GridOps Management Suite 3.10

File Uploading Interface

Functional Specification

Document Version: 1.0

Updated: June, 2024

Table of Contents

1. REFERENCES 6
2. ASSUMPTIONS 7
3. INTRODUCTION 8
3.1. General Architecture 8
4. INTERFACE OVERVIEW 9
5. SENDFILENOTIFICATION SERVICE 10
5.1. CreatedFileNotification Operation 10
5.1.1. Overview 10
5.1.2. Use Cases 12
6. MESSAGES 14
6.1. Common 14
6.1.1. Header 14
6.1.2. Reply and Fault 16
6.2. CreatedFileNotification Operation 17
6.2.1. Request 17
6.2.2. Response 20
6.2.3. Fault 20
7. DEPLOYMENT SPECIFICATION 21
8. INTERFACE CONFIGURATION 22
9. APPENDIX 23
9.1. WSDL 23
10. RELEASE NOTES 24
10.1. Software Version 3.8.0 24
10.2. Software Version 3.9 24
11. DEFINITIONS AND ABBREVIATIONS 25

Table of Figures

Figure 4.1 – The File Upload Integration use case diagram........ 9
Figure 5.1 – The CreatedFileNotification operation execution 11
Figure 6.1 – The header field........ 16
Figure 6.2 – The **Reply** and **Error** field contents........ 17
Figure 6.3 – The CreatedFileNotificationEvent message 18
Figure 6.4 – FileNotification.xsd........ 18
Figure 6.5 – The FileNotificationResponse message........ 20
Figure 6.6 – The FileNotificationFault message 20
Figure 7.1 – File Upload Integration deployment diagram........ 21

Table of Tables

Table 5.1 – The FileUpload operation use cases 12
Table 6.1 – The FileNotification attributes description 19
Table 7.1 – The deployment specification 21
Table 8.1 – The configuration files specification 22

Table of Documents

No table of figures entries found.

1. REFERENCES

#	Title	Description
1.	EcoStruxure GridOps Management Suite 3.10 Enterprise Integration Platform - Functional Specification	The document represents a set of common integration principles applied to all baseline integration adapters.
2.	EcoStruxure GridOps Management Suite 3.10 File Uploading Interface	EcoStruxure GridOps Management Suite 3.10 File Upload Interface zip file contains essential configuration information, as well as web service definitions.

2. ASSUMPTIONS

The File Upload Integration is developed under following assumptions:

- EcoStruxure GridOps has the possibility to send various data contained in a file (format of the file depends on the type of system that is being integrated) to external systems via predefined shared location.
- Housekeeping of remote (external) file share is done by the source system.
- Housekeeping of local file share is done by the File Upload Adapter.
- Separate local folder shall be used for each extract type.
- File Upload adapter calculates the hash value of the file that should be uploaded and send that value through the Hash field in the claim check notification message. Sending claim check notification message to notify external system about the file upload is optional.
- File Upload Adapter supports SHA256 or SHA512 hash algorithm types – depends on configuration.
- Both SFTP and CIFS (SMB) file share types are supported.

3. INTRODUCTION

EcoStruxure GridOps Management Suite is a family of solutions designed to help electric utilities in the operations and management of their grid. It is offered as EcoStruxure ADMS, EcoStruxure Grid Operation, EcoStruxure DERMS or EcoStruxure Energy Transmission Operation solutions, which share the same technology platform.

NOTE: The functionality described in this document applies to the following solutions: EcoStruxure ADMS, EcoStruxure Grid Operation and EcoStruxure DERMS.

NOTE: Most images presented in this document are related to the EcoStruxure ADMS solution and should be used as an example. The images for other solutions may differ slightly.

Depending on the quantity of the data being exchanged between integrated systems, there can be three approaches of data transferring, from the source to the destination system:

- In case of the small data quantity, the data is transferred as a payload within the message.
- In case of the large data quantity, the claim check pattern is used where the data is exported to a file and the target system is notified about the file location in the message.
- If the source side cannot implement claim check pattern, the file monitoring can be used. Data is exported to a file and uploaded to predefined shared location. Target system monitors mentioned location and consume new files.

When extract file of potentially large file size needs to be exchanged, it is more efficient that EcoStruxure GridOps first prepares the file on local file share and let the FILE Upload Adapter uploads mentioned file to external file share location (File Share), for external system to consume it. Process in charge for copying data file(s) from the EcoStruxure GridOps local file share to external shared location is named File Upload Adapter.

3.1. General Architecture

It is thoroughly described in the *EcoStruxure GridOps Management Suite 3.10 Enterprise Integration Platform - Functional Specification* [1].

4. INTERFACE OVERVIEW

File Upload Integration is implemented through the File Upload Adapter component. The aforementioned adapter implements file location monitor component which tracks the record of newly uploaded files to a predefined local shared location. After new file is stored on predefined location, the File Upload adapter copies it to the external file share and sends the claim check notification message. The following operations are implemented within the File Upload Adapter and corresponding web service client:

- SendFileNotificationService – used for sending file notifications:
    - CreatedFileNotification operation.

NOTE: Sending of file notification is optional, since some of the external utility' systems may not implement the appropriate web service.

The following chapters provide more details regarding briefly described interface above, along with the service operation, error handling scenarios, etc.

The use case diagram that represents common participants (actors) and users of the aforementioned interface in the File Upload Integration is given in Figure 4.1.

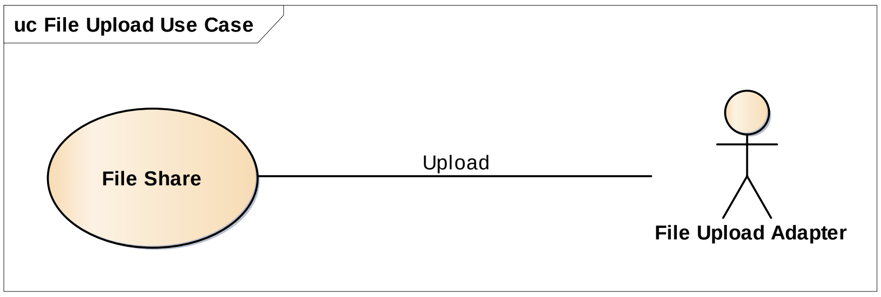

Figure 4.1 – The File Upload Integration use case diagram

5. SENDFILENOTIFICATION SERVICE

5.1. CreatedFileNotification Operation

5.1.1. Overview

The EcoStruxure GridOps users have two possibilities for sending extract files to client's corporate systems:

- Synchronously – via WCF service exposed by File Upload Adapter. In this approach, files are sent as serialized data via dedicated WCF service. Entire process is synchronous and end user will receive a notification about the process outcome (whether file was successfully uploaded to external shared location or not).
- Asynchronously – via shared location monitored by File Upload Adapter. In this approach, there are several steps that need to be covered within the specific file upload process. These steps are the following:
    - An appropriate service starts copying the file with a temporary extension (.temp). After the file is fully copied, service should remove the temporary extension (.temp).
    - Such action will trigger the File Upload Adapter which monitors the LFS.
    - File Upload Adapter fetches the published extract file from the LFS and uploads it to external file share.
    - Optionally, the appropriate notification message (CreatedFileNotificationEvent) is sent to external system, notifying it about the file upload.
    - Information about file names that are already processed are persisted in the File Upload Adapter internal memory (Adapter's Smart Cache) in order not to process same file multiple times and not to lose this information in case of service failure since information is replicated between Smart Caches on Hot and Stand By server. In order to accomplish this, file names need to be unique. Therefore, each extract file (zip or single extract file) needs to have timestamp as part of file name. When the file is deleted from the LFS, the filename is removed from the list of processed files persisted in Smart Cache of the File Upload Adapter in order to avoid keeping information that is not needed.

In both situations, utility is responsible for implementing next steps which include consuming of the file.

The high-level sequence diagram is given in Figure 5.1.

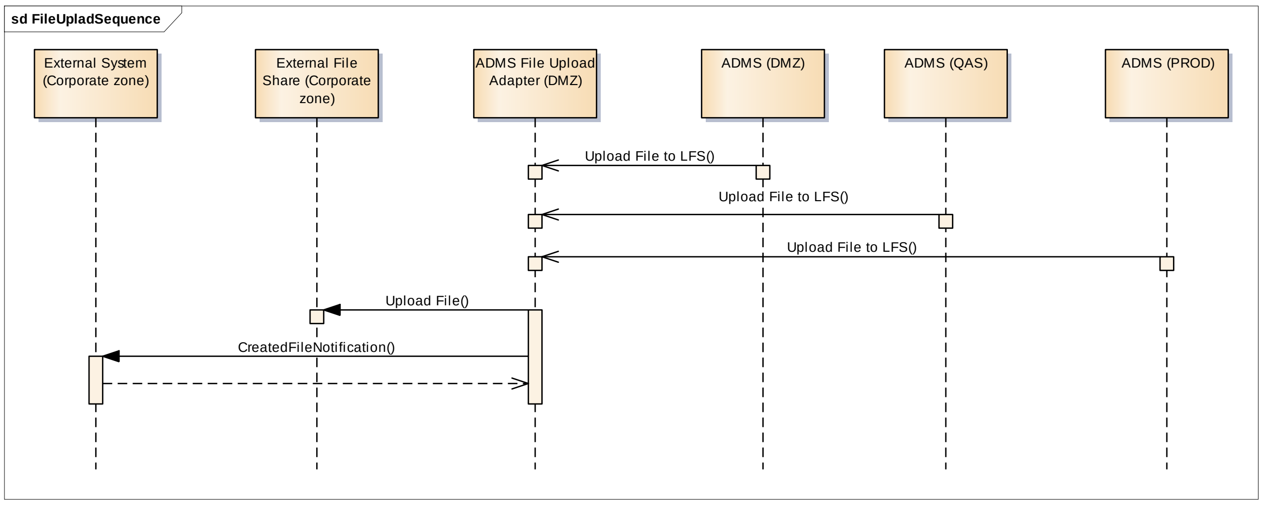

Figure 5.1 – The CreatedFileNotification operation execution

5.1.2. Use Cases

The list of possible use cases and corresponding faults is given in Table 5.1.

Table 5.1 – The FileUpload operation use cases

Use Case	Message Mapping			Action
	Property	Type	Value	
LFS not available	Result	String	FAILED	File Upload Adapter cannot access the local file share. Error is written to a log file and event is sent by File Upload Adapter.
	Error.code	String	5.3	
	Error.level	String	FATAL	
	Error.reason	String	UnavailableLocalFileShare	
	Error.details	String	Could not copy file: {0} from local file share since it is unavailable.	
Failed to copy extract	Result	String	FAILED	File Upload Adapter consumes the file from local file share, and tries to copy it to external file share configurable number of times. After configurable number of attempts, error is written to a log file and event is sent by File Upload Adapter.
	Error.code	String	N/A	
	Error.level	String	FATAL	
	Error.reason	String	UnavailableExternalFileShare	
	Error.details	String	Could not copy file: {0} to: {1} since FS is unavailable.	
Failed to send message to web service	Result	String	FAILED	File Upload Adapter consumes the file from local file share, and copies it successfully to external file share. After that, File Upload Adapter tries to send the message to externally hosted web service for configurable number of times. After configurable number of attempts, error is written to a log file and event is sent by File Upload Adapter.
	Error.code	String	N/A	
	Error.level	String	FATAL	
	Error.reason	String	UnavailableWebService	
	Error.details	String	Could not send file notification message: {0} to: {1} since web service is unavailable.	
File uploaded successfully	Result	String	OK	

Use Case	Message Mapping			Action
	Property	Type	Value	
	Error.code	String	N/A	File Upload Adapter consumes the file from local file share, and upload file successfully to external file share which is, if needed, accompanied by successful sending of file notification message. Information is specified in the log file and the event is created by File Upload Adapter.
	Error.level	String	N/A	
	Error.reason	String	N/A	
	Error.details	String	N/A	

6. MESSAGES

6.1. Common

6.1.1. Header

The header section is defined according to the IEC 61968-100. Currently, there are two required fields that must be populated:

- **Verb** – to identify a specific action to be taken. There is an enumerated set of valid verbs, where commonly used values include “get”, “create”, “change”, “cancel”, “close”, “execute” and “reply”. Within the event notification messages “past tense” verbs are used, which can include “created”, “changed”, “canceled”, “closed” and “executed”. Implementations should treat deprecated verbs “update” and “updated” as synonyms to “change” and “changed”.
- **Noun** – to identify the subject of the action and/or the type of the payload, such as the FileNotification.

Field that can be optionally supplied include the following:

- Revision – to indicate the revision of the message definition. By default, this should be “1”,
- ReplayDetection – this is a complex element with a timestamp and a nonce used to guard against replay attacks. The timestamp is generated by the source system to indicate when the message was created. The nonce is a sequence number or randomly generated string (e.g. UUID) that would not be repeated by the source system for at least a day. This serves to improve encryption.
- **Context** – a string that can be used to identify the context of the message. This can help provide an application level guard against incorrect message consumption in configurations where there may be multiple system environments running over the same messaging infrastructure. Some example values are PRODUCTION, TESTING, STUDY and TRAINING.
- Timestamp – an ISO 8601 compliant string that identifies the time the message was sent. This is analogous to the JMSTimestamp provided by JMS. Either Zulu (‘Z’) time or time with a time zone offset may be used.
- Source – identifying the source of the message, which should be the name of the system or organization.
- AsyncReplyFlag – the Boolean data type (“true” or “false” values) that indicates whether a reply message will be sent asynchronously. By default, replies are assumed to be sent synchronously.
- ReplyAddress – the address to which replies should be sent. This is typically used for asynchronous replies. This should take the form of a URL, topic name or queue name. This is analogous to the JMSReplyTo field provided by JMS. This is ignored when using unidirectional integration patterns (e.g., AckRequired=false). If the reply address is a topic, the topic name should be prefixed by “topic”. If the reply address is a queue, the queue name should be prefixed by “queue”. If the reply address is a web service, the reply address should be a URL beginning with “http://” or “https:/”.
- AckRequired – the Boolean data type (“true” or “false” values) that indicates whether an acknowledgement is required. If false, this would indicate that a unidirectional integration pattern is being used for communicating transactional messages.

- User – a complex structure that identifies the user and associated organization. Should be supplied as it may be required for some interfaces, depending upon underlying implementations. This allows the UsersID string and optional the Organization string as sub-elements.
- MessageID – a string that uniquely identifies a message. Use of the UUID or sequence number is recommended. This is analogous to the JMSMessageID provided by JMS. A process should not issue two messages using the same MessageID value.
- CorrelationID – this is used to “link” messages together. This can be supplied on a request, so that the client can correlate a corresponding reply message. The server will place the incoming CorrelationID value as the CorrelationID on the outgoing reply. If not supplied on the request, the CorrelationID of the reply should be set to the value of the MessageID that was used on the request, if present. This is analogous to the use of the JMSCorrelationID provided by JMS. Given that the CorrelationID is used to ‘link’ messages together, it may be reused on more than one message. Use of a UUID or sequence number is recommended.
- Comment – any descriptive text, but shall never be used for any processing logic.
- Property – a complex type that allows the custom name/value pairs to be conveyed. The source and targets would need to agree upon usage. These are analogous to a Property as defined by JMS.
- Any – it can be used for custom extensions.

Figure 6.1 shows the graphical representation of the header field.

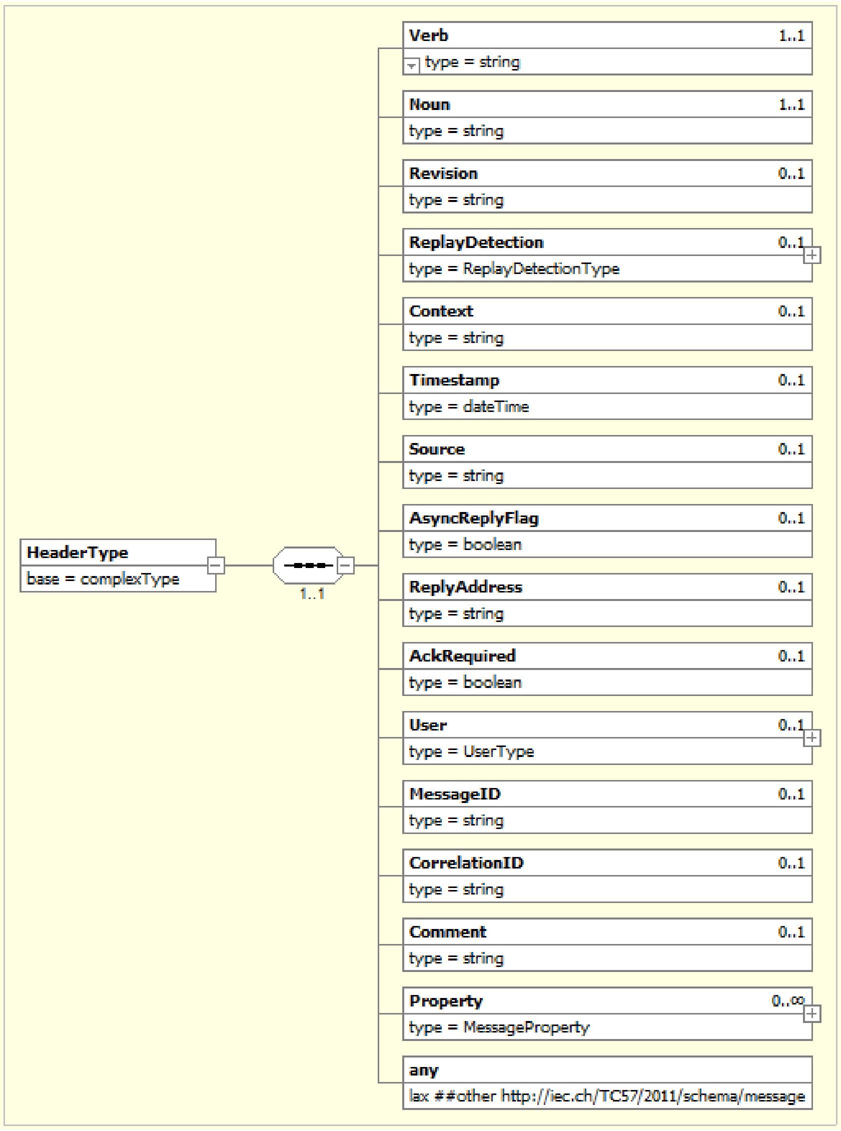

Figure 6.1 – The header field

6.1.2. Reply and Fault

The Reply.result value is an enumeration and would be populated in the following manner:

- "OK" – if there are no errors and all results have been returned. There is no requirement that a Reply.Error element be present.
- "PARTIAL" – if only a partial set of results has been returned, with or without errors. Existence of errors is indicated with one or more Reply.Error.code elements.
- “FAILED" – if no result can be returned due to one or more errors, indicated with one or more Reply.Error elements, each with a mandatory application level “code”.

If the result type is “PARTIAL” or “FAILED”, the **Error** field will be populated with the appropriate error description. The contents the **Reply** and **Error** fields are presented in Figure 6.2.

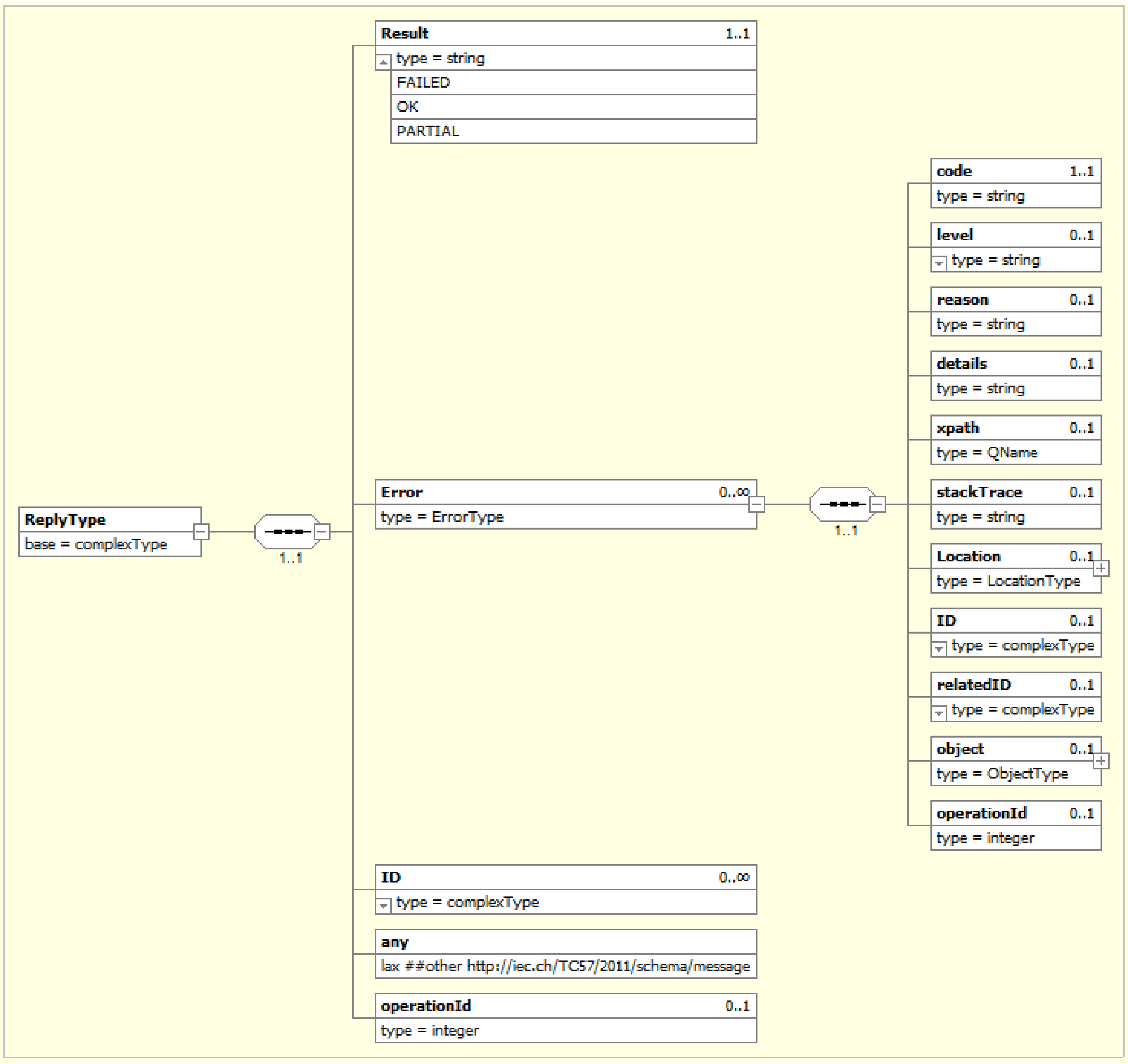

Figure 6.2 – The ***Reply*** *and* ***Error*** *field contents*

6.2. CreatedFileNotification Operation

The operation definition:

FileNotificationResponse CreatedFileNotification*(CreatedFileNotificationEvent)*

Both request and response messages contain the payload in form of the *FileNotification.xsd* schema which represents the CIM profile for the file notification.

6.2.1. Request

The *CreatedFileNotification* event message is defined according to the IEC 61968-100 and contains the following two section:

- Header
- Payload.

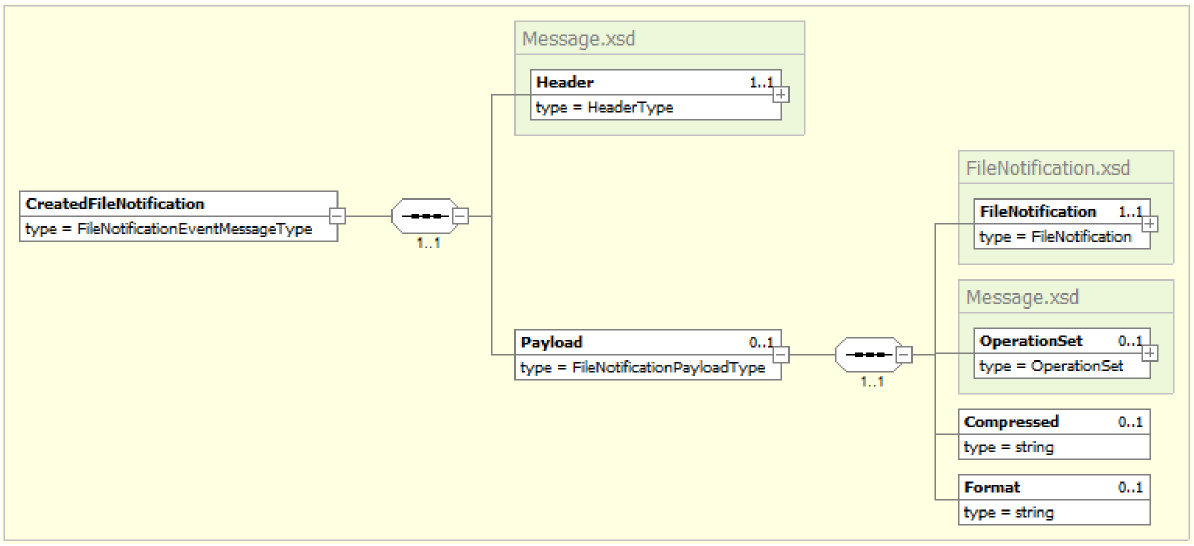

Figure 6.3 – The CreatedFileNotificationEvent message

The Payload field carries the CIM defined profile (CreatedFileNotification.xsd) for one or more files. Figure 6.4 depicts the CreatedFileNotification message.

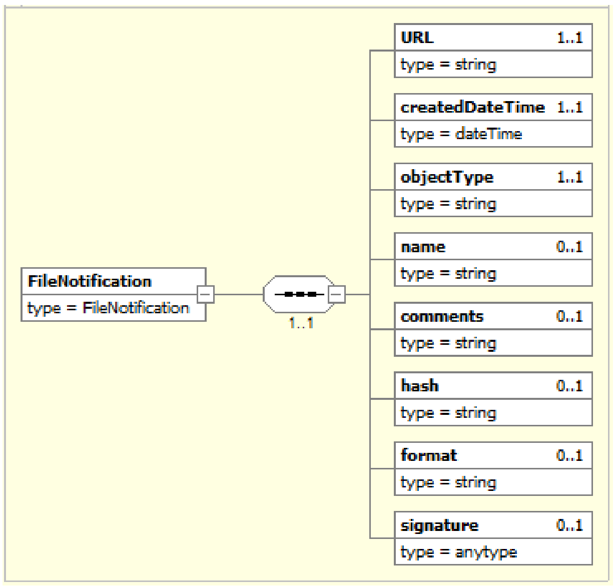

Figure 6.4 – FileNotification.xsd

Table 6.1 gives the description of entities that are a part of the *FileNotification.xsd*.

Table 6.1 – The FileNotification attributes description

CreatedFileNotification message			**Description**
Section	**Property**	**Type**	
Header	**Verb**	String	Identifier for a specific action to be taken. For this message, Verb is created.
Header	**Noun**	String	Identifier for the subject of the action and/or the type of the payload. For this message, Noun is FileNotification.
Header	Revision	String	Revision of CIM standard used. Default value is 2.0.
Header	**Timestamp**	DateTime	Timestamp when message was produced. Example: 2015-12-31T12:34:56+02:00
Header	Source	String	Source system or application that sends the message. For this message, Source is EcoStruxure GridOps.
Header	**MessageID**	String	Unique message ID to be used for tracking messages.
Header	**CorrelationID**	String	Same as message ID.
Request	URL	String	Location of the file on a FS, using a full path specification
Request	CreatedDateTime	DateTime	Time of creation or last update
Request	ObjectType	String	File type: GIS, CIS, Landbase, LP, etc. (List depends on the specific project requirements)
Request	Name	String	Object identifier, to uniquely distinguish the object when appropriate, such as a feeder name
Request	Comments	String	Comments describing file contents
Request	Hash	String	Hash of file contents using algorithm such as MD5 or SHA2
Request	Format	String	File format, e.g. pdf, xml, csv, zip, etc.
Request	Signature	Anytype	Publisher signature (Hash algorithm used for file hashing).

6.2.2. Response

After the file is uploaded and notification message sent to external system, the response is returned in form of the *FileNotificationResponse* message. The content of the response message is given in Figure 6.5.

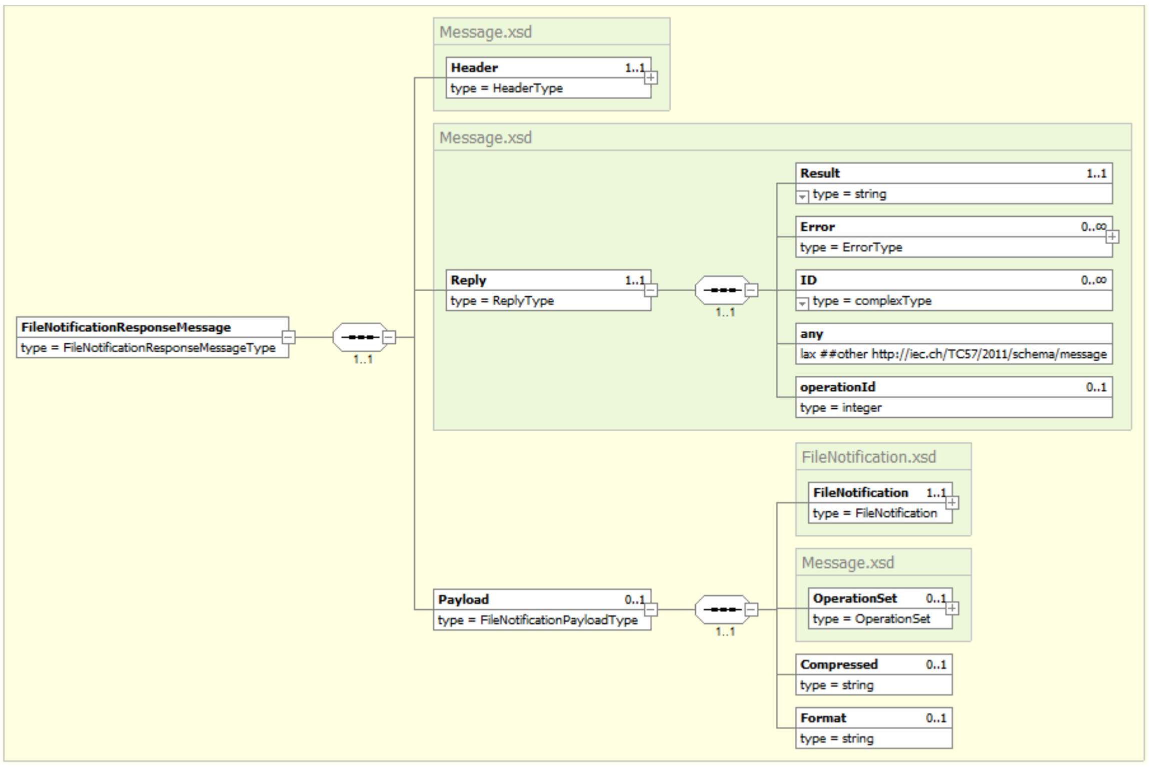

Figure 6.5 – The FileNotificationResponse message

6.2.3. Fault

The CreatedFileNotificationFault is depicted in Figure 6.6.

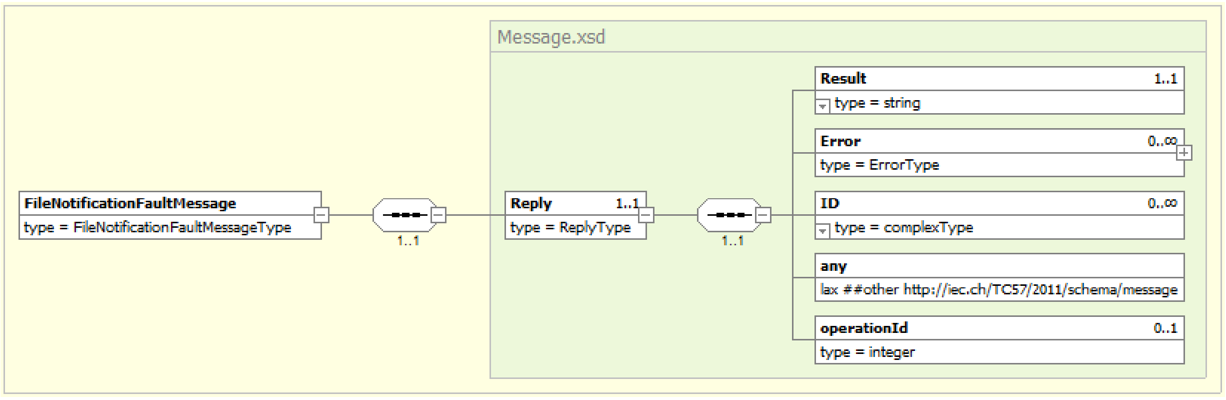

Figure 6.6 – The FileNotificationFault message

7. DEPLOYMENT SPECIFICATION

File Upload Adapter provides integration between the EcoStruxure GridOps and clients external applications such as: GIS, CIS, etc. The deployment specification is provided in the following table:

Table 7.1 – The deployment specification

Deployment Specification	
Application	FileUploadAdapter
Critical process	Yes
OASyS service	OASyS DNA DMS_INTEGRATION Service
Servers	pdmz-int-1, pdmz-int-2, bdmz-int-1, bdmz-int-2
Zone	pdmz, bdmz
Installation Type	Product
Installation add-on name	Integration Adapters

Figure 7.1 depicts standard deployment configuration for all File Upload Integration participants.

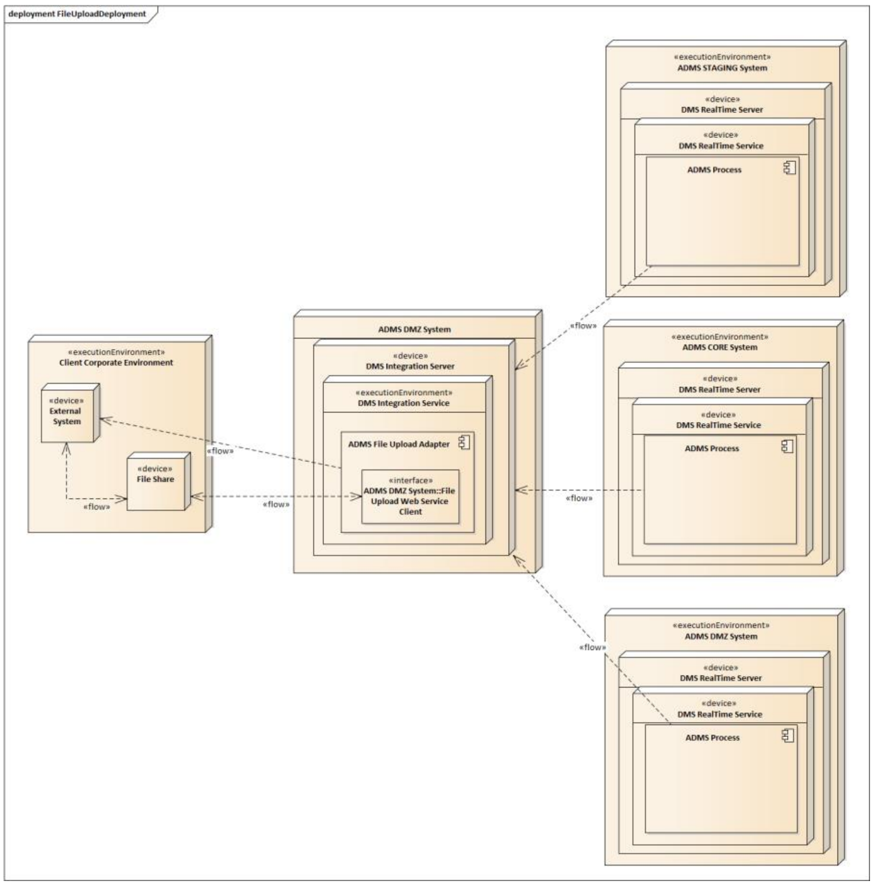

Figure 7.1 – File Upload Integration deployment diagram

8. INTERFACE CONFIGURATION

File Upload adapter provides certain amount of configurability so that smaller adjustments in the functionality can be easily applied to the system, without interface down time. Such feature is provided through dedicated configuration files of the File Upload adapter. Initially, following configuration files are used the adapter:

Table 8.1 – The configuration files specification

Name of the config file	Configuration File Description
AdapterFileUpload	Registry configuration xml file
AdapterFileUpload_WebServiceConfiguration	Web service configuration xml file
FileTypeConfiguration_FileUploadAdapter	File type configuration xml file (contains settings for source and destination path)

For more details about adapters configuration files refer to the *EcoStruxure GridOps Management Suite 3.10 Enterprise Integration Platform - Functional Specification* [1].

Detailed content of above-mentioned configuration files is provided within the *Configuration* folder in the *EcoStruxure GridOps Management Suite 3.10 File Uploading Interface.zip* file [2].

9. APPENDIX

9.1. WSDL

The WSDL file, XSD schemas and sample messages defined according to the IEC 61968-100 for the SendFileNotificationService is provided within the *Web Service Definitions* folder in the *EcoStruxure GridOps Management Suite 3.10 File Upload Interface.zip* file [2].

10. RELEASE NOTES

The following new features related to Product AMI Interfaces were introduced in the software, starting from version 3.8.

10.1. Software Version 3.8.0

Feature	Description
Individual Claim Check Notification Support	Individual claim check support was added to provide more granular way to configure when notification message needs to be sent to external system. It is now possible to configure such option per extract type.

10.2. Software Version 3.9

Feature	Description
File Upload Interface – File hash value calculation	To increase security, File Upload interface can now calculate the hash value of the file that should be uploaded and send this value through the Hash field in the claim check notification message.

11. DEFINITIONS AND ABBREVIATIONS

Definition/Abbreviation	Description
ADMS	Advanced Distribution Management System
CIM	Common Information Model
CIS	Customer Information Service
DMD	Dynamic Mimic Diagram
DMZ	Demilitarized Zone
ESB	Enterprise Service Bus
LFS	Local File Share
FS	File Share
NIS	Network Import Service
SFTP	SSH File Transfer Protocol
SOAP	Simple Object Access Protocol
WCF	Windows Communication Foundation
WS	Web Service
XML	Extensible Markup Language
XSD	XML Schema Definition