RESTful Network API for FileTransfer

Candidate Version 1.0 – 17 Apr 2012

Open Mobile Alliance

OMA-TS-REST_NetAPI_FileTransfer-V1_0-20120417-C

Contents

Figures

Tables

1. Scope

This specification defines a RESTful API for File Transfer using HTTP protocol bindings.

2. References

2.1 Normative References

[Autho4API_10]	"Authorization Framework for Network APIs", Open Mobile Alliance™, OMA-ER-Autho4API-V1_0, URL: http://www.openmobilealliance.org/
[IETF_ACR_draft]	"The acr URI for anonymous users", S.Jakobsson, K.Smith, July 2011, URL: http://tools.ietf.org/html/draft-uri-acr-extension-03
[IR.79]	"Image Share Interoperability Specification", URL: http://www.gsmworld.com/documents/ireg/ir79_14.pdf
[OMA-SIMPLE_IM]	"Instant Messaging using SIMPLE", Version 1.0, Open Mobile Alliance™, OMA-TS-SIMPLE_IM-V1_0, URL:http://www.openmobilealliance.org/
[RC_API_RD]	APIs for Rich Communications Requirements, OMA-RD-RC_API-V1_0, Open Mobile Alliance, URL: http://www.openmobilealliance.org/
[REST_NetAPI_Common]	"Common definitions for RESTful Network APIs", Open Mobile Alliance™, OMA-TS-REST_NetAPI_Common-V1_0, URL:http://www.openmobilealliance.org/
[REST_NetAPI_NotificationChannel]	"RESTful Network API for Notification Channel", Open Mobile Alliance™, OMA-TS-REST_NetAPI_NotificationChannel-V1_0, URL: http://www.openmobilealliance.org/
[REST_SUP_FileTransfer]	"XML schema for the RESTful Network API for File Transfer", Open Mobile Alliance™, OMA-SUP-XSD-rest_netapi_FileTransfer-V1.0, URL:http://www.openmobilealliance.org/
[RFC2045]	"Multipurpose Internet Mail Extensions(MIME) Part One: Format of Internet Message Bodies", N. Freed, November 1996, URL: http://www.ietf.org/rfc/rfc2045.txt
[RFC2046]	"Multipurpose Internet Mail Extensions(MIME) Part Two: Media Types", N. Freed, November 1996, URL: http://www.ietf.org/rfc/rfc2046.txt
[RFC2119]	"Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997, URL:http://www.ietf.org/rfc/rfc2119.txt
[RFC2327]	"SDP: Session Description Protocol", M. Handley, April 1998, URL: http://www.ietf.org/rfc/rfc2327.txt
[RFC2392]	"Content-ID and Message-ID Uniform Resource Locators", E. Levinson, August 1998, URL: http://www.ietf.org/rfc/rfc2392.txt
[RFC2616]	"Hypertext Transfer Protocol -- HTTP/1.1", R. Fielding et. al, January 1999, URL:http://www.ietf.org/rfc/rfc2616.txt
[RFC3261]	"SIP: Session Initiation Protocol", J. Rosenberg et al., June 2002, URL: http://www.ietf.org/rfc/rfc3261.txt
[RFC3966]	"The tel URI for Telephone Numbers", H.Schulzrinne, December 2004, URL: http://www.ietf.org/rfc/rfc3966.txt
[RFC3986]	"Uniform Resource Identifier (URI): Generic Syntax", R. Fielding et. al, January 2005, URL:http://www.ietf.org/rfc/rfc3986.txt
[RFC4627]	"The application/json Media Type for JavaScript Object Notation (JSON)", D. Crockford, July 2006, URL: http://www.ietf.org/rfc/rfc4627.txt
[RFC4975]	"The Message Session Relay Protocol (MSRP)", B. Campbell et. al, September 2007, URL: http://www.ietf.org/rfc/rfc4975.txt
[RFC5547]	"Session Description Protocol (SDP) Offer/Answer Mechanism to Enable File Transfer", M. Garcia-Martin, May 2009, URL: http://www.ietf.org/rfc/rfc5547.txt
[SCRRULES]	"SCR Rules and Procedures", Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:http://www.openmobilealliance.org/
[W3C_URLENC]	HTML 4.01 Specification, Section 17.13.4 Form content types, The World Wide Web Consortium, URL: http://www.w3.org/TR/html401/interact/forms.html#h-17.13.4.1
[XMLSchema1]	W3C Recommendation, XML Schema Part 1: Structures Second Edition, URL:

http://www.w3.org/TR/xmlschema-1/

[XMLSchema2] W3C Recommendation, XML Schema Part 2: Datatypes Second Edition, URL: http://www.w3.org/TR/xmlschema-2/

2.2 Informative References

[OMADICT] "Dictionary for OMA Specifications", Version 2.8, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_8, URL:http://www.openmobilealliance.org/

[REST_WP] "Guidelines for RESTful Network APIs", Open Mobile Alliance™, OMA-WP-Guidelines_for_RESTful_Network_APIs, URL:http://www.openmobilealliance.org/

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

For the purpose of this TS, all definitions from the OMA Dictionary apply [OMADICT].

Originator	The party that initiates a file transfer session.
Receiver	The party that is invited to a file transfer session to receive files.
Client-side Notification URL	An HTTP URL exposed by a client, on which it is capable of receiving notifications and that can be used by the client when subscribing to notifications.
Long Polling	A variation of the traditional polling technique, where the server does not reply to a request unless a particular event, status or timeout has occurred. Once the server has sent a response, it closes the connection, and typically the client immediately sends a new request. This allows the emulation of an information push from a server to a client.
Notification Channel	A channel created on the request of the client and used to deliver notifications from a server to a client. The channel is represented as a resource and provides means for the server to post notifications and for the client to receive them via specified delivery mechanisms. For example in the case of Long Polling the channel resource is defined by a pair of URLs. One of the URLs is used by the client as a call-back URL when subscribing for notifications. The other URL is used by the client to retrieve notifications from the Notification Server.
Notification Server	A server that is capable of creating and maintaining Notification Channels.
Server-side Notification URL	An HTTP URL exposed by a Notification Server, that identifies a Notification Channel and that can be used by a client when subscribing to notifications.

3.3 Abbreviations

ACR	Anonymous Customer Reference
API	Application Programming Interface
GIF	Graphics Interchange Format
HTTP	HyperText Transfer Protocol
JPEG	Joint Photographic Expert Group
JSON	JavaScript Object Notation
MIME	Multipurpose Internet Mail Extensions
OMA	Open Mobile Alliance
REST	REpresentational State Transfer
SCR	Static Conformance Requirements
SDP	Session Description Protocol
SIP	Session Initiation Protocol
TS	Technical Specification

URI	Uniform Resource Identifier
URL	Uniform Resource Locator
XML	eXtensible Markup Language
XSD	XML Schema Definition

4. Introduction

The Technical Specification of the RESTful Network API for File Transfer contains HTTP protocol bindings based on the requirements for File Transfer defined in [RC_API_RD], using the REST architectural style. The specification provides resource definitions, the HTTP verbs applicable for each of these resources, and the element data structures, as well as support material including flow diagrams and examples using the various supported message body formats (i.e. XML, JSON, and application/x-www-form-urlencoded).

4.1 Version 1.0

Version 1.0 of this specification supports the following operations:

- Managing subscriptions to file transfer-related notifications
- Managing file transfer sessions
- Sending files
- Receiving notifications about file transfer session invitations
- Receiving notifications about file transfer session events
- Receiving notifications about file content link
- Receiving notifications about Receiver acceptance

In addition, this specification provides:

- Support for scope values used with authorization framework defined in [Autho4API_10]
- Support for Anonymous Customer Reference (ACR) as an end user identifier
- Support for “acr:Authorization” as a reserved keyword in a resource URL variable that identifies an end user

5. File Transfer API definition

This section is organized to support a comprehensive understanding of the File Transfer API design. It specifies the definition of all resources, definition of all data structures, and definitions of all operations permitted on the specified resources.

Common data types, naming conventions, fault definitions and namespaces are defined in [REST_NetAPI_Common].

The remainder of this document is structured as follows:

Section 5 starts with a diagram representing the resources hierarchy, followed by a table listing all the resources (and their URL) used by this API, along with the data structure and the supported HTTP verbs (section 5.1). What follows are the data structures (section 5.2). A sample of typical use cases is included in section 5.3, described as high level flow diagrams.

Section 6 contains the detailed specification for each of the resources. Each such subsection defines the resource, the request URL variables that are common for all HTTP commands, the possible HTTP response codes, and the supported HTTP verbs. For each supported HTTP verb, a description of the functionality is provided, along with an example of a request and an example of a response. For each unsupported HTTP verb, the returned HTTP error status is specified, as well as what should be returned in the Allow header.

All examples in section 6 use XML as the format for the message body. Application/x-www-form-urlencoded examples are provided in Appendix C, while JSON examples are provided in Appendix D. Appendix B provides the Static Conformance Requirements (SCR).

0 provides the operations mapping to a pre-existing baseline specification, where applicable.

Appendix F provides a list of all lightweight resources, where applicable.

Appendix A defines authorization aspects to control access to the resources defined in this specification.

Note: Throughout this document client and application can be used interchangeably.

5.1 Resources Summary

This section summarizes all the resources used by the RESTful Network API for File Transfer.

The "apiVersion" URL variable SHALL have the value "v1" to indicate that the API corresponds to this version of the specification. See [REST_NetAPI_Common] which specifies the semantics of this variable.

The figure below visualizes the resource structure defined by this specification. Note that those nodes in the resource tree which have associated HTTP methods defined in this specification are depicted by solid boxes.

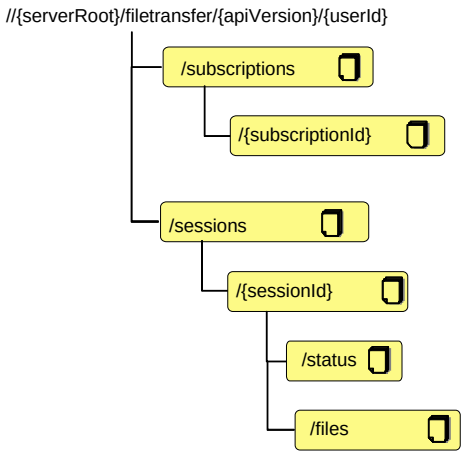

Figure 1 Resource structure defined by this specification

The following tables give a detailed overview of the resources defined in this specification, the data type of their representation and the allowed HTTP methods.

Purpose: To allow client to manage file transfer notifications subscriptions

Resource	URL Base URL: http://{serverRoot}/filetransfer/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
All subscriptions to file transfer notifications	/subscriptions	FileTransferSubscriptionList (used for GET) FileTransferSubscription (used for POST) common:ResourceReference (OPTIONAL alternative for POST response)	Read all active file transfer subscriptions related to a user	no	Create new subscription for file transfer notification of an user	no
Individual subscription to file transfer notifications	/subscriptions/{subscriptionId}	FileTransferSubscription	Retrieves an active file transfer subscription related to a user	no	no	Cancel subscription and stop corresponding notifications

Purpose: To allow client to manage 1-1 file transfer sessions

Resource	URL Base URL: http://{serverRoot}/filetransfer/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
All 1-1 file transfer sessions	/sessions	FileTransferSessionInformation (used for POST) common:ResourceReference (OPTIONAL alternative for POST response)	no	no	Create a new 1-1 file transfer session	no

Resource	URL Base URL: http://{serverRoot}/filetransfer/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Individual 1-1 file transfer session	/sessions/{sessionId}	FileTransferSessionInformation	Retrieve file transfer session information	no	no	Cancel invitation (Originator) Decline invitation (Receiver) Terminate session
Session status	/sessions/{sessionId}/status	ReceiverSessionStatus	no	no	Accept 1-1 file transfer invitation	no

Purpose: To allow client to manage sending 1-1 file transfer files

Resource	URL Base URL: http://{serverRoot}/filetransfer/{apiVersion}/{userId}/sessions/{sessionId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Files in a 1-1 file transfer session	/files	FileInformation (used for POST) common:ResourceReference (OPTIONAL alternative for POST response)	no	no	Send file	no

Purpose: To allow server to notify client about file transfer session status and file links

Resource	URL	Data Structures	HTTP verbs

	Base URL: <Specified by the client>		GET	PUT	POST	DELETE
Client notification about session invitation	Specified by client when subscription is created or provisioned	SessionInvitationNotification	no	no	Notify the client about the incoming file transfer invitation	no
Client notification about file transfer events	Specified by client when subscription is created or provisioned	FileTransferEventNotification	no	no	Notify the client about file transfer events	no
Client notification about file content link	Specified by client when subscription is created or provisioned	FileNotification	no	no	Notify the client about file content link for subsequent retrieval	no
Client notification about Receiver acceptance	Specified by client when subscription is created or provisioned	ReceiverAcceptanceNotification	no	no	Notify the client about Receiver accepted the invitation	no
Client notification about subscription cancellation	Specified by client when subscription is created or provisioned	SubscriptionCancellationNotification	no	no	Notify client that a subscription has been cancelled (e.g. expired)	no

5.2 Data Types

5.2.1 XML Namespaces

The XML namespace for the File Transfer API data types is:

urn:oma:xml:rest:netapi:filetransfer:1

The 'xsd' namespace prefix is used in the present document to refer to the XML Schema data types defined in XML Schema [XMLSchema1, XMLSchema2]. The 'common' namespace prefix is used in the present document to refer to the data types defined in [REST_NetAPI_Common]. The use of namespace prefixes such as 'xsd' is not semantically significant.

The XML schema for the data structures defined in the section below is given in [REST_SUP_FileTransfer].

5.2.2 Structures

The subsections of this section define the data structures used in the File Transfer API.

Some of the structures can be instantiated as so-called root elements.

For structures that contain elements which describe a user identifier, the statements in section 6 regarding 'tel', 'sip' and 'acr' URI schemes apply.

5.2.2.1 Type: FileTransferSessionInformation

This type represents information about a File Transfer Session.

Element	Type	Optional	Description
originatorAddress	xsd:anyURI	No	Address of the Originator of this file transfer session (e.g. 'sip' URI, 'tel' URI, 'acr' URI)
originatorName	xsd:string	Yes	Name of the Originator of this file transfer session
receiverAddress	xsd:anyURI	No	Address of the Receiver of this file transfer session(e.g. 'sip' URI, 'tel' URI, 'acr' URI)
receiverName	xsd:string	Yes	Name of the Receiver of this file transfer session
status	ReceiverStatus	Yes	Connection status of the Receiver Set by the server. SHALL NOT be present in request bodies during resource creation.
fileInformation	FileInformation[1..unbounded]	No	A set of file's attributes. Note: According to [RC API RD],

			there is only one file. But in SIMPLE IM File transfer [OMA-SIMPLE_IM], there may be multiple files.
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This field SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids re-sending the message in such situations. In case the field is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
resourceURL	xsd:anyURI	Yes	Self referring URL. The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named fileTransferSessionInformation of type FileTransferSessionInformation is allowed in request and/or response bodies.

Note that the clientCorrelator is used for purposes of error recovery as specified in [REST_NetAPI_Common], and internal client purposes. The server is NOT REQUIRED to use the clientCorrelator value in any form in the creation of the URL of the resource. The document [REST_NetAPI_Common] provides a recommendation regarding the generation of the value of this field.

5.2.2.2 Type: FileInformation

This type represents a set of attributes of a file.

Element	Type	Optional	Description
fileSelector	FileSelector	No	A tuple of file attributes that the SDP offerer includes in the SDP in order to select a file at the SDP

			answerer. File name, size, type and hash as specified in [IR.79] & [RFC5547]
fileDisposition	FileDisposition	Yes	It is used by the file sender to indicate a preferred disposition of the file. To indicate that a file should be automatically rendered, the value is “Render”. To indicate that a file should not be automatically rendered, the value is “Attachment”. See [RFC5547]. Default value is “Attachment”.
fileDescription	xsd:string	Yes	Human-readable short description of the file (corresponding to ‘i=’ line in SDP) which could be set by the Originator. See [IR.79] & [RFC2327].
fileDate	FileDate	Yes	The dates on which the file was created, modified, or last read as specified in [RFC5547].
fileIcon	xsd:anyURI	Yes	It is useful with certain file types such as images. It allows the file sender to include a pointer to a body that includes a small preview icon representing the contents of the file to be transferred, the file Receiver can use to determine whether it wants to receive such file. The ’file-icon’ contains a Content-ID URL, [RFC2392] pointing to an additional body that contains the actual icon in a MIME multipart/related body. See [RFC5547] &requirement of RCAPI-RCFT-001 in [RC API RD].
fileURL	xsd:anyURI	Yes	The URL link to actual file content. When this field is used in POST operation during resource creation of Create a new 1-1 file transfer session, it is the external file repository URL set by Originator. If it is present, it indicates that there is no file content included in the request operation.

			If it is not present, it indicates that the actual file content is included in the HTTP request body during resource creation of Create a new 1-1 file transfer session or Send file. File content can be represented as multipart/form-data entity bodies, where the first entry of the form are the root fields and the second entry of the form are the file content. When this field is used in file content notification, it is a URL link to actual file content where the client can download the file. This field can be used in response of Create file transfer session, Retrieve file transfer session information, Send file. It is the file content URL set by server. This field is not used in session invitation notifications.
resourceURL	xsd:anyURI	Yes	Self referring URL. The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named fileInformation of type FileInformation is allowed in request and/or response bodies.

5.2.2.3 Type: FileSelector

This type represents the basic information of a file such as name.

Element	Type	Optional	Description
name	xsd:string	No	The name of the file. See [IR.79] & [RFC5547]. Note: in a file transfer session, the file name is unique.
type	xsd:string	No	The MIME type of the file. It is concatenated by type, "/" and subtype. See [IR.79] & [RFC5547].

size	xsd: unsignedLong	Yes	The size of the file in octets. See [IR.79] & [RFC5547].
hash	HashInformation	Yes	The file hash information including hash algorithm and hash value. See [IR.79] & [RFC5547].

5.2.2.4 Type: FileDate

This type represents the dates on which the file was created, modified, or last read, it MAY contain any combination of "cDate", "mDate" and "rDate".

Element	Type	Optional	Description
cDate	xsd:dateTime	Yes	The date on which the file was last created. See [RFC5547].
mDate	xsd:dateTime	Yes	The date on which the file was last modified. See [RFC5547].
rDate	xsd:dateTime	Yes	The date on which the file was last read. See [RFC5547].

5.2.2.5 Type: HashInformation

This type represents the file hash information.

Element	Type	Optional	Description
algorithm	xsd:string	No	The hash algorithm used (only "sha-1" currently supported). See [RFC5547].
value	xsd:hexBinary	No	The hash value of the file. See [RFC5547].

5.2.2.6 Type: SessionInvitationNotification

This type represents the notification for a file transfer session invitation.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The 'callbackData' element if it was passed by the application in the 'callbackReference' element when creating a subscription to file transfer notifications. See [REST_NetAPI_Common].

link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification (e.g. related File Transfer session). The server SHOULD include a link to the related subscription.
originatorAddress	xsd:anyURI	No	Address of the originating user(e.g. 'sip' URI, 'tel' URI, 'acr' URI)
originatorName	xsd:string	Yes	The name of the Originator
receiverAddress	xsd:anyURI	No	Address of Receiver invited to the session (e.g. 'sip' URI, 'tel' URI, 'acr' URI)
receiverName	xsd:string	Yes	Name of Receiver
fileInformation	FileInformation [1..unbounded]	No	A set of file's attributes. Note: According to [RC API RD], there is only one file. But in SIMPLE IM File transfer [OMA-SIMPLE_IM], there may be multiple files.

A root element named sessionInvitationNotification of type SessionInvitationNotification is allowed in notification request bodies.

The recipient can accept the request by sending a POST request with a "receiverSessionStatus" root element in the body to the URL passed in the "href" attribute of the "link" element with rel="receiverSessionStatus".

Typically this URL is: http://{serverRoot}/filetransfer/{apiVersion}/{userId}/sessions/{sessionId}/status.

The recipient can decline the request by sending a DELETE request to the URL passed in the "href" attribute of the "link" element with rel="FileTransferSessionInformation".

Typically this URL is: http://{serverRoot}/filetransfer/{apiVersion}/{userId}/sessions/{sessionId}.

If the recipient fails to react within a time interval defined by service policies, the session invitation will time out, and the session will be terminated.

5.2.2.7 Type: ReceiverSessionStatus

This type represents the status of the Receiver in the file transfer session.

Element	Type	Optional	Description
status	ReceiverStatus	No	Status of the Receiver. To indicate that the Receiver accepts the session invitation, this element MUST be set to "Connected".

acceptedFile	xsd:anyURI [0..unbounded]	Yes	Accepted files. Receiver feedback of the list of files accepted for transfer, containing a list of resourceURL values received in the "fileInformation" child element of "SessionInvitationNotification". Note: According to [RC API RD], there is only one file. But in SIMPLE IM File transfer [OMA-SIMPLE_IM], there may be multiple files. If there is no acceptedFile the meaning is that Receiver client rejected the Originator media offer.

A root element named receiverSessionStatus of type ReceiverSessionStatus is allowed in request bodies.

5.2.2.8 Type: FileTransferEventNotification

This type represents a notification about file transfer events that only need to convey the type of event without additional type-specific parameters.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The 'callbackData' element if it was passed by the application in the 'callbackReference' element when creating a subscription to file transfer notifications. See [REST_NetAPI_Common].
link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification (e.g. related File Transfer session). The server SHOULD include a link to the related subscription, and MUST include links as defined **Error! Reference source not found.**for FileTransferEventNotification.
eventType	EventType	No	Type of event
eventDescription	xsd:string	Yes	Textual description of the event

A root element named fileTransferEventNotification of type FileTransferEventNotification is allowed in notification request bodies.

5.2.2.9 Type: FileTransferSubscriptionList

This type represents a list of File Transfer notification subscriptions.

Element	Type	Optional	Description
fileTransferSubscription	FileTransferSubscription [0..unbounded]	Yes	Array of File Transfer notification subscriptions

resourceURL	xsd:anyURI	No	Self referring URL

A root element named fileTransferSubscriptionList of type FileTransferSubscriptionList is only allowed in response bodies.

5.2.2.10 Type: FileTransferSubscription

This type represents a subscription to file transfer related notifications, i.e. FileTransferEventNotification, SessionInvitationNotification, FileNotification and ReceiverAcceptanceNotification targeted at a particular user.

Element	Type	Optional	Description
callbackReference	common:CallbackReference	No	Client's Notification URL and OPTIONAL callbackData
duration	xsd:int	Yes	Period of time (in seconds) notifications are provided for. If set to "0" (zero), a default duration time, which is specified by the service policy, will be used. If the parameter is omitted, the notifications will continue until the maximum duration time, which is specified by the service policy, unless the notifications are stopped by deletion of subscription for notifications. This element MAY be given by the client during resource creation in order to signal the desired lifetime of the subscription. The server SHOULD return in this element the period of time for which the subscription will still be valid.
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This field SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids re-sending the message in such situations. In case the field is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
resourceURL	xsd:anyURI	Yes	Self referring URL . The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns

			an entity body, and in PUT requests.

A root element named fileTransferSubscription of type FileTransferSubscription is allowed in request and/or response bodies.

Note that the clientCorrelator is used for purposes of error recovery as specified in [REST_NetAPI_Common], and internal client purposes. The server is NOT REQUIRED to use the clientCorrelator value in any form in the creation of the URL of the resource. The document [REST_NetAPI_Common] provides a recommendation regarding the generation of the value of this field.

5.2.2.11 Type: FileNotification

This type represents a notification delivering a file URL.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The 'callbackData' element if it was passed by the application in the 'callbackReference' element when creating a subscription to file transfer notifications. See [REST_NetAPI_Common].
link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification (e.g. related File information). The server SHOULD include a link to the related subscription.
fileInformation	FileInformation[1..unbounded]	No	A set of file' s attributes. Note: In deployments according to [RC API RD], there is only one file. But in deployments based on SIMPLE IM File transfer [OMA-SIMPLE_IM], there can be multiple files.

A root element named fileContentNotification of type FileContentNotification is allowed in notification request bodies.

5.2.2.12 Type: ReceiverAcceptanceNotification

This type represents the Receiver acceptance notification

Element	Type	Optional	Description
callbackData	xsd:string	Yes	The 'callbackData' element if it was passed by the application in the 'callbackReference' element when creating a subscription to file transfer notifications. See [REST_NetAPI_Common].
link	common:Link [0..unbounded]	Yes	Links to other resources that are in relationship to the notification (e.g. related ReceiverSessionStatus). The server SHOULD include a link to the

			related subscription.
receiverAddress	xsd:anyURI	No	Address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Receiver of this file transfer session
receiverName	xsd:string	Yes	Name of the Receiver of this file transfer session
receiverSessionStatus	ReceiverSessionStatus	No	Status of a Receiver in the file transfer session

A root element named receiverAcceptanceNotification of type ReceiverAcceptanceNotification is allowed in notification request bodies.

5.2.2.13 Type: SubscriptionCancellationNotification

A type containing the subscription cancellation notification.

Element	Type	Optional	Description
callbackData	xsd:string	Yes	CallbackData if passed by the application in the receiptRequest element during the associated subscription operation. See [REST_NetAPI_Common] for details.
reason	common:ServiceError	Yes	Reason notification is being discontinued. SHOULD be present if the reason is different from a regular expiry of the subscription.
link	common:Link[1..unbounded]	No	Link to other resources that are in relationship with the resource. There MUST be a link to the subscription that is cancelled.

A root element named subscriptionCancellationNotification of type SubscriptionCancellationNotification is allowed in request and/or response bodies.

5.2.3 Enumerations

The subsections of this section define the enumerations used in the File Transfer API.

5.2.3.1 Enumeration: FileDisposition

This enumeration models the possible dispositions of a file transmitted in file transfer.

Enumeration	Description
Render	Indicates that the file should be automatically rendered.
Attachment	Indicates that the file should not be automatically rendered.

5.2.3.2 Enumeration: EventType

This enumeration defines the types of events. It is used in notifications.

Enumeration	Description

SessionCancelled	The Originator has cancelled the file transfer session during the invite phase.
SessionEnded	The file transfer session has ended.
Declined	The Receiver has declined the file transfer session invite.
Successful	The file was successfully delivered.
Failed	The file delivery has failed due to errors.
Aborted	The file delivery was aborted by the Originator.

5.2.3.3 Enumeration: ReceiverStatus

This enumeration defines the possible values for a Receiver in a file transfer session.

Enumeration	Description
Invited	User was invited to the session.
Connected	User is connected to the session.
Disconnected	User is disconnected from the session.

5.2.4 Values of the Link "rel" attribute

The "rel" attribute of the Link element is a free string set by the server implementation, to indicate a relationship between the current resource and an external resource. The following are possible strings (list is non-exhaustive, and can be extended):

- FileTransferSessionInformation
- FileInformation
- ReceiverSessionStatus
- FileTransferSubscriptionList
- FileTransferSubscription

These values indicate the kind of resource that the link points to.

5.2.5 MIME multipart representation

Actual file icon and file content can be represented as multipart/form-data entity bodies, where the first entry of the form are the root fields and the second entry of the form are the attachments. Details about the structure of such messages are defined in [REST_NetAPI_Common].

The creating session operation, sending file operation and session invitation notification use MIME multipart representation to include multimedia contents of file and icon within the HTTP request. The message can contain only one content item or more than one content item.

The contents (icon, file, etc.) SHALL be included using one of the following two options:

a. When the message contains *only file content item:* By including a MIME body with:

Content-Disposition: form-data; name="attachments", filename="<Name of the file>"

Content-Type: <Corresponding Content-Type>

b. When the message contains *only file icon item:* By including a MIME body with:

Content-Disposition: form-data; name="attachments", filename="icon"

Content-Type: <Corresponding Content-Type>

c. When the message contains *more than one content item*: By including a form-field with a MIME body with:

Content-Disposition: form-data; name="attachments"

Content-Type: multipart/mixed

Then, the possible *file content* SHALL be included as subparts, with:

Content-Disposition: attachment; filename="<Name of the file>"

Content-Type: <Corresponding Content-Type>Then, the possible *file icon* SHALL be included as subparts, with:

Content-Disposition: attachment; filename="icon"

Content-Type: <Corresponding Content-Type>

5.3 Sequence Diagrams

The following sub-sections describe the resources, methods and steps involved in typical scenarios.

In a sequence diagram, a step which involves delivering a notification is labeled with "POST or NOTIFY", where "POST" refers to delivery via the HTTP POST method, and "NOTIFY" refers to delivery using the Notification Channel [REST_NetAPI_NotificationChannel].

5.3.1 Subscription to file transfer notifications

The figure below shows a scenario for an application subscribing to file transfer notifications.

The notification URL passed by the client during the subscription step can be a Client-side Notification URL, or a Server-side Notification URL. Refer to [REST_NetAPI_NotificationChannel] for sequence flows illustrating the creation of a Notification Channel and obtaining a Server-side Notification URL on the server-side, and its use by the client via Long Polling.

The resources:

- To subscribe to file transfer notifications, create a new resource under **http://{serverRoot}/filetransfer/{apiVersion}/{userId}/subscriptions**
- To notify the applications about the subscription cancellation, POST or NOTIFY a SubscriptionCancellationNotification to the applications supplied Notification URL during notification subscription.
- To cancel subscription to file transfer notifications, delete the resource under **http://{serverRoot}/filetransfer/{apiVersion}/{userId}/subscriptions/{subscriptionId}**

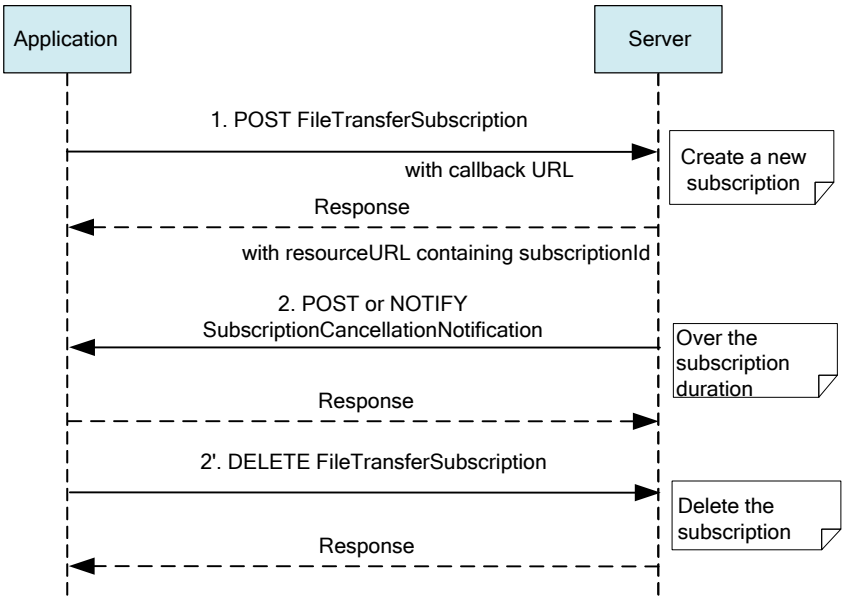

Figure 2 Subscribe to and unsubscribe from file transfer notifications

Outline of the flows:

1. An application subscribes to file transfer notifications using POST containing a FiletransferSubscription data structure to the resource containing all subscriptions and receives the result resource URL containing the subscriptionId.

2. When over the subscription duration, an application receives the subscription cancellation notification.

2'. Alternative, the application stops receiving notifications using DELETE with a resource URL containing the subscriptionId.

5.3.2 File Transfer with successful result

The figure below shows a scenario for a file transfer session with successful result, the initial API call can either include the actual file content or just external file repository URL, the application can also send actual file content by the send file API operation(refer to 5.3.4).The file transfer APIs support to transfer multi-files in one session.

The resources:

- To start a file transfer session, create a new resource with the FileTransferSessionInformation data structure under **http://{serverRoot}/filetransfer/{apiVersion}/{userId}/sessions**
- To accept a file transfer session invitation update the Receiver session status resource **http://{serverRoot}/filetransfer/{apiVersion}/{userId}/sessions/{sessionId}/status**
- To end a file transfer session delete the resource **http://{serverRoot}/filetransfer/{apiVersion}/{userId}/sessions/{sessionId}**
- To notify the applications about the incoming file transfer session invitation, POST or NOTIFY a SessionInvitationNotification to the applications supplied Notification URL during notification subscription.
- To notify the applications about the Receiver acceptance, POST or NOTIFY a ReceiverAcceptanceNotification to the applications supplied Notification URL during notification subscription.

- To notify the applications about the file URL link to actual file content for subsequent retrieval, POST or NOTIFY a FileNotification to the applications supplied Notification URL during notification subscription.

- To notify the applications about the status of the session and the status of the file transfer, POST or NOTIFY a FileTransferEventNotification to the applications supplied Notification URL during notification subscription.

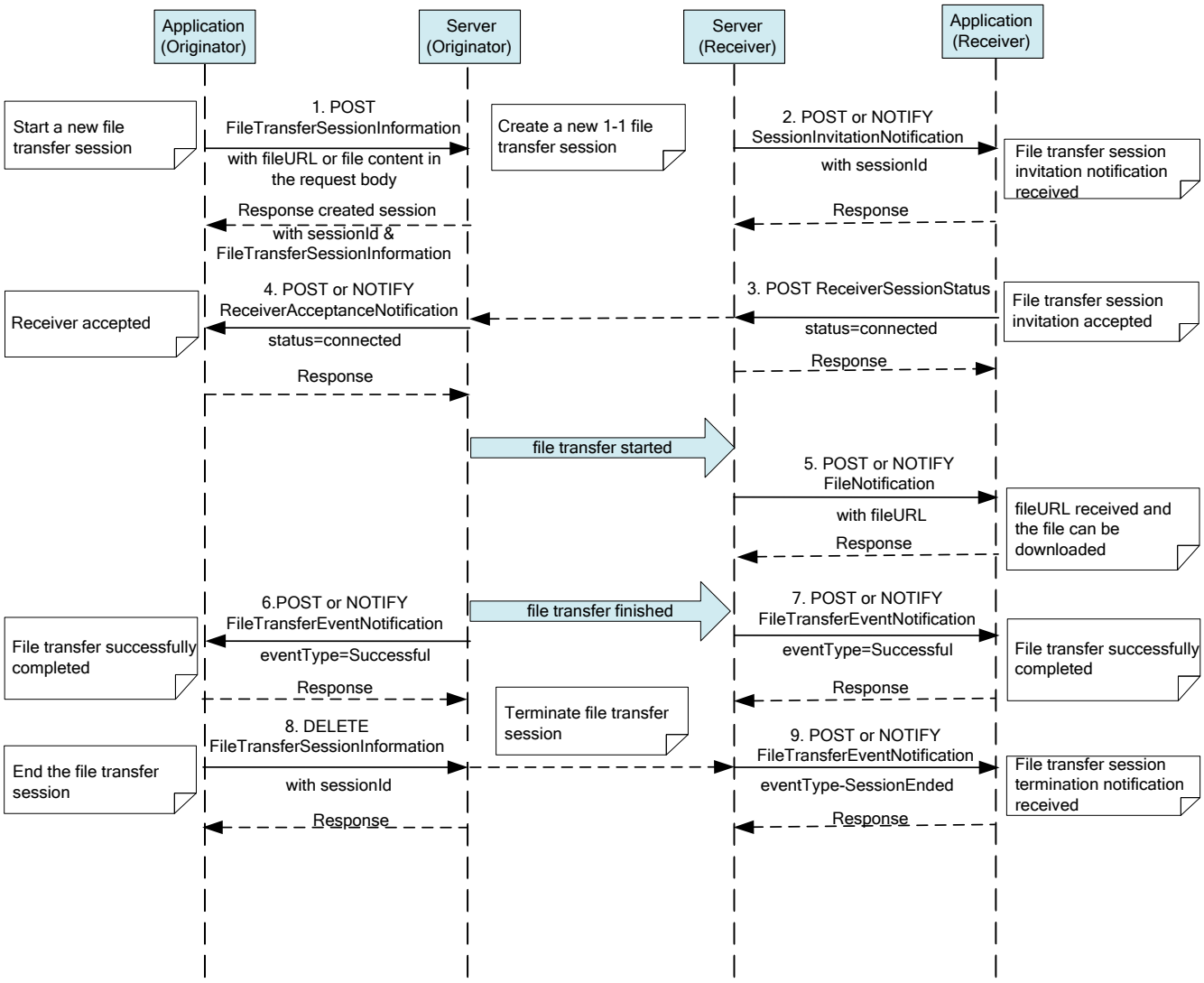

Figure 3 File Transfer session with successful result

Outline of the flows:

1. An application of the Originator starts a file transfer session using the POST method to submit the FileTransferSessionInformation data structure to the resource containing all file transfer sessions. Thereby the creation of a new file transfer session resource is triggered and the application of the Originator receives the resulting resource URL containing the sessionId. The request can include actual file content within MIME multipart entity bodies or the

external repository file URL. And the application of the Originator can also use send file operation (refer to 5.3.4) to transfer file content.

2. An application of the Receiver receives a file transfer session invitation notification.
3. The application of the Receiver accepts the file transfer session invitation using the POST method to submit the ReceiverSessionStatus data structure to the resource containing the session status and accepted files. The status MUST be set to “Connected”.
4. The application of the Originator receives a notification with ReceiverAcceptanceNotification structure indicating the Receiver has accepted the invitation. The application of the Originator starts to transfer the file.
5. After the file is ready for retrieval, the server of the Receiver notifies the application of the Receiver using FileNotification containing the fileURL which link to actual file content. The application of the Receiver can start downloading the file using the file URL received.

   Note: Depending on the implementations, the notification of the URL can be sent after the first chunk of data is received or when the complete file has been received (i.e. after step 7).

   Note: How the application retrieves the file using the URL is out of scope.
6. After the file transfer is completed, the server of the Originator notifies the application of the Originator about the successful of the file transfer using FileTransferEventNotification containing the eventType which is set to “Successful”.
7. The server of the Receiver notifies the application of the Receiver about the successful of the file transfer using FileTransferEventNotification containging the eventType which is set to “Successful” .
8. The application of the Originator ends the file transfer session using DELETE method on the resource URL of the session with sessionId

   Note: Both the application (Originator client) and application (Receiver client) can initiate ending the file transfer session.
9. The application of the Receiver receives a FileTransfertNotification data structure indicating the session has been ended.

Note: In case of the application (Receiver client) ends the file transfer session, the application (Originator client) receives a FileTransferEventNotification data structure indicating the session has been ended.

5.3.3 File transfer session failure

There are different causes which may lead to file transfer session failure, following are some options (not exclusive list):

a. The application of the Originator cancels the file transfer session.
b. The application of the Receiver reject or decline the file transfer session invitation
c. The file transfer failed due to the underlining network problem
d. The application of the Originator aborts the file transfer,

5.3.3.1 Canceling a file transfer invitation

The figure below shows a scenario for an application (Originator client) to cancel a file transfer session invitation.

The resources:

- To cancel a 1-1 file transfer session invitation, delete the session resource **http://{serverRoot}/filetransfer/{apiVersion}/{userId}/sessions/{sessionId}**

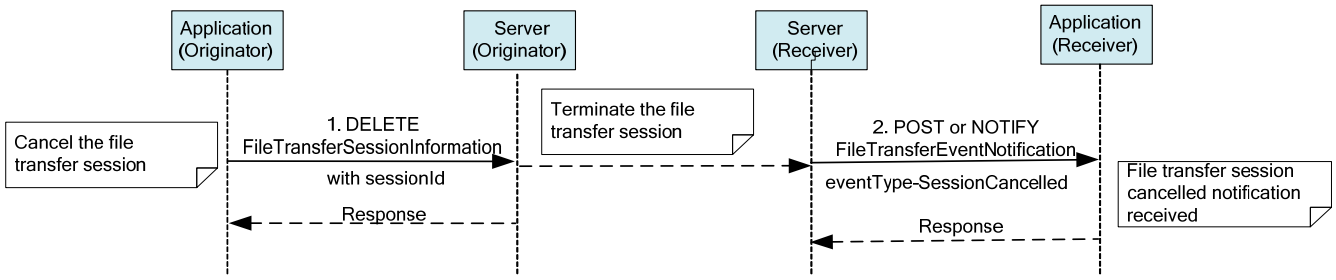

Figure 4 Cancelling a file transfer invitation

Outline of the flows:

An application of the Originator has created a file transfer session resource triggering a file transfer invitation sent to of the application of the Receiver (Refer to step 1 and step 2 in 5.3.2). Subsequently:

1. The application of the Originator can cancel a file transfer session invitation using the DELETE method on the resource URL of the session with sessionId and receives a response whether the request was successfully initiated.
2. A FileTransferEventNotification is sent to the application of the Receiver when the file transfer session has been cancelled, then the session is turned down.

Note that cancelling a session only works before the Receiver has accepted the file transfer invitation.

5.3.3.2 Declining a 1-1 file transfer sesssion invitation

The figure below shows a scenario for an application to decline a 1-1 file transfer session invitation.

The resources:

- To decline a 1-1 file transfer session invitation, delete the session resource **http://{serverRoot}/filetransfer/{apiVersion}/{userId}/sessions/{sessionId}**

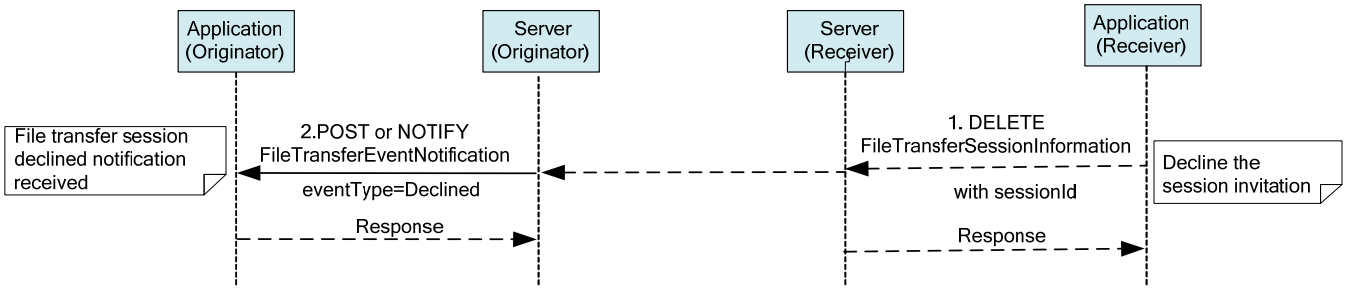

Figure 5 Declining a 1-1 file transfer session invitation

Outline of the flows:

1. An application of the Originator has created a file transfer session resource triggering a file transfer invitation sent to the application of the Receiver (Refer to step 1 and step 2 in 5.3.2). Subsequent, the application of the Receiver declines the file transfer session invitation using the DELETE method on the session resource including the sessionId.
2. The application of the Originator receives a FileTransferEventNotification structure indicating the Receiver has declined the invitation, then the session is torn down.

5.3.3.3 File transfer failed

The figure below shows a scenario for file transfer failed.

The resources:

- To notify the applications about file transfer failure, POST or NOTIFY a FileTransferEventNotification to the applications supplied Notification URL during notification subscription.

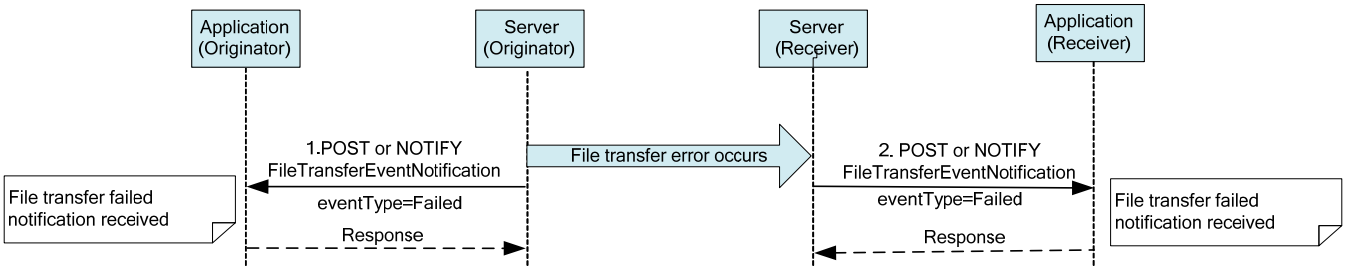

Figure 6 File transfer failed

Outline of the flows:

After an application of the Originator creates a file transfer session resource and the application of the Receiver accepts the file transfer session invitation (Refer to step 1 to step 4 in 5.3.2), the file transfer is started, subsequently:

1. When error occurs during file transfer, the server of the Originator notifies the application of the Originator using FileTransferEventNotification containing the eventType which is set to “Failed”.
2. The server of the Receiver also notifies the application of the Receiver using FileTransferEventNotification containing the eventType which is set to “Failed”

Note: When error occurs during the file transfer, the application (Receiver client) may already received the file URL and started to fetch the file, the server of the Receiver should cancel any HTTP request downloading the file using the file URL and disable the file URL. How the server implements this is out of scope.

5.3.3.4 File transfer aborted

The figure below shows a scenario for file transfer aborted.

The resources:

- To notify the application about file transfer abortion , POST or NOTIFY a FileTransferEventNotification to the applications supplied Notification URL during notification subscription.

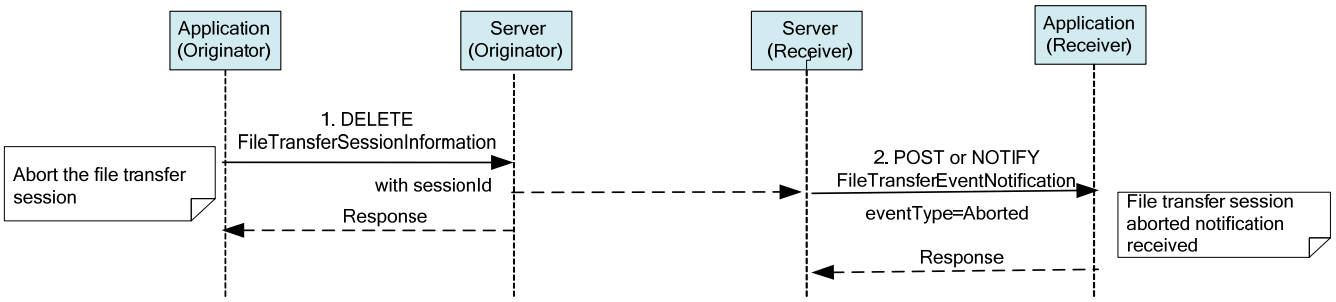

Figure 7 file transfer aborted

Outline of the flows:

After an application of the Originator creates a file transfer session resource and the application of the Receiver accepts the file transfer session invitation (Refer to step 1 to step 4 in 5.3.2), the file transfer is started, subsequently:

1. The application of the Originator can abort the file transfer using DELETE method on the resource URL of the session with sessionId.
2. The application of the Receiver receives a FileTransferEventNotification data structure indicating the file transfer has been aborted and the session is torn down.

Note that aborting the file transfer only works before the file has been completely transferred. After that, the DELETE method leads to a normal ending of the session.

Note: When the file transfer has been aborted, the application (Receiver client) may already received the file URL and started to fetch the file, the server of the Receiver should cancel any HTTP request downloading the file using the file URL and disable the file URL. How the server implements this is out of scope.

5.3.4 Send file

The figure below shows a scenario for an application to send file content.

The resources:

- To send file content, create a new resource at **http://{serverRoot}/filetransfer/{apiVersion}/{userId}/sessions/{sessionId}/files**

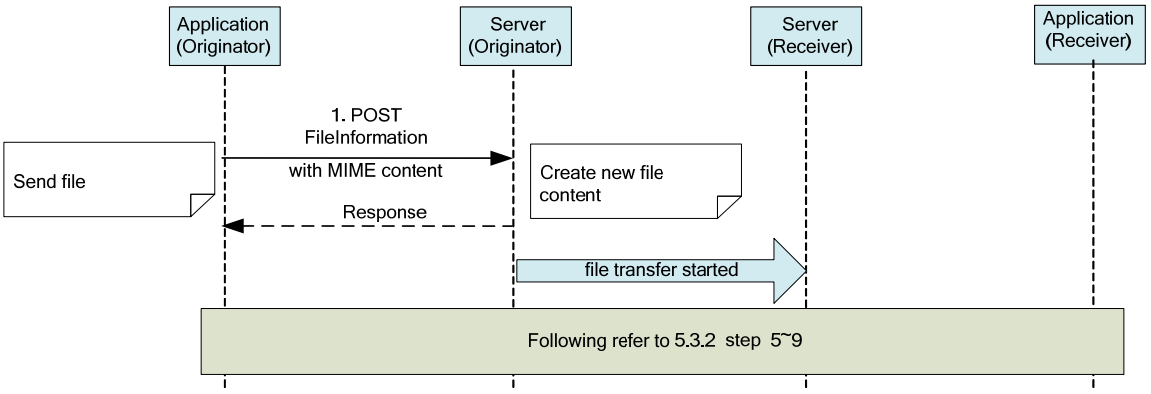

Figure 8 file transfer aborted

Outline of the flows:

After an application (Originator client) creates a file transfer session resource and the application (Receiver client) accepts the file transfer session invitation (Refer to step 1 to step 4 in 5.3.2), application (Originator client) can send an additional file :

1. The application (Originator client) sends the file using POST containing a FileInformation data structure and including the actual file content within a MIME multipart entity body to the resource representing all files in the session.

Then the server of the Originator starts file transfer to the server of the Receiver. Following refer to 5.3.2 step 5~9.

This operation is OPTIONAL.

6. Detailed specification of the resources

The following applies to all resources defined in this specification regardless of the representation format (i.e. XML, JSON, application/x-www-form-urlencoded):

- Reserved characters in URL variables (parts of a URL denoted below by a name in curly brackets) MUST be percent-encoded according to [RFC3986]. Note that this always applies, no matter whether the URL is used as a Request URL or inside the representation of a resource (such as in “resourceURL” and “link” elements).

- If a user identifier (e.g. address, userId, etc) of type anyURI is in the form of an MSISDN, it MUST be defined as a global number according to [RFC3966] (e.g. tel:+19585550100) The use of characters other than digits and the leading “+” sign SHOULD be avoided in order to ensure uniqueness of the resource URL. This applies regardless of whether the user identifier appears in a URL variable or in a parameter in the body of an HTTP message.

- If a user identifier (e.g. address, userId, etc) of type anyURI is in the form of a SIP URI, it MUST be defined according to [RFC3261].If a user identifier (e.g. address, userId, etc) of type anyURI is in the form of an Anonymous Customer Reference (ACR), it MUST be defined according to [IETF_ACR_draft], i.e. it MUST include the protocol prefix 'acr:' followed by the ACR.

    - The ACR ‘authorization’ is a supported reserved keyword, and MUST NOT be assigned as an ACR to any particular end user. See G.1.2 for details regarding the use of this reserved keyword.

- For requests and responses that have a body, the following applies: in the requests received, the server SHALL support JSON and XML encoding of the parameters in the body, and MAY support application/x-www-form-urlencoded parameters in the body. The Server SHALL return either JSON or XML encoded parameters in the response body, according to the result of the content type negotiation as specified in [REST_NetAPI_Common]. In notifications to the Client, the server SHALL use either XML or JSON encoding, depending on which format the client has specified in the related subscription. The generation and handling of the JSON representations SHALL follow the rules for JSON encoding in HTTP Requests/Responses as specified in [REST_ NetAPI_Common].

6.1 Resource: All subscriptions to file transfer notifications

The resource used is:

http://{serverRoot}/filetransfer/{apiVersion}/{userId}/subscriptions

This resource is used to manage subscriptions to file transfer event notifications. Note that there is one subscription per client instance.

This resource can be used in conjunction with a Client-side Notification URL, or in conjunction with a Server-side Notification URL. In this latter case, the application MUST first create a Notification Channel (see [REST_NetAPI_NotificationChannel]) before creating a subscription.

6.1.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL Example: example.com/exampleAPI

apiVersion	Version of the API clients want to use The value of this variable is defined in section **Error! Reference source not found.**.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.1.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful file transfer API, see section 7.

6.1.3 GET

This operation is used for reading the list of active file transfer notification subscriptions.

6.1.3.1 Example: Reading all active file transfer notification subscriptions (Informative)

6.1.3.1.1 Request

```
GET /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Accept: application/xml
Host: example.com
```

6.1.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSubscriptionList xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
    <fileTransferSubscription>
        <callbackReference>
            <notifyURL>http://application.example.com/filetransfer/notifications/77777</notifyURL>
            <callbackData>abcd</callbackData>
        </callbackReference>
        <duration>7200</duration>
        <clientCorrelator>12345</clientCorrelator>
        <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
    </fileTransferSubscription>
    <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions</resourceURL>
</ft:fileTransferSubscriptionList>
```

6.1.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC 2616].

6.1.5 POST

This operation is used to create a new subscription for file transfer notifications.

The notifyURL in the callbackReference either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

6.1.5.1 Example 1: Creating a new subscription to file transfer notifications, response with copy of created resource (Informative)

6.1.5.1.1 Request

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/ HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSubscription xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
<callbackReference>
  <notifyURL>http://application.example.com/filetransfer/notifications/77777</notifyURL>
   <callbackData>abcd</callbackData>
  </callbackReference>
  <duration>7200</duration>
  <clientCorrelator>12345</clientCorrelator>
</ft:fileTransferSubscription>
```

6.1.5.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSubscription xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
   <callbackReference>
          <notifyURL>http://application.example.com/filetransfer/notifications/77777</notifyURL>
          <callbackData>abcd</callbackData>
   </callbackReference>
  <duration>7200</duration>
   <clientCorrelator>12345</clientCorrelator>
   <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
```

```
</ft:fileTransferSubscription>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section 6.1.5.2.2).

6.1.5.2 Example 2: Creating a new subscription to file transfer notifications, response with location of created resource (Informative)

6.1.5.2.1 Request

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/ HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSubscription xmlns:ft ="urn:oma:xml:rest:netapi:filetransfer:1">
    <callbackReference>
        <notifyURL>http://application.example.com/filetransfer/notifications/77777</notifyURL>
        <callbackData>abcd</callbackData>
    </callbackReference>
    <duration>7200</duration>
    <clientCorrelator>12345</clientCorrelator>
</ft:fileTransferSubscription>
```

6.1.5.2.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
</common:resourceReference>
```

6.1.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: GET, POST’ field in the response as per section 14.7 of [RFC 2616].

6.2 Resource: Individual subscription to file transfer notifications

The resource used is:

http://{serverRoot}/filetransfer/{apiVersion}/{userId}/subscriptions/{subscriptionId}

This resource represents an individual subscription to file transfer notifications.

6.2.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section **Error! Reference source not found.**.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
subscriptionId	Identifier of the subscription

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.2.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful File transfer API, see section 7.

6.2.3 GET

This operation is used for reading an individual subscription.

6.2.3.1 Example: Reading an individual subscription (Informative)

This example shows also an alternative way to indicate desired content type in response from the server, by using URL query parameter "?resFormat" which is described in [REST_NetAPI_Common].

6.2.3.1.1 Request

```
GET /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001?resFormat=XML HTTP/1.1
Accept: application/xml
Host: example.com
```

6.2.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
```

```
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSubscription xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <callbackReference>
  <notifyURL>http://application.example.com/filetransfer/notifications/77777</notifyURL>
  <callbackData>abcd</callbackData>
 </callbackReference>
  <duration>7200</duration>
  <clientCorrelator>12345</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
</ft:fileTransferSubscription>
```

6.2.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC 2616].

6.2.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC 2616].

6.2.6 DELETE

This operation is used to cancel a subscription and to stop corresponding notifications.

6.2.6.1 Example: Cancelling a subscription (Informative)

6.2.6.1.1 Request

```
DELETE /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.2.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jun 2010 17:51:59 GMT
```

6.3 Resource: All 1-1 file transfer sessions

The resource used is:

http://{serverRoot}/filetransfer/{apiVersion}/{userId}/sessions

This resource represents the active 1-1 file transfer sessions for a particular user.

6.3.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section **Error! Reference source not found.**.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.3.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful File transfer API, see section 7.

6.3.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.3.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.3.5 POST

This operation is used to create a new 1-1 file transfer session. It can either include the actual file content or just external file repository URL, the application can also send actual file content by the sending file API operation (refer to 6.6.5). This operation supports to transfer multi-files in one session, these files' name shall not be same.

6.3.5.1 Example 1: Creating a new 1-1 file transfer session with file content (Informative)

6.3.5.1.1 Request

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456=="
Host: example.com
MIME-Version: 1.0
Content-Length: nnnn
Accept: application/xml

--===============123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSessionInformation xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <originatorAddress>tel:+19585550100</originatorAddress >
  <originatorName>Max Muster</originatorName>
  <receiverAddress>tel:+19585550102</receiverAddress>
  <receiverName>Peter E. Xample</receiverName>
  <fileInformation>
    <fileSelector>
      <name>sunset.jpg</name>
      <type>image/jpeg</type>
      <size>4096</size>
      <hash>
        <algorithm>sha-1</algorithm>
        <value>58231FE8653BBCF371362F86D471913EE4B1DF2F</value>
      </hash>
    </fileSelector>
    <fileDisposition>Attachment</fileDisposition>
    <fileDescription>This is my latest picture</fileDescription>
    <fileDate>
      <cDate>2011-08-21T00:00:00-04:00</cDate>
    </fileDate>
    <fileIcon>cid:id3@alicepc.example.com</fileIcon>
  </fileInformation>
  <clientCorrelator>104567</clientCorrelator>
</ft:fileTransferSessionInformation>

--===============123456==
Content-Disposition: form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”====aaabbb”
--====aaabbb
Content-Disposition:attachment; filename=”icon”
Content-Type: image/gif
Content-ID: cid:id3@alicepc.example.com

[..small preview icon...]
--====aaabbb--
```

```
Content-Disposition:attachment;filename=”sunset.jpg”
Content-Type: image/jpg

JPEG ...binary image data...
--====aaabbb--
--===============123456==--
```

6.3.5.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSessionInformation xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Max Muster</originatorName>
  <receiverAddress>tel:+19585550102</receiverAddress>
  <receiverName>Peter E. Xample</receiverName>
  <status>Invited</status>
  <fileInformation>
    <fileSelector>
      <name>sunset.jpg</name>
      <type>image/jpeg</type>
    </fileSelector>
    <fileURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001.jpg</fileURL>

<resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001</resourceURL>
  </fileInformation>
  <clientCorrelator>104567</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001</resourceURL>
</ft:fileTransferSessionInformation>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section 6.3.5.3.2).

6.3.5.2 Example 2: Creating a new 1-1 file transfer session with external file repository URL (Informative)

6.3.5.2.1 Request

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com
MIME-Version: 1.0
```

```
Content-Length: nnnn
Accept: application/xml

--===============123456==
Content-Disposition: /form-data; name=”root-fields”
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSessionInformation xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Max Muster</originatorName>
  <receiverAddress>tel:+19585550102</receiverAddress>
  <receiverName>Peter E. Xample</receiverName>
  <fileInformation>
    <fileSelector>
      <name>sunset.jpg</name>
      <type>image/jpeg</type>
      <size>4096</size>
      <hash>
        <algorithm>sha-1</algorithm>
        <value>58231FE8653BBCF371362F86D471913EE4B1DF2F</value>
      </hash>
    </fileSelector>
    <fileDisposition>Attachment</fileDisposition>
    <fileDescription>This is my latest picture</fileDescription>
    <fileDate>
      <cDate>2011-08-21T00:00:00-04:00</cDate>
    </fileDate>
    <fileIcon>cid:id3@alicepc.example.com</fileIcon>
    <fileURL>http://alicepc.example.com/sunset.jpg</fileURL>
  </fileInformation>
  <clientCorrelator>104567</clientCorrelator>
</ft:fileTransferSessionInformation>


--===============123456==
Content-Disposition: form-data; name=”attachments”, filename=”icon”
Content-Type: image/gif
Content-Transfer-Encoding: binary
Content-ID: <id3@alicepc.example.com>
Content-Length: [length of image]
[..small preview icon...]

--===============123456==--
```

6.3.5.2.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT
```

```
<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSessionInformation xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Max Muster</originatorName>
  <receiverAddress>tel:+19585550102</receiverAddress>
  <receiverName>Peter E. Xample</receiverName>
  <status>Invited</status>
  <fileInformation>
    <fileSelector>
      <name>sunset.jpg</name>
      <type>image/jpeg</type>
    </fileSelector>
    <fileURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001.jpg</fileURL>
<resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001 </resourceURL>
  </fileInformation>
  <clientCorrelator>104567</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001</resourceURL>
</ft:fileTransferSessionInformation>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section 6.3.5.3.2).

6.3.5.3 Example 3: Creating a new 1-1 file transfer session with external file repository URL without icon, response with location of created resource (Informative)

6.3.5.3.1 Request

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Host: example.com
Accept: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSessionInformation xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Max Muster</originatorName>
  <receiverAddress>tel:+19585550102</receiverAddress>
  <receiverName>Peter E. Xample</receiverName>
  <fileInformation>
    <fileSelector>
      <name>sunset.jpg</name>
      <type>image/jpeg</type>
      <size>4096</size>
      <hash>
        <algorithm>sha-1</algorithm>
        <value>58231FE8653BBCF371362F86D471913EE4B1DF2F</value>
      </hash>
    </fileSelector>
    <fileDisposition>Attachment</fileDisposition>
```

```
    <fileDescription>This is my latest picture</fileDescription>
    <fileDate>
      <cDate>2011-08-21T00:00:00-04:00</cDate>
    </fileDate>
    <fileURL>http://alicepc.example.com/sunset.jpg</fileURL>
  </fileInformation>
  <clientCorrelator>104567</clientCorrelator>
</ft:fileTransferSessionInformation>
```

6.3.5.3.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001</resourceURL>
</common:resourceReference>
```

6.3.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.4 Resource: Individual 1-1 file transfer session

The resource used is:

http://{serverRoot}/filetransfer/{apiVersion}/{userId}/sessions/{sessionId}

This resource represents a file transfer session.

6.4.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section **Error! Reference source not found.**.

userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the session

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.4.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful File transfer API, see section 7.

6.4.3 GET

This operation is used to retrieve file transfer session information.

6.4.3.1 Example : Retrieving file transfer session information (Informative)

6.4.3.1.1 Request

```
GET /exampleAPI/file transfer/v1/tel%3A%2B19585550100/sessions/sess001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.4.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSessionInformation xmlns:ft ="urn:oma:xml:rest:netapi:filetransfer:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Max Muster</originatorName>
  <receiverAddress>tel:+19585550102</receiverAddress>
  <receiverName>Peter E. Xample</receiverName>
  <status>Connected</status>
  <fileInformation>
   <fileSelector>
    <name>sunset.jpg</name>
    <type>image/jpeg</type>
    <size>4096</size>
    <hash>
     <algorithm>sha-1</algorithm>
     <value>58231FE8653BBCF371362F86D471913EE4B1DF2F</value>
    </hash>
   </fileSelector>
   <fileDisposition>Attachment</fileDisposition>
```

```
    <fileDescription>This is my latest picture</fileDescription>
    <fileDate>
      <cDate>2011-08-21T00:00:00-04:00</cDate>
    </fileDate>
    <fileIcon>cid:id3@alicepc.example.com</fileIcon>
    <fileURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/ file001.jpg</fileURL>
    <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001
</resourceURL>
  </fileInformation>
  <clientCorrelator>104567</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001</resourceURL>
</ft:fileTransferSessionInformation>
```

6.4.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC 2616].

6.4.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, DELETE' field in the response as per section 14.7 of [RFC 2616].

6.4.6 DELETE

When the file transfer session status is "Invited", the application of Originator can use this operation to cancel file transfer invitation.

When the file transfer session status is "Invited", the application of Receiver can use this operation to decline file transfer invitation.

When the file transfer session status is "Connected", both the application of Originator and the application of Receiver can use this operation to terminate the file transfer session.

6.4.6.1 Example: Cancel file transfer invitation/ Decline file transfer invitation/ Terminating a file transfer session (Informative)

6.4.6.1.1 Request

```
DELETE /exampleAPI/file transfer/v1/tel%3A%2B19585550100/sessions/sess001 HTTP/1.1
Accept: application/xml
Host: example.com
```

6.4.6.1.2 Response

```
HTTP/1.1 204 No Content
```

Date: Mon, 28 Jul 2011 17:51:59 GMT

6.5 Resource: session status

The resource used is:

http://{serverRoot}/file transfer/{apiVersion}/{userId}/sessions/{sessionId}/status

This resource represents the status of the session and is used for accepting a 1-1 file transfer invitation, by means of updating the status.

6.5.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section **Error! Reference source not found.**.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the session

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.5.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful File Transfer API, see section 7.

6.5.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.5.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.5.5 POST

This operation is used for accepting a 1-1 file transfer invitation, by means of updating the status.

6.5.5.1 Example : Accepting a 1-1 file transfer invitation (Informative)

6.5.5.1.1 Request

```
POST /exampleAPI/file transfer/v1/tel%3A%2B19585550102/sessions/sess001/status HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Accept: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<ft:receiverSessionStatus xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
<status>Connected</status>
<acceptedFile>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001</acceptedFile>
</ft:receiverSessionStatus>
```

6.5.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.5.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.6 Resource: Files in a 1-1 file transfer session

The resource used is:

http://{serverRoot}/file transfer/{apiVersion}/{userId}/sessions/{sessionId}/files

This resource represents all files in a file transfer session. In the current version of the specification, it is a “send-only” resource (i.e. files cannot be read back).

6.6.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path Port and base path are OPTIONAL Example: example.com/exampleAPI
apiVersion	Version of the API clients want to use The value of this variable is defined in section **Error! Reference source not found.**.
userId	Identifier of the user on whose behalf the application acts Examples: tel:+19585550100, acr:pseudonym123
sessionId	Identifier of the session

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.6.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful File transfer API, see section 7.

6.6.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.6.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.6.5 POST

This operation is used to send a file. This method MUST return either a common:resourceReference root element or a file transfer:FileInformation root element, where using the first option is RECOMMENDED.

6.6.5.1 Example 1: Sending a file, using tel URI and returning the copy of the created resource (Informative)

6.6.5.1.1 Request

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files HTTP/1.1
Content-Type: multipart/form-data; boundary="==========123456==";
Host: example.com
MIME-Version: 1.0
Accept: application/xml
Content-Length: nnnn
--==========123456==
Content-Disposition: form-data; name="root-fields"
```

```
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileInformation xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
     <fileSelector>
        <name>sunset.jpg</name>
        <type>image/jpeg</type>
        <size>4096</size>
        <hash>
           <algorithm>sha-1</algorithm>
           <value>58231FE8653BBCF371362F86D471913EE4B1DF2F </value>
        </hash>
     </fileSelector>
</ft:fileInformation>

--===============123456==
Content-Disposition: form-data; name="attachments", filename="sunset.jpg"
Content-Type: image/jpeg

JPEG ...binary image data...
--===============123456==--
```

6.6.5.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileInformation xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <fileSelector>
    <name>sunset.jpg</name>
    <type>image/jpeg</type>
  </fileSelector>
  <fileURL>http:// example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001.jpg</fileURL>
  <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001</resourceURL>
</ft:fileInformation>
```

Note that alternatively to returning a copy of the created resource, the location of created resource could be returned using the common:resourceReference root element (see section 6.6.5.2.2).

6.6.5.2 Example 2: Sending a file, using ACR and returning a location of the created resource (Informative)

6.6.5.2.1 Request

```
POST /exampleAPI/filetransfer/v1/acr%3Apseudonym123/sessions/sess001/files HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com
MIME-Version: 1.0
Content-Length: nnnn
Accept: application/xml

--===============123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileInformation xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
     <fileSelector>
        <name>sunset.jpg</name>
        <type>image/jpeg</type>
     </fileSelector>
</ft:fileInformation>

--===============123456==
Content-Disposition: form-data; name=”attachments”, filename=”sunset.jpg”
Content-Type: image/jpeg

JPEG ...binary image data...
--===============123456==--
```

6.6.5.2.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/filetransfer/v1/acr%3Apseudonym123/sessions/sess001/files/file001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>http://example.com/exampleAPI/filetransfer/v1/acr%3Apseudonym123/sessions/sess001/files/file001</resourceURL>
</common:resourceReference>
```

6.6.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.7 Resource: Client notification about session invitations

This resource is a callback URL provided by the client for notification about file transfer session invitations. The RESTful File Transfer API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.7.5.

The notification behaviour SHALL follow therule below:

Notification sent to	**Response to Notification**	**Link rel**	**Link href** Base URL: //{serverRoot}/ filetransfer/{apiVersion}/{userId}/sessions
Receiver	Accept invitation (6.5.5) or Decline invitation(6.4.6)	FileTransferSessionInformation	/{sessionId}

6.7.1 Request URL variables

Client provided if any.

6.7.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful File Transfer API, see section 7.

6.7.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.7.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.7.5 POST

This operation is used to notify the client about file transfer session invitations.

6.7.5.1 Example: Notify a client about file transfer session invitations(Informative)

This example notification is triggered by the request in example 6.3.5. Note that the {userId} resourceURL variable represents the userId of the user on whose behalf the application acts, not the one of the Originator.

6.7.5.1.1 Request

```
POST /filetransfer/notifications/77777 HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com
MIME-Version: 1.0
Content-Length: nnnn


--===============123456==
Content-Disposition: multipart/form-data; name=”root-fields”
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<ft:sessionInvitationNotification xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <callbackData>abcd</callbackData>
  <link rel="FileTransferSessionInformation"
     href=" http://example.com/exampleAPI/file transfer/v1/tel%3A%2B19585550100/sessions/sess001"/>
    <originatorAddress>tel:+19585550100</originatorAddress>
    <originatorName>Max Muster</originatorName>
    <receiverAddress>tel:+19585550102</receiverAddress>
    <receiverName>Peter E. Xample</receiverName>
    <fileInformation>
      <fileSelector>
        <name>sunset.jpg</name>
        <type>image/jpeg</type>
        <size>4096</size>
        <hash>
          <algorithm>sha-1</algorithm>
          <value>58231FE8653BBCF371362F86D471913EE4B1DF2F</value>
        </hash>
      </fileSelector>
      <fileDisposition>Attachment</fileDisposition>
      <fileDescription>This is my latest picture</fileDescription>
      <fileDate>
        <cDate>2011-08-21T00:00:00-04:00</cDate>
      </fileDate>
      <fileIcon>cid:id3@alicepc.example.com</fileIcon>
      <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001</resource
URL>
    </fileInformation>
</ft:sessionInvitationNotification>

--===============123456==
Content-Disposition: form-data; name=”attachments”, filename=”icon”
Content-Type: image/gif
Content-ID: <id3@alicepc.example.com>

[..small preview icon...]
```

```
--===============123456==--
```

6.7.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.7.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.8 Resource: Client notification about file transfer session events

This resource is a callback URL provided by the client for notification about various file transfer session events. The RESTful File Transfer API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.8.5.

The notification behaviour SHALL follow the rule below:

Notification sent to	**Response to Notification**	**Link rel**	**Link href** Base URL: //{serverRoot}/ filetransfer/{apiVersion}/{userId}/sessions
Originator & Receiver	n/a	FileTransferSessionInformation	/{sessionId}

6.8.1 Request URL variables

Client provided if any.

6.8.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful File Transfer API, see section 7.

6.8.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.8.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.8.5 POST

This operation is used to notify the client about file transfer session events.

6.8.5.1 Example: Notify a client about file transfer session events (Informative)

6.8.5.1.1 Request

```
POST /filetransfer/notifications/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnnHost: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferEventNotification xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <callbackData>abcd</callbackData>
  <link rel="FileTransferSessionInformation"
    href="http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001"/>
  <link rel="FileTransferSubscription"
    href="http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001"/>
  <eventType>SessionEnded</eventType>
  <eventDescription>The session has ended.</eventDescription>
</ft:fileTransferEventNotification>
```

6.8.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.8.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.9 Resource: Client notification about file content link

This resource is a callback URL provided by the client for notifications about file content link. The file URL link to actual file content is inlined in the notifications. After the Receiver has accepted the session invitation, the server will send the FileNotification to the Receiver which includes the link to actual file content on external file content repository or link to internal file content storage on the server side.

The RESTful File Transfer API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.9.5.

The notification behaviour SHALL follow the rule below:

Notification sent to	**Response to Notification**	**Link rel**	**Link href** Base URL: //{serverRoot}/ filetransfer/{apiVersion}/{userId}/sessions
Receiver	n/a	FileTransferSessionInformation	/{sessionId}/

6.9.1 Request URL variables

Client provided if any.

6.9.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful File Transfer API, see section 7.

6.9.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.9.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.9.5 POST

This operation is used to notify the client about file content link.

6.9.5.1 Example: Notify a client about file content link (Informative)

6.9.5.1.1 Request

```
POST /filetransfer/notifications/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileNotification xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <callbackData>abcd</callbackData>
  <link rel="FileTransferSubscription"
     href="http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550102/subscriptions/sub001"/>
  <fileInformation>
    <fileSelector>
      <name>sunset.jpg</name>
      <type>image/jpeg</type>
    </fileSelector>
    <fileURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001.jpg</fileURL>
<resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001</resourceURL>
  </fileInformation>
</ft:fileNotification>
```

6.9.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.9.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC 2616].

6.10 Resource: Client notification about Receiver acceptance

This resource is a callback URL provided by the client for notifications about Receiver acceptance.

The RESTful File Transfer API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.10.5.

The notification behaviour SHALL follow the rule below:

Notification sent to	Response to Notification	Link rel	Link href Base URL: //{serverRoot}/ filetransfer/{apiVersion}/{userId}/sessions
Originator	n/a	ReceiverSessionStatus	/{sessionId}/status

6.10.1 Request URL variables

Client provided if any.

6.10.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful File Transfer API, see section 7.

6.10.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.10.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.10.5 POST

This operation is used to notify the client about Receiver acceptance.

6.10.5.1 Example: Notify a client about Receiver acceptance (Informative)

6.10.5.1.1 Request

```
POST /filetransfer/notifications/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<ft:receiverAcceptanceNotification xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <callbackData>abcd</callbackData>
  <link rel="FileTransferSessionInformation"
    href="http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001"/>
  <link rel="FileTransferSubscription"
```

```
      href="http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001"/>
  <receiverAddress>tel:+19585550102</receiverAddress>
  <receiverName>Peter E. Xample</receiverName>
  <receiverSessionStatus>
     <status>Connected</status>
     <acceptedFile>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001</acceptedFile>
   </receiverSessionStatus>
</ft:receiverAcceptanceNotification>
```

6.10.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.10.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.11 Resource: Client notification about subscription cancellation

This resource is a callback URL provided by the client for notification about subscription cancellations, which are usually due to the subscription expiring. The RESTful File Transfer API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel in order to use Long Polling to obtain the notifications, in order to retrieve the notifications, the client needs to use the Long Polling mechanism described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.11.5.

The notification is sent by the server to the party to whom the cancelled subscription belongs.

The notification behaviour SHALL follow the rule below:

Notification sent to	**Response to Notification**	**Link rel**	**Link href** Base URL: //{serverRoot}/ filetransfer/{apiVersion}/{userId}
Originator & Receiver	n/a	FileTransferSubscription	/subscriptions/{subscriptionId}

6.11.1 Request URL variables

Client provided if any.

6.11.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to the RESTful File Transfer API, see section 7.

6.11.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.11.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

6.11.5 POST

This operation is used to notify the client about subscription cancellations.

6.11.5.1 Example: Notify a client about subscription cancellations (Informative)

6.11.5.1.1 Request

```
POST /filetransfer/notifications/77777 HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Host: application.example.com

<?xml version="1.0" encoding="UTF-8"?>
<ft:subscriptionCancellationNotification xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <callbackData>abcd</callbackData>
  <link rel="FileTransferSubscription "
     href="http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001"/>
</ft:subscriptionCancellationNotification>
```

6.11.5.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

6.11.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC 2616].

7. Fault definitions

7.1 Service Exceptions

For common Service Exceptions refer to [REST_NetAPI_Common]. None in this release.

7.2 Policy Exceptions

For common Policy Exceptions refer to [REST_NetAPI_Common]. The following additional Policy Exception codes are defined for the RESTful File Transfer API.

7.2.1 POL1015: Sending file transfer files not supported

Name	Description
MessageID	POL1015
Text	Sending file transfer files is not supported
Variables	none
HTTP response	403 Forbidden

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
n/a	n/a	No prior version

A.2 Draft/Candidate Version 1.0 History

Document Identifier	Date	Sections	Description
Draft Version OMA-TS-REST_NetAPI_FileTransfer-V1_0	03 May 2011	All	TS skeleton created
	13 Jun 2011	All	in line with INP #17R02, #60 and action REST-NetAPI-2011-A027
	01 Aug 2011	All	OMA-ARC-REST-NetAPI-2011-0133R02-CR_FileTransfer_Resources_alignment_with_new_resource_model OMA-ARC-REST-NetAPI-2011-0134R02-CR_FileTransfer_DataTypes_alignment_with_new_resource_model
	19 Sep 2011	All	OMA-ARC-REST-NetAPI-2011-0187-CR_FileTransfer_action OMA-ARC-REST-NetAPI-2011-0216R02-CR_FileTransfer_fix_CR180R01
	31 Oct 2011	All	OMA-ARC-REST-NetAPI-2011-0260-CR_XSD_for_FileTransfer OMA-ARC-REST-NetAPI-2011-0261-CR_FileTransfer_action OMA-ARC-REST-NetAPI-2011-0234R01-CR_FileTransfer_flow
	15 Nov 2011	All	OMA-ARC-REST-NetAPI-2011-0342R02-CR_FileTransfer_update OMA-ARC-REST-NetAPI-2011-0344R03-CR_FileTransfer_Sequence_Diagrams_update OMA-ARC-REST-NetAPI-2011-0359R01-CR_FileTransfer_CR325_resourceURL_blueprint
	29 Nov 2011	All	OMA-ARC-REST-NetAPI-2011-0387R03-CR_FileTransfer_Examples OMA-ARC-REST-NetAPI-2011-0394R01-CR_FileTransfer_SCR OMA-ARC-REST-NetAPI-2011-0397-CR_FileTransfer_Appendix_G OMA-ARC-REST-NetAPI-2011-0413-CR_Appendix_F_in_FileTransfer OMA-ARC-REST-NetAPI-2011-0414-CR_FileTransfer_Section_4
	20 Feb 2012	All	OMA-ARC-REST-NetAPI-2012-0047R02-CR_Filetransfer_TS_CONRR_comments_resolution
	23 Feb 2012	All	OMA-ARC-REST-NetAPI-2012-0080-CR_FileTransfer_fixes
	29 Mar 2012	All	OMA-ARC-REST-NetAPI-2012-0119R01-CR_Filetransfer_TS_resolution
	03 Apr 2012	All	Removed suffix "-clean" from document name. Editorial changes
Candidate Version OMA-TS-REST_NetAPI_FileTransfer-V1_0	17 Apr 2012	n/a	Status changed to Candidate by TP TP Ref # OMA-TP-2012-0171-INP_REST_NetAPI_FileTransfer_1_0_ERP_and_ETR_for_Candidate_Approval

Appendix B. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [SCRRULES].

B.1 SCR for REST.FileTransfer Server

Item	Function	Reference	Requirement
REST-FileTransfer-SUPPORT-S-001-M	Support for the RESTful File Transfer API	5, 6	
REST-FileTransfer-SUPPORT-S-002-M	Support for the XML request & response format	6	
REST-FileTransfer-SUPPORT-S-003-M	Support for the JSON request & response format	6	
REST-FileTransfer-SUPPORT-S-004-O	Support for the application/x-www-form-urlencoded format	Appendix C	

B.1.1 SCR for REST.FileTransfer.Subscriptions Server

Item	Function	Reference	Requirement
REST-FileTransfer- SUBSCR-001-M	Support for subscriptions to file transfer notifications	**Error! Reference source not found.**	
REST-FileTransfer- SUBSCR-002-O	Read all active subscriptions - GET	**Error! Reference source not found.**	
REST-FileTransfer- SUBSCR-003-M	create a new subscription for file transfer notifications - POST	**Error! Reference source not found.**	

B.1.2 SCR for REST.FileTransfer.Individual.Subscription Server

Item	Function	Reference	Requirement
REST-FileTransfer- IND-SUBSCR-001-M	Support for individual subscription to file transfer notifications	**Error! Reference source not found.**	
REST-FileTransfer- IND-SUBSCR-002-O	Read an individual subscription - GET	**Error! Reference source not found.**	
REST-FileTransfer- IND-SUBSCR-003-M	Cancel a subscription and stop corresponding notifications - DELETE	**Error! Reference source not found.**	

B.1.3 SCR for REST.FileTransfer.Sessions Server

Item	Function	Reference	Requirement
REST-FileTransfer-Sess-001-M	Support for active 1-1 file transfer sessions for a particular user	**Error! Reference source not found.**	
REST-FileTransfer-Sess-002-M	Create a new 1-1 file transfer session – POST	**Error! Reference source not found.**	

B.1.4 SCR for REST.FileTransfer.Individual.Session Server

Item	Function	Reference	Requirement
REST-FileTransfer-IND-Sess-001-M	Support for a file transfer session for a particular user	**Error! Reference source not found.**	
REST-FileTransfer-IND-Sess-002-O	Retrieve file transfer session information – GET	**Error! Reference source not found.**	
REST-FileTransfer-IND-Sess-003-M	Cancel file transfer invitation/ Decline file transfer invitation/ Terminating a file transfer session – DELETE	**Error! Reference source not found.**	

B.1.5 SCR for REST.FileTransfer.Individual.Session.Status Server

Item	Function	Reference	Requirement
REST-FileTransfer-IND-Sess-Stat-001-M	Support for status of the session	**Error! Reference source not found.**	
REST-FileTransfer-IND-Sess-Stat-002-M	Accept a 1-1 file transfer invitation – POST	6.5.5	

B.1.6 SCR for REST.FileTransfer.Session.Files Server

Item	Function	Reference	Requirement
REST-FileTransfer-IND-Sess-Files-001-M	Support for all files in a file transfer session	**Error! Reference source not found.**	
REST-FileTransfer-IND-Sess-Files-002-M	Send a file – POST	6.6.5	

B.1.7 SCR for REST.FileTransfer.SessionInvitation.Notifications Server

Item	Function	Reference	Requirement
REST-FileTransfer-INV-NOTIF-001-M	Support for notifying application about file transfer session invitations	6.7	
REST-FileTransfer-INV-NOTIF-002-M	Notify application about file transfer session invitations - POST	6.7.5	

B.1.8 SCR for REST.FileTransfer.Events.Notifications Server

Item	Function	Reference	Requirement
REST-FileTransfer-Event-NOTIF-001-M	Support for notifying application about various file transfer session events	6.8	
REST-FileTransfer-Event-NOTIF-002-M	Notify application about file transfer session events - POST	6.8.5	

B.1.9 SCR for REST.FileTransfer.Link.Notifications Server

Item	Function	Reference	Requirement
REST-FileTransfer-Link-NOTIF-001-M	Support for notifying application about file content link	6.9	
REST-FileTransfer-Link-NOTIF-002-M	Notify application about file content link - POST	6.9.5	

B.1.10 SCR for REST.FileTransfer.ReceiverAcceptance.Notifications Server

Item	Function	Reference	Requirement
REST-FileTransfer-RA-NOTIF-001-M	Support for notifying application about Receiver acceptance	6.10	
REST-FileTransfer-RA-NOTIF-002-M	Notify application about Receiver acceptance - POST	6.10.5	

B.1.11 SCR for REST.FileTransfer.SubscriptionCancellation. Notifications Server

Item	Function	Reference	Requirement
REST-FileTransfer-SUBCXL-NOTIF-001-M	Support for notifying application about Subscription	6.11	

Item	Function	Reference	Requirement
	Cancellation		
REST-FileTransfer-SUBCXL-NOTIF-002-M	Notify application about Subscription Cancellation - POST	6.11.5	

Appendix C. Application/x-www-form-urlencoded Request Format for POST Operations (Normative)

This section defines a format for the RESTful File Transfer API requests where the body of the request is encoded using the application/x-www-form-urlencoded MIME type.

Note: only the request body is encoded as application/x-www-form-urlencoded, the response is still encoded as XML or JSON depending on the preference of the client and the capabilities of the server. Names and values MUST follow the application/x-www-form-urlencoded character escaping rules from [W3C_URLENC].

The encoding is defined below for the following File Transfer REST operations which are based on POST requests:

- Creating a new subscription to file transfer notifications
- Creating a new file transfer session
- Accepting a file transfer session invitation
- Sending a file

C.1 Creating a new subscription to file transfer notifications

This operation is used to create a new subscription to file transfer notifications. See section 6.1.5.

The notifyURL either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

The request parameters are as follows:

Name	Type/Values	Optional	Description
notifyURL	xsd:anyURI	No	Notification endpoint definition
callbackData	xsd:string	Yes	Data the application can register with the server when subscribing to notifications, and that are passed back unchanged in each of the related notifications.
notificationFormat	common:NotificationFormat	Yes	Application can specify format of the resource representation in notifications that are related to this subscription. The choice is between {XML, JSON}. Default: XML
duration	xsd:int	Yes	Period of time (in seconds) notifications are provided for. If set to “0” (zero), a default duration time, which is specified by the service policy, will be used. If the parameter is omitted, the notifications will continue until the maximum duration time, which is specified by the service policy, unless the notifications are stopped by deletion of subscription for notifications. This element MAY be given by the client during resource creation in order to signal the

			desired lifetime of the subscription. The server SHOULD return in this element the period of time for which the subscription will still be valid.
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This field SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids re-sending the message in such situations. In case the field is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.

If the operation was successful, it returns an HTTP Status of “201 Created”.

C.1.1 Example (Informative)

C.1.1.1 Request

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

notifyURL=http%3A%2F%2Fapplication.example.com%2Ffiletransfer%2Fnotifications%2F77777&
callbackData=abcd&
duration=7200&
clientCorrelator=12345
```

C.1.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSubscription xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <callbackReference>
    <notifyURL>http://application.example.com/filetransfer/notifications/77777</notifyURL>
     <callbackData>abcd</callbackData>
  </callbackReference>
 <duration>7200</duration>
```

```
  <clientCorrelator>12345</clientCorrelator>
  <resourceURL>http://exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001</resourceURL>
</ft:fileTransferSubscription>
```

C.2 Creating a new file transfer session

This operation is used to create a new file transfer session. See section 6.3.5.

The request parameters are as follows:

Name	Type/Values	Optional	Description
originatorAddress	xsd:anyURI	No	The address (e.g. 'sip' URI, 'tel' URI, 'acr' URI) of the Originator
originatorName	xsd:string	Yes	Human readable name of the Originator
receiverAddress	xsd:anyURI	No	Address of the Receiver of this file transfer session(e.g. 'sip' URI, 'tel' URI, 'acr' URI)
receiverName	xsd:string	Yes	Name of the Receiver of this file transfer session
name	xsd:string[1..unbounded]	No	The name of the file. See [IR.79] & [RFC5547]. Note: in a file transfer session, the file name is unique.
type	xsd:string[1..unbounded]	No	The MIME type of the file. It is concatenated by type, “/” and subtype. See [IR.79] & [RFC5547].
size	xsd: unsignedLong[1..unbounded]	Yes	The size of the file in octets. See [IR.79] & [RFC5547].
algorithm	xsd:string[1..unbounded]	Yes	The hash algorithm used (only "sha-1" currently supported). See [RFC5547].
value	xsd:hexBinary[1..unbounded]	Yes	The hash value of the file. See [RFC5547].
fileDisposition	FileDisposition[1..unbounded]	Yes	It is used by the file sender to indicate a preferred disposition of the file. To indicate that a file should be automatically rendered, the value is “render”. To indicate that a file should not be automatically rendered, the value is “attachment”. See [RFC5547]. Default value is “attachment”

fileDescription	xsd:string[1..unbounded]	Yes	Human-readable short description of the file (corresponding to 'i=' line in SDP) which could be set by the Originator. See [IR.79] & [RFC2327].
cDate	xsd:dateTime[1..unbounded]	Yes	The date on which the file was last created. See [RFC5547].
mDate	xsd:dateTime[1..unbounded]	Yes	The date on which the file was last modified. See [RFC5547].
rDate	xsd:dateTime[1..unbounded]	Yes	The date on which the file was last read. See [RFC5547].
fileIcon	xsd:anyURI[1..unbounded]	Yes	It is useful with certain file types such as images. It allows the file sender to include a pointer to a body that includes a small preview icon representing the contents of the file to be transferred, the file Receiver can use to determine whether it wants to receive such file. The 'file-icon' contains a Content-ID URL, [RFC2392] pointing to an additional body that contains the actual icon in a MIME multipart/related body See [RFC5547] &requirement of RCAPI-RCFT-001 in [RC API RD]
fileURL	xsd:anyURI[1..unbounded]	Yes	The URL link to actual file content. When this field is used in POST operation during resource creation of Create a new 1-1 file transfer session, it is the external file repository URL set by Originator. If it is present, it indicates that there is no file content included in the request operation. If it is not present, it indicates that the actual file content is included in the HTTP request body during resource creation of Create a new 1-1 file transfer session or Send file. File content can be represented as multipart/form-data entity bodies, where the first entry of the form are the root fields and the second entry of the form are the file content. When this field is used in file content notification, it is a URL link to actual file content where the client can download the file. This field can be used in response of Create

			file transfer session, Retrieve file transfer session information, Send file. It is the file content URL set by server. This field is not used in session invitation notifications.
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This field SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids re-sending the message in such situations. In case the field is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.

If the operation was successful, it returns an HTTP Status of “201 Created”.

C.2.1 Example (Informative)

C.2.1.1 Request

```
POST /exampleAPI/file transfer/v1/tel%3A%2B19585550100/sessions/
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com
MIME-Version: 1.0
Content-Length: nnnn
Accept: application/xml


--===============123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn

originatorAddress=tel%3A%2B19585550100&
originatorName= Max Muster&
receiverAddress=tel%3A%2B19585550102&
receiverName= Peter E. Xample&
name=sunset.jpg&
type=image/jpeg&
size=4096&
algorithm=sha-1&
value=58231FE8653BBCF371362F86D471913EE4B1DF2F &
fileDisposition=Attachment&
fileDescription=This is my latest picture&
```

```
cDate=2011-08-21T00:00:00-04:00&
fileIcon=cid:id3@alicepc.example.com&
clientCorrelator=12345

--===============123456==
Content-Disposition: form-data; name="attachments"
Content-Type: multipart/mixed; boundary="====aaabbb"
--====aaabbb
Content-Disposition:attachment; filename="icon"
Content-Type: image/gif
Content-ID: cid:id3@alicepc.example.com

[..small preview icon...]
--====aaabbb--


Content-Disposition:attachment;filename="sunset.jpg"
Content-Type: image/jpg

JPEG ...binary image data...
--====aaabbb--
--===============123456==--
```

C.2.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001
Content-Length: nnnn
Date: Mon, 28 Jul 2011 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileTransferSessionInformation xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <originatorAddress>tel:+19585550100</originatorAddress>
  <originatorName>Max Muster</originatorName>
  <receiverAddress>tel:+19585550102</receiverAddress>
  <receiverName>Peter E. Xample</receiverName>
  <status>Invited</status>
  <fileInformation>
    <fileSelector>
      <name>sunset.jpg</name>
      <type>image/jpeg</type>
    </fileSelector>
    <fileURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001.jpg</fileURL>
    <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001</resourceURL>
  </fileInformation>
  <clientCorrelator>104567</clientCorrelator>
  <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001</resourceURL>
</ft:fileTransferSessionInformation>
```

C.3 Accepting a file transfer session invitation

This operation is used to accept a file transfer session invitation, see section 6.5.5.

The request parameters are as follows:

Name	Type/Values	Optional	Description
status	ReceiverStatus	No	Status of the Receiver. To indicate that the Receiver accepts the session invitation, this element MUST be set to “Connected”.
acceptedFile	xsd:anyURI [0..unbounded]	Yes	Accepted files Receiver feedback of the list of files accepted for transfer, containing a list of resourceURL values received in the “fileInformation” child element of “SessionInvitationNotification”. Note: According to [RC API RD], there is only one file. But in SIMPLE IM File transfer [OMA-SIMPLE_IM], there may be multiple files. If there is no acceptedFile meanings Receiver client reject Originator media offer.

If the operation was successful, it returns an HTTP Status of “200 OK”.

C.3.1 Example (Informative)

C.3.1.1 Request

```
POST /exampleAPI/file transfer/v1/tel%3A%2B19585550102/sessions/sess001/status
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn
Accept: application/xml
Host: example.com

status=Connected&
acceptedFile= http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001
```

C.3.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

C.4 Sending a file

This operation is used to send a file, see section 6.6.5.

The request parameters are as follows:

Name	Type/Values	Optional	Description
name	xsd:string	No	The name of the file. See [IR.79] & [RFC5547]. Note: in a file transfer session, the file name is unique.
type	xsd:string	No	The MIME type of the file. It is concatenated by type, "/" and subtype. See [IR.79] & [RFC5547].
size	xsd: unsignedLong	Yes	The size of the file in octets. See [IR.79] & [RFC5547].
algorithm	xsd:string	Yes	The hash algorithm used (only "sha-1" currently supported). See [RFC5547].
value	xsd:hexBinary	Yes	The hash value of the file. See [RFC5547].
fileDisposition	FileDisposition	Yes	It is used by the file sender to indicate a preferred disposition of the file. To indicate that a file should be automatically rendered, the value is "render". To indicate that a file should not be automatically rendered, the value is "attachment". See [RFC5547]. Default value is "attachment".
fileDescription	xsd:string	Yes	Human-readable short description of the file (corresponding to 'i=' line in SDP) which could be set by the Originator. See [IR.79] & [RFC2327].
cDate	xsd:dateTime	Yes	The date on which the file was last created. See [RFC5547].
mDate	xsd:dateTime	Yes	The date on which the file was last modified. See [RFC5547].
rDate	xsd:dateTime	Yes	The date on which the file was last read. See [RFC5547].

fileIcon	xsd:anyURI	Yes	It is useful with certain file types such as images. It allows the file sender to include a pointer to a body that includes a small preview icon representing the contents of the file to be transferred, the file Receiver can use to determine whether it wants to receive such file. The 'file-icon' contains a Content-ID URL, [RFC2392] pointing to an additional body that contains the actual icon in a MIME multipart/related body. See [RFC5547] &requirement of RCAPI-RCFT-001 in [RC API RD].

If the operation was successful, it returns an HTTP Status of "200 OK".

C.4.1 Example (Informative)

C.4.1.1 Request

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files HTTP/1.1
Content-Type: multipart/form-data; boundary="==========123456==";
Host: example.com
MIME-Version: 1.0
Accept: application/xml
Content-Length: nnnn


--==========123456==
Content-Disposition: form-data; name="root-fields"
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn

name=sunset.jpg&
type=image/jpeg&
size=4096&
algorithm=sha-1&
value=58231FE8653BBCF371362F86D471913EE4B1DF2F

--==========123456==
Content-Disposition: form-data; name="attachments", filename="sunset.jpg"
Content-Type: image/jpeg

JPEG ...binary image data...
--==========123456==--
```

C.4.1.2 Response

```
HTTP/1.1 201 Created
Content-Type: application/xml
```

```
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<ft:fileInformation xmlns:ft="urn:oma:xml:rest:netapi:filetransfer:1">
  <fileSelector>
    <name>sunset.jpg</name>
   <type>image/jpeg</type>
  </fileSelector>
  <fileURL> http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001.jpg</fileURL>
  <resourceURL>http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001</resourceURL>
</ft:fileInformation>
```

Appendix D. JSON examples (Informative)

JSON (JavaScript Object Notation) is a lightweight, text-based, language-independent data interchange format. It provides a simple means to represent basic name-value pairs, arrays and objects. JSON is relatively trivial to parse and evaluate using standard JavaScript libraries, and hence is suited for REST invocations from browsers or other processors with JavaScript engines. Further information on JSON can be found at [RFC4627].

The following examples show the request and response for various operations using the JSON data format. The examples follow the XML to JSON serialization rules in [REST_NetAPI_Common]. A JSON response can be obtained by using the content type negotiation mechanism specified in [REST_NetAPI_Common].

For full details on the operations themselves please refer to the section number indicated.

D.1 Reading all active file transfer notification subscriptions (section 6.1.3.1)

Request:

```
GET /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

{"fileTransferSubscriptionList": {
   "fileTransferSubscription": {
      "callbackReference": {
         "callbackData": "abcd",
         "notifyURL": "http://application.example.com/filetransfer/notifications/77777"
      },
      "clientCorrelator": "12345",
       "duration":"7200",
      "resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001"
   },
   "resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions"
}}
```

D.2 Creating a new subscription to file transfer notifications, response with copy of created resource (section 6.1.5.1)

Request:

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/ HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
```

```
Accept: application/json
Host: example.com

{"fileTransferSubscription": {
   "callbackReference": {
      "callbackData": "abcd",
      "notifyURL": "http://application.example.com/filetransfer/notifications/77777"
   },
   "clientCorrelator": "12345"
   "duration":"7200"
}}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

{"fileTransferSubscription": {
   "callbackReference": {
      "callbackData": "abcd",
      "notifyURL": "http://application.example.com/filetransfer/notifications/77777"
   },
   "clientCorrelator": "12345",
   "duration":"7200",
   "resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001"
}}
```

D.3 Creating a new subscription to file transfer notifications, response with location of created resource (section 6.1.5.2)

Request:

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/ HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"fileTransferSubscription": {
   "callbackReference": {
      "callbackData": "abcd",
      "notifyURL": "http://application.example.com/filetransfer/notifications/77777"
   },
   "clientCorrelator": "12345"
    "duration":"7200"
}}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

{"resourceReference": {"resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001"}}
```

D.4 Reading an individual subscription (section 6.2.3.1)

Request:

```
GET /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001?resFormat=XML HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"fileTransferSubscription": {
    "callbackReference": {
        "callbackData": "abcd",
        "notifyURL": "http://application.example.com/filetransfer/notifications/77777"
    },
    "clientCorrelator": "12345",
    "duration":"7200",
    "resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001"
}}
```

D.5 Cancelling a subscription (section 6.2.6.1)

Request:

```
DELETE /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jun 2010 17:51:59 GMT
```

D.6 Creating a new 1-1 file transfer session with file content (section 6.3.5.1)

Request:

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: multipart/form-data; boundary="==========123456==";
Host: example.com
MIME-Version: 1.0
Content-Length: nnnn
Accept: application/json


--==========123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/json
Content-Length: nnnn

{"fileTransferSessionInformation": {
   "clientCorrelator": "104567",
   "fileInformation": {
      "fileDate": {"cDate": "2011-08-21T00:00:00-04:00"},
      "fileDescription": "This is my latest picture",
      "fileDisposition": "Attachment",
      "fileIcon": "cid:id3@alicepc.example.com",
      "fileSelector": {
         "hash": {
            "algorithm": "sha-1",
            "value": "58231FE8653BBCF371362F86D471913EE4B1DF2F"
         },
         "name": "sunset.jpg",
         "size": "4096",
         "type": "image/jpeg"
      }
   },
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Max Muster",
   "receiverAddress": "tel:+19585550102",
   "receiverName": "Peter E. Xample"
}}

--==========123456==
Content-Disposition: form-data; name=”attachments”
Content-Type: multipart/mixed; boundary=”====aaabbb”
--====aaabbb
Content-Disposition:attachment; filename=”icon”
Content-Type: image/gif
Content-ID: <id3@alicepc.example.com>

[..small preview icon...]
--====aaabbb--


Content-Disposition:attachment;filename=”sunset.jpg”
```

```
Content-Type: image/jpg

JPEG ...binary image data...
--====aaabbb--
--===============123456==--
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

{"fileTransferSessionInformation": {
   "clientCorrelator": "104567",
   "fileInformation": {
      "fileSelector": {
         "name": "sunset.jpg",
         "type": "image/jpeg"
      },
      "fileURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001.jpg",
      "resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001"
   },
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Max Muster",
   "receiverAddress": "tel:+19585550102",
   "receiverName": "Peter E. Xample",
   "resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001",
   "status": "Invited"
}}
```

D.7 Creating a new 1-1 file transfer session with external file repository URL (section 6.3.5.2)

Request:

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com
MIME-Version: 1.0
Content-Length: nnnn
Accept: application/json


--===============123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/json
Content-Length: nnnn
```

```
{"fileTransferSessionInformation": {
   "clientCorrelator": "104567",
   "fileInformation": {
      "fileDate": {"cDate": "2011-08-21T00:00:00-04:00"},
      "fileDescription": "This is my latest picture",
      "fileDisposition": "Attachment",
      "fileIcon": "cid:id3@alicepc.example.com",
      "fileSelector": {
         "hash": {
            "algorithm": "sha-1",
            "value": "58231FE8653BBCF371362F86D471913EE4B1DF2F"
         },
         "name": "sunset.jpg",
         "size": "4096",
         "type": "image/jpeg"
      },
      "fileURL": "http://alicepc.example.com/sunset.jpg"
   },
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Max Muster",
   "receiverAddress": "tel:+19585550102",
   "receiverName": "Peter E. Xample"
}}

--===============123456==
Content-Disposition: form-data; name=”attachments”, filename=”icon”
Content-Type: image/gif
Content-Transfer-Encoding: binary
Content-ID: <id3@alicepc.example.com>
Content-Length: [length of image]
 [..small preview icon...]

--===============123456==--
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

{"fileTransferSessionInformation": {
   "clientCorrelator": "104567",
   "fileInformation": {
      "fileSelector": {
         "name": "sunset.jpg",
         "type": "image/jpeg"
      },
     "fileURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001.jpg",
"resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001 "
   },
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Max Muster",
   "receiverAddress": "tel:+19585550102",
```

```
   "receiverName": "Peter E. Xample",
   "resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001",
   "status": "Invited"
}}
```


D.8 Creating a new 1-1 file transfer session with external file repository URL without icon, response with location of created resource (section 6.3.5.3)

Request:


```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions HTTP/1.1
Content-Type: application/xml
Content-Length: nnnn
Host: example.com
Accept: application/xml

{"fileTransferSessionInformation": {
   "clientCorrelator": "104567",
   "fileInformation": {
      "fileDate": {"cDate": "2011-08-21T00:00:00-04:00"},
      "fileDescription": "This is my latest picture",
      "fileDisposition": "Attachment",
      "fileSelector": {
         "hash": {
            "algorithm": "sha-1",
            "value": "58231FE8653BBCF371362F86D471913EE4B1DF2F"
         },
         "name": "sunset.jpg",
         "size": "4096",
         "type": "image/jpeg"
      },
      "fileURL": "http://alicepc.example.com/sunset.jpg"
   },
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Max Muster",
   "receiverAddress": "tel:+19585550102",
   "receiverName": "Peter E. Xample"
}}
```


Response:


```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001
Content-Length: nnnn
Date: Thu, 28 Jul 2011 17:51:59 GMT

{"resourceReference": {"resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001"}}
```

D.9 Retrieving file transfer session information (section 6.4.3.1)

Request:

```
GET /exampleAPI/file transfer/v1/tel%3A%2B19585550100/sessions/sess001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"fileTransferSessionInformation": {
   "clientCorrelator": "104567",
   "fileInformation": {
      "fileDate": {"cDate": "2011-08-21T00:00:00-04:00"},
      "fileDescription": "This is my latest picture",
      "fileDisposition": "Attachment",
      "fileIcon": "cid:id3@alicepc.example.com",
      "fileSelector": {
         "hash": {
            "algorithm": "sha-1",
            "value": "58231FE8653BBCF371362F86D471913EE4B1DF2F"
         },
         "name": "sunset.jpg",
         "size": "4096",
         "type": "image/jpeg"
      },
      "fileURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001.jpg",
      "resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001 "
   },
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Max Muster",
   "receiverAddress": "tel:+19585550102",
   "receiverName": "Peter E. Xample",
   "resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001",
   "status": "Connected"
}}
```

D.10 Cancel file transfer invitation/ Decline file transfer invitation/ Terminating a file transfer session (section 6.4.6.1)

Request:

```
DELETE /exampleAPI/file transfer/v1/tel%3A%2B19585550100/sessions/sess001 HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Mon, 28 Jun 2010 17:51:59 GMT
```

D.11 Accepting a 1-1 file transfer invitation (section 6.5.5.1)

Request:

```
POST /exampleAPI/file transfer/v1/tel%3A%2B19585550102/sessions/sess001/status HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Accept: application/json
Host: example.com

{"receiverSessionStatus": {
   "acceptedFile": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001",
   "status": "Connected"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.12 Sending a file, using tel URI and returning the copy of the created resource (section 6.6.5.1)

Request:

```
POST /exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files HTTP/1.1
Content-Type: multipart/form-data; boundary="================123456==";
Host: example.com
MIME-Version: 1.0
Content-Length: nnnn
Accept: application/json


--================123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/json
Content-Length: nnnn

{"fileInformation": {"fileSelector": {
   "hash": {
      "algorithm": "sha-1",
      "value": "58231FE8653BBCF371362F86D471913EE4B1DF2F "
   },
   "name": "sunset.jpg",
   "size": "4096",
   "type": "image/jpeg"
```

```
}}}

--===============123456==
Content-Disposition: form-data; name=”attachments”, filename=”sunset.jpg”
Content-Type: image/jpeg

JPEG ...binary image data...
--===============123456==--
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"fileInformation": {
   "fileSelector": {
      "name": "sunset.jpg",
      "type": "image/jpeg"
   },
   "fileURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001.jpg",
   "resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001"
}}
```

D.13 Sending a file, using ACR and returning a location of the created resource (section 6.6.5.2)

Request:

```
POST /exampleAPI/filetransfer/v1/acr%3Apseudonym123/sessions/sess001/files HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com
MIME-Version: 1.0
Content-Length: nnnn
Accept: application/json


--===============123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/json
Content-Length: nnnn

{"fileInformation": {"fileSelector": {
   "name": "sunset.jpg",
   "type": "image/jpeg"
}}}
```

```
--===============123456==
Content-Disposition: form-data; name=”attachments”, filename=”sunset.jpg”
Content-Type: image/jpeg

JPEG ...binary image data...
--===============123456==--
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://example.com/exampleAPI/filetransfer/v1/acr%3Apseudonym123/sessions/sess001/files/file001
Content-Length: nnnn
Date: Mon, 28 Jun 2010 17:51:59 GMT

{"resourceReference": {"resourceURL":
"http://example.com/exampleAPI/filetransfer/v1/acr%3Apseudonym123/sessions/sess001/files/file001"}}
```

D.14 Notify a client about file transfer session invitations (section 6.7.5.1)

Request:

```
POST /filetransfer/notifications/77777 HTTP/1.1
Content-Type: multipart/form-data; boundary="===============123456==";
Host: example.com
MIME-Version: 1.0
Content-Length: nnnn


--===============123456==
Content-Disposition: form-data; name=”root-fields”
Content-Type: application/json
Content-Length: nnnn

{"sessionInvitationNotification": {
   "callbackData": "abcd",
   "fileInformation": {
      "fileDate": {"cDate": "2011-08-21T00:00:00-04:00"},
      "fileDescription": "This is my latest picture",
      "fileDisposition": "Attachment",
      "fileIcon": "cid:id3@alicepc.example.com",
      "fileSelector": {
         "hash": {
            "algorithm": "sha-1",
            "value": "58231FE8653BBCF371362F86D471913EE4B1DF2F"
         },
         "name": "sunset.jpg",
         "size": "4096",
         "type": "image/jpeg"
```

```
      },
      "resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001"
   },
   "link": {
      "href": " http://example.com/exampleAPI/file transfer/v1/tel%3A%2B19585550100/sessions/sess001",
      "rel": "FileTransferSessionInformation"
   },
   "originatorAddress": "tel:+19585550100",
   "originatorName": "Max Muster",
   "receiverAddress": "tel:+19585550102",
   "receiverName": "Peter E. Xample"
}}
--===============123456==
Content-Disposition: form-data; name=”attachments”, filename=”icon”
Content-Type: image/gif
Content-ID: <id3@alicepc.example.com>

[..small preview icon...]

--===============123456==--
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.15 Notify a client about file transfer session events (section 6.8.5.1)

Request:

```
POST /filetransfer/notifications/77777 HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Host: application.example.com

{"fileTransferEventNotification": {
   "callbackData": "abcd",
   "eventDescription": "The session has ended.",
   "eventType": "SessionEnded",
   "link": [
      {
         "href": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001",
         "rel": "FileTransferSessionInformation"
      },
      {
         "href": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001",
         "rel": "FileTransferSubscription"
      }
   ]
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.16 Notify a client about file content link (section 6.9.5.1)

Request:

```
POST /filetransfer/notifications/77777 HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Host: application.example.com

{"fileNotification": {
   "callbackData": "abcd",
   "fileInformation": {
      "fileSelector": {
         "name": "sunset.jpg",
         "type": "image/jpeg"
      },
      "fileURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001.jpg",
      "resourceURL": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001"
   },
   "link": {
      "href": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550102/subscriptions/sub001",
      "rel": "FileTransferSubscription"
   }
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.17 Client notification about Receiver acceptance (section 6.10.5.1)

Request:

```
POST /filetransfer/notifications/77777 HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Host: application.example.com

{"receiverAcceptanceNotification": {
   "callbackData": "abcd",
   "link": [
      {
         "href": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001",
         "rel": "FileTransferSessionInformation"
      },
      {
         "href": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001",
```

```
            "rel": "FileTransferSubscription"
        }
    ],
    "receiverAddress": "tel:+19585550102",
    "receiverName": "Peter E. Xample",
    "receiverSessionStatus": {
        "acceptedFile": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/sessions/sess001/files/file001",
        "status": "Connected"
    }
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

D.18 Client notification about subscription cancellation (section 6.11.5.1)

Request:

```
POST /filetransfer/notifications/77777 HTTP/1.1
Content-Type: application/json
Content-Length: nnnn
Host: application.example.com

{"subscriptionCancellationNotification": {
    "callbackData": "abcd",
    "link": {
        "href": "http://example.com/exampleAPI/filetransfer/v1/tel%3A%2B19585550100/subscriptions/sub001",
        "rel": "FileTransferSubscription "
    }
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 28 Jul 2010 02:51:59 GMT
```

Appendix E. Operations mapping to a pre-existing baseline specification (Informative)

As this specification does not have a baseline specification, this appendix is empty.

Appendix F. Light-weight resources (Informative)

As this version of the specification does not define any light-weight resources, this Appendix is empty.

Appendix G. Authorization aspects (Normative)

This appendix specifies how to use the RESTful File Transfer API in combination with some authorization frameworks.

G.1 Use of Autho4API

The RESTful File Transfer API MAY support the Autho4API authorization framework defined in [Autho4API_10].

A RESTful File Transfer API supporting Autho4API:

- SHALL conform to section D.1 of [REST_NetAPI_Common];
- SHALL conform to this section G.1.

G.1.1 Scope values

G.1.1.1 Definitions

An Autho4API Authorization Server serving Autho4API Clients requests for getting authorized access to the resources exposed by the RESTful File Transfer API:

- SHALL support the scope values defined in Table 1 below;
- MAY support scope values not defined in this specification.

Scope value	Description	For one-time access token
oma_rest_filetransfer.all_{apiVersion}	Provide access to all defined operations on the resources in this version of the API. The {apiVersion} part of this identifier SHALL have the same value as the “apiVersion” URL variable which is defined in section 5.1. This scope value is the union of the other scope values listed in next rows of this table.	No
oma_rest_filetransfer.sessions	Provide access to all defined operations on handling of 1-1 file transfer sessions	No
oma_rest_filetransfer.subscr	Provide access to all defined operations on handling of file transfer subscriptions	No
oma_rest_filetransfer.files	Provide access to all defined operations on handling 1-1 file transfer files	No

Table 1 Autho4API scope values for RESTful File Transfer API

G.1.1.2 Downscoping

In the case where the Autho4API client requests authorization for “ oma_rest_filetransfer.all_{apiVersion}” scope, the Autho4API Authorization Server and/or resource owner MAY restrict the granted scope to some of the following scope values:

- “oma_rest_filetransfer.sessions”

- “oma_rest_filetransfer.subscr”
- “oma_rest_filetransfer.files”

G.1.1.3 Mapping with resources and methods

Tables 2, 3 and 4 of this section specify the scope values required for an AuthoAPI Client to request which SHALL represent authorized access to the REST resources and methods of the RESTful File Transfer API. defined in. In these tables, the root "oma_rest_filetransfer." of scope values is omitted for readability reasons.

Resource	URL Base URL: http://{serverRoot}/filetransfer/{apiVersion}/{userId}	Section refe-rence	HTTP verbs			
			GET	PUT	POST	DELETE
All 1-1 file transfer sessions	/sessions	6.3	n/a	n/a	all_{apiVersion} or sessions	n/a
Individual 1-1 file transfer session	/sessions/{sessionId}	6.4	all_{apiVersion} or sessions	n/a	n/a	all_{apiVersion} or sessions
Session Status	/sessions/{sessionId}/status	6.5	n/a	n/a	all_{apiVersion} or in_regist	n/a

Table 2 Required scope values for: Handling of 1-1 file transfer sessions

Resource	URL Base URL: http://{serverRoot}/filetransfer/{apiVersion}/{userId}	Section refe-rence	HTTP verbs			
			GET	PUT	POST	DELETE
All subscriptions to file transfer notifications	/subscriptions	6.1	all_{apiVersion} or subscr	n/a	all_{apiVersion} or subscr	n/a
Individual subscription to file transfer notifications	/subscriptions/{subscriptionId}	6.2	all_{apiVersion} or subscr	n/a	n/a	all_{apiVersion} or subscr

Table 3 Required scope values for: Handling of file transfer subscriptions

Resource	URL Base URL: http://{serverRoot}/filetransfer/{apiVersion}/{userId}/sessions/{sessionId}	Section refe-rence	HTTP verbs			
			GET	PUT	POST	DELETE

Files in a 1-1 file transfer session	/files	6.6	n/a	n/a	all_{apiVersion} or files	n/a
Individual file	/files/{fileId}	6.7	n/a	n/a	n/a	n/a

Table 4 Required scope values for: Handling 1-1 file transfer files

G.1.2 Use of 'acr:Authorization'

This section specifies the use of 'acr:Authorization' in place of an end user identifier in a resource URL path.

An 'acr' URI of the form 'acr:Authorization', where 'authorization' is a reserved keyword MAY be used to avoid exposing a real end user identifier in the resource URL path.

A client MAY use 'acr:Authorization' in a resource URL in place of the{userId} resource URL variable in the resource URL path, when the RESTful File Transfer API is used in combination with [Autho4API_10].

In the case the RESTful File Transfer API supports [Autho4API_10], the server:

- SHALL accept 'acr:Authorization' as a valid value for the resource URL variable {userId}.
- SHALL conform to [REST_Common_TS] section 5.8.1.1 regarding the processing of 'acr:Authorization'.