XFall Arbeitsgruppe

XFall REST Webservice

Spezifikation

- ENTWURF -

Stand 2018-09-27

Historie

Datum	Bearbeiter	Änderung
2017-06-29	Andreas Huber, FJD	Dokument erstellt
2018-09-27	Andreas Huber, FJD	Betriebsmodi beschrieben

Inhaltsverzeichnis

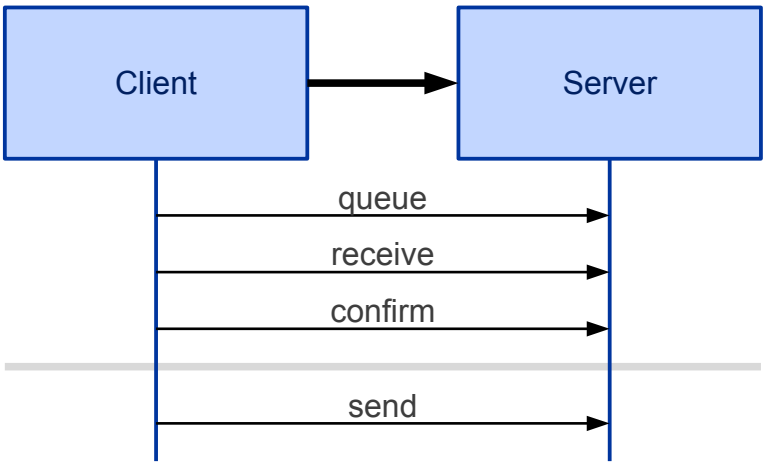
1 Einleitung

1.1 Änderungen gegenüber dem XFall-Queue-Webservice (SOAP)

- Zusätzliche Informationen in der QueueResponse: lokale Message-ID, XFall-Version und Leistung
- ReceiveResponse kann nur eine Nachricht enthalten
- Commit-Funktion zur Bestätigung der Abholung
- ReceiveFileRespoinse liefert nur die Datei, nicht die DocumentRepresentation, die bereits im Container enthalten war
- Die clientID im QueueRequest ist optional und wird in einem HTTP-Header übertragen. Alternativ kann die Client-ID auch in der Basisadresse enthalten sein.
- Die Funktion „send“ sendet nur noch Nachrichten ohne Transaktion. Das Absenden von Nachrichten mit Transaktion übernimmt die Funktion „commitTransaction“.
- Der Betriebsmodus „Push“ wurde hinzugefügt

2 Konzept

2.1 Betriebsmodus



Der Webservice kann auf zwei unterschiedliche Weisen eingesetzt werden, die hier als „Betriebsmodus“ bezeichnet werden. Diese sind:

- Queue-Modus
- Push-Modus

Beide Modi stehen jeweils in einer einfachen Variante und mit Transaktionen zur Verfügung. In der einfachen Variante müssen, falls vorhanden, alle Anlagen in den XFall-Container eingebetettet werden. Da sie dabei base64-codiert werden vergrößert sich die benötigte Bandbreite um 1/3. Soll also eine XFall-Nachricht mit einer 3MB großen Anlage übermittelt werden, so wird die resultierende Nachricht c.a. 4MB groß. Zu diesem Zweck können Transaktionen eingesetzt werden. In diesem Fall werden alle Anlagen und die Nachricht als getrennte Aufrufe übermittelt.

2.1.1 Queue-Betriebsmodus

Beim Queue-Modus erfolgen alle Zugriffe vom Client zum Server. Daher ist dieser anzuwenden, wenn der Client vom Server nicht zu erreichen ist, z.B. weil es sich um ein Fachverfahren in einem geschlossenen Behördennetz handelt.

2.1.1.1 Ohne Transaktionen

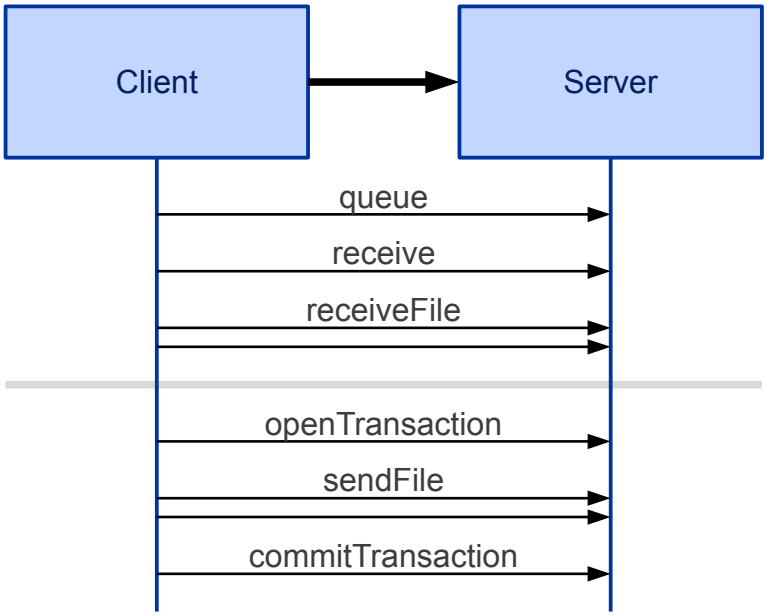
Bei der ersten Kontaktaufnahme legt der Server die Nachricht in den Postkorb und wartet, bis der Client sie abholt.

Der Client fragt regelmäßig mit der Funktion „**queue**“ seinen Postkorb ab.

Erhält er die Information, dass eine Nachricht vorliegt, so holt er sie mit der Funktion „**receive**“ ab.

Die Antwort wird mit der Funktion „**send**“ übermittelt.

2.1.1.2 Mit Transaktionen



Der Client fragt regelmäßig mit der Funktion „**queue**“ seinen Postkorb ab.

Erhält er die Information, dass eine Nachricht vorliegt, so holt er sie mit der Funktion „**receive**“ ab.

In dem übermittelten XFall-Container befindet sich eine Liste von Dokumenten. Diese holt er einzeln über die Funktion „**receiveFile**“ ab.

Um eine Antwort zu senden muss der Client zuerst mit der Funkion „**openTransaction**“ eine Transaktion eröffnen. Er erhält vom Server eine Transaktionskennung, die er bei den folgenden Aufrufen als Referenz angibt.

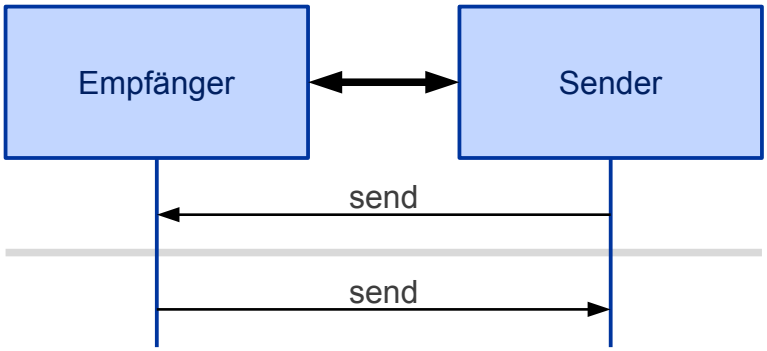
Im nächsten Schritt übermittelt er alle Anlagen mit der Funktion „**sendFile**“.

Zum Abschluss wird die eigentliche Nachricht mit der Funktion „**commitTransaction**“ übermittelt und die Transaktion damit abgeschlossen.

2.1.2 Push-Betriebsmodus

Im Push-Modus erfolgen die Zugriffe in beide Richtungen. Aufgrund der unmittelbaren Zustellung ist dieser Modus vorzuziehen, sofern eine bidirektionale Erreichbarkeit sichergestellt werden kann.

2.1.2.1 Ohne Transaktionen



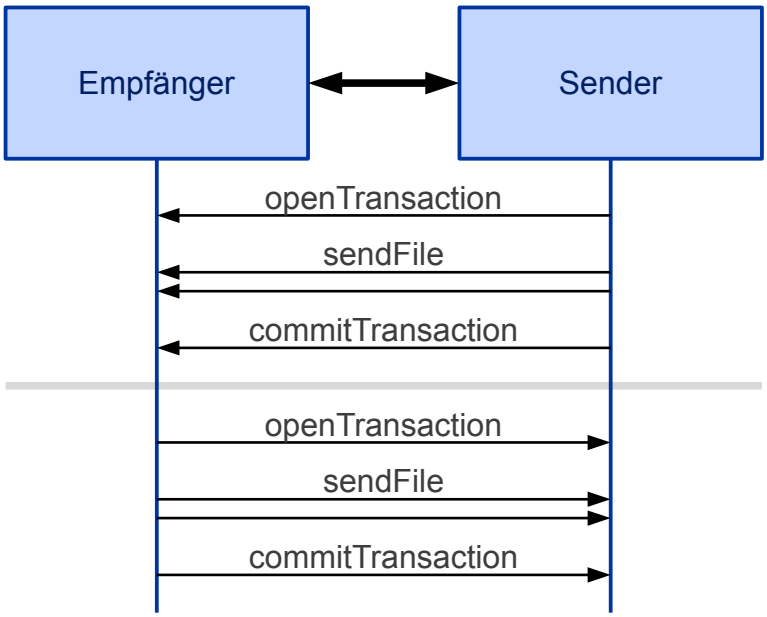
Der Sender übermittelt die Nachricht an den Empfänger mit der Funktion „**send**“.

Genauso sendet der Empfänger die Antwort mit der Funktion „**send**“.

2.1.2.2 Mit Transaktionen

Sowohl bei der Übermittlung der Nachricht wie auch der Antwort wird erst eine Transaktion eröffnet („**openTransaction**“), um dann die Anlagen („**sendFile**“) zu übermitteln.

Zum Abschluss wird die eigentliche Nachricht bzw. Antwort mit der Funktion „**commitTransaction**“ übermittelt und die Transaktion damit abgeschlossen.



3 API

Bei allen Zugriffen muss eine Authentisierung erfolgen. Entweder über „HTTP Basic Authentication“ oder einen API-Key im Header.

Es wird immer XML ausgetauscht.

3.1 Queue

3.1.1 GET /msg – Abfrage der Queue

Funktion „queue“

Request Header:

- X-Fall-Client-ID [0:1]

Request Parameter:

- fromDate [0:1]
- searchCriterion [0:1]
- searchLocation [0:*]

Response:

- queue
  - message [0:*]
    - id [1] – Die lokale ID der Nachricht in der Queue, um sie über die API zu adressieren
    - mainApplicationID [0:1]
    - mainApplicationTitle [0:1]
    - partialApplicationID [1]
    - partialApplicationTitle [1]
    - xfall [0:1]
      - message [1]
      - version [1]
    - publicService [0:1]
      - uri [1]
      - description [0:1]

3.1.2 GET /msg/{id} – Nachricht abholen

Funktion „receive“

Request: (leer)

Response: XFall-Nachricht

3.1.3 POST /msg/{id} – Abholung der Nachricht bestätigen

Funktion „confirm“

Request:

- update
  - status
    - {status} =
      - queued – in der Queue
      - received – abgeholt
      - validated – überprüft
      - finished – beendet

```
<update>
  <status>received</status>
</update>
```

Response: (leer)

3.1.4 GET /msg/{id}/{file} – Datei abholen

Funktion „receiveFile“

Request: (leer)

Response: Datei

3.2 Nachrichtenübermittlung

3.2.1 PUT /msg – Versenden von Nachrichten ohne Transaktion

Funktion „send“

Request: XFall-Nachricht

Response: XFall-Nachricht (Antwort)

3.3 Transaktionen

3.3.1 PUT /tx – Transaktion anlegen

Funktion „openTransaction“

Request:

- openTransaction
  - partialApplicationID

Response:

- transaction
  - id
    - {txid}

```
<transaction>
  <id>{txid}</id>
</transaction>
```

3.3.2 PUT /tx/{txid}/{file} – Datei hochladen

Funktion „sendFile“

Request: Datei

Response: (leer)

3.3.3 POST /tx/{txid} – Transaktion commiten

Funktion „commitTransaction“

Request: XFall-Nachricht

Response: XFall-Nachricht (Antwort)