surescripts

XDR

surescripts

CONTENTS

1. XDR Overview

Introduction

Surescripts Clinical Direct Messaging (CDM) enables the secure, electronic exchange of clinical information between health care organizations. Built on the Direct standard maintained by DirectTrust, CDM allows providers, hospitals, health plans, and other authorized participants to send and receive clinical documents - such as referrals, transitions of care, discharge summaries, and care coordination notes - within their existing workflows.

Surescripts is an accredited Health Information Service Provider (HISP), Direct Certificate Authority (CA), and Direct Registration Authority (RA) through DirectTrust. Together, these accredited roles allow Surescripts to issue and manage the trust credentials, validate participant identity, and operate the messaging infrastructure that makes secure Direct message exchange possible across the community of DirectTrust-accredited HISPs. CDM holds Certified Health IT status under Office of the National Coordinator (ONC) requirements and supports Promoting Interoperability program measures.

Cross-Enterprise Document Reliable Interchange (XDR) is one of the connectivity methods CDM supports. This guide focuses on the XDR-based messaging model as implemented by Surescripts.

About XDR

What it is. XDR is an IHE (Integrating the Healthcare Enterprise) integration profile that defines how clinical documents are securely exchanged between systems using web services and standardized metadata. It is one of the transport methods supported by CDM for exchanging clinical documents between participants, and it enables secure exchange even in the absence of a document-sharing infrastructure such as an XDS Registry and Repository.

How it works. XDR packages clinical content and its required metadata into a single message that is electronically routed between a sending system and a receiving system. Each message is delivered using the IHE Provide and Register Document Set-b transaction [ITI-41], carried over web services. For each message that is sent, a response message is returned, and the type of response defines the status of the delivery.

Why it matters. XDR allows participants to exchange clinical documents while preserving message integrity and traceability throughout delivery. Depending on how each participant is configured, XDR can also support Delivery Status Notifications that communicate the outcome of message processing.

Bottom line. XDR provides a secure, standards-based method to move clinical documents between health care systems, with visibility into message processing and delivery.

Scope, Limitations, and What Is Included

This guide describes base XDR-based messaging as implemented by Surescripts. It covers how customers send and receive XDR messages, how messages are addressed and routed, how notifications and tracking work, how clinical documents and metadata are packaged, the connectivity and certificate requirements, and the requirements a customer's application must meet to be certified on the Surescripts network.

As a CDM connectivity method, XDR-based messaging includes the Surescripts HISP, Direct CA, and Direct RA services. Message routing across the Surescripts network and electronic message delivery are included with the service.

XDR is well suited to point-to-point exchange of clinical documents (for example, Consolidated Clinical Document Architecture [C-CDA] documents and PDFs), transitions of care and discharge summaries, and public health reporting. Specific use cases built on XDR - such as electronic case reporting (eCR) and closed-loop referrals (360X) - are outside the scope of this version of the guide and may be addressed in a future revision.

Surescripts places specific limitations on XDR relative to the base IHE profile. Those limitations are detailed in **Differences from Standard IHE XDR**.

Trust Framework and Flow-Down Terms

XDR-based messaging operates within the DirectTrust trust framework. Certain trust-framework obligations flow down to every participant that sends or receives messages on the Surescripts network. These flow-down terms are an important part of participating in the network.

> **Note:** The full flow-down terms are provided in **DirectTrust Trust Framework and Flow-Down Terms**.

About This Guide

This guide provides an overview of the XDR specifications and describes the message structures, workflows, and examples required to support XDR-based document exchange. It is intended for participants responsible for designing, developing, or supporting system integrations that use XDR-based messaging.

This guide supports and further clarifies the applicable IHE and DirectTrust specifications as used on the Surescripts network; it does not reproduce those standards in full. Requirements beyond those in the referenced standards are called out in this guide and are enforceable. Customers should read and understand the associated standards, listed in the document references below, before implementing.

Document References

This guide should be read together with the following materials. External standards must be obtained by the customer from the issuing organization.

Surescripts-provided materials:

Reference	Description
Connectivity and Authentication Guide	Communication and security protocol requirements, mutual TLS and certificate details, and the IP addresses associated with Surescripts services.
Directory Implementation Guide	Direct address search, display, and publishing requirements (see **Application Certification Requirements**, ACRs C.200-C.202).
Network Operations Guide	Network operations, support, and the Escalation Matrix.

External materials to be obtained by the customer:

Reference	Where to obtain
DirectTrust *XDR and XDM for Direct Secure Messaging Specification*, Version 2.1	**https://directtrust.org/standards**
IHE ITI Technical Framework, Volume 1, Section 15 - XDR (Cross-Enterprise Document Reliable Interchange)	**https://profiles.ihe.net/ITI/TF/Volume1/ch-15.html**
IHE ITI Technical Framework, Volume 1, Section 16 - XDM (Cross-Enterprise Document Media Interchange)	**https://profiles.ihe.net/ITI/TF/Volume1/ch-16.html**
IHE ITI Technical Framework, Volume 2b, Section 3.41 - Provide and Register Document Set-b [ITI-41]	**https://profiles.ihe.net/ITI/TF/Volume2/**
IHE ITI Technical Framework, Volume 3 - Metadata	**https://profiles.ihe.net/ITI/TF/Volume3/**
W3C SOAP Version 1.2	**https://www.w3.org/TR/soap12/**
W3C Web Services Addressing 1.0 - Core	**https://www.w3.org/TR/ws-addr-core/**
W3C SOAP Message Transmission Optimization Mechanism (MTOM)	**https://www.w3.org/TR/soap12-mtom/**
WS-I Basic Profile 2.0	**http://ws-i.org/Profiles/BasicProfile-2.0-2010-11-09.html**
HL7 Consolidated CDA (C-CDA)	**https://www.hl7.org/ccdasearch/**
HL7 OID Registry	**https://www.hl7.org/oid/index.cfm**

Sources of Authority

Different aspects of an XDR implementation are governed by different documents. This guide clarifies how they apply on the Surescripts network but does not supersede them; where it states a requirement beyond the referenced standards, that requirement is called out and is enforceable.

Area	Source of Authority
Message structure and the ITI-41 transaction	IHE IT Infrastructure Technical Framework (**Document References**), as clarified in this guide.
Metadata requirements	IHE and DirectTrust XDR/XDM specifications (**Document References**); Surescripts does not enforce metadata beyond the standards.
Connectivity, certificates, endpoints, networking, and transport	Surescripts Connectivity and Authentication Guide.
Direct address directory (publishing, search, and use)	Surescripts Directory Implementation Guide and the DirectTrust Aggregated Directory Data Sharing Policy.
What Surescripts validates for production	Surescripts Application Certification Requirements and additional business rules applied during certification.
Trust framework and flow-down obligations	**DirectTrust Trust Framework and Flow-Down Terms**
Best-practice guidance (recommended, not required)	**Application Certification Requirements**

2. Integration and Production

The following section defines technical and support elements needed to achieve certification to move XDR-based messaging into production, along with our Integration, Compliance, and Certification processes.

Integration Process

When a customer begins integrating a Surescripts product into their application(s), they will be provided access to all the Surescripts documentation pertaining to the product being implemented. In addition, customers will be provided access to the staging environment to design, develop and test the product integration within their application.

> **Note:** The time frame of the project can vary depending on the customer's resource allocation for the project.

Meeting Requirements

During Integration, customers will ensure their system has met all product requirements. The Certification Testing will focus on message format, application workflow and display in accordance with Surescripts documentation and the associated Application Certification Requirements (ACRs).

For requirements, consider the following:

- Surescripts ACRs are required to be met to achieve production status on the Surescripts network and will be enforced as part of certification.
- To ensure high-quality transactions, Surescripts applies additional business rules above and beyond the NCPDP schema defined requirements that will cause a message to be successful or rejected.
- Surescripts test cases do not cover all possible scenarios in production. Customers are responsible for testing any other applicable scenarios specific to their production environment.
- In accordance with the customer’s legal agreement with Surescripts, each customer is responsible for ensuring compliance with all applicable laws including, but not limited to, local and state laws/regulations where doing business.

Upon successful completion of certification and, when applicable, other pre-production network requirements (e.g., Identity Proofing, Attestations, DEA audit), customers will be enabled in production.

Transition to Production

Once message validation testing is complete and the contract is approved by the Surescripts Certification Review Board, the customer is ready to move into production. Surescripts will configure the production connection and ensure successful operations with the customer. Prior to transition into production, Surescripts Account Management will work with the customer and internal Surescripts teams to review the following items:

- **Production support contacts**

  To be captured in an Escalation Matrix—a tool that specifies how Surescripts should contact customers for support needs, including case routing and escalation paths.

- **Support process and training**
- **Support hours**

> **Note:** The Escalation Matrix for use by Surescripts customers can be found in the *Surescripts Network Operations Guide*.

Surescripts Terminology Usage

The following table outlines terminology usage for this guide.

Term	Term Usage
must	Requirements that are enforced as part of the production code or by Surescripts business rules. Some business rules reside in the Element Details section of this guide.
shall	The requirements customers are required to meet in order to be certified on the Surescripts network. These requirements will be enforced as part of certification, not through business rules.
should	Used for guidance and best practices to produce superior results and enhance electronic messaging. Best practices can also be found in the Best Practice sections. Customers are encouraged, but not required, to meet best practices in order to be certified on the Surescripts network.
C.####	Designates a Surescripts Clinical Direct Messaging ACR.

Communication Rules

Please refer to the Connectivity and Authentication Guide for details regarding communication and security protocol requirements as well as references to all links and IP addresses associated with Surescripts services. For the network to be reliable, there are communication rules to which all customers must adhere.

Connectivity and Mutual TLS (mTLS)

Surescripts requires Mutual Transport Layer Security (mTLS) authentication for XDR. Mutual authentication means both parties in the connection authenticate each other: the client authenticates the server, and the server authenticates the client. mTLS is standard across Surescripts products and is a common source of setup issues, so the direction-specific roles are described below. The complete certificate and connectivity requirements are maintained in the Connectivity and Authentication Guide, which is the authoritative source and must be followed for implementation.

For XDR, mTLS applies in both directions of exchange, and the customer's role differs by direction:

When the customer sends to Surescripts (customer-initiated), the customer acts as the client and presents a Surescripts-authorized client certificate. Surescripts signs this certificate from a Certificate Signing Request (CSR) the customer submits. See the Connectivity and Authentication Guide, Authentication section, "Customer Client-Side Certificates," for the CSR and client-certificate requirements.

When the customer receives from Surescripts (Surescripts-initiated), the customer acts as the server and presents a server certificate that Surescripts validates. See the Connectivity and Authentication Guide, Authentication section, "Customer Server-Side Certificates," for the server-certificate requirements.

Because XDR connections authenticate with mTLS, they do not require IP allow-listing.

> **Note:** The TLS protocol version, cipher requirements, certificate specifications (including key length, subject fields, and acceptable certificate authorities), certificate renewal, and IP address handling are all governed by the Connectivity and Authentication Guide. Refer to that guide rather than assuming any value.

Message Size

The maximum size for a single XDR message, including all attachments, is 20 MB.

A message larger than 20 MB is rejected at the transport channel before any XDR processing occurs, so the sender receives a transport-level failure rather than an XDR application-level error (see **XDR-Based Messaging Errors**).

20 MB is also the practical ceiling for broad interoperability, because not all systems in the exchange ecosystem accept messages at or near that size. Where a recipient's capacity is unknown, design for smaller messages.

Timeouts

For timeouts, consider the following:

When sending a message to Surescripts, the initiator should set the HyperText Transfer Protocol (HTTP) timeout to no less than 30 seconds.

A receiving system must return a valid synchronous response - a Provide and Register Document Set-b Response indicating success, or an error response - within 24 seconds. Setting the sender's timeout to at least 30 seconds allows the receiving system adequate time to generate its response within the required 24 seconds.

When a message is accepted for delivery but its final delivery status is not yet known, a Delivery Status Notification (DSN) is expected to follow. If a follow-up DSN is not received within 60 minutes, the message times out and Surescripts generates a DSN Failure to the sender.

> **Note:** See **Message Notifications and Tracking** for how synchronous responses and DSNs work together.

UTC Time Format

By using Coordinated Universal Time (UTC), the receiver of a message will know the time regardless of their time zone. For example, if a message was sent from Boston at 5:30 PM EDT (Eastern Daylight Time), the time would be sent as 21:30 UTC. If this message was received in Chicago CDT (Central Daylight Time), the 21:30 UTC could be converted to the local CDT time of 4:30 PM.

UTC time must be synchronized with the National Institute of Standards and Technology (NIST), and the difference must be less than one minute.

When sending only a date (not date and time), it should be sent in the local time zone, and the receiver should interpret it in their local time. Neither party should attempt to convert the date to UTC.

The format of date/time fields in the XML schema must use the `xsd:dateTime` format: `CCYY-MM-DDTHH:MM:SS.FZ`, where the UTC time zone may be specified as `Z, +00:00`, or `-00:00`. For example, `2026-01-15T16:09:04.5Z` represents 16 hours, 09 minutes, 04 seconds, and 5 fractional seconds. The fractional seconds portion is not required. For simple date fields that do not include the time portion, the format is `CCYY-MM-DD`.

Character Set

The character set contains ASCII values 32 – 126, which includes:	
Symbols	! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [\ ] ^ _ ` { | } ~
Numerals	0 to 9
Letters, upper and lower case	A to Z, a to z

UTF-8 is the required character encoding for XML. Other encoding formats are not supported by Surescripts.

> Surescripts recommends that customers declare their UTF-8 formatting in the message header.

XML Escape Characters

XML uses certain characters to describe the structure of the document. When an element value needs to include one of these special characters, the character must be replaced with its predefined entity representation. The quote and apostrophe only need to be escaped when used in an attribute. Most XML processor libraries handle this replacement automatically.

Character	Character Name	XML Escape
"	Quote	&quot;
'	Apostrophe	&apos;
&	Ampersand	&amp;
<	Less than	&lt;
>	Greater than	&gt;

> **Note:** String lengths are calculated based on the un-escaped value. For example, >98 degrees is encoded as >98 degrees but is counted as a string length of 11, not 14.

Compliance

The goal of Surescripts is efficiency and consistency across the network so all customers can meet the highest measures of patient safety, end-to-end reliability, and quality. To ensure that customers comply with and adhere to the approved certification requirements, Surescripts:

- Monitors customers in production to ensure all network customers remain in compliance with certification requirements and contractual terms.
- Initiates a remediation process for identified compliance issues.

To ensure network and patient safety, customer agrees to notify Surescripts of any modifications through use of the Surescripts recertification form. Upon receipt of this form, changes will be reviewed, and a determination made as to what (if any) level of certification may be required before being placed into production.

When changes are made to a customer's implementation, the customer should advise their account manager. The customer will be given a recertification guide to provide details regarding the changes made. Upon receipt of the completed document, the details will be reviewed, and a determination will be made as to what (if any) level of recertification is necessary.

As a reminder, Surescripts conducts certification with customers to ensure the application adheres to network requirements. Surescripts will enforce mandatory fields as required by the Standards body and Surescripts guide requirements. To maximize interoperability, customers are recommended to support optional fields that have been created to address gaps in discrete data needs and the many solutions that are in place for the benefit of the receiver. Surescripts encourages, but does not guarantee, the use of optional discrete fields to support end user workflows.

This guide is intended for certification on our network only and is not intended to ensure compliance with state and federal law. In accordance with the customer's legal agreement with Surescripts, each customer is responsible for conducting its own due

diligence to ensure compliance with all applicable laws and requirements, including, but not limited to, local and state laws and regulations in which the customer's application is deployed and used.

3. How XDR Works: Flows and Notifications

This section describes how XDR-based messages move across the Surescripts network, the roles each system plays, and how message notifications and tracking work. The message construction details for each direction are covered in **Sending XDR Messages to Surescripts** and **Receiving XDR Messages from Surescripts**.

Roles and Architecture

XDR is based on the IHE Provide and Register Document Set-b transaction [ITI-41], which defines two roles: the Document Source (the system that initiates the transaction) and the Document Recipient (the system that receives it). On the Surescripts network, Surescripts operates as the Health Information Service Provider (HISP) that routes messages between the sending and receiving systems, and the roles shift depending on the direction of exchange:

When a customer **sends** a message, the customer's XDR interface is the Document Source and Surescripts is the Document Recipient.

When a customer **receives** a message, Surescripts is the Document Source and the customer's XDR interface is the Document Recipient.

Electronic message routing uses Hypertext Transfer Protocol over TLS (HTTPS), a request-response protocol. For each message that is sent, a response message is returned, and the status of the delivery is defined by the type of response (see **Message Notifications and Tracking**).

Differences from Standard IHE XDR

Surescripts places the following limitations on XDR as it is defined in the IHE ITI Technical Framework, Volume 2b, Section 3.41 (Provide and Register Document Set-b [ITI-41]):

The message must carry at least one document.

Each Provide and Register Document Set-b transaction uses the synchronous web services exchange model: the response is returned synchronously on the same connection, not as a separate asynchronously correlated message.

Only new document submissions are supported. There is no support for replacing or referencing documents from previous transactions.

XDS Folder definitions are not supported.

The Basic Patient Privacy Enforcement option is not supported.

Customers must follow all requirements in DirectTrust's *XDR and XDM for Direct Secure Messaging Specification*, Version 2.1.

> **Note:** The synchronous exchange model governs the acknowledgement of each individual transaction - the Provide and Register Document Set-b Response returned when a message is submitted. It does not describe the final delivery outcome. When delivery notifications are used, the outcome is communicated later and asynchronously through a Delivery Status Notification (DSN), which is itself a separate Provide and Register Document Set-b transaction sent in the reverse direction. See **Message Notifications and Tracking** for how the synchronous response and the asynchronous DSN work together.

Message Routing and Addressing

XDR-based messages are addressed using the direct:addressBlock SOAP header defined in the *XDR and XDM for Direct Secure Messaging Specification*, Version 2.1. The address must be part of the Direct domain assigned to the customer on the network. The address block identifies the sender and one or more recipients:

Address Block Element	Must Contain
direct:from	The Direct address of the user or role that sent the message.
direct:to	At least one recipient Direct address.

A sending system is **not** required to include a separate text/plain message body document. An XDR message should include the clinical document(s), metadata, and routing and addressing information required by the applicable standards and this guide. Some implementations choose to represent user-authored message body text as a separate text/plain document; when they do, a classCode of 56444-3 (Healthcare Communication) may be used for that general communication content. This is an optional representation pattern, not a requirement for XDR exchange. Clinical documents (for example, a C-CDA) carry their own type codes appropriate to their content and use case. Because a plaintext body is not a required interoperability artifact, it is not guaranteed to be retained end-to-end. Senders should not rely on message body text surviving transmission, and should place any clinically or operationally significant information in the clinical document(s) and metadata. Receiving systems should not rely on plaintext body content as the authoritative clinical payload.

Message metadata may conform to either Minimal Metadata or Full XDS Metadata, as defined in the *XDR and XDM for Direct Secure Messaging Specification*, Version 2.1, Section 6.1. Guidance on choosing between them is provided in **XDR-Based Messaging Best Practices**.

> **Note:** While the standard, and Surescripts, allows more than one direct:to element, multiple-recipient messages are discouraged for senders. See **Message Structure and Addressing** for sending guidance and **Multiple Recipients** for how multiple-recipient messages are handled.

Message Notifications and Tracking

This section describes the notification and tracking behavior that customers are expected to support in production. Two distinct responses are involved, and they occur at different times:

A **synchronous** Provide and Register Document Set-b Response acknowledges each transaction at the moment it is submitted (see **Synchronous Response (Provide and Register Document Set-b Response)**).

When delivery notifications are used, an **asynchronous** Delivery Status Notification (DSN) reports the final delivery outcome as a later, separate message (see **Delivery Status Notification (DSN)**).

Synchronous Response (Provide and Register Document Set-b Response)

The Provide and Register Document Set-b Response is the synchronous response exchanged between the customer's system and the Surescripts system. The system receiving the XDR message or DSN must generate the response to the system that sent it. The response value can be:

Success - the message is routable and addressed to a valid Direct domain.

Failure - the message has an invalid Direct domain or an invalid message format.

Important: A Success response means only that the message was accepted and has moved to the next stage in processing. It does **not** mean the message was delivered to the end recipient. Wait for the DSN to confirm the final delivery status.

Delivery Status Notification (DSN)

A Delivery Status Notification (DSN) is an asynchronous message that reports the final outcome of a message's delivery to its intended recipient. It is returned after the synchronous response, once a final delivery disposition is known. See **DSN Examples**.

Whether a sender receives DSNs is governed by the sender's delivery notification configuration (opt-in or opt-out), not by the message alone. Surescripts determines DSN behavior from that configuration; a direct:notification element in a submitted message is not used as the opt-in toggle and does not control DSN behavior. (On the DSN itself, direct:notification is used to correlate the notification to the original message - see **DSN Examples**.) A sender that is not configured to receive delivery notifications receives the synchronous response but no later DSNs.

Because Surescripts routes across multiple connectivity methods, the final disposition that drives a DSN may originate from the downstream delivery path - for example, an XDR Success or Failure response, a Net2Net Verify or Error, or a Message Disposition Notification (MDN) from a destination HISP. Surescripts uses that disposition to generate the DSN it returns to the sender:

A **DSN Success** indicates the message was delivered to the recipient.

A **DSN Failure** indicates the message was not delivered; the failure body includes a reason.

If a final disposition is not received within 60 minutes, the message times out and Surescripts generates a DSN Failure indicating that the message was sent but final delivery confirmation timed out.

For multiple-recipient messages, a sender may receive more than one DSN, because each recipient or downstream message split can be processed independently. Design message tracking to handle multiple final outcomes for a single submission, including mixed results where one recipient succeeds and another fails or times out.

> **Important:** Surescripts requires its customers to support Delivery Notification Tracking. However, a message may be routed to a recipient served by another HISP that does not support delivery tracking, in which case a final delivery confirmation is never returned. When that happens, the message times out as described above and Surescripts returns a DSN Failure. Timeouts are therefore common for certain recipients, and there is no way to know in advance which recipients they will affect. Design message tracking to treat a timeout as an unconfirmed outcome rather than a definite delivery failure.

Workflow Glossary

The workflows in this guide describe the exchange between the two endpoints of an XDR-based message. Surescripts, acting as a Health Information Service Provider (HISP), routes messages between these endpoints. Where the counterparty is served by a different HISP, Surescripts routes the message to that HISP, so a given endpoint may be a Surescripts customer or a system reached through another HISP.

Term	Definition
Sending System	The system that originates the XDR-based message and submits it for delivery. For a Surescripts customer sending a message, this is the customer's XDR interface; a message may also originate from a system served by another HISP and reach the customer through Surescripts.
Receiving System	The system to which the message is delivered for the recipient. This may be a Surescripts customer's XDR interface or a system reached through another HISP.
Recipient	The end user, provider, or organization for whom the message is intended, within the receiving system.

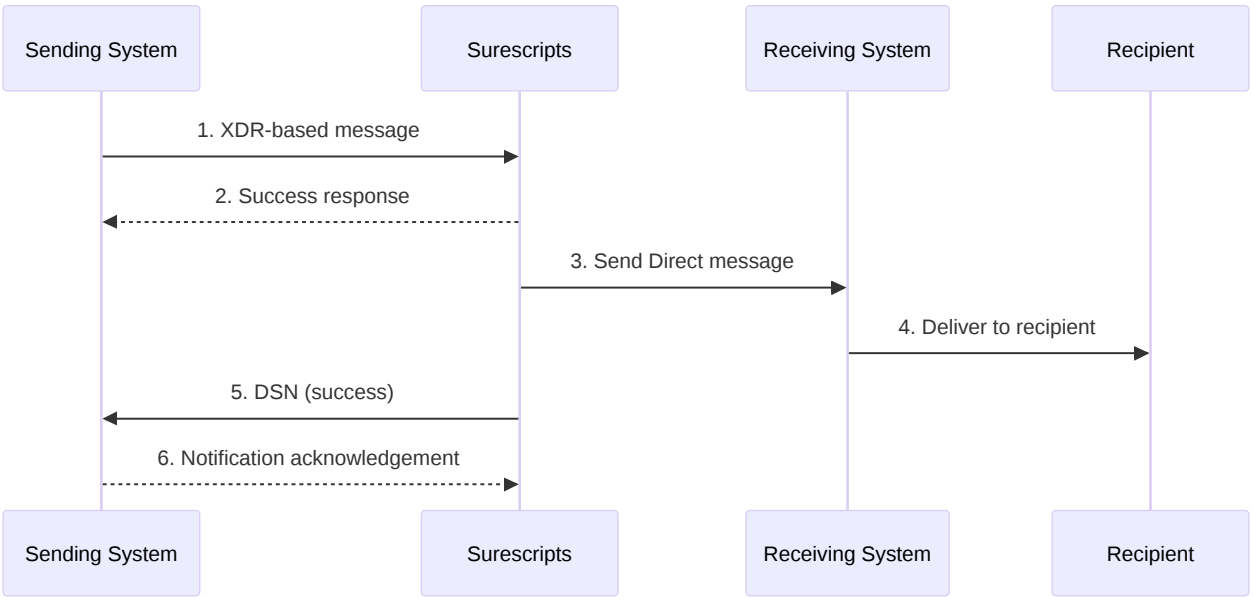
Detailed XDR Workflows

The following workflows illustrate the requests and responses exchanged between the sending and receiving systems for XDR-based messages and Delivery Status Notifications. They do not represent a comprehensive list of all possible scenarios. **Consolidated Message Examples** consolidates every message example in the guide and walks through a complete end-to-end exchange in sequence.

> **Note:** The workflows illustrate single-recipient messaging. Multiple-recipient messaging is handled similarly, but is discouraged for senders (see **Multiple Recipients**).

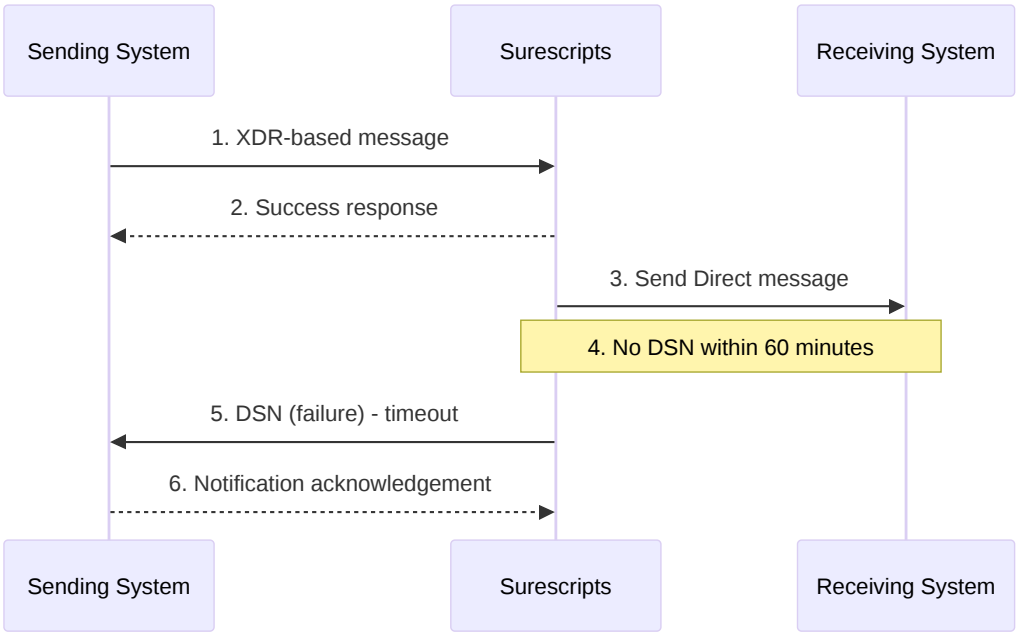
Successful Delivery with Delivery Notification

1. The Sending System sends an XDR-based message to Surescripts.
2. Surescripts returns a Success response to the Sending System.
3. Surescripts delivers the Direct message to the Receiving System.
4. The Receiving System delivers the message to the Recipient.
5. Surescripts sends a Delivery Status Notification (success) to the Sending System.
6. The Sending System returns a notification acknowledgement to Surescripts.



Timeout Without Receiving System Response

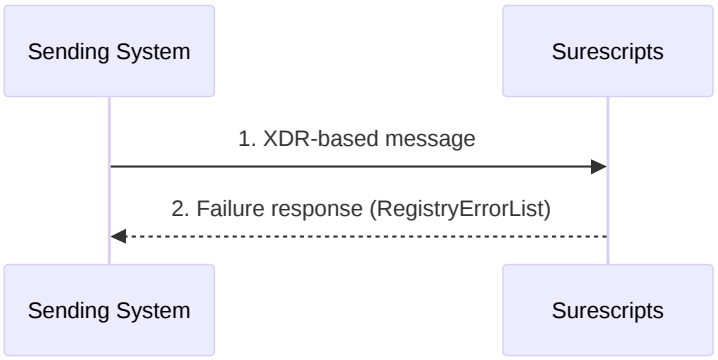
1. The Sending System sends an XDR-based message to Surescripts.
2. Surescripts returns a Success response to the Sending System.
3. Surescripts delivers the Direct message to the Receiving System.
4. The Receiving System does not return a follow-up DSN within 60 minutes.
5. Surescripts sends a Delivery Status Notification (failure) to the Sending System.
6. The Sending System returns a notification acknowledgement to Surescripts.



Synchronous Failure Response

1. The Sending System sends an XDR-based message to Surescripts.
2. Surescripts (or the Receiving System) cannot accept the message and returns a Failure response containing a RegistryErrorList.

   The Sending System processes the failure and does not receive a further DSN for the message.



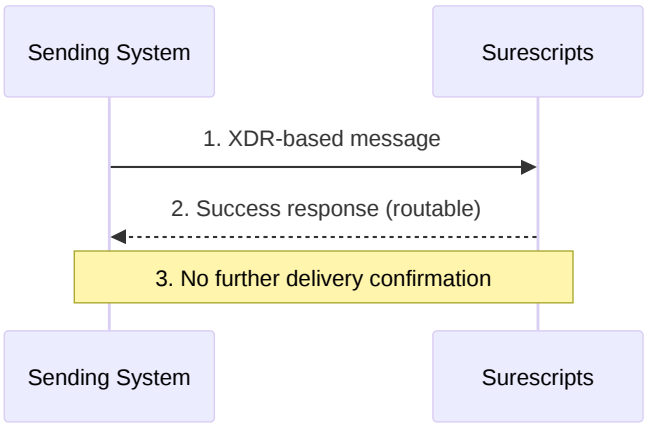
> **Note:** Failure responses can originate at different points in the transport or processing flow. See **XDR-Based Messaging Errors** for the distinction between transport/SOAP faults and application-level errors.

No Delivery Notification Tracking

When the sending system is not configured to receive delivery notifications, Surescripts confirms only that the Direct address is routable and returns a Success response. The sending system receives no further messages confirming delivery, and the final status must be obtained by contacting the recipient directly.

1. The Sending System sends an XDR-based message to Surescripts.
2. Surescripts verifies the address is routable and returns a Success response.

3. No further delivery confirmation is sent.



Multiple Recipients

Surescripts supports addressing a single XDR-based message to multiple recipients, but splits the message for routing so that delivery and notification are handled per recipient. A separate Delivery Status Notification is returned for each recipient on the message.

Because of this routing behavior, Surescripts does not encourage multiple-recipient messaging for senders; sending a separate message per recipient is recommended (see **Message Structure and Addressing** and **XDR-Based Messaging Best Practices**). On the receiving side, each receiving system receives only the recipients it hosts, and each split is handled independently, including its responses and Delivery Status Notifications (see **Message Structure**).

4. Sending XDR Messages to Surescripts

This section describes how a customer's XDR interface sends an XDR-based message to Surescripts. Sending and receiving are documented independently; a customer that only sends messages can implement the sending workflow here without implementing the full receiving workflow in **Receiving XDR Messages from Surescripts**. Note, however, that even a send-only implementation must be able to accept an inbound Provide and Register Document Set-b transaction and return its own valid synchronous response - a Success or Failure Provide and Register Document Set-b Response. A send-only system should expect to receive not only the delivery notifications (DSNs) it is configured to receive, but also replies and other valid XDR-based messages addressed to it. Even when the system will not accept or process the document, it must still respond, rejecting the message with a Failure response rather than leaving it unanswered, so that the message does not remain in an error delivery state. A send-only implementation must therefore support the synchronous response described in **Sending the Message Response**, even if it does not implement full inbound message processing. Receiving is otherwise covered in **Receiving XDR Messages from Surescripts**.

Sending Flow Overview

Sending an XDR-based message occurs when a customer's XDR interface initiates a Provide and Register Document Set-b [ITI-41] transaction that submits a clinical document and its metadata to Surescripts. Surescripts then routes the message toward the intended recipient. In this direction, the customer's XDR interface is the Document Source and Surescripts is the Document Recipient (see **Roles and Architecture**).

The high-level flow is:

- The customer's XDR interface constructs the message: the SOAP envelope, the direct:addressBlock, message metadata, and one or more documents.
- The customer submits the message to Surescripts over the mutual TLS connection.
- Surescripts returns a synchronous Provide and Register Document Set-b Response.
- If delivery notifications are used, a Delivery Status Notification follows asynchronously.

Message Structure and Addressing

An XDR-based message must include the direct:addressBlock SOAP header defined in the *XDR and XDM for Direct Secure Messaging Specification*, Version 2.1. The sending address must be part of the Direct domain assigned to the customer on the network.

Address Block Element	M/C	Description
direct:from	M	The Direct address of the user or role sending the message.
direct:to	M	At least one recipient Direct address.

The message body and its documents are then packaged into the Provide and Register Document Set-b request:

Body text (optional). A separate text/plain body document is not required, and message body text is not guaranteed to be retained end-to-end - senders should not rely on it and should place clinically significant information in the clinical document(s) and metadata. If a sender represents user-authored message body text as a separate document, a classCode of 56444-3 (Healthcare Communication) may be used for that content.

Clinical documents. One or more clinical documents (for example, a C-CDA or PDF) are included as additional documents, each with metadata appropriate to its content.

Metadata. The customer may generate either Minimal Metadata or Full XDS Metadata, as defined in the *XDR and XDM for Direct Secure Messaging Specification*, Version 2.1, Section 6.1. See **Sending XDR Messages to Surescripts** for the key metadata elements and **XDR-Based Messaging Best Practices** for guidance on choosing between them.

> **Note:** Although Surescripts accepts more than one direct:to element, sending to multiple recipients in a single message is discouraged. Surescripts splits multiple-recipient messages for routing, and sending a separate message per recipient produces more predictable delivery and tracking. See **XDR-Based Messaging Best Practices** for the corresponding best practice.

Message Metadata

XDR messages carry XDS metadata defined by the applicable DirectTrust and IHE specifications. This section is implementation guidance to help you construct messages that route and process across the network. It does not restate or replace the authoritative DirectTrust/IHE metadata requirements, and Surescripts does not enforce a separate metadata requirement model beyond those standards. Consult DirectTrust and IHE for the exact requirement level (required, required-if-known, or optional) of each element for the profile you implement, including limitedMetadata behavior.

The metadata profile is selected with the metadata-level element in the address block:

Minimal Metadata (metadata-level=minimal) is the profile used for most Direct messaging on the Surescripts network. It relaxes some XDS requirements and is flagged with the limitedMetadata classification node, per the standards. It does not prevent a sender from including richer document classifications, and in practice many senders (for example, EHRs) do.

Full XDS Metadata is the complete XDS profile, used when integrating with an XDS registry or repository, or when a use case or regulation requires it. See **XDR-Based Messaging Best Practices** for guidance on choosing between them.

The tables below summarize the metadata elements most relevant to constructing a message and supporting routing, patient matching, and downstream processing. For each element they give its practical purpose and the impact if it is missing. They are **not** a substitute for the DirectTrust/IHE metadata requirement tables; follow the standards for the exact requirement level. The classification and identification scheme identifiers (UUIDs) are the canonical IHE XDS values in ITI TF Volume 3. The **Complete Referral Example** shows where each element appears in a complete message.

Submission Set metadata:

Element	Practical Purpose	Implementation Guidance	Standards Reference	Impact if Missing
submissionTime	Records when the submission was created.	Populate in the XDS time format.	ITI TF Vol 3	Submission timing and audit ordering are lost.
SubmissionSet.uniqueId	Uniquely identifies the submission set.	Assign a globally unique OID.	ITI TF Vol 3	The submission cannot be uniquely referenced; may be rejected.
SubmissionSet.sourceId	Identifies the submitting system.	Use the source system's OID.	ITI TF Vol 3	The source system cannot be identified.
SubmissionSet.patientId	Ties the submission to a patient.	CX format with assigning authority (**Message Identification**); match the document's patientId.	ITI TF Vol 3; DirectTrust XDR/XDM v2.1	Patient association is weakened and matching is harder.
contentTypeCode	Describes the clinical activity of the submission.	Code the overall purpose (for example, a referral).	ITI TF Vol 3 / affinity-domain value set	Receiver has less context for triage and routing.
author (authorPerson, authorInstitution, authorTelecommunication, authorRole, authorSpecialty)	Identifies who and what sent the submission.	Populate the author details available.	ITI TF Vol 3	Provenance is unclear to the receiver.
intendedRecipient	Identifies the intended recipient(s).	Include recipient identity and telecom where available.	ITI TF Vol 3	Recipient context is reduced.

Document Entry metadata (per document):

Element	Practical Purpose	Implementation Guidance	Standards Reference	Impact if Missing
objectType	Marks the entry as a stable document.	Use the stable DocumentEntry objectType UUID.	ITI TF Vol 3	The entry may be invalid or rejected.
mimeType	Declares the document's media type.	Set correctly (for example, text/xml for a C-CDA).	ITI TF Vol 3	The receiver cannot parse or render the document.
DocumentEntry.uniqueId	Uniquely identifies the document.	Assign a globally unique OID.	ITI TF Vol 3	The document cannot be uniquely referenced; may be rejected.
DocumentEntry.patientId	Ties the document to a patient.	CX format; match SubmissionSet.patientId.	ITI TF Vol 3; DirectTrust XDR/XDM v2.1	Patient matching is weakened.
sourcePatientId	Patient identifier as known by the source system.	Include when available.	ITI TF Vol 3	Cross-referencing to the source patient is harder.
sourcePatientInfo	Patient demographics (name, DOB, gender, address) as HL7 PID fields.	Include when available.	ITI TF Vol 3	Fewer demographics for patient matching.
hash, size	Integrity check and size of the document.	Compute from the document content.	ITI TF Vol 3	Integrity and size validation are unavailable.
author (authorPerson, authorInstitution)	Identifies who authored the document.	Populate the author details available.	ITI TF Vol 3	Document provenance is unclear.
Description / title	Human-readable label for the document.	Provide a meaningful, PHI-free label.	ITI TF Vol 3	Users have a harder time identifying the document.
classCode	High-level document class.	Use the applicable class value set.	ITI TF Vol 3 / affinity-domain value set	Coarser classification and filing.
typeCode	Precise document type (for example, LOINC 34133-9 or 57133-1).	Use the applicable type value set.	ITI TF Vol 3 / affinity-domain value set	The receiver cannot precisely categorize or file the document.
formatCode	Technical format (for example, the C-CDA format code).	Use the applicable format value set.	ITI TF Vol 3 / affinity-domain value set	The receiver cannot select the correct parser or renderer.
confidentialityCode	Confidentiality level of the document.	Use the applicable confidentiality value set.	ITI TF Vol 3	Confidentiality handling may default incorrectly.

Element	Practical Purpose	Implementation Guidance	Standards Reference	Impact if Missing
healthcareFacilityTypeCode	Type of facility where the document originated.	Use the applicable value set.	ITI TF Vol 3 / affinity-domain value set	Less routing and clinical context.
practiceSettingCode	Clinical specialty or setting.	Use the applicable value set.	ITI TF Vol 3 / affinity-domain value set	Less routing and clinical context.
creationTime	When the document was created.	Populate in the XDS time format.	ITI TF Vol 3	Document timing is unknown.
languageCode	Language of the document.	For example, en-US.	ITI TF Vol 3	Language handling may default incorrectly.

Note: Under full XDS metadata, the document classifications (classCode, typeCode, formatCode, confidentialityCode, healthcareFacilityTypeCode, practiceSettingCode) are Required; under the minimal (limited) metadata profile they relax to Required-if-Known, so populate them when the values are available. Full-featured senders commonly include them to improve downstream routing and filing (see **Complete Referral Example**).

Note: The limitedMetadata classification is defined by the IHE and DirectTrust specifications; consult those standards to determine when and how it applies. It is attached as a Classification carrying only a classificationNode (no code value or nodeRepresentation) - its presence alone is the flag. In the **Complete Referral Example** it appears on the submission set (urn:uuid:5003a9db-8d8d-49e6-bf0c-990e34ac7707) and on each document (urn:uuid:ab9b591b-83ab-4d03-8f5d-f93b1fb92e85).

Message Identification

The W3C *Web Services Addressing 1.0 - Core* recommendation defines the MessageID and RelatesTo headers. Surescripts uses these headers to uniquely identify each XDR-based message and to associate replies with the messages they respond to. At a minimum, customers must populate the MessageID header.

MessageID must be globally unique. Surescripts recommends a Universally Unique Identifier (UUID).

RelatesTo is populated on a reply or notification with the MessageID of the original message, so the reply can be associated with the correct workflow item (such as an open referral) in the recipient's system.

Message Identification Headers (MessageID / RelatesTo)

Example

```
<wsa:MessageID>urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f</wsa:MessageID>
<wsa:RelatesTo>urn:uuid:1a2b3c4d-5e6f-4a7b-8c9d-0e1f2a3b4c5d</wsa:RelatesTo>
```

> **Note:** This example is for illustrative purposes only.

Patient Identification

When a message concerns a specific patient, include a patient identifier in the XDS metadata, formatted according to the applicable DirectTrust and IHE requirements (the HL7 CX data type). Where available, include an assigning authority so receiving systems can interpret the identifier in the correct organizational context, because the same identifier value can mean different things across organizations.

The following illustrates one common way a patient identifier and assigning authority may be represented:

```
PATID1234^^^&1.3.6.1.4.1.99999.1&ISO
```

This example is illustrative and does not redefine the DirectTrust, IHE, or HL7 formatting requirements. Invalidly formatted or incomplete identifiers may be accepted for processing but may not translate cleanly downstream or be usable for patient matching.

> **Note:** A single XDR-based message should concern a single patient. If a message is not related to a specific patient, do not populate patient identification.

Because patient identifiers are passed unchanged from multiple parties, Surescripts cannot guarantee that a patient identifier is unique. See **XDR-Based Messaging Best Practices** for best practices on patient matching, including verifying matches and allowing users to correct mismatches.

Managing Attachment Names

Messages may traverse different packaging and transport models as they move through Surescripts processing. For example, an originating message may package content as XDM while the final recipient is configured for XDR; in that case the XDR recipient receives an XDR message containing the extracted document(s), not the original XDM archive.

Attachment file names are **not preserved end-to-end** between senders and receivers, and should not be used as the basis for document identification, workflow routing, document classification, patient matching, or reconciliation. Receiving systems should rely instead on metadata, document identifiers, MIME/content type, document classification, and document content, and should expect file names to be generated, normalized, or transformed.

> **Note:** The URI element of the Extrinsic Object may carry a document file name, and when messages are converted to XDM attachments the IHE XDM media file naming constraints (the ISO 9660 convention: eight uppercase characters plus a three-character extension) may be applied. Treat this as packaging/interchange behavior, not a guarantee that file names persist or remains meaningful to the receiver.

Common clinical attachment types and extensions:

Document Type	Extension
Consolidated CDA (C-CDA)	.xml
PDF	.pdf
Plain text	.txt
Continuity of Care Record (CCR)	.xml
Image	.jpg, .png

Submitting the Message

An XDR-based message is submitted to Surescripts as an HTTP POST to the Surescripts XDR document repository endpoint, over the mutual TLS connection described in **Connectivity and Mutual TLS (mTLS)**. The endpoint URL differs by environment and is provided during onboarding; see the Connectivity and Authentication Guide for the endpoint values for your environment.

Packaging. The request uses SOAP 1.2 with MTOM/XOP (Message Transmission Optimization Mechanism) and is sent as a multipart/related body. The SOAP envelope is the root part. Binary document content may be either:

inlined as Base64 inside the `<Document>` element, or

carried in a separate MIME part and referenced from `<Document>` using an `<xop:Include href="cid:..."/>` element (MTOM). MTOM is recommended for larger attachments.

Required header. The request carries a single Content-Type header describing the multipart/related MTOM body:

Text

```
Content-Type: multipart/related; type="application/xop+xml"; start="<http://tempuri.org/0>"; start-info="application/soap+xml"; bo
```

The parameters must be consistent with the message body:

start must equal the Content-ID of the root (SOAP) part - for example, < `<http://tempuri.org/0>` >.

boundary may be any unique token, but it must exactly match the -- `<boundary>` delimiter that precedes each MIME part and the closing -- `<boundary>` -- delimiter at the end of the body.

type and start-info identify the root part as an MTOM-optimized SOAP 1.2 message and do not change.

The SOAP action is carried in the WS-Addressing `<a:Action>` element inside the SOAP header (see the **Complete Referral Example**); a separate SOAPAction HTTP header is not required.

Response. A successful submission returns HTTP 200 OK with a SOAP RegistryResponse in the body, indicating Success or Failure (**Handling the Synchronous Response**). A transport or TLS problem may instead surface as an HTTP-level error or a SOAP Fault; see **XDR-Based Messaging Errors** for the distinction between transport/SOAP faults and application-level errors.

Submitting a Message With Curl (Mutual TLS)

Example

```
curl --cert client-cert.pem --key client-key.pem --cacert surescripts-ca-chain.pem \
  --request POST "https://[xdr-endpoint-provided-at-onboarding]" \
  --header 'Content-Type: multipart/related; type="application/xop+xml"; start="<http://tempuri.org/0>"; start-info="application/s
  --data-binary @referral-message.mtom
```

> **Note:** This example is for illustrative purposes only. Replace the certificate and key paths and the endpoint with the values for your environment (provided during onboarding; see the Connectivity and Authentication Guide). The file referral-message.mtom is the MTOM/multipart/related payload (see the **Complete Referral Example**), and its internal boundary must match the boundary parameter above.

Handling the Synchronous Response

When the customer submits a message, Surescripts returns a synchronous Provide and Register Document Set-b Response (see **Synchronous Response (Provide and Register Document Set-b Response)**):

Success - Surescripts accepted the message for delivery. The response contains a RegistryResponse with a status of Success.

Failure - Surescripts could not accept the message. The response contains a RegistryResponse with a status of Failure and a RegistryErrorList element describing one or more errors.

Important: A Success response confirms only that the message was accepted for delivery. It does not confirm delivery to the recipient (see **Synchronous Response (Provide and Register Document Set-b Response)**). Delivery is confirmed by the DSN.

A Failure response carries a RegistryResponse with a Failure status and a RegistryErrorList containing one or more RegistryError elements. Each RegistryError provides an errorCode, a severity, and a codeContext.

Important: In a failure that Surescripts generates, errorCode carries a human-readable description of the condition detected rather than a standards-defined code, and codeContext carries a fixed general value. Do not branch programmatically on either one, and do not assume the text is stable across releases. Treat the RegistryResponse Failure status as the machine-readable outcome, log the complete RegistryError for diagnosis, and surface the errorCode text to users or support staff. See **XDR-Based Messaging Errors** and **XDR-Based Messaging Best Practices**.

Synchronous Failure Response (RegistryResponse with a RegistryErrorList)

XML

```xml
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
    xmlns:a="http://www.w3.org/2005/08/addressing">
    <s:Header>
        <a:Action s:mustUnderstand="1">urn:ihe:iti:2007:ProvideAndRegisterDocumentSet-bResponse</a:Action>
        <a:RelatesTo>urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f</a:RelatesTo>
    </s:Header>
    <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <RegistryResponse status="urn:oasis:names:tc:ebxml-regrep:ResponseStatusType:Failure"
            xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:rs:3.0">
            <RegistryErrorList>
                <RegistryError errorCode="Recipient Direct address not found in the Directory: intake@direct.example-hospital.com"
                    codeContext="Invalid XDR Message"
                    severity="Error"
                    location=""/>
            </RegistryErrorList>
        </RegistryResponse>
    </s:Body>
</s:Envelope>
```

Note: This example is for illustrative purposes only. In a Surescripts-generated failure, errorCode carries a description of the condition detected, codeContext carries a fixed general value such as Invalid XDR Message, and severity is the short token Error rather than the full urn:oasis:names:tc:ebxml-regrep:ErrorSeverityType:Error URN. The location attribute may contain a non-meaningful placeholder value; ignore it. The specific text returned depends on the condition detected and should not be parsed.

For the distinction between transport/SOAP faults and application-level errors, and how to handle each, see **XDR-Based Messaging Errors**. For error-handling best practices, see **XDR-Based Messaging Best Practices**.

Interpreting Delivery Status Notifications

When the sending system is configured to receive delivery notifications, Surescripts returns a Delivery Status Notification (DSN) asynchronously once delivery is determined. A message addressed to multiple recipients may produce more than one DSN (see **Delivery Status Notification (DSN)**):

A **DSN Success** confirms the message was delivered to the recipient.

A **DSN Failure** indicates the message was not delivered; the failure body includes a reason (see **DSN Examples**).

The sending system should use the DSN - not the synchronous Success response - as the signal of final delivery, and should reflect that status in the user's workflow (see **XDR-Based Messaging Best Practices**).

No Delivery Notification Tracking

When the sending system is not configured to receive delivery notifications, Surescripts verifies that the Direct address is routable and, if it is, returns a Success response. The sending system receives no further messages confirming delivery, and the final status must be obtained by contacting the recipient directly.

Synchronous Success Response (No Delivery Notification Tracking)

XML

```xml
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
    xmlns:a="http://www.w3.org/2005/08/addressing">
    <s:Header>
        <a:Action s:mustUnderstand="1">urn:ihe:iti:2007:ProvideAndRegisterDocumentSet-bResponse</a:Action>
        <a:RelatesTo>urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f</a:RelatesTo>
    </s:Header>
    <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <RegistryResponse status="urn:oasis:names:tc:ebxml-regrep:ResponseStatusType:Success"
            xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:rs:3.0"/>
    </s:Body>
</s:Envelope>
```

Notes:

- This example is for illustrative purposes only. A Success response indicates only that the message was accepted for delivery, not that it reached the recipient.
- A send-only Direct address must still return a response to any valid XDR-based message it receives (for example, an automatic acknowledgement or error) so that the message does not remain in an error delivery state (see **Sending the Message Response**).

Complete Referral Example

A complete clinical document submission - a referral carrying a C-CDA clinical summary, modeled on a validated production-style message - is provided in **Consolidated Message Examples**, Example C.3-1. It uses minimal metadata (note the limitedMetadata classification nodes) while still including the document classifications that full-featured senders commonly send; the C-CDA payload is omitted, with a comment marking where it belongs.

In that example, addresses are shown without a URI scheme; some senders use the mailto: scheme instead (mailto:dr.smith@...). The classification and identification scheme identifiers (UUIDs) are canonical IHE XDS values (ITI TF Volume 3); code values are representative and should be chosen per use case. See **Sending XDR Messages to Surescripts** for what each element means and **Submitting the Message** for the HTTP headers.

5. Receiving XDR Messages from Surescripts

This section describes how a customer's XDR interface receives an XDR-based message delivered from Surescripts. Sending and receiving are documented independently; a customer that only receives messages can implement from this section without implementing **Sending XDR Messages to Surescripts**.

Receiving Flow Overview

Receiving an XDR-based message occurs when a customer's XDR interface accepts a message delivered from Surescripts. In this direction, Surescripts is the Document Source and the customer's XDR interface is the Document Recipient (see **Roles and Architecture**). Surescripts delivers the message over the mutual TLS connection described in **Connectivity and Mutual TLS (mTLS)**, and the customer's system returns a synchronous response (see **Sending the Message Response**).

Customers should publish and maintain their Direct addresses in the Surescripts Directory. **Directory-based addressing is the expected and strongly recommended routing model** for Surescripts Direct messaging: Surescripts matches the recipient Direct address to a registered provider or organization record, which identifies the customer interface for delivery. Publishing addresses to the Directory is what makes recipients reliably discoverable and routable across the network, and Surescripts expects customers to publish and keep their addresses current.

Domain-based routing - routing on the recipient's Direct domain rather than on an individually registered address - is technically supported and can be enabled per domain. It is **not** recommended: it reduces recipient discoverability and tends to produce stale or unpublished addresses and other data-quality and operational issues over time. Use domain-based routing only when specific addresses genuinely cannot be published in the Directory, and publish addresses to the Directory wherever possible even where domain-based routing is enabled.

For address publication, endpoint configuration, participant onboarding, and routing configuration, refer to the Surescripts Directory Implementation Guide.

Message Structure

Surescripts conforms to Section 6.1 of the *XDR and XDM for Direct Secure Messaging Specification*, Version 2.1, for the SOAP headers used in Direct Secure Messaging. An incoming message contains the following:

Element	Contains
direct:addressBlock (SOAP header)	One direct:from element (the sender) and one or more direct:to elements (the recipients).
XDS metadata authorPerson	Sender information.
intendedRecipient	Recipient address or addresses.

A received message carries the same XDS metadata structure shown in the **Complete Referral Example** - a submission set and one or more document entries, each with their classifications and identifiers. The elements above are those most relevant when consuming a message. For a complete received message as delivered to a receiving endpoint, see **Consolidated Message Examples**.

Multiple recipients. Surescripts splits a message addressed to multiple recipients by receiving system. An XDR receiving system receives a message addressed to the recipient(s) it hosts; recipients served by other systems are delivered separately and are generally not listed on the message your system receives. When more than one recipient is hosted by the same receiving system, those recipients appear together as multiple direct:to elements in the single message delivered to that system, as shown below.

Each split is handled independently: the responses and Delivery Status Notifications for one receiving system are not shared with another.

Illustrative Incoming Address Block: Two Recipients Hosted by the Same Receiving System

Example

```
<direct:addressBlock soap:role="urn:direct:addressing:destination" soap:relay="true" xmlns:direct="urn:direct:addressing">
    <direct:from>dr.smith@direct.example-clinic.com</direct:from>
    <direct:to>intake@direct.example-hospital.com</direct:to>
    <direct:to>records@direct.example-hospital.com</direct:to>
</direct:addressBlock>
```

Note: This example is illustrative and shows two recipients hosted by the same receiving system. Recipients served by other systems are delivered as separate messages and would not appear here.

Patient Identification

Surescripts forms the outbound XDR message using information available from the inbound Clinical Direct message after normalizing and processing it for routing. Because messages can originate through different entry points and formats, the patient identification available in XDR metadata can vary.

When the originating system supplies complete XDR or XDM metadata, that metadata is the most reliable source for patient identifiers and demographics in the outbound XDR message. When it does not, some patient information may be present only within clinical attachments (such as C-CDA XML or HTML, HL7 v2 content, or PDFs), or may not be available in structured metadata at all.

Receiving systems should not rely on XDR metadata alone for patient matching. Determine whether your workflows need to inspect the attachment types you expect to receive for additional demographics or identifiers, and provide review handling when patient data is missing, incomplete, or ambiguous.

> **Note:** Because patient identifiers are passed unchanged from external parties, the receiving system must not rely on the identifier alone to match the clinical information to a patient record. See **XDR-Based Messaging Best Practices** for patient-matching best practices, including verifying matches and allowing users to correct mismatches.

Sending the Message Response

When the customer's XDR interface receives a message, it must return a synchronous Provide and Register Document Set-b Response:

If the system can accept the message for delivery, it returns a Success response.

If the system cannot accept the message, it returns a Failure response containing a RegistryErrorList element within the RegistryResponse element.

To interoperate with the Surescripts network, the customer must implement, at a minimum, the same XDR extensions that Surescripts implements from the *XDR and XDM for Direct Secure Messaging Specification*, Version 2.1. This includes the Direct address block and the Minimal Metadata specification. Surescripts also accepts Full XDS Metadata.

> **Note:** See **XDR-Based Messaging Best Practices** for best practices on handling temporary connectivity failures, queuing, and retries.

No Delivery Notification Tracking

When the receiving system is not configured to send delivery notifications, Surescripts translates the synchronous responses into delivery status as follows:

If the message was received and delivered to the recipient, the Success response is treated as a DSN Success.

If the message was not successfully delivered, the Failure response is treated as a DSN Failure.

The message is then identified with its final status, and no further responses are expected.

DSN Examples

A Delivery Status Notification (DSN) confirms whether a message was delivered successfully or failed. It is sent asynchronously, after the synchronous response, as a separate ProvideAndRegisterDocumentSet-b transaction in the reverse direction (from the receiving side back toward the original sender). The disposition (success or failure) is Base64-encoded in the Document element, and the direct:notification element in the address block links the DSN to the original message.

The examples below show the notification header, the success and failure disposition bodies, a complete DSN message, the sender's acknowledgement of the DSN, and where each part is located.

> **Note:** The direct:notification element in the address block links a DSN to its original message. Whether a sender receives DSNs is governed by the sender's delivery notification configuration. Base64-decode the Document content to read the disposition.

Notification Header

Direct:notification in the address block, linking the DSN to the original message

```
<direct:addressBlock xmlns:direct="urn:direct:addressing" s:role="urn:direct:addressing:destination" s:relay="1">
    <direct:from>intake@direct.example-hospital.com</direct:from>
    <direct:to>dr.smith@direct.example-clinic.com</direct:to>
    <direct:notification relatesTo="urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f"/>
</direct:addressBlock>
```

Success Disposition

DSN success disposition (before Base64 encoding)

```
<direct:messageDisposition xmlns:direct="urn:direct:addressing">
    <direct:recipient>intake@direct.example-hospital.com</direct:recipient>
    <direct:disposition>success</direct:disposition>
</direct:messageDisposition>
```

Failure Disposition

DSN failure disposition with a reason (before Base64 encoding)

```
<direct:messageDisposition xmlns:direct="urn:direct:addressing">
    <direct:recipient>intake@direct.example-hospital.com</direct:recipient>
    <direct:disposition>failure</direct:disposition>
    <direct:reasonForFailure>Mailbox not found.</direct:reasonForFailure>
</direct:messageDisposition>
```

The direct:reasonForFailure element should contain a specific, actionable description. Examples of useful reason text include:

- Mailbox not found.
- Message exceeds the recipient's maximum size limit.
- Unsupported attachment type.
- Recipient system unavailable - delivery timed out.

Full DSN Message

A DSN is delivered as a complete ProvideAndRegisterDocumentSet-b transaction. The a:RelatesTo header and the direct:notification element both carry the original message's MessageID, and the Document holds the Base64-encoded disposition. A complete success DSN is shown in **Consolidated Message Examples**,

DSN Acknowledgement

Because the DSN is itself a ProvideAndRegisterDocumentSet-b transaction, the system that receives it (the original sender) returns a synchronous RegistryResponse to acknowledge it, closing the workflow. The a:RelatesTo header carries the DSN's MessageID.

Acknowledgement of the DSN (synchronous response)

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope" xmlns:a="http://www.w3.org/2005/08/addressing">
    <s:Header>
        <a:Action s:mustUnderstand="1">urn:ihe:iti:2007:ProvideAndRegisterDocumentSet-bResponse</a:Action>
        <a:RelatesTo>urn:uuid:2c8b0e13-9f1a-4b6e-8b5a-6d7e8f901234</a:RelatesTo>
    </s:Header>
    <s:Body>
        <RegistryResponse status="urn:oasis:names:tc:ebxml-regrep:ResponseStatusType:Success" xmlns="urn:oasis:names:tc:ebxml-regr
    </s:Body>
</s:Envelope>
```

DSN Locations in the Message File

Part	Location in the message
Notification header	Envelope → Header → direct:addressBlock → direct:notification
Success or failure disposition	Envelope → Body → ProvideAndRegisterDocumentSetRequest → Document (Base64-encoded)

Note: An error description should be included in the direct:reasonForFailure element of the failure disposition.

6. XDR-Based Messaging Best Practices

This section describes best practices that improve message reliability, interoperability, and workflow continuity. They complement the **Application Certification Requirements** by reducing operational risk and improving end-user outcomes. Customers are encouraged, but not required, to follow them.

Message Submission and Retry

XDR-based messaging operates across multiple systems and networks, so temporary outages can occur. Design for retry and resiliency to prevent message loss and reduce manual rework.

Retry submissions when temporary connection failures occur, using a controlled schedule with exponential backoff (for example, after 5, 15, 30, and 60 minutes) over a bounded window of roughly two hours.

Distinguish retryable errors (network timeouts, 503 Service Unavailable) from non-retryable errors (authentication failures, invalid message format). Do not retry non-retryable errors; surface them for correction.

Queue outbound messages that cannot be sent immediately so users can continue working without resending manually.

Treat retry as an operational safeguard, not an error state; users should only be involved when retries are ultimately exhausted.

Delivery Status Notifications

Synchronous responses confirm processing, not final delivery. Use DSNs (**Delivery Status Notification (DSN)**) as the signal of delivery outcome, and reflect that status in the user's workflow.

Ensure your account is configured (opt-in) to receive delivery notifications; DSN behavior is governed by that configuration, not by the message alone (**Delivery Status Notification (DSN)**).

Surface delivery status clearly to users, using a state model such as the one below, and display the reasonForFailure text on failures.

Distinguish a timeout-driven DSN Failure from a genuine delivery failure. A timeout means no delivery confirmation was returned - often because the recipient is served by a HISP that does not support delivery tracking - so the message may still have been delivered. Treat it as unconfirmed rather than definitively failed (**Delivery Status Notification (DSN)**).

Process a DSN for each recipient when a message was addressed to multiple recipients.

The following state model reconciles the synchronous response and the DSN into a single delivery status you can present to users:

State	Meaning	Source
Submitted	The message has been sent to Surescripts; no synchronous response yet.	Local (sending workflow)
Accepted	Surescripts accepted the message for delivery; not yet delivered.	Synchronous Success response
Rejected	Surescripts could not accept the message.	Synchronous Failure response
Delivered	The message reached the recipient.	DSN Success
Failed	The message was not delivered; a reason is provided.	DSN Failure with a delivery error
Unconfirmed	No delivery confirmation was returned within the timeout; the message may still have been delivered.	DSN Failure due to timeout
Partially Delivered	For a multi-recipient message, outcomes differ across recipients (some delivered, some failed or unconfirmed).	Per-recipient DSNs

> **Note:** "Retrying" is a useful transient state while a submission is being retried after a temporary connection failure (**Message Submission and Retry**), before it reaches Accepted or Rejected.

Message Context and Threading

Many clinical workflows rely on back-and-forth communication. Proper message identification keeps replies associated with the right conversation.

Populate MessageID on every message and RelatesTo on every reply or notification (**Message Identification**).

Associate replies with the originating request so follow-up actions - referral responses, care-coordination questions, public-health clarifications - can be tracked accurately.

Patient Matching

Accurate patient identification is essential when exchanging documents across organizations. Even when identifiers are present, downstream systems must verify and reconcile them.

Include patient identifiers and demographics whenever available to support accurate association.

Do not rely on XDR metadata alone for matching; when metadata is insufficient, inspect the attachment types you expect to receive (C-CDA XML or HTML, HL7 v2, PDFs, and other supported documents) for additional demographics or identifiers.

Use Object Identifiers (OIDs) as the assigning authority for patient identifiers where possible. For example, a CX-formatted identifier is PATID1234^^^&1.3.6.1.4.1.99999.1&ISO, where 1.3.6.1.4.1.99999.1 is the assigning authority's OID. Organizations can obtain an OID from the HL7 OID registry (see **Document References**).

Design workflows that let users review and confirm patient matches when identifiers originate from external systems, and correct mismatches without re-sending.

Route messages with missing or ambiguous patient data to a triage or review queue rather than silently failing or misrouting.

Clinical Attachments

Participants vary in the document types they can send and receive. Supporting multiple formats improves interoperability.

Accept common clinical document types, in roughly this priority: C-CDA, PDF, plain text, CCR, and images (PNG, JPG). At minimum, support C-CDA and PDF.

For the best user experience, provide direct viewing support for common clinical attachment types where practical. Regardless of viewing capability, applications must still satisfy the attachment download requirements in ACR C.205.

When an attachment is filtered or removed, notify the recipient that it was removed, in accordance with ACR C.205. A sender-facing notification may also be implemented as a workflow enhancement, but it does not substitute for recipient-facing notification.

Do not rely on attachment file names for document identification, routing, classification, or patient matching; file names are not preserved end-to-end. Use metadata, document identifiers, MIME/content type, and document content instead.

Treat the clinical document(s) and their metadata as the authoritative content of a message; do not rely on plaintext message body text as a clinical payload, since it is optional and is not guaranteed to be retained end to end.

Metadata

Metadata supports routing, matching, and downstream processing. Richer metadata improves interoperability even when document content varies.

Follow the applicable DirectTrust/IHE metadata requirements for the profile you implement; Surescripts does not enforce metadata beyond the standards. Include the common document classifications (**Message Metadata**) when your system can produce them, as they improve downstream routing and filing.

Use Full XDS Metadata when integrating with an XDS registry or repository, or when a use case or regulation requires it.

Preserve metadata consistently as messages pass through systems to avoid data loss or misinterpretation.

Error Management

Handling errors you receive:

- Handle SOAP Faults and XDR application-level errors as distinct failure types (**XDR-Based Messaging Errors**).
- Do not assume a SOAP success implies message acceptance or delivery.
- Do not branch programmatically on RegistryError/@errorCode or codeContext in a Surescripts-generated failure. errorCode carries descriptive text that can change between releases, and codeContext carries a fixed general value; neither is a stable identifier. Use the RegistryResponse Failure status as the machine-readable signal and treat the rest as diagnostic detail.
- Log complete RegistryError details for troubleshooting, and display user-friendly messages rather than raw SOAP faults or raw errorCode text.

Producing errors your system returns (see **Sending the Message Response**):

- When you cannot accept a message, return a valid RegistryResponse with a Failure status and a RegistryErrorList; do not fail silently or return a Success you cannot honor.
- Populate a specific, standards-based errorCode and an actionable codeContext, so the sender can respond programmatically and a person can understand the cause.
- For delivery failures reported by DSN, provide a specific reasonForFailure (for example, "Mailbox not found.") rather than a generic message.
- Do not include PHI or other sensitive detail in error text, consistent with ACR C.203, which keeps PHI out of message subjects.

- Keep your error semantics consistent so trading partners can rely on them.

Directory Management

To support consistent, searchable, high-quality Direct address directory data:

Publish and maintain your Direct addresses in the Surescripts Directory; directory publication is the expected routing model (**Receiving Flow Overview**). See the Surescripts Directory Implementation Guide for details, and **Relationship to Your Surescripts Agreement** for the corresponding Directory obligations.

Keep the addresses you publish synchronized with your source systems and accurate and up to date; stale or incorrect entries cause misrouting and delivery failures.

Support manual entry of Direct addresses. Not every valid Direct address is published in the Directory, so let users enter and save an address they have obtained directly and send to recipients who are not listed.

Display key provider information alongside each Direct address so users can confidently identify the intended recipient.

Clearly distinguish individual-provider addresses from organizational (non-provider) addresses to reduce misrouting.

Support intuitive search (provider name, Direct address, specialty, organization, NPI) and filtering, and provide data-quality or verification indicators.

Prefer Single-Recipient Messages

Although Surescripts accepts multiple direct:to recipients on one message, it splits multiple-recipient messages for routing. Sending a separate message per recipient produces more predictable delivery and tracking, and simplifies per-recipient DSN handling. Reserve multiple-recipient messages for cases where the sending workflow genuinely requires them.

7. Application Certification Requirements

The following Application Certification Requirements (ACRs) are validated during certification testing. All requirements marked "shall" must be met to achieve production status on the Surescripts network.

ACR	Requirement Description
C.200	When end-users are searching for a destination address, a minimum of the following data elements shall be made available to end-users: • Last name (required for Provider records only) • First name (required for Provider records only) • Clinic or Organization name (required for Organization records only) • Address line 1 • Address line 2 • City • State • Zip • Specialty
C.201	Applications receiving Direct addresses from Surescripts' Directory shall consume, and make available for use, both Provider and Organization records.
C.202	Applications receiving Direct addresses from Surescripts' Directory shall reciprocate by publishing their own Direct addresses to the Surescripts Directory.
C.203	Applications shall not include Protected Health Information (PHI) in the message subject.
C.204	Applications shall record and make available message delivery status(es).
C.205	The participant's application shall be able to display and make available for download, at a minimum, the attachment types listed below. (This does not apply when messages are ingested systematically.) If an attachment is filtered for any reason, such as for security purposes, the participant's application shall notify the recipient of the message the attachment was removed and the filename of the attachment. • HTML • PDF • TXT • CDA

Note: ACRs C.200-C.202 require integration with the Surescripts Directory service. See the Surescripts Directory Implementation Guide for address search, display, and publishing requirements.

8. DirectTrust Trust Framework and Flow-Down Terms

Clinical Direct Messaging (CDM) operates within the DirectTrust trust framework. Surescripts is an accredited Health Information Service Provider (HISP), Direct Certificate Authority (CA), and Direct Registration Authority (RA), and it participates in the DirectTrust aggregated Directory. Because of this, certain obligations from the governing DirectTrust policies flow down to you and your end users, and Surescripts binds customers to them by agreement. These terms apply to CDM as a whole, not only to XDR-based messaging, and this section calls out the ones most relevant to CDM implementation and operation; it is not the complete set.

> **Important:** This is a plain-language summary provided for awareness only. It does not reproduce, replace, or interpret the DirectTrust policies, and it is not legal advice. You are responsible for reading and complying with the full, current policies and with your agreement with Surescripts. If anything here differs from a DirectTrust policy or your Surescripts agreement, those documents take precedence.

Governing Policies

Three DirectTrust policies contain the obligations summarized here. This summary reflects the policy versions noted below; obtain current copies from DirectTrust at **directtrust.org/resources/compliance-and-key-policies**, or through your Surescripts contact, and always work from the current version.

Policy	Version summarized	Governs
DirectTrust HISP Policy	2.0.3 (July 3, 2025)	HISP operation, edge connections, and end-user obligations.
DirectTrust Community X.509 Certificate Policy	2.1 (May 27, 2025)	Certificate issuance, identity proofing, and key obligations.
DirectTrust Aggregated Directory Data Sharing Policy	4.0 (October 18, 2023)	Publishing and using Direct address directory data.

Where the HISP Policy and the Certificate Policy conflict, the Certificate Policy takes precedence (HISP Policy, Section 1.1.2).

Certificates for Your Direct Domain

As your HISP, CA, and RA, Surescripts obtains, holds, and manages the Direct certificate and its private keys for your domain, and renews the certificate automatically as long as your account is active and your domain information is current and accurate. Your obligations:

Protect your server and edge-system credentials for connecting to the Surescripts service from unauthorized use.

Keep your domain registration information current and accurate.

Promptly notify Surescripts if your access credentials are compromised or misused, or if your domain information changes or becomes inaccurate, so Surescripts can take any needed action (such as reissuing your access credentials).

Identity Proofing Your Organization and the Businesses You Enable

During onboarding, Surescripts identity-proofs your organization and the individuals who act as your organizational representative, Information System Security Officer, or device sponsor. You provide accurate registration information, documentation of your organization's legal existence, and attestation of your HIPAA category. Surescripts completes the proofing through an approved identity-verification service or, if you prefer, a notary-completed form; your implementation contact coordinates either option.

If you onboard other organizations to your software, you are required to identity-proof each organization you enable. You must be bound by a legally binding contract with, or an attestation to, Surescripts (as the RA) to do so, and you must proof each organization to the assurance level required for its certificate. Surescripts retains documentation of that binding, and you must make your identity-proofing records available on request.

Identity Proofing Your End Users

You are responsible for identity-proofing each of your end users to the assurance level required for the certificate they use, and for keeping that proofing current as users are added or change.

Your end users must protect their credentials, use them only for authorized and lawful Direct messaging, and promptly report a suspected credential compromise.

Your system must keep an accounting of end-user access sufficient to identify which end user sent or received messages at a given time, and make it available to Surescripts for auditing on request.

Accessing and Sharing the Directory

You are responsible for meeting the Directory terms below and for enforcing them with the organizations you onboard. Publish and maintain your Direct addresses - and those of the organizations you enable - in the Surescripts Directory.

Access is reciprocal and per-organization. An organization must publish at least one valid Direct address to access the aggregated Directory, and an organization that does not contribute does not receive access - even if other organizations you serve do.

Keep entries accurate and current, and update them promptly when information changes.

Use Directory data only for Direct messaging and permitted searches - never for marketing, solicitation, surveys, research, or resale, and do not share it outside authorized users.

Ensure non-messaging (Ancillary) users cannot view or use the Direct address field.

Remove an entry promptly when a provider is offboarded or an address is no longer valid, and respond promptly to any inaccuracy or misuse Surescripts reports to you (message sending may be blocked otherwise).

Ensure your users consent to the Directory terms before using Directory data.

Relationship to Your Surescripts Agreement

Your Surescripts agreement, including any Business Associate Agreement and terms carried through from these policies, may impose additional or more specific requirements than this summary. Certain obligations also survive the end of your participation: prohibited-use restrictions continue to apply, and you must remove Directory data that is not part of an ongoing exchange relationship.

9. XDR-Based Messaging Errors

This section provides non-normative, informational guidance on how errors and faults may be reported when exchanging XDR-based messages, and how to handle each. It is intended to help interpret failure responses and design appropriate error-handling behavior, both for failures you receive and for failures your own system returns (see **Error Management** for guidance on producing errors). For a complete failure response, see Example 4.8-1 (**Handling the Synchronous Response**). Delivery-outcome failures reported asynchronously - DSN failures and delivery timeouts - are covered in **Delivery Status Notification (DSN)** and **XDR-Based Messaging Best Practices**).

Errors fall into two broad categories:

Category	When it occurs	Examples	How to handle
Transport / SOAP-level fault	Before XDR processing occurs; the SOAP stack detects the fault, no XDR RegistryResponse is returned.	Authentication or authorization failures; TLS or connectivity errors; malformed or invalid SOAP envelopes.	Retry with exponential backoff for transient conditions (connectivity, timeouts); verify certificates, credentials, and endpoint. Do not retry authentication or malformed-message faults without fixing the cause. Escalate to Surescripts support if persistent.
XDR application-level error	After the SOAP exchange completes successfully; the response contains a RegistryResponse with a Failure status and a RegistryErrorList.	Invalid or missing addressing; message validation or schema errors; sender or recipient configuration issues; transformation or packaging failures.	Review RegistryErrorList details; correct the message content, addressing, or configuration. Do not retry without fixing the root cause. Note that errorCode carries descriptive text rather than a stable code; see "What Surescripts Returns in a Failure Response" below.

What Surescripts Returns in a Failure Response

When Surescripts rejects an XDR message, it populates the RegistryError as follows:

Attribute	What Surescripts sends
errorCode	A human-readable description of the condition detected, not a standards-defined code. The text varies by condition and is not guaranteed to be stable across releases.
codeContext	A fixed, general value such as Invalid XDR Message.
severity	The short token Error, not the full urn:oasis:names🇳🇿ebxml-regrep:ErrorSeverityType:Error URN.
location	May contain a non-meaningful placeholder value. Ignore it.

Important: Because errorCode carries variable descriptive text rather than a stable identifier, do not use it for programmatic branching or automated retry decisions, and do not pattern-match against the standard IHE codes listed below. Use the RegistryResponse Failure status as the machine-readable outcome, log the complete RegistryError for diagnosis, and present the errorCode text to users or support staff. Retry decisions should be based on the failure category in the table above and on the transport conditions described in **Message Submission and Retry**.

Standard IHE Error Codes for Your Own Responses

Surescripts does not emit the codes below. They are the standard IHE ITI Technical Framework (Volume 3) values defined for the Provide and Register Document Set-b transaction, and they are provided here as a reference for the failure responses **your** system returns when it cannot accept a message (see **Sending the Message Response** and **Error Management**). Using standards-defined codes in your own responses lets your trading partners handle your failures programmatically.

errorCode	Typical meaning	Handling
XDSRegistryError / XDSRepositoryError	A general error processing or storing the submission, used when no more specific code applies.	Review the accompanying detail; may be transient (retry with backoff) or may require correcting the message.
XDSRegistryBusy / XDSRepositoryBusy	The service is temporarily unable to process the request.	Transient - retry with exponential backoff (**Message Submission and Retry**).
XDSRegistryMetadataError	Required metadata is missing, malformed, or invalid.	Correct the metadata (**Message Metadata**) and resubmit; do not retry unchanged.
XDSRegistryDuplicateUniqueIdInMessage	Two objects in the same submission share a uniqueId.	Assign distinct uniqueId values and resubmit.
XDSDuplicateUniqueIdInRegistry	A uniqueId in the submission already exists on the receiving side.	Use a new uniqueId; do not resend the same submission unchanged.
XDSMissingDocument	Metadata references a document that was not included in the submission.	Include the referenced document, or remove its metadata, and resubmit.
XDSMissingDocumentMetadata	A MIME/MTOM document part has no corresponding metadata.	Add the document's metadata and resubmit.
XDSUnknownPatientId	The patient identifier is not recognized by the receiving side.	Verify the identifier and assigning authority (**Patient Identification**); correct and resubmit.
XDSPatientIdDoesNotMatch	The patientId differs between the submission set and a document.	Make the identifiers consistent and resubmit.
XDSNonIdenticalHash	A document's hash or size does not match the value declared in its metadata.	Recompute the hash and size from the actual content so they match, and resubmit.

Error and Fault Terminology

This guide avoids the ambiguous term "SOAP error." For clarity:

SOAP Fault - a failure that prevents XDR message processing from occurring.

XDR application-level error - a failure reported within a successful SOAP response, using a RegistryResponse with a RegistryErrorList.

This distinction separates transport-level failures from application-level rejections. See **Handling the Synchronous Response** and **Error Management** .

10. Consolidated Message Examples

This section collects the message examples used throughout the guide into one place and arranges them in the order they occur in a live exchange. It also adds complete examples for scenarios the body describes but does not show in full: a complete failure Delivery Status Notification (DSN) and a complete received message. Each entry notes the section where the example is introduced or discussed.

All examples use the synthetic, illustrative data described in the guide (Example Clinic sending to Example Hospital, patient Jane Doe). The same original message identifier (urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f) and DSN identifier appear across the examples, so a single message can be traced from submission through delivery notification and acknowledgement.

Index of Examples

End-to-End Exchange (Happy Path, with Delivery Notification)

The happy-path exchange proceeds in four steps, and the same original message identifier links every step. The full submission and DSN messages are shown after the steps below; the shorter responses are shown inline here.

Step 1 - The sender submits the message.

The customer POSTs the complete Provide and Register Document Set-b request (MTOM / multipart/related) to Surescripts. The full message is in **Complete Submission Message**, and the curl transport is in the **Submitting a Message With Curl (Mutual TLS)** example. Its MessageID is urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f.

Step 2 - Surescripts returns the synchronous response.

Surescripts accepts the message and returns a synchronous Success response whose RelatesTo carries the original MessageID. If the message could not be accepted, Surescripts would instead return a Failure response (**Synchronous Failure Response (RegistryResponse with a RegistryErrorList)** Example).

Synchronous Success response

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
    xmlns:a="http://www.w3.org/2005/08/addressing">
    <s:Header>
        <a:Action s:mustUnderstand="1">urn:ihe:iti:2007:ProvideAndRegisterDocumentSet-bResponse</a:Action>
        <a:RelatesTo>urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f</a:RelatesTo>
    </s:Header>
    <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <RegistryResponse status="urn:oasis:names:tc:ebxml-regrep:ResponseStatusType:Success"
            xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:rs:3.0"/>
    </s:Body>
</s:Envelope>
```

Step 3 - Surescripts returns the DSN.

Once delivery is determined, Surescripts sends the asynchronous DSN back toward the sender as its own Provide and Register Document Set-b transaction. The DSN's RelatesTo and direct:notification both carry the original MessageID, and the DSN has its own MessageID. See the complete success DSN and failure variant examples in this section

Step 4 - The sender acknowledges the DSN.

The sender returns a synchronous acknowledgement whose RelatesTo carries the DSN's MessageID, closing the workflow.

DSN acknowledgement

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
    xmlns:a="http://www.w3.org/2005/08/addressing">
    <s:Header>
        <a:Action s:mustUnderstand="1">urn:ihe:iti:2007:ProvideAndRegisterDocumentSet-bResponse</a:Action>
        <a:RelatesTo>urn:uuid:2c8b0e13-9f1a-4b6e-8b5a-6d7e8f901234</a:RelatesTo>
    </s:Header>
    <s:Body>
        <RegistryResponse status="urn:oasis:names:tc:ebxml-regrep:ResponseStatusType:Success"
            xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:rs:3.0"/>
    </s:Body>
</s:Envelope>
```

Failure path. If the submission is rejected at Step 2, Surescripts returns **Synchronous Failure Response (RegistryResponse with a RegistryErrorList)** instead of the Success response, and no DSN follows (**Synchronous Failure Response**). If the message is accepted but delivery later fails, Step 3 returns a failure DSN (**Complete DSN Message (Failure)**) rather than a success DSN.

Complete Submission Message

This is the complete submission referenced by **Complete Referral Example** a referral carrying a C-CDA clinical summary, modeled on a validated production-style message. It uses minimal metadata (note the limitedMetadata classification nodes) while still including the document classifications that full-featured senders commonly send. The C-CDA payload is omitted, with a comment marking where it belongs.

Complete Submission: Minimal Metadata With Document Classifications, MTOM (Attachment Payload Omitted) Example

Example

```
Content-Type: multipart/related; type="application/xop+xml"; start="<http://tempuri.org/0>"; start-info="application/soap+xml"; bo

--MIMEBoundary_xdr_example
Content-ID: <http://tempuri.org/0>
Content-Transfer-Encoding: 8bit
Content-Type: application/xop+xml; charset=utf-8; type="application/soap+xml"
MIME-Version: 1.0

<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">urn:ihe:iti:2007:ProvideAndRegisterDocumentSet-b</a:Action>
    <a:MessageID>urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f</a:MessageID>
    <a:ReplyTo><a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address></a:ReplyTo>
    <a:To s:mustUnderstand="1">https://[xdr-endpoint-provided-at-onboarding]/DocumentRepository_Service.svc</a:To>
    <direct:addressBlock s:role="urn:direct:addressing:destination" s:relay="1"
                         xmlns:direct="urn:direct:addressing">
      <direct:from>dr.smith@direct.example-clinic.com</direct:from>
      <direct:to>intake@direct.example-hospital.com</direct:to>
      <direct:metadata-level>minimal</direct:metadata-level>
    </direct:addressBlock>
  </s:Header>
  <s:Body>
    <ProvideAndRegisterDocumentSetRequest xmlns="urn:ihe:iti:xds-b:2007">
      <SubmitObjectsRequest xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:lcm:3.0">
        <RegistryObjectList xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0">

          <!-- ===== Document Entry: the referral C-CDA ===== -->
          <ExtrinsicObject id="doc-1" mimeType="text/xml"
                           objectType="urn:uuid:7edca82f-054d-47f2-a032-9b2a5b5186c1"
                           status="urn:oasis:names:tc:ebxml-regrep:StatusType:Approved">
            <Slot name="creationTime"><ValueList><Value>20260722200019</Value></ValueList></Slot>
            <Slot name="languageCode"><ValueList><Value>en-US</Value></ValueList></Slot>
            <Slot name="sourcePatientId">
              <ValueList><Value>PATID1234^^^&amp;1.3.6.1.4.1.99999.1&amp;ISO</Value></ValueList>
            </Slot>
            <Slot name="sourcePatientInfo">
              <ValueList>
                <Value>PID-3|PATID1234^^^&amp;1.3.6.1.4.1.99999.1&amp;ISO</Value>
                <Value>PID-5|Doe^Jane^^^^^L</Value>
                <Value>PID-7|19600615</Value>
                <Value>PID-8|F</Value>
                <Value>PID-11|123 Main St^^Portland^OR^97201^US</Value>
              </ValueList>
            </Slot>
            <Slot name="hash"><ValueList><Value>28f1b1d1d49a85bdd7a7cb4d7d87eb5d93b8ef45</Value></ValueList></Slot>
            <Slot name="size"><ValueList><Value>36100</Value></ValueList></Slot>
            <Name><LocalizedString value="Cardiology Referral"/></Name>
            <!-- limitedMetadata marker (minimal metadata) -->
            <Classification classifiedObject="doc-1" id="cl-doc-limited"
                            classificationNode="urn:uuid:ab9b591b-83ab-4d03-8f5d-f93b1fb92e85"/>
            <!-- author -->
            <Classification classificationScheme="urn:uuid:93606bcf-9494-43ec-9b4e-a7748d1a838d"
                            classifiedObject="doc-1" id="cl-doc-author" nodeRepresentation="">
              <Slot name="authorPerson"><ValueList><Value>E12345^Smith^John^^^^MD^^&amp;2.16.840.1.113883.3.9999.2&amp;ISO</Value>
              <Slot name="authorInstitution"><ValueList><Value>Example Clinic^^^^^&amp;2.16.840.1.113883.3.9999&amp;ISO^^^^0001</V
```

```
    </Classification>
    <!-- classCode -->
    <Classification classificationScheme="urn:uuid:41a5887f-8865-4c09-adf7-e362475b143a"
                    classifiedObject="doc-1" id="cl-doc-class" nodeRepresentation="57133-1">
      <Slot name="codingScheme"><ValueList><Value>2.16.840.1.113883.6.1</Value></ValueList></Slot>
      <Name><LocalizedString value="Referral Note"/></Name>
    </Classification>
    <!-- confidentialityCode -->
    <Classification classificationScheme="urn:uuid:f4f85eac-e6cb-4883-b524-f2705394840f"
                    classifiedObject="doc-1" id="cl-doc-conf" nodeRepresentation="N">
      <Slot name="codingScheme"><ValueList><Value>2.16.840.1.113883.5.25</Value></ValueList></Slot>
      <Name><LocalizedString value="Normal"/></Name>
    </Classification>
    <!-- formatCode -->
    <Classification classificationScheme="urn:uuid:a09d5840-386c-46f2-b5ad-9c3699a4309d"
                    classifiedObject="doc-1" id="cl-doc-format" nodeRepresentation="urn:hl7-org:sdwg:ccda-structuredBody:2
      <Slot name="codingScheme"><ValueList><Value>1.3.6.1.4.1.19376.1.2.3</Value></ValueList></Slot>
      <Name><LocalizedString value="HL7 C-CDA structured body"/></Name>
    </Classification>
    <!-- healthcareFacilityTypeCode -->
    <Classification classificationScheme="urn:uuid:f33fb8ac-18af-42cc-ae0e-ed0b0bdb91e1"
                    classifiedObject="doc-1" id="cl-doc-hft" nodeRepresentation="394747008">
      <Slot name="codingScheme"><ValueList><Value>2.16.840.1.113883.6.96</Value></ValueList></Slot>
      <Name><LocalizedString value="Ambulatory care site"/></Name>
    </Classification>
    <!-- practiceSettingCode -->
    <Classification classificationScheme="urn:uuid:cccf5598-8b07-4b77-a05e-ae952c785ead"
                    classifiedObject="doc-1" id="cl-doc-ps" nodeRepresentation="394579002">
      <Slot name="codingScheme"><ValueList><Value>2.16.840.1.113883.6.96</Value></ValueList></Slot>
      <Name><LocalizedString value="Cardiology"/></Name>
    </Classification>
    <!-- typeCode -->
    <Classification classificationScheme="urn:uuid:f0306f51-975f-434e-a61c-c59651d33983"
                    classifiedObject="doc-1" id="cl-doc-type" nodeRepresentation="57133-1">
      <Slot name="codingScheme"><ValueList><Value>2.16.840.1.113883.6.1</Value></ValueList></Slot>
      <Name><LocalizedString value="Referral Note"/></Name>
    </Classification>
    <!-- DocumentEntry.patientId -->
    <ExternalIdentifier identificationScheme="urn:uuid:58a6f841-87b3-4a3e-92fd-a8ffeff98427"
                        registryObject="doc-1" id="ei-doc-pid"
                        value="PATID1234^^^&amp;amp;1.3.6.1.4.1.99999.1&amp;amp;ISO">
      <Name><LocalizedString value="XDSDocumentEntry.patientId"/></Name>
    </ExternalIdentifier>
    <!-- DocumentEntry.uniqueId -->
    <ExternalIdentifier identificationScheme="urn:uuid:2e82c1f6-a085-4c72-9da3-8640a32e42ab"
                        registryObject="doc-1" id="ei-doc-uid"
                        value="2.16.840.1.113883.3.9999.1.100001">
      <Name><LocalizedString value="XDSDocumentEntry.uniqueId"/></Name>
    </ExternalIdentifier>
  </ExtrinsicObject>

  <!-- ===== Submission Set ===== -->
  <RegistryPackage id="SubmissionSet01"
                   status="urn:oasis:names:tc:ebxml-regrep:StatusType:Approved">
    <Slot name="submissionTime"><ValueList><Value>20260722200019</Value></ValueList></Slot>
    <Slot name="intendedRecipient">
      <ValueList><Value>Example Hospital^^^^^&amp;amp;2.16.840.1.113883.3.9999.3&amp;amp;ISO^^^^0002||^^Internet^intake@direct.exa
    </Slot>
    <Name><LocalizedString value="Cardiology referral"/></Name>
```

```
            <!-- author (with role and specialty) -->
            <Classification classificationScheme="urn:uuid:a7058bb9-b4e4-4307-ba5b-e3f0ab85e12d"
                            classifiedObject="SubmissionSet01" id="cl-ss-author" nodeRepresentation="">
              <Slot name="authorPerson"><ValueList><Value>E12345^Smith^John^^^^MD^^&amp;2.16.840.1.113883.3.9999.2&amp;ISO</Value>
              <Slot name="authorInstitution"><ValueList><Value>Example Clinic^^^^^&amp;2.16.840.1.113883.3.9999&amp;ISO^^^^0001</\
              <Slot name="authorRole"><ValueList><Value>Referring Physician</Value></ValueList></Slot>
              <Slot name="authorSpecialty"><ValueList><Value>Cardiology</Value></ValueList></Slot>
              <Slot name="authorTelecommunication"><ValueList><Value>^^Internet^dr.smith@direct.example-clinic.com</Value></ValueL
            </Classification>
            <!-- contentTypeCode -->
            <Classification classificationScheme="urn:uuid:aa543740-bdda-424e-8c96-df4873be8500"
                            classifiedObject="SubmissionSet01" id="cl-ss-content" nodeRepresentation="57133-1">
              <Slot name="codingScheme"><ValueList><Value>2.16.840.1.113883.6.1</Value></ValueList></Slot>
              <Name><LocalizedString value="Referral Note"/></Name>
            </Classification>
            <!-- limitedMetadata marker (minimal metadata) -->
            <Classification classifiedObject="SubmissionSet01" id="cl-ss-limited"
                            classificationNode="urn:uuid:5003a9db-8d8d-49e6-bf0c-990e34ac7707"/>
            <!-- SubmissionSet.sourceId -->
            <ExternalIdentifier identificationScheme="urn:uuid:554ac39e-e3fe-47fe-b233-965d2a147832"
                                registryObject="SubmissionSet01" id="ei-ss-src"
                                value="2.16.840.1.113883.3.9999.1">
              <Name><LocalizedString value="XDSSubmissionSet.sourceId"/></Name>
            </ExternalIdentifier>
            <!-- SubmissionSet.uniqueId -->
            <ExternalIdentifier identificationScheme="urn:uuid:96fdda7c-d067-4183-912e-bf5ee74998a8"
                                registryObject="SubmissionSet01" id="ei-ss-uid"
                                value="2.16.840.1.113883.3.9999.1.1">
              <Name><LocalizedString value="XDSSubmissionSet.uniqueId"/></Name>
            </ExternalIdentifier>
            <!-- SubmissionSet.patientId -->
            <ExternalIdentifier identificationScheme="urn:uuid:6b5aea1a-874d-4603-a4bc-96a0a7b38446"
                                registryObject="SubmissionSet01" id="ei-ss-pid"
                                value="PATID1234^^^&amp;1.3.6.1.4.1.99999.1&amp;ISO">
              <Name><LocalizedString value="XDSSubmissionSet.patientId"/></Name>
            </ExternalIdentifier>
          </RegistryPackage>

          <!-- Classifies the RegistryPackage as a Submission Set -->
          <Classification classifiedObject="SubmissionSet01" id="cl-ss-node"
                          classificationNode="urn:uuid:a54d6aa5-d40d-43f9-88c5-b4633d873bdd"/>

          <!-- The document is a member of the submission set -->
          <Association associationType="urn:oasis:names:tc:ebxml-regrep:AssociationType:HasMember"
                       sourceObject="SubmissionSet01" targetObject="doc-1" id="as-01">
            <Slot name="SubmissionSetStatus"><ValueList><Value>Original</Value></ValueList></Slot>
          </Association>

        </RegistryObjectList>
      </SubmitObjectsRequest>

      <!-- Referral C-CDA, carried as a separate MIME part via MTOM/XOP.
           The binary payload is omitted here; see the second MIME part below. -->
      <Document id="doc-1">
        <xop:Include href="cid:http://tempuri.org/1" xmlns:xop="http://www.w3.org/2004/08/xop/include"/>
      </Document>

    </ProvideAndRegisterDocumentSetRequest>
  </s:Body>
```

```
</s:Envelope>

--MIMEBoundary_xdr_example
Content-ID: <http://tempuri.org/1>
Content-Transfer-Encoding: binary
Content-Type: text/xml

***** The referral C-CDA document bytes belong here. *****
The binary content of the referral attachment is omitted from this example.
This MIME part is referenced by the <xop:Include> element in the Document above.
--MIMEBoundary_xdr_example--
```

Note: This example is for illustrative purposes only, and represents one valid way to construct a message; other structures and metadata combinations are equally valid. Addresses, OIDs, identifiers, demographics, and code values are samples, and the referral document content is omitted where indicated.

Complete DSN Message - Success

This is the complete success DSN referenced by **Full DSN Message**, delivered as its own ProvideAndRegisterDocumentSet-b transaction back toward the original sender. The a:RelatesTo header and the direct:notification element both carry the original message's MessageID, and the Base64-encoded Document decodes to the success disposition.

Complete DSN Message Example

Example

```
Content-Type: multipart/related; type="application/xop+xml"; start="<http://tempuri.org/0>"; start-info="application/soap+xml"; bc

--MIMEBoundary_dsn_example
Content-ID: <http://tempuri.org/0>
Content-Transfer-Encoding: 8bit
Content-Type: application/xop+xml; charset=utf-8; type="application/soap+xml"

<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">urn:ihe:iti:2007:ProvideAndRegisterDocumentSet-b</a:Action>
    <a:MessageID>urn:uuid:2c8b0e13-9f1a-4b6e-8b5a-6d7e8f901234</a:MessageID>
    <a:RelatesTo>urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f</a:RelatesTo>
    <a:From><a:Address>urn:oid:2.16.840.1.113883.3.9999.3</a:Address></a:From>
    <a:ReplyTo><a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address></a:ReplyTo>
    <direct:addressBlock s:role="urn:direct:addressing:destination" s:relay="1"
                         xmlns:direct="urn:direct:addressing">
      <direct:from>intake@direct.example-hospital.com</direct:from>
      <direct:to>dr.smith@direct.example-clinic.com</direct:to>
      <direct:metadata-level>minimal</direct:metadata-level>
      <direct:notification relatesTo="urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f"/>
    </direct:addressBlock>
    <a:To s:mustUnderstand="1">https://xdr.example-clinic.com/DocumentRepository_Service.svc</a:To>
  </s:Header>
  <s:Body>
    <ProvideAndRegisterDocumentSetRequest xmlns="urn:ihe:iti:xds-b:2007">
      <SubmitObjectsRequest xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:lcm:3.0">
        <RegistryObjectList xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0">
          <RegistryPackage id="SubmissionSet01"
                           status="urn:oasis:names:tc:ebxml-regrep:StatusType:Approved">
            <Slot name="submissionTime"><ValueList><Value>20260722200021</Value></ValueList></Slot>
            <Classification classificationScheme="urn:uuid:a7058bb9-b4e4-4307-ba5b-e3f0ab85e12d"
                            classifiedObject="SubmissionSet01" id="ss-c-1" nodeRepresentation="">
              <Slot name="authorTelecommunication"><ValueList><Value>^^Internet^intake@direct.example-hospital.com</Value></ValueL
            </Classification>
            <!-- limitedMetadata marker -->
            <Classification classifiedObject="SubmissionSet01" id="ss-c-2"
                            classificationNode="urn:uuid:5003a9db-8d8d-49e6-bf0c-990e34ac7707"/>
            <ExternalIdentifier id="ss-ei-1" registryObject="SubmissionSet01"
                                identificationScheme="urn:uuid:554ac39e-e3fe-47fe-b233-965d2a147832"
                                value="2.16.840.1.113883.3.9999.3">
              <Name><LocalizedString value="XDSSubmissionSet.sourceId"/></Name>
            </ExternalIdentifier>
            <ExternalIdentifier id="ss-ei-2" registryObject="SubmissionSet01"
                                identificationScheme="urn:uuid:96fdda7c-d067-4183-912e-bf5ee74998a8"
                                value="2.16.840.1.113883.3.9999.3.1">
              <Name><LocalizedString value="XDSSubmissionSet.uniqueId"/></Name>
            </ExternalIdentifier>
          </RegistryPackage>
          <Classification classifiedObject="SubmissionSet01" id="ss-classifier"
                          classificationNode="urn:uuid:a54d6aa5-d40d-43f9-88c5-b4633d873bdd"/>
          <ExtrinsicObject id="doc-1" mimeType="text/xml"
                           objectType="urn:uuid:7edca82f-054d-47f2-a032-9b2a5b5186c1">
            <Slot name="creationTime"><ValueList><Value>20260722200021</Value></ValueList></Slot>
            <Slot name="size"><ValueList><Value>221</Value></ValueList></Slot>
            <Slot name="hash"><ValueList><Value>149B15E48A99C96FDFFEF1473D1F71E5EF3FC5A0</Value></ValueList></Slot>
```

```
            <!-- limitedMetadata marker -->
            <Classification classifiedObject="doc-1" id="doc-c-1"
                            classificationNode="urn:uuid:ab9b591b-83ab-4d03-8f5d-f93b1fb92e85"/>
            <ExternalIdentifier id="doc-ei-1" registryObject="doc-1"
                                identificationScheme="urn:uuid:2e82c1f6-a085-4c72-9da3-8640a32e42ab"
                                value="2.16.840.1.113883.3.9999.3.2">
              <Name><LocalizedString value="XDSDocumentEntry.uniqueId"/></Name>
            </ExternalIdentifier>
          </ExtrinsicObject>
          <Association associationType="urn:oasis:names:tc:ebxml-regrep:AssociationType:HasMember"
                       sourceObject="SubmissionSet01" targetObject="doc-1" id="as-1">
            <Slot name="SubmissionSetStatus"><ValueList><Value>Original</Value></ValueList></Slot>
          </Association>
        </RegistryObjectList>
      </SubmitObjectsRequest>
      <!-- Base64-encoded direct:messageDisposition from Example 5.6-2 -->
      <Document id="doc-1">PGRpcmVjdDptZXNzYWdlRGlzcG9zaXRpb24geG1sbnM6ZGlyZWN0PSJ1cm46ZGlyZWN0OmFkZHJlc3NpbmciPg0KICA8ZGlyZWN0Onj
    </ProvideAndRegisterDocumentSetRequest>
  </s:Body>
</s:Envelope>
--MIMEBoundary_dsn_example--
```

Note: This example is for illustrative purposes only. The Document value Base64-decodes to the success disposition; the size and hash correspond to that decoded disposition.

Complete DSN Message - Failure

The body shows the failure disposition body and a complete success DSN (Example C.4-1), but not a complete failure DSN. The example below combines them: a complete DSN transaction whose Base64-encoded Document decodes to the failure disposition (mailbox not found). It is structurally identical to Example C.4-1 except for the disposition payload, its size and hash, and the DSN's own MessageID.

Complete DSN Message (Failure) Example

Example

```
Content-Type: multipart/related; type="application/xop+xml"; start="<http://tempuri.org/0>"; start-info="application/soap+xml"; bc

--MIMEBoundary_dsn_failure_example
Content-ID: <http://tempuri.org/0>
Content-Transfer-Encoding: 8bit
Content-Type: application/xop+xml; charset=utf-8; type="application/soap+xml"

<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">urn:ihe:iti:2007:ProvideAndRegisterDocumentSet-b</a:Action>
    <a:MessageID>urn:uuid:3d9c1f24-a0b2-4c5d-9e6f-7a8b9c0d1e2f</a:MessageID>
    <a:RelatesTo>urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f</a:RelatesTo>
    <a:From><a:Address>urn:oid:2.16.840.1.113883.3.9999.3</a:Address></a:From>
    <a:ReplyTo><a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address></a:ReplyTo>
    <direct:addressBlock s:role="urn:direct:addressing:destination" s:relay="1"
                         xmlns:direct="urn:direct:addressing">
      <direct:from>intake@direct.example-hospital.com</direct:from>
      <direct:to>dr.smith@direct.example-clinic.com</direct:to>
      <direct:metadata-level>minimal</direct:metadata-level>
      <direct:notification relatesTo="urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f"/>
    </direct:addressBlock>
    <a:To s:mustUnderstand="1">https://xdr.example-clinic.com/DocumentRepository_Service.svc</a:To>
  </s:Header>
  <s:Body>
    <ProvideAndRegisterDocumentSetRequest xmlns="urn:ihe:iti:xds-b:2007">
      <SubmitObjectsRequest xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:lcm:3.0">
        <RegistryObjectList xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0">
          <RegistryPackage id="SubmissionSet01"
                           status="urn:oasis:names:tc:ebxml-regrep:StatusType:Approved">
            <Slot name="submissionTime"><ValueList><Value>20260722200521</Value></ValueList></Slot>
            <Classification classificationScheme="urn:uuid:a7058bb9-b4e4-4307-ba5b-e3f0ab85e12d"
                            classifiedObject="SubmissionSet01" id="ss-c-1" nodeRepresentation="">
              <Slot name="authorTelecommunication"><ValueList><Value>^^Internet^intake@direct.example-hospital.com</Value></ValueL
            </Classification>
            <!-- limitedMetadata marker -->
            <Classification classifiedObject="SubmissionSet01" id="ss-c-2"
                            classificationNode="urn:uuid:5003a9db-8d8d-49e6-bf0c-990e34ac7707"/>
            <ExternalIdentifier id="ss-ei-1" registryObject="SubmissionSet01"
                                identificationScheme="urn:uuid:554ac39e-e3fe-47fe-b233-965d2a147832"
                                value="2.16.840.1.113883.3.9999.3">
              <Name><LocalizedString value="XDSSubmissionSet.sourceId"/></Name>
            </ExternalIdentifier>
            <ExternalIdentifier id="ss-ei-2" registryObject="SubmissionSet01"
                                identificationScheme="urn:uuid:96fdda7c-d067-4183-912e-bf5ee74998a8"
                                value="2.16.840.1.113883.3.9999.3.11">
              <Name><LocalizedString value="XDSSubmissionSet.uniqueId"/></Name>
            </ExternalIdentifier>
          </RegistryPackage>
          <Classification classifiedObject="SubmissionSet01" id="ss-classifier"
                          classificationNode="urn:uuid:a54d6aa5-d40d-43f9-88c5-b4633d873bdd"/>
          <ExtrinsicObject id="doc-1" mimeType="text/xml"
                           objectType="urn:uuid:7edca82f-054d-47f2-a032-9b2a5b5186c1">
            <Slot name="creationTime"><ValueList><Value>20260722200521</Value></ValueList></Slot>
            <Slot name="size"><ValueList><Value>294</Value></ValueList></Slot>
            <Slot name="hash"><ValueList><Value>0E2C7D036FDB41DEE1538F4ECD47553C384001A0</Value></ValueList></Slot>
```

```
            <!-- limitedMetadata marker -->
            <Classification classifiedObject="doc-1" id="doc-c-1"
                            classificationNode="urn:uuid:ab9b591b-83ab-4d03-8f5d-f93b1fb92e85"/>
            <ExternalIdentifier id="doc-ei-1" registryObject="doc-1"
                                identificationScheme="urn:uuid:2e82c1f6-a085-4c72-9da3-8640a32e42ab"
                                value="2.16.840.1.113883.3.9999.3.12">
              <Name><LocalizedString value="XDSDocumentEntry.uniqueId"/></Name>
            </ExternalIdentifier>
          </ExtrinsicObject>
          <Association associationType="urn:oasis:names:tc:ebxml-regrep:AssociationType:HasMember"
                       sourceObject="SubmissionSet01" targetObject="doc-1" id="as-1">
            <Slot name="SubmissionSetStatus"><ValueList><Value>Original</Value></ValueList></Slot>
          </Association>
        </RegistryObjectList>
      </SubmitObjectsRequest>
      <!-- Base64-encoded direct:messageDisposition (failure) - decodes to Example 5.6-3 -->
      <Document id="doc-1">PGRpcmVjdDptZXNzYWdlRGlzcG9zaXRpb24geG1sbnM6ZGlyZWN0PSJ1cm46ZGlyZWN0OmFkZHJlc3NpbmciPg0KICA8ZGlyZWN0Onj
    </ProvideAndRegisterDocumentSetRequest>
  </s:Body>
</s:Envelope>
--MIMEBoundary_dsn_failure_example--
```

Note: This example is for illustrative purposes only. The Document value Base64-decodes to the failure disposition (a reasonForFailure of "Mailbox not found."). The size and hash correspond to that decoded disposition.

Complete Received Message

Message Structure notes that a received message carries the same XDS metadata structure as the **Complete Referral Example** and shows only the incoming address block (**Illustrative Incoming Address Block: Two Recipients Hosted by the Same Receiving System**). The example below shows a complete message as delivered to a receiving customer's endpoint: the receive-direction address block, the a:To addressed to the receiving customer's repository, and a submission set with one document entry. The metadata is abbreviated relative to **Complete Submission: Minimal Metadata With Document Classifications, MTOM (Attachment Payload Omitted) Example**; a received message may carry the full classification set shown there.

Complete Received Message (As Delivered to the Receiving System) Example

Example

```
Content-Type: multipart/related; type="application/xop+xml"; start="<http://tempuri.org/0>"; start-info="application/soap+xml"; bc

--MIMEBoundary_xdr_delivery_example
Content-ID: <http://tempuri.org/0>
Content-Transfer-Encoding: 8bit
Content-Type: application/xop+xml; charset=utf-8; type="application/soap+xml"
MIME-Version: 1.0

<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
            xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">urn:ihe:iti:2007:ProvideAndRegisterDocumentSet-b</a:Action>
    <a:MessageID>urn:uuid:7f4d7c90-d803-4c33-b053-92f77a59c10f</a:MessageID>
    <a:ReplyTo><a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address></a:ReplyTo>
    <a:To s:mustUnderstand="1">https://xdr.example-hospital.com/DocumentRepository_Service.svc</a:To>
    <direct:addressBlock s:role="urn:direct:addressing:destination" s:relay="1"
                         xmlns:direct="urn:direct:addressing">
      <direct:from>dr.smith@direct.example-clinic.com</direct:from>
      <direct:to>intake@direct.example-hospital.com</direct:to>
      <direct:metadata-level>minimal</direct:metadata-level>
    </direct:addressBlock>
  </s:Header>
  <s:Body>
    <ProvideAndRegisterDocumentSetRequest xmlns="urn:ihe:iti:xds-b:2007">
      <SubmitObjectsRequest xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:lcm:3.0">
        <RegistryObjectList xmlns="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0">

          <ExtrinsicObject id="doc-1" mimeType="text/xml"
                           objectType="urn:uuid:7edca82f-054d-47f2-a032-9b2a5b5186c1"
                           status="urn:oasis:names:tc:ebxml-regrep:StatusType:Approved">
            <Slot name="creationTime"><ValueList><Value>20260722200019</Value></ValueList></Slot>
            <Slot name="sourcePatientId">
              <ValueList><Value>PATID1234^^^&amp;1.3.6.1.4.1.99999.1&amp;ISO</Value></ValueList>
            </Slot>
            <Name><LocalizedString value="Cardiology Referral"/></Name>
            <!-- limitedMetadata marker (minimal metadata) -->
            <Classification classifiedObject="doc-1" id="cl-doc-limited"
                            classificationNode="urn:uuid:ab9b591b-83ab-4d03-8f5d-f93b1fb92e85"/>
            <!-- classCode -->
            <Classification classificationScheme="urn:uuid:41a5887f-8865-4c09-adf7-e362475b143a"
                            classifiedObject="doc-1" id="cl-doc-class" nodeRepresentation="57133-1">
              <Slot name="codingScheme"><ValueList><Value>2.16.840.1.113883.6.1</Value></ValueList></Slot>
              <Name><LocalizedString value="Referral Note"/></Name>
            </Classification>
            <!-- DocumentEntry.patientId -->
            <ExternalIdentifier identificationScheme="urn:uuid:58a6f841-87b3-4a3e-92fd-a8ffeff98427"
                                registryObject="doc-1" id="ei-doc-pid"
                                value="PATID1234^^^&amp;1.3.6.1.4.1.99999.1&amp;ISO">
              <Name><LocalizedString value="XDSDocumentEntry.patientId"/></Name>
            </ExternalIdentifier>
            <!-- DocumentEntry.uniqueId -->
            <ExternalIdentifier identificationScheme="urn:uuid:2e82c1f6-a085-4c72-9da3-8640a32e42ab"
                                registryObject="doc-1" id="ei-doc-uid"
                                value="2.16.840.1.113883.3.9999.1.100001">
              <Name><LocalizedString value="XDSDocumentEntry.uniqueId"/></Name>
            </ExternalIdentifier>
```

```
          </ExtrinsicObject>

          <RegistryPackage id="SubmissionSet01"
                           status="urn:oasis:names:tc:ebxml-regrep:StatusType:Approved">
            <Slot name="submissionTime"><ValueList><Value>20260722200019</Value></ValueList></Slot>
            <Slot name="intendedRecipient">
              <ValueList><Value>Example Hospital^^^^^&amp;2.16.840.1.113883.3.9999.3&amp;ISO^^^^0002||^^Internet^intake@direct.exa
            </Slot>
            <Name><LocalizedString value="Cardiology referral"/></Name>
            <!-- limitedMetadata marker (minimal metadata) -->
            <Classification classifiedObject="SubmissionSet01" id="cl-ss-limited"
                            classificationNode="urn:uuid:5003a9db-8d8d-49e6-bf0c-990e34ac7707"/>
            <!-- SubmissionSet.uniqueId -->
            <ExternalIdentifier identificationScheme="urn:uuid:96fdda7c-d067-4183-912e-bf5ee74998a8"
                                registryObject="SubmissionSet01" id="ei-ss-uid"
                                value="2.16.840.1.113883.3.9999.1.1">
              <Name><LocalizedString value="XDSSubmissionSet.uniqueId"/></Name>
            </ExternalIdentifier>
            <!-- SubmissionSet.patientId -->
            <ExternalIdentifier identificationScheme="urn:uuid:6b5aea1a-874d-4603-a4bc-96a0a7b38446"
                                registryObject="SubmissionSet01" id="ei-ss-pid"
                                value="PATID1234^^^&amp;1.3.6.1.4.1.99999.1&amp;ISO">
              <Name><LocalizedString value="XDSSubmissionSet.patientId"/></Name>
            </ExternalIdentifier>
          </RegistryPackage>

          <!-- Classifies the RegistryPackage as a Submission Set -->
          <Classification classifiedObject="SubmissionSet01" id="cl-ss-node"
                          classificationNode="urn:uuid:a54d6aa5-d40d-43f9-88c5-b4633d873bdd"/>

          <!-- The document is a member of the submission set -->
          <Association associationType="urn:oasis:names:tc:ebxml-regrep:AssociationType:HasMember"
                       sourceObject="SubmissionSet01" targetObject="doc-1" id="as-01">
            <Slot name="SubmissionSetStatus"><ValueList><Value>Original</Value></ValueList></Slot>
          </Association>

        </RegistryObjectList>
      </SubmitObjectsRequest>
      <Document id="doc-1">
        <xop:Include href="cid:http://tempuri.org/1" xmlns:xop="http://www.w3.org/2004/08/xop/include"/>
      </Document>
    </ProvideAndRegisterDocumentSetRequest>
  </s:Body>
</s:Envelope>

--MIMEBoundary_xdr_delivery_example
Content-ID: <http://tempuri.org/1>
Content-Transfer-Encoding: binary
Content-Type: text/xml

***** The referral C-CDA document bytes belong here. *****
The binary content of the referral attachment is omitted from this example.
--MIMEBoundary_xdr_delivery_example--
```

Note: This example is for illustrative purposes only. For traceability it reuses the original MessageID; because Surescripts forms the outbound message from the inbound one (**Patient Identification**), the delivered message may carry a different identifier. The receiving system must return a synchronous Provide and Register Document Set-b Response (**Sending the Message Response**) with the same structure as **Synchronous Success Response (No Delivery Notification Tracking)** or **Synchronous Failure Response (RegistryResponse with a RegistryErrorList)**.

11. Implementation Readiness Checklist

Use this checklist to confirm your implementation covers the core capabilities described in this guide before certification. It summarizes requirements and expectations detailed elsewhere; the referenced sections are authoritative.

- [ ] Establish outbound XDR connectivity to Surescripts using mutual TLS (**Connectivity and Mutual TLS (mTLS)**).
- [ ] Receive inbound XDR messages and DSNs, and return valid synchronous Provide and Register Document Set-b responses (**Sending Flow Overview**, **Sending the Message Response**).
- [ ] Construct XDR messages with the required addressing, metadata, and attachments (**Sending XDR Messages to Surescripts**).
- [ ] Track delivery status independently from the synchronous response, using DSNs as the signal of final delivery (**Delivery Status Notification (DSN)**, **Delivery Status Notifications**).
- [ ] Process multiple DSNs for multi-recipient messages, including mixed outcomes (**Multiple Recipients**, **Message Structure**).
- [ ] Publish and maintain your Direct addresses in the Surescripts Directory, and support sending to valid addresses that are not listed (**Receiving Flow Overview**, **Directory Management**).
- [ ] Distinguish SOAP faults from XDR application-level errors, and produce well-formed error responses (**Error Management**).
- [ ] Support the required attachment handling, including display/download and removed-attachment notification (**Clinical Attachments**, **ACR C.205**).
- [ ] Meet the applicable Application Certification Requirements (**Application Certification Requirements**).
- [ ] Complete identity proofing and trust-framework obligations (**DirectTrust Trust Framework and Flow-Down Terms**).

12. Glossary

This section defines acronyms, abbreviations, and technical terms used or referenced throughout this guide.

Term	Definition
Assigning Authority	The organization, identified by an OID, that guarantees an identifier's uniqueness within its namespace; carried in the HL7 CX data type.
CA (Direct Certificate Authority)	An authority that issues and manages Direct certificates. Surescripts is an accredited Direct CA.
C-CDA (Consolidated Clinical Document Architecture)	An HL7 library of templated clinical document formats commonly exchanged through CDM.
CCR (Continuity of Care Record)	A clinical document format summarizing a patient's health information; may be attached to XDR messages.
classCode	An XDS metadata classification giving the high-level class of a document.
Clinical Direct Message	A secure, standards-based electronic message used to exchange patient-related clinical documents between health care participants over the Surescripts network.
Custodian	The party that holds and manages a subscriber's private keys; Surescripts acts as Custodian for its customers' Direct keys.
CX	The HL7 data type representing an identifier together with its assigning authority, for example PATID1234^^^&1.3.6.1.4.1.99999.1&ISO.
Direct Address	An email-like address used in Direct messaging to identify senders and recipients (for example, user@domain.direct.example.com).
DirectTrust	The accreditation body and trust framework that governs Direct exchange among HISPs and maintains the policies referenced in Section 8.
Document Recipient	The IHE ITI-41 role for the system that receives a Provide and Register Document Set-b transaction.
Document Source	The IHE ITI-41 role for the system that initiates a Provide and Register Document Set-b transaction.
DSN (Delivery Status Notification)	An asynchronous message communicating the outcome of an XDR message delivery attempt, indicating success or failure for each recipient.
edge system	A customer's connecting system that sits outside the HISP boundary and exchanges messages with Surescripts on behalf of its end users.
HISP (Health Information Service Provider)	An organization that provides the trust and transport services for Direct messaging. Surescripts is a DirectTrust-accredited HISP.
IHE (Integrating the Healthcare Enterprise)	An initiative that promotes the coordinated use of health IT standards, including XDR.
ITI-41 (Provide and Register Document Set-b)	The IHE IT Infrastructure transaction used in XDR to submit clinical documents and associated metadata.
limitedMetadata	An XDS classification node indicating that a submission set or document was produced under the minimal (limited) metadata profile.
MDN (Message Disposition Notification)	An email-based notification used in SMTP-based Direct messaging. For XDR-based messaging, delivery outcomes are conveyed by a DSN, not an MDN.

Term	Definition
Minimal Metadata	A reduced XDS metadata profile defined by the XDR and XDM specifications, flagged with limitedMetadata classification nodes; the default for most Direct messaging.
Full XDS Metadata	The complete XDS metadata profile, used when integrating with an XDS registry or repository or when otherwise required.
MTOM (Message Transmission Optimization Mechanism)	A W3C mechanism for efficiently transmitting binary content in SOAP messages; used by XDR for document attachments.
mTLS (Mutual Transport Layer Security)	Two-way TLS authentication in which both client and server present and validate certificates (Section 2.5).
OID (Object Identifier)	A globally unique identifier used to identify organizations, systems, and assigning authorities within HL7 and IHE standards.
PHI (Protected Health Information)	Individually identifiable health information as defined under HIPAA.
RA (Direct Registration Authority)	An authority that verifies the identity of organizations and individuals before certificates are issued. Surescripts is an accredited Direct RA.
RegistryError	An element within a RegistryErrorList describing a single application-level error, including an errorCode.
RegistryResponse	The XDR response element indicating success or failure of a Provide and Register Document Set-b transaction.
SOAP (Simple Object Access Protocol)	A messaging protocol for exchanging structured information in web services, used for XDR transactions.
SOAP Fault	An error generated when a SOAP request cannot be processed (for example, authentication or formatting issues).
XDM (Cross-Enterprise Document Media Interchange)	An IHE specification for document exchange using media-based packaging, referenced by the XDR and XDM specification.
XDR (Cross-Enterprise Document Reliable Interchange)	An IHE integration profile for secure, reliable, point-to-point exchange of clinical documents using web services.
XDS (Cross-Enterprise Document Sharing)	An IHE document-sharing standard referenced for metadata handling and full XDS metadata support.

13. Disclaimers

Guide Disclaimer

In the event that the Participant chooses to make any software changes based upon recommendations in this guide, the Participant acknowledges and agrees that Surescripts Network shall bear no responsibility or liability for the Participant's changes or any effects thereof. The Participant shall also be required to transition to the new guide at such time as said guide is published, which may involve different or additional parameters than are published in this guide.

Examples Disclaimer

Examples provided throughout this guide are not intended to be all-inclusive. This pertains to example workflows, element-specific (field) examples, or message examples. Participants should not restrict coding to the examples used herein.

When developing, this guide should be used along with additional documentation referenced and applicable customary coding standards.

Copyright

14. Document Change Log

The table below tracks significant changes made to the document since it was last published.

2026-08-21

Section Title	Change Description
N/A	N/A - Developer Portal - initial release