# Confirmation Flow

Confirmation of a request begins with a `requestConfirmation` message and ends with either a confirmed request (such as `confirmationAgreed`) or a retraction of the request via a `requestConfirmationRetracted` message.

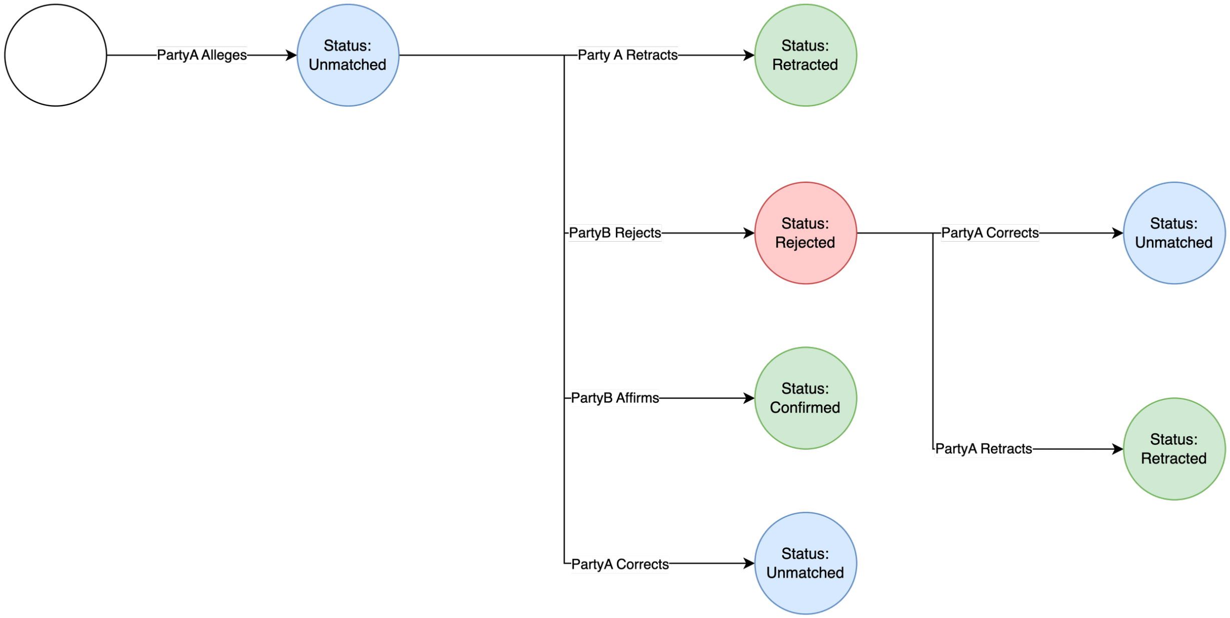
Confirmation can be achieved through one of the following processes:

- **Affirmation** - the counterparty responds by accepting the alleging party's request
- **Matching** - the counterparty responds with a request of their own; confirmation happens only if the terms of the two requests match

## Confirmation Flow from Party A's Perspective

The confirmation flow begins with the alleging party submitting a `requestConfirmation`, at which point the request has a status of `Unmatched`. Both parties are notified and can perform certain actions:

- The alleging party, Party A, can retract their request or correct it.
  - If they retract the request, the request lifecycle is complete.
  - If they correct the request, it proceeds in `Unmatched` status.
- Party B can reject or affirm the request.
  - If they reject the request, Party A must decide whether to correct or retract the request.
  - If they affirm the request, it moves into a `Confirmed` status, which completes the confirmation lifecycle.



## Message Flow

The following message types are used throughout the confirmation lifecycle:

| Instruction Message Type | Produced Client Notifications | Key Attributes |
|---|---|---|
| `RequestConfirmation`<br>(new trade or correction) | • `ConfirmationAcknowledgement`<br>• `ConfirmationStatus:Alleged` | `sentBy`, `partyTradeId`, `messageId`<br>- required for identifying the request for |

| Instruction Message Type | Produced Client Notifications | Key Attributes |
|---|---|---|
| | • ConfirmationStatus:Unmatched | further processing |
| ConfirmationRetracted | • ConfirmationAcknowledgement<br>• ConfirmationStatus (both parties) | sentBy, partyTradeId - used to identify the correct trade to retract (does not require Stellar ID) |
| ConfirmationDisputed | • ConfirmationAcknowledgement<br>• ConfirmationDisputed | • sentBy, partyTradeId - used to identify the correct trade to reject (requires Stellar ID to correctly identify the trade in the Ledger)<br>• messageId (from confirmationStatus:Alleged ) - must be in the inReplyTo field |
| ConfirmationAgreed | • ConfirmationAcknowledgement<br>• ConfirmationAgreed (both parties, with the relevant private data) | |

# Error Codes

Users are notified of errors during message flow management using the Exception FpML message type's `messageRejected` element and the following error codes.

Code series legend:

- `100x` - general errors
- `200x` - errors related to external services
- `400x` - validation errors
    - `401x` - errors where the message is missing an element or a key element cannot be evaluated
    - `402x` - errors resulting from entitlement validation issues
    - `403x` - state validation errors
- `500x` - errors on post-trade lifecycle events
- `600x` - business validation errors

| Error Code | Feature | Description | Comments |
|---|---|---|---|
| `1001` | Trade Capture - Affirmation | Unknown Error | Generic error that we default to when we do not know what the error is |

| Error Code | Feature | Description | Comments |
|---|---|---|---|
| `2001` | Confirmation | An external service required is not available | Generic error code to indicate an issue with an integration with external components |
| `2010` | Member Services API | Member services API is not reachable | Occurs when we try to reach the member services endpoint but get no response |
| `2011` | Member Services API | This Executing Unit does not exist | Occurs when the Confirmation Service is able to reach the API endpoint but does not get a response back with data for the submitted executing unit |
| `2012` | Member Services API | This Executing Unit is not active | This EU is present in the Member Services API response but is marked not active; this validation will not be necessary, as inactive EUs will be removed from the participant |

| Error Code | Feature | Description | Comments |
|---|---|---|---|
| | | | management interface |
| `2021` | Member Services API | This Entitlement Value does not match the Executing Unit | The entitlement value was not found in the available entitlement values for the executing unit being looked up |
| `2022` | Trade Capture | The Entitlement Value is missing | |
| `2023` | Trade Capture | The counterparty Entitlement Value is not valid | |
| `3001` | Trade Capture - Affirmation | The `requestConfirmation` message submitted did not pass FPML schema validation | Error code `300x` is an extension of the `300` validation error code suggested in the FpML spec |
| `3002` | Trade Capture - Affirmation | The `requestConfirmation` referenced does not exist | Occurs when sending a non-`requestConfirmation` command that references a `requestConfirmation` that does not exist |

| Error Code | Feature | Description | Comments |
|---|---|---|---|
| 3010 | Trade Capture - Affirmation | This trade version is not current | |
| 3020 | Trade Capture - Affirmation | The requestConfirmationRetracted message could not be processed because the counter-party already responded to the requestConfirmation message | |
| 3021 | Trade Capture - Affirmation | The confirmationDisputed message could not be processed because the counter-party already retracted the alleged trade | |
| 4008 | Trade Capture | The Stellar ID provided does not match the internal partyTradeIdentifier of the referenced request | |