# ETSITS 119 524-2 V1.1.1 (2019-02)



Electronic Signatures and Infrastructures (ESI);
Testing Conformance and Interoperability of
Electronic Registered Delivery Services;
Part 2: Test suites for interoperability testing of
Electronic Registered Delivery Service Providers

#### Reference

#### DTS/ESI-0019524-2

#### Keywords

electronic registered delivery, interoperability, testing

#### **ETSI**

650 Route des Lucioles F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C Association à but non lucratif enregistrée à la Sous-Préfecture de Grasse (06) N° 7803/88

#### Important notice

The present document can be downloaded from: http://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at <a href="https://www.etsi.org/deliver">www.etsi.org/deliver</a>.

Users of the present document should be aware that the document may be subject to revision or change of status.

Information on the current status of this and other ETSI documents is available at <a href="https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx">https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx</a>

If you find errors in the present document, please send your comment to one of the following services: https://portal.etsi.org/People/CommiteeSupportStaff.aspx

#### **Copyright Notification**

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.

The content of the PDF version shall not be modified without the written authorization of ETSI.

The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI 2019. All rights reserved.

**DECT**<sup>™</sup>, **PLUGTESTS**<sup>™</sup>, **UMTS**<sup>™</sup> and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP**<sup>™</sup> and **LTE**<sup>™</sup> are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners.

oneM2M<sup>™</sup> logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners.

**GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

# Contents

Intell	lectual Property Rights	5
Forev	word	5
Moda	al verbs terminology	5
1	Scope	6
2	References	6
2.1	Normative references	
2.2	Informative references.	
3	Definition of terms, symbols and abbreviations	
3.1	Terms	
3.2 3.3	SymbolsAbbreviations	
4	Technical approach	
4.1	Components of test cases and their identifiers	8
4.2	Adding new test cases to the test suite	9
5	Scenarios	10
5.1	Introduction	10
5.2	Abbreviations used in tables defining scenarios	11
5.3	Black-box model scenarios	12
5.3.1	Introduction	12
5.3.2	Scenarios without notification for acceptance	
5.3.3	Scenarios with notification for acceptance	
5.4	Scenarios for 4-corner model	
5.4.1	Introduction	
5.4.2	Scenarios where RERDS does not use notification for acceptance	
5.4.3	Scenarios where RERDS uses notification for acceptance	
5.5 5.5.1	Scenarios for extended model	
5.5.2	Scenarios where RERDS does not use notification for acceptance	
5.5.3	Scenarios where RERDS uses notification for acceptance	
	·	
6	ERD Messages instances	60
6.1	Introduction and technical approach	
6.2	Combinations of fields for headers in ERD envelopes	
6.2.1 6.2.2	Introduction	
6.2.3	Combinations of AS4 metadata promed in E1S1 EN S19 322-4	
6.2.4	Combinations of AS4 metadata profiled and relay metadata	
6.3	Instances of ERD payload	
6.4	Instances of ERDS receipts	
6.5	Instances of ERD dispatch	
7	Other named sets	67
7.1	Named sets of participating ERDSs	
7.2	Named sets of additional requirements	
7.3	Named sets of entities in receiving side	
8	Test cases definition	67
8.1	Introduction	
8.1.1	General	
8.1.2	Notation for black box model scenarios	
8.1.3	Notation for 4 corner and extended models scenarios	68
8.1.4	Reasons for Failures	
8.2	Test cases for black box model	
8.3	Test cases for 4-cornel and extended models	
8.3.1	General	70

History			73
Annex A	(informative):	Bibliography	72
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	reactor parame		
8.3.7		trizing additional requirements	
8.3.6	Rules for parame	trizing failure reasons	71
3.3.5		trizing entities at receiving side	
3.3.4		trizing ERDS receipts	
3.3.3		trizing ERD payloads	
3.3.2	Rules for parame	trizing ERD dispatches	70

# Intellectual Property Rights

#### Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: "Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards", which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

#### **Trademarks**

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

### **Foreword**

This Technical Specification (TS) has been produced by ETSI Technical Committee Electronic Signatures and Infrastructures (ESI).

The present document is part 2 of a multi-part deliverable. Full details of the entire series can be found in part 1 [i.1].

# Modal verbs terminology

In the present document "shall", "shall not", "should", "should not", "may", "need not", "will", "will not", "can" and "cannot" are to be interpreted as described in clause 3.2 of the <u>ETSI Drafting Rules</u> (Verbal forms for the expression of provisions).

"must" and "must not" are NOT allowed in ETSI deliverables except when used in direct citation.

# 1 Scope

The present document defines:

- 1) A test suite for supporting interoperability tests within the field of Electronic Registered Delivery Services (ERD services or ERDS hereinafter) as specified in ETSI EN 319 522 parts 1 [1], 2 [2], 3 [3] and 4 [4], [5] and [6]. The test suite defines test cases for the following environments:
  - Environments that correspond to the basic model as defined in ETSI EN 319 522-1 [1] where sender and all the entities at receiving side are subscribed to the same ERDS.
  - Environments that correspond to the 4-corner model as defined in ETSI EN 319 522-1 [1] where sender
    is subscribed to one ERDS and the entities at receiving side are subscribed to another one, and no
    intermediate ERDS is required for relaying ERD messages between them.
  - Environments that correspond to the extended model as defined in ETSI EN 319 522-1 [1] where sender is subscribed to one ERDS and the entities at receiving side are subscribed to another one, and intermediate ERDSs are required for relaying ERD messages between them.
- 2) A mechanism for documenting new test cases and expanding the aforementioned test suite.

### 2 References

### 2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at <a href="https://docbox.etsi.org/Reference/">https://docbox.etsi.org/Reference/</a>.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1]	ETSI EN 319 522-1: "Electronic Signatures and Infrastructures (ESI); Electronic Registered Delivery Services; Part 1: Framework and Architecture".
[2]	ETSI EN 319 522-2: "Electronic Signatures and Infrastructures (ESI); Electronic Registered Delivery Services; Part 2: Semantic contents".
[3]	ETSI EN 319 522-3: "Electronic Signatures and Infrastructures (ESI); Electronic Registered Delivery Services; Part 3: Formats".
[4]	ETSI EN 319 522-4-1: "Electronic Signatures and Infrastructures (ESI); Electronic Registered Delivery Services; Part 4: Bindings; Sub-part 1: Message delivery bindings".
[5]	ETSI EN 319 522-4-2: "Electronic Signatures and Infrastructures (ESI); Electronic Registered Delivery Services; Part 4: Bindings; Sub-part 2: Evidence and identification bindings".
[6]	ETSI EN 319 522-4-3: "Electronic Signatures and Infrastructures (ESI); Electronic Registered Delivery Services; Part 4: Bindings; Sub-part 3: Capability/requirements bindings".
[7]	ETSI EN 319 532-3: "Electronic Signatures and Infrastructures (ESI); Registered Electronic Mail (REM) Services; Part 3: Formats".
[8]	OASIS Standard (January 2013): "AS4 Profile of ebMS 3.0 Version 1.0".

### 2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI TS 119 524-1: "Electronic Signatures and Infrastructures (ESI); Testing Conformance and

Interoperability of Electronic Registered Delivery Services; Part 1: Testing conformance".

[i.2] OASIS Standard (October 2007): "ebXML Messaging Services Version 3.0: Part 1, Core

Features".

NOTE: Available at <a href="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/ebms">http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/ebms</a> core-3.0-spec.pdf.

# 3 Definition of terms, symbols and abbreviations

#### 3.1 Terms

For the purposes of the present document, the terms given in ETSI EN 319 522-1 [1] apply.

### 3.2 Symbols

Void.

### 3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ACC\_REJ\_EXP ACCeptance REJection EXPiry
AS4 Applicability Statement 4
CONS\_ACC CONSignment ACCeptance
CONS\_NOT CONSignment NOTification

CONS\_NOT\_FAIL CONSignment NOTification FAILure

CONS\_REJ CONSignment REJection CONT\_CONS CONTent CONSignment

CONT\_CONS\_FAIL CONTent CONSignment FAILure

CONT\_HAND CONTent HANDover

CONT\_HAND\_FAIL CONTent HANDover FAILure ebMS ebXML Messaging Services

ebXML Electronic Business using eXtensible Markup Language

ERD Electronic Registered Delivery
ERDS Electronic Registered Delivery Service

EV\_SET Evidence - SET

IERDS Intermediate Electronic Registered Delivery Service

NOT\_F\_ACC NOTification For ACCceptance

OASIS Organization for the Advancement of Structured Information Standards

REC\_F\_NERDS RECeived From Non ERDS REL\_ACC RELay ACCeptance

REL\_FAIL RELay FAILure
REL\_REJ RELay REJection
REL\_T\_NERDS RELay To Non ERDS

REL\_T\_NERDS\_FAIL RELay To Non ERDS FAILure

REM Registered Electronic Mail
REMS Registered Electronic Mail Service

RERDS Recipient's Electronic Registered Delivery Service

SCN\_ID Scenario IDentifier

SERDS Sender's Electronic Registered Delivery Service

SUB\_ACCSUBmission ACCeptanceSUB\_REJSUBmission REJectionURIUniversal Resource IdentifierXMLeXtensible Mark-up Language

# 4 Technical approach

### 4.1 Components of test cases and their identifiers

As it has been mentioned before the present document defines:

- 1) A test suite for supporting interoperability tests within the field of Electronic Registered Delivery (ERD hereinafter) as specified in as specified in ETSI EN 319 522 parts 1 [1], 2 [2], 3 [3] and 4 [4], [5] and [6].
- 2) A mechanism for documenting new test cases and expanding the aforementioned test suite.

The present document follows a layered approach for building the definition of the test cases in the test suite, which can be summarized as follows:

1) Clause 5 defines a number of parameterized scenarios. A scenario consists of a number of entities, namely: sender, one or more ERDSs, and the entities at receiving side (one or more recipients and/or one or more recipients' delegates), which exchange different ERD messages with time. Each scenario corresponds to one of the three models presented in ETSI EN 319 522-1 [1]. This clause presents a template for defining one scenario, in a way that resembles to some templates used for defining use cases scenarios in software engineering.

#### This template:

- Includes the enumeration of the original message and all the ERD messages exchanged by the participating entities. This list of exchanged ERD messages is one of the parameters of the scenario.
- Also includes a list of ERDS evidence sets, which, in the scenario, are incorporated in some ERD messages.

One scenario may be used for defining several test cases depending on:

- The specific components of each exchanged ERD message (suppressing or adding an optional metadata component, or changing the value of a certain metadata component results in a different ERD message and consequently a different test case).
- The entities at receiving part (for instance, changing one recipient by one recipient's delegate, or two recipients and one recipient's delegate results in a different the test case).
- A named set of additional requirements (for instance details of the original message, like whether it contains or not attachments, is signed, is encrypted, etc.).

This means that one test case corresponds to one scenario where all the exchanged ERD messages have been completely defined in terms of their components, all the participating entities have been established, and all the additional requirements have also been defined. Taking the functional notation this can be expressed as follows:

TestCase#i = Scenario\_id(,<Receiving side identifier>, <ERD message 1 details>, <ERD message 2 details>, ..., <ERD message N details>, <additional requirements set identifier)

#### Where:

- < Receiving side identifier> is the identifier assigned to a certain set of entities at receiving side;

- <message identifier I> is the identifier of a specific instantiation of the aforementioned message, defined in clauses 6.3, 6.4 and 6.5. These clauses define specific instantiations of ERD payloads, ERD receipts and ERD dispatches respectively.
- <additional requirements set identifier> is the identifier of a named set of additional requirements. Clause 7.2 defines a number of these named sets.
- 2) Clauses 6.3, 6.4 and 6.5 define specific instantiations of ERD payloads, ERD receipts and ERD dispatches respectively. Each type of ERD message is composed by several components, with their metadata components and payloads as specified in ETSI EN 319 522-4-1 [4] and ETSI EN 319 522-4-2 [5]. The present document defines a number of combinations of metadata components in clauses 6.2.2 and 6.2.3, and assigns to each one a unique identifier. This allows to use again the functional notation, and define one instantiation of a certain type of ERD message as follows:

ERD message instance = Sequence(Metadata(<AS4 profiled metadata combination details>, <payload for ERDS relay metadata combination details>) <payload for User Content>\* <payload for ERDS Evidence>\*)

Where '\*' stands for 0 or more occurrences of the payload.

NOTE: The payloads for user content and for ERDS evidence can be present at certain types of ERD messages but are forbidden in other types.

3) Clauses 6.2.2 and 6.2.3 define named combinations of metadata components defined in OASIS: "AS4 Profile of ebMS 3.0 Version 1.0" [8] and profiled in ETSI EN 319 522-4-1 [4] and ETSI EN 319 522-4-2 [5], and the relay metadata components defined in ETSI EN 319 522-3 [3] respectively. Each clause define different instances of the aforementioned components and assigns them unique identifiers that are used for defining specific instances of the different ERD messages as shown above. Once this level is reached, the specific test case is fully defined as: a scenario where fully defined, ERD messages are exchanged between a specific set participating entities, and where a specific set of additional requirements are imposed.

### 4.2 Adding new test cases to the test suite

The strategy followed for building the definitions of the test cases makes it easy to expand the test suite by incorporation of new test cases.

For defining a new test case the following steps are required:

- 1) Identify the **set of receiving entities**. If none of the predefined set of entities at the receiving side is the one required, assign a name to this set (**RECEIVING\_ENTITIES** >) and incorporate it to the repertoire of named sets as specified in clause 7.3). The sender is always present by default.
- 2) Define the ERDSs that will participate in the test case.
- 3) If the set of participating ERDSs is not equal to none of the scenarios already identified in the present document, the new scenario will require to be defined in a new template.
- 4) Identify the **sequence of actions** performed by each actor and their order of occurrence and assign a new unique identifier (**SCN ID** >) to the scenario.
- 5) Identify all the ERD messages generated by the actors as they go through the sequence of actions. For each message:
  - a) Identify its ebMS payloads, e.g. the parts of the user content or XML document with relay meta-data.
  - b) Check if the combinations of metadata components have already been defined in the present document. If not, add the required combination of metadata components to the repertoire of named combinations to the corresponding clause (clause 6.2.2 or 6.2.3).
  - c) List all the ERD messages exchanged as parameters of the scenario.
  - d) Identify the ERDS evidence format and the set of ERDS evidence for each ERD message including them and add the names of the ERDS evidence sets to the Var section of the template.

6) Identify and define any other additional requirement for completely define the test case. If the set of requirements is different than all the sets already define, assign a name to it (**ADD\_REQ\_COMB>**) and add it to the repertoire of named sets of additional requirements in Table 12 (clause 7.2).

## 5 Scenarios

#### 5.1 Introduction

The present clause defines a number of selected scenarios that will be used in clause 8.

Clause 5.3 defines scenarios where sender and recipient(s) are subscribed to the same ERDS (black-box model described in ETSI EN 319 522-1 [1]).

Clause 5.4 defines scenarios where the sender and the recipient(s) are subscribed to different ERDSs and there are not intermediate ERDSs between them (4-corner model described in ETSI EN 319 522-1 [1]).

Clause 5.5 defines scenarios where sender is subscribed to a ERDS and the recipient(s) is(are) not subscribed to the same ERDS and there are one or more intermediate ERDSs (extended model described in ETSI EN 319 522-1 [1]).

Figure 1 of clause 4 of ETSI EN 319 522-2 [2] shows three structures being exchanged between ERD-UAs and ERDSs, namely:

- 1) The structure {submission metadata, user content}, which receives the name of original message in Table 1 of clause 4 of ETSI EN 319 522-2 [2].
- 2) The structure {ERDS handover metadata, ERDS evidence} for allowing access to ERDS evidences to users.
- 3) The structure {ERDS handover metadata, user content, ERDS evidence} for allowing the R-ERDS the submission of the user content (and optionally ERDS evidences) to the recipient.

Because of that the following decisions have been adopted for building the scenarios:

- 1) Neither S-ERDS nor R-ERDS will submit {ERDS handover metadata, ERDS evidence} structures to their users, except when the ERDS evidence is an evidence of some kind of relevant rejection by the ERDS (see the first scenario, for instance). Identical scenarios including the submission of such structures can be easily defined and used in interoperability test events.
- 2) The scenarios will show the R-ERDS submitting {ERDS handover metadata, user content, ERDS evidence} or {ERDS handover metadata, user content} structures to the receiving side.
- 3) The acronym hndvMet is used for ERDS handover metadata.

Table 1 shows the template for defining one scenario.

Table 1: Template for the tabular definition of one scenario

	Scenario id: <scn id=""></scn>					
Par	ameter: <erds_receipt>_with_XML_SUB_</erds_receipt>	_REJ <parameter< td=""><td>Var SET_EV#1 = {, .</td><td>}</td><td></td></parameter<>	Var SET_EV#1 = {, .	}		
1 th	1 that helps to fully the scenario. Their number depends on the Named sets of ERDS evidence used in					
spe	cific scenario>		the definition of the sce	nario.		
Par	ameter: <parameter 2=""></parameter>		Var SET_EV#2 = {	.}		
Par	ameter: <parameter n=""></parameter>		Var SET_EV#N = {	.}		
	Sec	uence of actions				
<se< td=""><td>EQUENCE OF ACTIONS. THERE IS ONE</td><td>COLUMN PER PAI</td><td>RTICIPATING ACTOR&gt;</td><td></td><td></td></se<>	EQUENCE OF ACTIONS. THERE IS ONE	COLUMN PER PAI	RTICIPATING ACTOR>			
#	Sender		ERDS	Receiving side		
The	e sequence is composed of a number of nu	merated steps, whe	n the actors perform cer	tain actions as		
sho	wn below.					
Sor	me frequent actions: send original message	, accept submission	n, reject submission, cor	nsign, generate		
one	ERDS evidence, generate one ERD mess	age, etc.				
1	Sender sends original message					
2		Rejects submission	n. Generates			
		XML_SUB_REJ EI	RDS evidence			
3		Generates				
<erds_receipt>_with_XML_SUB_REJ</erds_receipt>						
4		Sends				
		<erds_receipt>_v</erds_receipt>	with_XML_SUB_REJ			
5	Receives					
	<erds_receipt>_with_XML_SUB_REJ</erds_receipt>					

Each scenario is assigned a unique identifier <SCN\_ID>. The reasons why the scenario has been defined are shown in column "Purpose".

The definition of each scenario requires that parties exchange a number of ERD messages, which appear listed as parameters in the rows immediately below the headers row. Its number depends on the specific scenario.

Below the list of parameters, the table shows a sequence of actions performed by different involved entities, which results in that a set of ERD messages is generated and exchanged.

The definition of each scenario also can use a number of named ERDS evidence sets, which are listed in cells started with Var. Each ERDS evidence set is given a name EV\_SET#<i>, being <i> a non-negative integer number.

The involved entities are sender (or sender's delegate, the scenario definition does not make any distinction between them), one or more ERDSs, and the entities at the receiving side (for the same scenario there may be different sets of entities, for instance one recipient, one recipient's delegate, one or more recipients, or one or more recipients and one or more recipients' delegates).

Each step is assigned a positive integer number. The actions performed include: submission of messages, generation of ERD messages, generation of ERD messages, rejection of ERD messages, consignment of ERD messages, retrieval of ERD messages by entities at receiving side, failures, etc.

### 5.2 Abbreviations used in tables defining scenarios

This clause shows some abbreviations (which have already been anticipated in clause 3.3) used in the tables that define the scenarios.

Table 2 shows the abbreviations used for the different ERDS evidence.

Table 2: ERDS evidence abbreviations

ERDS Evidence name	ERDS Evidence abbreviation
SubmissionAcceptance	SUB_ACC
SubmissionRejection	SUB_REJ
RelayAcceptance	REL_ACC
RelayRejection	REL_REJ
RelayFailure	REL_FAIL
NotificationForAcceptance	NOT_F_ACC
NotificationForAcceptanceFailure	CONS_ACC
ConsignmentAcceptance	CONS_REJ
ConsignmentRejection	CONT_CONS
AcceptanceRejectionExpiry	ACC_REJ_EXP
ContentConsignment	CONT_CONS
ContentConsignmentFailure	CONT_CONS_FAIL
ConsignmentNotification	CONS_NOT
ConsignmentNotificationFailure	CONS_NOT_FAIL
ContentHandover	CONT_HAND
ContentHandoverFailure	CONT_HAND_FAIL
RelayToNonERDS	REL_T_NERDS
RelayToNonERDSFailure	REL_T_NERDS_FAIL
ReceivedFromNonERDS	REC_F_NERDS

ETSI EN 319 522-1 [1] specify a XML format for ERDS evidence, but also allows that ERDS Evidences are signed PDF documents. The notation defined in the present document makes it clear that all the test cases are defined for XML ERDS Evidence using the **XML** prefix for the ERDS evidence abbreviations.

EXAMPLE: The abbreviation for the XML SubmissionAcceptance ERDS evidence will be XML SUB ACC.

NOTE: In case some format for PDF ERDS Evidence is defined and ERDS providers need to test interoperability with them, it is always possible to replace the test cases defined in the present document by identical test cases where PDF ERDS Evidences are generated and exchanged instead XML ERDS Evidences.

The tables defining the Scenarios use the following abbreviations for the different participating ERDSs:

- **SERDS** stands for the ERDS serving the sender, in the scenarios where it is different than the ERDS serving the entities at receiving side.
- **RERDS** stands for the ERDS serving the entities at receiving side, in the scenarios where it is different than the ERDS serving the sender.
- **IERDS** stands for a ERDS that does not directly serves neither to the sender nor to the recipient(s)/recipient's delegate, but instead is an intermediate ERDS that relies ERD messages from SERDS to RERDS and from RERDS to SERDS.

### 5.3 Black-box model scenarios

#### 5.3.1 Introduction

The present clause defines scenarios where the sender and the entities at the receiving side are subscribed to the same ERDS and consequently the user content is not relayed between different ERDSs.

Clause 5.3.2 defines scenarios where the ERDS operates in Store and Forward style.

Clause 5.3.3 defines scenarios where the ERDS operates in Store and Notify style.

### 5.3.2 Scenarios without notification for acceptance

Table 3 defines a number of scenarios for the case where sender and the entities at the receiving side are subscribed to the same ERDS and the ERDS does not send notification for acceptance to the entities at the receiving side.

Table 3: Scenarios for intra-ERDS without notifications for acceptance (1/8)

		Purpose		
Para	ameter: {hndvMet,XML_SUB_REJ}			The simplest scenario: the
		Sequence of actions		ERDS rejects the original
#	Sender	ERDS	Receiving side	message submitted by the
1	Sender sends original message			sender and sends back a {ERDS
2		Rejects submission. Generates XML_SUB_REJ ERDS evidence		handover metadata,  XML_SUB_REJ} structure
3		Sends {hndvMet, XML_SUB_REJ} structure to the sender		SubmissionRejection ERDS evidence.
4	Receives {hndvMet, XML_SUB_REJ} structure			

Table 3a: : Scenarios for intra-ERDS without notifications for acceptance (2/8)

		Scenario id: ERDS_BB_NC	O_NOT_F_ACC#2			Purpose
	meter: {hndvMet, user content}					he simplest successful cenario. The ERDS:
	meter: XML_SOB_ACC				1.	. Accepts the submission of
		Sequence of a	ctions			the original message.
#	Sender	ERDS		Receiving side	2.	Generates the
1	Sender sends original message					SubmissionAcceptance ERDS evidence, and stores
2		Accepts submission				it.
3		Generates and stores XML_SUB	S_ACC ERDS evidence		3	. Aggregates the user content
4		Generates {hndvMet, user content	nt} structure			to ERDS handover metadata
5		Consigns {hndvMet, user content	t} structure to the N			and performs as many
		entities at the receiving side				consignments as required by
6				{hndvMet, user content} structure		the number of entities in the
				correctly consigned to the N entities at		receiving side.
<u> </u>				the receiving side	4.	. Generates the
7		Generates and stores XML_CON	IT_CONS ERDS evidence			ContentConsignment ERDS
						evidence.

Table 3b: Scenarios for intra-ERDS without notifications for acceptance (3/8)

		Scenario id: ERDS_BB_NO_NOT_F_ACC#3		Purpose
Para	ameter: {hndvMet, user content}			The ERDS:
Para	ameter: XML_SUB_ACC			Accepts the submission of
Para	ameter: XML_CONT_CONS			the original message.
Para	ameter: XML_CONT_HAND			2. Generates the
		Sequence of actions		SubmissionAcceptance
#	Sender	ERDS	Receiving side	ERDS evidence, and stores
1	Sender sends original message			<ul><li>it.</li><li>3. Aggregates the user content</li></ul>
2		Accepts submission.		to ERDS handover metadata
3		Generates and stores XML_SUB_ACC ERDS		and performs as many
		evidence		consignments as required by
4		Generates {hndvMet, user content} structure		the number of entities in the
5		Consigns {hndvMet, user content} structure to the N		receiving side.
_		entities at the receiving side		4. Generates the
6		Generates and stores XML_CONT_CONS ERDS	{hndvMet, user content} structure	ContentConsignment ERDS
		evidence	correctly consigned to N entities at	evidence.
7			receiving side	5. All entities at receiving side
'			All the entities retrieve the ERD dispatch	successfully handover the
8			<del>                                     </del>	consigned structure. 6. ERDS Generates and stores
		Generates and stores XML_CONT_HAND ERDS	N handovers of {hndvMet, user content}	
		evidence for the N handover and stores	structure succeed	evidence for the N
				handovers and stores.

Table 3c: Scenarios for intra-ERDS without notifications for acceptance (4/8)

	Scenario	Purpose		
Para	nmeter: {hndvMet, user content}			As scenario
Para	meter: XML_SUB_ACC			ERDS_BB_NO_NOT_F_ACC#3,
Para	meter: XML_CONT_HAND			but one of the handing over fails.
Para	meter: XML_CONT_HAND_FAIL			Hereinafter, the scenarios do not
		Sequence of actions		show handing over, but only
#	Sender	ERDS	Receiving side	consignment. However, a set of scenarios including handing over
1	Sender sends original message			could be easily built based on
2		Accepts submission. Generates and stores XML_SUB_ACC ERDS evidence		them.
3		Generates {hndvMet, user content}structure		
4		Consigns {hndvMet, user content} structure to the N entities at the receiving side		
5		FRDS evidence	{hndvMet, user content} structure successfully consigned to N entities at the receiving side	
6			Entities try to retrieve the ERD dispatch, but one fails	
7			N-1 handovers of {hndvMet, user content} structure succeed. One fails	

Table 3d: Scenarios for intra-ERDS without notifications for acceptance (5/8)

		Purpose		
Para	ameter: {hndvMet, user content, XML_S	SUB_ACC}		As scenario
Para	ameter: XML_CONT_CONS			ERDS_BB_NO_NOT_F_ACC#2
		Sequence of actions		but now the ERDS builds a
#	Sender	ERDS	Receiving side	structure with the user content
1	Sender sends the original message			and the ERDS evidence, and consigns it to the receiving side.
2		Accepts submission.		consigns it to the receiving side.
3		Generates and stores XML_SUB_ACC ERDS evidence		
4		Generates {hndvMet, user content, XML_SUB_ACC}		
		structure		
5		Consigns <{hndvMet, user content, XML_SUB_ACC}		
		structure to the receiving side		
6			{hndvMet, user content,	
		Generates and stores XML_CONT_CONS ERDS evidence	XML_SUB_ACC} structure successfully	
		555.24.55 22 5.6.55 7 <u>_</u> 56111_56116 E11.56 611461166	consigned to the N entities at the	
<u> </u>			receiving side	
7		Generates and stores XML_CONT_CONS ERDS evidence		

Table 3e: Scenarios for intra-ERDS without notifications for acceptance (6/8)

	Scenario id: ERDS_BB_NO_NOT_F_ACC#6				
Parameter: {h	nndvMet, user content}			As scenario	
Parameter: X	ML_SUB_ACC			ERDS_BB_NO_NOT_F_ACC#2	
Parameter: {N	NotificationOfConsignment}			but now the ERDS sends a	
Parameter: X	ML_CONS_NOT			ERDS notification of	
Parameter: X	ML_CONT_CONS			consignment to receiving side.	
		Sequence of actions			
#	Sender	ERDS	Receiving side		
1 Sender	sends the original message				
2		Accepts submission. Generates and stores XML SUB ACC ERDS evidence			
3 4		Generates {hndvMet, user content} structure			
5		Consigns hndvMet, user content} structure to receiving side			
6			{hndvMet, user content} structure successfully consigned to the N entities of the receiving side		
7		Generates and stores XML_CONT_CONS ERDS evidence	-		
8		Generates (NotificationOfConsignment) for N entities			
9		Sends (NotificationOfConsignment) to receiving side			
10		Generates and stores XML_CONS_NOT ERDS evidence	{NotificationOfConsignment} received by N entities at the receiving side		

Table 3f: Scenarios for intra-ERDS without notifications for acceptance (7/8)

	Scenario id: ERDS_BB_NO_NOT_F_ACC#7				Purpose
Para	ameter: {hndvMet, user content}				As scenario
	ameter: XML_SUB_ACC				ERDS_BB_NO_NOT_F_ACC#2
Para	ameter: {NotificationOfConsignment}				but now one of the ERDS
Para	ameter: XML_CONS_NOT				notification of consignment fails
Para	ameter: XML_CONS_NOT_FAIL				
		Sequence of a	actions		
#	Sender		ERDS	Receiving side	
1	Sender sends the original message				
2		Accepts submiss	sion		
3			stores XML_SUB_ACC		
		ERDS evidence			
4			vMet, user content}		
			entities at the receiving side		
5			Met, user content}		
		structure to rece	eiving side		
6				N consignments of {hndvMet, user	
				content} structure succeed	
7			ificationOfConsignment} for		
		N entities in rece			
8			icationOfConsignment} for		
		N entities in rece	eiving side but one fails	N 4 (N 2) (1 O (O )	_
9		0	-t VAIL CONIC NICT	N-1 {NotificationOfConsignment} are	
		Generates and s	stores XML_CONS_NOT	successfully received; one is not	
10		Congretos and	otoroo ono	received	_
10		Generates and s	OT_FAIL ERDS evidence		
			nd XML_CONS_NOT		
		ERDS evidence			

### Table 3g: Scenarios for intra-ERDS without notifications for acceptance (8/8)

		Purpose		
Para	ameter: {hndvMet, user content}			As scenario
Para	ameter: XML_SUB_ACC			ERDS_BB_NO_NOT_F_ACC#2
Para	ameter: XML_CONT_CONS			but now one of the ERD dispatch
Para	ameter: XML_CONS_FAIL			consignments fails
		Sequence of actions		
#	Sender	ERDS	Receiving side	
1	Sender sends original message			1
2		Accepts submission.		
3		Generates and stores XML_SUB_ACC ERDS evidence		
4		Generates {hndvMet, user content} structure		
5		Consigns (hndvMet, user content) structure to receiving		
		side		
6			N-1 {hndvMet, user content} structures	
			are successfully consigned. One fails	
7		Generates XML_CONT_CONS ERDS evidence for N-1		
		entities and one XML_ CONS_FAIL ERDS evidence for		
		one entity and stores them		

### 5.3.3 Scenarios with notification for acceptance

Table 4 defines a number of scenarios for the case where sender and the entities at receiving side are subscribed to the same ERDS and the ERDS sends notification for acceptance to the entities at the receiving side.

Table 4: Scenarios for intra-ERDS with notifications for acceptance (1/6)

		Purpose		
Para	meter: {hndvMet, user content}			First scenario where the ERDS
Para	meter: {NotificationForAcceptance	e}		asks to receiving side for
Para	meter: XML_NOT_F_ACC			acceptance, and all the entities
Para	meter: XML_CONS_ACC			at receiving side accept.
Para	meter: XML_CONT_CONS			
		Sequence of actions		
#	Sender	ERDS	Receiving side	
1	Sender sends original			
	message			
2		Accepts submission		
3		Generates and stores XML_SUB_ACC ERDS evidence		
4		Generates (NotificationForAcceptance)		
5		Generates and stores XML_NOT_F_ACC ERDS evidence		
6		Sends {NotificationForAcceptance}		
7			All entities in receiving side receive one {NotificationForAcceptance}and answer positively	
8		Receives positive responses from receiving side and Generates and stores XML_CONS_ACC ERDS evidence for the N entities at receiving side		
9		Generates {hndvMet, user content} structure		
10		Consigns {hndvMet, user content} to N entities in receiving side		
11		Generates and stores XML_CONT_CONS ERDS evidence	N consignments of {hndvMet, user content} succeed	

Table 4a: Scenarios for intra-ERDS with notifications for acceptance (2/6)

		Purpose		
Para	ameter: {hndvMet, user content}			As before but one of the entities
Para	meter: {NotificationForAcceptance}			at the receiving side does not
Para	meter: XML_SUB_ACC			accept consignment.
Para	meter: XML_NOT_F_ACC			
Para	meter: XML_CONS_ACC			
Para	meter: XML_CONS_REJ			
Para	meter: XML_CONT_CONS			
		Sequence of actions		
#	Sender	ERDS	Receiving side	
1	Sender sends original message			
2		Accepts submission.		
3		Generates and stores XML_SUB_ACC ERDS evidence		
4		Generates (NotificationForAcceptance) for N entities at		
		receiving side		
5		Sends (NotificationForAcceptance) to N entities at receiving	g	
		side		
6			N entities correctly receive	
		Generates and stores XML_NOT_F_ACC ERDS evidence		
		for N entities	accept. One does not accept	
7		Descives N.1 positive anguers and any negative anguer	consignment	_
8		Receives N-1 positive answers and one negative answer Generates XML CONS ACC ERDS evidence for N-1		_
0		entities and one XML_CONS_REJ ERDS evidence for 1		
		entity		
9		Generates {hndvMet, user content} structure for N-1 entition	s l	<del>- </del>
10		Consigns them to the N-1 entities at receiving side		<del>- </del>
11	<u> </u>	Somoigno them to the TV T offices at receiving side	N-1 {hndvMet, user content} correctly	<del>- </del>
''		Generates XML_CONT_CONS for N-1 entities	consigned to N-1 entities at receiving	
			side	
12		Generates XML_CONT_CONS ERDS evidence for N-1		
		entities		

Table 4b: Scenarios for intra-ERDS with notifications for acceptance (3/6)

		Purpose		
	eter: {hndvMet, user content}	As the first scenario of the		
Param	eter: {NotificationForAcceptance}			present table but one of the
	eter: XML_SUB_ACC			entities at the receiving side
Param	eter: XML_NOT_F_ACC			does not answer in time.
	eter: XML_CONS_ACC			
	eter: XML_ACC_REJ_EXP			
Param	eter: XML_CONT_CONS			
		Sequence of actions		
#	Sender	ERDS	Receiving side	
1	Sender sends original message			
2		Accepts submission		
3		Generates and stores XML_SUB_ACC ERDS evidence		
4		Generates (NotificationForAcceptance) for N entities at		
		receiving side		_
5		Sends (NotificationForAcceptance) to N entities at		
		receiving side		
6		Generates and stores XML_NOT_F_ACC ERDS	N entities correctly receive	
		evidence for N entities	{NotificationForAcceptance}. N-1	
7		Descives N.4 positive ensures	accept. One does not answer in time	-
8		Receives N-1 positive answers		-
8		Generates {hndvMet, user content} structure for N-1 entities		
9		Consigns {hndvMet, user content} structure to the N-1		_
9		entities that have accepted		
10		When the expiration time is reached Generates and	(hadyMat upor content) etructure	-
		stores XML_CONS_ACC ERDS evidence for N-1 entities	{hndvMet, user content} structure successfully consigned to N-1 entities	
		and one XML_ ACC_REJ_EXP ERDS evidence for one	at receiving side	
		entity	at receiving side	

Table 4c: Scenarios for intra-ERDS with notifications for acceptance (4/6)

		Scenario id: ERDS_B	BB_NOT_F_ACC#4		Purpose
Para	meter: {hndvMet, user content}				As the first scenario but now
Para	meter: {NotificationForAcceptanc	e}			there are N successful
	meter: XML_NOT_F_ACC				handovers of original message.
Para	meter: XML_CONS_ACC				
Para	meter: XML_CONT_CONS				
Para	meter: XML_CONT_HAND				
		Sequence o	of actions		
#	Sender	ERDS		Receiving side	
1	Sender sends original message				
2	-	Accepts submission.			
3		Generates and stores XML_SUB_A	ACC ERDS evidence		
4		Generates (NotificationForAcceptar	nce}		
5		Generates and stores XML_NOT_F	_ACC ERDS evidence		
6		Sends (NotificationForAcceptance)			
7				All entities in receiving side receive one {NotificationForAcceptance} and answer positively	
8		Receives positive responses from r and stores XML_CONS_ACC ERD at receiving side			
9		Generates {hndvMet, user content}			
10		Consigns {hndvMet, user content} t	o N entities in receiving side		
11		Generates and stores XML_CONT_		N consignments of {hndvMet, user content} structure succeed	
12				N handovers of {hndvMet, user content} structure succeed	
13		Generates and stores XML_CONT_ entities	HAND ERDS evidence for N		

Table 4d: Scenarios for intra-ERDS with notifications for acceptance (5/6)

		Scenario id: ERDS_BB_NOT_F_ACC#5		Purpose
Para	meter: {hndvMet, user content}			As the first scenario but now
Para	meter: {NotificationForAcceptanc	re}		there are N-1 successful
Para	meter: XML_NOT_F_ACC			handovers of original message,
Para	meter: XML_CONS_ACC			and 1 failed handover.
Para	meter: XML_CONT_CONS			
Para	meter: XML_CONT_HAND			
Para	meter: XML_CONT_HAND_FAIL			
		Sequence of actions		
#	Sender	ERDS	Receiving side	
4	Sender sends original			
<u> </u>	message			
2		Accepts submission.		
3		Generates and stores XML_SUB_ACC ERDS evidence		]
4		Generates {NotificationForAcceptance}		
5		Generates and stores XML_NOT_F_ACC ERDS evidence		
6		Sends {NotificationForAcceptance}		
7			All entities in receiving side receive one	
			{NotificationForAcceptance}and answer	
8		Receives positive responses from receiving side and Generates	, , , , , , , , , , , , , , , , , , ,	1
		and stores XML_CONS_ACC ERDS evidence for the N entities		
		at receiving side		
9		Generates {hndvMet, user content} structure		]
10		Consigns {hndvMet, user content} to N entities in receiving side		
11			N consignments of {hndvMet, user	]
		Generates and stores XML_CONT_CONS ERDS evidence	content} structure succeed	
12			N-1 handovers of {hndvMet, user	
			content) structure succeed. One fails	
13		Generates and stores XML_CONT_HAND ERDS evidence for		]
		N-1 entities, and XML_CONT_HAND_FAIL for 1 entity		

Table 4e: Scenarios for intra-ERDS with notifications for acceptance (6/6)

		Scenario id: ERDS_BB_NOT_F_ACC#6		Purpose
Para	meter: {hndvMet, user content}			One of the entities at the
Para	meter: {NotificationForAcceptance}			receiving side does not accept
Para	meter: XML_SUB_ACC			consignment. In addition there
Para	meter: XML_NOT_F_ACC			are N-2 successful and one
Para	meter: XML_CONS_ACC			failing handovers.
Para	meter: XML_CONS_REJ			
	meter: XML_CONT_CONS			
	meter: XML_CONT_HAND			
Para	meter: XML_CONT_HAND_FAIL			
		Sequence of actions		
#	Sender	ERDS	Receiving side	
1	Sender sends original message			
2		Accepts submission.		
3		Generates and stores XML_SUB_ACC ERDS evidence		
4		Generates {NotificationForAcceptance} for N entities at		
		receiving side		_
5		Sends (NotificationForAcceptance) to N entities at receiving		
		side		_
6		O A A A A A A A A A A A A A A A A A A A	N entities correctly receive	
		Generates and stores XML_NOT_F_ACC ERDS evidence	{NotificationForAcceptance}. N-1	
		for N entities	accept. One does not accept consignment	
7		Receives N-1 positive answers and one negative answer	consignment	-
8		Generates XML_CONS_ACC ERDS evidence for N-1		
0		entities and one XML_CONS_REJ ERDS evidence for 1		
		entity		
9		Generates {hndvMet, user content} structure for N-1 entities		<b>-</b>
10		Consigns them to the N-1 entities at receiving side		<b>-</b>
11			N-1 {hndvMet, user content} correctly	7
		Generates XML_CONT_CONS for N-1 entities	consigned to N-1 entities at receiving	
			side	
12		Generates XML_CONT_CONS ERDS evidence for N-1		7
		entities		
13			N-2 handovers of {hndvMet, user	
<u> </u>			content} structure succeed. One fails	_
14		Generates and stores XML_CONT_HAND ERDS evidence		
		for N-2 entities, and XML_CONT_HAND_FAIL for 1 entity		

### 5.4 Scenarios for 4-corner model

### 5.4.1 Introduction

The present clause defines test cases for scenarios that take place when the sender and the entities at the receiving side are subscribed to different ERDSs but there are not intermediate ERDSs between the SERDS and the RERDSs.

Clause 5.4.2 defines test cases when RERDSs does not notify for acceptance.

Clause 5.4.3 defines test cases when RERDS notifies for acceptance.

### 5.4.2 Scenarios where RERDS does not use notification for acceptance

Table 5 defines a number of scenarios for the case where RERDSs does not notify for acceptance.

The scenarios are based on scenarios at Table 3 adding the relay acceptance and relay rejection events at the some of the RERDSs and their corresponding ERDS evidences.

For the sake of simplicity, it will be supposes that all the entities at receiving side are served by the same RERDS. It could be possible to use the templates defined in the present document for defining scenarios where the aforementioned entities are served by different RERDSs.

Table 5: Scenarios for RERDS not using notification for acceptance (1/6)

		Purpose			
Para	ameter: {hndvMet, EV_SET#1}	_REL_REJ}	The simplest negative scenario:		
		Sequence of a	ctions		the RERDS rejects the ERD
#	Sender	SERDS	RERDS	Receiving side	dispatch submitted by the
1	Sender sends original message				SERDS and sends back to the
2		Accepts submission			sender a structure with the set of ERDS evidences generated
		Generates and stores XML_SUB_ACC ERDS evidence			(including the XML_REL_REJ).
3		Generates <erd_dispatch>_with_XML_SUB_A CC</erd_dispatch>			
4		Relies the <erd_dispatch>_with_XML_SUB_A CC to the RERDS</erd_dispatch>			
5	Receives ERDS receipt		The RERDSs Rejects the <erd_dispatch>_with_XML_SUB_ ACC</erd_dispatch>		
		Generates and stores XML_REL_REJ ERDS evidence			
		Generates {hndvMet, EV_SET#1}			
		structure			
		Sends it back to sender			
	Sender receives {hndvMet, EV_SET#1}				

Table 5a: Scenarios for RERDS not using notification for acceptance (2/6)

		Scenario id: 4C_RERDS_N	O_NOT_F_ACC #2		Purpose
Parar	meter: <erd_dispatch_1>_with_&gt;</erd_dispatch_1>	XML_SUB_ACC			The simplest successful
	meter: <erds_receipt_1>_with_&gt;</erds_receipt_1>	scenario: the SERDS accepts			
	meter: {hndvMet, user content}	the submission of the original			
Parar	meter: <erds_receipt_2>_with_&gt;</erds_receipt_2>				message, generates one ERD dispatch and relies to RERDS.
		Sequence of a			This accepts relay, builds its own
#	Sender	SERDS	RERDS	Receiving side	ERD dispatch and delivers it to
1	Sender sends original message		_		the N recipients in receiving
2		Accepts submission. Generates and			side. SERDS generates and
		stores XML_SUB_ACC ERDS evidence			sends back to the sender a ERD
3		Generates			receipt with one
J		<erd_dispatch_1>_with_XML_SUB</erd_dispatch_1>			SubmissiongAcceptance ERDS evidence, one RelayAcceptance,
		ACC			and one ContentConsignment
4		Relies it to RERDS			ERDS evidence.
5			Accepts it and generates and		Clause 4.3.2.1 of TBCHECKED
			stores XML_REL_ACC ERDS		ETSI EN 319 522-1 [1]
			evidence		shows a variation of this
			Generates		scenario where RERDS sends
			<erds_receipt_1>_with_XML_RE L_ACC</erds_receipt_1>		XML_REL_ACC and
			Sends		XML_CONT_CONS in different ERD receipts.
			<erds_receipt_1>_with_XML_RE</erds_receipt_1>		EKD receipts.
			L_ACC to SERDS		
6		Receives	Generates {hndvMet, user content}		
		<erds_receipt_1>_with_XML_REL</erds_receipt_1>	struct		
		_ACC to SERDS			
7			Consigns it to the receiving side	N. acrainments of the define	-
8			Generates XML_CONT_CONS	N consignments of {hndvMet, user content} structure	
			ERDS evidence	succeed	
9			Generates		1
			<erds_receipt_2>_with_XML_</erds_receipt_2>		
			CONT_CONS		
10			Sends it back to the SERDS		]
11		Receives			
		<erds_receipt_2>_with_XML_</erds_receipt_2>			
		CONT_CONS and sends it back to			
12		the sender Stores the XML_CONT_CONS			-
12		ERDS evidence for N entities			
		EVDO ENIGELICE IOL IA GLITTIES			

Table 5b: Scenarios for RERDS not using notification for acceptance (3/6)

		Scenario id: 4C_RERDS_N	O_NOT_F_ACC#3		Purpose
	meter: <erd_dispatch_1>_with_2</erd_dispatch_1>	XML_SUB_ACC Var	EV_SET#1 = {XML_XML_ CONT_CO	DNS}	As scenario
	meter: {hndvMet, user content}				4C_RERDS_NO_NOT_F_ACC
	meter: <erds_receipt_1>_with_B</erds_receipt_1>				#2 but now finalized with hand
Para	meter: Generates <erd_receipt_< td=""><td></td><td></td><td></td><td>over and RelayAcceptance and ContentConsignment travel</td></erd_receipt_<>				over and RelayAcceptance and ContentConsignment travel
		Sequence of a			together in the same ERD
#	Sender	ERDS	RERDS	Receiving side	receipt back to SERDS
1	Sender sends original message			Sender sends original message	
2		Accepts submission. Generates and		message	1
_		stores XML_SUB_ACC ERDS			
		evidence			
3		Generates			
		<erd_dispatch_1>_with_XML_SUB</erd_dispatch_1>			
4		_ACC			-
<u>4</u>		Relies it to RERDS	Accepts it and generates and		-
3			stores Generates XML_REL_ACC		
			ERDS evidence		
6			Generates {hndvMet, user content}		1
			structure for N recipients		
7			Consigns {hndvMet, user content}		
			structure to the receiving side		_
8			Generates XML_CONT_CONS	N consignments of {hndvMet, user content} structure	
			ERDS evidence for N entities	succeed	
9			Generates		1
			<erds_receipt>_with_XML_</erds_receipt>		
			EV_SET#1		
10			Send	Entities in receiving side	
			<erds_receipt_1>_with_XML_</erds_receipt_1>	retrieve user content	
11		Receives	EV_SET#1 to SERDS		-
111		<erds_receipt_1>_with_EV_SET#1</erds_receipt_1>	Generates XML_CONT_HAND		
		and stores the evidences within the	ERDS evidence for N entities		
		set			
12			Generates		]
			<erd_receipt_2>_with_CONT_HA</erd_receipt_2>		
40			ND		
13			Sends <= RD_receipt_2>_with_CONT_HA		
			ND to SERDS		
<u> </u>			ILE TO OFFICE	I .	

	Purpose		
14	Receives <erds_receipt_2>with_XML_CO NT_HAND and stores the ERDS evidence</erds_receipt_2>		

### Table 5c: Scenarios for RERDS not using notification for acceptance (4/6)

		Purpose			
Para	meter: <erd_dispatch_1>_with_&gt;</erd_dispatch_1>	KML_SUB_ACC Var	EV_SET#1 = {XML_REL_ACC, XML		As the previous scenario but
Para	meter: {hndvMet, user content}	Var	EV_SET#2 = { XML_CONT_HAND, >	KML_CONT_HAND_FAIL }	now t one of the handovers fails.
	meter: <erds_receipt_1>_with_&gt;</erds_receipt_1>				Hereinafter, the scenarios do not
Para	meter: ERDS_receipt_2>_with_XI				show handovers, but only
		Sequence of a			consignment. However, a set of scenarios including handovers
#	Sender	SERDS	RERDS	Receiving side	could be easily built based on
1	Sender sends original message			Sender sends original message	them.
2		Accepts submission. Generates XML_SUB_ACC ERDS evidence			
3		Generates <erd_dispatch_1>_with_XML_SUB _ACC</erd_dispatch_1>			
4		Relies it to RERDS			]
5			Accepts it and generates and stores XML_REL_ACC ERDS evidence		
6			Generates {hndvMet, user content} structure for N recipients		
7			Consigns it to the receiving side		
8			Generates XML_CONT_CONS ERDS evidence	N consignments of {hndvMet, user content} structure succeed	
9			Generates <pre><erds_receipt>_with_EV_SET#1</erds_receipt></pre>		
10			Sends <erds_receipt>_with_EV_SET#1 to SERDS</erds_receipt>	N -1entities in receiving side successfully retrieve user content. One fails	
11		Receives <erds_receipt>_with_XML_ EV_SET#1 and stores ERDS evidences</erds_receipt>	Generates XML_CONT_HAND ERDS evidence for N-1 entities and one XML_CONT_HAND_FAIL ERDS evidence for one entity and stores them		
12			Generates <erd_receipt_2>_with_EV_SET# 2</erd_receipt_2>		

		Purpose			
13			Sends <erd_receipt_2>_with_EV_SET# 2to SERDS</erd_receipt_2>		
14		Receives <erd_receipt_2>_with_EV_SET#2a nd stores ERDS evidences</erd_receipt_2>			

### Table 5d: Scenarios for RERDS not using notification for acceptance (5/6)

		Purpose			
Parameter: <erd_dispatch_1>_with_XML_SUB_ACC</erd_dispatch_1>					As scenario
	Parameter: {hndvMet, user content}			4C_RERDS_NO_NOT_F_ACC	
Para	meter: <erds_receipt_1>_WITH_</erds_receipt_1>	_XML_REL_ACC to SERDS			#2 but now one of the ERD
Para	meter: <erds_receipt_2>_with_&gt;</erds_receipt_2>				dispatch consignments fails
		Sequence of a	ections		
#	Sender	SERDS	RERDS	Receiving side	
1	Sender sends original message				
2		Accepts submission. Generates and			
		stores XML_SUB_ACC ERDS			
		evidence			
3		Generates			
		<erd_dispatch_1>_with_XML_SUB</erd_dispatch_1>			
4		_ACC			-
<u>4</u>		Relies it to RERDS	Accepts it and generates and		-
5			stores XML_REL_ACC ERDS		
			evidence		
6			Generates		
			<erds_receipt_1>_WITH_XML_R</erds_receipt_1>		
			EL_ACC		
7			Sends		
			<erds_receipt_1>_WITH_XML_R</erds_receipt_1>		
			EL_ACC to SERDS		
		Receives	Generates {hndvMet, user content}		
		<erds_receipt_1>_WITH_XML_RE</erds_receipt_1>	structure for N entities		
		L_ACC to SERDS			_
			Consigns {hndvMet, user content}		
			structure to receiving side		
8				N-1 consignments of	
				{hndvMet, user content} structure succeed. One	
				consignment fails	
				porisigninent falls	

	Scenario id: 4C_RERDS_NO_NOT_F_ACC #5	Purpose
9	Generates XML_CONT_CONS ERDS evidence related to N-1 entities Generates XML_CONT_CONS_FAIL related to one entity	
10	Generates <erds_receipt_2>_with_XML_ EV_SET#1</erds_receipt_2>	
11	Sends it back to the SERDS	
12	Receives <erds_receipt_2>_with_XML_ EV_SET#1 and stores ERDS evidences</erds_receipt_2>	

The set of scenarios shown above could very easily be used for generating other scenarios where ERD payloads replace ERD dispatches.

Below follows the generation of a new scenario resulting from replacing ERD dispatches by ERD payloads in scenario 4C\_RERDS\_NO\_NOT\_F\_ACC #5. Differences between both scenarios have been highlighted in yellow.

Table 5e: Scenarios for RERDS not using notification for acceptance (6/6)

		Purpose			
Para	meter: <erd_payload_1></erd_payload_1>	Va	Var EV_SET#1 = {CONT_CONS, CON_CONS_FAIL}		As scenario
Para	meter: {hndvMet, user content}				4C_RERDS_NO_NOT_F_ACC
Para	meter: <erds_receipt_1>_WITH</erds_receipt_1>	_XML_REL_ACC to SERDS			#5 replacing ERD dispatches by
Para	meter: <erds_receipt_2>_with_&gt;</erds_receipt_2>	KML_ EV_SET#1			ERD payloads
		Sequence of	actions		
#	Sender	SERDS	RERDS	Receiving side	
1	Sender sends original message				
2		Accepts submission. Generates and			
		stores XML_SUB_ACC ERDS			
		evidence			
3		Generates < ERD_payload_1>			
4		Relies it to RERDS			
5			Accepts < ERD_payload_1>and		
			generates and stores		
			XML_REL_ACC ERDS evidence		
6		Generates			
			<erds_receipt_1>_WITH_XML_R</erds_receipt_1>		
			EL_ACC		

	Scenario id: 4C_RERDS_N	IO_NOT_F_ACC #6		Purpose
7		Sends <erds_receipt_1>_WITH_XML_R EL_ACC to SERDS</erds_receipt_1>		
	Receives <erds_receipt_1>_WITH_XML_RE L_ACC to SERDS</erds_receipt_1>	Generates {hndvMet, user content} structure for N entities		
		Consigns {hndvMet, user content} structure to receiving side		
8			N-1 consignments of {hndvMet, user content} structure succeed. One consignment fails	
9		Generates XML_CONT_CONS ERDS evidence related to N-1 entities Generates XML_CONT_CONS_FAIL related to one entity		
10		Generates <erds_receipt_2>_with_XML_ EV_SET#1</erds_receipt_2>		
11		Sends it back to the SERDS		
12	Receives <erds_receipt_2>_with_XML_ EV_SET#1 and stores ERDS evidences</erds_receipt_2>			

### 5.4.3 Scenarios where RERDS uses notification for acceptance

Table 6 defines a number of scenarios for the case where RERDS uses notification for acceptance.

The scenarios are based on scenarios at Table 4 adding the relay acceptance and relay rejection events at the some of the RERDSs and their corresponding ERDS evidences.

For the sake of simplicity, it is supposed that all the entities at receiving side are served by the same RERDS. It could be possible to use the templates defined in the present document for defining scenarios where the aforementioned entities are served by different RERDSs.

Again, a similar set of scenarios can be obtained by replacing ERD dispatches with ERD payloads in the scenarios shown below.

Table 6: Scenarios where RERDSs uses notification for acceptance (1/5)

Scenario id: 4C_REDS_NOT_F_ACC#1					Purpose
Parameter. <erd_dispatch_t>_with_xivit_50b_ACC</erd_dispatch_t>			Var EV_SET#1 = { XML_NOT_F_ACC, XML_CONS_ACC, XML_CONT_CONS, XML_CONT_HAND}		First scenario where the ERDS asks to all the entities at
Parameter: {hndvMet, user content}					receiving side for acceptance,
Parameter: <erds_receipt_1>_WITH_XML_REL_ACC</erds_receipt_1>					and all the entities at receiving
	meter: {NotificationforAccep				side accept.
Para	meter: <erds_receipt_2>_</erds_receipt_2>				
		Sequence o			
#	Sender	SERDS	RERDS	Receiving side	
1	Sender sends original message				
2		Accepts submission. Generates and stores XML_SUB_ACC ERDS evidence			
3		Generates <erd_dispatch_1>_with_XML_SUB_A CC</erd_dispatch_1>			
4		Relies <erd_dispatch_1>_with_XML_SUB_A CC to RERDS</erd_dispatch_1>			
5			Accepts <erd_dispatch_1>_with_XML_SUB_ ACC to RERDS</erd_dispatch_1>		
6			Generates and stores XML_REL_ACC ERDS evidence		
7			Generates <erds_receipt_1>_WITH_XML_REL _ACC</erds_receipt_1>		
8			Sends <erd_receipt_1>_WITH_XML_REL_ ACC back to SERDS</erd_receipt_1>		
9		Receives <erds_receipt_1>_WITH_XML_REL_A CC</erds_receipt_1>	Struct		
10			Sends {NotificationForAcceptance} struct to receiving side		
11			Generates XML_NOT_F_ACC ERDS evidence	All entities in receiving side answer positively	
12			Generates XML_CONS_ACC ERDS evidence		
13			Generates {hndvMet, user content} struct for N entities		
14			Consigns hndvMet, user content} struct to receiving side		

		Purpose			
15				N consignments of {hndvMet, user content} struct succeed	
16		Gene evide		All the entities retrieve user content	
17		Gene evide	nerates XML_CONT_HAND ERDS lence		
18			nerates RDS_receipt_2>_WITH_EV_SET#1		
19			ds RDS_receipt_2>_WITH_EV_SET#1 ERDS		
20	Receives <erds_receipt and="" erd<="" stores="" td=""><td>_2&gt;_WITH_EV_SET#1 S evidences</td><td></td><td></td><td></td></erds_receipt>	_2>_WITH_EV_SET#1 S evidences			

### Table 6a: Scenarios where RERDSs uses notification for acceptance (2/5)

		Purpose			
			Var EV_SET#1 = { XML_NOT_F_ACC, XML_CONS_ACC, XML_CONS_REJ, XML_CONT_CONS, CONT_HAND }		As previous scenario but one of the entities at the receiving side
Para	meter: <erd_dispatch_2>_</erd_dispatch_2>	with_XML_SUB_ACC		·	does not accept consignment.
<er< td=""><td>DS_receipt_1&gt;_WITH_XML</td><td>_REL_ACC</td><td></td><td></td><td></td></er<>	DS_receipt_1>_WITH_XML	_REL_ACC			
	meter: {NotificationforAccep	-			<u> </u>
Para	meter: <erds_receipt_2>_</erds_receipt_2>	with_EV_SET#1			
		Sequence of	of actions		
#	Sender	SERDS	RERDS	Receiving side	
1	Sender sends original				
	message				
2		Accepts submission. Generates and			
		stores XML_SUB_ACC ERDS evidence	)		
3		Generates			
		<erd_dispatch_1>_with_XML_SUB_A</erd_dispatch_1>			
4		CC Relies			_
4		<pre><erd_dispatch_1>_with_XML_SUB_A</erd_dispatch_1></pre>			
		CC to RERDS			
5		00 10 1121120	Accepts		
1			<erd_dispatch_1>_with_XML_SUB_</erd_dispatch_1>		
			ACC to RERDS		
6			Generates XML_REL_ACC ERDS		
			evidence		

	Scenario id: 4C_RERDS	NOT F ACC##2		Purpose
7	_	Generates <erds_receipt_1>_WITH_XML_REL _ACC</erds_receipt_1>		
8		Sends <erd_receipt_1>_WITH_XML_REL_ ACC back to SERDS</erd_receipt_1>		
9	Receives <erds_receipt_1>_WITH_XML_REL_A CC and stores ERDS evidence</erds_receipt_1>	Generates {NotificationforAcceptance} structure		
10		Sends {NotificationforAcceptance} structure to receiving side		
11		Generates and stores XML_NOT_F_ACC ERDS evidence	N-1 entities in receiving side answer positively. One answers negatively	
12		Generates and stores XML_CONS_ACC ERDS evidence for N-1 entities and one XML_CONS_REJ ERDS evidence for one entity		
13		Generates <{hndvMet, user content} struct		
14		Sends {hndvMet, user content} structure to N-1 accepting entities at receiving side		
15			{hndvMet, user content} structure consigned to N-1 entities in receiving side	
16		Generates XML_CONT_CONS ERDS evidence for N-1 entities	N-1 entities retrieve user content	
17		Generates XML_CONT_HAND ERDS evidence for N-1 entities		
18		Generates <erds_receipt_2>_WITH_EV_SET#1</erds_receipt_2>		
19		Sends <erds_receipt_2>_WITH_EV_SET#1 to SERDS</erds_receipt_2>		
20	Receives <erds_receipt_2>_WITH_EV_SET#1, extracts ERDS evidences and stores them</erds_receipt_2>			

Table 6b: Scenarios where RERDSs uses notification for acceptance (3/5)

		Scenario id: 4C_RERD	S_NOT_F_ACC##3		Purpose
	meter: <erd_dispatch_1>_</erd_dispatch_1>	_with_XML_SUB_ACC \	/ar EV_SET#1 = {		As previous scenario but one of the entities at the receiving side
	meter: {hndvMet, user cont				does not answer in time.
	DS_receipt_1>_WITH_XML				
	meter: {NotificationforAccept				_
Para	meter: <erds_receipt_2>_</erds_receipt_2>				
		Sequence o			
#	Sender	SERDS	RERDS	Receiving side	
1	Sender sends original message				
2		Accepts submission. Generates and stores XML_SUB_ACC ERDS evidence			
3		Generates <erd_dispatch_1>_with_XML_SUB_A CC</erd_dispatch_1>			
4		Relies <erd_dispatch_1>_with_XML_SUB_A CC to RERDS</erd_dispatch_1>			
5			Accepts <erd_dispatch_1>_with_XML_SUB_ ACC to RERDS</erd_dispatch_1>		
6			Generates XML_REL_ACC ERDS evidence		
7			Generates <erds_receipt_1>_WITH_XML_REL _ACC</erds_receipt_1>		
8			Sends <erd_receipt_1>_WITH_XML_REL_ ACC back to SERDS</erd_receipt_1>		
9		Receives <erds_receipt_1>_WITH_XML_REL_A CC</erds_receipt_1>	Generates {NotificationforAcceptance} structure		
10			Sends {NotificationforAcceptance} structure to receiving side		
11			Generates XML_NOT_F_ACC ERDS evidence	N-1 entities in receiving side answer positively. One does not answer in time	
12			Generates XML_CONS_ACC ERDS evidence for N-1 entities and one XML_ACC_REJ_EXP ERDS evidence for one entity		
13			Generates {hndvMet, user content} structure for N-1 entities		

	Scenario id: 4C_RERDS	_NOT_F_ACC##3		Purpose
14		Sends {hndvMet, user content} structure to N-1 accepting entities at receiving side		
15			{hndvMet, user content} structure consigned to N-1 entities in receiving side	
16		Generates XML_CONT_CONS ERDS evidence	N-1 entities retrieve user content	
17		Generates XML_CONT_HAND ERDS evidence for N-1 entities		
18		Generates <erds_receipt_2>_WITH_EV_SET#1</erds_receipt_2>		
19		Sends <erds_receipt_2>_WITH_EV_SET#1 to SERDS</erds_receipt_2>		
20	Receives <erds_receipt_2>_WITH_EV_SET#, extracts ERDS evidences and stores them</erds_receipt_2>			

## Table 6c: Scenarios where RERDSs uses notification for acceptance (4/5)

		Purpose			
Para	meter: <erd_dispatch_1>_</erd_dispatch_1>	with_XML_SUB_ACC \	/ar EV_SET#1 = {XML_NOT_F_ACC, XML	_CONS_ACC,	As first scenario in the present
		>	<u> (ML_CONT_CONS, XML_CONS_FAIL, XN</u>	/IL_CONT_HAND }	table, but one of the
Para	meter: {hndvMet, user conte	nt}			consignments fails.
Para	meter: <erds_receipt_1>_\</erds_receipt_1>	WITH_XML_REL_ACC			
Para	meter: {NotificationforAccept	tance}			
Para	meter: <erds_receipt_2>_v</erds_receipt_2>	with_ EV_SET#1			
		Sequence o	f actions		
#	Sender	SERDS	RERDS	Receiving side	
1	Sender sends original				
<u> </u>	message				
2		Accepts submission. Generates XML_SUB_ACC ERDS evidence			
3		Generates <erd_dispatch_1>_with_XML_SUB_A CC</erd_dispatch_1>			
4		Relies <erd_dispatch_1>_with_XML_SUB_A CC to RERDS</erd_dispatch_1>			
5			Accepts <erd_dispatch_1>_with_XML_SUB_ ACC to RERDS</erd_dispatch_1>		

	Scenario id: 4C_RERDS	NOT F ACC##4		Purpose
6		Generates XML_REL_ACC ERDS evidence		·
7		Generates <erds_receipt_1>_WITH_XML_REL _ACC</erds_receipt_1>		
8		Sends <erd_receipt_1>_WITH_XML_REL_ ACC back to SERDS</erd_receipt_1>		
9	Receives <erds_receipt_1>_WITH_XML_REL_A CC</erds_receipt_1>	Generates {NotificationforAcceptance} structure for N entities		
10		Sends (NotificationforAcceptance) structure to receiving side		
11		Generates XML_NOT_F_ACC ERDS evidence	All the entities in receiving side answer positively	
12		Generates XML_CONS_ACC ERDS evidence for all the entities in receiving side	, , , , , , , , , , , , , , , , , , , ,	
13		Generates {hndvMet, user content} structure for N entities		
14		Sends {hndvMet, user content} structure to N entities		
15			N-1 {hndvMet, user content} structure consignments succeed. One fails	
16		Generates XML_CONT_CONS ERDS evidence for N-1 entities and XML_CONS_FAIL for one entity	N-1 entities retrieve user content	
17		Generates XML_CONT_HAND ERDS evidence		
18		Generates <erds_receipt_2>_WITH_EV_SET#1</erds_receipt_2>		
19		Sends <erds_receipt_2>_WITH_EV_SET#1 to SERDS</erds_receipt_2>		
20	Receives <erds_receipt_2>_WITH_EV_SET#1, and stores the ERDS evidences</erds_receipt_2>			

Table 6d: Scenarios where RERDSs uses notification for acceptance (5/5)

		Scenario id: 4C_RERD	S_NOT_F_ACC##5		Purpose
Para	meter: <erd_dispatch_1>_</erd_dispatch_1>	with_XML_SUB_ACC	/ar EV_SET#1 = {XML_NOT_F_ACC, XM (ML_CONT_CONS, XML_CONS_FAIL, X (ML_CONT_HAND_FAIL)		As first scenario in the present table, but one of the retrievals fails.
	meter: {hndvMet, user conte				
	DS_receipt_1>_WITH_XML				
	meter: {NotificationforAccep				
Para	meter: <erds_receipt_2>_</erds_receipt_2>				
		Sequence of			
#	Sender	SERDS	RERDS	Receiving side	
1	Sender sends original message				
2		Accepts submission. Generates and stores XML_SUB_ACC ERDS evidence			
3		Generates <erd_dispatch_1>_with_XML_SUB_A CC</erd_dispatch_1>			
4		Relies <erd_dispatch_1>_with_XML_SUB_A CC to RERDS</erd_dispatch_1>			
5			Accepts <erd_dispatch_1>_with_XML_SUB_ ACC to RERDS</erd_dispatch_1>		
6			Generates XML_REL_ACC ERDS evidence		
7			Generates <erds_receipt_1>_WITH_XML_REL _ACC</erds_receipt_1>		
8			Sends <erd_receipt_1>_WITH_XML_REL_ ACC back to SERDS</erd_receipt_1>		
9		Receives <erds_receipt_1>_WITH_XML_REL_A CC</erds_receipt_1>	Generates {NotificationforAcceptance} structure		
10			Sends (NotificationforAcceptance) structure to receiving side		
11			Generates XML_NOT_F_ACC ERDS evidence	All the entities in receiving side answer positively	
12			Generates XML_CONS_ACC ERDS evidence for all the entities in receiving side		
13			Generates {hndvMet, user content} for N entities		

	Scenario id: 4C_RERDS	S_NOT_F_ACC##5		Purpose
14		Sends {hndvMet, user content} to N-1 accepting entities at receiving side		
15			N consignments of {hndvMet, user content} structure succeed	
16		Generates XML_CONT_CONS ERDS evidence	N-1 entities retrieve user content. One fails	
17		Generates XML_CONT_HAND ERDS evidence for N-1 entities and one XML_CONT_HAND_FAIL for one entity		
18		Generates <a href="#"><erds_receipt_2>_WITH_EV_SET#1</erds_receipt_2></a>		
19		Sends <erds_receipt_2>_WITH_EV_SET#1 to SERDS</erds_receipt_2>		
20	Receives <erds_receipt_2>_WITH_EV_SET#1 and stores ErDS evidences</erds_receipt_2>			

43

## 5.5 Scenarios for extended model

## 5.5.1 Introduction

The present clause defines test cases for scenarios that take place when the sender and the entities at the receiving side are subscribed to different ERDSs and there is one intermediate IERDS between the SERDS and the RERDSs.

Clause 5.5.2 defines test cases where the RERDSs does not use notification for acceptance.

Clause 5.5.3 defines test cases where the RERDS uses notification for acceptance.

As with previous scenarios defined in the present document, new sets of scenarios can be obtained by replacing ERD dispatches with ERD payloads in the scenarios shown below.

## 5.5.2 Scenarios where RERDS does not use notification for acceptance

Table 7 shows scenarios where RERDS does not use notification for acceptance.

Table 7: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (1/5)

		Scena	rio id: EXT_RERDS_NO_NOT_	F_ACC#1		Purpose
#	Sender	SERDS	IERDS	RERDS	Receiving side	•
Para	meter: <erd_dispatch_< td=""><td>1&gt;_with_XML_SUB_ACC</td><td></td><td></td><td></td><td>First scenario where</td></erd_dispatch_<>	1>_with_XML_SUB_ACC				First scenario where
Para	meter: <erds_receipt_< td=""><td>1&gt;_with_XML_REL_ACC</td><td></td><td></td><td></td><td>the N entities</td></erds_receipt_<>	1>_with_XML_REL_ACC				the N entities
		2>_with_XML_REL_ACC				successfully retrieve
	meter: {hndvMet, user co					the ERD dispatch with
		3>_with_XML_CONT_CONS				the user content.
Para	meter: <erds_receipt_4< td=""><td>4&gt;_with_XML_CONT_HAND</td><td></td><td></td><td></td><td></td></erds_receipt_4<>	4>_with_XML_CONT_HAND				
			Sequence of actions			
#	Sender	SERDS	IERDS	RERDS	Receiving side	
1	Sender sends original message					
2		Accepts submission and				
1		generates and stores				
		XML_SUB_ACC ERDS evidence				
3		Generates				
		<erd_dispatch_1>_with_X</erd_dispatch_1>				
		ML_SUB_ACC				
4		Relies it to IERDS				
5			Receives			
			<erd_dispatch_1>_with_XM L_SUB_ACC</erd_dispatch_1>			
6			Generates and stores			
			XML_REL_ACC ERDS			
7			evidence Generates			
1			<erds_receipt_1>_with_XM</erds_receipt_1>			
			L_REL_ACC			
8			Sends it back to SERDS			
9		Receives	Relies			
		<erds_receipt_1>_with_X</erds_receipt_1>	<pre><erd_dispatch_1>_with_XM</erd_dispatch_1></pre>			
		ML_REL_ACC and stores	L_SUB_ACC to RERDS			
4.0		ERDS evidence		<u></u>		
10				Receives		
				<erd_dispatch_1>_with_XML_ SUB_ACC to RERDS</erd_dispatch_1>		
11				Generates XML_REL_ACC		<del></del>
' '				ERDS evidence		
12				Generates		
-				<erds_receipt_2>_with_XML_</erds_receipt_2>		
				SUB_ACC		
13				Sends it back to IERDS		

		Scena	rio id: EXT_RERDS_NO_NOT_	F_ACC#1		Purpose
#	Sender	SERDS	IERDS	RERDS	Receiving side	•
14			Receives <erds_receipt_2>_with_XM L_SUB_ACC</erds_receipt_2>	Generates {hndvMet, user content} structure		
15				Consigns {hndvMet, user content} structure to receiving side		
16					N consignments of {hndvMet, user content} structure succeed	
17				Generates XML_CONT_CONS ERDS evidence for N entities		
18				Generates <erds_receipt_3>_with_XML_ CONT_CONS</erds_receipt_3>	All the entities in receiving side successfully retrieve {hndvMet, user content} structure	
19				Sends it back to IERDS		
20			Receives <erds_receipt_3>_with_XM L_CONT_CONS</erds_receipt_3>	Generates XML_CONT_HAND for N entities		
21			Sends it back to SERDS	Generates <erds_receipt_4>_with_XML_ CONT_HAND</erds_receipt_4>		
22		Receives <erds_receipt_3>_with_X ML_CONT_CONS and stores ERDS evidences</erds_receipt_3>		Sends it back to IERDS		
23		Sends it back to sender	Receives <erds_receipt_4>_with_XM L_CONT_HAND</erds_receipt_4>			
24	Receives <erds_receipt_3>_wi th_XML_CONT_CON S</erds_receipt_3>		Sends it back to SERDS			
25		Receives <erds_receipt_4>_with_X ML_CONT_HAND and stores ERDS evidences</erds_receipt_4>				

Table 7a: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (2/5)

	Scenario id: EXT_RERDS_NO_NOT_F_ACC#2						
#	Sender	SERDS	IERD	S	RERDS	Receiving side	
		1>_with_XML_SUB_ACC I>_with_XML_REL_ACC		Var SET_EV	#1 = { XML_CONT_CONS, XML_C		As the first scenario but now one of the
		2>_with_XML_REL_ACC					consignments fails.
	meter: {hndvMet, user co						
	meter: <erds_receipt_3< td=""><td></td><td></td><td></td><td></td><td></td><td></td></erds_receipt_3<>						
Para	meter: <erds_receipt_4< td=""><td>1&gt;_with_XML_CONT_HAND</td><td>Common</td><td>of cotions</td><td></td><td></td><td></td></erds_receipt_4<>	1>_with_XML_CONT_HAND	Common	of cotions			
- ш	Sender	SERDS	Sequence IERD		RERDS	Desciving olds	
#		SERUS	IERD	3	RERUS	Receiving side	
1	Sender sends original message						
2		Accepts submission and generates and stores XML_SUB_ACC ERDS evidence					
3		Generates <erd_dispatch_1>_with_X ML_SUB_ACC</erd_dispatch_1>					
4		Relies it to IERDS					
5			Receives <erd_dispatch_ L_SUB_ACC</erd_dispatch_ 				
6			Generates XML_ ERDS evidence				
7			Generates <erds_receipt_ L_REL_ACC</erds_receipt_ 				
8			Sends it back to	SERDS			
9		Receives <erds_receipt_1>_with_X ML_REL_ACC and stores ERDS evidence</erds_receipt_1>	Relies <erd_dispatch_ L_SUB_ACC to I</erd_dispatch_ 				
10					Receives <erd_dispatch_1>_with_XML_ SUB_ACC</erd_dispatch_1>		
11					Generates and stores XML_REL_ACC ERDS evidence		
12					Generates <erds_receipt_2>_with_XML_ REL_ACC</erds_receipt_2>		
13					Sends it back to IERDS		

Scenario id: EXT_RERDS_NO_NOT_F_ACC#2								
#	Sender	SERDS	IERDS	RERDS	Receiving side	•		
14			Receives <erds_receipt_2>_with_XM L_REL_AC and stores ERDS evidence</erds_receipt_2>	Generates {hndvMet, user content} structure for N entities				
15				Consigns it to receiving side				
16					N-1 {hndvMet, user content} structure consignments succeed. One fails			
17				Generates and stores XML_CONT_CONS ERDS evidence for N-1 entities and XML_CONT-CONS_FAIL ERDS evidence for one entity				
18				Generates <erds_receipt_3>_with_EV_S ET#1</erds_receipt_3>	N-1 entities in receiving side retrieve {hndvMet, user content} structure			
19				Sends it back to IERDS				
20			Receives <erds_receipt_3>_with_EV_ SET#1</erds_receipt_3>	Generates and stores XML_CONT_HAND for N-1 entities				
21			Sends it back to SERDS	Generates <erds_receipt_4>_with_XML_ CONT_HAND</erds_receipt_4>				
22		Receives <erds_receipt_3>_with_E V_SET#1 and stores ERDS evidences</erds_receipt_3>		Sends it back to IERDS				
23			Receives <erds_receipt_4>_with_XM L_CONT_HAND</erds_receipt_4>					
24			Sends it back to SERDS					
25		Receives <erds_receipt_4>_with_X ML_CONT_HAND and stores ERDS evidences</erds_receipt_4>						

Table 7b: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (3/5)

		Scenar	io id: EXT_RERD	S NO NOT	F ACC #3		Purpose
Para	meter: <erd_dispatch_1< td=""><td>&gt;_with_XML_SUB_ACC</td><td></td><td></td><td>#1 = { XML_CONT_HAND, XML_C</td><td>ONT_HAND_FAIL }</td><td>As the first scenario</td></erd_dispatch_1<>	>_with_XML_SUB_ACC			#1 = { XML_CONT_HAND, XML_C	ONT_HAND_FAIL }	As the first scenario
Para	meter: <erds_receipt_1< td=""><td>&gt;_with_XML_REL_ACC</td><td></td><td></td><td></td><td>•</td><td>but now one of the</td></erds_receipt_1<>	>_with_XML_REL_ACC				•	but now one of the
		!>_with_XML_REL_ACC					entities fails when
	meter: {hndvMet, user co			Parameter: {h	indvMet, user content}		trying to retrieve the
		>_with_XML_CONT_CONS					user content.
Para	meter: <erds_receipt_4< td=""><td>-&gt;_with_EV_SET#1</td><td></td><td></td><td></td><td></td><td></td></erds_receipt_4<>	->_with_EV_SET#1					
	<u> </u>	1	Sequence				
#	Sender	SERDS	IERD	<u>S</u>	RERDS	Receiving side	
1	Sender sends original message						
	message	Accepts submission and					<del> </del>
_		generates and stores					
2		XML_SUB_ACC ERDS					
		evidence					
		Generates					
3		<erd_dispatch_1>_with_X</erd_dispatch_1>					
		ML_SUB_ACC					
4		Relies it to IERDS	Danaina				_
5			Receives <erd_dispatch_< td=""><td>1. with VM</td><td></td><td></td><td></td></erd_dispatch_<>	1. with VM			
5			L_SUB_ACC	ı>_wıtıı_∧ıvı			
			Generates and st	tores			
6			XML_REL_ACC	ERDS			
			evidence				
_			Generates				
7			<erds_receipt_< td=""><td>1&gt;_with_XM</td><td></td><td></td><td></td></erds_receipt_<>	1>_with_XM			
8			L_REL_ACC Sends it back to \$	SERDS			
		Receives					
9		<erds_receipt_1>_with_X</erds_receipt_1>	Relies <erd_dispatch_< td=""><td>1- with VM</td><td></td><td></td><td></td></erd_dispatch_<>	1- with VM			
9		ML_REL_ACC and stores	L_SUB_ACC to F				
		ERDS evidence	L_00B_7100 to 1	KENDO			
4.0					Receives		
10					<erd_dispatch_1>_with_XML_</erd_dispatch_1>		
					SUB_ACC to RERDS Generates and stores		<del> </del>
11					XML_REL_ACC ERDS		
					evidence		
					Generates		
12					<erds_receipt_2>_with_XML_</erds_receipt_2>		
					SUB_ACC		
13					Sends it back to IERDS		

	Scenar	io id: EXT_RERDS_NO_NOT_	F_ACC #3		Purpose
14		Receives <erds_receipt_2>_with_XM L_SUB_ACC and stores ERDS evidence</erds_receipt_2>	Generates {hndvMet, user content} structure		·
15			Consigns it to receiving side		
16				N {hndvMet, user content} structure consignments succeed	
17			Generates and stores XML_CONT_CONS ERDS evidence for N entities		
18			Generates <erds_receipt_3>_with_ XML_CONT_CONS</erds_receipt_3>	N-1 entities in receiving side successfully retrieve {hndvMet, user content} structure. One entity fails	
19			Sends it back to IERDS		
20		Receives <erds_receipt_3>_with_ XML_CONT_CONS</erds_receipt_3>	Generates XML_CONT_HAND ERDS evidence for N-1 entities and XML_CONT_HAND_FAIL for one entity		
21		Sends it back to SERDS	Generates <erds_receipt_4>_with_EV_S ET#1</erds_receipt_4>		
22	Receives <erds_receipt_3>_with_ XML_CONT_CONS and stores ERDS evidences</erds_receipt_3>		Sends it back to IERDS		
23		Receives <erds_receipt_4>_with_EV_ SET#1</erds_receipt_4>			
24		Sends it back to SERDS			
25	Receives <erds_receipt_4>_with_E V_SET#1 and stores ERDS evidences</erds_receipt_4>				

Table 7c: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (4/5)

		Scena	rio id: EXT_RERDS_NO_NOT_F	ACC #4		Purpose
#	Sender	SERDS	IERDS	RERDS	Receiving side	
		1>_with_XML_SUB_ACC				As the first scenario
	meter: <erds_receipt_1< td=""><td></td><td></td><td></td><td></td><td>but now the IERDS</td></erds_receipt_1<>					but now the IERDS
Para	meter: {hndvMet,XML_R	EL_REJ}				rejects relaying.
.,	01	05000	Sequence of actions	DEDDO		
#	Sender	SERDS	IERDS	RERDS	Receiving side	
1	Sender sends original message					
2	, and the second	Accepts submission and generates and stores XML_SUB_ACC ERDS evidence				
3		Generates <erd_dispatch_1>_with_X ML_SUB_ACC</erd_dispatch_1>				
4		Relies it to IERDS				
5			Rejects relying			
6			Generates and stores XML_REL_REJ ERDS evidence			
7			Generates <erds_receipt_1>_with_XM L_REL_REJ</erds_receipt_1>			
8			Sends it back to SERDS			
9		Receives <erds_receipt_1>_with_X ML_REL_REJ and stores ERDS evidence</erds_receipt_1>				
10		Generates {hndvMet,XML_REL_REJ} structure				
11		Sends {hndvMet,XML_REL_REJ} structure back to sender				
12	Receives {hndvMet,XML_REL_ REJ} structure					

Table 7d: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (5/5)

		Scena	rio id: EXT_RERDS_NO_NOT_	F_ACC #5		Purpose
#	Sender	SERDS	IERDS	RERDS	Receiving side	
ara		1>_with_XML_SUB_ACC 1>_with_XML_REL_ACC 2>_with_XML_REL_RE.L				As the first scenario but now the RERDS rejects relaying
uic	ameter: <endo_receipt_z< td=""><td></td><td>Sequence of actions</td><td></td><td></td><td></td></endo_receipt_z<>		Sequence of actions			
#	Sender	SERDS	IERDS	RERDS	Receiving side	
1	Sender sends original message					
2	gu	Accepts submission and generates XML_SUB_ACC ERDS evidence				
3		Generates <erd_dispatch_1>_with_X ML_SUB_ACC</erd_dispatch_1>				
4		Relies it to IERDS				
5			Receives <erd_dispatch_1>_with_XM L_SUB_ACC</erd_dispatch_1>			
6			Generates XML_REL_ACC ERDS evidence			
7			Generates <erds_receipt_1>_with_XM L_REL_ACC</erds_receipt_1>			
8			Sends it back to SERDS			
9		Receives <erds_receipt_1>_with_X ML_REL_ACC</erds_receipt_1>	Relies <erd_dispatch_1>_with_XM L_SUB_ACC to RERDS</erd_dispatch_1>			
10				Receives <erd_dispatch_1>_with_XML_ SUB_ACC but it rejects it</erd_dispatch_1>		
11				Generates XML_REL_REJ ERDS evidence		
12				Generates <erds_receipt_2>_with_XML_ REL_REJ</erds_receipt_2>		
13				Sends it back to IERDS		
14			Receives <erds_receipt_2>_with_XM L_REL_REJ</erds_receipt_2>			
15			Sends it back to SERDS			
16		Receives <erds_receipt_2>_with_X ML_REL_REJ</erds_receipt_2>				

	Scenario id: EXT_RERDS_NO_NOT_F_ACC #5						
#	Sender	SERDS	IERDS	RERDS	Receiving side		
17		Sends it back to sender					
18	Receives <erds_receipt_2>_wi th_XML_REL_REJ</erds_receipt_2>						

# 5.5.3 Scenarios where RERDS uses notification for acceptance

Table 8 shows scenarios where RERDS submits notification for acceptance.

Table 8: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (1/3)

		Scer	nario id: EXT_RERDS_NOT_F_A	ACC #1		Purpose
#	Sender	SERDS	IERDS	RERDS	Receiving side	
Para	meter: <erd_dispatch_1< td=""><td>&gt;_with_XML_SUB_ACC</td><td></td><td></td><td></td><td>First scenario where all</td></erd_dispatch_1<>	>_with_XML_SUB_ACC				First scenario where all
	meter: <erds_receipt_1< td=""><td></td><td></td><td></td><td></td><td>the entities at receiving</td></erds_receipt_1<>					the entities at receiving
	meter: {NotificationforAcc					side successfully
		>_with_XML_ REL_ACC				retrieve the user
		>_with_ XML_NOT_F_ACC				content
		>_with_ XML_CONS_ACC				_
Para	meter: {hndvMet, user co	ntent}	Coguenes of estions			<del>-</del>
	Candan	CERRO	Sequence of actions	DEDDE	Desciving side	<del>-</del>
#	Sender	SERDS	IERDS	RERDS	Receiving side	<del> </del>
1	Sender sends original message					
2	Illessage	Accepts submission and				<del> </del>
_		generates and stores				
		XML_SUB_ACC ERDS				
		evidence				
3		Generates				
		<erd_dispatch_1>_with_X</erd_dispatch_1>				
4		ML_SUB_ACC				<del>_</del>
<u>4</u> 5		Relies it to IERDS	Receives			
3			<erd_dispatch_1>_with_XM</erd_dispatch_1>			
			L_SUB_ACC			
6			Generates and stores			
			XML_REL_ACC ERDS			
			evidence			
7			Generates			
			<erds_receipt_1>_with_XM</erds_receipt_1>			
			L_SUB_ACC			

		Scer	nario id: EXT_RERDS_NOT_F_	ACC #1		Purpose
#	Sender	SERDS	IERDS	RERDS	Receiving side	
3			Sends it back to SERDS			
9		Receives <erds_receipt_1>_with_X ML_SUB_ACC and stores ERDS evidence</erds_receipt_1>	Relies <erd_dispatch_1>_with_XM L_SUB_ACC } to RERDS</erd_dispatch_1>			
0				Receives <erd_dispatch_1>_with_XML_ SUB_ACC</erd_dispatch_1>		
11				Generates and stores XML_REL_ACC ERDS evidence		
12				Generates <erds_receipt_2>_with_XML_ REL_ACC</erds_receipt_2>		
13				Sends <erds_receipt_2>_with_XML_ REL_ACC back to IERDS</erds_receipt_2>		
14			Receives <erds_receipt_2>_with_XM L_REL_ACC</erds_receipt_2>	Sends {NotificationforAcceptance}		
15				Generates and stores XML_NOT_F_ACC ERDS evidence	All the parties at receiving side receive {NotificationforAcceptance}	
16				Generates <erds_receipt_3>_with_XML_ NOT_F_ACC</erds_receipt_3>	All the parties at receiving side access IERDS and accept consignment	
17				Sends it back to IERDS		
18			Receives <erds_receipt_3>_with_XM L_NOT_F_ACC</erds_receipt_3>	Generates {hndvMet, user content} structure		
19			Sends it back to SERDS	Consigns {hndvMet, user content} structure to N entities at receiving side		
20		Receives <erds_receipt_3>_with_X ML_NOT_F_ACC and stores ERDS evidences</erds_receipt_3>		Generates and stores XML_CONS_ACC N entities	N {hndvMet, user content} structure consignments succeed	
21				Generates <erds_receipt_4>_with_XML_ CONS_ACC</erds_receipt_4>		
22				Sends it back to IERDS		
23			Receives <erds_receipt_4>_with_XM L_CONS_ACC</erds_receipt_4>			

	Scenario id: EXT_RERDS_NOT_F_ACC #1					
#	Sender	SERDS	IERDS	RERDS	Receiving side	
24			Sends it back to SERDS			
25		Receives				
		<erds_receipt_4>_with_X</erds_receipt_4>				
		ML_CONS_ACC and stores				
		ERDS evidence				

## Table 8a: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (2/3)

		Scen	ario id: EXT_RER	RDS_NOT_F_	ACC #2		Purpose
#	Sender	SERDS	IERDS	S	RERDS	Receiving side	
		>_with_XML_SUB_ACC		Var EV_SET#	1 = {XML_CONS_ACC, XML_	CONS_REJ}	As first scenario but
	meter: <erds_receipt_1< td=""><td></td><td></td><td></td><td></td><td></td><td>now one of the entities</td></erds_receipt_1<>						now one of the entities
Para	meter: {NotificationforAcc	ceptance}					at receiving side
		>_with_XML_ REL_ACC					rejects consignment
		>_with_ XML_NOT_F_ACC					
	meter: <erds_receipt_4< td=""><td></td><td></td><td></td><td></td><td></td><td></td></erds_receipt_4<>						
		>_with_ XML_REL_ACC					
	meter: {hndvMet, user co						
		>_with_XML_CONT_CONS					
Para	meter: <erds_receipt_ <="" td=""><td>&gt;_with_ XML_CONT_HAND</td><td>0</td><td>-f</td><td></td><td></td><td><u> </u></td></erds_receipt_>	>_with_ XML_CONT_HAND	0	-f			<u> </u>
<del></del>		25550	Sequence of		55556		
#	Sender	SERDS	IERDS	<u>s</u>	RERDS	Receiving side	
1	Sender sends original						
2	message	Asserts submission and					
2		Accepts submission and generates and stores					
		XML_SUB_ACC ERDS					
		evidence					
3		Generates					
		<erd_dispatch_1>_with_X</erd_dispatch_1>					
		ML_SUB_ACC					
4		Relies it to IERDS					
5			Receives				
			<erd_dispatch_< td=""><td>1&gt;_with_XM</td><td></td><td></td><td></td></erd_dispatch_<>	1>_with_XM			
			L_SUB_ACC				
6			Generates and st				
			XML_REL_ACC E evidence	בעטס			
7			Generates				
'			<erds_receipt_1< td=""><td>1 with XM</td><td></td><td></td><td></td></erds_receipt_1<>	1 with XM			
			L_SUB_ACC	1.5 _ ANITH _ VIAN			
8			Sends it back to S	SERDS			

		Scer	nario id: EXT_RERDS_NOT_F_			Purpose
#	Sender	SERDS	IERDS	RERDS	Receiving side	
9		Receives <erds_receipt_1>_with_X ML_SUB_ACC and stores ERDS evidence</erds_receipt_1>	Generates {NotificationforAcceptance}			
10			Relies <erds_notification_1>_forAc ceptance to RERDS</erds_notification_1>			
11				Receives <erds_notification_1>_forAcc eptance</erds_notification_1>		
12				Generates XML_REL_ACC ERDS evidence		
13				Generates <erds_receipt_2>_with_XML_ REL_ACC</erds_receipt_2>		
14				Sends <erds_receipt_2>_with_XML_ REL_ACC back to IERDS</erds_receipt_2>		
15			Receives <erds_receipt_2>_with_XM L_REL_ACC</erds_receipt_2>	Sends {NotificationforAcceptance}		
16				Generates and stores XML_NOT_F_ACC ERDS evidence	All the parties at receiving side receive {NotificationforAcceptance}	
17				Generates <erds_receipt_3>_with_XML_ NOT_F_ACC</erds_receipt_3>		
18				Sends it back to IERDS		
19			Receives <erds_receipt_3>_with_XM L_NOT_F_ACC</erds_receipt_3>	Generates and stores XML_CONS_ACC ERDS evidence for N-1 entities at receiving side and one XML_CONS_REJ ERDS evidence for one entity at receiving side	N-1 parties at receiving side accept consignment. One party rejects it	
20			Sends it back to SERDS	Generates <erds_receipt_4>_with_EV_S ET#1</erds_receipt_4>		
21		Receives <erds_receipt_3>_with_X ML_NOT_F_ACC and stores ERDS evidence</erds_receipt_3>		Sends it back to IERDS		
22			Receives <erds_receipt_4>_with_EV_ SET#1</erds_receipt_4>	Generates hndvMet, user content} structure		

		Scer	nario id: EXT_RERDS_NOT_F_	ACC #2		Purpose
#	Sender	SERDS	IERDS	RERDS	Receiving side	
23			Sends it back to SERDS	Consigns hndvMet, user content} structure to N-1 entities at receiving side		
24		Receives <erds_receipt_4>_ with_EV_SET#1 and stores ERDS evidences</erds_receipt_4>	Relies <erd_dispatch_1>_with_XM L_SUB_ACC to RERDS to RERDS</erd_dispatch_1>	Generates and stores XML_CONT_CONS ERDS evidence for N-1 entities	N-1 hndvMet, user content} structure consignments succeed	
25		Sends it back to sender		Generates <erds_receipt_5>_with_ XML_CONT_CONS</erds_receipt_5>		
26	Receives <erds_receipt_4>_ with_EV_SET#1</erds_receipt_4>			Relies it to IERDS		
27			Receives <erds_receipt_5>_with_ XML_CONT_CONS</erds_receipt_5>			
28			Relies it to SERDS			
29		Receives <erds_receipt_5>_with_ XML_CONT_CONS and stores ERDS evidence</erds_receipt_5>				

## Table 8b: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (3/3)

		Scenario id: SERD	S_SF_IERDS_SN	I_RERDS_BB	_NO_NOT_F_ACC#3		Purpose
#	Sender	SERDS	IERDS	S	RERDS	Receiving side	
Para	meter: <erd_dispatch_1< td=""><td>&gt;_with_XML_SUB_ACC</td><td></td><td>Var EV_SET#</td><td><math>1 = \{XML\_CONS\_ACC, XML\_CC\}</math></td><td>NS_REJ}</td><td>As first scenario but</td></erd_dispatch_1<>	>_with_XML_SUB_ACC		Var EV_SET#	$1 = \{XML\_CONS\_ACC, XML\_CC\}$	NS_REJ}	As first scenario but
Para	rameter: <erds_receipt_1>_with_XML_REL_ACC</erds_receipt_1>						
Para	arameter: <erds_notification_1>_forAcceptance</erds_notification_1>						
Para	meter: <erds_receipt_2< td=""><td>&gt;_with_XML_ REL_ACC</td><td></td><td></td><td></td><td></td><td>rejects consignment</td></erds_receipt_2<>	>_with_XML_ REL_ACC					rejects consignment
Para	Parameter: <erds 3="" receipt=""> with XML NOT F ACC and another one entity</erds>						
Para	meter: <erds_receipt_4< td=""><td>&gt;_with_ EV_SET#1</td><td></td><td></td><td></td><td></td><td>fails in retrieving the</td></erds_receipt_4<>	>_with_ EV_SET#1					fails in retrieving the
Para	meter: <erds_receipt_5< td=""><td>&gt;_with_ XML_CONT_CONS</td><td></td><td></td><td></td><td></td><td>user content.</td></erds_receipt_5<>	>_with_ XML_CONT_CONS					user content.
			Sequence of	of actions			
#	Sender	SERDS	IERDS	S	RERDS	Receiving side	
1	Sender sends original						
ļ !	message						
2		Accepts submission and generates XML SUB ACC					
		ERDS evidence					

	Scenario id: SERDS_SF_IERDS_SN_RERDS_BB_NO_NOT_F_ACC#3						
#	Sender	SERDS	IERDS	RERDS	Receiving side		
3		Generates					
		<erd_dispatch_1>_with_X</erd_dispatch_1>					
		ML_SUB_ACC					
ļ.		Relies it to IERDS					
			Receives				
			<erd_dispatch_1>_with_XM</erd_dispatch_1>				
			L_SUB_ACC and stores it				
3			Generates XML_REL_ACC				
, <u> </u>			ERDS evidence				
'			Generates				
			<erds_receipt_1>_with_XM</erds_receipt_1>				
			L_SUB_ACC				
			Sends it back to SERDS				
)		Receives	Generates				
		<erds_receipt_1>_with_X</erds_receipt_1>	<erds_notification>_forAcce</erds_notification>				
		ML_SUB_ACC	ptance				
0		002_,100	Relies				
0			<erds_notification_1>_forAc</erds_notification_1>				
			ceptance to RERDS				
4			ceptance to KEKDS	Receives			
1				<pre><erds_notification_1>_forAcc</erds_notification_1></pre>			
_				eptance			
2				Generates XML_REL_ACC			
				ERDS evidence			
3				Generates			
				<erds_receipt_2>_with_XML_</erds_receipt_2>			
				REL_ACC			
14				Sends			
				<erds_receipt_2>_with_XML_</erds_receipt_2>			
				REL_ACC back to IERDS			
15			Receives	Sends			
			<erds_receipt_2>_with_XM</erds_receipt_2>	<erds_notification_1>_forAcc</erds_notification_1>			
			L_REL_ACC	eptance			
16					All the parties at receiving		
. ]				Generates XML_NOT_F_ACC	side receive		
				ERDS evidence	<erds_notification_1>_for</erds_notification_1>		
				2.120 041001100	Acceptance		
7			-	Generates	/ todeptarioe		
'				<pre><erds_receipt_3>_with_XML_</erds_receipt_3></pre>			
					-		
				NOT_F_ACC			
8				Sends it back to IERDS			
19			Receives				
			<erds_receipt_3>_with_XM L_NOT_F_ACC</erds_receipt_3>				

	Scenario id: SERDS_SF_IERDS_SN_RERDS_BB_NO_NOT_F_ACC#3 Purpose				Purpose	
#	Sender	SERDS	IERDS	RERDS	Receiving side	
20			Sends it back to SERDS		N-1 parties at receiving side access IERDS and accept consignment. One party rejects it	
21		Receives <erds_receipt_3>_with_X ML_NOT_F_ACC</erds_receipt_3>	Generates XML_CONS_ACC ERDS evidence for N-1 entities at receiving side and one XML_CONS_REJ ERDS evidence for one entity at receiving side			
22			Generates <erds_receipt_4>_with_EV_ SET#1</erds_receipt_4>			
23 24		Receives <erds_receipt_4>_ with_EV_SET#1</erds_receipt_4>	Relies <erd_dispatch_1>_with_XM L_SUB_ACC to RERDS to RERDS</erd_dispatch_1>			
25		Sends it back to sender		Accepts <erd_dispatch_1>_with_XML_ SUB_ACC</erd_dispatch_1>		
26	Receives <erds_receipt_4>_ with_EV_SET#1</erds_receipt_4>			Generates XML_REL_ACC ERDS evidence		
27				Generates <erds_receipt_5>_with_XML_ REL_ACC</erds_receipt_5>		
28				Sends it back to IERDS		
29				Generates <erd_dispatch_2>_with_XML_ SUB_ACC</erd_dispatch_2>		
30			Receives <erds_receipt_5>_with_XM L_REL_ACC</erds_receipt_5>	Consigns <erd_dispatch_2>_with_XML_ SUB_ACC to N-1 entities at receiving side</erd_dispatch_2>		
31					<erd_dispatch_2>_with_X ML_REL_ACC is consigned to N-1 entities at receiving side</erd_dispatch_2>	
32				Generates XML_CONT_CONS ERDS evidence for N-1 entities		
33				Generates <erds_receipt_6>_with_XML_ CONT_CONS</erds_receipt_6>		

Scenario id: SERDS_SF_IERDS_SN_RERDS_BB_NO_NOT_F_ACC#3				Purpose		
#	Sender	SERDS	IERDS	RERDS	Receiving side	
34					N-2 entities in receiving side retrieve user content, one entity fails	
35			Receives <erds_receipt_6>_with_XM L_CONT_CONS</erds_receipt_6>	Generates XML_CONT _HAND evidence for N-2 entities and XML_CONT_HAND_FAIL for one entity		
36			sends it back to SERDS	Generates <erds_receipt_7>_with_EV_S ET#2</erds_receipt_7>		
37		Receives <erds_receipt_7>_with_X ML_CONT_CONS</erds_receipt_7>		Sends it back to IERDS		
38		Sends it back to sender	Receives <erds_receipt_7>_ with_EV_SET#2</erds_receipt_7>			
39	Receives <erds_receipt_7>_wi th_XML_CONT_CON S</erds_receipt_7>		sends it back to SERDS			
40		Receives <erds_receipt_7>_ with_EV_SET#2</erds_receipt_7>				
41		Sends it back to sender				
42	Receives <erds_receipt_7>_ with_EV_SET#2</erds_receipt_7>					

# 6 ERD Messages instances

## 6.1 Introduction and technical approach

The present clause defines a number of instances of the different types of ERD Messages, namely: ERDS notification, ERD payload, ERDS receipt, and ERD dispatch, as defined in ETSI EN 319 522-1 [1]. These instances are used in clause 8 for defining different test cases.

The set of ERD messages instances is built following the technical approach shown below:

- The set includes instances of each type of ERD message.
- For each type of ERD message there will be at least one instance including all the optional components of relay metadata components defined in ETSI EN 319 522-3 [3].
- The present document first defines different combinations for the metadata components profiled in ETSI EN 319 522-4-1 [4] and in ETSI EN 319 522-4-2 [5], and for the relay metadata components defined in ETSI EN 319 522-2 [2].
- Each instance of a certain type of ERD message is defined as a composition of different metadata components combinations already mentioned.

The rest of the present clause is organized as follows:

- Clause 6.2 presents a number of combinations of metadata components for the different headers that can be present in the different ERD message types. These combinations are specified separately as they are used in the definition of instances of different ERD message types.
- Clause 6.3 defines instances of ERD payloads.
- Clause 6.4 defines instances of ERDS receipts.
- Clause 6.5 defines instances of ERD dispatches.

As mentioned in clause 4.2 new combinations of metadata components may be defined for each header, and new instances of ERD messages may be added to the current set, for defining new test cases.

# 6.2 Combinations of fields for headers in ERD envelopes

### 6.2.1 Introduction

Clause 6.2.2 defines combinations of values for the metadata components defined in OASIS Standards [i.2] and [8] and further profiled in ETSI EN 319 522-4-1 [4].

Clause 6.2.3 defines combinations of values for the relay metadata components defined in ETSI EN 319 522-3 [3].

Clause 6.2.4 defines aggregations of combinations defined in clauses 6.2.2 and 6.2.3.

# 6.2.2 Combinations of AS4 metadata profiled in ETSI EN 319 522-4

The present clause defines different combinations of values for the metadata components defined in OASIS Standards [i.2] and [8] and further profiled in ETSI EN 319 522-4-1 [4].

These combinations are shown in Table 9.

Cells in column "Component name" contain the name of the metadata components.

Cells in column "Value" shows the value to be assigned to the metadata component. These cells may contain the following values:

- As specified in ETSI EN 319 522-4-1 [4]. This value is reserved for cases where the ETSI EN 319 522-4-1 [4] specifies a mandatory value for the metadata component.
- As recommended in ETSI EN 319 522-4-1 [4]. This value is reserved for cases where the ETSI EN 319 522 4-1 [4] recommends one certain value for the metadata component (usually using the modal verb should). In these cases, the metadata component has this recommended value in the combination.
- TEST\_DEPENDANT means that the value of the metadata component will be defined in test cases for testing ERD message formats.

Cells in "Notes/Additional requirements" contain one or more letters or/and one or more integer numbers. The letters correspond to additional requirements that are given after the table. The numbers correspond to numbers of notes that appear after the aforementioned additional requirements.

Table 9: Combinations of metadata values as defined in OASIS specifications

Combination identifier	Component / Feature name	Value	Notes/Additional requirements
	PMode.Initiator	As specified in ETSI EN 319 522-4-1 [4]	
	PMode.Responder	As specified in ETSI EN 319 522-4-1 [4]	
	PMode.Initiator.Role	As specified in ETSI EN 319 522-4-1 [4]	
	PMode.Responder.Role	As specified in ETSI EN 319 522-4-1 [4]	
	PMode[1].BusinesssInfo.Service	As specified in ETSI EN 319 522-4-1 [4]	
AS4Met#1	PMode[1].Security.SendReceipt	As specified in ETSI EN 319 522-4-1 [4]	1
	PMode[1].Security.SendReceipt.NonRepudiation	As specified in ETSI EN 319 522-4-1 [4]	1
	PMode[1].Security.SendReceipt.ReplyPattern	As specified in ETSI EN 319 522-4-1 [4]	2
	PMode[1].ErrorHandling.Report.AsResponse	As specified in ETSI EN 319 522-4-1 [4]	2
	AS4 Reception Awareness Feature	As recommended in ETSI EN 319 522-4-1 [4]	
	Duplication elimination function	As recommended in ETSI EN 319 522-4-1 [4]	

#### 6.2.3 Combinations of components of relay metadata

Table 10 shows combinations of values of the relay metadata defined in ETSI EN 319 522-3 [3].

For the purpose of defining the test cases, the field "ApplicablePolicy" shall always consist in a single URI. New combinations may be added where this field consists in a sequence of URIs.

Cells in "Component name" column contain the name of relay metadata component in the ERD message. The names used are the ones defined in ETSI EN 319 522-3 [3].

Cells in "Component value" column contain either:

- the value of the relay metadata component whose name is the one indicated in the previous column; or 1)
- 2) AS\_PER\_TESTCASE, meaning that for this test case the ERDS is free to give to the aforementioned field the value it considers worth.

Cells in "Notes/Additional requirements" contain one or more letters or/and one or more integer numbers. The letters correspond to additional requirements that are given after the table. The numbers correspond to numbers of notes that appear after the aforementioned additional requirements.

Cells in "Purpose" contain a description of the purpose of the combination defined in the row.

Table 10 defines parameterized combinations for new headers defined in ETSI EN 319 532-3 [7]. The combinations have as parameters the assurance levels and the consignment mode.

Table 10: Parameterized combinations for relay metadata components

Combination identifier	Header field name	Header field value	Notes/Additional requirements	Purpose	
RelayMet#1	RelayTime	AS_PER_TESTCASE	a,	Use in tests where: the contents of the components are	
	ExpirationTime	AS_PER_TESTCASE	b	correct; there is no indication neither of assurance	
	ScheduledDeliveryTime	AS_PER_TESTCASE	С	levels nor of mode of consignment.	
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#2	RelayTime	AS_PER_TESTCASE	а	Use in tests where: the contents of the components are	
	ExpirationTime	AS_PER_TESTCASE	b	correct; the assurance levels required is one	
	RequiredAssuranceLevel	Parameter		parameter, and there is no indication of mode of	
	ScheduledDeliveryTime	AS_PER_TESTCASE	С	consignment.	
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#3	RelayTime	AS_PER_TESTCASE	а	Use in tests where: the contents of the components are	
	ExpirationTime	AS_PER_TESTCASE	b	correct; relay metadata include all the optional	
	RequiredAssuranceLevel	Parameter		components present; and the assurance levels and the	
	RequestedConsignmentMode	Parameter		mode of consignment are parameters.	
	ScheduledDeliveryTime	AS_PER_TESTCASE	С		
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#4	RelayTime	AS_PER_TESTCASE	a,	Use in negative tests where: the scheduled delivery is	
	ExpirationTime	AS_PER_TESTCASE	b	after the expiration date; assurance levels indication is	
	REM-RecipientAssuranceLeve	Parameter		one parameter, and there is no indication of mode of	
	ScheduledDeliveryTime	AS_PER_TESTCASE	d	consignment.	
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#5	RelayTime	AS_PER_TESTCASE	а	Use in negative tests where: the scheduled delivery is	
•	ExpirationTime	AS_PER_TESTCASE	b	after the expiration date; and the assurance levels and	
	RequiredAssuranceLevel	Parameter		the mode of consignment are parameters.	
	RequestedConsignmentMode	Parameter			
	ScheduledDeliveryTime	AS_PER_TESTCASE	d		
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#6	RelayTime	AS_PER_TESTCASE	а	As RelayMet#1 but without ScheduledDeliveryTime	
•	ExpirationTime	AS_PER_TESTCASE			
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#7	RelayTime	AS_PER_TESTCASE	а	As RelayMet#2 but without ScheduledDeliveryTime.	
	ExpirationTime	AS_PER_TESTCASE			
	RequiredAssuranceLevel	Parameter			
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#8	RelayTime	AS_PER_TESTCASE	а	As RelayMet#3 but without ScheduledDeliveryTime.	
-	ExpirationTime	AS_PER_TESTCASE		7	
	RequiredAssuranceLevel	Parameter		7	
	RequestedConsignmentMode	Parameter		7	
	ApplicablePolicy	AS_PER_TESTCASE			

#### Additional requirements:

- a) The date and time indicated in this component shall be earlier than the date and time indicated in "ERD-ExpirationDate" and "ERD-ScheduledDelivery" (if present).
- b) The date and time indicated in this component shall be later than the date and time indicated in "ERD-RelayDate" and "ERD-ScheduledDelivery" (if present).
- c) The date and time indicated in this component shall be earlier than the date and time indicated in "ERD-ExpirationDate" and "ERD-ScheduledDelivery" and later than the date and time indicated in "ERD-RelayDate.
- d) The date and time indicated in this component shall be later than the date and time indicated in "ERD-RelayDate" and later than "ERD-ExpirationDate" (if present).

## 6.2.4 Combinations of AS4 metadata profiled and relay metadata

The present clause defines aggregations of AS4 metadata components defined in OASIS Standards [i.2] and [8] and further profiled in ETSI EN 319 522-4-1 [4], and relay metadata. These combinations are shown in Table 11.

Each metadata combination instance is defined as the aggregation of the combination defined in Table 9 and one of the combinations defined in Table 10.

Each combination defined in Table 10 has 4 parameters, namely:

- 1) An integer identifying the AS4 metadata combination in table of clause 6.2.2.
- 2) An integer identifying the relay metadata combination in table of clause 6.2.3.
- 3) The value of the required assurance levels (nil if this information is not present in the combination).
- 4) The value for consignment mode (nil if this information is not present).

All the metadata combinations instances can be obtained from the following expression:

#### Metadata(AS4-metadata#I, Relay-Metadata#J, AssuranceLevelComb, ConsignmentMode) where:

- I, for the combinations specified in the present document is always 1.
- J is one of {1 to 8}.
- AssuranceLevelCombs is one of {nil, low/low, subs/subs, high/high}.
- ConsignmentMode is one of {nil, basic, consented, consentedSigned}.

Table 11 defines a number of possible combinations of these parameters when the required assurance levels are the same for sender and entities at receiving side. It also defines combinations where the consignment mode field is not present. Finally, it also defines two illegal combinations, that can be used in negative test cases.

Table 11: ERD messagesmetada combinations

For test cases without ShceduledDelivery			
OutermostHeade(1,6,nil,nil)	AS4Met#1 + RelayMet#6	No indications neither of assurance levels nor consignment mode	
Metadata(1,7,low/low,nil) Metadata(1,7,sub/subs,nil) Metadata(1,7,high/high,nil) Metadata (1,8,low/low,basic) Metadata(1,8,subs/subs,basic)	AS4Met#1 + RelayMet#7(low/low) AS4Met#1 + RelayMet#7(substantial/substantial) AS4Met#1 + RelayMet#7(high/high) AS4Met#1 + RelayMet#8(low/low, basic) AS4Met#1 + RelayMet#8(substantial/substantial, basic) AS4Met#1 + RelayMet#8(high/high, basic)	No consignment mode indicated and assurance level Basic consignment mode and assurance level	
Metadata (1,8,high/high,basic) Metadata (1,8,low/low,consented) Metadata(1,8,subs/subs,consented) Metadata (1,8,high/high,consented)	AS4Met#1 + RelayMet#8(low/low, consented)  AS4Met#1 + RelayMet#8(substantial/substantial, consented)  AS4Met#1 + RelayMet#8(high/high, consented)	Consented consignment mode and assurance level	
Metadata (1,8,low/low,consentedSigned)	AS4Met#1 + RelayMet#8(low/low, consentedSigned)	Consented and signed consignment mode and	
Metadata(1,8,subs/subs,consentedSigned)	AS4Met#1 + RelayMet#8(substantial/substantial, consentedSigned)	assurance level	
Metadata (1,8,high/high,consentedSigned)	AS4Met#1 + RelayMet#8(high/high, consentedSigned)		
	or test cases with ShceduledDelivery		
OutermostHeade(1,1,nil,nil)	AS4Met#1 + RelayMet#1	No indications neither of assurance levels nor consignment mode	
Metadata(1,2,low/low,nil) Metadata(1,2,sub/subs,nil) Metadata(1,2,high/high,nil)	AS4Met#1 + RelayMet#2(low/low) AS4Met#1 + RelayMet#7(substantial/substantial) AS4Met#1 + RelayMet#2(high/high)	No consignment mode indicated and assurance level	
Metadata (1,3,low/low,basic)	AS4Met#1 + RelayMet#3(low/low, basic)		

For test cases without ShceduledDelivery			
Metadata(1,3,subs/subs,basic)	AS4Met#1 + RelayMet#8(substantial/substantial,	Basic consignment mode	
·	basic)	and assurance level	
Metadata (1,3,high/high,basic)	AS4Met#1 + RelayMet#3(high/high, basic)		
Metadata (1,3,low/low,consented)	AS4Met#1 + RelayMet#3(low/low, consented)	Consented consignment	
Metadata(1,3,subs/subs,consented)	AS4Met#1 + RelayMet#3(substantial/substantial,	mode and assurance level	
	consented)		
Metadata (1,3,high/high,consented)	AS4Met#1 + RelayMet#3(high/high, consented)		
Metadata (1,3,low/low,consentedSigned)	AS4Met#1 + RelayMet#3(low/low,	Consented and signed	
	consentedSigned)	consignment mode and	
Metadata(1,3,subs/subs,consentedSigned)	AS4Met#1 + RelayMet#3(substantial/substantial,	assurance level	
	consentedSigned)		
Metadata (1,3,high/high,consentedSigned)	AS4Met#1 + RelayMet#3(high/high,		
	consentedSigned)		
Metadata (1,4,low/low,nil)	AS4Met#1 + RelayMet#4(low/low)	Combinations for negative	
Metadata (1,5,low/low,basic)	AS4Met#1 + RelayMet#5(low/low, basic)	test cases (cause of	
		submission rejection for	
		instance)	

# 6.3 Instances of ERD payload

As an ERD payload is the aggregation of metadata and user content, the present document defines as many ERD payload instances as metadata instances have been defined in clause 6.2.4.

# 6.4 Instances of ERDS receipts

An ERD receipt is the aggregation of one metadata payload and one or more ERDS evidence payloads. The number and contents of these payloads will depend on the specific test case.

The present document uses the following convention for identifying sets of ERDS receipts:

ERD\_ReceiptInst (Metadata(AS4Met#1,RelayMet#J,nil,nil), userContent,<EVID#M>+)

#### Where:

• J is a positive integer between 1 and 9 (inclusive), and the + symbol in <EVID#M> indicates that in each case, the ERDS receipt instance shall contain as many payloads for ERDS evidence as ERDS evidences indicated in the test case where the ERDS receipt is used.

This unique content identifies all the possible ERDS receipts that will be needed in the definitions of the test cases.

# 6.5 Instances of ERD dispatch

The present document uses the following convention for identifying sets of ERD dispatches:

 ${\tt ERDS\_DispatchInst(Metadata(AS4Met\#1,RelayMet\#J, AssLevelComb, ConsignmentModeId), < EVID\#M>+)} \\$ 

#### Where:

• J is a positive integer between 1 and 9 (inclusive), AssLevelComb is one of {low/low, substantial/substantial, high/high}, ConsignmentModeId is one of {basic,consented,consentedSigned}, and the + symbol in <EVID#M> indicates that in each case, the ERD dispatch instance shall contain as many payloads for ERDS evidence as ERDS evidences indicated in the test case where the ERDS receipt is used.

This unique content identifies all the possible ERD dispatches that will be needed in the definitions of the test cases.

## 7 Other named sets

# 7.1 Named sets of participating ERDSs

The present clause defines named sets of participating REMSs that are used for naming test cases in clause 5.

The details on the participating ERDSs are given using the following pattern: ERDS(I,J), where:

- I stands for the number of Intermediate ERDSs (IERDSs); and
- J stands for the number of the Recipient's ERDSs (RERDSs).

# 7.2 Named sets of additional requirements

The present clause defines named sets of additional requirements that are used for building different test cases based on the same scenarios of REM messages.

Table 12 shows the named sets of additional requirements.

Table 12: Named sets of additional requirements

Name of the set	Additional requirements in the set
Additional Daggett	Original message: not signed, no attachment
AdditionalReqs#1	Sender sends original message
	Original message: not signed, no attachment
AdditionalReqs#2	Sender's delegate
	Sender's delegate sends original message

# 7.3 Named sets of entities in receiving side

The present clause defines named sets of entities that are present at receiving side. This allows using one scenario in defining different test cases by changing the entities in the receiving side.

EXAMPLE: Scenarios defined for one recipient could be used in test cases where the scenarios involve only one delegate of one recipient.

In order to define the names of the sets, the pattern RecSide(I,J,K) is used where:

- I stands for the number of recipients.
- J stands for the number of recipient's delegates.
- K stands for the number of recipients each delegate is delegate of.

K shall always be less or equal than I. If I is not 0 then K shall also be different from 0.

# 8 Test cases definition

## 8.1 Introduction

#### 8.1.1 General

The notations shown in clauses 4.1, 6 and 7, allow building a compact notation for defining tests cases.

The present document defines sets of test cases. Each set of test cases is expressed as a function of a number of parameters (some of them are integers, other are tuples of several values, other -mainly reasons for failures- are enumerated values specified in another ETSI deliverable).

Under these conditions one specific test case is obtained when the set is particularized by assigning a single value to each parameter.

#### 8.1.2 Notation for black box model scenarios

No ERD messages are generated in the scenarios defined for the black box model. Consequently, the only parameters the test cases depend on are the additional requirements specified in clause 7.2 and the entities present at receiving side, identified in clause 7.3.

Below follows the notation used for identifying the set of test cases for scenario ERDS\_BB\_NOT\_F\_ACC#1:

```
ERDS_BB_NOT_F_ACC#1 (
    RecSide(F,G,H),
    AssLevelComb,ConsignmentModelId,
    AdditionalReqs#P
)
```

This part shows the components required for defining the test cases for this scenario. For this scenario each test case in the set will require providing details of:

- The entities in the receiving side (RecSide). The notation for identifying one of the different alternatives is as specified in clause 7.3.
- The additional requirements, whose notation has been specified in clause 7.2.

Scenarios where there is some failure also include the FailureReason parameter.

Below follows the second part of the definition of the set of test cases:

#### Where:

- For RecSide: (F,G,H) is one of {(1,0,0), (3,0,0), (3,1,2),(3,1,3)}: define test cases for one recipient, several recipients, and several recipients and one recipient's delegate.
- For AdditionalReqs: P is one of {1,2}.

#### 8.1.3 Notation for 4 corner and extended models scenarios

For helping in understanding the notation, below follows the definition of the set of test cases for the scenario 4C\_RERDS\_NO\_NOT\_F\_ACC #2. The definition of a set of test cases has two parts.

Below follows the first one:

```
4C_RERDS_NO_NOT_F_ACC #2 (
    RecSide(F,G,H),
    ERD_dispatchInst_with_XML_SUB_ACC
          (Metadata(AS4Met#1, RelayMet#I, AssLevelComb, ConsignmentModeId), UserContent,XML_SUB_ACC),
          ERDS_receipt_with_XML_REL_ACC
          (Metadata(AS4Met#1, RelayMet#I, AssLevelComb, ConsignmentModeId),XML_REL_ACC),
AdditionalReqs#P
)
```

This part shows the components required for defining the test cases for this scenario. For this scenario each test case in the set will require providing details of:

- The entities in the receiving side (RecSide). The notation for identifying one of the different alternatives is as specified in clause 7.3.
- The ERDS Dispatch instance, which also carries an ERDS Evidence (XML\_SUB\_ACC). The notation for completely defining one specific instance among all the possibilities offered by the different parameters, is as specified in clause 6.5.

- The ERDS Receipt, carrying a XML\_REL\_ACC ERDS evidence. The notation for completely defining one specific instance among all the possibilities offered by the different parameters, is as specified in clause 6.4.
- The additional requirements, whose notation has been specified in clause 7.2.

Scenarios where there is some failure also include the FailureReasonId parameter.

Some scenarios include REM payloads instead of REM dispatches. The details of the components of a REM payload are provided as the details of components of a REM dispatch.

Below follows the second part of the definition of the set of test cases:

#### Where:

- For RecSide: (F,G,H) is one of {(1,0,0), (3,0,0), (3,1,2),(3,1,3)}: define test cases for one recipient, several recipients, and several recipients and one recipient's delegate.
- For ERDS\_dispatchInst\_with\_XML\_SUB\_ACC and REMS\_receipt\_with\_XML\_REL\_ACC:
  - I is one of {1..8}, AssLevelComb is one of {nil, low/low, substantial/substantial, high/high}, and ConsignmentModeId is one of {nil, basic, consented, consentedSigned.
- For Additional Reqs: P is one of {1,2}.

This part shows the different values that the parameters present in the first part, can have.

Each legal combination of all the parameters will collapse the set in ONE test case. For instance:

```
RecSide(1,0,0),
ERDS_dispatchInst_with_XML_SUB_ACC
      (OuterMostHeader(AS4#1, RelayMet#8, high/high, consentedSigned), UserContent,XML_SUB_ACC)),
ERDS_receipt_with_XML_REL_ACC
      (OuterMostHeader(RFCFields#1, NewFields#6, nil, nil), XML_REL_ACC),
AdditionalReqs#1
```

Defines ONE test case in the set, where:

- The REM-RelayDate and REM-ScheduledDelivery header fields are absent in the outermost header of the REM Dispatch and the REMS Receipt (NewFields#11 combination of new header fields).
- The assurance level combination indication is present and its value is high/high.
- The consignment mode indication is present and its value is consentedSigned.
- There is only one recipient at receiving side.

In addition, the following set of rules govern the selection of coherent triplets (RelayMet#I, AssLevelComb, ConsignmentModeId):

- 1) Absence of Assurance levels indication is indicated by a nil value.
- 2) Absence of Consignment mode indication is indicated by a nil value.
- 3) For I one of {1, 6} both AssLevelComb and ConsignmentModeId are nil.
- 4) For I one of {2, 4, 7} AssLevelComb is not nil, and ConsignmentModeId is nil.
- 5) For I one of {3, 5, 8} neither AssLevelComb nor ConsignmentModeId are nil.

Any combination (RelayMet#I, AssLevelComb, ConsignmentModeId)in a specific test case has to meet the rules 3 to 5.

## 8.1.4 Reasons for Failures

The present clause shows the different values that may have the reason for failure parameters in those scenarios where failures occur:

- For XML\_SUB\_REJ, the FailureReasonId is one of {RA02, RA03, RA04, RA05} as specified in clause 8.3.3.1 of ETSI EN 319 522-2 [2].
- For XML\_REL\_REJ, the FailureReasonId is one of {RB02, RB03, RB04, RB05, RB06, RB07, RB08, RB09, RB010} as specified in clause 8.3.3.2 of ETSI EN 319 522-2 [2].
- For XML\_CONS\_FAIL, and XML\_CONS\_NOT\_FAIL, the FailureReasonId is one of {RD03, RD04, RD05, RD06} as specified in clause 8.3.3.4 of ETSI EN 319 522-2 [2].
- For XML\_CONT\_HAND\_FAIL, the FailureReasonId is one of {RE03, RE04} as specified in clause 8.3.3.5 of ETSI EN 319 522-2 [2].
- For XML\_CONS\_REJ, the FailureReasonId is RC08 as specified in clause 8.3.3.3 of ETSI EN 319 522-2 [2].
- For XML\_ACC\_REJ\_EXP, the FailureReasonId is RC09 as specified in clause 8.3.3.3 of ETSI EN 319 522-2 [2].

## 8.2 Test cases for black box model

For each scenario the present document defines the set of test cases defined by the following expression:

```
SCENARIOID (
   RecSide(F,G,H),
   AssLevelComb,ConsignmentModelId,
   AdditionalReqs#P,
   FailureReasonId
)
```

#### Where:

- For RecSide: (F,G,H) is one of {(1,0,0), (3,0,0), (3,1,2),(3,1,3)}: define test cases for one recipient, several recipients, and several recipients and one recipient's delegate.
- For Additional Reqs: P is one of {1,2}.
- For FailureReasonId: see clause 8.1.4.

## 8.3 Test cases for 4-cornel and extended models

#### 8.3.1 General

The present clause, instead of showing one test case set expression for each scenario, which would make the present document very long, defines rules for building these expressions for each scenario in function of: the ERD dispatches, ERDs receipts, ERD payloads present in them, the entities at the receiving side, the failure reasons, and the additional requirements.

Some of the rules are defined for parametrizing the different ERD messages appearing in each scenario, other for parametrizing the receiving side, others for parametrizing the failure reasons, and others for parametrizing the additional requirements. The application of all these rules for one scenario results in generating test cases set expressions as the one shown in clause 8.1.3 which define a set of test cases for such an scenario.

# 8.3.2 Rules for parametrizing ERD dispatches

The ERD dispatches for these scenarios will be built on the following components:

```
(\texttt{Metadata}(\texttt{AS4Met} \# \texttt{1}, \texttt{RelayMet} \# \texttt{I}, \texttt{AssLevelComb}, \texttt{ConsignmentModeId}), \texttt{UserContent}, \texttt{<EVID>+}) \\
```

#### Where:

• i is one of {1,2,3,4,5,6,7,8}, AssLevelComb is one of {nil, low/low, substantial/substantial, high/high}, ConsignmentModeId is one of {nil, basic, consented, consentedSigned }, and <EVID>+ stands for the ERDS evidences present in the ERD dispatch as per the scenario.

## 8.3.3 Rules for parametrizing ERD payloads

The ERD payloads for these scenarios will be built on the following components:

(Metadata(AS4Met#1, RelayMet#I, AssLevelComb, ConsignmentModeId), UserContent

#### Where:

• i is one of {1,2,3,4,5,6,7,8}, AssLevelComb is one of {nil, low/low, substantial/substantial, high/high}, and ConsignmentModeId is one of {nil, basic, consented, consentedSigned}.

## 8.3.4 Rules for parametrizing ERDS receipts

The ERDS receipts for these scenarios will be built on the following components:

(Metadata(AS4Met#1, RelayMet#I, AssLevelComb, nil), <EVID>+)

#### Where:

• i is one of {1,2,3,4,5,6,7,8}, AssLevelComb is one of {nil, low/low, substantial/substantial, high/high}, and <EVID>+ stands for the ERDS evidences present in the ERDS receipt as per the scenario.

## 8.3.5 Rules for parametrizing entities at receiving side

The number and type of entities present at the receiving side will be parametrized as indicated below:

RecSide(F,G,H),

Where, as specified in clause 7.3:

- F stands for the number of recipients.
- G stands for the number of recipient's delegates.
- H stands for the number of recipients each delegate is delegate of.

H shall always be less or equal than F. If F is not 0 then H shall also be different from 0.

# 8.3.6 Rules for parametrizing failure reasons

The FailureReasonId shall follow the requirements specified in clause 8.1.4.

# 8.3.7 Rules for parametrizing additional requirements

The additional requirements will be parametrized as indicated below:

AdditionalReqs#P

Where P is one of  $\{1,2\}$  with the meaning specified in clause 7.2.

# Annex A (informative): Bibliography

- ETSI EN 319 532-1: "Electronic Signatures and Infrastructures (ESI); Registered Electronic Mail (REM) Services; Part 1: Framework and Architecture".
- ETSI EN 319 532-2: "Electronic Signatures and Infrastructures (ESI); Registered Electronic Mail (REM) Services; Part 2: Semantic Contents".
- ETSI EN 319 532-4: "Electronic Signatures and Infrastructures (ESI); Registered Electronic Mail (REM) Services; Part 4: Interoperability profiles".

# History

Document history			
V1.1.1	February 2019	Publication	