DAMAS IT

Web Services Interface

Documentation

Damas IT
Web Services Interface
Documentation

Jankovcova 1037/49, CZ – 170 00 Prague 7

Project:	DAMAS IT
Project – Subject:	Web Services Interface
Document Title:	Documentation
Author:	Unicorn
Contact:	E-mail: info@unicornsystems.eu

1. List of Contents

2. INTRODUCTION

2.1 Revision History

Version	Date	Author	Description
1.0	25.12.2007	Unicorn	First version of documentation of Web Services.
1.1	30.05.2008	Unicorn	Revision of documentation.
1.2	08.12.2008	Unicorn	Revision of documentation after the next implementation phase for the 1st January 2008.
1.3	15.12.2008	Unicorn	Revision of xml examples.
1.4	20.07.2008	Unicorn	Revision of Description of <wsa:To> element (chapter 6.3.1 SOAP Request Description)
1.5	23.11.2009	Unicorn	Revision of Senders Time Series Identification element (8.2.3.2 Specifications of Schedule Message Elements)
1.6	18.12.2009	Unicorn	Documentation update (clarification) in section 7.3.2. Downloading Daily ATC
1.7	10.2.2011	Unicorn	Revision of Capacity Agreement Identification in Schedule Messages (8.2.3.2 Specifications of Schedule Message Elements) Revision of 8.3.3 Capacity Contract Type
2.0	10.2.2011	Unicorn	Final document after DITA4 implementation
3.0	13.2.2013	Unicorn	Update after DITA5 implementation – Intraday Review of the document prior to publication for TERNA
3.1	04.09.2015	Unicorn	Added FAQ section
4.0	01.02.2019	Unicorn	Updated due to Damas Upgrade.
5.0	18.03.2019	Unicorn	Added link to Damas Hash password converter

2.2 Purpose of the Document

This document is specification for general approach that must be taken when accessing Damas web service. Web services are built on industry standard technologies. They are available on the Internet and ensure the same level of the privacy and security as Damas web site.

2.3 Document Organization

Chapter 3 provides a brief introduction to web services with emphasis on web service properties and existing standards. Chapter 4 focuses on explaining the implemented Damas web service interface and its use for automated communication with other systems. Detailed technical information about the implemented SOAP format and WSDL is presented in Chapter 5. The main subject of Chapter 6 is the security of web services and data transfers. A comprehensive description of all existing data flows for downloading and uploading data from/to the Damas system is included in Chapter 7. Detailed technical description of web services is given in Chapter 8 that contains the specification and explanation for all XSD schemas used in the Damas system.

3. Web Services Description

3.1 What Are Web Services?

Web services are the cornerstone of the movement towards distributed computing on the Internet. Open standards and focus on communication among and collaboration of people and applications have created an environment where web services are becoming the platform for application integration. Applications are constructed using multiple web services from various sources that work together regardless of where they are located or how they were implemented.

3.2 Benefits

Web services selected as the platform to enable the integration of the Damas system with other systems represent an up-to-date and high-performance technology and have the following major advantages:

- Full platform independence - Web services can be utilized by any application that is able to communicate with its environment using the HTTP/HTTPS protocol. It does not matter what kind of technology is used to run the web service itself.
- Easy implementation and consumption - Because of the advanced technological development, today it is very easy to create a web service and to provide it to clients or to the public.

Web services simplify the process of data preparation and processing, as the human end user is replaced by an end user that is actually an application. Obviously, the human factor is not fully eliminated from the process but the end user can move the burden of his/her requirements to the application. The end user submits his or her requirements to this application and requests all the data necessary to make decisions. Users do not have to care how the application handles or acquires the data. The application handles the entire communication with the Damas system.

Key advantages for the user include:

- Elimination of routine and burdensome communication with the Damas system - It is not necessary to login to the system, clicking and selecting the file, waiting for the confirmation that the file was received, etc.
- Integration of all user activities into a single system - If the company has software designed to facilitate trading with electricity, then it's likely that the data related to the Damas system will be processed by such system. In such a case, the data do not have to be exported and transferred to the Damas system in a complicated way, but after creating a simple communication component, they can be forwarded to the proper location by a simple click.

Of course the described method can be combined with the existing conventional approach. The user can always decide how to communicate with the Damas to achieve his or her goal.

3.3 Standards

Web services are applications whose logic and functions are accessible using the standard Internet protocols and data formats, such as Hypertext Transfer Protocol (HTTP) and Extensible Markup Language (XML).

XML web services provide useful functionality to web users through the Simple Object Access Protocol (SOAP), which is the essential standard for information exchange in a decentralized, distributed environment. It is an XML-based protocol that consists of an envelope that defines a framework for describing what is in a message and how to process it and of a convention for representing remote procedure calls and responses.

4. Web Services Interface

This chapter provides an overview and explanation of major properties of Damas web services implementation. In the first part of the chapter, principles of the transfer technology are introduced and the structure of data formats is briefly described. The second part contains instructions for calling web services.

4.1 Transfer Technology

Web services in the Damas system can be used for automated data exchange or for machine-controlled data exchange. Use of this technology significantly simplifies the communication among the Damas system, local operator and TSOs.

The main transfer unit of is a text file containing an SOAP XML message. The format of the SOAP message in the Damas system was designed according to the SOAP 1.1 specification recommended by W3C (http://www.w3.org/TR/2000/NOTE-SOAP-20000508/).

The supported communication protocol is HTTPS. A common authentication process, containing login name and password, is defined. Login and password must be sent with each SOAP message so the message could be processed. All actioned performed using web services are executed in Damas system with permissions of the user whose creadentials are provided in SOAP message.

Creadentials must be provideded in form of Username token in accordance with Web Services Security specification. For details of Web Services Security, see http://www.oasis-open.org/committees/tc_home.php_?wg_abbrev=wss.

4.2 Data Format

Every web service message used in Damas consists of two parts:

- Header of the web service message
- Body of the web service message.

For all data flows designed for sending data to the Damas system, the XML file containing business data to be transferred is included in the body of the web service message. The structure of the XML file is defined by XSD charts, which make it possible to validate semantics of a XML message. The structure is based on the ESS standard provided by the ETSO (http://www.etso-net.org/) and defines the XML formats in the power-producing domain in Europe. The XML files used in the Damas system are implemented according to the ESS standard v3r0 with Code List v4r0 (http://www.edi.etso-net.org/). All the XSD definitions specifying the formats of the XML files to be exchanged are listed in the attachment to this document.

For proper data exchange, it is necessary to synchronize the mechanism of entity (auction participants and their partners) identification in order to match the scheduling

charts. The Damas system assumes that the international EIC standard is used to identify the entities and their partners abroad. The EIC codes are standardized by the ETSO as the substitute for the current national identification standards, as well as the substitute for the EAN codes used in some countries, because of high acquisition costs.

4.3 Interface of Damas Web Services

The IS Damas web services are accessible at following addresses:

Environment	Address	Protocol	Port
Testing environment	https://damas-test.terna.it/DamasService.svc?singlewsdl	https	443
Business environment	https://damas.terna.it/DamasService.svc?singlewsdl	https	443

The following web service interfaces are implemented in the Damas system to provide communication with neighboring systems:

Name	SOAP request	SOAP response	Description
Synchronous request	RunSynchrous	RunSynchrous Response	It provides synchronous exchange of the commercial data with Damas.
Asynchronous request	RunAsynchrous	RunAsynchrous Response	It provides asynchronous exchange of the commercial data with Damas.
Asynchronous request state	CheckRQResult	CheckRQResult Response	It returns status of the asynchronous request that is being processed by Damas.
Actual date and time	GetActualDateTime	GetActualDateTime Response	It returns current system date and time that is important for automatic operations carried out by the system.

The web services of the Damas system can be used in either synchronous or asynchronous mode:

Synchronous call of a web service - Data are passed to the web service via the RunSynchrous method. By performing this step a synchronous request is established within the Damas system and processed, and the result is returned back.The output parameter of this web service is an XML with a structure varying for the individual data streams.
Asynchronous call of a web service - Data in this case are passed to the web service via the RunAsynchrous method. A request is established within the Damas system as asynchronous. In this case, the output of the web service does not include the processing of established request but rather it contains only an ID of the request. This ID is used to request the result of such request later on.

5. WEB SERVICE TECHNOLOGIES

Technical information and detailed description of web service implementation are provided in this Chapter. Firstly, the structure of the SOAP message used in the Damas system is introduced and explained. After that, the WSDL description of the request and response SOAP message formats for all web services is presented.

5.1 SOAP

Structure of the SOAP message is implemented according to the SOAP 1.1 specification recommended by W3C (http://www.w3.org/TR/2000/NOTE-SOAP-20000508/).

5.2 SOAP Message

The SOAP message implemented in Damas consists of the SOAP header and body.

UTF-8 encoding is required for all SOAP messages passed into Damas. All outgoing messages are UTF-8 encoded as well.

The SOAP header contains information that is essential for user authorization, such as the user's login name, password and digital signature.

```
<soap:Header>
  <!-- WSS Security Header -->
</soap:Header>
```

WSS Security Header contains security tokens necessary to authenticate sender and check message integrity. These tokens are electronic sign and user credentials. For details of WSS Security see Chapter Web service security.

The body of SOAP message includes the element, which contains the input/output parameter class. The element name is derived from the name of the web service that is used.

```
<soap:Body>
  <WebServiceName xmlns="http://www.terna.it/damas/wse">
     Input/Output Parameters
  </WebServiceName>
</soap:Body>
```

Whole element `soap:Body` must be signed together with credentials data located in `soap:Header`. For details of WSS Security, Header, see Chapter Web service security.

5.2.1 Input Parameters

The parameter class defined for input parameters:

```
    <Input>
      <FID>FID</FID>
```

```
        <Parameters>
          <XXXParam Name="param_name1">param_val1</XXXParam>
          <XXXParam Name="param_name2">param_val2</XXXParam>
          …
          <XXXParam Name="param_nameN">param_valN</XXXParam>
        </Parameters>
      </Input>
```

The highlighted parameter shall be replaced by values according to following rules:

Parameter	Type	Description	Note
FID	String	Identification of the dataflow. See chapter Data flows.	Unique for each data flow.
XXXParam		Element name of the parameter represents its data type. For overview of the supported data types see table below.	Depends on the data flow.
param_nameX	String	Name of data flow input parameter	Depends on the data flow.
param_valX	String, Number, Date	Value of data flow input parameter	Depends on the data flow.

List of the input parameter data types:

Data type element name	Corresponding XSD type	Example
BooleanParam	xs:Boolean	True
DateParam	xs:date	2002-09-24
DateTimeParam	xs:dateTime	2002-09-24T09:30:10Z
DecimalParam	xs:decimal	999.50
IntParam	xs:int	999
StringParam	xs:string	TEXT
XmlParam	Any XML node tree (corresponds to <xs:any> XSD element).	Any XML node

Elements with data flow input parameters (XXXParam) must be alphabetically ordered by their type names (that is `<BooleanParam>` elements come first, `<DateParam>` elements come second etc.).

5.2.2 Output Parameters

The parameter class defined for output parameters:

```
<Output>
  <RQID>RQID</RQID>
  <Result>resultXML</Result>
  <RQState>
    <Code>RQState_Code</Code>
    <Description>RQState_Description</Description>
  </RQState>
</Output>
```

The highlighted parameter shall be replaced by values according to following rules:

Parameter	Type	Description	Note
RQID	Number	Unique identification of the asynchronous request in Damas system	
resultXML	String	Contains the result of a request	Depends on the data flow; see data flows description
RQState_Code,	String	Code of state of asynchronous request. (For list of the possible codes see Chapter SOAP CheckRQResultResponse).	This field is used only when calling CheckRQResult method (see Chapter Asynchronous Request State for details). For all other methods this item is not present.
RQState_Description	String	Description of the state the request.	This field is used only when calling CheckRQResult method (see Chapter Asynchronous Request State for details). For all other methods this item is not present.

5.2.3 Error Handling

Errors returned by Damas web services interface are divided into two basic groups:

- Business errors – These errors originate in business control algorithms and it express that imported business data violates business rules. This applies only to input data flows (see Chapter Data Flows for Sending Data). These errors are returned in form of the Acknowledgement as standard output of the data flow (see Chapter Output Parameters) and therefore are not subject of this chapter.
- System errors – These errors represent non-business faults. This includes user authentication errors, bad format of the SOAP xml, input parameters etc. System errors are listed below.

Errors are distributed to the client by using <soap:Fault> element, as defined in SOAP/1.1 specification (see http://www.w3.org/TR/2000/NOTE-SOAP-20000508/#_Toc478383507).

Detailed information about error is carried in the `<Error>` element (see example of the SOAP fault below):

```
<soap:Fault>
  <faultcode>faultcode</faultcode>
  <faultstring>faultstring</faultstring>
  <detail>
    <e:Error xmlns:e="http://www.terna.it/damas/xsd/errors.xsd">
      <ErrID>errID</ErrID>
      <ErrDescr>errDescr</ErrDescr>
      <ErrXML>errXML</ErrXML>
    </e:Error>
  </detail>
</soap:Fault>
```

The highlighted parameter shall be replaced by values according to the following rules:

Parameter	Type	Description	Note
faultcode	String	The code of the error as specified in SOAP/1.1.	
faultstring	String	The description of the error as specified in SOAP/1.1.	
ErrID	Number	The identification number of the error.	See List of Standard Errors for more information.
ErrDescr	String	The short description of the error.	See List of Standard Errors for more information.
ErrXML	XML	Additional debug information.	

The `<e:Error>` element doesn't have to be present in the Fault message. It is present only for errors with faultcode of the "soap:Client" value or "soap:Server" value (see link http://www.w3.org/TR/2000/NOTE-SOAP-20000508/#_Toc478383507 for details on faultcode).

List of the Standard Errors:

Error ID	Error Description	Fault Code
-130	User is not authorized for the requested data stream.	soap:Client
-501	Date is invalid.	soap:Client
-502	Unknown entity code. It concerns local operators codes.	soap:Client
-506	Unknown Control Area code.	soap:Client
-508	Unknown capacity type code.	soap:Client
-509	Unknown UPV/UCV code.	Soap:Client
-510	Data flow with requested FID does not exist.	soap:Client
-512	Invalid XML format. Errors occurred during XSD validation of submitted data XML.	soap:Client
-513	Invalid data flow input parameters	soap:Client
-514	Internal server error	soap:Server
-515	Requested data has not been published yet.	soap:Client
-516	Requested date range is not valid	soap:Client
-517	Asynchronous request does not exist	soap:Client
-518	Requested operation is not permitted for this data flow	soap:Client
-520	Invalid date range	soap:Client
-521	Control Area 1 must be TERNA code.	soap:Client

5.3 WSDL

This part of the document contains description of all web services provided by the Damas system as interface for automatic communication with other system.

The SOAP request format for establishing the synchronous request in Damas

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://sche-
mas.xmlsoap.org/wsdl/soap/" xmlns:soapenc="http://sche-
```

```
mas.xmlsoap.org/soap/encoding/" xmlns:tm="http://mi-
crosoft.com/wsdl/mime/textMatching/"
xmlns:tns="http://www.terna.it/damas/wse" target-
Namespace="http://www.terna.it/damas/wse">
   <wsdl:types>
      <s:schema elementFormDefault="qualified" target-
Namespace="http://www.terna.it/damas/wse">
         <s:element name="CheckRQResult">
            <s:complexType>
               <s:sequence>
                  <s:element minOccurs="1" maxOccurs="1" name="RQID"
type="s:int" />
               </s:sequence>
            </s:complexType>
         </s:element>
         <s:element name="CheckRQResultResponse">
            <s:complexType>
               <s:sequence>
                  <s:element minOccurs="1" maxOccurs="1" name="Output"
nillable="true" type="tns:OutParams" />
               </s:sequence>
            </s:complexType>
         </s:element>
         <s:complexType name="OutParams">
            <s:sequence>
               <s:element minOccurs="1" maxOccurs="1" name="RQID"
type="s:int" />
               <s:element minOccurs="0" maxOccurs="1" name="Result">
                  <s:complexType mixed="true">
                     <s:sequence>
                        <s:any />
                     </s:sequence>
                  </s:complexType>
               </s:element>
               <s:element minOccurs="1" maxOccurs="1" name="RQState"
nillable="true" type="tns:RequestState" />
            </s:sequence>
         </s:complexType>
         <s:complexType name="RequestState">
            <s:sequence>
               <s:element minOccurs="0" maxOccurs="1" name="Code"
type="s:string" />
               <s:element minOccurs="0" maxOccurs="1" name="Descrip-
tion" type="s:string" />
            </s:sequence>
         </s:complexType>
         <s:element name="RunAsynchrous">
            <s:complexType>
               <s:sequence>
                  <s:element minOccurs="1" maxOccurs="1" name="Input"
nillable="true" type="tns:InParams" />
```

```
               </s:sequence>
            </s:complexType>
         </s:element>
         <s:complexType name="InParams">
            <s:sequence>
               <s:element minOccurs="0" maxOccurs="1" name="FID"
type="s:string" />
               <s:element minOccurs="1" maxOccurs="1" name="Message-
Header" nillable="true" type="tns:SAPMessageHeader" />
               <s:element minOccurs="0" maxOccurs="1" name="Parame-
ters" type="tns:ParametersCollection" />
            </s:sequence>
         </s:complexType>
         <s:complexType name="SAPMessageHeader">
            <s:sequence>
               <s:element minOccurs="0" maxOccurs="1" name="SubsCode"
type="s:string" />
               <s:element minOccurs="0" maxOccurs="1" name="Task"
type="s:string" />
               <s:element minOccurs="0" maxOccurs="1" name="RequestID"
type="s:string" />
               <s:element minOccurs="0" maxOccurs="1" name="SchemaVer-
sion" type="s:string" />
               <s:element minOccurs="1" maxOccurs="1" name="Confirma-
tion" type="s:long" />
               <s:element minOccurs="1" maxOccurs="1" name="Crea-
tionDateTime" type="s:dateTime" />
            </s:sequence>
         </s:complexType>
         <s:complexType name="ParametersCollection">
            <s:sequence>
               <s:element minOccurs="0" maxOccurs="unbounded"
name="BooleanParam" nillable="true" type="tns:BooleanParam" />
               <s:element minOccurs="0" maxOccurs="unbounded"
name="DateParam" nillable="true" type="tns:DateParam" />
               <s:element minOccurs="0" maxOccurs="unbounded"
name="DateTimeParam" nillable="true" type="tns:DateTimeParam" />
               <s:element minOccurs="0" maxOccurs="unbounded"
name="DecimalParam" nillable="true" type="tns:DecimalParam" />
               <s:element minOccurs="0" maxOccurs="unbounded"
name="IntParam" nillable="true" type="tns:IntParam" />
               <s:element minOccurs="0" maxOccurs="unbounded"
name="StringParam" nillable="true" type="tns:StringParam" />
               <s:element minOccurs="0" maxOccurs="unbounded"
name="XmlParam" nillable="true" type="tns:XmlParam" />
               <s:element minOccurs="0" maxOccurs="unbounded"
name="XmlStringParam" nillable="true" type="tns:XmlStringParam" />
            </s:sequence>
         </s:complexType>
         <s:complexType name="BooleanParam">
            <s:simpleContent>
```

```
            <s:extension base="s:boolean">
               <s:attribute name="Name" type="s:string" />
            </s:extension>
         </s:simpleContent>
      </s:complexType>
      <s:complexType name="DateParam">
         <s:simpleContent>
            <s:extension base="s:date">
               <s:attribute name="Name" type="s:string" />
            </s:extension>
         </s:simpleContent>
      </s:complexType>
      <s:complexType name="DateTimeParam">
         <s:simpleContent>
            <s:extension base="s:dateTime">
               <s:attribute name="Name" type="s:string" />
            </s:extension>
         </s:simpleContent>
      </s:complexType>
      <s:complexType name="DecimalParam">
         <s:simpleContent>
            <s:extension base="s:decimal">
               <s:attribute name="Name" type="s:string" />
            </s:extension>
         </s:simpleContent>
      </s:complexType>
      <s:complexType name="IntParam">
         <s:simpleContent>
            <s:extension base="s:int">
               <s:attribute name="Name" type="s:string" />
            </s:extension>
         </s:simpleContent>
      </s:complexType>
      <s:complexType name="StringParam">
         <s:simpleContent>
            <s:extension base="s:string">
               <s:attribute name="Name" type="s:string" />
            </s:extension>
         </s:simpleContent>
      </s:complexType>
      <s:complexType name="XmlParam">
         <s:sequence>
            <s:any minOccurs="0" maxOccurs="1" />
         </s:sequence>
         <s:attribute name="Name" type="s:string" />
      </s:complexType>
      <s:complexType name="XmlStringParam">
         <s:simpleContent>
            <s:extension base="s:string">
               <s:attribute name="Name" type="s:string" />
            </s:extension>
```

```
            </s:simpleContent>
         </s:complexType>
         <s:element name="RunAsynchrousResponse">
            <s:complexType>
               <s:sequence>
                  <s:element minOccurs="1" maxOccurs="1" name="Output"
nillable="true" type="tns:OutParams" />
               </s:sequence>
            </s:complexType>
         </s:element>
         <s:element name="RunSynchrous">
            <s:complexType>
               <s:sequence>
                  <s:element minOccurs="1" maxOccurs="1" name="Input"
nillable="true" type="tns:InParams" />
               </s:sequence>
            </s:complexType>
         </s:element>
         <s:element name="RunSynchrousResponse">
            <s:complexType>
               <s:sequence>
                  <s:element minOccurs="1" maxOccurs="1" name="Output"
nillable="true" type="tns:OutParams" />
               </s:sequence>
            </s:complexType>
         </s:element>
         <s:element name="GetActualDateTime">
            <s:complexType />
         </s:element>
         <s:element name="GetActualDateTimeResponse">
            <s:complexType>
               <s:sequence>
                  <s:element minOccurs="1" maxOccurs="1" name="Output"
nillable="true" type="tns:OutParams" />
               </s:sequence>
            </s:complexType>
         </s:element>
      </s:schema>
   </wsdl:types>
   <wsdl:message name="CheckRQResultSoapIn">
      <wsdl:part name="parameters" element="tns:CheckRQResult" />
   </wsdl:message>
   <wsdl:message name="CheckRQResultSoapOut">
      <wsdl:part name="parameters" element="tns:CheckRQResultResponse"
/>
   </wsdl:message>
   <wsdl:message name="RunAsynchrousSoapIn">
      <wsdl:part name="parameters" element="tns:RunAsynchrous" />
   </wsdl:message>
   <wsdl:message name="RunAsynchrousSoapOut">
```

```
      <wsdl:part name="parameters" element="tns:RunAsynchrousResponse"
/>
   </wsdl:message>
   <wsdl:message name="RunSynchrousSoapIn">
      <wsdl:part name="parameters" element="tns:RunSynchrous" />
   </wsdl:message>
   <wsdl:message name="RunSynchrousSoapOut">
      <wsdl:part name="parameters" element="tns:RunSynchrousResponse"
/>
   </wsdl:message>
   <wsdl:message name="GetActualDateTimeSoapIn">
      <wsdl:part name="parameters" element="tns:GetActualDateTime" />
   </wsdl:message>
   <wsdl:message name="GetActualDateTimeSoapOut">
      <wsdl:part name="parameters" element="tns:GetActualDateTi-
meResponse" />
   </wsdl:message>
   <wsdl:portType name="DamasServiceSoap">
      <wsdl:operation name="CheckRQResult">
         <documentation xmlns="http://schemas.xmlsoap.org/wsdl/">This
method
returns the status of an asynchronous request that is being processed
by
the Damas system.</documentation>
         <wsdl:input message="tns:CheckRQResultSoapIn" />
         <wsdl:output message="tns:CheckRQResultSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="RunAsynchrous">
         <documentation xmlns="http://schemas.xmlsoap.org/wsdl/">This
method
provides the asynchronous exchange of commercial data with the Damas
system.</documentation>
         <wsdl:input message="tns:RunAsynchrousSoapIn" />
         <wsdl:output message="tns:RunAsynchrousSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="RunSynchrous">
         <documentation xmlns="http://schemas.xmlsoap.org/wsdl/">This
method
provides the synchronous exchange of commercial data with the Damas
system.</documentation>
         <wsdl:input message="tns:RunSynchrousSoapIn" />
         <wsdl:output message="tns:RunSynchrousSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="GetActualDateTime">
         <documentation xmlns="http://schemas.xmlsoap.org/wsdl/">This
method
returns the current system date and time that is important for auto-
matic
operations carried out by the system.</documentation>
         <wsdl:input message="tns:GetActualDateTimeSoapIn" />
         <wsdl:output message="tns:GetActualDateTimeSoapOut" />
```

```
      </wsdl:operation>
   </wsdl:portType>
   <wsdl:binding name="DamasServiceSoap" type="tns:DamasServiceSoap">
      <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document" />
      <wsdl:operation name="CheckRQResult">
         <soap:operation soapAc-
tion="http://www.terna.it/damas/wse/CheckRQResult" style="document" />
         <wsdl:input>
            <soap:body use="literal" />
         </wsdl:input>
         <wsdl:output>
            <soap:body use="literal" />
         </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="RunAsynchrous">
         <soap:operation soapAc-
tion="http://www.terna.it/damas/wse/RunAsynchrous" style="document" />
         <wsdl:input>
            <soap:body use="literal" />
         </wsdl:input>
         <wsdl:output>
            <soap:body use="literal" />
         </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="RunSynchrous">
         <soap:operation soapAc-
tion="http://www.terna.it/damas/wse/RunSynchrous" style="document" />
         <wsdl:input>
            <soap:body use="literal" />
         </wsdl:input>
         <wsdl:output>
            <soap:body use="literal" />
         </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="GetActualDateTime">
         <soap:operation soapAc-
tion="http://www.terna.it/damas/wse/GetActualDateTime" style="docu-
ment" />
         <wsdl:input>
            <soap:body use="literal" />
         </wsdl:input>
         <wsdl:output>
            <soap:body use="literal" />
         </wsdl:output>
      </wsdl:operation>
   </wsdl:binding>
   <wsdl:service name="DamasService">
      <documentation xmlns="http://sche-
mas.xmlsoap.org/wsdl/">&lt;p&gt;This
```

```
service is secured in accordance with &lt;a href="http://www.oa-
sisopen.
org/committees/tc_home.php?wg_abbrev=wss"&gt;OASIS Web Services
Security&lt;/a&gt; specification. For details on this topic see Damas
Web
Services Documentation &lt;/p&gt;&lt;p&gt;Here you can download the
public key of current certificate which is used to sign service
responses:&lt;br&gt;&lt;/p&gt;</documentation>
      <wsdl:port name="DamasServiceSoap" binding="tns:DamasService-
Soap">
         <soap:address location="https://damas.terna.it/DamasSer-
vice.svc?singlewsdl" />
      </wsdl:port>
   </wsdl:service>
</wsdl:definitions>
```

5.3.1 Synchronous Request

This web service ensures synchronous exchange of the commercial data with Damas.

5.3.1.1 SOAP RunSynchrous

The SOAP request format for establishing the synchronous request in Damas

```
POST /wse/DamasService.asmx HTTP/1.1
Host: procedure.terna.it
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction:"http://www.terna.it/damas/wse/RunSynchrous"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://sche-
mas.xmlsoap.org/soap/envelope/" xmlns:wsa="http://sche-
mas.xmlsoap.org/ws/2004/03/addressing" >
  <soap:Header>
    <!-- WSS Security Header -->
  </soap:Header>
  <soap:Body>
    <RunSynchrous xmlns="http://www.terna.it/damas/wse">
      <Input>
        <FID>FID</FID>
        <Parameters>
          <XXXParam Name="param_name1">param_val1</XXXParam>
          <XXXParam Name="param_name2">param_val2</XXXParam>
          …
          <XXXParam Name="param_nameN">param_valN</XXXParam>
        </Parameters>
      </Input>
    </RunSynchrous>
  </soap:Body>
```

```
</soap:Envelope>
```

5.3.1.2 SOAP RunSynchrous Response

The SOAP response format with result of the synchronous request returned from Damas:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://sche-
mas.xmlsoap.org/soap/envelope/" xmlns:wsa="http://sche-
mas.xmlsoap.org/ws/2004/03/addressing" >
  <soap:Header>
    <!-- WSS Security Header -->
  </soap:Header>
  <soap:Body>
    <RunSynchrousResponse xmlns="http://www.terna.it/damas/wse">
      <Output>
        <RQID>-1</RQID>
        <Result>resultXML</Result>
        <RQState/>
      </Output>
    </RunSynchrousResponse>
  </soap:Body>
</soap:Envelope>
```

For details of the WSS Security, Header, see Chapter Web service security. Note that <RQID> element in this case contains -1 (id of the request is not returned for synchronous requests). Also note that <RQState/>element is empty (this element is used only when calling Check-RQResult method, see Chapter Asynchronous Request State).

5.3.2 Asynchronous Request

This web service ensures the asynchronous exchange of the commercial data with Damas. The RunSynchrous and RunAsynchrous methods use almost identical formats of the SOAP request and response. Asynchronous call is used for uploading larger XML (more than 2 timeseries).

5.3.2.1 SOAP RunAsynchrous

The SOAP request format for establishing the asynchronous request in the Damas system

```
POST /wse/DamasService.asmx HTTP/1.1
Host: procedure.terna.it
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction:“http://www.terna.it/damas/wse/RunAsynchrous"
```

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://sche-
mas.xmlsoap.org/soap/envelope/" xmlns:wsa="http://sche-
mas.xmlsoap.org/ws/2004/03/addressing" >
  <soap:Header>
    <!-- WSS Security Header -->
  </soap:Header>
  <soap:Body>
    <RunAsynchrous xmlns= "http://www.terna.it/damas/wse" >
      <Input>
        <FID>FID</FID>
        <Parameters>
          <XXXParam Name="param_name1">param_val1</XXXParam>
          <XXXParam Name="param_name2">param_val2</XXXParam>
          …
          <XXXParam Name="param_nameN">param_valN</XXXParam>
        </Parameters>
      </Input>
    </RunAsynchrous>
  </soap:Body>
</soap:Envelope>
```

5.3.2.2 SOAP RunAsynchrous Response

The SOAP response format with result of the asynchronous request returned from Damas

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://sche-
mas.xmlsoap.org/soap/envelope/" xmlns:wsa="http://sche-
mas.xmlsoap.org/ws/2004/03/addressing" >
  <soap:Header>
    <!-- WSS Security Header -->
  </soap:Header>
  <soap:Body>
    <RunAsynchrousResponse xmlns="http://www.terna.it/damas/wse">
      <Output>
        <RQID>RQID</RQID>
        <Result/ >
        <RQState/>
      </Output>
    </RunAsynchrousResponse>
  </soap:Body>
</soap:Envelope>
```

For details of WSS Security, Header, see Chapter Web service security. Note that <Result>and <RQState/>elements are in this case empty; the result is not available at this moment, only ID of asynchronous request is returned (RQID). You can check the request result later by calling CheckRQResult method (see Chapter Asynchronous Request State for details).

5.3.3 Asynchronous Request State

This web service returns the status of an asynchronous request that is being processed in the Damas system. The asynchronous request is identified by request id which can be obtained by calling RunAsynchrous method (see Chapter Asynchronous Request).

5.3.3.1 SOAP CheckRQResult

The SOAP request format for downloading a status of the asynchronous request from Damas

```
POST /wse/DamasService.asmx HTTP/1.1
Host: procedure.terna.it
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: http://www.terna.it/damas/wse/CheckRQResult

<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soap:Header>
      <!-- WSS Security Header -->
   </soap:Header>
   <soap:Body>
      <CheckRQResult xmlns="http://www.terna.it/damas/wse">
         <RQID>RQID</RQID>
      </CheckRQResult>
   </soap:Body>
</soap:Envelope>
```

For details of the WSS Security, Header, see Chapter Web service security. Note that highlighted item RQID must be replaced with ID of the existing asynchronous request.

5.3.3.2 SOAP CheckRQResultResponse

The SOAP response format with status of the asynchronous request returned from Damas

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
```

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://sche-
mas.xmlsoap.org/soap/envelope/" xmlns:wsa="http://sche-
mas.xmlsoap.org/ws/2004/03/addressing" >
  <soap:Header>
    <!-- WSS Security Header -->
  </soap:Header>
  <soap:Body>
    <CheckRQResultResponse xmlns="http://www.terna.it/damas/wse">
      <Output>
        <RQID>RQID</RQID>
        <Result>resultXML</Result>
        <RQState>
          <Code>RQState_Code</Code>
          <Description>RQState_Description</Description>
        </RQState>
      </Output>
    </CheckRQResultResponse>
  </soap:Body>
</soap:Envelope>
```

For details of the WSS Security, Header, see Chapter Web service security. Element `<RQState>` contains information about state of the asynchronous request. Following table contains overview of the possible request states:

Code	Description	Note
REGISTERED	Asynchronous request is registered for execution.	<Result> element is empty; you should check request state later.
COMPLETED	Asynchronous request is completed.	<Result> element is filled with result of asynchronous request.
RUNNING	Asynchronous request is not completed.	<Result> element is empty; you should check request state later.
ERROR	Error occurred while running asynchronous request.	Internal server error occurred; in this case you should contact the system administrator.

5.3.4 Current Date and Time

This web service returns current system date and time that is important for automatic operations carried out by system. This service is also accessible via the RunSynchrous web service as data flow with id “GETDATETIME”.

5.3.4.1 SOAP GetActualDateTime

The SOAP request format for downloading current date and time from Damas

```
POST /wse/DamasService.asmx HTTP/1.1
Host: procedure.terna.it
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction:“http://www.terna.it/damas/wse/GetActualDateTime"
```

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://sche-
mas.xmlsoap.org/soap/envelope/" xmlns:wsa="http://sche-
mas.xmlsoap.org/ws/2004/03/addressing" >
  <soap:Header>
    <!-- WSS Security Header -->
  </soap:Header>
  <soap:Body>
    <GetActualDateTime xmlns="http://www.terna.it/damas/wse" />
  </soap:Body>
</soap:Envelope>
```

5.3.4.2 SOAP GetActualDateTime Response

The SOAP response format with current date and time returned from Damas

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://sche-
mas.xmlsoap.org/soap/envelope/" xmlns:wsa="http://sche-
mas.xmlsoap.org/ws/2004/03/addressing" >
  <soap:Header>
    <!-- WSS Security Header -->
  </soap:Header>
  <soap:Body>
    <GetActualDateTimeResponse xmlns="http://www.terna.it/damas/wse">
      <Output>
        <RQID>-1</RQID>
        <Result>resultXML</Result>
        <RQState/>
      </Output>
    </GetActualDateTimeResponse>
  </soap:Body>
</soap:Envelope>
```

6. Web Service Security

6.1 General Description

6.1.1 Security

Secure communication is integral part of the securing your distributed application to protect sensitive data, including credentials, passed to and from your application. Applications sharing sensitive business data among several different organizations across the Internet bring up challenges associated with robust mechanism for keeping confidentiality of the exchanged information based on world-wide encryption standards.

6.2 Damas Security Model

The Damas security model is built upon a system of user accounts. For each user who wants to use system services, a user account must be created first, with the username and password security elements assigned.

The Damas web service interface security is implemented in accordance with the Web Services Security standard (see http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss). Based on this standard, the following security issues are addressed: transferring credentials (username and password) in SOAP requirements.

6.3 SOAP Request Preparation

In addition to web service input parameters, the SOAP request also includes authentication data of the Damas user account and the digital signature of sent data.

6.3.1 SOAP Request Description

The SOAP request format with user authentication information and digital signature:

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-secext-1.0.xsd" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soap:Header>
      <wsa:Action>damas_soap_method</wsa:Action>
      <wsa:To>http_addressing_url</wsa:To>
      <wsse:Security soap:mustUnderstand="1">
         <wsse:UsernameToken wsu:Id="username_token_id">
            <wsse:Username>username</wsse:Username>
            <wsse:Password Type="http://docs.oasis-
open.org/wss/2004/01/oasis- 200401-wss-username-token-profile-
1.0#PasswordText">password_hash</wsse:Password>
```

```
            <wsse:Nonce>nonce_value</wsse:Nonce>
            <wsu:Created>utc_datetime</wsu:Created>
         </wsse:UsernameToken>
      </wsse:Security>
   </soap:Header>
   <soap:Body wsu:Id="soap_body_id">
      <RunSynchrous xmlns="http://www.terna.it/damas/wse">
         <!-- Input parameters comes here -->
      </RunSynchrous>
   </soap:Body>
</soap:Envelope>
```

6.3.1.1 Description of `<wsa:To>` element

This element contains addressing point in Damas infrastructure.

XML Element	Description
To	Testing environment: https://damas-test.terna.it/DamasService.svc?singlewsdl Production environment https://damas.terna.it/DamasService.svc?singlewsdl

6.3.1.2 Description of `<wsa:Action>` element

This element defines the SOAP action name. It contains name of the SOAP method – depends whether the synchronous or asynchronous mode is used.

XML Element	Description
Action	http://www.terna.it/damas/wse/RunSynchrous http://www.terna.it/damas/wse/RunAsynchrous

6.3.1.3 Description of `<wsse:Security>` element

According to the WSS all security tokens and signatures are included in the wsse:Security element. This element is part of the SOAP header and consists of the following items:

User authentication information (username and password)

Notes: Digital signing of SOAP messages is not enabled at TERNA implementation

6.3.1.4 Description of `<wsse:UsernameToken>` element

This element contains username and password assigned to the relevant Damas user account.

XML Element	Description
Username	Login name for Damas system user account
Password	Password for Damas system user account. Password is not transferred directly, but rather its MD5 hash is transferred encoded in BASE64 format.
Password/@Type	Type of used UsernameToken; must be always “http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText”.

Nonce	This element must be filled with a random value. This value is used as countermeasure against replay attacks. Server maintains a cache of used nonces. When client tries to use the same nonce more than once, the server will raise SOAP fault. Nonce value must be BASE64 encoded.
Created	Date and time of Nonce creation. Nonces older than 10 minutes are automatically rejected. Date and time must be in UTC time. (Example: 2005-05-30T09:30:10Z).

6.4 SOAP Response Parsing

The server SOAP response is signed using digital certificate. Unlike in the SOAP request, no authentication data of Damas user account are transferred in this case.

6.4.1 SOAP Response Description

The SOAP response format with a digital signature:

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://sche-
mas.xmlsoap.org/soap/envelope/"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-secext-1.0.xsd"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
  <soap:Header>
    <wsse:Security soap:mustUnderstand="1">
    </wsse:Security>
  </soap:Header>
  <soap:Body wsu:Id="soap_body_id">
    <RunSynchrousResponse xmlns="http://www.terna.it/damas/wse">
      <!-- Output parameters come here -->
    </RunSynchrousResponse>
  </soap:Body>
</soap:Envelope>
```

Provided example is similar to request example from Chapter SOAP Request Description. For detailed description of each element, see Chapter SOAP Request Description.

The SOAP response differs from the SOAP request in following points:

Credentials are not sent back to client – element `<wsse:UsernameToken>` is missing.

7. Data Flows

This chapter provides description of all data flows for downloading and uploading data from/to Damas. The summary overview of the input and output data flows is presented in first part. Second part of the chapter deals with detailed definition of the data flows. Each data flow is described in detail, with input and output parameters explained.

7.1 List of Data Flows

Following table shows various data flows that are available for uploading data into Damas. User types allowed to use data flows are indicated in “User” column in overview below.

Sending data to the Damas system			
Data Flows	**FID**	**Description**	**User**
Sending nominations	DMSWS_RD_IN	Sending long-term or daily nominations for a business day, a border direction and a local operator.	LO

The following tables show various data flows available for downloading data from the Damas system.

Downloading data from the Damas system			
Data Flows	**FID**	**Description**	**User**
Downloading Nomination	DMSWS_RD_OUT	Downloading nominations for a business day, a border and a local operator. Data are not aggregated. Input parameters: • Business day • ControlArea1 (TERNA) • ControlArea2 (neighbour TSO) • EIC – possibility to select all	LO, TERNA
Downloading actual date and time	GETDATETIME	Downloading the current date and time in the Damas system.	TSO, LO, TERNA

7.2 Data Flows for Sending Data

Delivery of values to the Damas system is automatically confirmed and the sender is immediately informed about the processing result. For all data flows, the user receives an Acknowledgement Message confirming the data delivery.

7.2.1 Sending Nominations

Sending nominations to the Damas system

7.2.1.1 Description

This data flow enables sending of the XML Schedule Message with nominations for a business day, a local operator and a border direction. The resolution PT60M is used for schedules and the resolution PT60M. The XML file contains time series with 24 (23, 25) values for every business day, a local operator, border direction and schedule type.

Local operators are allowed to submit nominations on boder directions with explicit nominations. They can submit nomination on one or more counterparts according to market rules on particular border direction and capacity type. Nominations on yearly, monthly, periodic and reserved capacity are allowed before gate closure for long term schedules. Nominations on daily capacity are allowed after publishing daily auction result and before gate closure for daily schedules.

Long-term and Daily auction for Delivery Day before gate closure on respective Border.

7.2.1.2 Input Parameters

List of the input parameters:

Name	Type	Description	Note
FID		DMSWS_RD_IN	
XML	XmlParam	XML Schedule Message with schedule time series.	See Scheduled Message definition. It is possible to send schedules for multiple business days, local operators and border directions.

7.2.1.3 Output Parameters

User receives an Acknowledgement message as a response confirming the data delivery and describing processing results.

7.3 Data Flows for Data Download

7.3.1 Downloading Nominations

Downloading nominations for a given business day, a local operator, a border from the Damas system.

7.3.1.1 Description

Downloading the XML Schedule Message with nominatins for a given business day, a local operator, all border directions and all capacity types. The resolution PT60M is used. The XML file can contain multiple time series with 24 (23, 25) values for each business day, local operator, border direction and capacity.

The web service is accessible to local operators. A local operator is entitled to download only schedules nominated by the local operator.

7.3.1.2 Input Parameters

List of the input parameters:

Name	Type	Description	Note
FID		DMSWS_RD_OUT	
Date	DateParam	Business day in format YYYY-MM-DD	Central European Time
ControlArea1	StringParam	Home Control Area It must containt TERNA control area EIC (10YIT-GRTN-----B)	
ControlArea2	StringParam	Control Area defining the border Example: 10YAT-APG------L	See List of Control Areas.
Subject	StringParam	EIC of Local Operator	Nominations of all local operators will be downloaded when the Subject parameter is empty (this feature is available only for TERNA manager or TSO.

An example of input parameters:

```
<Input>
   <FID>DMSWS_RD_OUT</FID>
   <Parameters>
      <DateParam Name="Date">2005-08-01</DateParam>
      <StringParam Name="ControlArea1">10YIT-GRTN-----B</StringParam>
      <StringParam Name="ControlArea2">10YAT-APG------L</StringParam>
      <StringParam Name="Subject">26X-TLO-00000001</StringParam>
   </Parameters>
</Input>
```

7.3.1.3 Output Parameters

Capacity nominations are received in the XML Schedule Message.

7.3.2 Downloading Actual Date and Time

Downloading the current date and time from Damas

7.3.2.1 Description

Downloading current date and time from Damas; this web service is accessible to all users.

7.3.2.2 Input Parameters

List of the input parameters:

Name	Type	Description	Note
FID		GETDATETIME	

Example of the input parameters:

```
<Input>
  <FID>GETDATETIME</FID>
  <Parameters/>
</Input>
```

7.3.2.3 Output Parameters

The current date and time in Damas is received. Time is in form of Coordinated Universal Time (UTC).

8. XSD Schemas

This part of the document provides detailed technical description of all XSD schemas used in Damas. Each XSD description contains model of the XSD schema structure, detailed description of the schema and explanation of the meaning of all XSD elements. Examples of the XML files are attached to the document.

8.1 List of XSD Schemas

Name of XSD Schema	Description	Web Services
Schedule Message	The schedule message is mainly used for downloading or sending daily and long-term schedules.	DMSWS_RD_OUT DMSWS_RD_IN

8.2 Description of XSD Schemas

8.2.1 Schedule Message

8.2.1.1 Schedule Message Description

The Schedule Message is used for uploading or downloading schedules to/from the Damas system.

All tags in the XML file are populated with data in accordance with ETSO rules. The PT60M resolution is used for nominations.

Primary information about the message, such as the message identification and version, or identification of its sender and recipient, are stored in the message header. Each schedule is represented by the ScheduleTimeSeries element that contains all necessary information about the schedule, such as its unique identification number, the version number, identification of the source and destination Control Areas, identification of the Local Operator and its Counterpart, Capacity Agreement Identification and the schedule type.

The Period element defines the business day for which the schedule is entered and its time resolution. Values for each hour of the business day are listed in the Interval element. The XML file with the PT60M resolution contains schedules with 24 values (23 when switching from winter time to summer time, 25 when switching from summer time to winter time). In case of 25 values, the additional hour is inserted into the right position inside the diagram and the remaining hourly values are shifted up.

The special elements introduced in ESS v3r0 (Domain, Subject Party, Subject Role, Matching Period) are not used.

8.2.1.2 Specifications of Schedule Message Elements

The list of XML elements included in the ScheduleMessage element:

Element	Description	Values	Applicability

Message Identification	Identification of the message.	The recommended naming convention is: <BusinessDay>_<Party ID> Business Day – First day of the time interval for which the schedule is submitted. Party ID – Local operator EIC code.	Mandatory
Message Version	The version number assigned to the message. This value must be incremented after each edit operation.	Non-signed integer value.	Mandatory
Message Type	The coded type of the message being sent.	A01 (Balance responsible schedule)	Mandatory
Process Type	The nature of the process that the message is directed at.	A01 (Day ahead)	Mandatory
Schedule Classification Type	The type that is used to classify the schedule by aggregation or classification.	A01 (Exchange)	Mandatory
Sender Identification	Identification of the party sending the message.	A01 (ETSO) coding scheme.	Mandatory
Sender Role	Identification of the role played by the sender.	A04 (System operator) – TSO – For downloading schedules. A08 (Balance responsible party) – Local operator – For submitting schedules.	Mandatory
Receiver Identification	Identification of the party receiving the message.	EIC code of the receiver. A01 (ETSO) coding scheme.	Mandatory
Receiver Role	Identification of the role played by the receiver.	A04 (System operator) – TSO – For submitting schedules. A08 (Balance responsible party) – Local operator – For downloading schedules.	Mandatory
Message Date Time	Date and time of transmission of the message.	UTC coding. Format: YYYY-MM-DDTHH:MM:SSZ	Mandatory

Schedule Time Interval	The beginning and the ending date and time of the period covered by the message. One business day for CAS, CBS files. UTC coding. Format: YYYY-MM-DDTHH:MMZ/ YYYY-MM-DDTHH:MMZ	The interval contains one business day. On all Italian borders, the CET time zone is used. The schedule time interval must always be from 22:00 to 22:00 UTC in summer time and 23:00 to 23:00 UTC in wintertime.	Mandatory
Schedule Time Series	Time series containing schedule.		Optional

The list of XML elements included in the ScheduleTimeSeres element:

Element	Description	Values	Applicability
Senders Time Series Identification	The sender's identification of the time series instance. A unique identification within the message assigned by the sender.	In Damas it represents an identification string of schedule. For uploading new schedules no value or "-1" value can be used (SendersTimeSeriesIdentification v="" or SendersTimeSeriesIdentification v="-1"), system will asign the unique ID automatically	Mandatory
Senders Time Series Version	The version number assigned to the time series.	Non-signed integer value. Always the same version as Message Version.	Mandatory
Business Type	Identifies the trading nature of an energy product.	A03 (External trade explicit capacity) – Submitting schedules.	Mandatory
Product	Identification of an energy product.	8716867000016 (Active power).	Mandatory
Object Aggregation	This identifies the extent to which the object is aggregated.	A01 (Area) – Submitting schedules.	Mandatory
In Area	The identification of the destination area of the border direction.	EIC code of destination area. A01 (ETSO) coding scheme.	Mandatory

Out Area	The identification of the source area of the border direction.	EIC code of source area. A01 (ETSO) coding scheme.	Mandatory
Metering Point Identification	The identification of the location where one or more products are metered.	String.	Not used.
In Party	The identification of the destination party.	EIC code of destination auction participant. A01 (ETSO) coding scheme. If InArea is TERNA, the Local Operator is located in this element and its Counterpart is located in the OutParty element.	Mandatory
Out Party	The identification of the source party.	EIC code of source auction participant. A01 (ETSO) coding scheme. If OutArea is TERNA, the Local Operator is located in this element and its Counterpart is located in the InParty element.	Mandatory
Capacity Contract Type	The contract type defines the conditions under which the capacity was allocated and handled.	Values used for nominations: A01 (Daily) A03 (Monthly) A04 (Yearly) Z01 (Reserved) A06 (Periodic) A07 (Intraday) Z06 (ML Cagno) Z07 (ML Tirano)	Mandatory

Capacity Agreement Identification	The identification of an agreement for the allocation of capacity to a party.	The CapacityAgreementIdentifíi cation element is dependent it might be requested based on the business requirements for given border direction and Capacity Contract Type. Please refer to other respective documents defining the rules for particular border direction	Mandatory
Measurement Unit	The unit of measure applied to the quantities in which the time series is expressed.	MAW (Mega watt).	Mandatory
Reason	Reasons for the modifications performed on the time series.		Not used.
Period			

* If the edited schedule has been uploaded by the web services, its SendersTimeSeriesIdentification is returned in the AcknowledgementMessage. If it has been uploaded via Schedule form, its identification number is part of the message sent to Damas menu. You can find the schedule number to edit in the Schedule Overview form in the Schedule Detail part as well.

The list of XML elements included in the Period element:

Element	Description	Values	Applicability
Time Interval	The start and the end date and time of the time interval of the period. UTC coding. Format: YYYY-MM-DDTHH:MMZ/ YYYY-MM-DDTHH:MMZ	The same rules as for Schedule Time Interval.	Mandatory
Resolution	The resolution defining the number of periods that the time interval is divided into.	PT60M – Submitting schedules.	Mandatory
Interval	Time period interval.		Mandatory

The list of XML elements included in the Interval element:

Element	Description	Values	Applicability

Pos	The relative position of a period within the time interval.	Non-signed integer value.	Mandatory
Qty	The quantity of the product scheduled for the position within the time interval.	Non-signed integer value.	Mandatory

Schedule Message Examples

An example of Schedule Message for daily schedule (submitted by local operator for 1.6.2008 and for border direction TERNA-SWG):

```
<?xml version="1.0" encoding="UTF-8"?>
<ScheduleMessage xmlns="http://www.terna.it/damas/xsd/ScheduleMes-
sage.xsd" DtdVersion="2" DtdRelease="3">
   <MessageIdentification v="20080601_ 26X-TLO-00000001" />
   <MessageVersion v="1" />
   <MessageType v="A01" />
   <ProcessType v="A01" />
   <ScheduleClassificationType v="A01" />
   <SenderIdentification v=" 26X-TLO-00000001" codingScheme="A01" />
   <SenderRole v="A08" />
   <ReceiverIdentification v="10X1001A1001A345" codingScheme="A01" />
   <ReceiverRole v="A04" />
   <MessageDateTime v="2008-01-07T15:05:31Z" />
   <ScheduleTimeInterval v="2008-05-31T22:00Z/2008-06-01T22:00Z" />
   <ScheduleTimeSeries>
      <SendersTimeSeriesIdentification v="1000653" />
      <SendersTimeSeriesVersion v="1" />
      <BusinessType v="A03" />
      <Product v="8716867000016" />
      <ObjectAggregation v="A01" />
      <InArea v="10YCH-SWISSGRIDZ" codingScheme="A01" />
      <OutArea v="10YIT-GRTN-----B" codingScheme="A01" />
      <InParty v=" 26X-TLO-00000001" codingScheme="A01" />
      <OutParty v=" 26X-TLO-00000001" codingScheme="A01" />
      <UpvUcv v="UCV_SWGDTESTLO1O" codingScheme="A01" />
      <CapacityContractType v="A01" />
      <CapacityAgreementIdentification v="26XTLOCAI1" />
      <MeasurementUnit v="MAW" />
      <Period>
         <TimeInterval v="2008-05-31T22:00Z/2008-06-01T22:00Z" />
         <Resolution v="PT60M" />
         <Interval>
            <Pos v="1" />
            <Qty v="2" />
         </Interval>
         …
         <Interval>
            <Pos v="24" />
```

```
            <Qty v="2" />
          </Interval>
        </Period>
    </ScheduleTimeSeries>
</ScheduleMessage>
```

8.3 Code List

8.3.1 Role

Code list for role element is defined according to ESS CodeList v4r0. Following codes are used in the Damas system:

Code	Role Name	Description
A04	System operator	This code should be used by TSO operators.
A07	Transmission capacity allocator	This code should be used by auction office.
A08	Balance responsible party	This code should be used by local operators.
A29	Capacity Trader	This code should be used by auction participants.

8.3.2 Product

Code list for product element is defined according to ESS CodeList v4r0. Following codes are used in the Damas system:

Code	Product Name	Description
8716867000016	Active power	This code is used for schedules.

8.3.3 Capacity Contract Type

Code list for capacity contract type element is extension of ESS CodeList v4r0. Following codes are used in the Damas system:

Code	Contract Type Name	Description
A01	Daily	This code should be used for capacity allocated by daily auction or daily transmission procedures.
A03	Monthly	This code should be used for capacity allocated by monthly auction or monthly transmission procedures.
A04	Yearly	This code should be used for capacity allocated by yearly auction or yearly transmission procedures.
A07	Intraday	This code should be used for capacity allocated by intraday auction.

Z01	Reserved	
A06	Periodic	This code should be used for capacity allocated by monthly and yearly auction or monthly and yearly transmission procedures.
Z06	ML Cagno	This code should be used for capacity allocated and handled by long term trades on Merchant Line.
Z07	ML Tirano	This code should be used for capacity allocated and handled by long term trades on Merchant Line.

8.3.4 Measurement Unit

Code list for measurement unit element is defined according to ESS CodeList v4r0. Following codes are used in the Damas system:

Code	Measurement Unit Name	Description
MAW	Mega watt	A unit of bulk power flow, which can be defined as the rate of energy transfer/consumption where a current of 1000 amperes flows due to a potential of 1000 volts at unity power factor expressed in millions of a watt.

8.3.5 Message Type

Code list for message type is extension of ESS CodeList v4r0. Following codes are used in the Damas system:

Code	Message Type Name	Description
A01	Balance responsible schedule	A schedule that has been prepared by a balance responsible party providing planned schedule information. This code should be used for schedules.
A02	Allocated capacity schedule	A schedule that has been prepared by a capacity allocator providing allocated capacity.
A13	Interconnection Capacity	Document for cross-border capacity exchanges. This code should be used for ATC files.

8.3.6 Process Type

Code list for process type is defined according to ESS CodeList v4r0. Following codes are used in the Damas system:

Code	Process Type Name	Description

A01	Day ahead	The information provided concerns a day ahead schedule.
A07	Capacity allocation	The information provided concerns the capacity allocation processs.

8.3.7 Classification Type

Code list for classification type is defined according to ESS CodeList v4r0. Following codes are used in the Damas system:

Code	Classification Type Name	Description
A01	Exchange type	The schedule is classified as providing the detailed trades between two entities (all external trades between two entities). It is used for data without aggregation.

8.3.8 Coding Scheme

Code list for coding scheme is defined according to ESS CodeList v4r0. Following codes are used in the Damas system:

Code	Coding Scheme Name	Description
A01	ETSO	The coding scheme for the preceding attribute is the ETSO Identification Coding Scheme (EIC), maintained by ETSO.

8.3.9 Business Type

Code list for business type is extension of ESS CodeList v4r0. Following codes are used in the Damas system:

Code	Coding Scheme Name	Description
A01	ETSO	The coding scheme for the preceding attribute is the ETSO Identification Coding Scheme (EIC), maintained by ETSO.

8.3.10 Object Aggregation

Code list for object aggregation is extension of ESS CodeList v4r0. Following codes are used in the Damas system:

Code	Role Name	Description
A01	Area	The object being described concerns an area.

A03	Party	The object being described concerns a party.

8.3.11 Currency

Code list for currency is defined according to ESS CodeList v4r0. Following codes are used in the Damas system:

Code	Currency Name	Description
EUR	EURO	The European legal tender.

9. FAQ – DAMAS WEB SERVICES

9.1 User cannot be authenticated

When the authentication of the user fails on the server, the request is not processed and the user is notified about rejection of the request. The response message is: The security token could not be authenticated or authorized.

In this situation we recommend to check whether the username and password are filled correctly and the account exists on the target Damas environment. This could be done by login to the web application Damas by used credentials. For more information related to authentication please see chapter *6.3 SOAP Request Preparation*.

9.2 Can I verify whether the password hash is correct?

For security reasons the password is not transferred as plain text directly, but rather it is MD5 hash encoded in BASE64 format. We have created a testing site where you can check and also convert the password into the demanding hash: Damas Hash Password Converter

9.3 Destination Unreachable

Destination Unreachable error is mostly caused by some invalid value in the soap:Envelope section of the request. In case of this error the response message also contains the error stack with invalid fields listed. For more details please see chapter: *6.3 SOAP Request Preparation*.

This error message could also be caused by using HTTP as transfer protocol instead of HTTPS.

9.4 Business checks

Before processing of the sent request all appropriate validations related to the Data Flows are performed, e.g. the state of the system for the requested action, user rights for the requested action, checking of mandatory values in the xml etc. If some validation fails the user should be notified in the response about the validation errors.

The standard scenario in this situation is to take these validations into account and adjust the requests according to system requirements.