query succeeded. ResponseCode attribute is an enumeration of possible responses whilst the contained Message(s) contained results intended for human consumption.
- 2) The SubjectDataRef and it's associated SubjectKey attribute indicate which SubjectData corresponds to the result.
- 3) The SubjectData associated with the requested SubjectID and referenced via (2).

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

4.1.3 Response – Invalid SubjectID

```
<tmf:getSubjectDataResponse>
  <tmf-t:ResponseSubjectID ResponseCode="Invalid" Value="LW3CU9RP"> {1}
    <tmf-t:Message>SubjectID does not exist.</tmf-t:Message>
  </tmf-t:ResponseSubjectID>
</tmf:getSubjectDataResponse>
```

- 1) Response code indicating that SubjectID provided in the request is invalid.

4.1.4 Response – Unknown StudyOID/MDV OID (Fault)

```
<soapenv:Fault>
  <faultcode>soapenv:Server.generalException</faultcode> {1}
  <faultstring></faultstring>
  <detail>
    <tmf-t:StudyFault>
      <tmf-t:Reason>UnknownStudyOID</tmf-t:Reason> {2}
    </tmf-t:StudyFault>
  </detail>
</soapenv:Fault>
```

- 1) General SOAP fault code
- 2) TMF specific fault code. Enumeration of possible faults as defined in associated interface description (i.e. machine readable)

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

4.2 Get (and Create) PID

4.2.1 Request

```
<tmf:getSubjectID> {1}
  <tmf-t:RequestSubjectID Sureness="+|-"? ForceCreation="true|false"?> {2}
    <tmf-t:SubjectDataRef SubjectKey="SubjKey.AAA"/>
  </tmf-t:RequestSubjectID>
  <cdisc:ClinicalData MetaDataVersionOID="Version.001" StudyOID="Study.TMF"> {3}
    <cdisc:SubjectData SubjectKey="SubjKey.AAA">
      <cdisc:StudyEventData StudyEventOID="E.01">
        <cdisc:FormData FormOID="F.01">
          <cdisc:ItemGroupData ItemGroupOID="IG.01">
            <cdisc:ItemData ItemOID="Name" Value="M&#xC3;&#x152;ller"/>
            <cdisc:ItemData ItemOID="Vorname" Value="Peter"/>
            <cdisc:ItemData ItemOID="Geburtsdatum" Value="1954-11-23"/>
            <cdisc:ItemData ItemOID="Geschlecht" Value="1"/>
            <cdisc:ItemData ItemOID="Sicherheit" Value="1"/>
          </cdisc:ItemGroupData>
        </cdisc:FormData>
      </cdisc:StudyEventData>
    </cdisc:SubjectData>
  </cdisc:ClinicalData>
</tmf:getSubjectID>
```

- 1) Command to be performed.
- 2) Request parameters such as Sureness and ForceCreation
- 3) SubjectData information for lookup.

4.2.2 Response – No Match

```
<tmf:getSubjectIDResponse>
  <tmf-t:ResponseSubjectID ResponseCode="NoMatch"> {1}
    <tmf-t:Message>SubjectID does not exist.</tmf-t:Message>
    <tmf-t:SubjectDataRef SubjectKey="SubjKey.AAA"/>
  </tmf-t:ResponseSubjectID>
</tmf:getSubjectIDResponse>
```

- 1) Response indicating that no match was found

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

4.2.3 Response – Single match (Retrieve)

```
<tmf:getSubjectIDResponse>
  <tmf-t:ResponseSubjectID ResponseCode="Retrieved" Value="KW4M2LY8"> {1}
    <tmf-t:Message>SubjectID retrieved successfully.</tmf-t:Message>
    <tmf-t:SubjectDataRef SubjectKey="SubjKey.AAA"/>
  </tmf-t:ResponseSubjectID>
</tmf:getSubjectIDResponse>
```

- 1) Response indicating that the Subject ID was successfully retrieved. The SubjectID is returned in the Value attribute.

4.2.4 Response – Multiple match (Retrieve)

```
<tmf:getSubjectIDResponse>
  <tmf-t:ResponseSubjectID ResponseCode="MultipleMatches"> {1}
    <tmf-t:Message>Multiple results found.</tmf-t:Message>
    <tmf-t:SubjectDataRef SubjectIDValue="KW4M2LY8" SubjectKey="SubjKey.AAA"/> {2}
    <tmf-t:SubjectDataRef SubjectIDValue="LW3CU9RP" SubjectKey="SubjKey.BBB"/> {2}
  </tmf-t:ResponseSubjectID>
  <cdisc:ClinicalData MetaDataVersionOID="Version.001" StudyOID="Study.TMF">
    <cdisc:SubjectData SubjectKey="SubjKey.AAA"> {3}
      <cdisc:StudyEventData StudyEventOID="E.01">
        <cdisc:FormData FormOID="F.01">
          <cdisc:ItemGroupData ItemGroupOID="IG.01">
            <cdisc:ItemData ItemOID="Name" Value="M&#xC3; &#x152;ller"/>
            <cdisc:ItemData ItemOID="Vorname" Value="Peter"/>
            <cdisc:ItemData ItemOID="Geburtsdatum" Value="1954-11-23"/>
            <cdisc:ItemData ItemOID="Geschlecht" Value="1"/>
            <cdisc:ItemData ItemOID="Sicherheit" Value="1"/>
          </cdisc:ItemGroupData>
        </cdisc:FormData>
      </cdisc:StudyEventData>
    </cdisc:SubjectData>
    <cdisc:SubjectData SubjectKey="SubjKey.BBB"> {3}
      <cdisc:StudyEventData StudyEventOID="E.01">
        <cdisc:FormData FormOID="F.01">
          <cdisc:ItemGroupData ItemGroupOID="IG.01">
            <cdisc:ItemData ItemOID="Name" Value="Maier"/>
            <cdisc:ItemData ItemOID="Vorname" Value="Peter"/>
            <cdisc:ItemData ItemOID="Geburtsdatum" Value="1954-11-23"/>
            <cdisc:ItemData ItemOID="Geschlecht" Value="1"/>
            <cdisc:ItemData ItemOID="Sicherheit" Value="1"/>
          </cdisc:ItemGroupData>
        </cdisc:FormData>
      </cdisc:StudyEventData>
    </cdisc:SubjectData>
```

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

```
</cdisc:ClinicalData>
</tmf:getSubjectIDResponse>
```

- 1) Response indicating that multiple matches were found.
- 2) SubjectDataRef elements reference the SubjectData elements which match the supplied SubjectData from the request. Since multiple results are available, the respective SubjectIDs are returned as SubjectIDValue attribute of the SubjectDataRef elements instead of the ResponseSubjectID itself.
- 3) The SubjectData elements carry the data of the subjects which matched the request data. Each of these SubjectData elements will be referenced from a SubjectDataRef element.

4.2.5 Response – Created

```
<tmf:getSubjectIDResponse>
  <tmf-t:ResponseSubjectID ResponseCode="Created" Value="KW4M2LY8"> {1}
    <tmf-t:Message>SubjectID created succefully.</tmf-t:Message>
    <tmf-t:SubjectDataRef SubjectKey="SubjKey.AAA"/> {2}
  </tmf-t:ResponseSubjectID>
</tmf:getSubjectIDResponse>
```

- 1) The ResponseSubjectID indicates that a SubjectID was created. The SubjectID is returned in the Value attribute of the response element.
- 2) The SubjectDataRef element holds a reference to the SubjectData element in the request which lead to the creation of this SubjectData. As in this example, it is not necessary to return the SubjectData itself since it was provided by the client in the first place.

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

4.3 Is Subject ID Valid

4.3.1 Request

```
<tmf:isSubjectIDValid> {1}
  <tmf-t:RequestSubjectID Value="KW4M2LY8" /> {2}
</tmf:isSubjectIDValid>
```

- 1) Command to be performed.
- 2) SubjectID to be checked.

4.3.2 Response – valid

```
<tmf:isSubjectIDValidResponse>
  <tmf-t:ResponseSubjectID ResponseCode="Valid" Value="KW4M2LY8"> {1}
    <tmf-t:Message>SubjectID is valid.</tmf-t:Message> {2}
  </tmf-t:ResponseSubjectID>
</tmf:isSubjectIDValidResponse>
```

- 1) Response indicating that the given SubjectID is valid.
- 2) Response message for human consumption.

4.3.3 Response – invalid

```
<tmf:isSubjectIDValidResponse>
  <tmf-t:ResponseSubjectID ResponseCode="Invalid" Value="SubjID.Invalid"> {1}
    <tmf-t:Message>SubjectID is not valid.</tmf-t:Message> {2}
  </tmf-t:ResponseSubjectID>
</tmf:isSubjectIDValidResponse>
```

- 1) Response indicating that the given SubjectID is invalid.
- 2) Response message for human consumption.

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

5. Security

The main facets of security are encryption, and authentication.

It is assumed at this stage that encryption, message authentication, and server authentication will be provided by well know and secure underlying transport mechanisms such as HTTPS/SSL and server certificates.

For user authentication, three scenarios can be envisaged:

- 1) client certificate authentication – also using underlying HTTPS protocol. In this case, no further consideration need be made in the SOAP message since authentication is below the SOAP level.
- 2) uid/password pair in header of SOAP request. This scenario comes in use when no client side certificate is available/to be used. Instead, authentication is performed by means of a user identifier and secret password.
- 3) trusted gateway. In this scenario, authentication between client and gateway is performed by means outside the scope of this interface (although probably according to 1 or 2 above). The gateway in turn authenticates itself using 1 with the TMF server. It adds a the uid of the authenticated client in the SOAP message header allowing the TMF server to log activity/define privileges according to the real client which submitted the request. This scenario may also be used in trusted gateways which provide, for example, interface to the TMF interface via an HTML front end. In such a case, the gateway in the request chain is also the initiator of a SOAP request.

6. WSDL

Documents forming part of the WSDL for the example interface discussed in this document are provided separately.

tmf.wsdl	WSDL definition itself. References the following two documents.
tmf-types.xsd	TMF/Interface specific type schema.
ODM1-2-0.xsd	CDISC ODM schema (source: http://www.cdisc.org/schema/ODM1-2-0.xsd).

tmf-interface-concept 4.sxw	Version: 0.4
Interface concept for TMF Patient List Server / PID Generator	Date: 2005-01-20

7. Appendix

Example of alternative SOAP message payload modelled on today's TMF Server

7.1 Get subject data (TMF Model)

7.1.1 Request

```
<tmf:getSubjectData> {1}
  <tmf:RequestPID Value="PID.AAA" /> {2}
</tmf:getSubjectData>
```

7.1.2 Response

```
<tmf:getSubjectDataResponse>
  <tmf:ResponsePID ResponseCode="Retrieved" Value="PID.AAA"> {1}
    <tmf:Message>SubjectData retrieved succefully.</tmf:Message>
    <tmf:SubjectData>
      <tmf:ItemData Name="Name" Value="M&#xC3;&#x152;ller" />
      <tmf:ItemData Name="Vorname" Value="Peter" />
      <tmf:ItemData Name="Geburtsdatum" Value="1954-11-23" />
      <tmf:ItemData Name="Geschlecht" Value="1" />
      <tmf:User>
        <tmf:DisplayName>Dr. Hans Meier</tmf:DisplayName>
        <tmf:FirstName>Hans</tmf:FirstName>
      </tmf:User>
      <tmf:Location>
        <tmf:DisplayName>Frankfurt</tmf:DisplayName>
      </tmf:Location>
    </tmf:SubjectData>
  </tmf:ResponsePID>
</tmf:getSubjectDataResponse>
```