Ruschlikon
Global (re)insurance
Best Practice Guide ePlacing

A consistent community approach to implementing the

ACORD Global Reinsurance and Large Commercial (GRLC) standards

Issued: March 2026

VERSION: 1.0

Content

1. Introduction

ACORD and Ruschlikon

Faster cash, higher efficiency, better information, enhanced client service – the Ruschlikon Initiative enables leading players of the (re)insurance industry to implement advanced processes for risk placing, technical accounting, claims, and settlement using the ACORD Global Reinsurance and Large Commercial (GRLC) Standards.

Rüschlikon is a small town outside Zürich, Switzerland. Here, a group of pioneering (re)insurers and brokers agreed to a common vision for reducing back office frictional costs and streamlining processes for the (re)insurance sector by implementing global ACORD Data Standards together with an agreed set of business processes and rules. This initiative was named Ruschlikon in honour of its beginning.

ACORD is a not-for-profit standards Organisation that has been working with the Global Reinsurance and Large Commercial Insurance industries to create electronic messaging standards since 2001. Today ACORD continues to maintain and publish these "Global Reinsurance and Large Commercial" (GRLC) standards for the industry and has a close working relationship with implementation communities such as Ruschlikon.

Purpose of the document

This Global (Re)insurance ePlacing Best Practice Guide is designed to support cedents, brokers, and (re)insurers in the implementation of ACORD Global Reinsurance and Large Commercial (GRLC) messages across international markets.

The guide currently focuses on the ePlacing process, providing practical guidance for the build, submission, quotation, order, and post-placement stages, applicable to all lines of business.

Separate best practice documents are available, including:

- Ruschlikon Best Practice Guide (Post-Placement)
- Ruschlikon Best Practice Guide (Accounting and Claims)

All Best Practice Guide can be accessed via the Ruschlikon website: Link

It's important to note that this guide does not replace or override any contractual terms, provisions, or content outlined in insurance or reinsurance agreements. Instead, it provides the framework for exchanging electronic messages in alignment with Ruschlikon best practices.

2. ACORD GRLC Standards

ACORD GRLC standards implementation

The Ruschlikon community, in collaboration with ACORD, has developed a comprehensive set of documents and resources to guide members through the Ruschlikon implementation process.

The main materials are available on the **ACORD GRLC Data Standards** (accessible by ACORD members only) page at: www.acord.org

For more detailed implementation guidance, see the **Ruschlikon e-Deployment Guide.**

ACORD Global Reinsurance and Large Commercial (GRLC) message exchange process

Each of the processes illustrated in the flow diagrams below—Build, Quotation, and Order—consists of multiple Placing Messages exchanged throughout the placement lifecycle. Each message is initiated by a specific event or notification.

Within the Ruschlikon community, it is recommended that the following key placement lifecycle events/notifications be implemented as the core **in-scope items**.

- **Placement Object:** Represents the point at which the sender shares initial placement information without a formal contract structure, enabling early engagement and information exchange.
- **Submission**: Represents the point at which the sender formally submits a contract to initiate the underwriting and quotation process.
- **Decline to Participate:** Indicates that a party does not wish to participate in the placement.
- **Quotation Request:** Represents a request for pricing and terms (i.e., quote) for the risk.
- **Decline to Quote:** Indicates that a party has chosen not to provide a quotation.
- **Quotation:** Represents the provision of pricing and terms in response to a quotation request.
- **Placing Cancelled/NTU**: Indicates that the placement has been withdrawn or marked as Not Taken Up (NTU), and no further placing activity will proceed for the contract.
- **Bindable Quotation Request:** Represents a request for terms that, if accepted, can be bound without further negotiation.
- **Bindable Quotation Response:** Indicates the provision of bindable terms that can be accepted to form a binding contract.
- **Request for Line:** Represents a request for a firm written line on the risk.

- **Decline to Write:** Indicates that a party has declined to provide a written line on the risk.
- **Unconditional Line:** Indicated that the sending party has provided a firm order that may or may not contain conditions. **IF conditions are provided,** the conditions, subjectivities or warranties are specified in the contract wording, **NOT** as structured data within the message. Which would result in a change to the terms of the contract.
- **Acceptance of Condition:** Indicates that the receiving party has formally accepted the stated conditions, allowing the placement to proceed in accordance with the agreed terms.
- **Decline Condition:** Indicates that the receiving party has rejected the stated conditions, preventing the placement from proceeding under the proposed terms.
- **Signed Line Advice**: Confirms the signed line and participation on the risk

As Best Practice, the Ruschlikon community have defined the touchpoints above, and agree that these touchpoints are required for initial readiness. Any additional touchpoints implemented outside the scope above would be brought to the group and discussed.

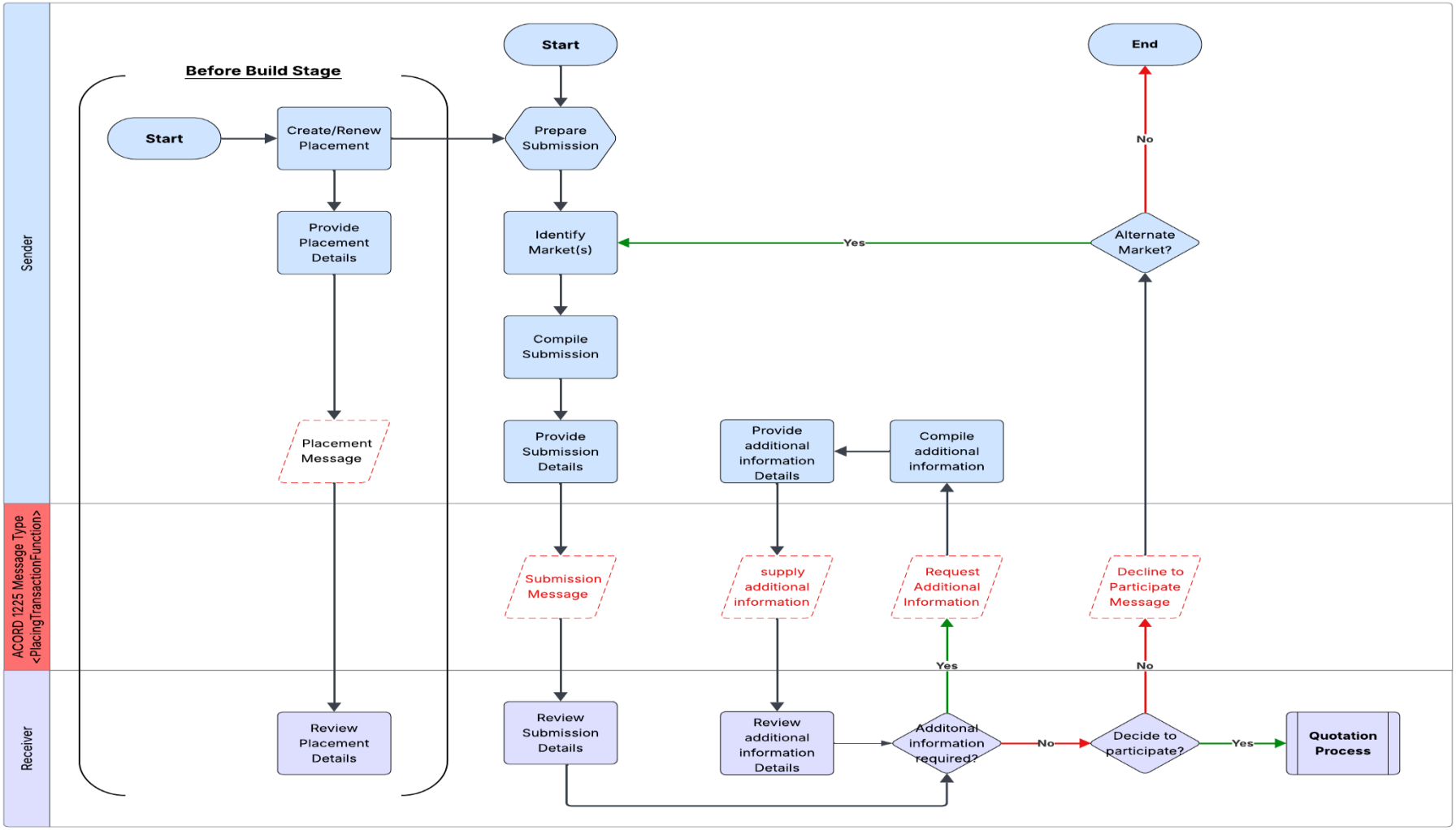
Conditional lines are currently out of scope for the initial implementation phase.

- **Conditional Line:** Indicated that the sending party has provided a firm order that includes conditions, subjectivities, or warranties captured as structured data within the message, resulting in a change to the contract terms.

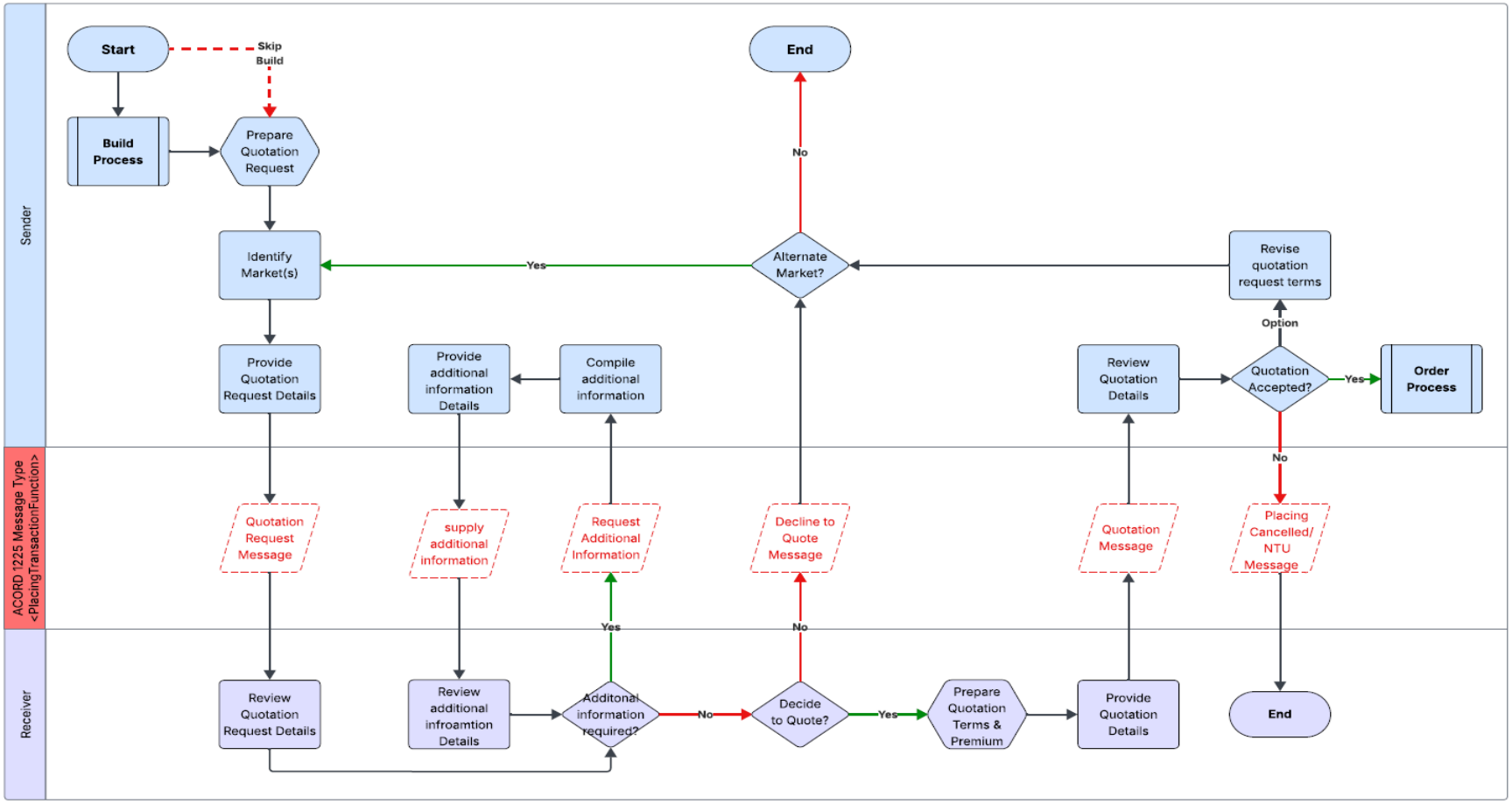
Using all these events/notifications represents the full placing lifecycle encompassed in Phase 1. However, the sequence in which an organisation chooses to begin, such as starting with Submission, Request for Quote, or another stage—will depend on its internal structure and implementation strategy. However, an organisation cannot be considered fully compliant with Phase 1 until all touchpoints have been implemented. Successful implementation of these stages represents the agreed “happy path.”

Placing Stages:

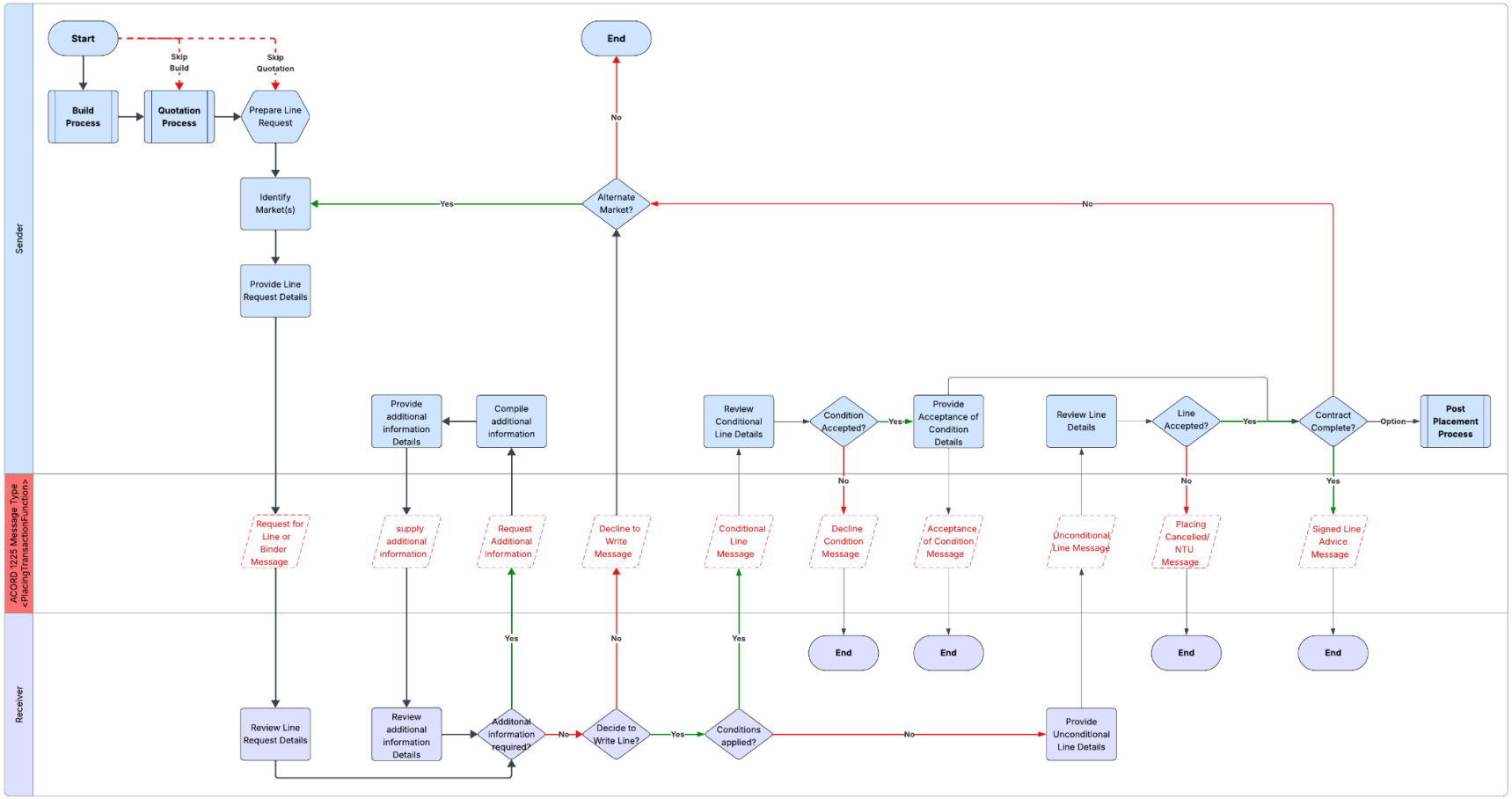
ACORD A83 Placing Stage: Build



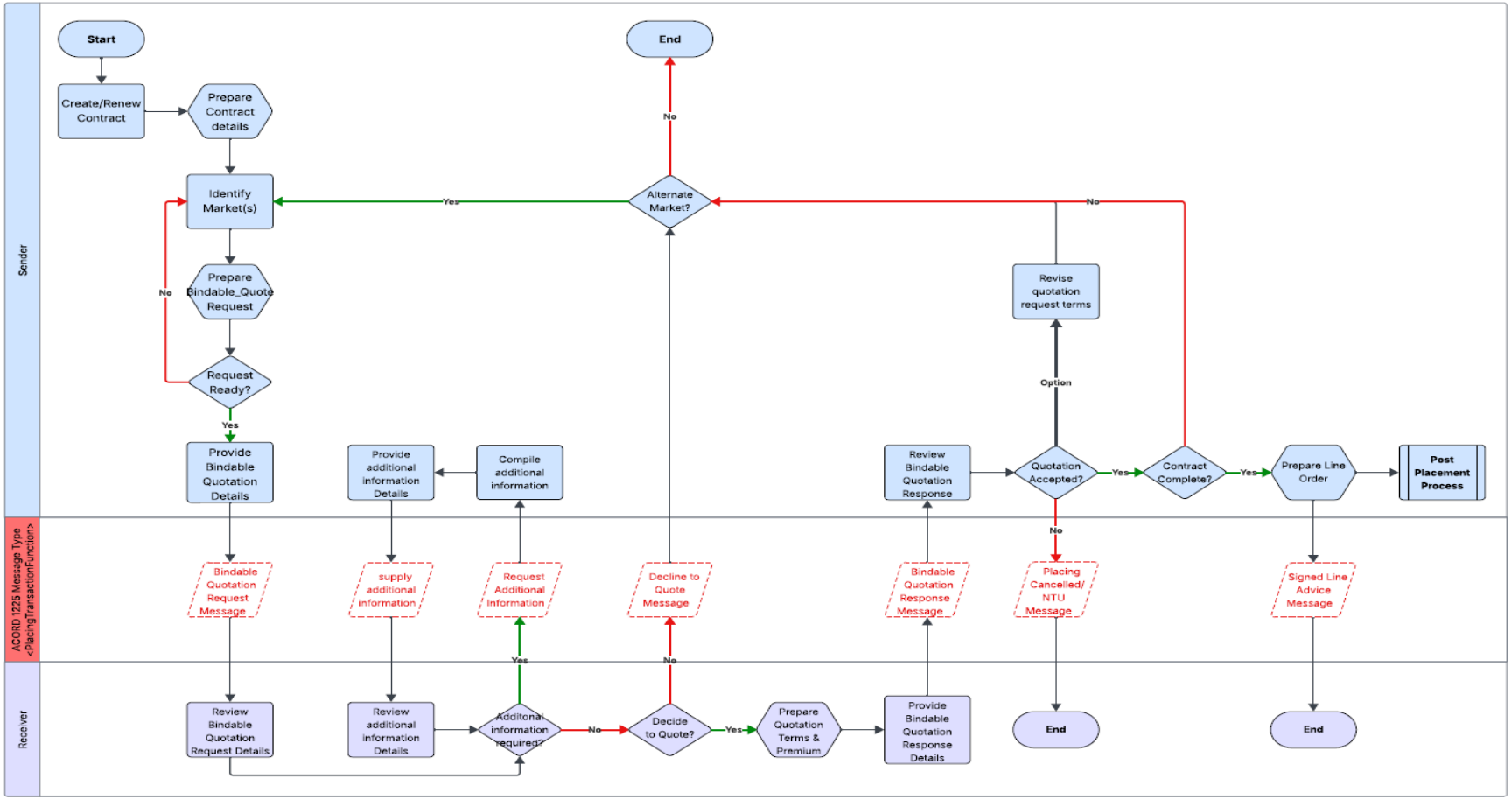
ACORD A83 Placing Stage: Quotation



ACORD A83 Placing Stage: Order



ACORD A83 Placing Stage: Quotation (Bindable Quotation Process)



Validation

Validation plays a critical role in ensuring that data exchanged between systems and members is correctly structured, syntactically accurate, and free from errors. By validating messages against a defined schema and setting common business rules, we can:

- Prevent data-related bugs and integration issues
- Enhance the security and integrity of transmitted information
- Ensure consistency and reliability in communication between stakeholders and systems
- Improve interoperability across different platforms and technologies

To support these goals, ACORD GRLC provides a standardized approach to validations, the full validation approach is accessible via the ACORD GRLC page on www.acord.org.

A distinction has been drawn between application-level validation and business-level validation.

Application Validation (L3 Equivalent):

- Application Validation: Refers to the technical structural integrity of the payload – field presence, format, schema compliance etc. It is a precondition to any processing (Syntax & Schema).
- They ensure structural interoperability across senders and receivers.

Implementation Specific Business Validation:

- Implementation Specific Business Validation: Involves checking logical conditions or underwriting rules that have been agreed as standard by a shared community (e.g. Ruschlikon, London Market Venture specific validations) – Missing reinsurer codes, specific codelist, additional codelist subset etc.
- These form part of the standards

Organisation Specific Business Rules:

- Business Rules: Involves very specific checks and logical conditions that are applicable to individual organisation only. These are not part of the ACORD GRLC Standards, and are additional levels of validation that an organisation may choose to implement to support more effective processing - premium threshold, message sent to the wrong team, incorrect class of business being selected, message sent to the wrong underwriter/broker etc.

This shared approach promotes streamline troubleshooting and ensures that all messages conform to the agreed structure and business rules, thereby reducing the risk of misinterpretation or processing failures. Establishing a common validation framework is especially important in multi-party ecosystems like (re)insurance, where accurate, timely, and reliable data exchange is essential for smooth digital placing and downstream processes.

Summary of above outlines 'Layered Validation Approach':

1. **Application Validation** – Schema and structural check
2. *IF SUCCESSFUL* > **Implementation Specific Business Validation** – Ruschlikon specific additional validation, specific codelist/subsets
3. *IF SUCCESSFUL* > **Organisation Specific Business Rules** – Context sensitive business logic checks - Wrong teams, wrong class of business selected

Document Exchange Standards

The Ruschlikon member community has agreed that document exchange will be facilitated through a DRI (Document Repository Interface) server, which manages the secure storage and retrieval of documents. The DRI server supports a series of standardized message types including Upload Request (Upload Rq), Upload Response (Upload Rs), Notify Request (Notify Rq), Download Request (Download Rq), and Download Response (Download Rs) ensuring consistent, traceable, and interoperable document exchange across all participating parties.

The ACORD message standard and technical solution underpinning this approach are detailed in the Supporting Document, available on the ACORD GRLC page at www.acord.org. This document provides the implementation guidance, message structures, and integration requirements for enabling compliant DRI-based document exchange within the market.

Adherence to the guidelines and best practices

The Ruschlikon member community is committed to ensuring consistency and interoperability across all participants, the data fields used in these messages are categorized into four types of conditions:

- **Mandatory** – These fields must always be populated.
- **Conditional/Mandatory** – These fields must be populated when specific conditions are met (e.g., certain transaction types or business scenarios).
- **Recommendation** – These fields are strongly advised to be used, as they improve processing efficiency, clarity, or downstream integration.
- **Optional** – These fields may be used at the discretion of the sender and/or receiver, depending on business needs.

No additional data elements beyond those defined in the agreed standard are permitted. If a new data item is required, it must be raised with ACORD through the **standard enhancement request process**.

This process ensures that proposed additions are reviewed and agreed upon by the Ruschlikon ePlacing Business Implementation group before being incorporated into the official specification.

This structured approach ensures:

- Consistent message structures across all parties
- Clear expectations for data provision and consumption
- Controlled governance for any changes or enhancements to the standard

Extensions

The Ruschlikon member community has agreed to include an **"Extension"** section within the JSON message structure. This section enables senders to provide additional data attributes that are unique to an individual member, intended for use in a **peer-to-peer communication** context.

While the use of extensions has been agreed, any data attributes contained within them **do not form part of the GRLC Standards** and are therefore **not subject to validation** under the ACORD framework.

As these extensions fall outside the scope of ACORD governance, they are managed entirely between the parties involved. It is the sole responsibility of those parties to ensure their use of the extension does not conflict with or compromise the processing of the standardized data set.

This approach provides participants with the flexibility to meet specific business, or market needs while ensuring integrity**, interoperability, and predictability** of the core ACORD GRLC Standards remain intact.

3. Implementation Recommendations

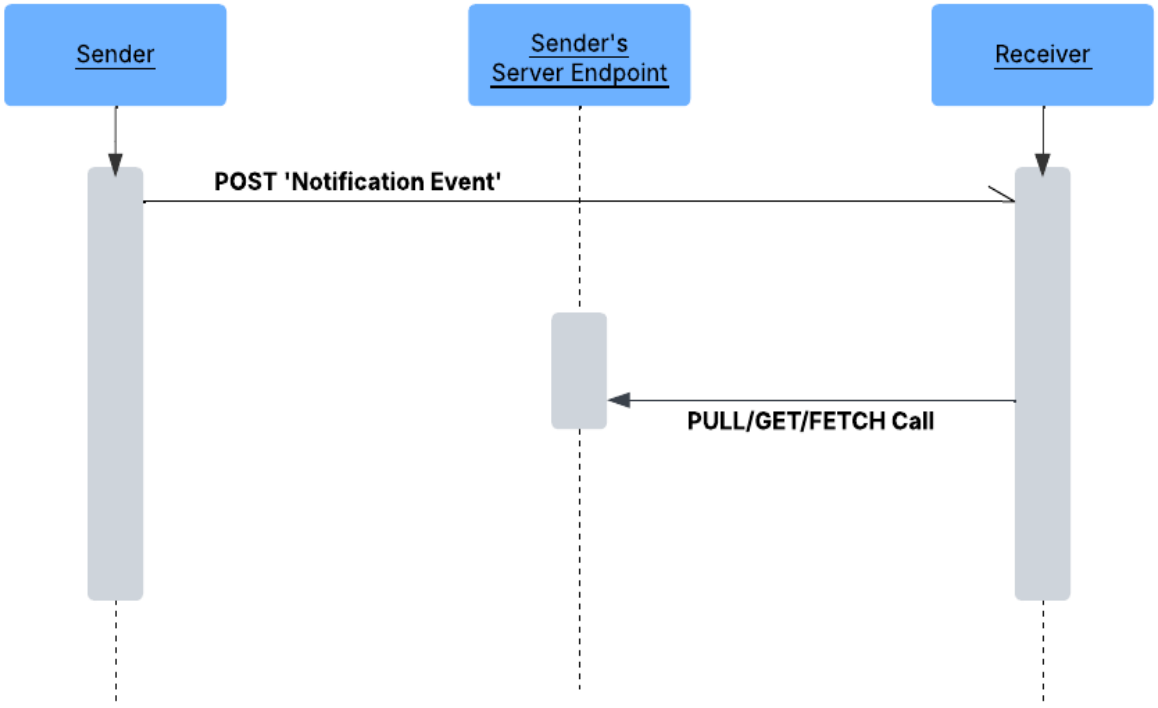
Notify/Pull API Implementation Approach

The agreed implementation approach within the Ruschlikon member community is the **Notify-Pull API Architecture**. This approach is not part of the GRLC Standards and has been specifically adopted by the Ruschlikon community. However, this is the recommended approach and the members of Ruschlikon agree to follow the best practices.

This model is commonly used in B2B integrations where decoupling of data generation and consumption is desirable for scalability, reliability, or security reasons.

Under this model:

- The **sender** is able to send a lightweight notification (e.g. message in queues, status flag, or small metadata call).
- The **receiver** can receive and process the notification/event.
- The **receiver** is then able to make a PULL/GET/FETCH API call to retrieve the full JSON payload from a server endpoint.
- The **receiver** can parse the data and decide what to do with it next in their internal system



Benefits of a Notify/Pull APIs Approach

1. **Control:** Each organization determines when and how often to retrieve data, based on their processing capacity and internal logic.
2. **Simplicity:** The model is easy to implement, test, and maintains using existing REST infrastructure, with no need to manage inbound traffic.
3. **Load Management:** Clients pull data when ready, avoiding overloads and respecting rate limits.
4. **Retry Logic:** Failed or timed-out requests can simply be retried.
5. **Secure Retrieval:** Data is securely retrieved from a protected endpoint and processed according to business rules.

Notification Best Practice Guidance:

The following guidance outlines recommended behaviours, controls, and technical considerations for implementing notification-based messaging within a Notify-Pull API architecture.

Retrieval (GET/PULL/FETCH) SLA Recommendations:

- The **preferred** target is to pull the message **immediately**, ideally within a few *seconds/minutes* of receiving the notification, where systems and business processes allow.
- A maximum window of up to **24 hours** may be tolerated as the outer boundary to support considerations such as batch processes, time zone difference & overnight operations.
- An acknowledgment mechanism should be implemented to confirm whether the retrieval was successful. Should the sender not receive a 200 success or 400 error acknowledgment within the 24 hours tolerance period, they may then suspend or stop the message being eligible for pulling. Communication should then be processed with each implementor to resolve any outstanding issues.

Retry Frequency and Retry Window Recommendations

- Receivers should implement controlled retry logic to manage transient failures and prevent unnecessary load.
- Recommended approach: **maximum of 5 retries within a 10–30-minute window**.
    - First 2 retries – within a 10-minute window
    - Next 3 – 4 retries within a 2-hour window
    - 5th retry – within 24 hours (triage and communicate)
- Retry behaviour should avoid excessive polling or unnecessary load on either system (i.e., back-to-back retirees).
- After maximum retries have been exhausted without success, then a technical failure should be logged, and parties should follow their internal out of band communication process (e.g., direct email or call) to resolve the issue and coordinate next steps.

Handling Failed or Rejected Messages (Queue Behaviour)

- A dedicated dead-letter queue (DLQ) is **currently not a primary design requirement**.
- Under the current model, rejected messages would remain in the queue and would simply **not be pulled** into the receiver's system.
- If a message is **rejected or fails**, the **sender** is expected to correct the issue and issue a **new notification**.
- Once a message is **successfully accepted**, the sender should remove it from the queue (or ensure the queue marks it as processed).

General Target Operating Model

- Each organisation must ensure that its internal TOM is prepared to accommodate any outage.
- This aligns with each organisation broader **TOM (Target Operating Model)** considerations.
  Not all exceptions can be resolved solely through system logic.
  Human communication and operational processes remain essential for reconciling missed or stuck messages.

Notification Data Structure:

A Notification message serves to inform a recipient that new or updated information is available for retrieval. It typically includes key metadata such as a *Message ID, Timestamp, and Resource URI*, enabling traceability, auditing, and efficient event-driven communication.

The Ruschlikon member community has agreed on the following standardized **Notification message structure**, ensuring consistency and interoperability across implementations:

Attribute	Pathway
NotificationType:	notificationType
NotificationID:	notificationID
PlacementUUID:	placementID
ContractUUID:	contractID
ContractSectionUUID:	contractSectionID
[Sender]NotifyingPartner	
NotifyingRoleCode	notifyingPartner.roleCode
PartyIdentification:	notifyingPartner.partyIdentities[].typeid
PartyIdentificationAgency:	notifyingPartner.partyIdentities[].typeCode
PartyName:	notifyingPartner.name.fullName
PartyContactDescription:	notifyingPartner.contacts[].description
PartyContactEmailAddress:	notifyingPartner.contacts[].communication.emailAddress
PartyContactTelephoneNumber:	notifyingPartner.contacts[].communication.phoneNumber
PartyAddressNumberAndStreet:	notifyingPartner.address.line1
PartyAddressCityName:	notifyingPartner.address.city
PartyAddressCountrySubEntity:	notifyingPartner.address.subEntityCode
PartyAddressCountry:	notifyingPartner.address.countryCode
PartyAddressZipOrPostCode:	notifyingPartner.address.postalCode
PartyContactPersonName:	notifyingPartner.contacts[].name
[Receiver] ToBeNotified	
NotifyingRoleCode	toBeNotified.roleCode
PartyIdentification:	toBeNotified.partyIdentities[].typeid
PartyIdentificationAgency:	toBeNotified.partyIdentities[].typeCode
PartyName:	toBeNotified.name.fullName
PartyContactDescription:	toBeNotified.contacts[].description
PartyContactEmailAddress:	toBeNotified.contacts[].communication.emailAddress
PartyContactTelephoneNumber:	toBeNotified.contacts[].communication.phoneNumber
PartyAddressNumberAndStreet:	toBeNotified.address.line1
PartyAddressCityName:	toBeNotified.address.city
PartyAddressCountrySubEntity:	toBeNotified.address.subEntityCode
PartyAddressCountry:	toBeNotified.address.countryCode
PartyAddressZipOrPostCode:	toBeNotified.address.postalCode
PartyContactPersonName:	toBeNotified.contacts[].name

This structure provides a clear and consistent framework for implementing notifications, ensuring that systems across the market can identify, route, and process event data efficiently and reliably.

Notification Type

The Notification Type defines the purpose or nature of the notification being sent. Each type indicates a specific event or action within the placement or contract lifecycle.

The Ruschlikon member community has agreed on the following permitted Notification Types within the notification message:

Notification Type(s)	Definition
new_placement:	Indicates the creation of a new or renewed placement record.
new_contract:	Notifies the recipient of a newly created contract, section or submission.
updated_contract:	Signals that an existing contract, section or submission has been amended or updated.
end_placing:	Indicates that the placing process has either been formally completed or cancelled and no further placement activity is expected for the risk.
decline_to_participate:	Indicate the point in which the underwriter does not wish to participate on the placement, contract, section anymore.
new_document:	Informs that a new document is available for retrieval.
updated_document:	Indicates that an existing document has been modified or replaced.
quote_request:	Represents a request to quote for a particular section.
decline_to_quote:	Indicates that the sender has declined to provide a quotation.
Quotation:	Communicates a quotation provided in response to a request.
bindable_quotation_request:	Indicates a request for terms that, if accepted, can be immediately bound without further negotiation.
bindable_quotation:	Indicates an offer being made in response to a bindable quotation request, that if accepted, can immediately form a binding contract.

Notification Type(s)	Definition
request_for_line_or_binder:	Requests line input or binder information from the recipient.
unconditional_line_offer:	Indicates an offer or order being made based on prior quotation with or without any conditions (**IF conditions are provided,** the conditions, subjectivities or warranties are specified in the contract wording)
conditional_line_offer:	Indicates an offer or order being made based on prior quotation with conditions that require structural changes to the contract terms (currently out of scope).
decline_to_write:	Communicates that an order has been declined.
signed_line_advice:	Confirms a signed line or binding agreement.
post_placement:	Bridges the gap between placement and accounting by identifying what should be accounted for and how.
Endorsement:	(to be defined)

These standardized notification types enable consistent event handling, ensuring all parties interpret and respond to notifications in a uniform manner across implementations.

Placing Stage vs. Notification Type(s)

ACORD A83: Placing Stage	ACORD 1225: PlacingTransactionFunction		Notification Type(s)
	Original Message	*Response Message*	
N/A:	Placement Message >		*new_placement*
Build:	Submission Light >	Decline to Participate	*new_contract; updated_contract; decline_to_participate*
	Submission >	Decline to Participate	*new_contract; updated_contract; decline_to_participate*
Quotation:	Quotation Request >	Decline to Quote	*quote_request; decline_to_quote;*
	Quotation Request >	Quotation	*quote_request; quotation*
	Quotation >	Placing Cancelled/NTU	*quotation; end_placing*
	Bindable Quotation Request >	Decline to Quote	*bindable_quotation_request; decline_to_quote*
	Bindable Quotation Request >	Bindable Quotation Response	*bindable_quotation_request; bindable_quotation_response*
	Bindable Quotation Response >	Placing Cancelled/NTU	*bindable_quotation_response; end_placing*
	Bindable Quotation Response >	Signed Line Advice	*bindable_quotation_response; signed_line_advice*
Order:	Request for Line >	Decline to Write	*request_for_line_or_binder; decline_to_write*
	Request for Line >	Conditional Line	*request_for_line_or_binder; conditional_line_offer*
	Request for Line >	Unconditional Line	*request_for_line_or_binder; unconditional_line_offer*
	Unconditional Line >	Placing Cancelled/NTU	*unconditional_line_offer; end_placing*
	Conditional Lines >		*Currently out of scope*
	Signed Line Advice >		*signed_line_advice,*
Build, Quotation & Order:	Request Additional Information >	Supply Additional Information	*new_document; updated_document*
Post Placement:	Pre Accounting Advice >		*post_placement*

ACORD JSON Schema Versioning

As the JSON schema evolves and new updates are introduced, robust version management is essential to maintain consistency and interoperability across implementations, while ensuring that the latest schema version is referenced when transmitting messages. A commonly adopted industry practice is **URL path versioning**, where the version number is embedded directly within the endpoint URL.

Example Endpoints:

POST: https://reinsurer.com/eplacing/1.3/notification
POST: https://reinsurer.com/eplacing/1.4/notification
POST: https://reinsurer.com/eplacing/1.5/notification

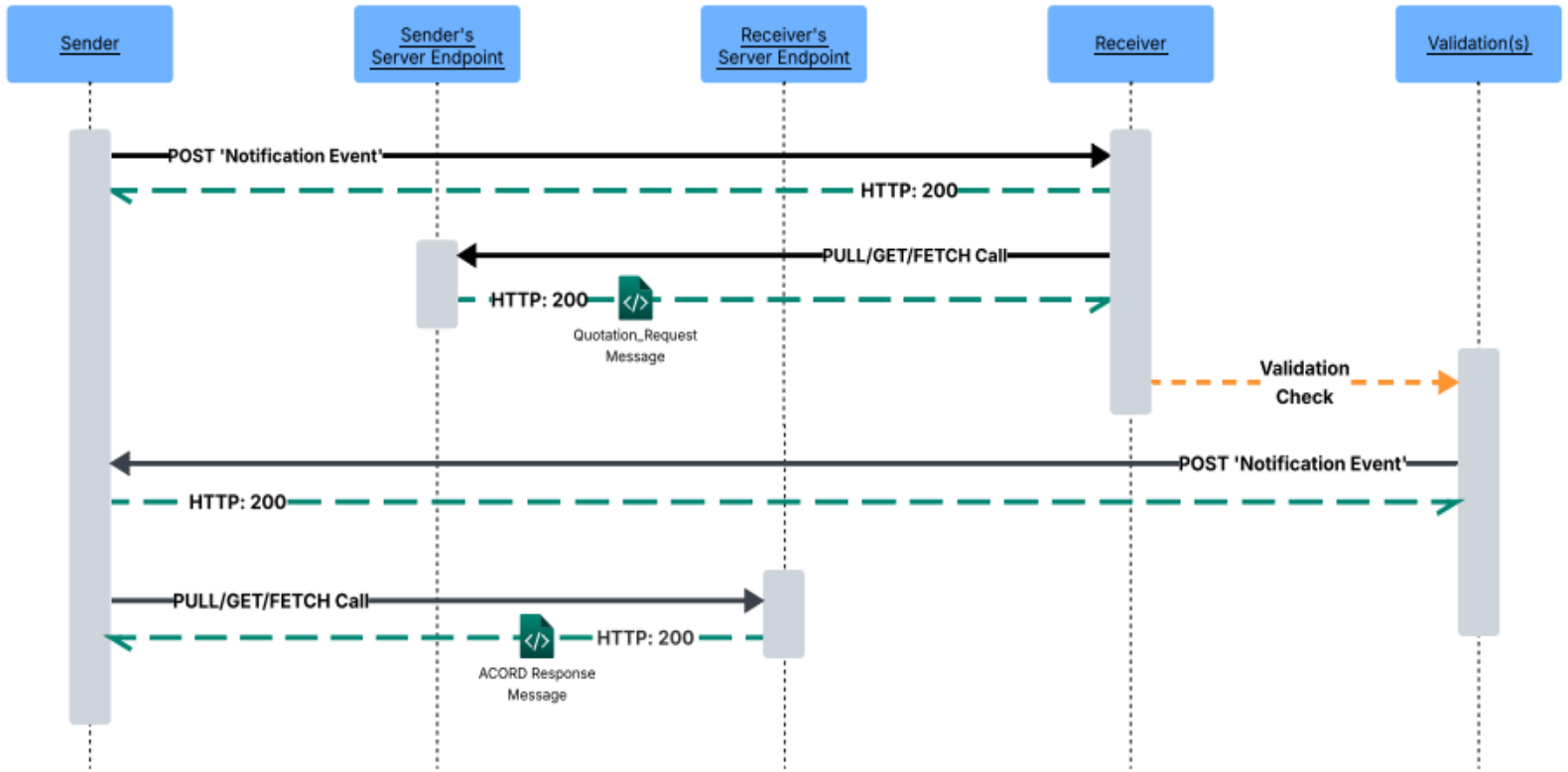
URL Path Versioning:

- This method makes it easy to test, identify, and differentiate between API versions, as the version number is clearly visible in the endpoint path. It allows multiple versions to coexist, supporting smooth transitions and backward compatibility during upgrades.
- However, depending on the underlying architecture, this approach can also increase implementation and maintenance complexity — particularly when managing version-specific routing, validation, and documentation across environments.
- In practice, the chosen versioning strategy should balance clarity, maintainability, and long-term interoperability, ensuring consistent communication across systems while minimising disruption during version updates.

Response Message Flow

The ACORD message standard and technical solution underpinning this approach are detailed in the Response Message, available on the ACORD GRLC page at www.acord.org. This message provides the implementation guidance, structure, and integration requirements necessary to enable compliant DRI-based document exchange within the market.

The Ruschlikon community have agreed on a Notify-Pull API approach, which also applies to the associated response messages.



Response Message Process

The Ruschlikon community has outlined a clear distinction between a technical response and an ACORD GRLC message response.

Technical Response:

- Technical acknowledgement response typically uses HTTP 200 (success) or HTTP 400 (failure).
- Any validation checks **must not** occur within the technical response
- In the event of a HTTP 400 failure, the technical acknowledgement response **does not** include any detailed reason for the failure

- If a technical acknowledgement returns a HTTP 400-level error, the organisation should follow the ***"Retry Frequency and Retry Window Recommendations"*** outlined above.
- If the HTTP 400 error persists beyond the 24-hour tolerance window and no additional information has been received, communication should move outside of the system. At that point, parties should use their internal out-of-band communication channels (e.g., direct email or phone call) to resolve the issue and agree on next steps.

ACORD GRLC Response:

- ACORD GRLC Response is completed once the receiver has successfully PULLED/FETCHED/GET the ePlacing message call
- ACORD GRLC Response would provide a status of 'acknowledged', 'rejected' or 'pending'
- In the event of a rejection, the response **MUST** include detailed reason for rejections and the failed validations
- The response (acknowledgement) message flow is supported by the ACORD GRLC standards and technical materials available to members on the ACORD website.

4. Ruschlikon Guidelines

Security and Confidentiality

Global Data Projection Regulation (GDPR) compliance does not stem from the data standards themselves, but from how those standards are implemented. Organisations must ensure that any methods used to store or transmit data comply with GDPR requirements. It is strongly recommended that data exchange is carried out using established web security protocols.

External Certificate Authorities will be used to authenticate senders and receivers when exchanging messages in production between business partners. Any messages transmitted over a public network must be encrypted, with transport-level encryption (such as SSL) considered the minimum requirement. All message content is treated as confidential and is intended solely for supporting reporting obligations in accordance with the terms of the reinsurance contracts.

Interoperability amongst trading partners

Implementers can develop and maintain a globally compliant ACORD messaging and process capability, enabling the deployment of a single operational model across all current and future trading partners. Since interoperability relies on the functions provided by the ACORD standards, trading partners are strongly encouraged to participate in discussions on ongoing standard enhancements—either through the Ruschlikon ePlacing SPG/BIG or by engaging with the relevant Ruschlikon Regional Implementation Group(s).

Ruschlikon Business Implementation Group ePlacing: BIG- ePlacing
Regional Implementation Groups: Regional Implementation Groups

Company internal system changes/upgrade

If a sender or receiver needs to upgrade their internal systems or datasets for reasons unrelated to a messaging update—such as a new system release or the introduction of new data fields to maintain compliance with the Ruschlikon Global (Re)insurance Best Practice Guide—the connected partners should collaborate to thoroughly test the changes and assess any impact on messaging.

To allow sufficient time for planning and adjustments, all affected partners must be notified of such changes in a timely manner.

Partner Identifier

Within the Ruschlikon community, DUNS is the established method for identifying partners where applicable. While it is the preferred identifier, it is not the only option—if a DUNS number is not available (for example, for a cedent or original policyholder), another appropriate unique identifier should be used.

The DUNS number was selected as the standard unique identifier for message exchange across the Ruschlikon community. As a general principle, each business entity should have its own DUNS code to support seamless message transfer wherever possible.

Industry partners are encouraged to proactively notify their counterparts if administrative responsibilities are transferred from one entity to another, ensuring sufficient time for preparation and testing.

About Dun & Bradstreet: The Dun & Bradstreet D-U-N-S® Number is a unique nine-digit identifier linked to a company's business identity. It is used globally to help organisations verify partners, pursue contracts, apply for financing, and more. Existing DUNS numbers can be searched or requested via: https://www.dnb.com/duns-number/lookup.html, or through a local Dun & Bradstreet representative office.

Business Continuity and Disaster Recovery

Each business partner is responsible for its own disaster recovery process to ensure there is no significant interruption of message exchange.

The infrastructure of the trading partner is expected to work 24/7. Exceptions are scheduled maintenance windows or unexpected service interruptions. Whilst the maintenance windows will be communicated to trading partners in a timely fashion, unexpected service interruption must be solved as soon as possible.

In the event of a critical situation within the sender's or receiver's production system, it may be necessary for the sender or receiver to request an emergency change to messaging without the usual lead time for production changes. In this situation, the business partners should communicate the changes, and plan in appropriate testing, as soon as possible to ensure the change does not impact other conditions within the messaging environment.

As part of the implementation process, contact details of counterparts on sender's and receiver's side are exchanged.

Companies shall define and agree on SLA's. Those SLA's are in the sole responsibility of the connecting partners. Generally if the system is not working business resumes via email.

5. Concluding remarks

There are clear business benefits to participating in the Ruschlikon initiative and the ePlacing community. Ruschlikon is a committed network of brokers, carriers, (re)insurers, and their vendors, all working to reduce frictional costs and streamline processes across the (re)insurance sector by implementing global ACORD Data Standards alongside an agreed set of business processes and rules.

For assistance, please contact contact@ruschlikon.com or reach out to your nearest Regional Implementation Group chair(s): Regional Implementation Groups (acord.org).

Useful Resources

- Login (acord.org)
- What is Ruschlikon?
- How does Ruschlikon achieve these benefits
- Ruschlikon Case Studies
- Link to Ruschlikon Adoption Directory (RAD)
- Regional Implementation Groups (acord.org)
- Engaging with Ruschlikon – Selecting Technology Options and vendors
- Ruschlikon-edeployment-guide_may-2024-1.pdf (acord.org)
- Best Practice Guide (Post-Placement) v 1.0 (acord.org)

This Page was Intentionally Left Blank