

# Analisi dei Requisiti

2024-02-03 — v1.1.0



[overture.unipd@gmail.com](mailto:overture.unipd@gmail.com)

Destinatari	Prof. Tullio Vardanega Prof. Riccardo Cardin <i>Zextras</i> Gruppo <i>Overture</i>
Responsabile	Alex Vedovato
Redattori	Eleonora Amadori Michele Bettin Riccardo Bonavigo Francesco Costantino Bulychov Riccardo Fabbian Francesco Furno Alex Vedovato
Verificatori	Eleonora Amadori Michele Bettin Riccardo Bonavigo Francesco Costantino Bulychov Riccardo Fabbian Francesco Furno Alex Vedovato

## Registro delle modifiche

Versione	Data	Autori	Verificatori	Dettaglio
1.1.0	2024-02-03	Riccardo Fabbian	Riccardo Bonavigo	Modifiche conseguenti ai commenti derivanti dalla revisione RTB
1.0.1	2024-02-03	Eleonora Amadori	Riccardo Bonavigo	Sistemati i riferimenti ai documenti esterni
1.0.0	2024-01-13	Francesco Furno	Riccardo Bonavigo	Approvazione per RTB
0.3.0	2024-01-07	Riccardo Bonavigo, Alex Vedovato	Riccardo Fabbian	Rimozione di tutto ciò che riguarda contatti, rubriche, calendari ed appuntamenti
0.2.2	2024-01-05	Alex Vedovato	Riccardo Fabbian	Correzioni generali ai requisiti
0.2.1	2024-01-02	Riccardo Bonavigo	Riccardo Fabbian	Ripristinati i requisiti di vincolo
0.2.0	2023-12-30	Francesco Furno	Riccardo Fabbian	Casi d'uso: aggiornati i diagrammi UML
0.1.3	2023-12-28	Alex Vedovato	Riccardo Fabbian	Casi d'uso: approfonditi maggiormente i casi d'uso basandosi sulle specifiche di JMAP
0.1.2	2023-12-23	Francesco Furno	Riccardo Fabbian	Casi d'uso: aggiunti alcuni errori mancanti presi dalle specifiche di JMAP
0.1.1	2023-12-17	Eleonora Amadori, Riccardo Bonavigo, Riccardo Fabbian, Alex Vedovato	Francesco Furno	Correzioni alle sezioni di introduzione Correzioni alle sezioni di descrizione generale del prodotto Casi d'uso: eliminazione degli stress test Aggiunte le caption ad immagini e tabelle
0.1.0	2023-12-14	Riccardo Bonavigo, Riccardo Fabbian, Alex Vedovato	Francesco Costantino Bulychov	Casi d'uso: ridefinizione degli attori Casi d'uso: approfondita la gestione degli errori Casi d'uso: separati gli errori provenienti da condition differenti
0.0.9	2023-12-02	Francesco Furno	Alex Vedovato	Rimozione dei requisiti di vincolo Rimozione dei requisiti di prestazione
0.0.8	2023-12-02	Michele Bettin	Alex Vedovato	Casi d'uso: correzione degli stress test
0.0.7	2023-12-01	Eleonora Amadori	Michele Bettin	Casi d'uso: correzione creazione delle cartelle Casi d'uso: correzione gestione contenuti delle cartelle

0.0.6	2023-12-01	Francesco Costantino Bulychov	Michele Bettin	Casi d'uso: correzione login base Casi d'uso: correzione scambio delle email
0.0.5	2023-11-25	Michele Bettin	Alex Vedovato	Produzione completa dei diagrammi UML dei casi d'uso. Stesura testuale dei diagrammi dei casi d'uso. Prima bozza stesura sezione requisiti: nomenclatura e lista.
0.0.4	2023-11-21	Francesco Costantino Bulychov	Alex Vedovato	Prima bozza dei diagrammi UML dei casi d'uso.
0.0.3	2023-11-13	Michele Bettin	Eleonora Amadori	Obiettivo prodotto. Funzionalità del prodotto. Caratteristiche utente. Tecnologie e struttura di progetto.
0.0.2	2023-11-11	Francesco Costantino Bulychov	Eleonora Amadori	Scopo del prodotto. Glossario. Riferimenti.
0.0.1	2023-11-10	Michele Bettin	Eleonora Amadori	Struttura di base del documento e prima parte introduzione.

## Indice

<b>1) Introduzione .....</b>	9
1.1) Scopo del documento .....	9
1.2) Scopo del prodotto .....	10
1.2.1) Spiegazione dello scenario di riferimento .....	10
1.2.2) Il protocollo JMAP .....	10
1.2.3) Cosa richiede <i>Zextras</i> ? .....	10
1.3) Glossario .....	11
1.4) Riferimenti .....	11
1.4.1) Riferimenti normativi .....	11
1.4.2) Riferimenti informativi .....	11
<b>2) Descrizione generale del prodotto richiesto da <i>Zextras</i> .....</b>	12
2.1) Obiettivo fissato .....	12
2.2) Funzionalità del prodotto e requisiti .....	12
2.3) Caratteristiche utente .....	12
2.3.1) Utenti di Carbonio .....	12
2.3.2) Utenti del nostro prodotto .....	12
2.4) Tecnologie e analisi della struttura di progetto .....	13
<b>3) Casi d'uso .....</b>	14
3.1) Obiettivi .....	14
3.2) Introduzione ai casi d'uso .....	14
3.3) Attori .....	14
3.3.1) Alcuni dettagli sui client .....	14
3.4) Gestione degli errori .....	15
3.5) Elenco dei casi d'uso .....	16
3.5.1) UC1 - Autenticazione .....	16
3.5.2) UC2 - Autenticazione fallita .....	17
3.5.3) UC3 - Ottenimento della risorsa JMAP Session .....	18
3.5.4) UC4 - Errore “unknownCapability” .....	18
3.5.5) UC5 - Errore “notJSON” .....	18
3.5.6) UC6 - Errore “notRequest” .....	19
3.5.7) UC7 - Errore “limit” .....	19
3.5.8) UC8 - Errore “serverUnavailable” .....	19
3.5.9) UC9 - Errore “serverFail” .....	20
3.5.10) UC10 - Errore “serverPartialFail” .....	20
3.5.11) UC11 - Errore “unknownMethod” .....	21
3.5.12) UC12 - Errore “invalidArguments” .....	21
3.5.13) UC13 - Errore “invalidResultReference” .....	21
3.5.14) UC14 - Errore “forbidden” .....	22
3.5.15) UC15 - Errore “accountNotFound” .....	22
3.5.16) UC16 - Errore “accountNotSupportedByMethod” .....	22
3.5.17) UC17 - Errore “accountReadOnly” .....	23
3.5.18) UC18 - Errore “cannotCalculateChanges” .....	23
3.5.19) UC19 - Errore “overQuota” .....	24
3.5.20) UC20 - Errore “notFound” .....	24
3.5.21) UC21 - Errore “willDestroy” .....	24
3.5.22) UC22 - Errore “tooLarge” .....	25
3.5.23) UC23 - Errore “rateLimit” .....	25

---

3.5.24) UC24 - Errore "invalidPatch" .....	25
3.5.25) UC25 - Errore "invalidProperties" .....	26
3.5.26) UC26 - Errore "singleton" .....	26
3.5.27) UC27 - Errore "requestTooLarge" .....	26
3.5.28) UC28 - Errore "stateMismatch" .....	27
3.5.29) UC29 - Errore "blobNotFound" .....	27
3.5.30) UC30 - Errore "tooManyKeywords" .....	28
3.5.31) UC31 - Errore "tooManyMailboxes" .....	28
3.5.32) UC32 - Errore "alreadyExists" .....	28
3.5.33) UC33 - Errore "fromAccountNotFound" .....	29
3.5.34) UC34 - Errore "fromAccountNotSupportedByMethod" .....	29
3.5.35) UC35 - Errore "anchorNotFound" .....	29
3.5.36) UC36 - Errore "unsupportedSort" .....	30
3.5.37) UC37 - Errore "unsupportedFilter" .....	30
3.5.38) UC38 - Errore "tooManyChanges" .....	31
3.5.39) UC39 - Errore "mailboxHasChild" .....	31
3.5.40) UC40 - Errore "mailboxHasEmail" .....	31
3.5.41) UC41 - Errore "invalidEmail" .....	32
3.5.42) UC42 - Errore "tooManyRecipients" .....	32
3.5.43) UC43 - Errore "noRecipients" .....	32
3.5.44) UC44 - Errore "invalidRecipients" .....	33
3.5.45) UC45 - Errore "forbiddenMailFrom" .....	33
3.5.46) UC46 - Errore "forbiddenFrom" .....	34
3.5.47) UC47 - Errore "forbiddenToSend" .....	34
3.5.48) UC48 - Invio email .....	35
3.5.49) UC49 - Ricezione email .....	51
3.5.50) UC50 - Eliminazione email .....	58
3.5.51) UC51 - Ricezione cartella .....	63
3.5.52) UC52 - Creazione cartella .....	68
3.5.53) UC53 - Modifica cartella .....	75
3.5.54) UC54 - Eliminazione cartella .....	82
3.5.55) UC55 - Gestione contenuti cartella .....	88
3.5.56) UC56 - Creazione condivisione cartella .....	98
3.5.57) UC57 - Modifica principale .....	110
3.5.58) UC58 - Eliminazione principale .....	117
3.5.59) UC59 - Modifica/Eliminazione condivisione cartella .....	123
3.5.60) UC60 - Sincronizzazione email .....	130
3.5.61) UC61 - Sincronizzazione cartelle .....	135
4) Requisiti .....	140
4.1) Requisiti di funzionalità .....	140
4.2) Requisiti di qualità .....	154
4.3) Requisiti di vincolo .....	155
5) Tracciamento Requisiti .....	156
5.1) Riepilogo .....	160
5.2) Conclusioni .....	160

## **Lista della immagini**

Figura 1: UC1 - Autenticazione .....	16
Figura 2: UC3 - Ottenimento della risorsa JMAP Session .....	17
Figura 3: UC48 - Invio email .....	35
Figura 4: Sottocasi UC48 - Invio email .....	37
Figura 5: Sottocasi UC48.2 - Costruzione delle chiamate di metodo necessarie all'invio di una email ...	
38	
Figura 6: Sottocasi UC48.2.1 - Inserimento dei parametri del metodo "Email/set" necessari alla creazione di una email .....	39
Figura 7: Sottocasi UC48.2.1.3 - Inserimento del parametro "create" del metodo "Email/set" necessario alla creazione di una email .....	41
Figura 8: Sottocasi UC48.2.2 - Inserimento dei parametri del metodo "EmailSubmission/set" necessari all'invio di una email .....	46
Figura 9: Sottocasi UC48.2.2.3 - Inserimento del parametro "create" del metodo "EmailSubmission/set" necessario all'invio di una email .....	48
Figura 10: UC49 - Ricezione email .....	51
Figura 11: UC49 - Sottocasi ricezione email .....	52
Figura 12: Sottocasi UC49.2 - Costruzione delle chiamate di metodo necessarie alla ricezione di una email .....	53
Figura 13: Sottocasi UC49.2.1 - Inserimento dei parametri del metodo "Email/get" necessari alla ricezione di una email .....	54
Figura 14: UC50 - Eliminazione email .....	58
Figura 15: Sottocasi UC50 - Eliminazione email .....	59
Figura 16: Sottocasi UC50.2 - Costruzione delle chiamate di metodo necessarie all'eliminazione di una email .....	60
Figura 17: Sottocasi UC50.2.1 - Inserimento dei parametri del metodo "Email/set" necessari all'eliminazione di una email .....	61
Figura 18: UC51 - Ricezione cartella .....	63
Figura 19: Sottocasi UC51 - Ricezione cartella .....	64
Figura 20: Sottocasi UC51.2 - Costruzione delle chiamate di metodo necessarie alla ricezione di una cartella .....	65
Figura 21: Sottocasi UC51.2.1 - Inserimento dei parametri del metodo "Mailbox/get" necessari alla ricezione di una cartella .....	66
Figura 22: UC52 - Creazione cartella .....	68
Figura 23: Sottocasi UC52 - Creazione cartella .....	69
Figura 24: Sottocasi UC52.2 - Costruzione delle chiamate di metodo necessarie alla creazione di una cartella .....	70
Figura 25: Sottocasi UC52.2.1 - Inserimento dei parametri del metodo "Mailbox/set" necessari alla creazione di una cartella .....	71
Figura 26: Sottocasi UC52.2.1.3 - Inserimento del parametro "create" del metodo "Mailbox/set" necessario alla creazione di una cartella .....	73
Figura 27: UC53 - Modifica cartella .....	75
Figura 28: Sottocasi UC53 - Modifica cartella .....	76
Figura 29: Sottocasi UC53.2 - Costruzione delle chiamate di metodo necessarie alla modifica di una cartella .....	77
Figura 30: Sottocasi UC53.2.1 - Inserimento dei parametri del metodo "Mailbox/set" necessari alla modifica di una cartella .....	78
Figura 31: Sottocasi UC53.2.1.4 - Inserimento del parametro "update" del metodo "Mailbox/set"	

---

necessario alla modifica di una cartella .....	80
Figura 32: UC54 - Eliminazione cartella .....	82
Figura 33: Sottocasi UC54 - Eliminazione cartella .....	83
Figura 34: Sottocasi UC54.2 - Costruzione delle chiamate di metodo necessarie all'eliminazione di una cartella .....	84
Figura 35: Sottocasi UC54.2.1 - Inserimento dei parametri del metodo "Mailbox/set" necessari all'eliminazione di una cartella .....	85
Figura 36: UC55 - Gestione contenuti cartella .....	88
Figura 37: Sottocasi UC55 - Gestione contenuti cartella .....	90
Figura 38: Sottocasi UC55.2 - Costruzione delle chiamate di metodo necessarie alla gestione dei contenuti di una cartella .....	91
Figura 39: Sottocasi UC55.2.1 - Inserimento dei parametri del metodo "Email/set" necessari alla gestione dei contenuti di una cartella .....	92
Figura 40: Sottocasi UC55.2.1.4 - Inserimento del parametro "update" del metodo "Email/set" necessario alla gestione dei contenuti di una cartella .....	94
Figura 41: UC56 - Creazione condivisione cartella .....	98
Figura 42: Sottocasi UC56 - creazione condivisione cartella .....	99
Figura 43: Sottocasi UC56.2 - Costruzione delle chiamate di metodo necessarie alla creazione della condivisione di una cartella .....	100
Figura 44: Sottocasi UC56.2.1 - Inserimento dei parametri del metodo "Principal/set" necessari alla creazione della condivisione di una cartella .....	102
Figura 45: Sottocasi UC56.2.1.3 - Inserimento del parametro "create" del metodo "Principal/set" necessario alla creazione della condivisione di una cartella .....	104
Figura 46: Sottocasi UC56.2.2 - Inserimento dei parametri del metodo "Mailbox/set" necessari alla creazione della condivisione di una cartella .....	106
Figura 47: Sottocasi UC56.2.2.4 - Inserimento del parametro "update" del metodo "Mailbox/set" necessario alla creazione della condivisione di una cartella .....	108
Figura 48: UC57 - Modifica principale .....	110
Figura 49: Sottocasi UC57 - Modifica principale .....	111
Figura 50: Sottocasi UC57.2 - Costruzione delle chiamate di metodo necessarie alla modifica di un principale .....	112
Figura 51: Sottocasi UC57.2.1 - Inserimento dei parametri del metodo "Principal/set" necessari alla modifica di un principale .....	113
Figura 52: Sottocasi UC57.2.1.4 - Inserimento del parametro "update" del metodo "Principal/set" necessario alla modifica di un principale .....	115
Figura 53: UC58 - Eliminazione principale .....	117
Figura 54: Sottocasi UC58 - Eliminazione principale .....	118
Figura 55: Sottocasi UC58.2 - Costruzione delle chiamate di metodo necessarie all'eliminazione di un principale .....	119
Figura 56: Sottocasi UC58.2.1 - Inserimento dei parametri del metodo "Principal/set" necessari all'eliminazione di un principale .....	120
Figura 57: UC59 - Modifica/Eliminazione condivisione cartella .....	123
Figura 58: Sottocasi UC59 - Modifica/eliminazione condivisione cartella .....	124
Figura 59: Sottocasi UC59.2 - Costruzione delle chiamate di metodo necessarie alla modifica/eliminazione della condivisione di una cartella .....	125
Figura 60: Sottocasi UC59.2.1 - Inserimento dei parametri del metodo "Mailbox/set" necessari alla modifica/eliminazione della condivisione di una cartella .....	126
Figura 61: Sottocasi UC59.2.1.4 - Inserimento del parametro "update" del metodo "Mailbox/set" necessario alla modifica/eliminazione della condivisione di una cartella .....	128

Figura 62: UC60 - Sincronizzazione email .....	130
Figura 63: Sottocasi UC60 - Sincronizzazione email .....	131
Figura 64: Sottocasi UC60.2 - Costruzione delle chiamate di metodo necessarie alla sincronizzazione delle email .....	132
Figura 65: Sottocasi UC60.2.1 - Inserimento dei parametri del metodo “Email/changes” necessari alla sincronizzazione delle email .....	133
Figura 66: UC61 - Sincronizzazione cartelle .....	135
Figura 67: Sottocasi UC61 - Sincronizzazione cartelle .....	136
Figura 68: Sottocasi UC61.2 - Costruzione delle chiamate di metodo necessarie alla sincronizzazione delle cartelle .....	137
Figura 69: Sottocasi UC60.2.1 - Inserimento dei parametri del metodo “Mailbox/changes” necessari alla sincronizzazione delle cartelle .....	138

## **Lista delle tabelle**

Tabella 1: requisiti di funzionalità .....	140
Tabella 2: requisiti di qualità .....	154
Tabella 3: requisiti di vincolo .....	155
Tabella 4: tracciamento requisiti .....	156
Tabella 5: riepilogo .....	160

## 1) Introduzione

### 1.1) Scopo del documento

L'Analisi dei Requisiti è un documento fondamentale per tutti i progetti di sviluppo software che vogliono essere in ottemperanza con gli standard di *qualità* definiti dalla materia di Ingegneria del Software.

Lo scopo del documento è quello di definire le *funzionalità* che il sistema sarà in grado di offrire, ovvero i requisiti obbligatori, desiderati e opzionali che devono essere soddisfatti dal prodotto sviluppato al fine di essere conforme alle richieste fatte dal *proponente*.

L'analisi non deve fornire la soluzione al problema, ma deve essere consapevole della sua fattibilità tecnologica (definendo di conseguenza un confine tra analisi del problema e il suo design, quindi la soluzione). In particolare, le finalità di questo documento possono essere definite e sintetizzate nei seguenti punti:

- **Definire le esigenze dei proponenti:**

Il documento di Analisi dei Requisiti fonda le sue basi sulle esigenze del proponente, ovvero le sue aspettative in merito al prodotto software che dovremmo andare a sviluppare. Queste richieste verranno raccolte tramite i documenti forniti dal proponente *Zextras* e dalle varie interviste svolte durante lo svolgimento del progetto.

- **Tracciare i requisiti del sistema:**

Una volta raccolte le esigenze del proponente, il documento di Analisi dei Requisiti deve identificare tutti i casi d'uso necessari e i corrispettivi requisiti associati ad ognuno di essi suddividendoli in funzionali, qualitativi e di vincolo.

- **Verificazione e validazione dei requisiti:**

Il *processo* di *verifica dei requisiti* ha lo scopo di garantire che le attività siano svolte in modo corretto, senza errori, ponendo in primo piano il Way of Working del gruppo.

La *validazione dei requisiti* invece consiste nel accertare che il prodotto corrisponda alle attese, ponendo attenzione al prodotto software finale.

- **Fornire una base per la progettazione del sistema:**

Il documento di Analisi dei Requisiti fornisce una base per la progettazione del sistema, in quanto definisce le funzionalità che il sistema deve offrire. I programmatorei possono utilizzare il documento per comprendere le esigenze del proponente e identificare le soluzioni più appropriate per soddisfare tali esigenze.

- **Minimizzare i rischi e ottimizzare i costi:**

Un documento di Analisi dei Requisiti completo e accurato può aiutare a ridurre i rischi del progetto e quantificare al meglio i suoi costi. Ciò è dovuto al fatto che il documento aiuta a garantire che i requisiti siano effettivamente corretti e completi, evitando così errori e ritardi nello sviluppo del sistema.

Arrivati al punto in cui si ha una chiara visione dello scopo e delle funzionalità del prodotto, dei suoi requisiti e degli attori del sistema software, in questo documento se ne darà una formale rappresentazione grafica utilizzando i diagrammi dei casi d'uso.

## 1.2) Scopo del prodotto

### 1.2.1) Spiegazione dello scenario di riferimento

Il proponente del capitolato d'appalto C8 è l'azienda vicentina *Zextras*. Il core business di *Zextras* si incentra sulla vendita del loro prodotto di punta: Carbonio. Carbonio può essere definito come una suite di funzionalità volte a incrementare la produttività di un'organizzazione o team che si affida a questo prodotto.

Infatti, questa soluzione software permette di raggruppare insieme un pool di utenti offrendo loro una serie di strumenti fondamentali per la collaborazione in un contesto aziendale, come:

1. **Messaggistica avanzata:** fornendo un tool di posta elettronica moderna, veloce e intuitiva, profili multipli, account e cartelle condivise, anteprima degli allegati e strumenti di gestione come tag e filtri;
2. Calendari ed eventi pianificati;
3. Videochiamate e chat;
4. Editing collaborativo;
5. Gestione file condivisi.

Questo applicativo è implementato con un software backend (installabile solamente su un Server Ubuntu) e i client possono usufruire di questa soluzione comodamente tramite browser previo apposito login.

### 1.2.2) Il protocollo JMAP

Tra tutte le funzionalità offerte da Carbonio, *Zextras* pone particolare attenzione, nel suo capitolato, sulla funzionalità di messaggistica che utilizza come base per la sua implementazione la posta elettronica. Questa tecnologia utilizza IMAP (Internet Message Access Protocol), un protocollo di comunicazione utilizzato per accedere alle email conservate su un server di posta compatibile. Risale al lontano 1986, come successore del precedente protocollo POP (Post Office Protocol) e consente un accesso più avanzato e interattivo al contenuto delle caselle di posta elettronica, ad esempio permettendo la sincronizzazione di una casella di posta in più dispositivi.

Negli ultimi anni però è nato JMAP: un interessante alternativa a IMAP, il cui pregio principale rispetto a IMAP è la velocità, in quanto sfrutta il formato JSON (JavaScript Object Notation) e definisce un set di API per permettere ai client di accedere ai dati e gestirli in modo efficiente.

### 1.2.3) Cosa richiede *Zextras*?

*Zextras* non ci chiede direttamente di apporre modifiche a Carbonio, ma di implementare un sistema informatico capace di sfruttare il protocollo JMAP per l'invio di email tra diversi utenti. In particolare dovremmo andare a sviluppare un **server mail** che metta a disposizione a un qualsiasi tipo di **client** (capace di supportare il protocollo JMAP ovviamente) una serie di funzionalità (che detteremo in seguito con la lista dei requisiti).

A questo punto, utilizzando uno dei client open source reperibili nelle repository pubbliche, dovremmo già essere in grado di sfruttare questa tecnologia di posta elettronica grazie al server che abbiamo implementato.

Il reale scopo di *Zextras* però non è tanto quello di ricevere il know how che noi andremo ad acquisire programmando il server, ma quello di ricevere i risultati dei benchmark che andremo a realizzare sul sistema rapportandolo alle specifiche hardware e software dell'impianto utilizzato.

Basandosi su questi risultati *Zextras* deciderà quindi se è vantaggioso apportare delle modifiche a Carbonio e implementare un sistema di posta elettronica che sfrutti il protocollo JMAP come quello sviluppato nella nostra demo o se abbandonare l'idea.

## 1.3) Glossario

Per evitare ambiguità o incomprensioni riguardanti la terminologia usata nel documento, è stato deciso di adottare un glossario in cui vengono riportate le varie definizioni. In questa maniera in esso verranno riportati tutti i termini specifici del dominio d'uso con relativi significati.

La presenza di un termine all'interno del **Glossario** viene indicata applicando *questo stile*.

## 1.4) Riferimenti

### 1.4.1) Riferimenti normativi

- Norme di Progetto v1.0.0:  
[https://overture-unipd.github.io/docs/rtb/interni/norme\\_di\\_progetto\\_v1.0.0.pdf](https://overture-unipd.github.io/docs/rtb/interni/norme_di_progetto_v1.0.0.pdf)
- Capitolato d'appalto C8: JMAP, il nuovo protocollo standard per la comunicazione email  
<https://www.math.unipd.it/~tullio/IS-1/2023/Progetto/C8.pdf>

### 1.4.2) Riferimenti informativi

- T5 - Analisi dei requisiti  
<https://www.math.unipd.it/~tullio/IS-1/2023/Dispense/T5.pdf>
- P2 - I Diagrammi dei Casi d'Uso (UML)  
<https://www.math.unipd.it/~rardin/swea/2022/Diagrammi%20Use%20Case.pdf>
- Documentazione JMAP: the core protocol  
<https://jmap.io/spec-core.html>
- Documentazione JMAP: email  
<https://jmap.io/spec-mail.html>
- Documentazione JMAP: sharing  
<https://jmap.io/spec-sharing.html>
- RFC8620:  
<https://datatracker.ietf.org/doc/html/rfc8620>
- RFC8621:  
<https://datatracker.ietf.org/doc/html/rfc8621>
- Glossario v1.0.0:  
[https://overture-unipd.github.io/docs/rtb/interni/glossario\\_v1.0.0.pdf](https://overture-unipd.github.io/docs/rtb/interni/glossario_v1.0.0.pdf)

## 2) Descrizione generale del prodotto richiesto da *Zextras*

### 2.1) Obiettivo fissato

L'obiettivo a cui vogliamo arrivare alla fine di questo *capitolato* e' capire se ha senso per *Zextras* investire tempo e denaro per estendere questo standard (JMAP) nel software *Carbonio*, mantenendo i vecchi standard per compatibilità con i client attualmente supportati ma permettendogli di espandere le funzionalità offerte ai client di nuova generazione.

### 2.2) Funzionalità del prodotto e requisiti

Nel sistema informatico che dovremmo andare a realizzare dobbiamo mettere a disposizione dei client le seguenti funzionalità:

- L'invio e la ricezione di una mail;
- La gestione delle cartelle;
- La gestione dei contenuti di una cartella;
- L'eliminazione di un oggetto;
- L'eliminazione di una cartella;
- La condivisione di una cartella;
- L'eliminazione di una condivisione di cartella;
- [Opzionale] l'implementazione di un sistema di sincronizzazione che permetta ad un client di mantenersi aggiornato con gli ultimi aggiornamenti della casella di posta visualizzata;

Inoltre il prodotto deve poter essere eseguibile in un container *Docker*, permettendo all'azienda di effettuare in batteria i *test* di funzionalità e di performance, ed è preferibile che sia sviluppato per essere scalabile mediante l'inizializzazione di più nodi *stateless*.

### 2.3) Caratteristiche utente

#### 2.3.1) Utenti di Carbonio

Gli utenti della soluzione software Carbonio sono principalmente:

1. Agenzie di telecomunicazioni, provider (ISP) e operatori mobili poiché si integra perfettamente ai servizi preesistenti e garantisce un reale valore aggiunto ai propri clienti;
2. Il settore pubblico e i settori regolamentati, poiché fornisce un ambiente di lavoro digitale sicuro e conforme alle normative locali come, ad esempio, le leggi sulla protezione dei dati e sulla sovranità del dato;
3. Utenti privati e aziende private tradizionali che desiderano una piattaforma open source per l'e-mail e la collaborazione digitale che offre una soluzione di collaborazione online on-premise.

#### 2.3.2) Utenti del nostro prodotto

Il prodotto che il gruppo Overture andrà a sviluppare ha la sola funzionalità di testare una nuova tecnologia, quindi non ha né un segmento di mercato ben definito né un vero e proprio scopo commerciale. Gli utenti del nostro prodotto quindi sono tutti coloro che desiderano testare il nuovo protocollo JMAP e capire le motivazioni che hanno spinto gli sviluppatori di tutto il mondo e la *IETF* a mettere in discussione un protocollo ampiamente utilizzato come IMAP, cercando di crearne uno più efficiente e moderno

## 2.4) Tecnologie e analisi della struttura di progetto

Come discusso nei paragrafi precedenti, lo scopo finale del nostro prodotto è realizzare un'infrastruttura server, con il fine di realizzare gli *stress test* sui requisiti funzionali richiesti dal proponente.

Per questo motivo, dato che il risultato finale è incentrato sulla performance del protocollo, non ci interessa andare a realizzare un vero e proprio client di posta elettronica completo (e.g.: Thunderbird) perchè per testare i servizi messi a disposizione dal nostro server mail andremo ad utilizzare un client open source.

Per realizzare gli stress test invece, sappiamo che non ci sono client che forniscono una soluzione già pronta (i programmati di tutto il mondo hanno realizzato dei client commerciali, non per svolgere test) quindi è possibile che dovremo andare a realizzare noi un piccolo client con una serie di funzionalità ristrette ma capace di performare un benchmark sull'*efficienza* del protocollo. Questo comunque sarà da decidere in fase di progettazione.

## 3) Casi d'uso

### 3.1) Obiettivi

Questa sezione identifica e descrive tutti i casi d'uso individuati a seguito dell'analisi del capitolo d'appalto e dei dialoghi con l'azienda proponente.

### 3.2) Introduzione ai casi d'uso

I casi d'uso seguono una struttura logica descrivendo ognuno di questi seguendo questo modello:

- **Attori coinvolti:** Il soggetto che esegue una certa azione;
- **Descrizione:** Titolo del *caso d'uso* e breve commento;
- **Precondizioni:** Stato del sistema prima del caso d'uso;
- **Postcondizioni:** Stato del sistema dopo l'esecuzione dello scenario del caso d'uso;
- **Scenario principale:** Una descrizione dettagliata delle azioni che un *attore* deve intraprendere per completare un caso d'uso. È un'espressione formale delle ipotesi e dei risultati del caso d'uso.
- **Estensioni:** L'estensione è una relazione tra due casi d'uso in cui il caso d'uso esteso può essere modificato o arricchito dal caso d'uso estensore. L'estensione è rappresentata da una freccia che punta dal caso d'uso estensore al caso d'uso esteso.
- **Inclusioni:** L'inclusione è una relazione tra due casi d'uso in cui il caso d'uso incluso è eseguito sempre all'interno del caso d'uso che include. L'inclusione è rappresentata da una freccia che punta dal caso d'uso che include al caso d'uso incluso.
- **Generalizzazioni:** La generalizzazione è una relazione tra due casi d'uso in cui il caso d'uso generalizzato rappresenta una categoria più ampia di cui il caso d'uso specifico è un'istanza. La generalizzazione è rappresentata da una freccia che punta dal caso d'uso specifico al caso d'uso generalizzato.

### 3.3) Attori

Gli attori che consideriamo nel prodotto che andremo a realizzare sono:

- Client di posta elettronica.

#### 3.3.1) Alcuni dettagli sui client

Il sistema informatico che andremo a sviluppare in questo progetto ha il fine di testare una nuova tecnologia (il protocollo JMAP) e riportare i risultati dei benchmark effettuati su un determinato server. Per questo motivo, essendo quindi tutto finalizzato a svolgere dei test, non ci è stato esplicitamente riferito di gestire particolari aspetti legati alla sicurezza informatica.

I client hanno a disposizione le funzionalità citate nel (Cap 2.2) fruibili tramite API in modo che noi sviluppatori possiamo testare l'efficienza di queste operazioni con gli stress test.

Si nota però che in un sistema di posta elettronica, per poter visualizzare le email, occorre prima identificarsi con il server mail fornendo appunto la propria email e la password associata. Allo stesso modo per inviare una email il server mail avrà bisogno di conoscere il mittente e quindi ha bisogno di riconoscere il client che intende effettuare questa operazione.

Per questo motivo nel nostro sistema andremo a prevedere un classico sistema di autenticazione composto da email e password.

### 3.4) Gestione degli errori

Nei diagrammi dei casi d'uso che andremo a riportare nelle sezioni successive, troveremo spesso dei casi d'uso di errore volti a garantire la consistenza dei dati.

Per esempio: se un utente vuole creare una cartella gli verrà chiesto di fornire attraverso il client le proprietà dell'oggetto Mailbox da creare (nome, eventuale genitore, ruolo ed altri dettagli), se l'utente fornisce dei parametri che non sono validi o accettabili, è chiaro che il sistema debba fornire un errore. A livello pratico però, utilizzando un classico client di posta elettronica dotato di interfaccia grafica, nel momento in cui vogliamo effettuare il tipo di operazione sopra descritta saremo totalmente guidati dall'interfaccia grafica, dunque difficilmente potremo fornire dei parametri non validi.

Tuttavia in seguito ad una profonda analisi, il gruppo Overture ha stabilito che il focus principale del progetto è quello di implementare un backend che metta a disposizione un servizio tramite API, dunque il client che fruisce dei nostri *endpoints* non è sotto il nostro dominio (potrebbe essere anche un *client CLI*, da linea di comando, dove è l'utente che manualmente digita i parametri da fornire) e quindi dobbiamo prevedere dei controlli aggiuntivi che garantiscano l'integrità dei dati inviati da ogni client. Per essere i più precisi possibile, abbiamo seguito nella definizione degli errori le specifiche dello standard JMAP.

### 3.5) Elenco dei casi d'uso

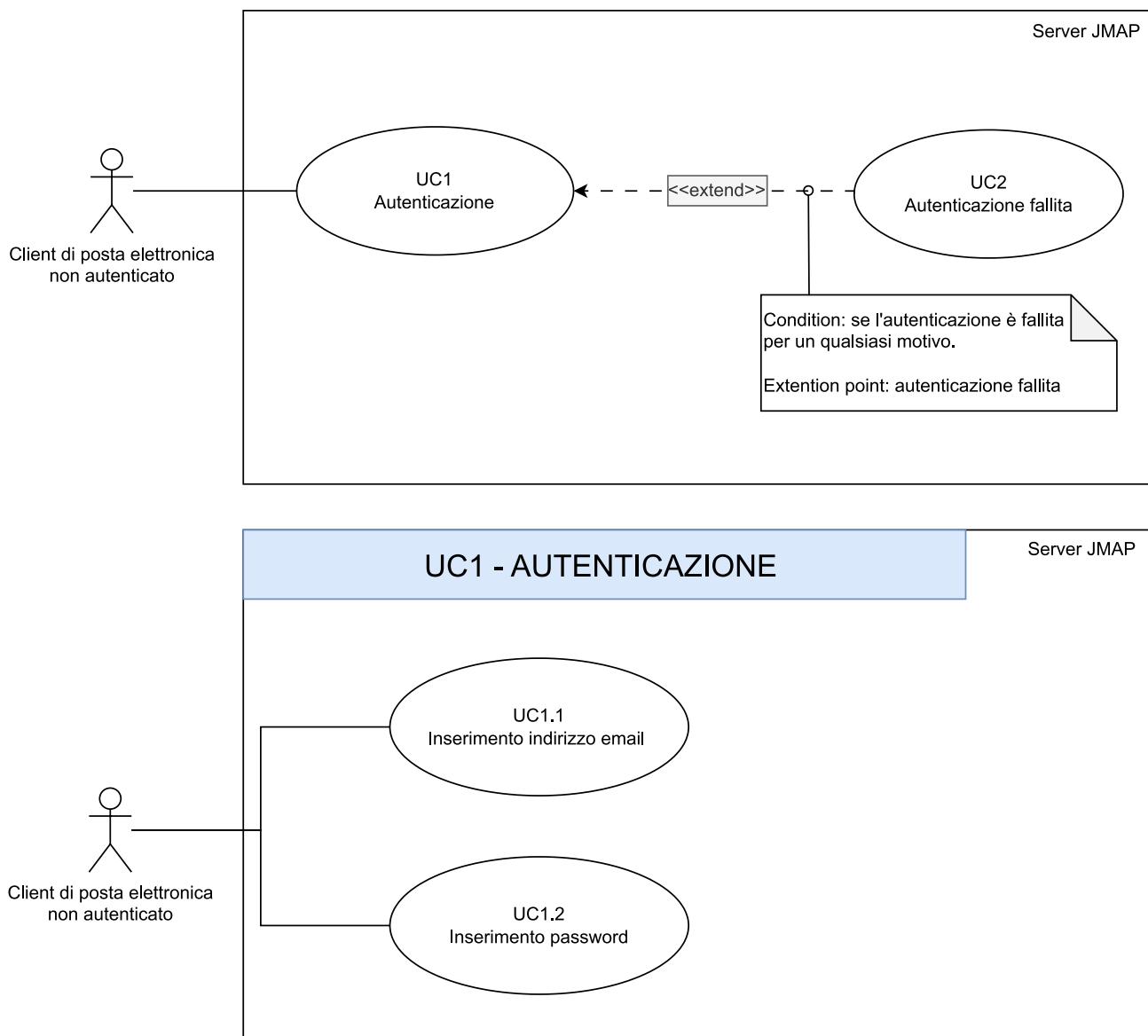


Figura 1: UC1 - Autenticazione

#### 3.5.1) UC1 - Autenticazione

- **Attore principale:** Client di posta elettronica non autenticato;
- **Descrizione:** Un client di posta elettronica non autenticato ha l'intenzione di autenticarsi presso il server JMAP per interagire con dati e servizi offerti da quest'ultimo;
- **Precondizioni:** Il client deve essere in possesso delle credenziali necessarie per l'autenticazione;
- **Postcondizioni:** Se le credenziali fornite sono valide, il server ha risposto al client, confermando l'autenticazione. Altrimenti il server ha restituito una risposta che indica il fallimento dell'autenticazione;
- **Scenario principale:**
  1. Il client inizia la procedura di autenticazione inviando una richiesta al server;
  2. Il server riceve la richiesta e verifica le credenziali;
- **Estensioni:**
  - Autenticazione fallita;

- **Inclusioni:** /
- **Generalizzazioni:** /

### 3.5.1.1) UC 1.1 - Inserimento indirizzo email

- **Attore principale:** Client di posta elettronica non autenticato;
- **Descrizione:** Un client di posta elettronica non autenticato ha l'intenzione di autenticarsi presso il server JMAP per interagire con dati e servizi offerti da quest'ultimo, e quindi inserisce all'interno della richiesta le prime delle credenziali necessarie: l'indirizzo email;
- **Precondizioni:** Il client deve essere in possesso dell'indirizzo email necessario per l'autenticazione;
- **Postcondizioni:** Il client ha inserito all'interno della richiesta la prima delle credenziali: l'indirizzo email;
- **Scenario principale:**
  1. Il client costruisce la richiesta inserendo al suo interno l'indirizzo email personale necessario per l'autenticazione, il quale è stato fornito dall'utente che sta utilizzando il client;

### 3.5.1.2) UC 1.2 - Inserimento password

- **Attore principale:** Client di posta elettronica non autenticato;
- **Descrizione:** Un client di posta elettronica non autenticato ha l'intenzione di autenticarsi presso il server JMAP per interagire con dati e servizi offerti da quest'ultimo, e quindi inserisce all'interno della richiesta la seconda delle credenziali necessarie: la password;
- **Precondizioni:** Il client deve essere in possesso della password relativa all'indirizzo email, necessaria per l'autenticazione;
- **Postcondizioni:** Il client ha inserito all'interno della richiesta la seconda delle credenziali: la password;
- **Scenario principale:**
  1. Il client costruisce la richiesta inserendo al suo interno la password necessaria per l'autenticazione, la quale è stata fornita dall'utente che sta utilizzando il client;

### 3.5.2) UC2 - Autenticazione fallita

- **Attore principale:** Client di posta elettronica non autenticato;
- **Descrizione:** Un client di posta elettronica non autenticato, con l'intenzione di autenticarsi presso il server JMAP per interagire con dati e servizi offerti da quest'ultimo, non è riuscito nel suo intento perché ha fallito la fase di autenticazione;
- **Precondizioni:** Il client ha tentato di autenticarsi presso il server ma l'autenticazione è fallita;
- **Postcondizioni:** Il client riceve dal server una risposta che indica il fallimento dell'autenticazione con eventuali dettagli;
- **Scenario principale:**
  1. Il server ha ricevuto la richiesta di connessione da parte del client e verificando le credenziali non è riuscito ad autenticarlo;

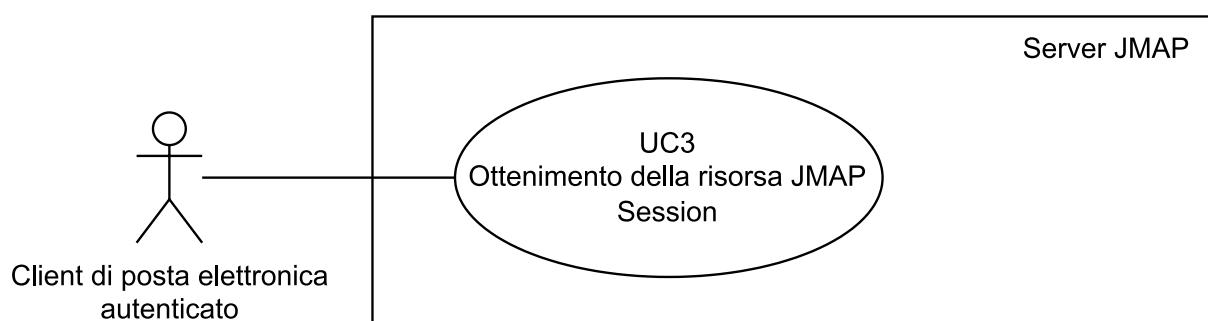


Figura 2: UC3 - Ottenimento della risorsa JMAP Session

### 3.5.3) UC3 - Ottenimento della risorsa JMAP Session

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica autenticato ha l'intenzione di connettersi al server JMAP per interagire con dati e servizi offerti da quest'ultimo. Di conseguenza necessita di ottenere la risorsa JMAP Session, contenente informazioni sulle capacità del server, dettagli sull'account dell'utente e le URL per le richieste API future;
- **Precondizioni:** Il client deve essere autenticato ed in possesso del URL per la risorsa JMAP Session;
- **Postcondizioni:** Il server ha restituito un oggetto JSON che rappresenta la sessione, contenente informazioni sulle capacità del server, dettagli sull'account dell'utente e le URL per le richieste API future;
- **Scenario principale:**
  1. Il client invia una richiesta all'URL della risorsa JMAP Session sul server;
  2. Il server riceve la richiesta e risponde con l'oggetto JSON-encoded che rappresenta la sessione, ovvero la risorsa JMAP Session;

### 3.5.4) UC4 - Errore “unknownCapability”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma questa includeva una capacità nella proprietà “using” che il server non supporta. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma non è riuscito a soddisfare il suo bisogno perché la richiesta includeva una capacità nella proprietà “using” che il server non supporta;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando quali capacità specificate nella proprietà “using” il server non supporta;

### 3.5.5) UC5 - Errore “notJSON”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma il tipo di contenuto della richiesta non era application/json o la richiesta non è stata interpretata come I-JSON. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma non è riuscito a soddisfare il suo bisogno perché il tipo di contenuto della richiesta non era application/json o la richiesta non è stata interpretata come I-JSON;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;

- 
2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che il tipo di contenuto della richiesta non era application/json o la richiesta non è stata interpretata come I-JSON;

### 3.5.6) UC6 - Errore “notRequest”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma, nonostante la richiesta sia stata interpretata come JSON, essa non ha corrisposto alla firma di tipo dell'oggetto di richiesta (Request). Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma, nonostante la richiesta sia stata interpretata come JSON, essa non ha corrisposto alla firma di tipo dell'oggetto di richiesta (Request);
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nonostante la richiesta sia stata interpretata come JSON, essa non ha corrisposto alla firma di tipo dell'oggetto di richiesta (Request);

### 3.5.7) UC7 - Errore “limit”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma la richiesta non è stata elaborata in quanto avrebbe superato uno dei limiti di richiesta definiti sull'oggetto di capacità, come maxSizeRequest, maxCallsInRequest o maxConcurrentRequests. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma la richiesta non è stata elaborata in quanto avrebbe superato uno dei limiti di richiesta definiti sull'oggetto di capacità, come maxSizeRequest, maxCallsInRequest o maxConcurrentRequests;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che la richiesta è più grande di quanto il server sia disposto a elaborare;

### 3.5.8) UC8 - Errore “serverUnavailable”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma alcune risorse interne del server non erano disponibili temporaneamente (tentare la stessa operazione in seguito, magari dopo un ritardo con un fattore casuale, potrebbe avere successo). Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma alcune risorse interne del server non erano disponibili temporaneamente;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che alcune risorse interne del server non erano disponibili temporaneamente ma tentare la stessa operazione in seguito, magari dopo un ritardo con un fattore casuale, potrebbe avere successo;

### 3.5.9) UC9 - Errore “serverFail”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma si è verificato un errore inaspettato o sconosciuto durante l'elaborazione della richiesta (non sono state apportate modifiche allo stato del server e ci si aspetta che un nuovo tentativo della stessa operazione fallisca nuovamente). Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma si è verificato un errore inaspettato o sconosciuto durante l'elaborazione della richiesta;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che si è verificato un errore inaspettato o sconosciuto durante l'elaborazione della richiesta e fornendo nella risposta una proprietà “description” che dovrebbe fornire ulteriori dettagli sull'errore;

### 3.5.10) UC10 - Errore “serverPartialFail”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma si è verificato un errore inaspettato o sconosciuto durante l'elaborazione della richiesta (alcuni, ma non tutti, i cambiamenti previsti descritti dalla richiesta sono avvenuti, dunque il client DEVE risincronizzare i dati interessati per determinare lo stato del server). Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma si è verificato un errore inaspettato o sconosciuto durante l'elaborazione della richiesta;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che si è verificato un errore inaspettato o sconosciuto durante l'elaborazione della richiesta;

---

razione della richiesta e fornendo nella risposta una proprietà “description” che dovrebbe fornire ulteriori dettagli sull’errore;

### 3.5.11) UC11 - Errore “unknownMethod”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session, ma il server non riconosce il nome di un metodo. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all’URL dell’API ma il server non riconosce il nome di un metodo;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l’esito fallimentare dell’operazione, indicando che il server non riconosce il nome di un metodo;

### 3.5.12) UC12 - Errore “invalidArguments”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session, ma uno degli argomenti di un metodo forniti ha un tipo errato oppure è non valido oppure manca un argomento obbligatorio. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all’URL dell’API ma uno degli argomenti di un metodo forniti ha un tipo errato oppure è non valido oppure manca un argomento obbligatorio;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l’esito fallimentare dell’operazione, indicando che uno degli argomenti di un metodo forniti ha un tipo errato oppure è non valido oppure manca un argomento obbligatorio. Nella risposta potrebbe essere presente una proprietà “description” per facilitare il debug, con una spiegazione del problema. Questa è una stringa non localizzata e non è destinata a essere mostrata direttamente agli utenti finali;

### 3.5.13) UC13 - Errore “invalidResultReference”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session, ma un metodo ha utilizzato un riferimento di risultato per uno dei suoi argomenti che non è stato possibile risolvere. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all’URL dell’API ma un metodo ha utilizzato un riferimento di risultato per uno dei suoi argomenti che non è stato possibile risolvere;

- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che un metodo ha utilizzato un riferimento di risultato per uno dei suoi argomenti che non è stato possibile risolvere;

### 3.5.14) UC14 - Errore “**forbidden**”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma è presente in essa un metodo i cui argomenti sono validi, però l'esecuzione di quest'ultimo violerebbe una Access Control List (ACL) o un'altra policy di autorizzazione. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma è presente in essa un metodo i cui argomenti sono validi, però l'esecuzione di quest'ultimo violerebbe una Access Control List (ACL) o un'altra policy di autorizzazione;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che in essa è presente un metodo i cui argomenti sono validi, però l'esecuzione di quest'ultimo violerebbe una Access Control List (ACL) o un'altra policy di autorizzazione;

### 3.5.15) UC15 - Errore “**accountNotFound**”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma l'"accountID" fornito non corrisponde a un account valido. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma l'"accountID" fornito non corrisponde a un account valido;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che l'"accountID" fornito non corrisponde a un account valido;

### 3.5.16) UC16 - Errore “**accountNotSupportedByMethod**”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma, nonostante l'"accountID" fornito corrisponda a un account valido, quest'ultimo non supporta un metodo o tipo

di dati specificato nella richiesta. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma, nonostante l'"accountID" fornito corrisponda a un account valido, quest'ultimo non supporta un metodo o tipo di dati specificato nella richiesta;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che, nonostante l'"accountID" fornito corrisponda a un account valido, quest'ultimo non supporta un metodo o tipo di dati specificato nella richiesta;

### 3.5.17) UC17 - Errore “accountReadOnly”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma in essa è presente un metodo che tenta di modificare lo stato nonostante l'account sia in sola lettura (come specificato sull'oggetto Account corrispondente nella risorsa JMAP Session). Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma in essa è presente un metodo che tenta di modificare lo stato nonostante l'account sia in sola lettura;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che in essa è presente un metodo che tenta di modificare lo stato nonostante l'account sia in sola lettura;

### 3.5.18) UC18 - Errore “cannotCalculateChanges”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma il server non può calcolare le modifiche dello stato dalla stringa di stato fornita dal client. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma il server non può calcolare le modifiche dello stato dalla stringa di stato fornita dal client;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che il server non può calcolare le modifiche dello stato dalla stringa di stato fornita dal client;

### 3.5.19) UC19 - Errore “overQuota”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session, ma la creazione contenuta nella richiesta supererebbe un limite definito dal server sul numero o sulla dimensione totale degli oggetti di questo tipo. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all’URL dell’API ma la creazione contenuta nella richiesta supererebbe un limite definito dal server sul numero o sulla dimensione totale degli oggetti di questo tipo;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l’esito fallimentare dell’operazione, indicando che la creazione contenuta nella richiesta supererebbe un limite definito dal server sul numero o sulla dimensione totale degli oggetti di questo tipo;

### 3.5.20) UC20 - Errore “notFound”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session, ma qualche ID fornito non può essere trovato. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all’URL dell’API ma qualche ID fornito non può essere trovato;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l’esito fallimentare dell’operazione, indicando che qualche ID fornito non può essere trovato;

### 3.5.21) UC21 - Errore “willDestroy”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session, ma il client ha richiesto che un oggetto fosse sia aggiornato che distrutto nella stessa richiesta, di conseguenza il server ha deciso di ignorare l’aggiornamento. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all’URL dell’API ma il client ha richiesto che un oggetto fosse sia aggiornato che distrutto nella stessa richiesta, di conseguenza il server ha deciso di ignorare l’aggiornamento;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**

1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che, siccome il client ha richiesto che un oggetto fosse sia aggiornato che distrutto nella stessa richiesta, il server, di conseguenza, ha deciso di ignorare l'aggiornamento;

### 3.5.22) UC22 - Errore “tooLarge”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma l'azione comporterebbe la creazione di un oggetto che supera il limite definito dal server per la dimensione massima di un singolo oggetto di questo tipo. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma l'azione comporterebbe la creazione di un oggetto che supera il limite definito dal server per la dimensione massima di un singolo oggetto di questo tipo;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che l'azione comporterebbe la creazione di un oggetto che supera il limite definito dal server per la dimensione massima di un singolo oggetto di questo tipo;

### 3.5.23) UC23 - Errore “rateLimit”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma l'azione comporterebbe la creazione di un oggetto per cui sono stati creati troppi oggetti di questo tipo di recente, ed è stato raggiunto un limite di frequenza definito dal server (potrebbe funzionare se riprovato più tardi). Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma l'azione comporterebbe la creazione di un oggetto per cui sono stati creati troppi oggetti di questo tipo di recente, ed è stato raggiunto un limite di frequenza definito dal server;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che l'azione comporterebbe la creazione di un oggetto per cui sono stati creati troppi oggetti di questo tipo di recente, ed è stato raggiunto un limite di frequenza definito dal server;

### 3.5.24) UC24 - Errore “invalidPatch”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma il PatchO-

bject fornito per aggiornare il record non era una patch valida. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma il PatchObject fornito per aggiornare il record non era una patch valida;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che il PatchObject fornito per aggiornare il record non era una patch valida;

### 3.5.25) UC25 - Errore “invalidProperties”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma il record fornito non è valido per un qualche motivo. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma il record fornito non è valido per un qualche motivo;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che il record fornito non è valido per un qualche motivo;

### 3.5.26) UC26 - Errore “singleton”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma l'operazione che sta tentando di svolgere agisce su un tipo singleton, quindi il server non può crearne un altro né distruggere quello esistente. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma l'operazione che sta tentando di svolgere agisce su un tipo singleton, quindi il server non può crearne un altro né distruggere quello esistente;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che l'operazione che sta tentando di svolgere agisce su un tipo singleton, quindi il server non può crearne un altro né distruggere quello esistente;

### 3.5.27) UC27 - Errore “requestTooLarge”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma il numero totale di azioni supera il massimo che il server è disposto a elaborare in una singola chiamata di metodo interna alla richiesta. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma il numero totale di azioni supera il massimo che il server è disposto a elaborare in una singola chiamata di metodo interna alla richiesta;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che il numero totale di azioni supera il massimo che il server è disposto a elaborare in una singola chiamata di metodo interna alla richiesta;

### 3.5.28) UC28 - Errore “stateMismatch”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma è stato fornito un argomento ifInState, e non corrisponde allo stato attuale. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma è stato fornito un argomento ifInState, e non corrisponde allo stato attuale;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che è stato fornito un argomento ifInState e non corrisponde allo stato attuale;

### 3.5.29) UC29 - Errore “blobNotFound”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma almeno un ID blob fornito per una parte del corpo dell'email non esiste. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma almeno un ID blob fornito per una parte del corpo dell'email non esiste;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;

2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che almeno un ID blob fornito per una parte del corpo dell'email non esiste. Deve essere inclusa una proprietà aggiuntiva notFound di tipo Id[] nell'oggetto SetError contenente ogni blobId referenziato da una parte del corpo dell'email che non può essere trovato sul server;

### 3.5.30) UC30 - Errore “tooManyKeywords”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma la modifica alle parole chiave dell'email contenuta nella richiesta supererebbe un massimo definito dal server. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma la modifica alle parole chiave dell'email contenuta nella richiesta supererebbe un massimo definito dal server;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che la modifica alle parole chiave dell'email contenuta nella richiesta supererebbe un massimo definito dal server;

### 3.5.31) UC31 - Errore “tooManyMailboxes”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma la modifica all'insieme di caselle di posta a cui appartiene l'email contenuta nella richiesta supererebbe un massimo definito dal server. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma la modifica all'insieme di caselle di posta a cui appartiene l'email contenuta nella richiesta supererebbe un massimo definito dal server;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che la modifica all'insieme di caselle di posta a cui appartiene l'email contenuta nella richiesta supererebbe un massimo definito dal server;

### 3.5.32) UC32 - Errore “alreadyExists”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma il server vieta i duplicati, e il record definito dalla richiesta esiste già nell'account di destinazione. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma il server vieta i duplicati, e il record definito dalla richiesta esiste già nell'account di destinazione;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che il server vieta i duplicati, e il record definito dalla richiesta esiste già nell'account di destinazione. Una proprietà existingId di tipo Id DEVE essere inclusa nell'oggetto SetError con l'ID del record esistente;

### 3.5.33) UC33 - Errore “fromAccountNotFound”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma nella richiesta è contenuto un fromAccountId che non corrisponde a un account valido. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma nella richiesta è contenuto un fromAccountId che non corrisponde a un account valido;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nella richiesta è contenuto un fromAccountId che non corrisponde a un account valido;

### 3.5.34) UC34 - Errore “fromAccountNotSupportedByMethod”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma nella richiesta è contenuto un fromAccountId che, nonostante corrisponda a un account valido, non supporta questo tipo di dati. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma nella richiesta è contenuto un fromAccountId che, nonostante corrisponda a un account valido, non supporta questo tipo di dati;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nella richiesta è contenuto un fromAccountId che, nonostante corrisponda a un account valido, non supporta questo tipo di dati;

### 3.5.35) UC35 - Errore “anchorNotFound”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma nella richiesta è stato fornito un argomento di ancoraggio che non può essere trovato nei risultati della query. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma nella richiesta è stato fornito un argomento di ancoraggio che non può essere trovato nei risultati della query;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nella richiesta è stato fornito un argomento di ancoraggio che non può essere trovato nei risultati della query;

### 3.5.36) UC36 - Errore “unsupportedSort”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma nella richiesta è presente una clausola di ordinamento che è sintatticamente valida, ma include una proprietà sulla quale il server non supporta l'ordinamento o un metodo di collazione che non riconosce. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma nella richiesta è presente una clausola di ordinamento che è sintatticamente valida, ma include una proprietà sulla quale il server non supporta l'ordinamento o un metodo di collazione che non riconosce;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nella richiesta è presente una clausola di ordinamento che è sintatticamente valida, ma include una proprietà sulla quale il server non supporta l'ordinamento o un metodo di collazione che non riconosce;

### 3.5.37) UC37 - Errore “unsupportedFilter”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma nella richiesta è presente un filtro che è sintatticamente valido, ma il server non è in grado di elaborarlo. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma nella richiesta è presente un filtro che è sintatticamente valido, ma il server non è in grado di elaborarlo;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**

1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nella richiesta è presente un filtro che è sintatticamente valido, ma il server non è in grado di elaborarlo;

### 3.5.38) UC38 - Errore “tooManyChanges”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma nella richiesta ci sono più modifiche rispetto all'argomento maxChanges del client. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma nella richiesta ci sono più modifiche rispetto all'argomento maxChanges del client;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nella richiesta ci sono più modifiche rispetto all'argomento maxChanges del client;

### 3.5.39) UC39 - Errore “mailboxHasChild”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma nella richiesta la cartella(Mailbox) che si vuole eliminare ha ancora almeno una cartella figlia. Il client DEVE rimuovere quest'ultima prima di poter eliminare la cartella principale. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma nella richiesta la cartella(Mailbox) che si vuole eliminare ha ancora almeno una cartella figlia;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nella richiesta la cartella(Mailbox) che si vuole eliminare ha ancora almeno una cartella figlia e che il client DEVE rimuovere quest'ultima prima di poter eliminare la cartella principale;

### 3.5.40) UC40 - Errore “mailboxHasEmail”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma nella richiesta la cartella(Mailbox) che si vuole eliminare ha almeno una email ad essa assegnato e l'argomento onDestroyRemoveEmails era impostato su false. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma nella richiesta la cartella(Mailbox) che si vuole eliminare ha almeno una email ad essa assegnato e l'argomento onDestroyRemoveEmails era impostato su false;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nella richiesta la cartella(Mailbox) che si vuole eliminare ha almeno una email ad essa assegnato e l'argomento onDestroyRemoveEmails era impostato su false;

### 3.5.41) UC41 - Errore “invalidEmail”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma nella richiesta è presente un'email da inviare che è in qualche modo non valida. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma nella richiesta è presente un'email da inviare che è in qualche modo non valida;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nella richiesta è presente un'email da inviare che è in qualche modo non valida;

### 3.5.42) UC42 - Errore “tooManyRecipients”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma nella richiesta è presente un envelope che ha più destinatari di quanti il server consenta. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma nella richiesta è presente un envelope che ha più destinatari di quanti il server consenta;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nella richiesta è presente un envelope che ha più destinatari di quanti il server consenta;

### 3.5.43) UC43 - Errore “noRecipients”

- **Attore principale:** Client di posta elettronica autenticato;

- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma nella richiesta è presente un envelope che non ha destinatari indicati. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma nella richiesta è presente un envelope che non ha destinatari indicati;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nella richiesta è presente un envelope che non ha destinatari indicati;

#### 3.5.44) UC44 - Errore “invalidRecipients”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma nella richiesta è presente un envelope che contiene almeno un indirizzo email destinatario non valido. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma nella richiesta è presente un envelope che contiene almeno un indirizzo email destinatario non valido;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che nella richiesta è presente un envelope che contiene almeno un indirizzo email destinatario non valido;

#### 3.5.45) UC45 - Errore “forbiddenMailFrom”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session, ma il server non permette all'utente di inviare un messaggio con questo indirizzo mittente nell'envelope (From address). Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all'URL dell'API ma il server non permette all'utente di inviare un messaggio con questo indirizzo mittente nell'envelope (From address);
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l'esito fallimentare dell'operazione, indicando che il server non permette all'utente di inviare un messaggio con questo indirizzo mittente nell'envelope (From address);

### 3.5.46) UC46 - Errore “**forbiddenFrom**”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session, ma il server non permette all’utente di inviare un messaggio con il campo di intestazione From del messaggio da inviare. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all’URL dell’API ma il server non permette all’utente di inviare un messaggio con il campo di intestazione From del messaggio da inviare;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l’esito fallimentare dell’operazione, indicando che il server non permette all’utente di inviare un messaggio con il campo di intestazione From del messaggio da inviare;

### 3.5.47) UC47 - Errore “**forbiddenToSend**”

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica con un determinato scopo ha eseguito una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session, ma il server non permette all’utente di inviare un messaggio in questo momento. Di conseguenza il client riceverà una risposta che indica il motivo del fallimento;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione. Ha poi proceduto ad eseguire una richiesta POST autenticata all’URL dell’API ma il server non permette all’utente di inviare un messaggio in questo momento;
- **Postcondizioni:** Il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client ha inviato la richiesta contenente le chiamate di metodo necessarie per raggiungere un determinato scopo;
  2. Il server ha elaborato la richiesta e inviato una risposta JSON che contiene l’esito fallimentare dell’operazione, indicando che il server non permette all’utente di inviare un messaggio in questo momento;

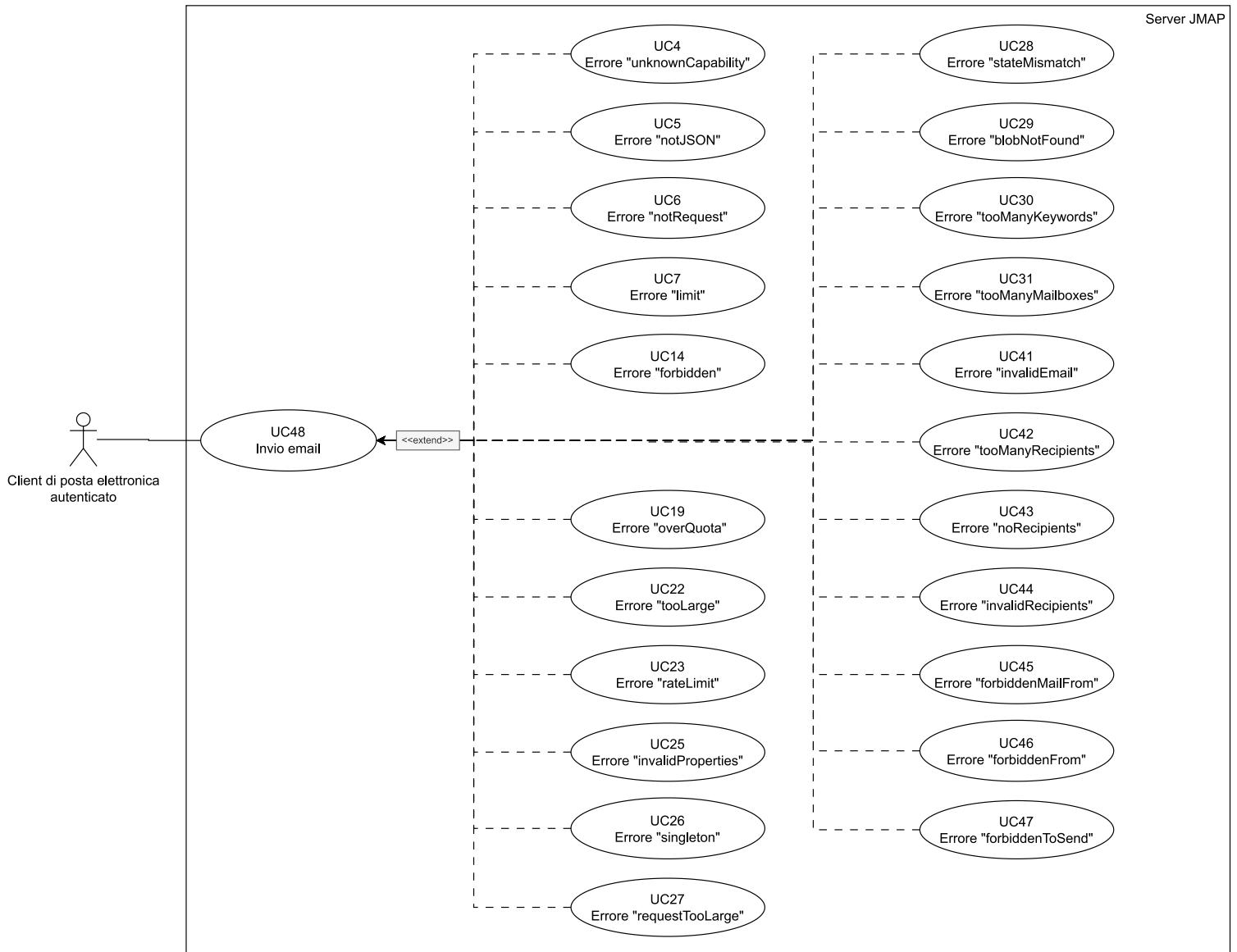


Figura 3: UC48 - Invio email

### 3.5.48) UC48 - Invio email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** L'email è stata inviata con successo ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per inviare un'email;
  2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
  3. Il server elabora la richiesta API e invia una risposta JSON che contiene l'esito dell'operazione di invio dell'email;
- **Estensioni:**
  - Errore "unknownCapability";

- Errore “notJSON”;
- Errore “notRequest”;
- Errore “limit”;
- Errore “forbidden”;
- Errore “overQuota”;
- Errore “tooLarge”;
- Errore “rateLimit”;
- Errore “invalidProperties”;
- Errore “singleton”;
- Errore “requestTooLarge”;
- Errore “stateMismatch”;
- Errore “blobNotFound”;
- Errore “tooManyKeywords”;
- Errore “tooManyMailboxes”;
- Errore “invalidEmail”;
- Errore “tooManyRecipients”;
- Errore “noRecipients”;
- Errore “invalidRecipients”;
- Errore “forbiddenMailFrom”;
- Errore “forbiddenFrom”;
- Errore “forbiddenToSend”;
- **Inclusioni:** /
- **Generalizzazioni:** /

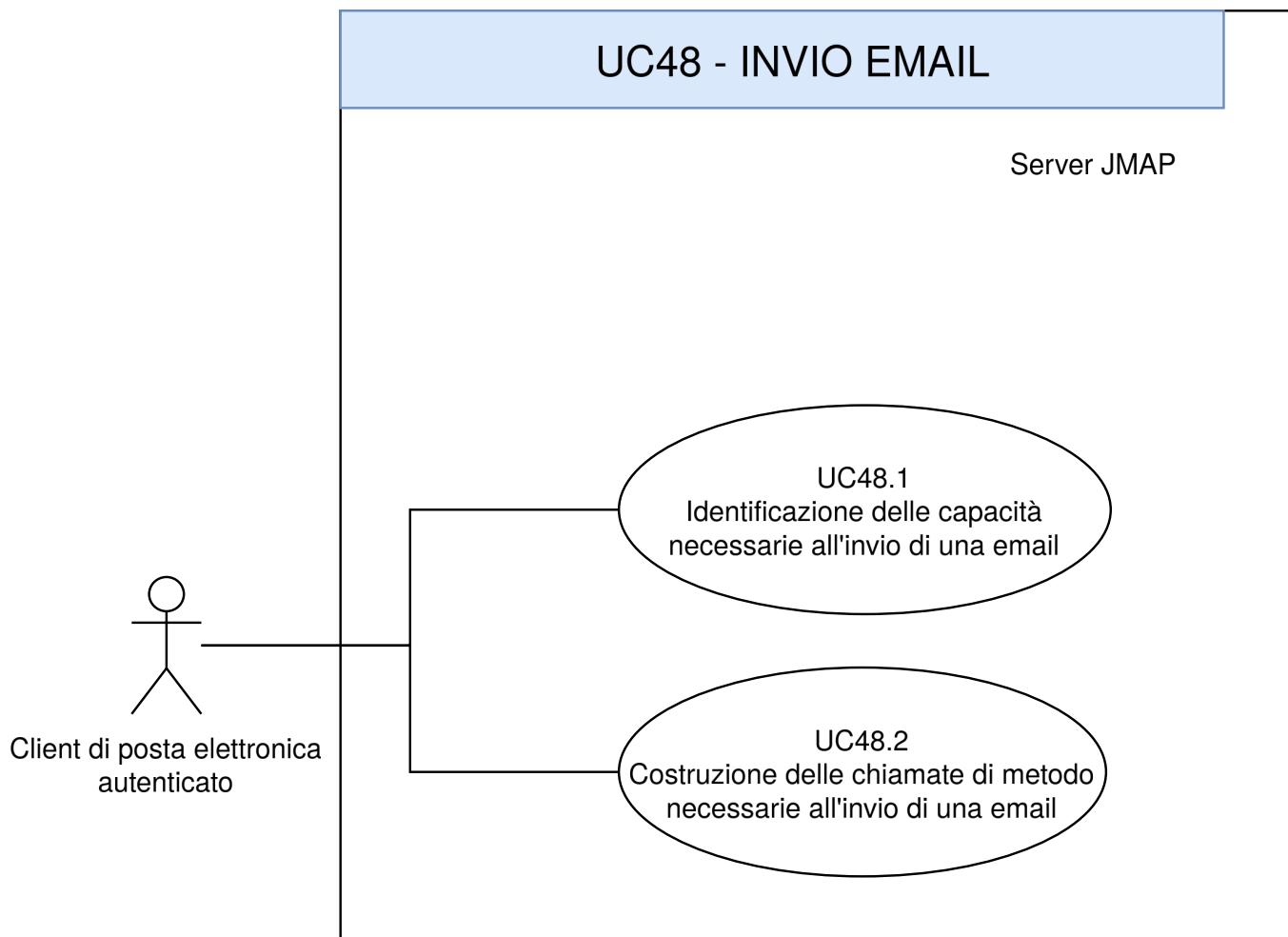


Figura 4: Sottocasi UC48 - Invio email

### 3.5.48.1) UC48.1 - Identificazione delle capacità necessarie all'invio di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per l'invio di una email;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all'interno della richiesta le capacità JMAP necessarie per l'invio di una email;
- **Scenario principale:**
  1. Il client include nella sezione “using” dell’oggetto Request le capacità JMAP “urn:ietf:params:jmap:core” e “urn:ietf:params:jmap:mail”;

### 3.5.48.2) UC48.2 - Costruzione delle chiamate di metodo necessarie all'invio di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per l'invio di una email;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'array “methodCalls” della richiesta le chiamate di metodo necessarie per l'invio di una email;
- **Scenario principale:**
  1. Il client crea un array di chiamate di metodo denominato “methodCalls” internamente all'oggetto Request, all'interno del quale inserisce il metodo “Email/set”, un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo, ed il metodo “EmailSubmission/set”, un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

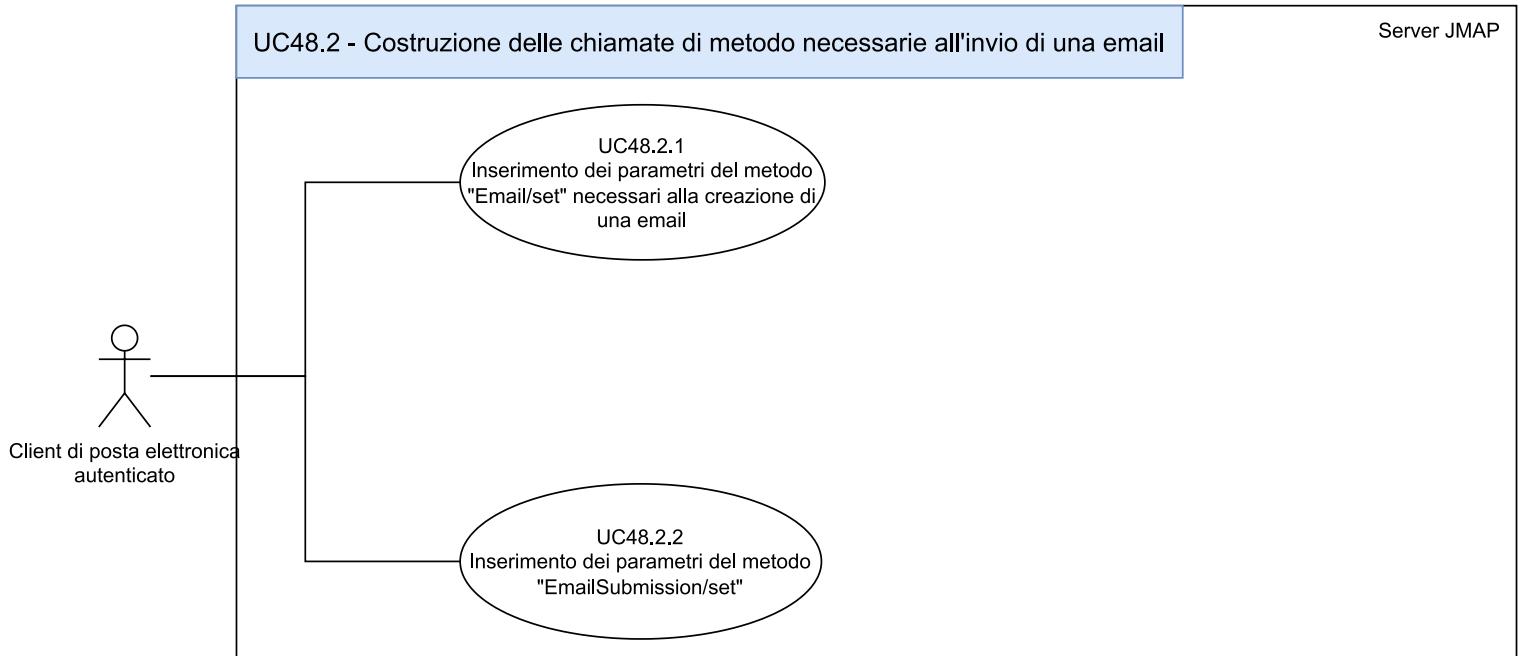


Figura 5: Sottocasi UC48.2 - Costruzione delle chiamate di metodo necessarie all'invio di una email

### 3.5.48.2.1) UC48.2.1 - Inserimento dei parametri del metodo “Email/set” necessari alla creazione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Email/set”, necessario per la creazione di una email, per il quale vanno specificati i parametri necessari;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell'array “methodCalls” della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all'interno dell'oggetto di argomenti della chiamata di metodo “Email/set”, la quale è contenuta nell'array di chiamate di metodo denominato “methodCalls”, contenuto a sua volta all'interno dell'oggetto Request;

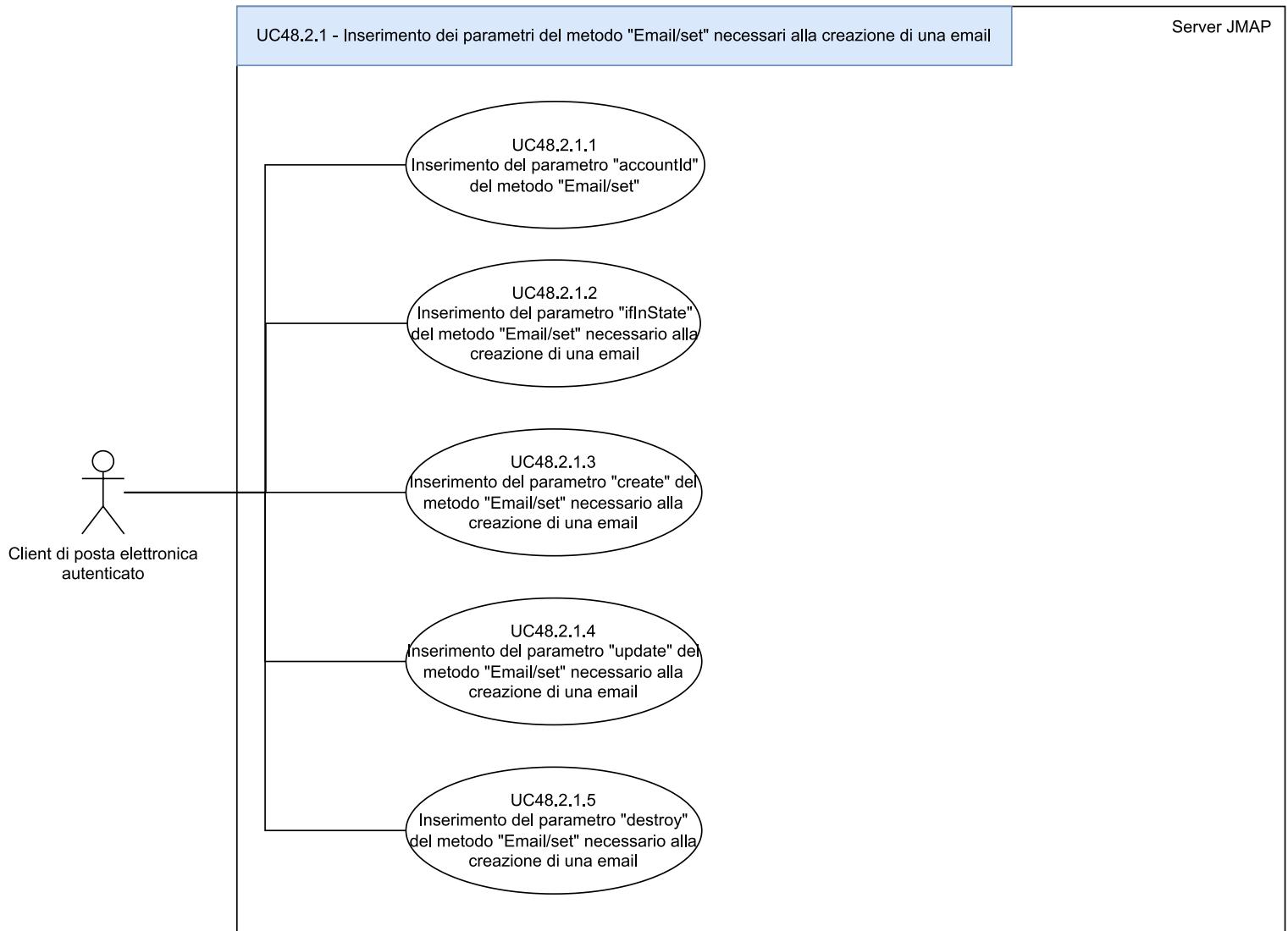


Figura 6: Sottocasi UC48.2.1 - Inserimento dei parametri del metodo "Email/set" necessari alla creazione di una email

### 3.5.48.2.1.1) UC48.2.1.1 - Inserimento del parametro "accountId" del metodo "Email/set" necessario alla creazione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "Email/set", necessario per la creazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è "accountId";
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Email/set", contenuto nell'array "methodCalls" della richiesta, il parametro "accountId";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "accountId", il quale rappresenta l'identificativo dell'account da utilizzare;

---

**3.5.48.2.1.2) UC48.2.1.2 - Inserimento del parametro “ifInState” del metodo “Email/set” necessario alla creazione di una email**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Email/set”, necessario per la creazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “ifInState”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “ifInState”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “ifInState”, il quale è un meccanismo di controllo che impedisce la sovrascrittura di dati obsoleti o incoerenti sul server. Esso garantisce che la richiesta venga eseguita solo se il client è a conoscenza dello stato attuale degli oggetti e che tale stato sia valido al momento della richiesta;

**3.5.48.2.1.3) UC48.2.1.3 - Inserimento del parametro “create” del metodo “Email/set” necessario alla creazione di una email**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Email/set”, necessario per la creazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “create”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “create”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “create”, il quale è utilizzato per specificare l’intenzione di creare una email. Questo parametro è una mappa, in cui le chiavi sono identificatori temporanei di creazione (creation id) assegnati dal client ed i valori sono oggetti Email da creare. La definizione di un oggetto Email può contenere valori predefiniti per le proprietà. Qualsiasi proprietà con un valore predefinito può essere omessa dal client. Inoltre Il client DEVE omettere qualsiasi proprietà che può essere impostata solo dal server;

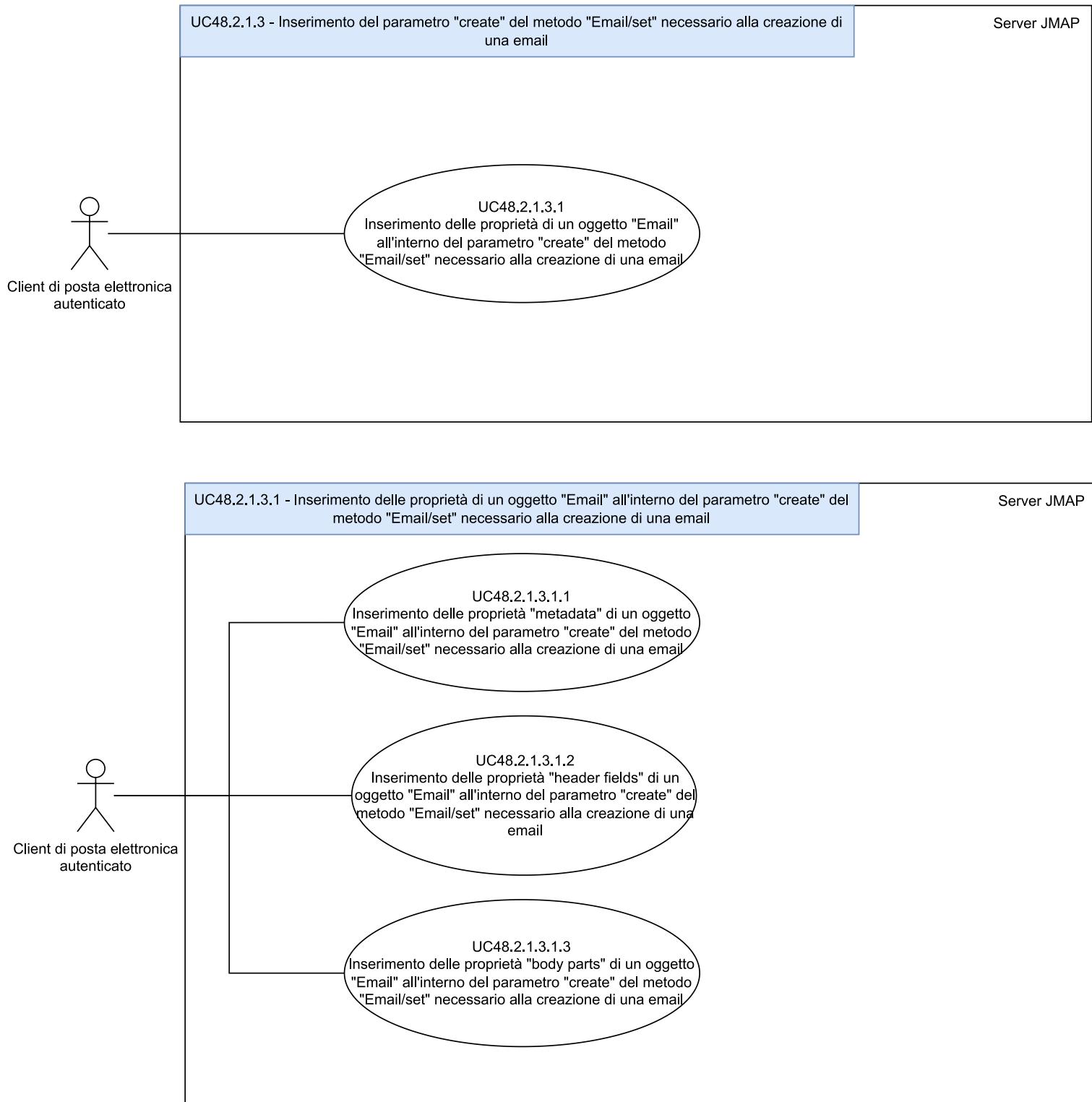


Figura 7: Sottocasi UC48.2.1.3 - Inserimento del parametro "create" del metodo "Email/set" necessario alla creazione di una email

### 3.5.48.2.1.3.1) UC48.2.1.3.1 - Inserimento delle proprietà di un oggetto "Email" all'interno del parametro "create" del metodo "Email/set" necessario alla creazione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session.

Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Email/set”, necessario per la creazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “create”. All’interno di questo parametro vanno definite le proprietà dell’email da inviare;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito le proprietà di un oggetto “Email” all’interno del parametro “create”, contenuto a sua volta all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta;
- **Scenario principale:**
  1. Il client specifica all’interno del parametro “create”, contenuto nell’oggetto di argomenti della chiamata di metodo, le proprietà “metadata” dell’oggetto “Email” da inviare;
  2. Il client specifica all’interno del parametro “create”, contenuto nell’oggetto di argomenti della chiamata di metodo, le proprietà “header” dell’oggetto “Email” da inviare;
  3. Il client specifica all’interno del parametro “create”, contenuto nell’oggetto di argomenti della chiamata di metodo, le proprietà “body parts” dell’oggetto “Email” da inviare;

#### **3.5.48.2.1.3.1.1) UC48.2.1.3.1.1 - Inserimento delle proprietà “metadata” di un oggetto “Email” all’interno del parametro “create” del metodo “Email/set” necessario alla creazione di una email**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Email/set”, necessario per la creazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “create”. All’interno di questo parametro vanno definite le proprietà dell’email da inviare. Tra queste troviamo le proprietà “metadata”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito le proprietà “metadata” di un oggetto “Email” all’interno del parametro “create”, contenuto a sua volta all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta;
- **Scenario principale:**
  1. Il client specifica all’interno del parametro “create”, contenuto nell’oggetto di argomenti della chiamata di metodo, le proprietà “metadata” dell’oggetto “Email” da inviare, tra le quali troviamo la proprietà “mailboxIds”, la quale contiene l’insieme degli ID delle caselle di posta a cui appartiene questa email, e la proprietà “keywords”, la quale indica eventuali parole chiave associate all’email;

#### **3.5.48.2.1.3.1.2) UC48.2.1.3.1.2 - Inserimento delle proprietà “header fields” di un oggetto “Email” all’interno del parametro “create” del metodo “Email/set” necessario alla creazione di una email**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Email/set”, necessario per la creazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “create”. All’interno di questo parametro vanno definite le proprietà dell’email da inviare. Tra queste troviamo le proprietà “header”;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito le proprietà “header” di un oggetto “Email” all’interno del parametro “create”, contenuto a sua volta all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta;
- **Scenario principale:**
  1. Il client specifica all’interno del parametro “create”, contenuto nell’oggetto di argomenti della chiamata di metodo, le proprietà “header” dell’oggetto “Email” da inviare, tra le quali troviamo le seguenti:
    - from: L’indirizzo email del mittente;
    - to: L’elenco degli indirizzi email dei destinatari principali;
    - cc: L’elenco degli indirizzi email dei destinatari in copia;
    - bcc: L’elenco degli indirizzi email dei destinatari in copia nascosta;
    - subject: L’oggetto o titolo del messaggio;
    - sentAt: La data e l’ora in cui il messaggio è stato inviato;
    - inReplyTo: L’elenco degli identificatori dei messaggi a cui si sta rispondendo, se presenti;
    - replyTo: L’indirizzo email a cui rispondere, se diverso dal mittente;
    - references: L’elenco degli identificatori dei messaggi a cui l’email si sta riferendo, se esistenti;
    - messageId: Un identificatore unico assegnato al messaggio;

#### **3.5.48.2.1.3.1.3) UC48.2.1.3.1.3 - Inserimento delle proprietà “body parts” di un oggetto “Email” all’interno del parametro “create” del metodo “Email/set” necessario alla creazione di una email**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Email/set”, necessario per la creazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “create”. All’interno di questo parametro vanno definite le proprietà dell’email da inviare. Tra queste troviamo le proprietà “body parts”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito le proprietà “body parts” di un oggetto “Email” all’interno del parametro “create”, contenuto a sua volta all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta;
- **Scenario principale:**
  1. Il client specifica all’interno del parametro “create”, contenuto nell’oggetto di argomenti della chiamata di metodo, le proprietà “body parts” dell’oggetto “Email” da inviare, tra le quali troviamo le seguenti:
    - bodyStructure: La struttura MIME completa del corpo del messaggio;
    - bodyValues: Mappa di partId a un oggetto EmailBodyValue per alcune parti o tutte le parti di tipo text;

- **textBody**: Lista di parti text/plain, text/html, image/, audio/ e/o video da visualizzare come corpo del messaggio, con una preferenza per text/plain;
- **htmlBody**: Lista di parti text/plain, text/html, image/, audio/ e/o video da visualizzare come corpo del messaggio, con una preferenza per text/html;
- **preview**: Frammento di testo del corpo del messaggio, inteso per essere visualizzato come anteprima quando si elencano i messaggi nello store di posta;

### 3.5.48.2.1.4) UC48.2.1.4 - Inserimento del parametro “update” del metodo “Email/set” necessario alla creazione di una email

- **Attore principale**: Client di posta elettronica autenticato;
- **Descrizione**: Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Email/set”, necessario per la creazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “update”;
- **Precondizioni**: Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni**: Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “update”;
- **Scenario principale**:
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “update”, il quale viene impostato a null, dato che nella creazione di un’email nessun oggetto deve essere aggiornato;

### 3.5.48.2.1.5) UC48.2.1.5 - Inserimento del parametro “destroy” del metodo “Email/set” necessario alla creazione di una email

- **Attore principale**: Client di posta elettronica autenticato;
- **Descrizione**: Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Email/set”, necessario per la creazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “destroy”;
- **Precondizioni**: Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni**: Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “destroy”;
- **Scenario principale**:
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “destroy”, il quale viene impostato a null, dato che nella creazione di un’email nessun oggetto deve essere distrutto;

### 3.5.48.2.2) UC48.2.2 - Inserimento dei parametri del metodo “EmailSubmission/set” necessari all’invio di una email

- **Attore principale**: Client di posta elettronica autenticato;
- **Descrizione**: Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo

“EmailSubmission/set”, necessario per l’invio di una email, per il quale vanno specificati i parametri necessari;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “EmailSubmission/set”, contenuto nell’array “methodCalls” della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all’interno dell’oggetto di argomenti della chiamata di metodo “EmailSubmission/set”, la quale è contenuta nell’array di chiamate di metodo denominato “methodCalls”, contenuto a sua volta all’interno dell’oggetto Request;

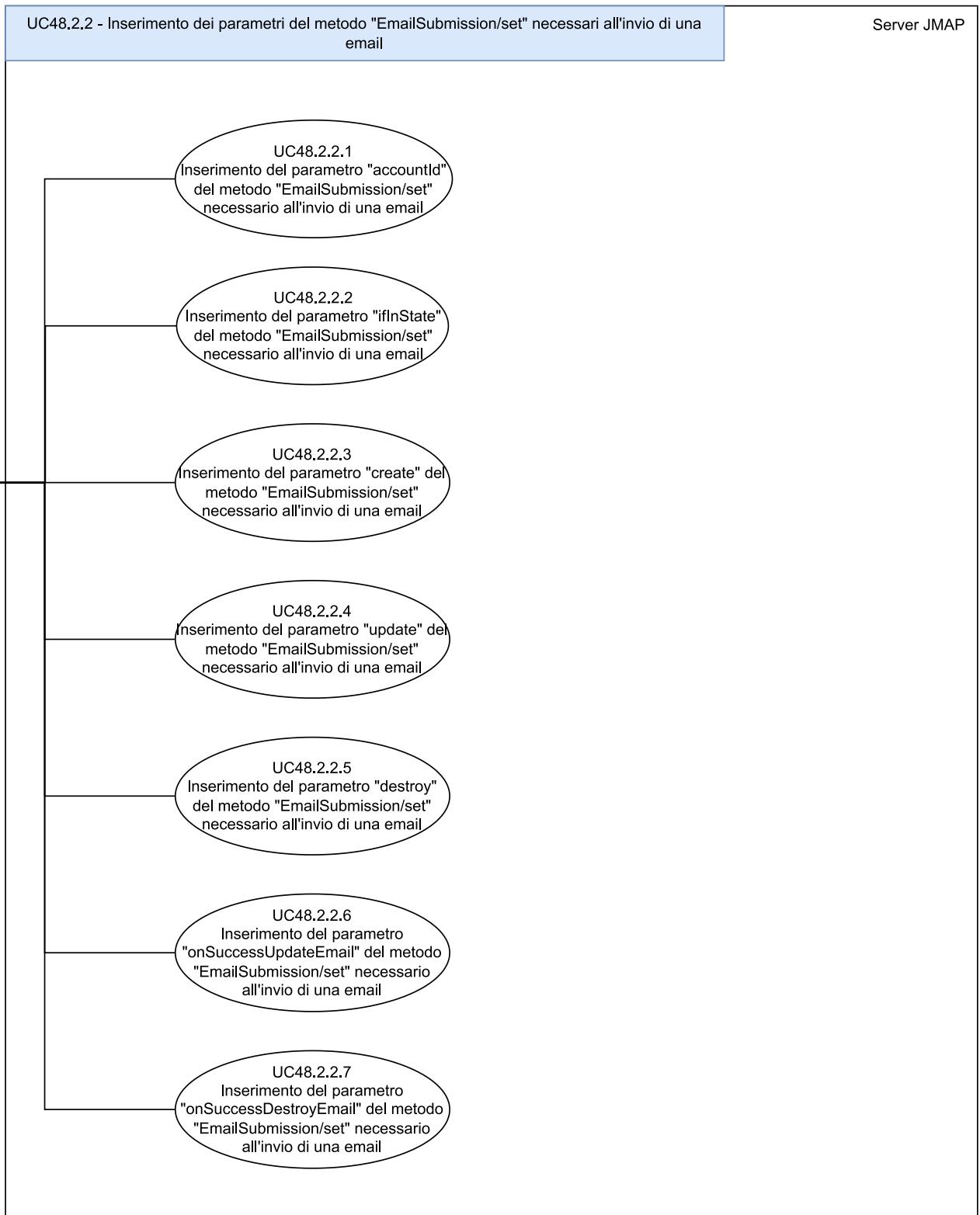


Figura 8: Sottocasi UC48.2.2 - Inserimento dei parametri del metodo "EmailSubmission/set" necessari all'invio di una email

**3.5.48.2.2.1) UC48.2.2.1 - Inserimento del parametro "accountId" del metodo "EmailSubmission/set" necessario all'invio di una email**

- **Attore principale:** Client di posta elettronica autenticato;

- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "EmailSubmission/set", necessario per l'invio di una email, per il quale vanno specificati i parametri necessari. Uno di questi è "accountId";
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "EmailSubmission/set", contenuto nell'array "methodCalls" della richiesta, il parametro "accountId";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "accountId", il quale rappresenta l'identificativo dell'account da utilizzare;

### 3.5.48.2.2.2) UC48.2.2.2 - Inserimento del parametro "ifInState" del metodo "EmailSubmission/set" necessario all'invio di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "EmailSubmission/set", necessario per l'invio di una email, per il quale vanno specificati i parametri necessari. Uno di questi è "ifInState";
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "EmailSubmission/set", contenuto nell'array "methodCalls" della richiesta, il parametro "ifInState";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "ifInState", il quale è un meccanismo di controllo che impedisce la sovrascrittura di dati obsoleti o incoerenti sul server. Esso garantisce che la richiesta venga eseguita solo se il client è a conoscenza dello stato attuale degli oggetti e che tale stato sia valido al momento della richiesta;

### 3.5.48.2.2.3) UC48.2.2.3 - Inserimento del parametro "create" del metodo "EmailSubmission/set" necessario all'invio di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "EmailSubmission/set", necessario per l'invio di una email, per il quale vanno specificati i parametri necessari. Uno di questi è "create";
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "EmailSubmission/set", contenuto nell'array "methodCalls" della richiesta, il parametro "create";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "create", il quale è utilizzato per specificare l'intenzione di creare l'invio di una email. Questo parametro è una mappa, in cui le chiavi sono identificatori temporanei di creazione (creation id) assegnati dal client ed i valori sono oggetti EmailSubmission da creare. La definizione di un oggetto EmailSubmission può contenere valori predefiniti per le proprietà. Qualsiasi proprietà con

un valore predefinito può essere omessa dal client. Inoltre Il client DEVE omettere qualsiasi proprietà che può essere impostata solo dal server;

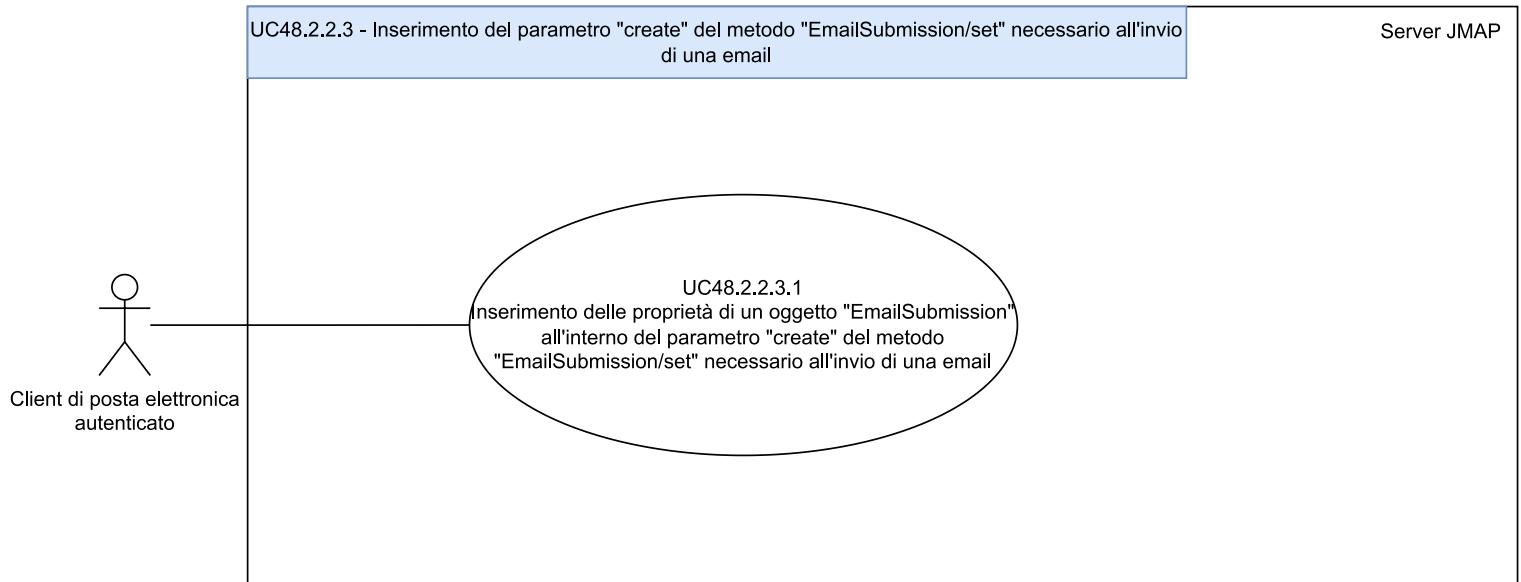


Figura 9: Sottocasi UC48.2.2.3 - Inserimento del parametro “create” del metodo “EmailSubmission/set” necessario all’invio di una email

**3.5.48.2.2.3.1) UC48.2.2.3.1 - Inserimento delle proprietà di un oggetto “EmailSubmission” all’interno del parametro “create” del metodo “EmailSubmission/set” necessario all’invio di una email**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “EmailSubmission/set”, necessario per l’invio di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “create”. All’interno di questo parametro vanno definite le proprietà dell’invio dell’email;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito le proprietà di un oggetto “EmailSubmission” all’interno del parametro “create”, contenuto a sua volta all’interno dell’oggetto di argomenti della chiamata di metodo “EmailSubmission/set”, contenuto nell’array “methodCalls” della richiesta;
- **Scenario principale:**
  1. Il client specifica all’interno del parametro “create”, contenuto nell’oggetto di argomenti della chiamata di metodo, le proprietà dell’oggetto “EmailSubmission” da creare, tra le quali troviamo le seguenti:
    - identityId: Identificatore dell’identità associata a questa sottomissione;
    - emailId: Identificatore dell’email da inviare. Corrisponderà all’email creata in precedenza;
    - envelope: Oggetto Envelope contenente informazioni per l’invio tramite [SMTP](#), ovvero:
      - mailFrom: Indirizzo email da utilizzare come mittente nell’invio;
      - rcptTo: Array di indirizzi email a cui inviare il messaggio;

- undoStatus: Stato che indica se la sottomissione può essere annullata. Può essere “pending”, “final” o “canceled”;

#### 3.5.48.2.2.4) UC48.2.2.4 - Inserimento del parametro “update” del metodo “EmailSubmission/set” necessario all’invio di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “EmailSubmission/set”, necessario per l’invio di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “update”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “EmailSubmission/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “update”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “update”, il quale viene impostato a null, dato che nell’invio di un’email nessun oggetto deve essere aggiornato;

#### 3.5.48.2.2.5) UC48.2.2.5 - Inserimento del parametro “destroy” del metodo “EmailSubmission/set” necessario all’invio di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “EmailSubmission/set”, necessario per l’invio di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “destroy”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “EmailSubmission/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “destroy”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “destroy”, il quale viene impostato a null, dato che nell’invio di un’email nessun oggetto deve essere distrutto;

#### 3.5.48.2.2.6) UC48.2.2.6 - Inserimento del parametro “onSuccessUpdateEmail” del metodo “EmailSubmission/set” necessario all’invio di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “EmailSubmission/set”, necessario per l’invio di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “onSuccessUpdateEmail”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;

- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “EmailSubmission/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “onSuccessUpdateEmail”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “onSuccessUpdateEmail”, il quale è una mappa che associa gli identificativi delle sottomissioni di email (EmailSubmission) a un oggetto PatchObject contenente proprietà da aggiornare sull'oggetto Email (Email) corrispondente quando la creazione, l'aggiornamento o la cancellazione avviene con successo. Questo meccanismo consente di specificare azioni aggiuntive da intraprendere sull'oggetto Email dopo che la sottomissione di email associata ha avuto successo;

#### **3.5.48.2.2.7) UC48.2.2.7 - Inserimento del parametro “onSuccessDestroyEmail” del metodo “EmailSubmission/set” necessario all'invio di una email**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di inviare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “EmailSubmission/set”, necessario per l'invio di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “onSuccessDestroyEmail”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “EmailSubmission/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “onSuccessDestroyEmail”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “onSuccessDestroyEmail”, il quale è una lista di identificativi di sottomissioni di email (EmailSubmission) per le quali l'oggetto Email con l'emailId corrispondente dovrebbe essere distrutto se l'operazione di creazione, aggiornamento o cancellazione avviene con successo. Questo meccanismo consente di specificare azioni aggiuntive da intraprendere sull'oggetto Email dopo che la sottomissione di email associata ha avuto successo;

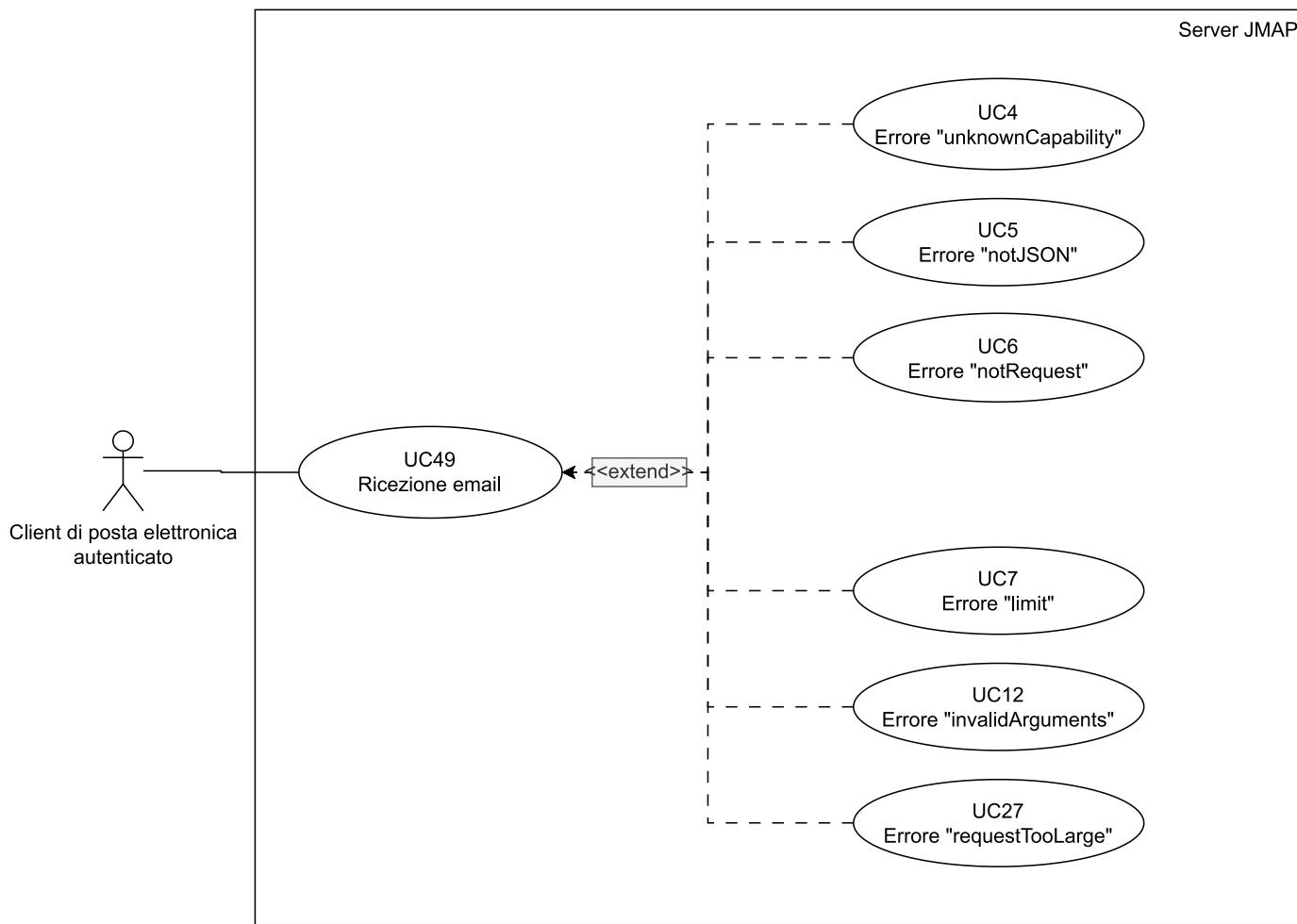


Figura 10: UC49 - Ricezione email

### 3.5.49) UC49 - Ricezione email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una email per visualizzarne il dettaglio, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** L'email è stata ricevuta con successo ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per ricevere un'email;
  2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
  3. Il server elabora la richiesta API e invia una risposta JSON che contiene l'email desiderata oppure una risposta che indica il motivo del fallimento;
- **Estensioni:**
  - Errore "unknownCapability";
  - Errore "notJSON";
  - Errore "notRequest";
  - Errore "limit";

- Errore “requestTooLarge”;
- Errore “invalidArguments”;
- **Inclusioni:** /
- **Generalizzazioni:** /

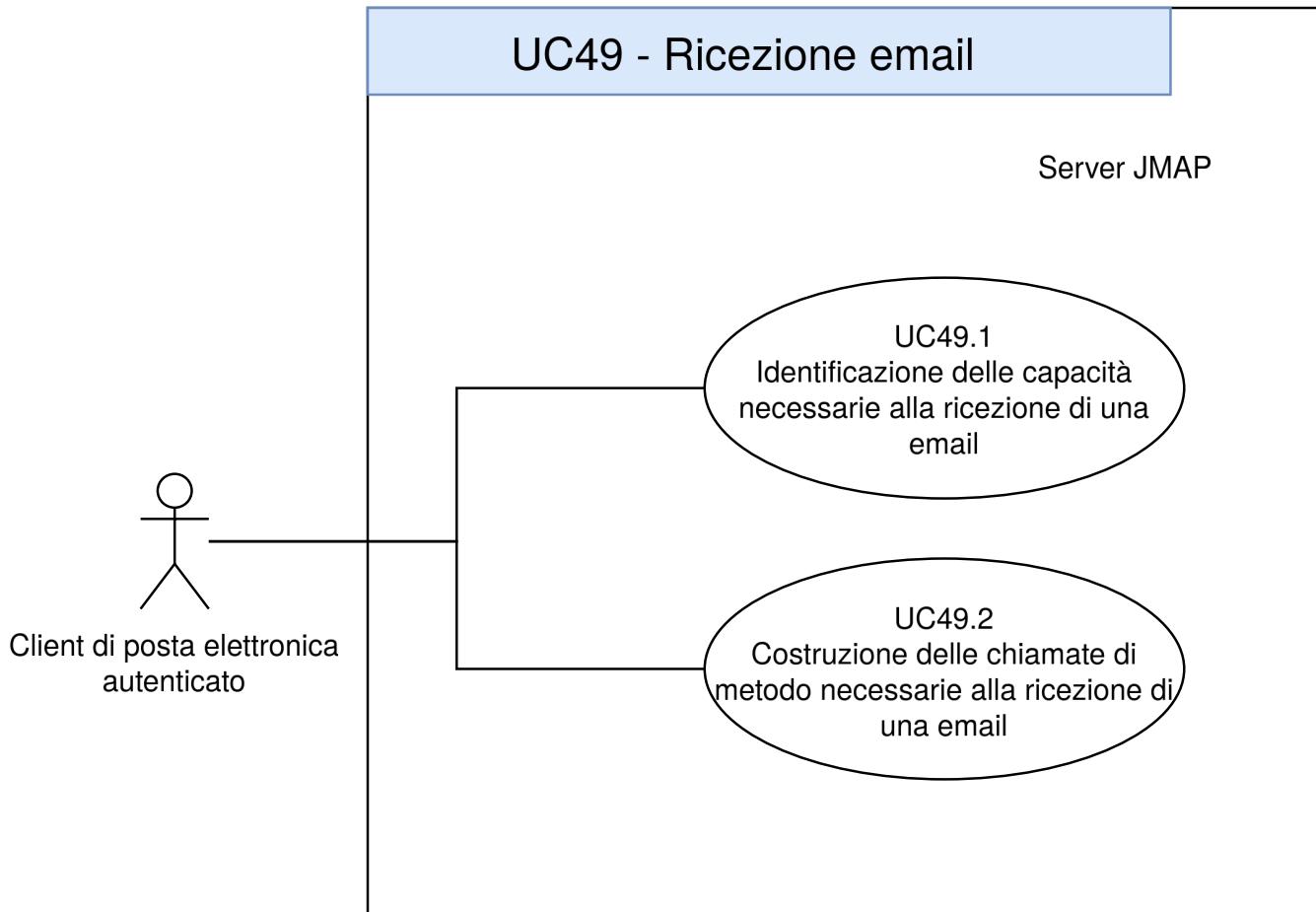


Figura 11: UC49 - Sottocasi ricezione email

### 3.5.49.1) UC49.1 - Identificazione delle capacità necessarie alla ricezione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una email per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per ricevere l’email;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all’interno della richiesta le capacità JMAP necessarie per ricevere l’email;
- **Scenario principale:**
  1. Il client include nella sezione “using” dell’oggetto Request le capacità JMAP “urn:ietf:params:jmap:core” e “urn:ietf:params:jmap:mail”;

### 3.5.49.2) UC49.2 - Costruzione delle chiamate di metodo necessarie alla ricezione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una email per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per ricevere l’email;

no della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per ricevere l'email;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'array "methodCalls" della richiesta le chiamate di metodo necessarie per ricevere l'email;
- **Scenario principale:**
  1. Il client crea un array di chiamate di metodo denominato "methodCalls" internamente all'oggetto Request, all'interno del quale inserisce il metodo "Email/get", un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

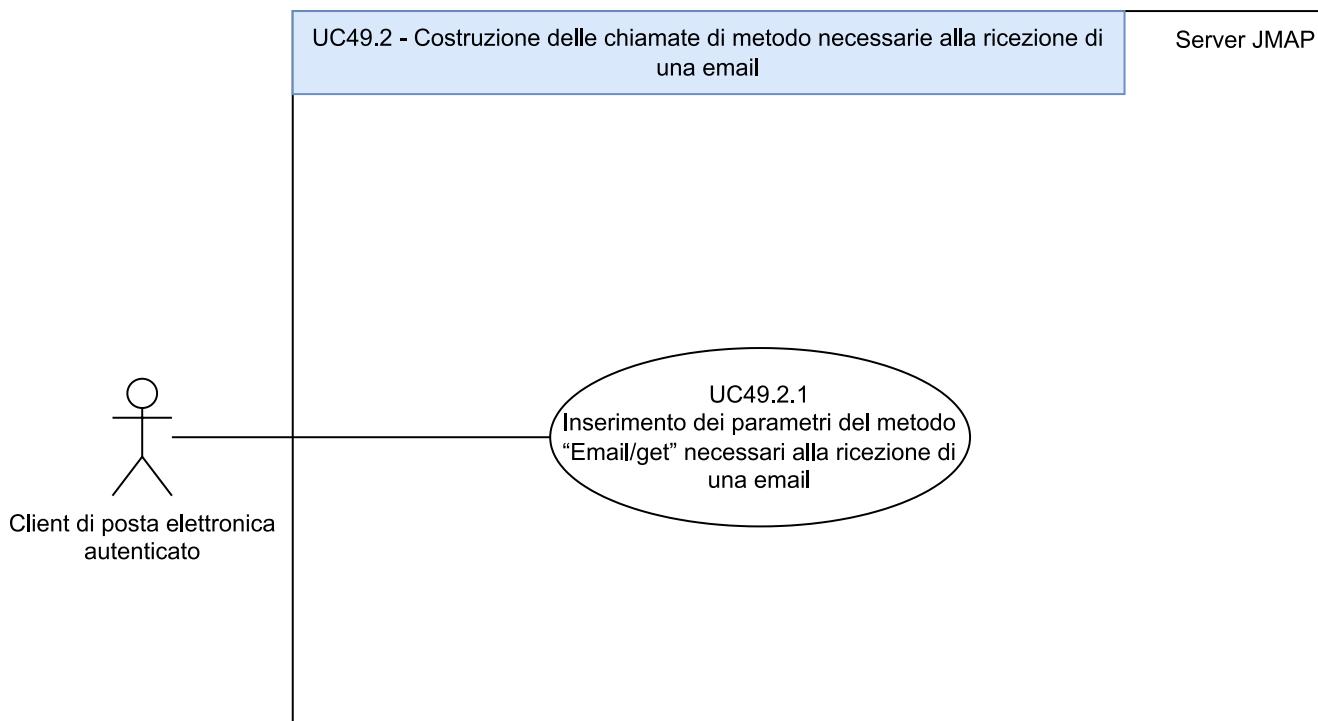


Figura 12: Sottocasi UC49.2 - Costruzione delle chiamate di metodo necessarie alla ricezione di una email

### 3.5.49.2.1) UC49.2.1 - Inserimento dei parametri del metodo "Email/get" necessari alla ricezione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una email per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Email/get", necessario per la ricezione di una email, per il quale vanno specificati i parametri necessari;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Email/get", contenuto nell'array "methodCalls" della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all'interno dell'oggetto di argomenti della chiamata di metodo "Email/get", la quale è contenuta nell'array di chiamate di metodo denominato "methodCalls", contenuto a sua volta all'interno dell'oggetto Request;

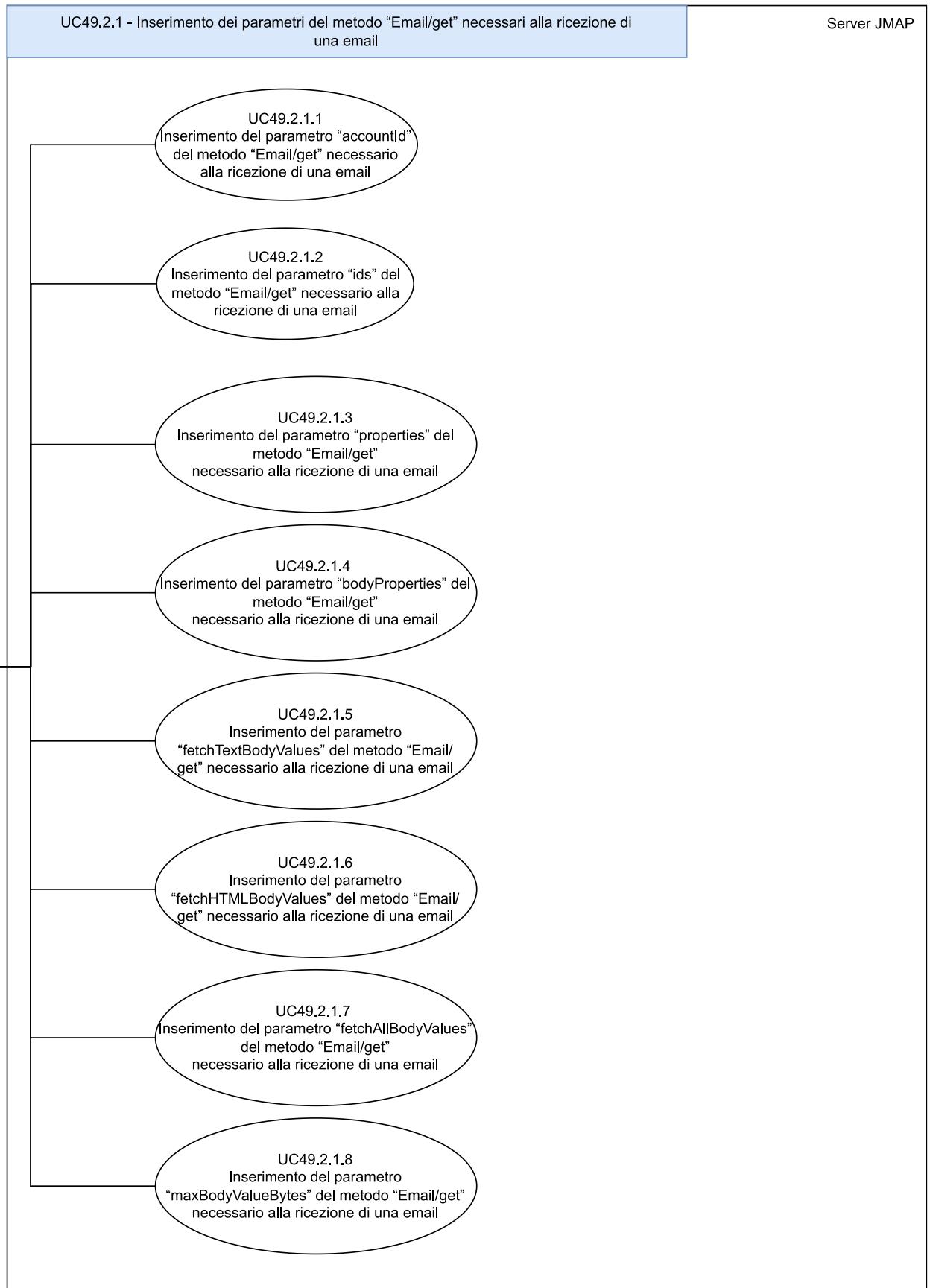


Figura 13: Sottocasi UC49.2.1 - Inserimento dei parametri del metodo "Email/get" necessari alla ricezione di una email

### 3.5.49.2.1.1) UC49.2.1.1 - Inserimento del parametro “accountId” del metodo “Email/get” necessario alla ricezione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una email per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/get”, necessario per la ricezione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “accountId”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/get”, contenuto nell’array “methodCalls” della richiesta, il parametro “accountId”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “accountId”, il quale rappresenta l’identificativo dell’account da utilizzare;

### 3.5.49.2.1.2) UC49.2.1.2 - Inserimento del parametro “ids” del metodo “Email/get” necessario alla ricezione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una email per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/get”, necessario per la ricezione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “ids”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/get”, contenuto nell’array “methodCalls” della richiesta, il parametro “ids”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “ids”, il quale è un array che contiene solamente l’identificativo dell’email da ricevere;

### 3.5.49.2.1.3) UC49.2.1.3 - Inserimento del parametro “properties” del metodo “Email/get” necessario alla ricezione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una email per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/get”, necessario per la ricezione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “properties”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/get”, contenuto nell’array “methodCalls” della richiesta, il parametro “properties”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “properties”, il quale è un array che, se fornito, serve per restituire solo le proprietà dell’email elencate nell’array stesso. Se è nullo, vengono restituite tutte le proprietà dell’email. La proprietà

“id” dell’email viene sempre restituita, anche se non richiesta esplicitamente. Se viene richiesta una proprietà non valida, la chiamata DEVE essere respinta con un errore “invalidArguments”;

#### 3.5.49.2.1.4) UC49.2.1.4 - Inserimento del parametro “bodyProperties” del metodo “Email/get” necessario alla ricezione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una email per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/get”, necessario per la ricezione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “bodyProperties”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/get”, contenuto nell’array “methodCalls” della richiesta, il parametro “bodyProperties”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “bodyProperties”, il quale è un elenco di proprietà da recuperare per ciascun EmailBodyPart restituito. Se omesso viene posto a dei valori di default;

#### 3.5.49.2.1.5) UC49.2.1.5 - Inserimento del parametro “fetchTextBodyValues” del metodo “Email/get” necessario alla ricezione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una email per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/get”, necessario per la ricezione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “fetchTextBodyValues”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/get”, contenuto nell’array “methodCalls” della richiesta, il parametro “fetchTextBodyValues”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “fetchTextBodyValues”, il quale è un flag, impostato di default a false, che se viene impostato a true fa sì che la proprietà bodyValues includa ogni parte text/\* contenuta nella proprietà textBody;

#### 3.5.49.2.1.6) UC49.2.1.6 - Inserimento del parametro “fetchHTMLBodyValues” del metodo “Email/get” necessario alla ricezione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una email per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/get”, necessario per la ricezione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “fetchHTMLBodyValues”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;

- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Email/get”, contenuto nell'array “methodCalls” della richiesta, il parametro “fetchHTMLBodyValues”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “fetchHTMLBodyValues”, il quale è un flag, impostato di default a false, che se viene impostato a true fa sì che la proprietà bodyValues includa ogni parte text/\* contenuta nella proprietà htmlBody;

### 3.5.49.2.1.7) UC49.2.1.7 - Inserimento del parametro “fetchAllBodyValues” del metodo “Email/get” necessario alla ricezione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una email per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/get”, necessario per la ricezione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “fetchAllBodyValues”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Email/get”, contenuto nell'array “methodCalls” della richiesta, il parametro “fetchAllBodyValues”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “fetchAllBodyValues”, il quale è un flag, impostato di default a false, che se viene impostato a true fa sì che la proprietà bodyValues includa ogni parte text/\* contenuta nella proprietà bodyStructure;

### 3.5.49.2.1.8) UC49.2.1.8 - Inserimento del parametro “maxBodyValueBytes” del metodo “Email/get” necessario alla ricezione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una email per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/get”, necessario per la ricezione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “maxBodyValueBytes”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Email/get”, contenuto nell'array “methodCalls” della richiesta, il parametro “maxBodyValueBytes”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “maxBodyValueBytes”, il quale è un numero che, se posto maggiore di zero, fa sì che la proprietà value di qualsiasi oggetto EmailBodyValue restituito in bodyValues debba essere troncata, se necessario, in modo che non superi questo numero di ottetti in dimensione. Se posto a 0 (impostazione predefinita) non si verificherà nessun troncamento;

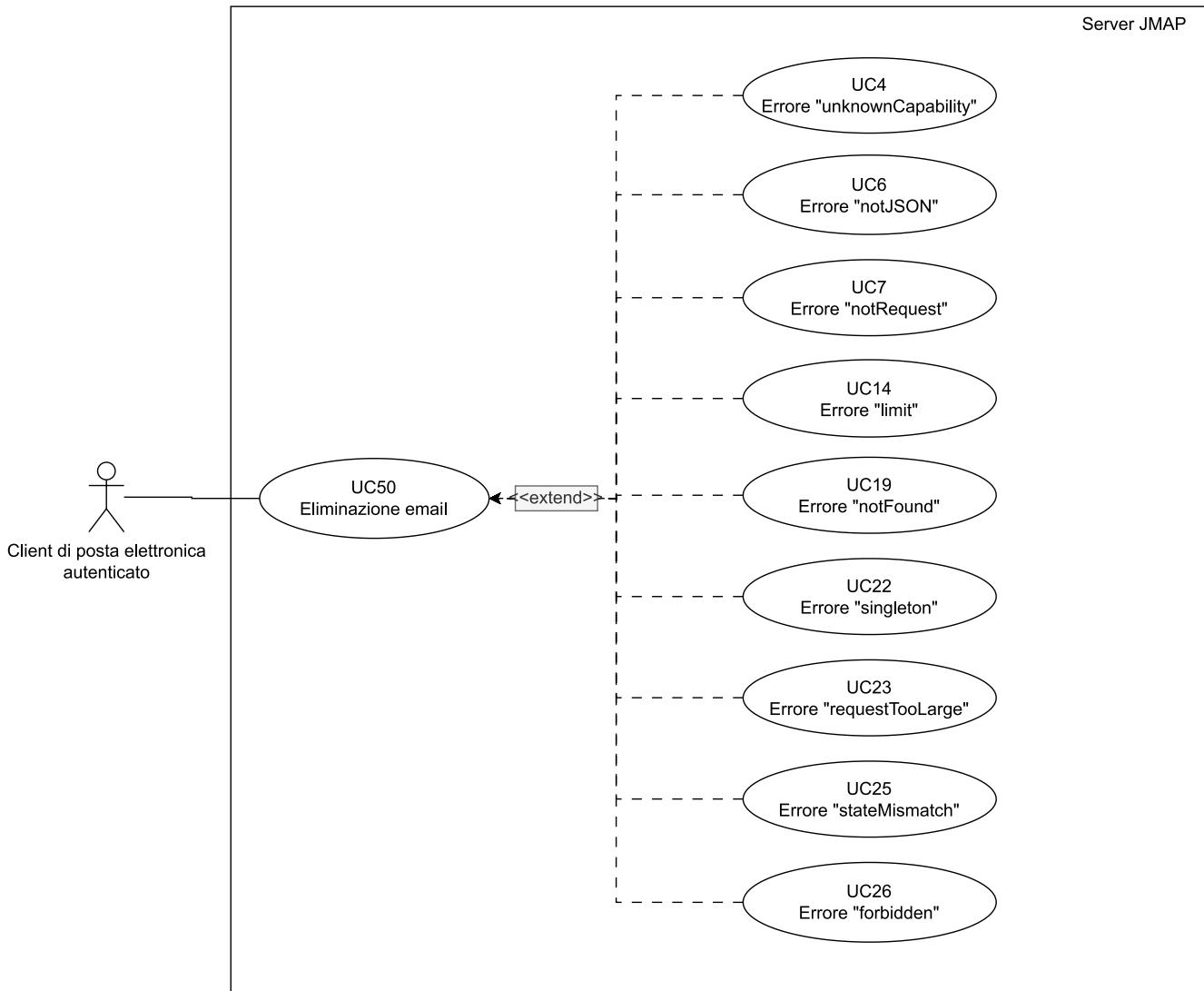


Figura 14: UC50 - Eliminazione email

### 3.5.50) UC50 - Eliminazione email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una email, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** L'email è stata eliminata con successo ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per eliminare un'email;
  2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
  3. Il server elabora la richiesta API e invia una risposta JSON che contiene l'esito dell'operazione di eliminazione dell'email;
- **Estensioni:**
  - Errore "unknownCapability";
  - Errore "notJSON";

- Errore “notRequest”;
- Errore “limit”;
- Errore “forbidden”;
- Errore “notFound”;
- Errore “singleton”;
- Errore “requestTooLarge”;
- Errore “stateMismatch”;
- **Inclusioni:** /
- **Generalizzazioni:** /

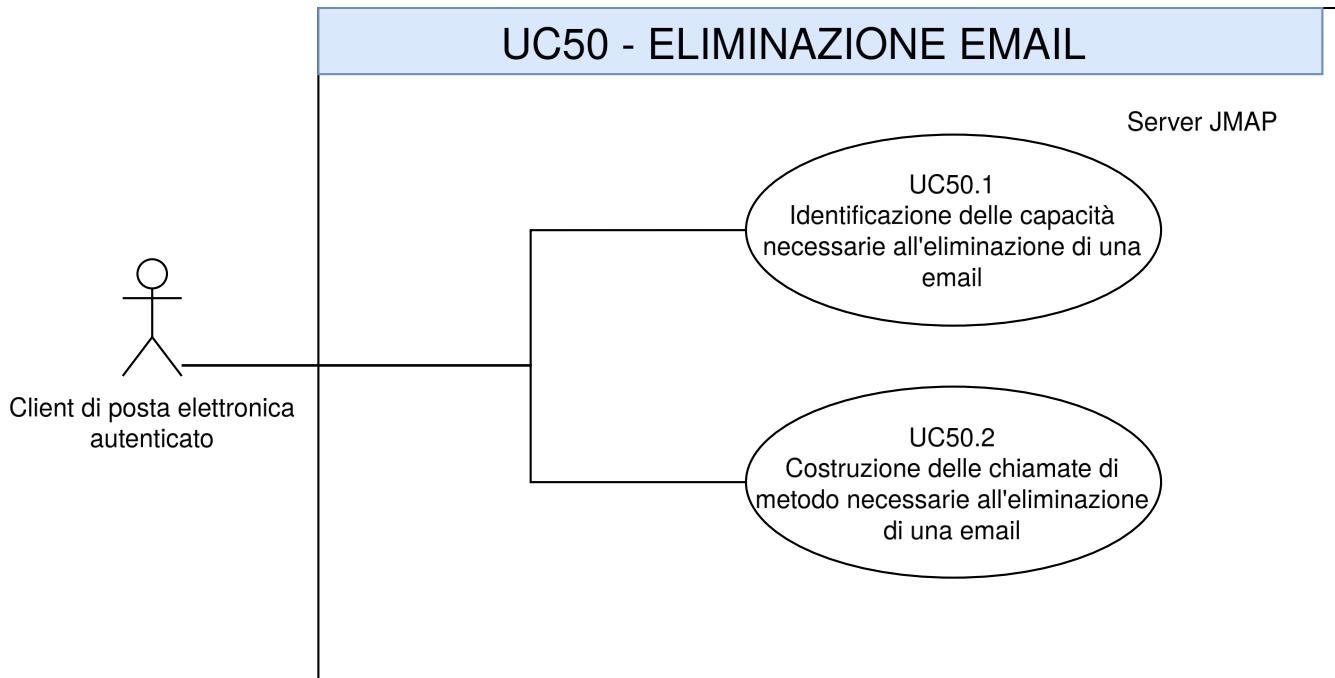


Figura 15: Sottocasi UC50 - Eliminazione email

### 3.5.50.1) UC50.1 - Identificazione delle capacità necessarie all'eliminazione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per l'eliminazione di una email;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all'interno della richiesta le capacità JMAP necessarie per l'eliminazione di una email;
- **Scenario principale:**
  1. Il client include nella sezione “using” dell'oggetto Request le capacità JMAP “urn:ietf:params:jmap:core” e “urn:ietf:params:jmap:mail”;

### 3.5.50.2) UC50.2 - Costruzione delle chiamate di metodo necessarie all'eliminazione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session.

Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per l'eliminazione di una email;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'array “methodCalls” della richiesta le chiamate di metodo necessarie per l'eliminazione di una email;
- **Scenario principale:**
  1. Il client crea un array di chiamate di metodo denominato “methodCalls” internamente all'oggetto Request, all'interno del quale inserisce il metodo “Email/set”, un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

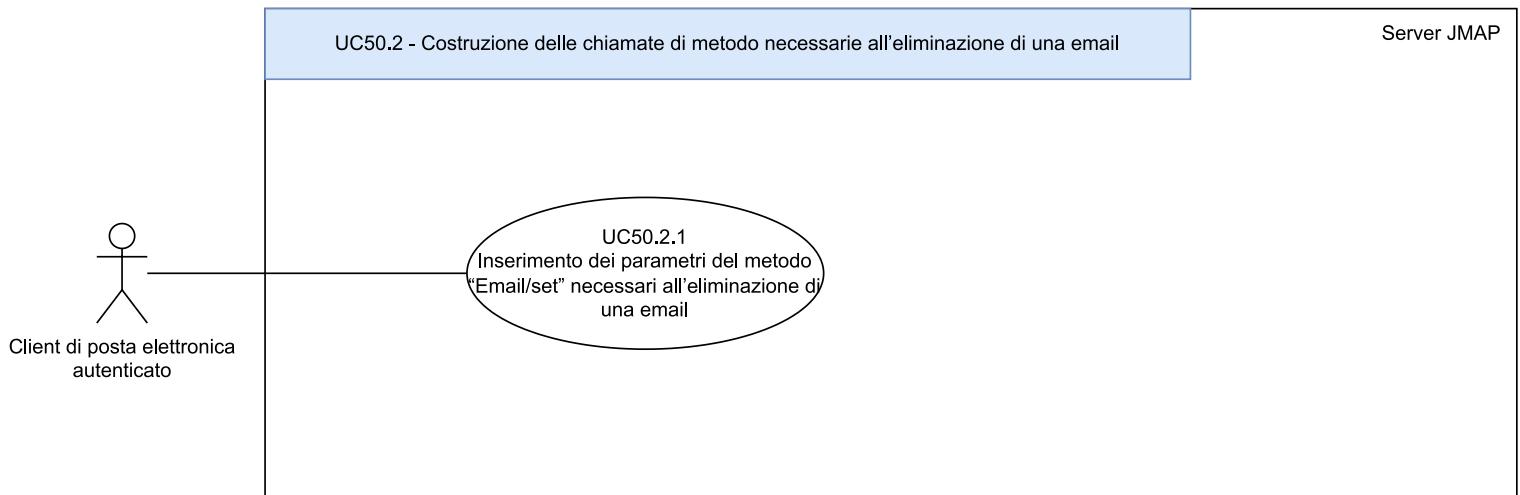


Figura 16: Sottocasi UC50.2 - Costruzione delle chiamate di metodo necessarie all'eliminazione di una email

### 3.5.50.2.1) UC50.2.1 - Inserimento dei parametri del metodo “Email/set” necessari all'eliminazione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per l'eliminazione di una email, per il quale vanno specificati i parametri necessari;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell'array “methodCalls” della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all'interno dell'oggetto di argomenti della chiamata di metodo “Email/set”, la quale è contenuta nell'array di chiamate di metodo denominato “methodCalls”, contenuto a sua volta all'interno dell'oggetto Request;

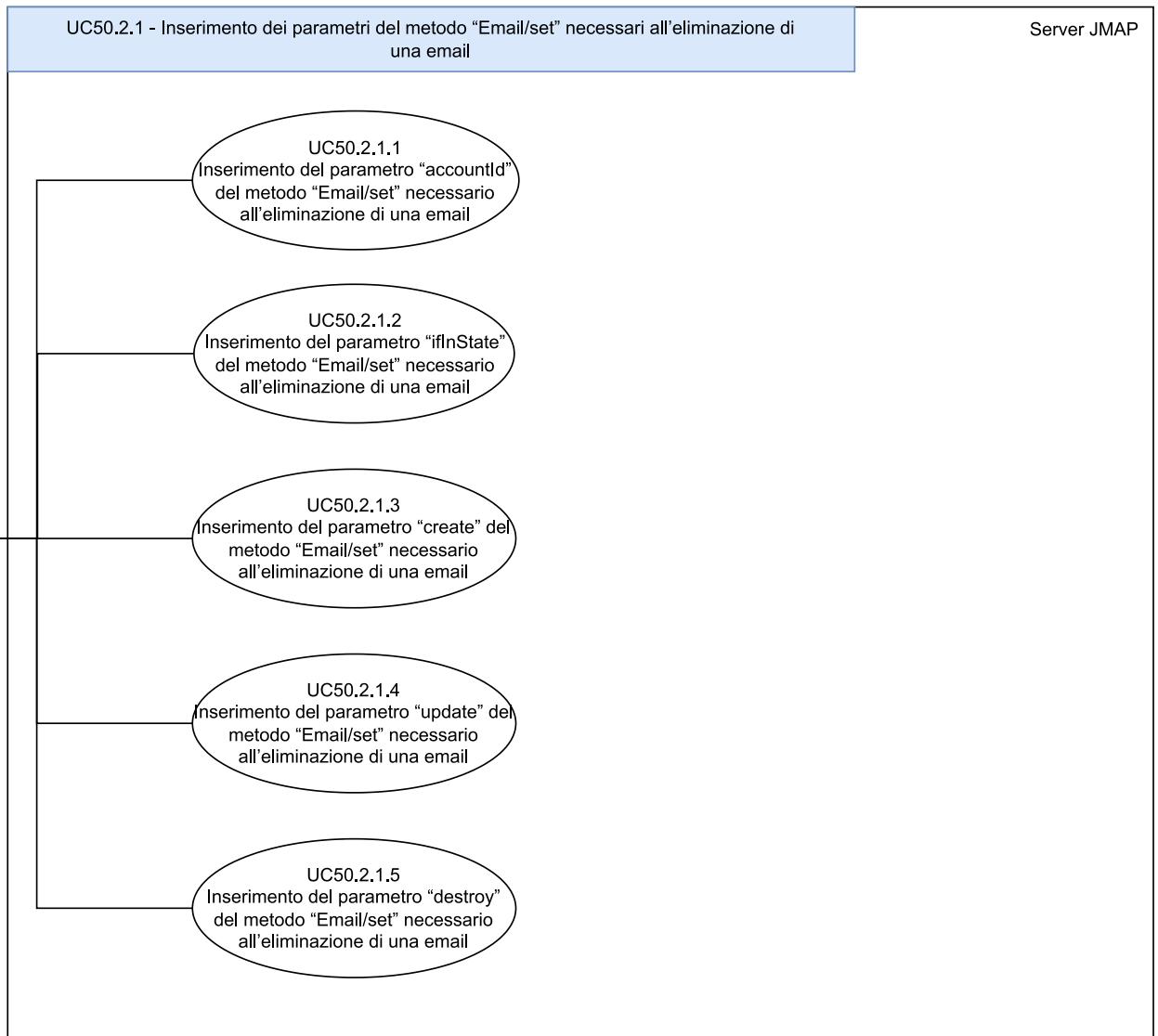


Figura 17: Sottocasi UC50.2.1 - Inserimento dei parametri del metodo “Email/set” necessari all’eliminazione di una email

### 3.5.50.2.1.1) UC50.2.1.1 - Inserimento del parametro “accountId” del metodo “Email/set” necessario all’eliminazione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per l’eliminazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “accountId”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “accountId”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “accountId”, il quale rappresenta l’identificativo dell’account da utilizzare;

### 3.5.50.2.1.2) UC50.2.1.2 - Inserimento del parametro “ifInState” del metodo “Email/set” necessario all’eliminazione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per l’eliminazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “ifInState”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “ifInState”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “ifInState”, il quale è un meccanismo di controllo che impedisce la sovrascrittura di dati obsoleti o incoerenti sul server. Esso garantisce che la richiesta venga eseguita solo se il client è a conoscenza dello stato attuale degli oggetti e che tale stato sia valido al momento della richiesta;

### 3.5.50.2.1.3) UC50.2.1.3 - Inserimento del parametro “create” del metodo “Email/set” necessario all’eliminazione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per l’eliminazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “create”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “create”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “create”, il quale viene impostato a null, dato che nell’eliminazione di un’email nessun oggetto deve essere creato;

### 3.5.50.2.1.4) UC50.2.1.4 - Inserimento del parametro “update” del metodo “Email/set” necessario all’eliminazione di una email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una email, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per l’eliminazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è “update”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “update”;
- **Scenario principale:**

- Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "update", il quale viene impostato a null, dato che nell'eliminazione di un'email nessun oggetto deve essere aggiornato;

### 3.5.50.2.1.5) UC50.2.1.5 - Inserimento del parametro "destroy" del metodo "Email/set" necessario all'eliminazione di una email

- Attore principale:** Client di posta elettronica autenticato;
- Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una email, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Email/set", necessario per l'eliminazione di una email, per il quale vanno specificati i parametri necessari. Uno di questi è "destroy";
- Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Email/set", contenuto nell'array "methodCalls" della richiesta, il parametro "destroy";
- Scenario principale:**
  - Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "destroy", il quale è utilizzato per specificare l'intenzione di eliminare una email. Questo parametro è un array che contiene solamente l'identificativo dell'email da eliminare;

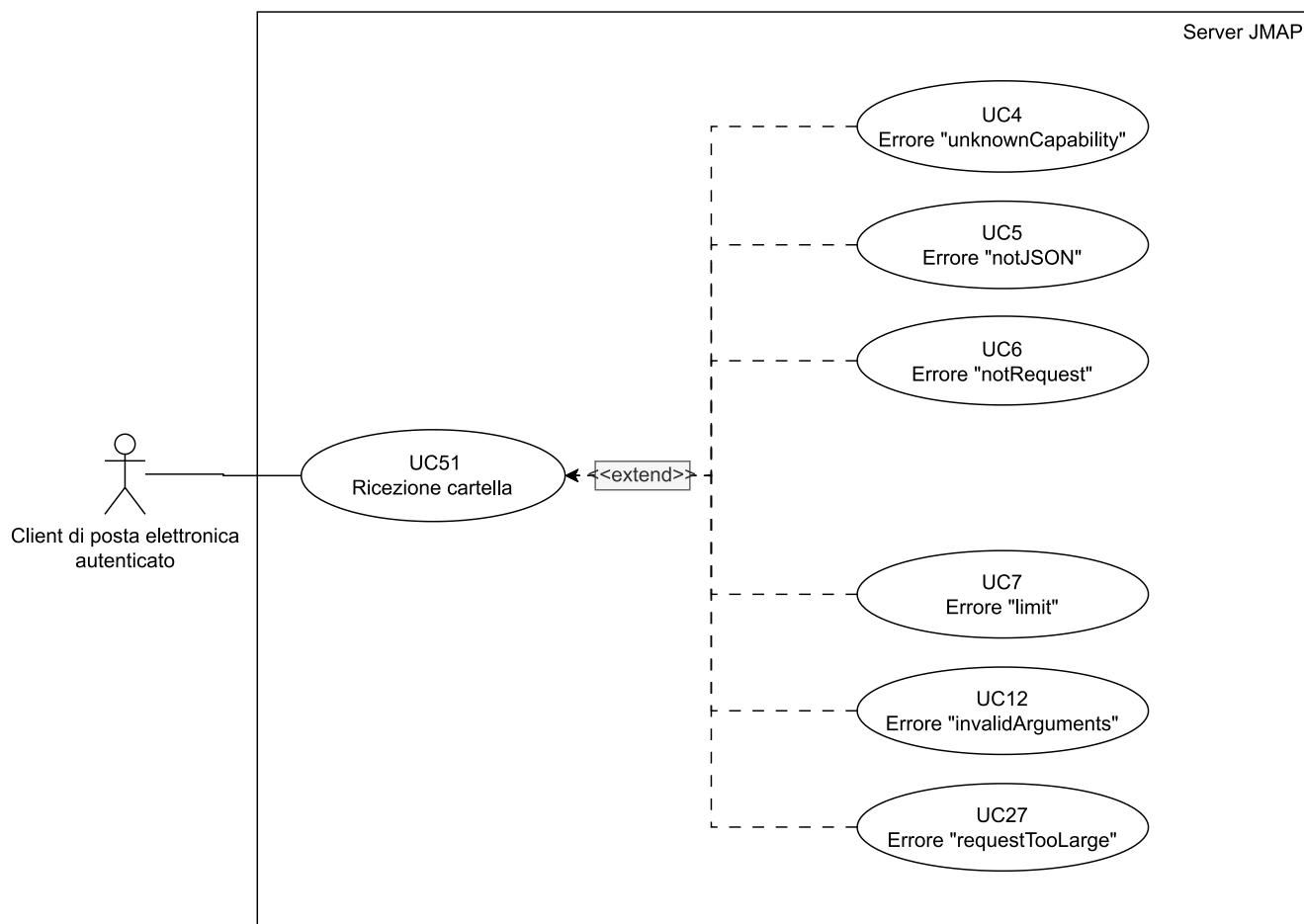


Figura 18: UC51 - Ricezione cartella

### 3.5.51) UC51 - Ricezione cartella

- Attore principale:** Client di posta elettronica autenticato;

- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una cartella per visualizzarne il dettaglio, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** La cartella è stata ricevuta con successo ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per ricevere una cartella;
  2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
  3. Il server elabora la richiesta API e invia una risposta JSON che contiene la cartella desiderata oppure una risposta che indica il motivo del fallimento;
- **Estensioni:**
  - Errore “unknownCapability”;
  - Errore “notJSON”;
  - Errore “notRequest”;
  - Errore “limit”;
  - Errore “requestTooLarge”;
  - Errore “invalidArguments”;
- **Inclusioni:** /
- **Generalizzazioni:** /

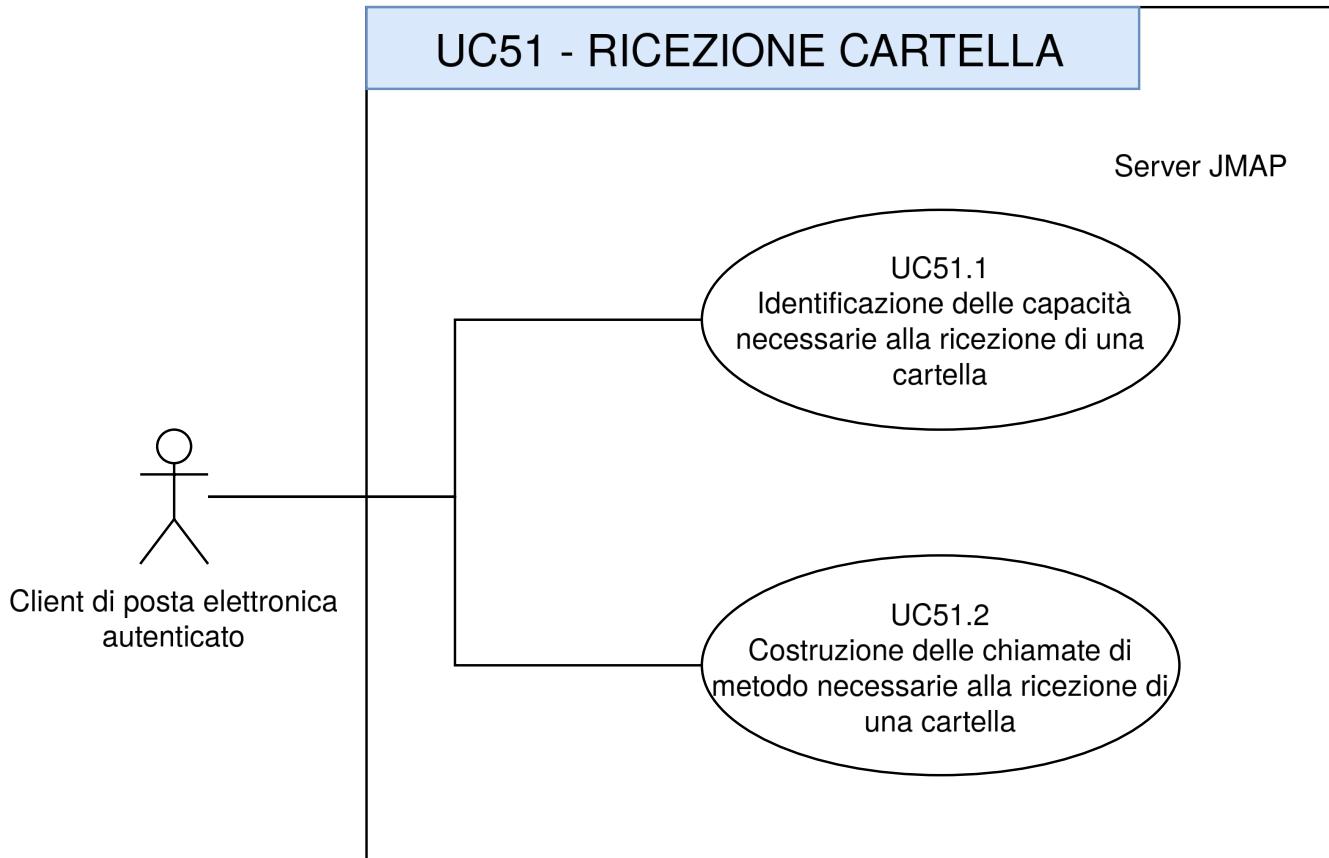


Figura 19: Sottocasi UC51 - Ricezione cartella

### 3.5.51.1) UC51.1 - Identificazione delle capacità necessarie alla ricezione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una cartella per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per ricevere una cartella;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all'interno della richiesta le capacità JMAP necessarie per ricevere una cartella;
- **Scenario principale:**
  1. Il client include nella sezione “using” dell'oggetto Request le capacità JMAP “urn:ietf:params:jmap:core” e “urn:ietf:params:jmap:mail”;

### 3.5.51.2) UC51.2 - Costruzione delle chiamate di metodo necessarie alla ricezione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una cartella per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per ricevere la cartella desiderata;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'array “methodCalls” della richiesta le chiamate di metodo necessarie per ricevere la cartella desiderata;
- **Scenario principale:**
  1. Il client crea un array di chiamate di metodo denominato “methodCalls” internamente all'oggetto Request, all'interno del quale inserisce il metodo “Mailbox/get”, un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

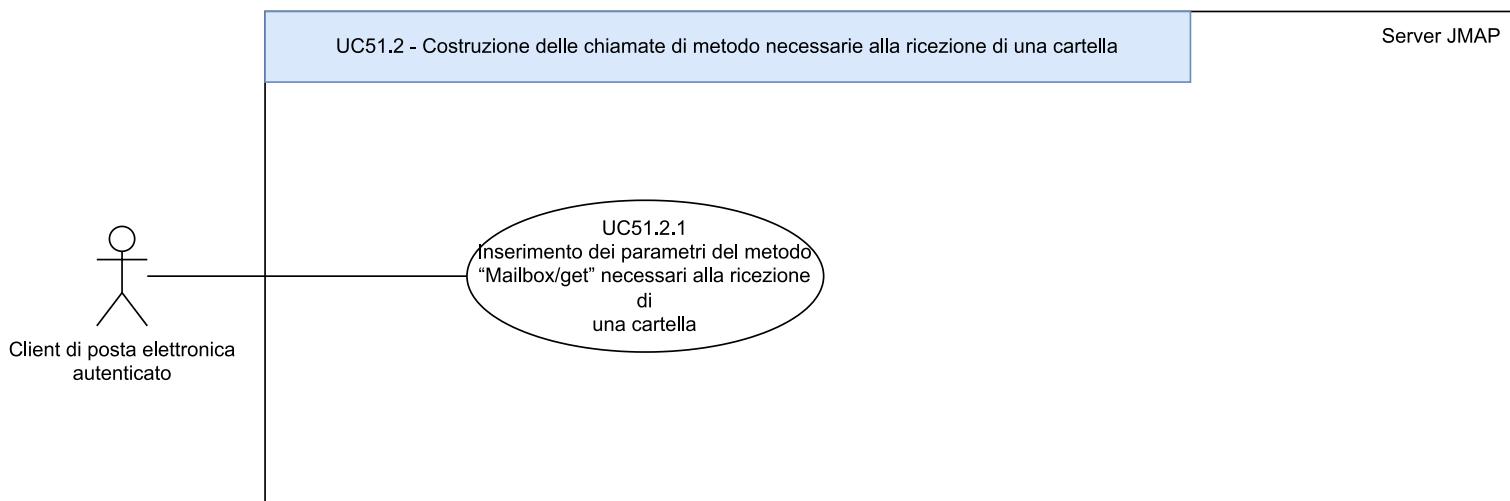


Figura 20: Sottocasi UC51.2 - Costruzione delle chiamate di metodo necessarie alla ricezione di una cartella

#### 3.5.51.2.1) UC51.2.1 - Inserimento dei parametri del metodo “Mailbox/get” necessari alla ricezione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;

- Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una cartella per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Mailbox/get", necessario per la ricezione di una cartella, per il quale vanno specificati i parametri necessari;
- Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/get", contenuto nell'array "methodCalls" della richiesta, i parametri necessari;
- Scenario principale:**
  - Il client specifica gli argomenti all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/get", la quale è contenuta nell'array di chiamate di metodo denominato "methodCalls", contenuto a sua volta all'interno dell'oggetto Request;

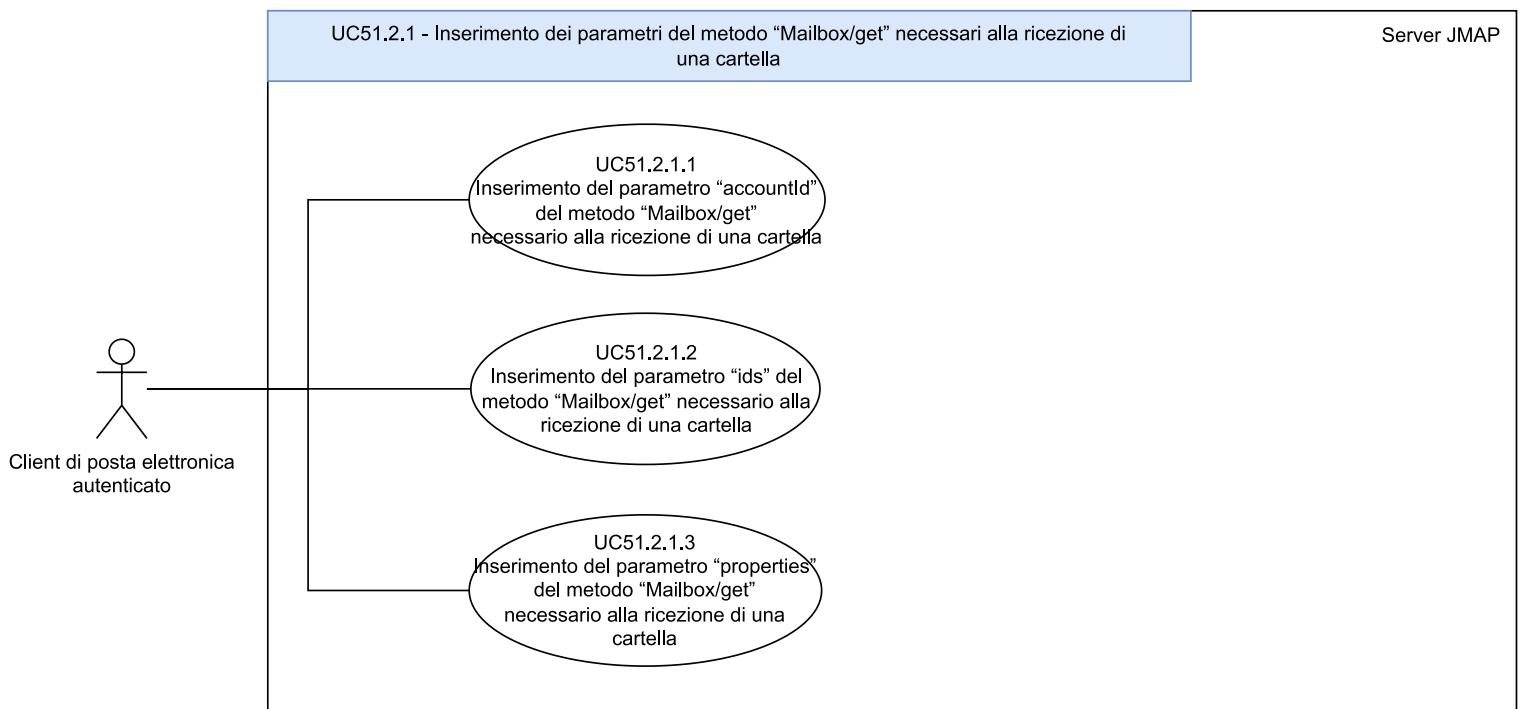


Figura 21: Sottocasi UC51.2.1 - Inserimento dei parametri del metodo "Mailbox/get" necessari alla ricezione di una cartella

### 3.5.51.2.1.1) UC51.2.1.1 - Inserimento del parametro "accountId" del metodo "Mailbox/get" necessario alla ricezione di una cartella

- Attore principale:** Client di posta elettronica autenticato;
- Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una cartella per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Mailbox/get", necessario per la ricezione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è "accountId";
- Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/get", contenuto nell'array "methodCalls" della richiesta, il parametro "accountId";
- Scenario principale:**

- 
1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “accountId”, il quale rappresenta l'identificativo dell'account da utilizzare;

#### **3.5.51.2.1.2) UC51.2.1.2 - Inserimento del parametro “ids” del metodo “Mailbox/get” necessario alla ricezione di una cartella**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una cartella per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/get”, necessario per la ricezione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “ids”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/get”, contenuto nell'array “methodCalls” della richiesta, il parametro “ids”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “ids”, il quale è un array che contiene solamente l'identificativo della cartella da ricevere. Se impostato a null il client intende ricevere tutte le cartelle associate all'account che vuole eseguire la richiesta;

#### **3.5.51.2.1.3) UC51.2.1.3 - Inserimento del parametro “properties” del metodo “Mailbox/get” necessario alla ricezione di una cartella**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di ricevere una cartella per visualizzarne il dettaglio, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/get”, necessario per la ricezione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “properties”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/get”, contenuto nell'array “methodCalls” della richiesta, il parametro “properties”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “properties”, il quale è un array che, se fornito, serve per restituire solo le proprietà della cartella elencate nell'array stesso. Se è nullo, vengono restituite tutte le proprietà della cartella. La proprietà “id” della cartella viene sempre restituita, anche se non richiesta esplicitamente. Se viene richiesta una proprietà non valida, la chiamata DEVE essere respinta con un errore “invalidArguments”;

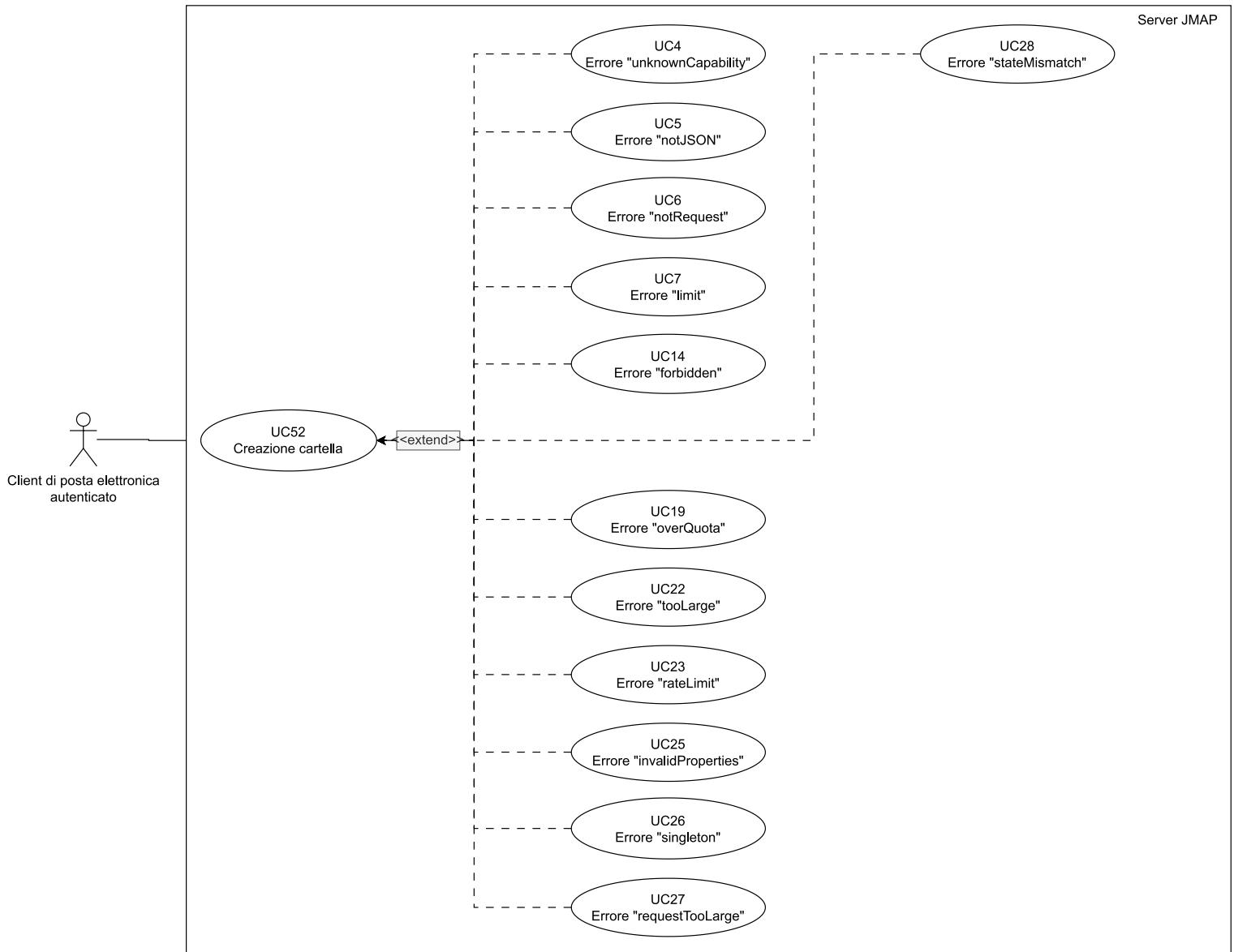


Figura 22: UC52 - Creazione cartella

### 3.5.52) UC52 - Creazione cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di creare una cartella, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** La cartella è stata creata con successo ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per creare una cartella;
  2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
  3. Il server elabora la richiesta API e invia una risposta JSON che contiene l'esito dell'operazione di creazione della cartella;
- **Estensioni:**

- Errore “unknownCapability”;
- Errore “notJSON”;
- Errore “notRequest”;
- Errore “limit”;
- Errore “forbidden”;
- Errore “overQuota”;
- Errore “tooLarge”;
- Errore “rateLimit”;
- Errore “invalidProperties”;
- Errore “singleton”;
- Errore “requestTooLarge”;
- Errore “stateMismatch”;
- **Inclusioni:** /
- **Generalizzazioni:** /

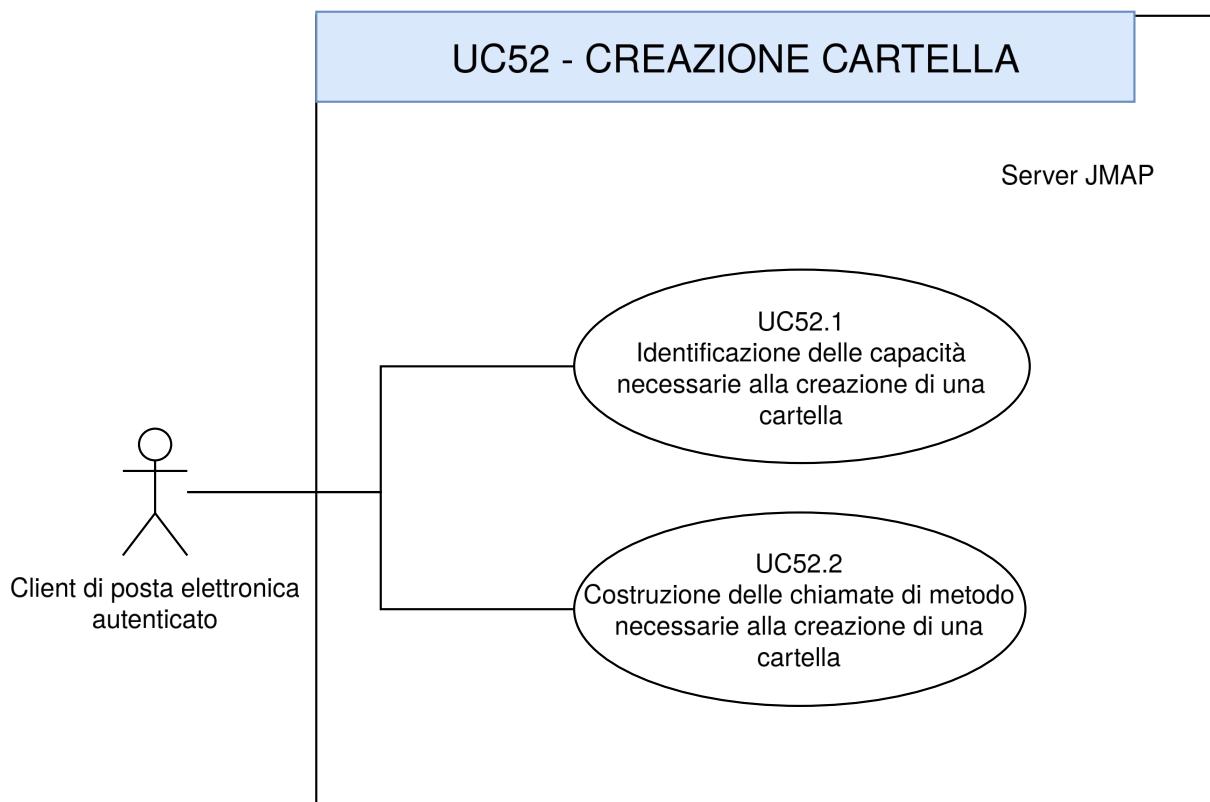


Figura 23: Sottocasi UC52 - Creazione cartella

### 3.5.52.1 UC52.1 - Identificazione delle capacità necessarie alla creazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di creare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per la creazione di una cartella;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all'interno della richiesta le capacità JMAP necessarie per la creazione di una cartella;
- **Scenario principale:**

- Il client include nella sezione “using” dell’oggetto Request le capacità JMAP “urn:ietf:params:jmap:core” e “urn:ietf:params:jmap:mail”;

### 3.5.52.2) UC52.2 - Costruzione delle chiamate di metodo necessarie alla creazione di una cartella

- Attore principale:** Client di posta elettronica autenticato;
- Descrizione:** Un client di posta elettronica, con lo scopo di creare una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per la creazione di una cartella;
- Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- Postcondizioni:** Il client ha inserito all’interno dell’array “methodCalls” della richiesta le chiamate di metodo necessarie per la creazione di una cartella;
- Scenario principale:**
  - Il client crea un array di chiamate di metodo denominato “methodCalls” internamente all’oggetto Request, all’interno del quale inserisce il metodo “Mailbox/set”, un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

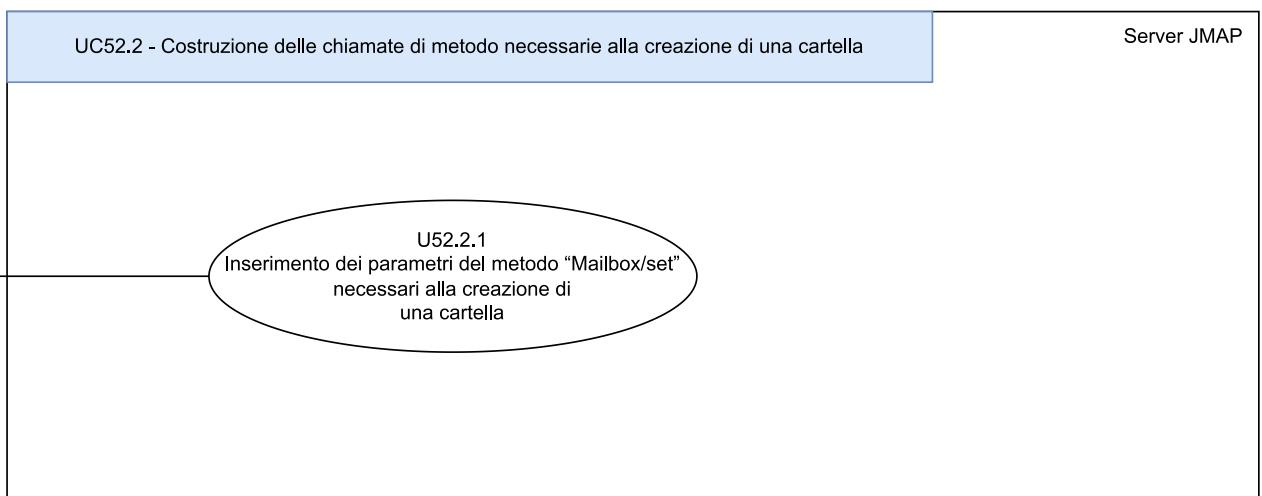


Figura 24: Sottocasi UC52.2 - Costruzione delle chiamate di metodo necessarie alla creazione di una cartella

#### 3.5.52.2.1) UC52.2.1 - Inserimento dei parametri del metodo “Mailbox/set” necessari alla creazione di una cartella

- Attore principale:** Client di posta elettronica autenticato;
- Descrizione:** Un client di posta elettronica, con lo scopo di creare una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per la creazione di una cartella, per il quale vanno specificati i parametri necessari;
- Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta, i parametri necessari;
- Scenario principale:**

- Il client specifica gli argomenti all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/set", la quale è contenuta nell'array di chiamate di metodo denominato "methodCalls", contenuto a sua volta all'interno dell'oggetto Request;

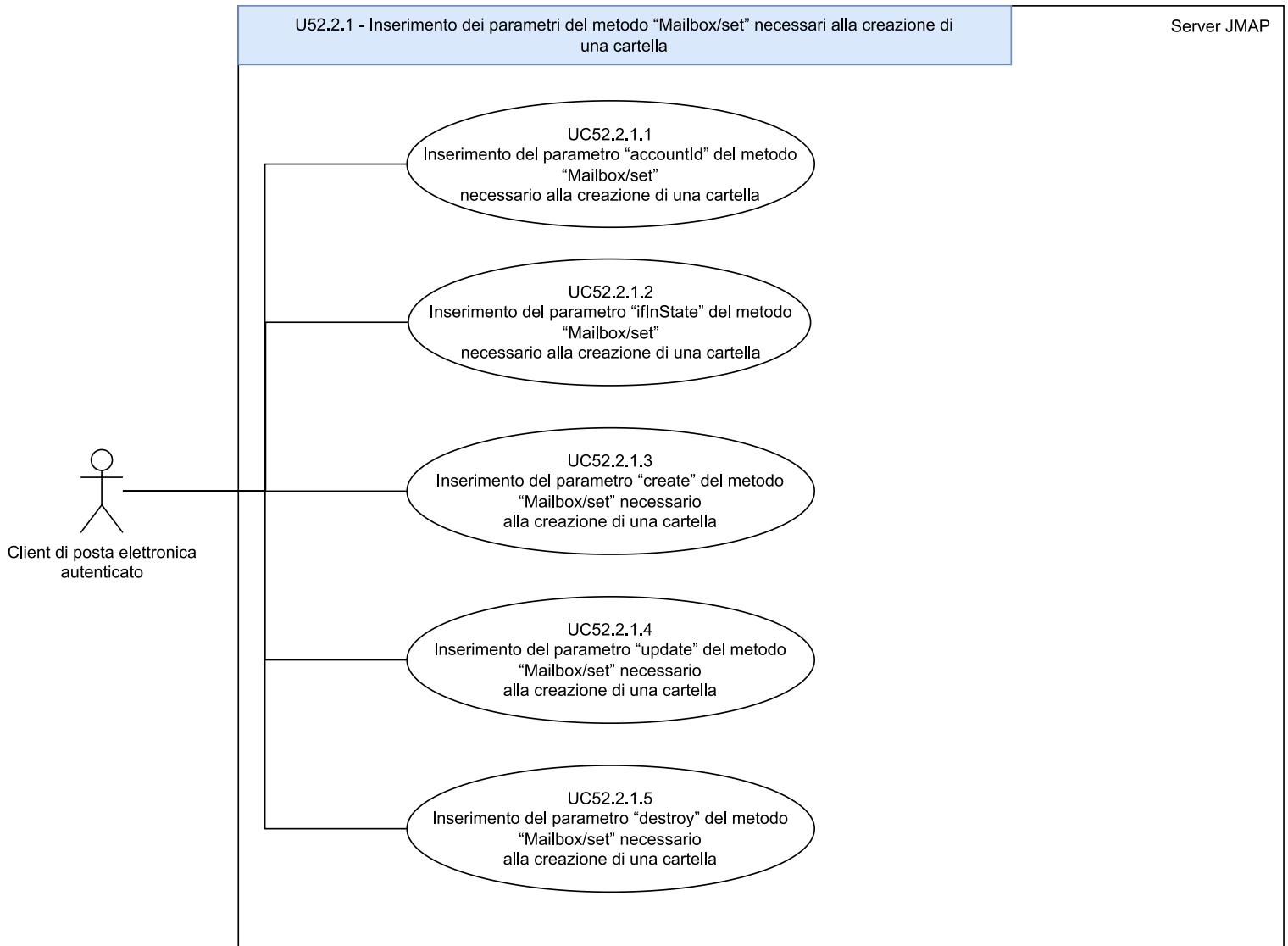


Figura 25: Sottocasi UC52.2.1 - Inserimento dei parametri del metodo "Mailbox/set" necessari alla creazione di una cartella

### 3.5.52.2.1.1) UC52.2.1.1 - Inserimento del parametro "accountId" del metodo "Mailbox/set" necessario alla creazione di una cartella

- Attore principale:** Client di posta elettronica autenticato;
- Descrizione:** Un client di posta elettronica, con lo scopo di creare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Mailbox/set", necessario per la creazione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è "accountId";
- Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/set", contenuto nell'array "methodCalls" della richiesta, il parametro "accountId";
- Scenario principale:**

- 
1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “accountID”, il quale rappresenta l'identificativo dell'account da utilizzare;

### 3.5.52.2.1.2) UC52.2.1.2 - Inserimento del parametro “ifInState” del metodo “Mailbox/set” necessario alla creazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di creare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per la creazione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “ifInState”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “ifInState”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “ifInState”, il quale è un meccanismo di controllo che impedisce la sovrascrittura di dati obsoleti o incoerenti sul server. Esso garantisce che la richiesta venga eseguita solo se il client è a conoscenza dello stato attuale degli oggetti e che tale stato sia valido al momento della richiesta;

### 3.5.52.2.1.3) UC52.2.1.3 - Inserimento del parametro “create” del metodo “Mailbox/set” necessario alla creazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di creare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per la creazione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “create”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “create”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “create”, il quale è utilizzato per specificare l'intenzione di creare una cartella. Questo parametro è una mappa, in cui le chiavi sono identificatori temporanei di creazione (creation id) assegnati dal client ed i valori sono oggetti Mailbox da creare. La definizione di un oggetto Mailbox può contenere valori predefiniti per le proprietà. Qualsiasi proprietà con un valore predefinito può essere omessa dal client. Inoltre Il client DEVE omettere qualsiasi proprietà che può essere impostata solo dal server;

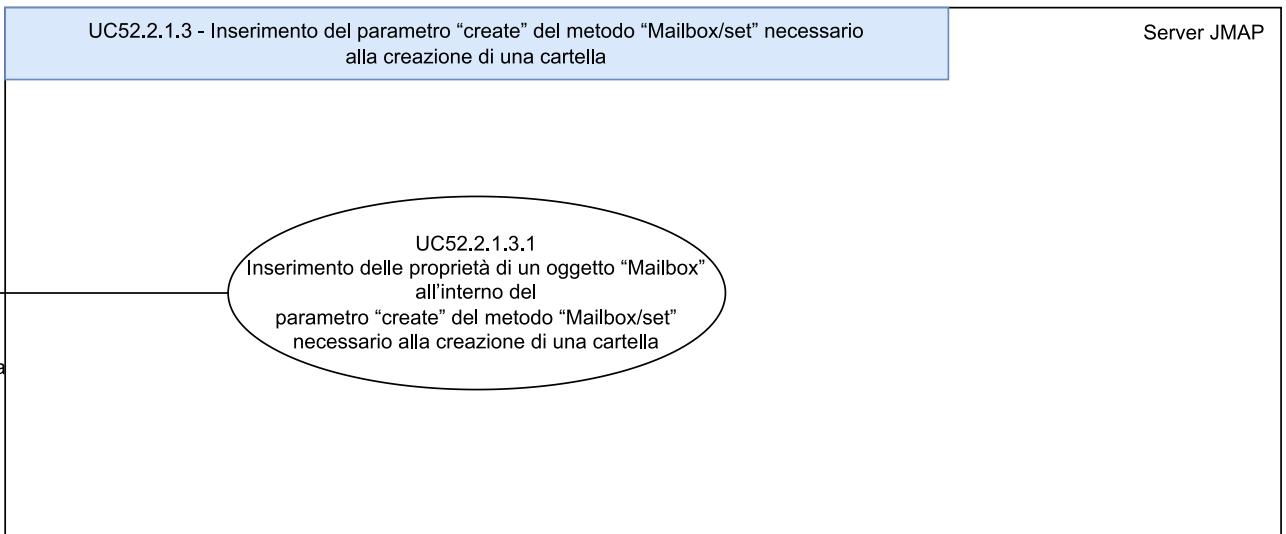


Figura 26: Sottocasi UC52.2.1.3 - Inserimento del parametro "create" del metodo "Mailbox/set" necessario alla creazione di una cartella

### 3.5.52.2.1.3.1) UC52.2.1.3.1 - Inserimento delle proprietà di un oggetto "Mailbox" all'interno del parametro "create" del metodo "Mailbox/set" necessario alla creazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di creare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Mailbox/set", necessario per la creazione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è "create". All'interno di questo parametro vanno definite le proprietà della cartella da creare;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito le proprietà di un oggetto "Mailbox" all'interno del parametro "create", contenuto a sua volta all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/set", contenuto nell'array "methodCalls" della richiesta;
- **Scenario principale:**
  1. Il client specifica all'interno del parametro "create", contenuto nell'oggetto di argomenti della chiamata di metodo, le proprietà dell'oggetto "Mailbox" da creare, tra le quali troviamo le seguenti:
    - name: Nome visibile dall'utente per la Mailbox, ad esempio, "Inbox". Deve essere una stringa Net-Unicode di almeno 1 carattere di lunghezza, soggetta alla dimensione massima specificata nell'oggetto di capacità. Non devono esistere due Mailbox fratelli con lo stesso genitore e lo stesso nome;
    - parentId: L'identificatore della Mailbox genitore di questa Mailbox, o null se questa Mailbox è al primo livello, ovvero non ha un genitore. Le Mailbox formano grafi aciclici (foreste) diretti dalla relazione figlio-genitore. Non deve esserci un loop. Di default viene posto a null;
    - role: Identifica le Mailbox che hanno uno scopo comune particolare (ad esempio, la "Inbox"), indipendentemente dalla proprietà name. Questo valore è condiviso con IMAP. Tuttavia, a differenza di IMAP, una Mailbox deve avere un solo ruolo e non devono esserci due Mailbox nello stesso account con lo stesso ruolo. Di default viene posto a null;

- sortOrder: Definisce l'ordine di visualizzazione delle Mailbox nell'interfaccia utente del client, in modo che sia consistente tra i dispositivi. Il numero deve essere un intero nell'intervallo  $0 \leq \text{sortOrder} < 2^{31}$ . Di default viene posto a null;
- isSubscribed: Indica se l'utente desidera vedere questa Mailbox nel proprio client. Questa proprietà dovrebbe essere impostata di default a false per le Mailbox negli account condivisi a cui l'utente ha accesso, e a true per qualsiasi nuova Mailbox creata dall'utente stesso;

#### 3.5.52.2.1.4) UC52.2.1.4 - Inserimento del parametro “update” del metodo “Mailbox/set” necessario alla creazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di creare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per la creazione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “update”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “update”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “update”, il quale viene impostato a null, dato che nella creazione di una cartella nessun oggetto deve essere aggiornato;

#### 3.5.52.2.1.5) UC52.2.1.5 - Inserimento del parametro “destroy” del metodo “Mailbox/set” necessario alla creazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di creare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per la creazione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “destroy”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “destroy”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “destroy”, il quale viene impostato a null, dato che nella creazione di una cartella nessun oggetto deve essere distrutto;

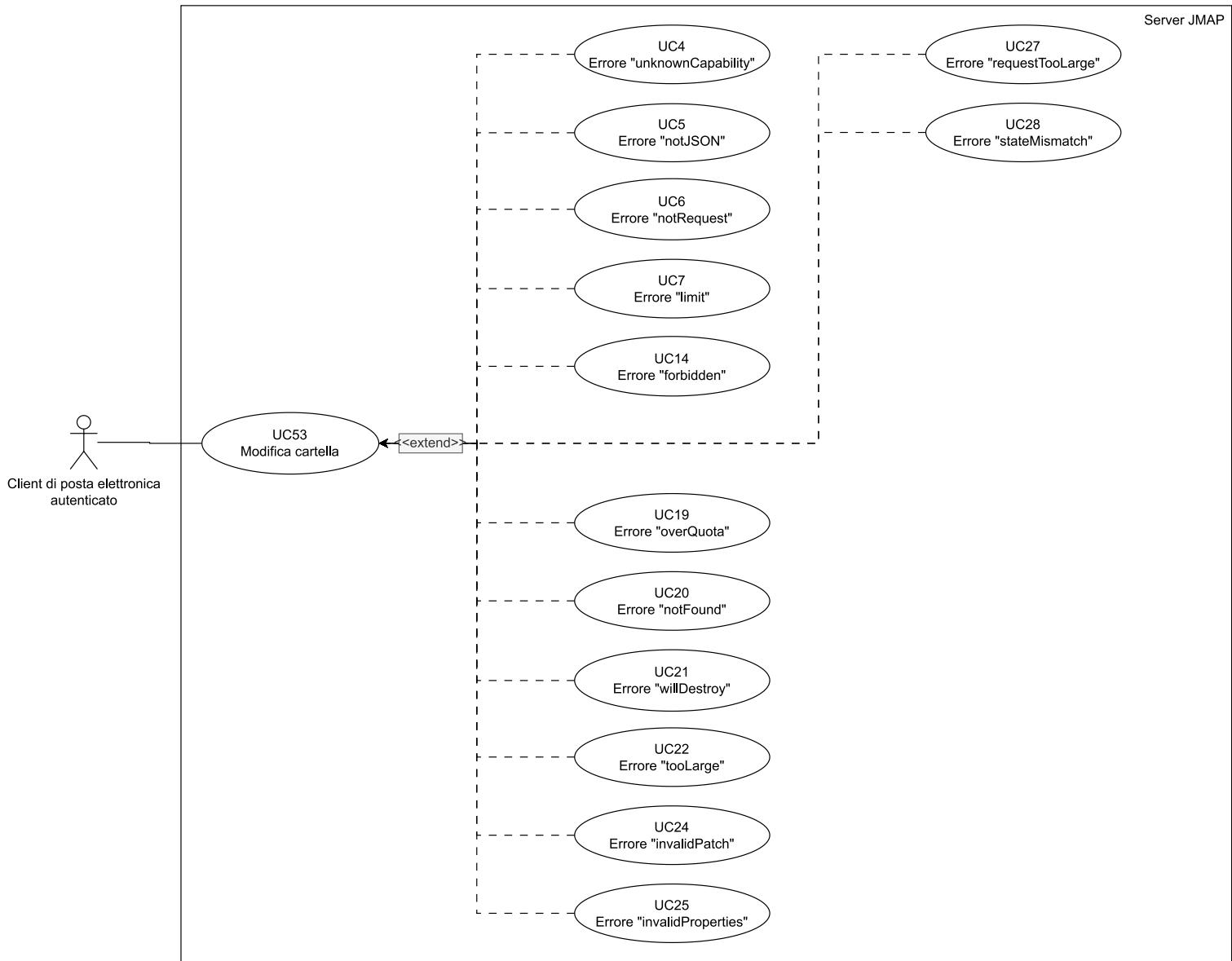


Figura 27: UC53 - Modifica cartella

### 3.5.53) UC53 - Modifica cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare una cartella, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** La cartella è stata modificata con successo ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per modificare una cartella;
  2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
  3. Il server elabora la richiesta API e invia una risposta JSON che contiene l'esito dell'operazione di modifica della cartella;

- **Estensioni:**
  - Errore “unknownCapability”;
  - Errore “notJSON”;
  - Errore “notRequest”;
  - Errore “limit”;
  - Errore “forbidden”;
  - Errore “overQuota”;
  - Errore “tooLarge”;
  - Errore “notFound”;
  - Errore “invalidPatch”;
  - Errore “willDestroy”;
  - Errore “invalidProperties”;
  - Errore “requestTooLarge”;
  - Errore “stateMismatch”;
- **Inclusioni:** /
- **Generalizzazioni:** /

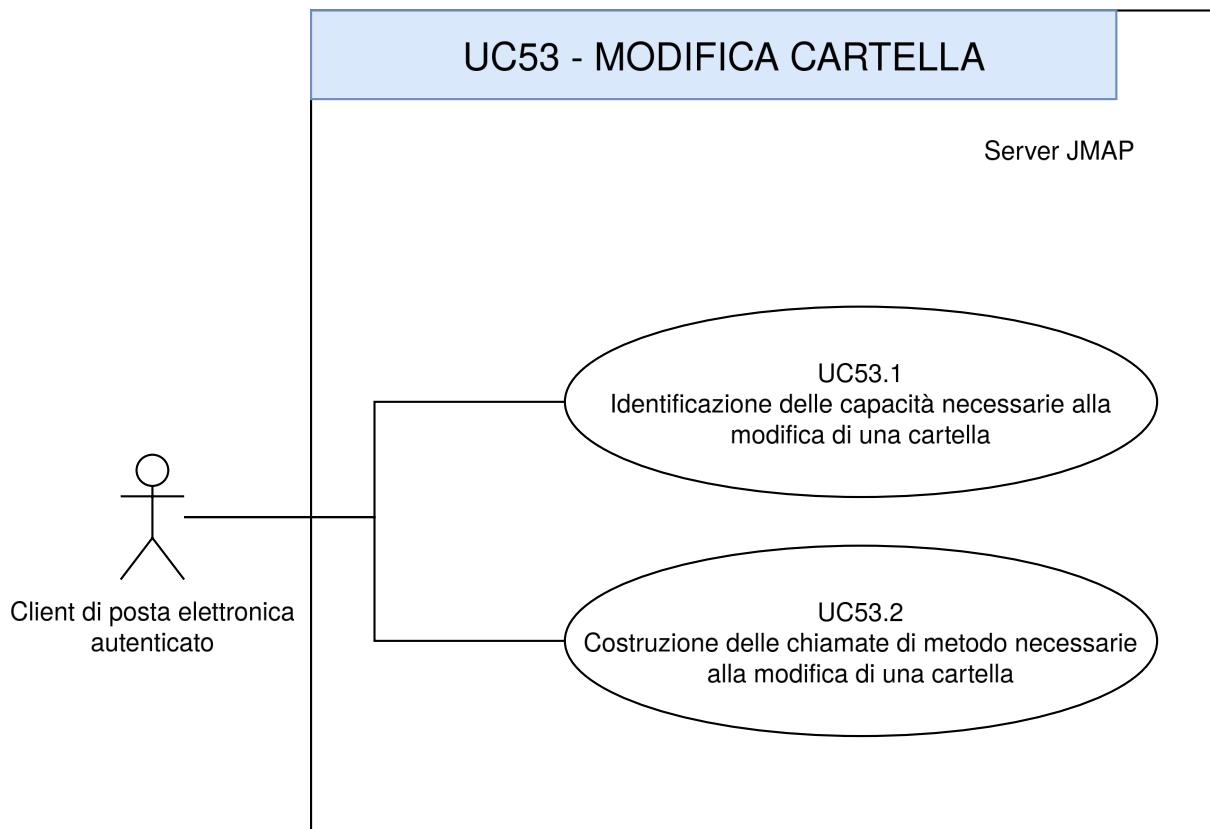


Figura 28: Sottocasi UC53 - Modifica cartella

### 3.5.53.1) UC53.1 - Identificazione delle capacità necessarie alla modifica di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per la modifica di una cartella;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;

- **Postcondizioni:** Il client ha identificato all'interno della richiesta le capacità JMAP necessarie per la modifica di una cartella;
- **Scenario principale:**
  1. Il client include nella sezione “using” dell’oggetto Request le capacità JMAP “urn:ietf:params:jmap:core” e “urn:ietf:params:jmap:mail”;

### 3.5.53.2) UC53.2 - Costruzione delle chiamate di metodo necessarie alla modifica di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per la modifica di una cartella;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’array “methodCalls” della richiesta le chiamate di metodo necessarie per la modifica di una cartella;
- **Scenario principale:**
  1. Il client crea un array di chiamate di metodo denominato “methodCalls” internamente all’oggetto Request, all’interno del quale inserisce il metodo “Mailbox/set”, un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

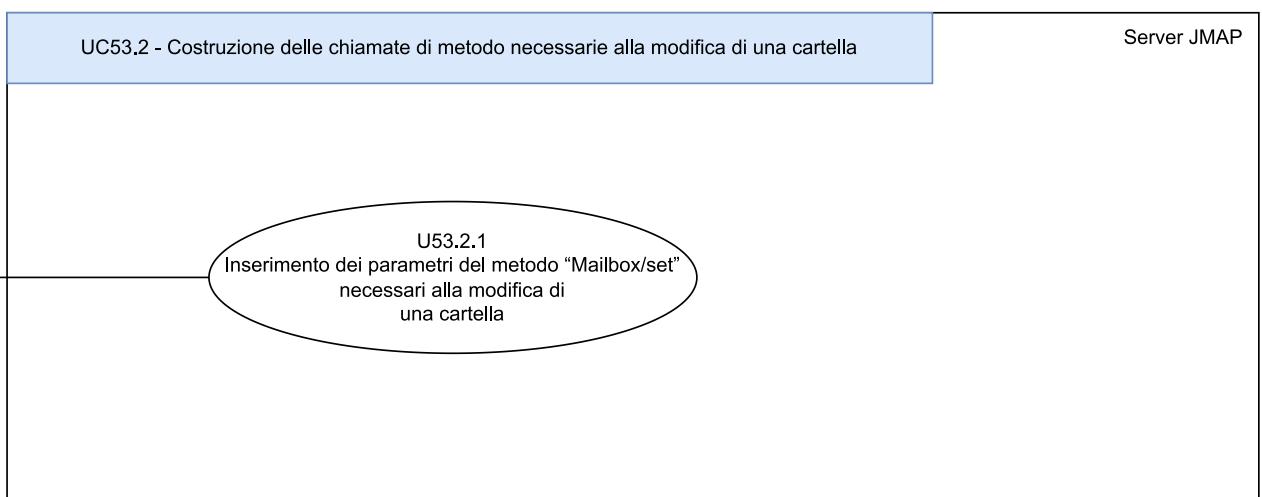


Figura 29: Sottocaso UC53.2 - Costruzione delle chiamate di metodo necessarie alla modifica di una cartella

#### 3.5.53.2.1) UC53.2.1 - Inserimento dei parametri del metodo “Mailbox/set” necessari alla modifica di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per la modifica di una cartella, per il quale vanno specificati i parametri necessari;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;

- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, la quale è contenuta nell'array di chiamate di metodo denominato “methodCalls”, contenuto a sua volta all'interno dell'oggetto Request;

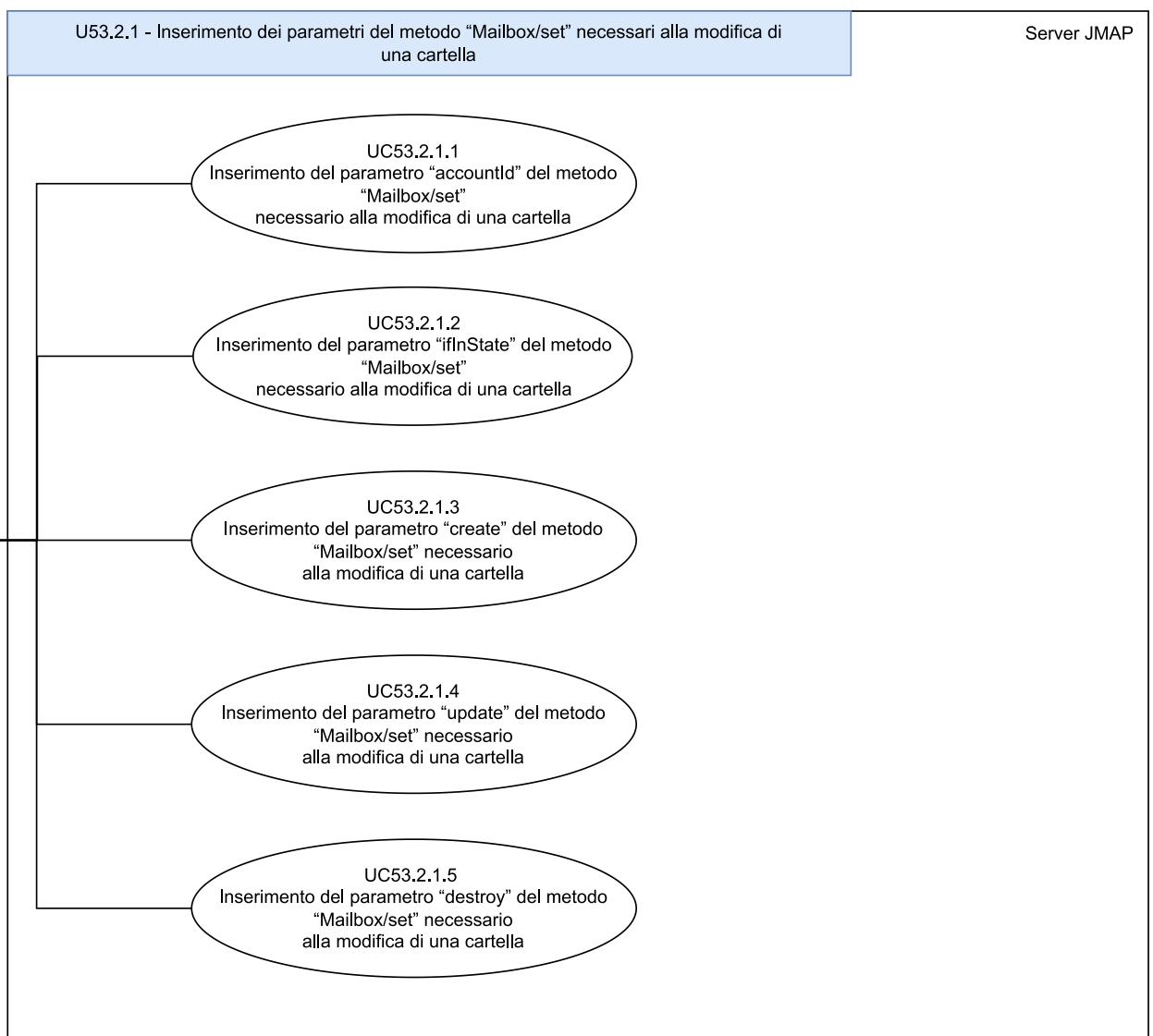


Figura 30: Sottocasi UC53.2.1 - Inserimento dei parametri del metodo “Mailbox/set” necessari alla modifica di una cartella

### 3.5.53.2.1.1) UC53.2.1.1 - Inserimento del parametro “accountId” del metodo “Mailbox/set” necessario alla modifica di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per la modifica di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “accountId”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;

- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “accountId”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “accountId”, il quale rappresenta l'identificativo dell'account da utilizzare;

### 3.5.53.2.1.2) UC53.2.1.2 - Inserimento del parametro “ifInState” del metodo “Mailbox/set” necessario alla modifica di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per la modifica di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “ifInState”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “ifInState”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “ifInState”, il quale è un meccanismo di controllo che impedisce la sovrascrittura di dati obsoleti o incoerenti sul server. Esso garantisce che la richiesta venga eseguita solo se il client è a conoscenza dello stato attuale degli oggetti e che tale stato sia valido al momento della richiesta;

### 3.5.53.2.1.3) UC53.2.1.3 - Inserimento del parametro “create” del metodo “Mailbox/set” necessario alla modifica di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per la modifica di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “create”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “create”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “create”, il quale viene impostato a null, dato che nella modifica di una cartella nessun oggetto deve essere creato;

### 3.5.53.2.1.4) UC53.2.1.4 - Inserimento del parametro “update” del metodo “Mailbox/set” necessario alla modifica di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per la modifica di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “update”;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/set", contenuto nell'array "methodCalls" della richiesta, il parametro "update";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "update", il quale è utilizzato per specificare l'intenzione di modificare una cartella. Questo parametro è una mappa che associa un identificativo a un oggetto di tipo patch (PatchObject) da applicare all'oggetto Mailbox corrente con quell'identificativo. Un PatchObject rappresenta un insieme non ordinato di modifiche;

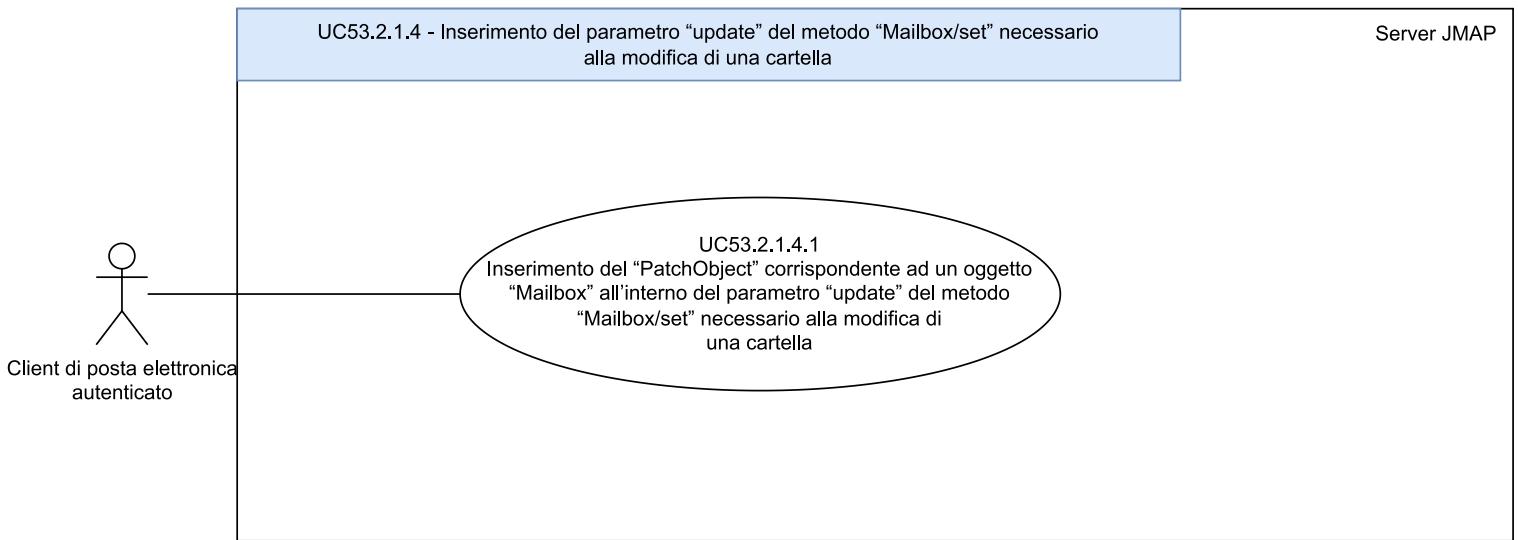


Figura 31: Sottocaso UC53.2.1.4 - Inserimento del parametro "update" del metodo "Mailbox/set" necessario alla modifica di una cartella

### 3.5.53.2.1.4.1) UC53.2.1.4.1 - Inserimento del "PatchObject" corrispondente ad un oggetto "Mailbox" all'interno del parametro "update" del metodo "Mailbox/set" necessario alla modifica di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Mailbox/set", necessario per la modifica di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è "update". All'interno di questo parametro va definito un "PatchObject", il quale rappresenta un insieme non ordinato di modifiche;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito il "PatchObject" di un oggetto "Mailbox" all'interno del parametro "update", contenuto a sua volta all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/set", contenuto nell'array "methodCalls" della richiesta;
- **Scenario principale:**
  1. Il client specifica all'interno del parametro "update", contenuto nell'oggetto di argomenti della chiamata di metodo, l'oggetto "PatchObject" necessario per la modifica di una cartella. Questo è un array di stringhe, il quale rappresenta un insieme non ordinato di patch, dove le chiavi sono un percorso nel formato JSON Pointer, con un "/" posto all'inizio implicitamente, che stanno ad indicare una particolare proprietà.

---

Tutti i percorsi utilizzati come chiave devono essere conformi a specifiche restrizioni, altrimenti la modifica verrà rifiutata rispondendo con un errore “invalidPatch”.

Per quanto riguarda il valore associato a ciascun percorso:

- se è nullo allora imposta il valore predefinito, se specificato, per la proprietà corrispondente; se non specificato, rimuove la proprietà dalla cartella modificata. Se la chiave non è presente nel padre, questa è un’operazione senza effetto;
- se non è nullo allora specifica il valore da impostare per la proprietà corrispondente (questa può essere una sostituzione o un’aggiunta alla cartella che viene modificata).

Eventuali proprietà impostate dal server possono essere incluse nella modifica se il loro valore è identico al valore corrente del server (prima di applicare le modifiche alla cartella). In caso contrario, la modifica DEVE essere respinta con un errore di tipo “invalidProperties”.

Questa definizione consente di apportare modifiche sia inviando l’intero nuovo oggetto “Mailbox” come patch, che inviando solo le differenze. Entrambi i metodi sono validi, e il server elabora le modifiche allo stesso modo in entrambi i casi;

### 3.5.53.2.1.5) UC53.2.1.5 - Inserimento del parametro “destroy” del metodo “Mailbox/set” necessario alla modifica di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per la modifica di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “destroy”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “destroy”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “destroy”, il quale viene impostato a null, dato che nella modifica di una cartella nessun oggetto deve essere distrutto;

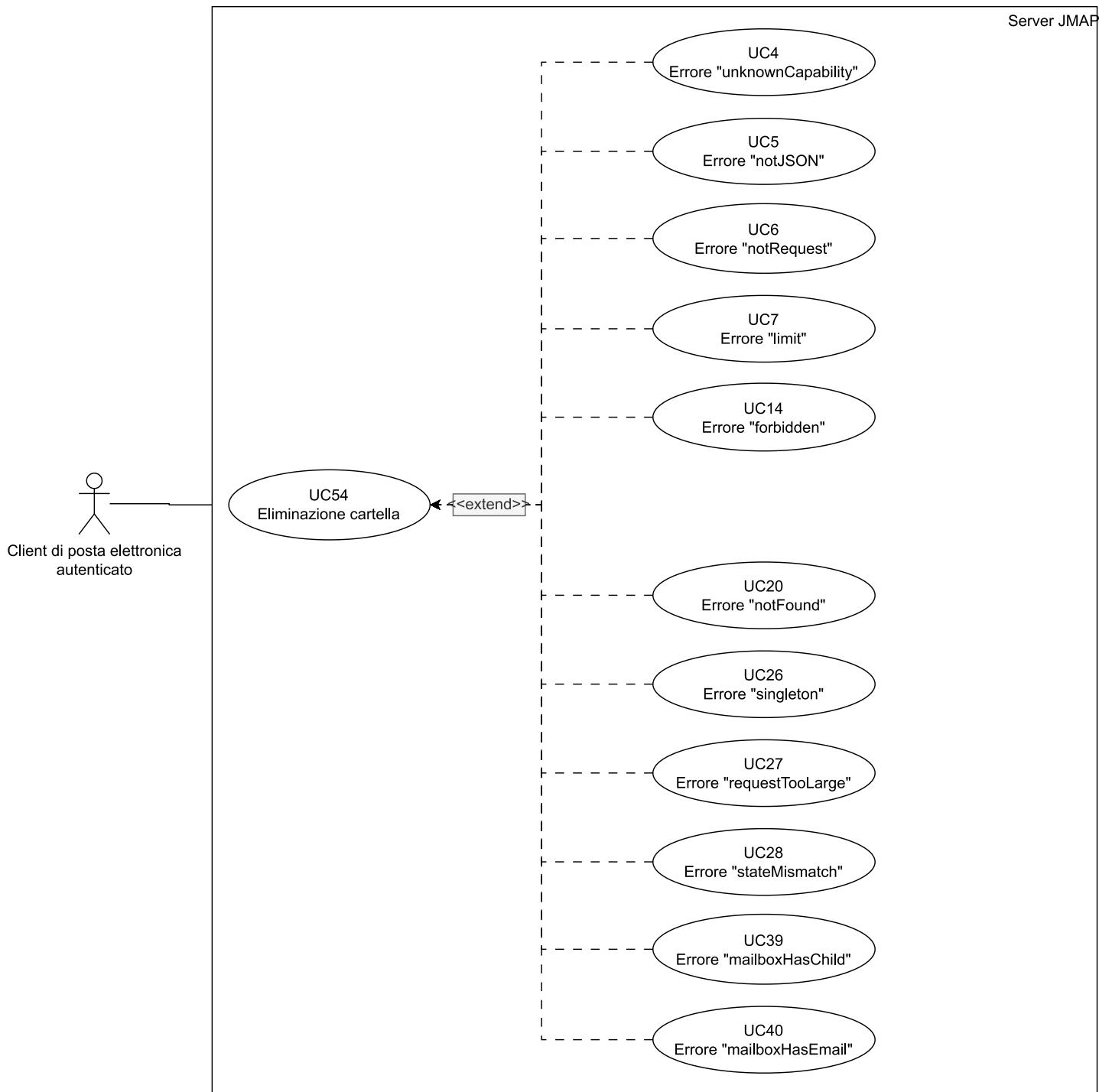


Figura 32: UC54 - Eliminazione cartella

### 3.5.54) UC54 - Eliminazione cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una cartella, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** La cartella è stata eliminata con successo ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**

1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per eliminare una cartella;
2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
3. Il server elabora la richiesta API e invia una risposta JSON che contiene l'esito dell'operazione di eliminazione della cartella;

- **Estensioni:**

- Errore "unknownCapability";
- Errore "notJSON";
- Errore "notRequest";
- Errore "limit";
- Errore "forbidden";
- Errore "notFound";
- Errore "singleton";
- Errore "requestTooLarge";
- Errore "stateMismatch";
- Errore "mailboxHasChild";
- Errore "mailboxHasEmail";

- **Inclusioni:** /

- **Generalizzazioni:** /

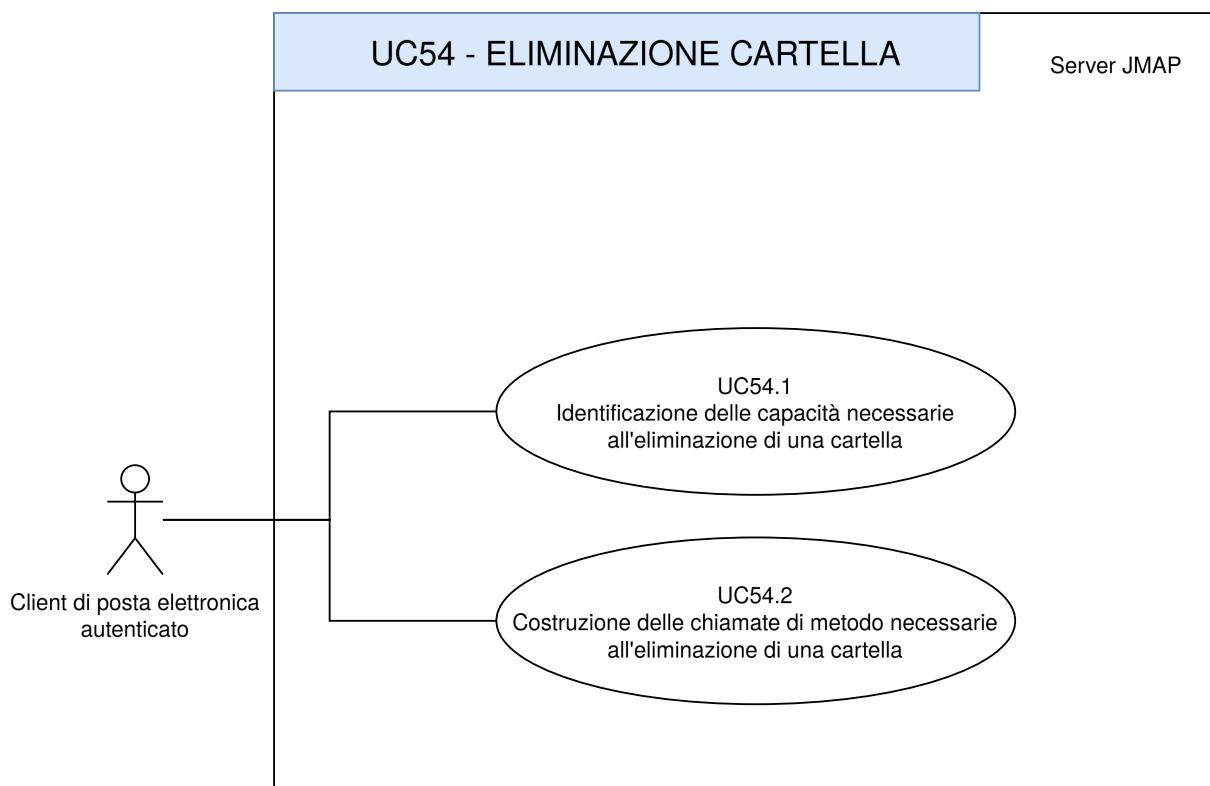


Figura 33: Sottocasi UC54 - Eliminazione cartella

#### 3.5.54.1 UC54.1 - Identificazione delle capacità necessarie all'eliminazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session.

Internamente a questa richiesta il client identifica le capacità JMAP necessarie per l'eliminazione di una cartella;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all'interno della richiesta le capacità JMAP necessarie per l'eliminazione di una cartella;
- **Scenario principale:**
  1. Il client include nella sezione “using” dell'oggetto Request le capacità JMAP “urn:ietf:params:jmap:core” e “urn:ietf:params:jmap:mail”;

### 3.5.54.2) UC54.2 - Costruzione delle chiamate di metodo necessarie all'eliminazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per l'eliminazione di una cartella;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'array “methodCalls” della richiesta le chiamate di metodo necessarie per l'eliminazione di una cartella;
- **Scenario principale:**
  1. Il client crea un array di chiamate di metodo denominato “methodCalls” internamente all'oggetto Request, all'interno del quale inserisce il metodo “Mailbox/set”, un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

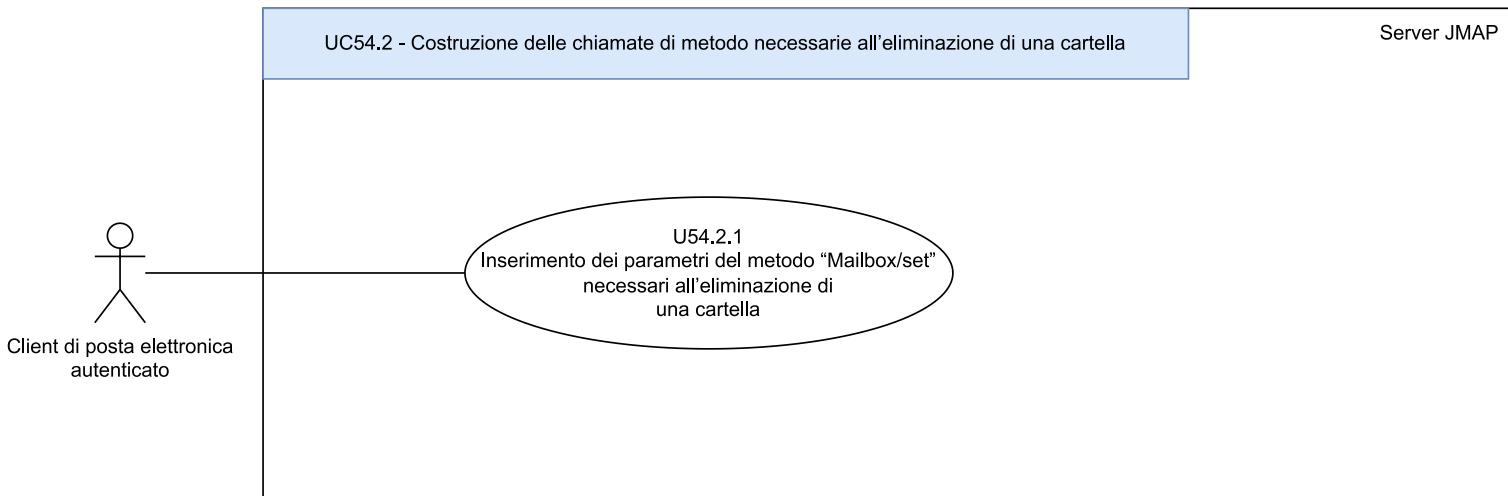


Figura 34: Sottocasi UC54.2 - Costruzione delle chiamate di metodo necessarie all’eliminazione di una cartella

#### 3.5.54.2.1) UC54.2.1 - Inserimento dei parametri del metodo “Mailbox/set” necessari all’eliminazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il

solo metodo “Mailbox/set”, necessario per l’eliminazione di una cartella, per il quale vanno specificati i parametri necessari;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, la quale è contenuta nell’array di chiamate di metodo denominato “methodCalls”, contenuto a sua volta all’interno dell’oggetto Request;

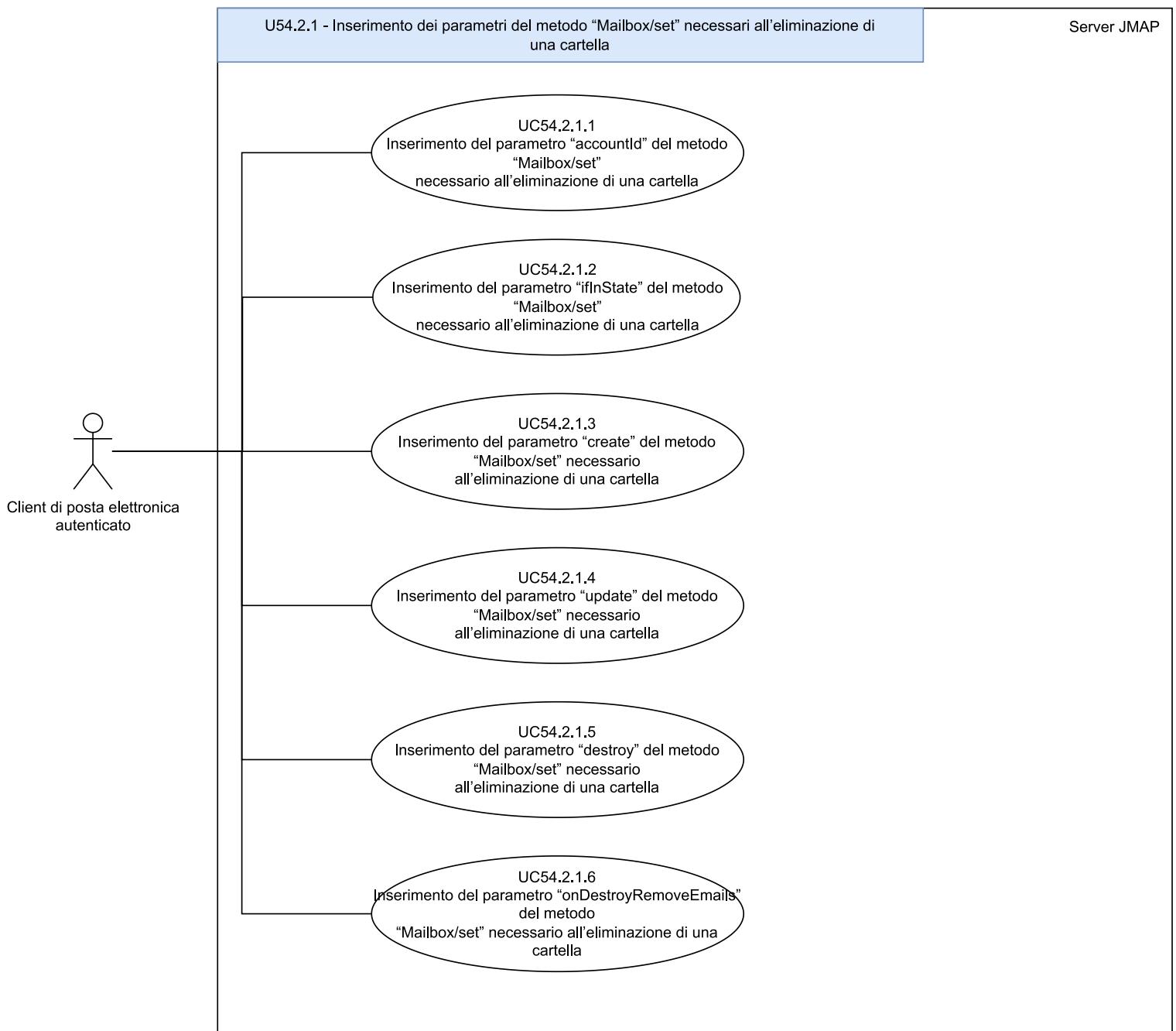


Figura 35: Sottocasi UC54.2.1 - Inserimento dei parametri del metodo “Mailbox/set” necessari all’eliminazione di una cartella

### 3.5.54.2.1.1) UC54.2.1.1 - Inserimento del parametro “accountId” del metodo “Mailbox/set” necessario all’eliminazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per l’eliminazione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “accountId”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “accountId”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “accountId”, il quale rappresenta l’identificativo dell’account da utilizzare;

### 3.5.54.2.1.2) UC54.2.1.2 - Inserimento del parametro “ifInState” del metodo “Mailbox/set” necessario all’eliminazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per l’eliminazione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “ifInState”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “ifInState”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “ifInState”, il quale è un meccanismo di controllo che impedisce la sovrascrittura di dati obsoleti o incoerenti sul server. Esso garantisce che la richiesta venga eseguita solo se il client è a conoscenza dello stato attuale degli oggetti e che tale stato sia valido al momento della richiesta;

### 3.5.54.2.1.3) UC54.2.1.3 - Inserimento del parametro “create” del metodo “Mailbox/set” necessario all’eliminazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per l’eliminazione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “create”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “create”;
- **Scenario principale:**

- 
1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “create”, il quale viene impostato a null, dato che nell'eliminazione di una cartella nessun oggetto deve essere creato;

#### 3.5.54.2.1.4) UC54.2.1.4 - Inserimento del parametro “update” del metodo “Mailbox/set” necessario all'eliminazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per l'eliminazione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “update”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “update”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “update”, il quale viene impostato a null, dato che nell'eliminazione di una cartella nessun oggetto deve essere aggiornato;

#### 3.5.54.2.1.5) UC54.2.1.5 - Inserimento del parametro “destroy” del metodo “Mailbox/set” necessario all'eliminazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per l'eliminazione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “destroy”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “destroy”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “destroy”, il quale è utilizzato per specificare l'intenzione di eliminare una cartella. Questo parametro è un array che contiene solamente l'identificativo della cartella da eliminare;

#### 3.5.54.2.1.6) UC54.2.1.6 - Inserimento del parametro “onDestroyRemoveEmails” del metodo “Mailbox/set” necessario all'eliminazione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Mailbox/set”, necessario per l'eliminazione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “onDestroyRemoveEmails”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;

- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “onDestroyRemoveEmails”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “onDestroyRemoveEmails”, il quale determina il comportamento desiderato quando si cerca di eliminare una cartella che contiene ancora delle email. La sua opzione predefinita è false. Se impostato su false qualsiasi tentativo di eliminare una cartella che contiene ancora delle email sarà respinto e verrà restituito un errore di tipo “mailboxHasEmail”, indicando che la cartella non può essere eliminata perché contiene ancora delle email. Se impostato su true allora quando si elimina una cartella, le email che erano contenute in essa saranno rimosse dalla cartella. Se queste email non sono presenti in nessun'altra cartella, verranno distrutte insieme alla cartella stessa;

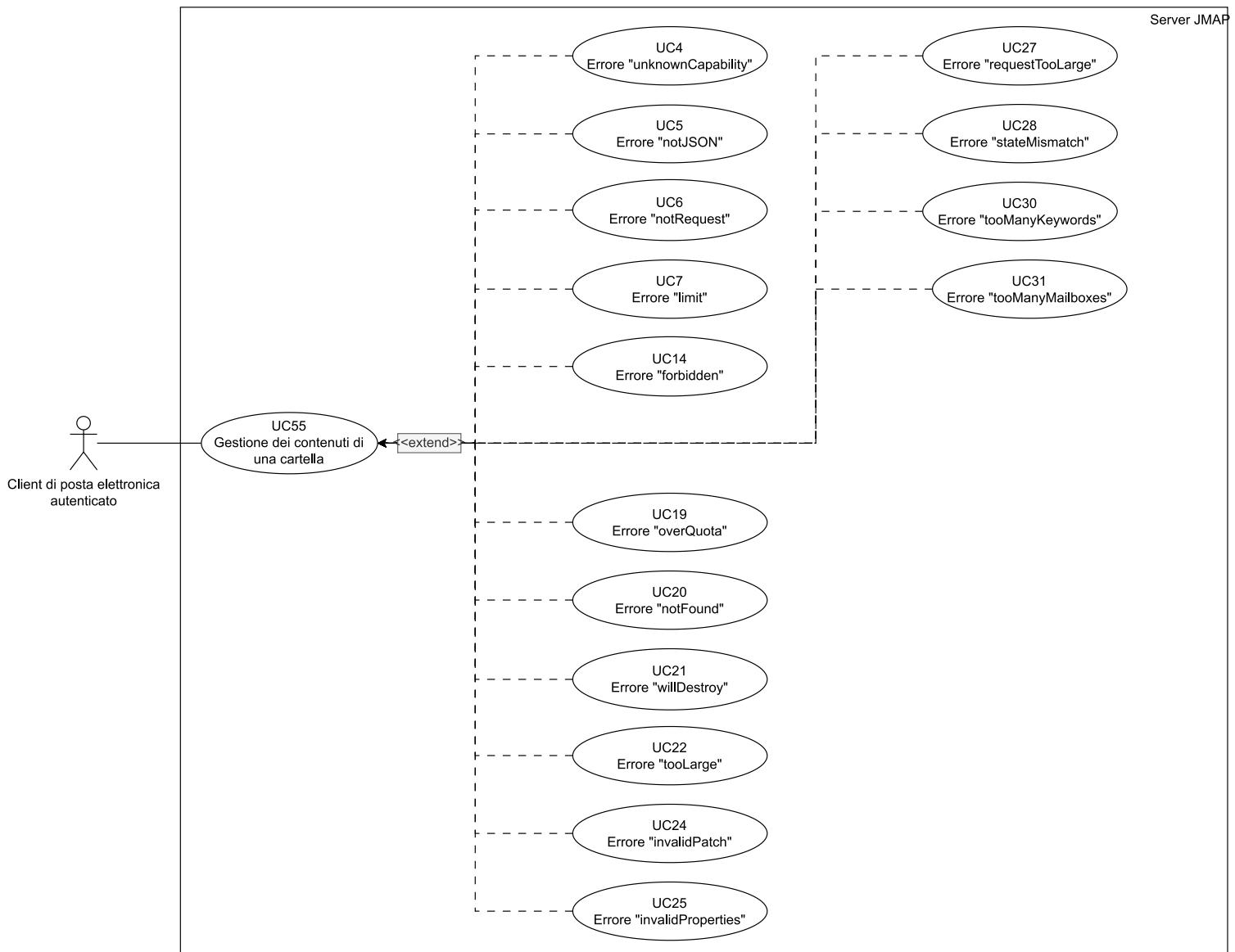


Figura 36: UC55 - Gestione contenuti cartella

### 3.5.55) UC55 - Gestione contenuti cartella

- **Attore principale:** Client di posta elettronica autenticato;

- **Descrizione:** Un client di posta elettronica, con lo scopo di gestire i contenuti di una cartella, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Le operazioni sui contenuti di una cartella sono avvenute con successo ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per gestire i contenuti una cartella;
  2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
  3. Il server elabora la richiesta API e invia una risposta JSON che contiene l'esito delle operazioni;
- **Estensioni:**
  - Errore “unknownCapability”;
  - Errore “notJSON”;
  - Errore “notRequest”;
  - Errore “limit”;
  - Errore “forbidden”;
  - Errore “overQuota”;
  - Errore “tooLarge”;
  - Errore “notFound”;
  - Errore “invalidPatch”;
  - Errore “willDestroy”;
  - Errore “invalidProperties”;
  - Errore “requestTooLarge”;
  - Errore “stateMismatch”;
  - Errore “tooManyKeywords”;
  - Errore “tooManyMailboxes”;
- **Inclusioni:** /
- **Generalizzazioni:** /

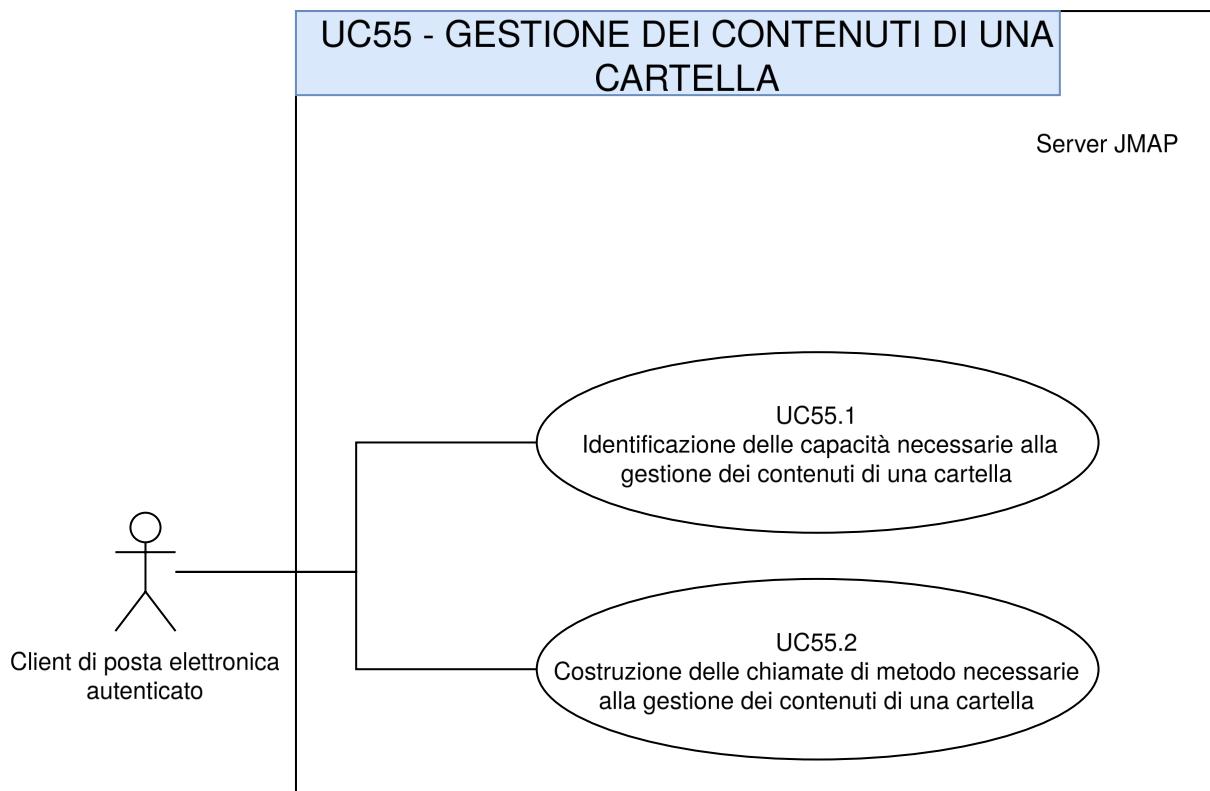


Figura 37: Sottocasi UC55 - Gestione contenuti cartella

### 3.5.55.1) UC55.1 - Identificazione delle capacità necessarie alla gestione dei contenuti di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di gestire i contenuti di una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per la gestione dei contenuti di una cartella;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all'interno della richiesta le capacità JMAP necessarie per la gestione dei contenuti di una cartella;
- **Scenario principale:**
  1. Il client include nella sezione “using” dell’oggetto Request le capacità JMAP “urn:ietf:params:jmap:core” e “urn:ietf:params:jmap:mail”;

### 3.5.55.2) UC55.2 - Costruzione delle chiamate di metodo necessarie alla gestione dei contenuti di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di gestire i contenuti di una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per la gestione dei contenuti di una cartella;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'array “methodCalls” della richiesta le chiamate di metodo necessarie per la gestione dei contenuti di una cartella;

- **Scenario principale:**

1. Il client crea un array di chiamate di metodo denominato “methodCalls” internamente all’oggetto Request, all’interno del quale inserisce il metodo “Email/set”, un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

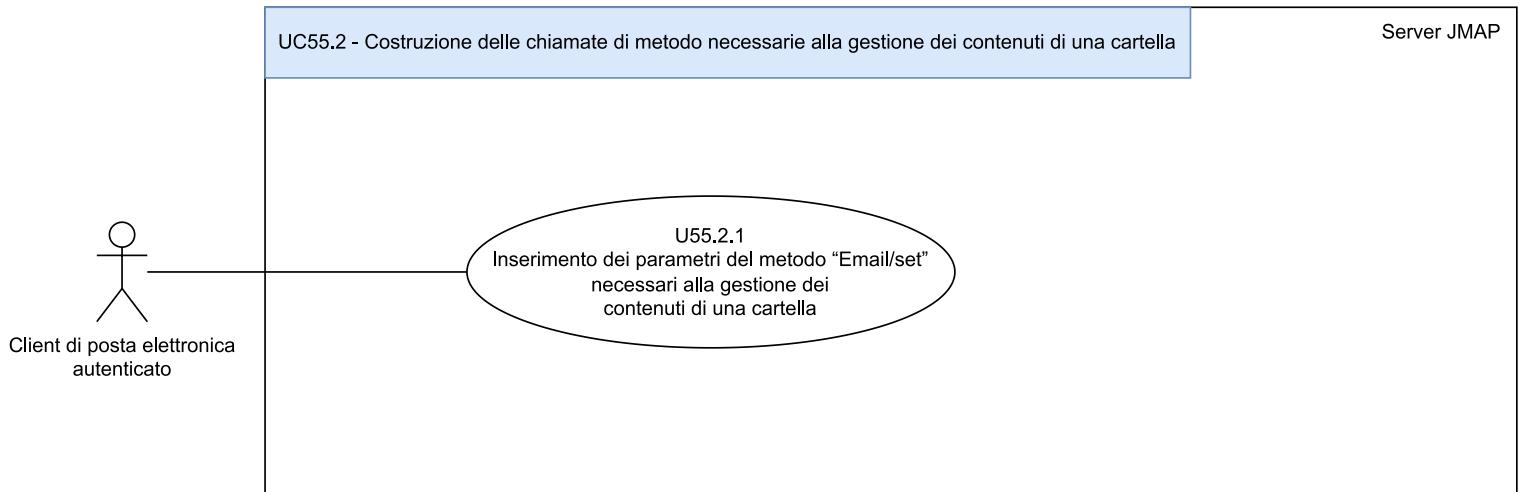


Figura 38: Sottocasi UC55.2 - Costruzione delle chiamate di metodo necessarie alla gestione dei contenuti di una cartella

#### 3.5.55.2.1) UC55.2.1 - Inserimento dei parametri del metodo “Email/set” necessari alla gestione dei contenuti di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di gestire i contenuti di una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per la gestione i contenuti una cartella, per il quale vanno specificati i parametri necessari;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, la quale è contenuta nell’array di chiamate di metodo denominato “methodCalls”, contenuto a sua volta all’interno dell’oggetto Request;

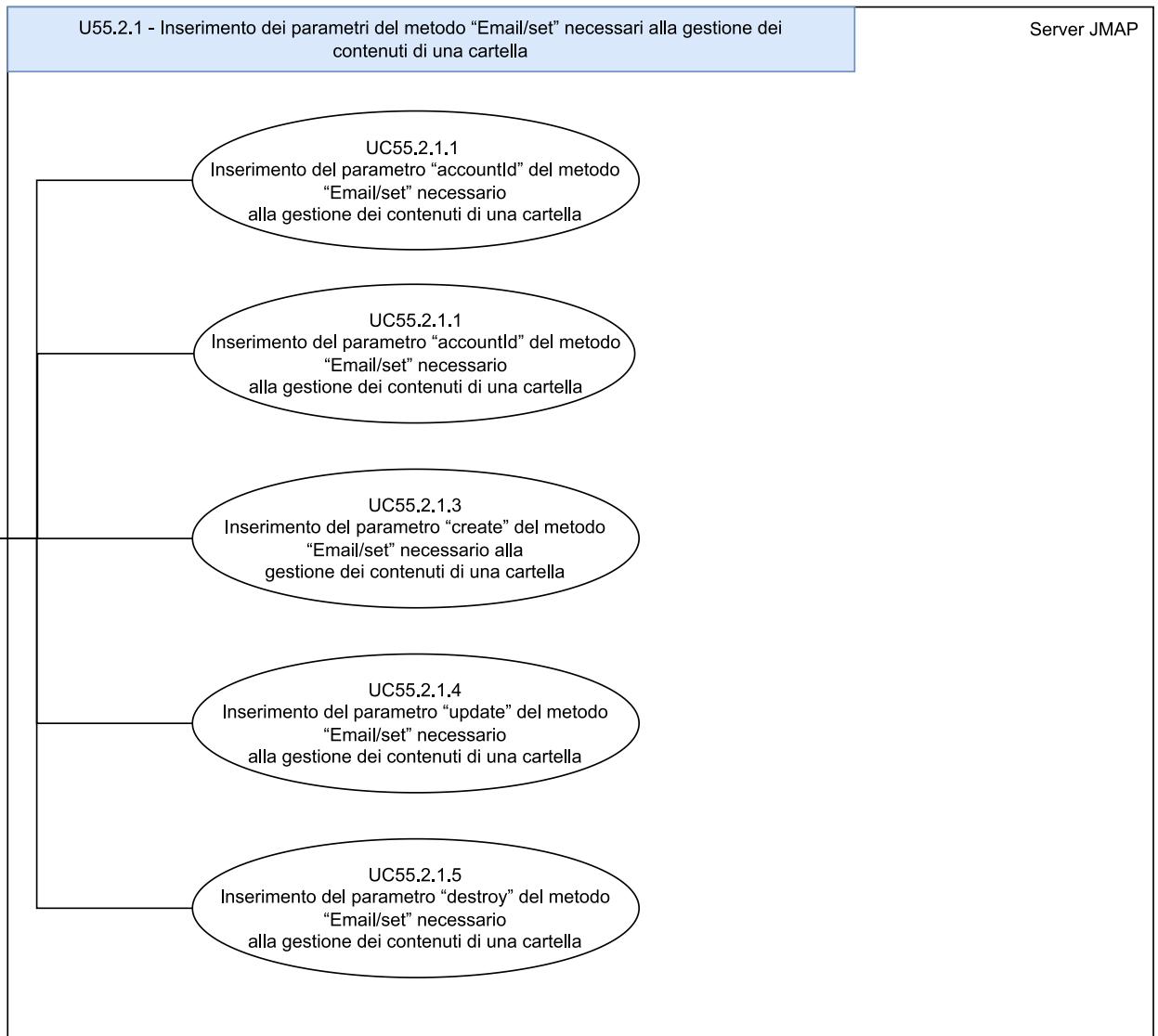


Figura 39: Sottocasi UC55.2.1 - Inserimento dei parametri del metodo "Email/set" necessari alla gestione dei contenuti di una cartella

### 3.5.55.2.1.1) UC55.2.1.1 - Inserimento del parametro "accountId" del metodo "Email/set" necessario alla gestione dei contenuti di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di gestire i contenuti di una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Email/set", necessario per la gestione dei contenuti di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è "accountId";
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Email/set", contenuto nell'array "methodCalls" della richiesta, il parametro "accountId";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "accountId", il quale rappresenta l'identificativo dell'account da utilizzare;

### 3.5.55.2.1.2) UC55.2.1.2 - Inserimento del parametro “ifInState” del metodo “Email/set” necessario alla gestione dei contenuti di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di gestire i contenuti di una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per la gestione dei contenuti di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “ifInState”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “ifInState”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “ifInState”, il quale è un meccanismo di controllo che impedisce la sovrascrittura di dati obsoleti o incoerenti sul server. Esso garantisce che la richiesta venga eseguita solo se il client è a conoscenza dello stato attuale degli oggetti e che tale stato sia valido al momento della richiesta;

### 3.5.55.2.1.3) UC55.2.1.3 - Inserimento del parametro “create” del metodo “Email/set” necessario alla gestione dei contenuti di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di gestire i contenuti di una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per la gestione dei contenuti di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “create”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “create”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “create”, il quale viene impostato a null, dato che nella gestione dei contenuti di una cartella nessun oggetto deve essere creato;

### 3.5.55.2.1.4) UC55.2.1.4 - Inserimento del parametro “update” del metodo “Email/set” necessario alla gestione dei contenuti di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di gestire i contenuti di una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per la gestione dei contenuti di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “update”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “update”;
- **Scenario principale:**

- Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “update”, il quale è utilizzato per specificare l'intenzione di modificare una o più email, per far sì che queste possano essere aggiunte ad una cartella, rimosse da una cartella o spostate da una cartella ad un'altra. Questo parametro è una mappa che associa un identificativo a un oggetto di tipo patch (PatchObject) da applicare all'oggetto Email con quell'identificativo. Un PatchObject rappresenta un insieme non ordinato di modifiche;

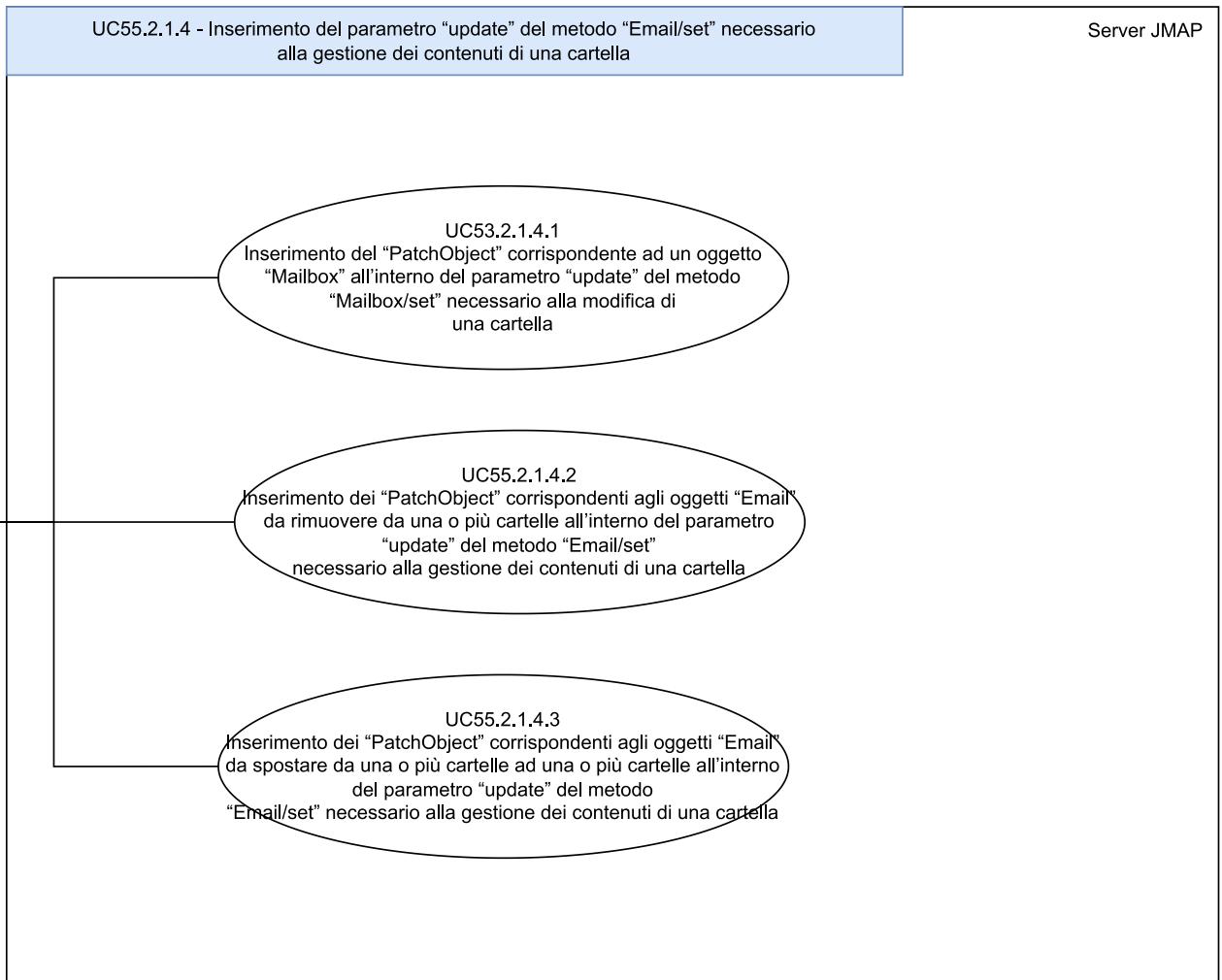


Figura 40: Sottocasi UC55.2.1.4 - Inserimento del parametro “update” del metodo “Email/set” necessario alla gestione dei contenuti di una cartella

### 3.5.55.2.1.4.1) UC55.2.1.4.1 - Inserimento dei “PatchObject” corrispondenti agli oggetti “Email” da aggiungere ad una o più cartelle all'interno del parametro “update” del metodo “Email/set” necessario alla gestione dei contenuti di una cartella

- Attore principale:** Client di posta elettronica autenticato;
- Descrizione:** Un client di posta elettronica, con lo scopo di aggiungere una o più email ad una o più cartelle e, quindi, di gestire i contenuti di quest'ultima, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per la gestione dei contenuti di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “update”. All'interno di questo parametro va definito un “PatchObject” per ogni email che il client vuole aggiungere ad una o più cartelle;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito il “PatchObject” di ogni oggetto “Email” che intende aggiungere ad una o più cartelle all'interno del parametro “update”, contenuto a sua volta all'interno dell'oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell'array “methodCalls” della richiesta;
- **Scenario principale:**

1. Il client specifica all'interno del parametro “update”, contenuto nell'oggetto di argomenti della chiamata di metodo, gli oggetti “PatchObject” necessari per l'aggiunta delle email ad una o più cartelle. Questo parametro è un array di stringhe, il quale rappresenta un insieme non ordinato di patch, dove le chiavi sono un percorso nel formato JSON Pointer, con un “/” posto all'inizio implicitamente, che stanno ad indicare una particolare proprietà. In questo caso ci interessa solamente la proprietà “mailboxIds” che verrà modificata per contenere gli identificativi di tutte le cartelle a cui il client intende aggiungere le email.

Tutti i percorsi utilizzati come chiave devono essere conformi a specifiche restrizioni, altrimenti la modifica verrà rifiutata rispondendo con un errore “invalidPatch”.

Per quanto riguarda il valore associato a ciascun percorso:

- se è nullo allora imposta il valore predefinito, se specificato, per la proprietà corrispondente; se non specificato, rimuove la proprietà dalla cartella modificata. Se la chiave non è presente nel padre, questa è un'operazione senza effetto;
- se non è nullo allora specifica il valore da impostare per la proprietà corrispondente (questa può essere una sostituzione o un'aggiunta alla cartella che viene modificata).

Eventuali proprietà impostate dal server possono essere incluse nella modifica se il loro valore è identico al valore corrente del server (prima di applicare le modifiche alla cartella). In caso contrario, la modifica DEVE essere respinta con un errore di tipo “invalidProperties”.

Questa definizione consente di apportare modifiche sia inviando l'intero nuovo oggetto “Email” come patch, che inviando solo le differenze. Entrambi i metodi sono validi, e il server elabora le modifiche allo stesso modo in entrambi i casi;

### 3.5.55.2.1.4.2) UC55.2.1.4.2 - Inserimento dei “PatchObject” corrispondenti agli oggetti “Email” da rimuovere da una o più cartelle all'interno del parametro “update” del metodo “Email/set” necessario alla gestione dei contenuti di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di rimuovere una o più email da una o più cartelle e, quindi, di gestire i contenuti di quest'ultima, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per la gestione dei contenuti di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “update”. All'interno di questo parametro va definito un “PatchObject” per ogni email che il client vuole rimuovere da una o più cartelle;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito il “PatchObject” di ogni oggetto “Email” che intende rimuovere da una o più cartelle all'interno del parametro “update”, contenuto a sua volta all'interno dell'oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell'array “methodCalls” della richiesta;
- **Scenario principale:**

1. Il client specifica all'interno del parametro “update”, contenuto nell'oggetto di argomenti della chiamata di metodo, gli oggetti “PatchObject” necessari per la rimozione delle email da una o più cartelle. Questo parametro è un array di stringhe, il quale rappresenta un insieme non ordinato di patch, dove le chiavi sono un percorso nel formato JSON Pointer, con un “/” posto all'inizio implicitamente, che stanno ad indicare una particolare proprietà. In questo caso ci interessa solamente la proprietà “mailboxIds” che verrà modificata per rimuovere gli identificativi di tutte le cartelle da cui il client intende rimuovere le email.

Tutti i percorsi utilizzati come chiave devono essere conformi a specifiche restrizioni, altrimenti la modifica verrà rifiutata rispondendo con un errore “invalidPatch”.

Per quanto riguarda il valore associato a ciascun percorso:

- se è nullo allora imposta il valore predefinito, se specificato, per la proprietà corrispondente; se non specificato, rimuove la proprietà dalla cartella modificata. Se la chiave non è presente nel padre, questa è un'operazione senza effetto;
- se non è nullo allora specifica il valore da impostare per la proprietà corrispondente (questa può essere una sostituzione o un'aggiunta alla cartella che viene modificata).

Eventuali proprietà impostate dal server possono essere incluse nella modifica se il loro valore è identico al valore corrente del server (prima di applicare le modifiche alla cartella). In caso contrario, la modifica DEVE essere respinta con un errore di tipo “invalidProperties”.

Questa definizione consente di apportare modifiche sia inviando l'intero nuovo oggetto “Email” come patch, che inviando solo le differenze. Entrambi i metodi sono validi, e il server elabora le modifiche allo stesso modo in entrambi i casi;

### 3.5.55.2.1.4.3) UC55.2.1.4.3 - Inserimento dei “PatchObject” corrispondenti agli oggetti “Email” da spostare da una o più cartelle ad una o più cartelle all'interno del parametro “update” del metodo “Email/set” necessario alla gestione dei contenuti di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di spostare una o più email da una o più cartelle ad una o più cartelle e, quindi, di gestire i contenuti di quest'ultime, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per la gestione dei contenuti di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “update”. All'interno di questo parametro va definito un “PatchObject” per ogni email che il client vuole spostare da una o più cartelle ad una o più cartelle;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito il “PatchObject” di ogni oggetto “Email” che intende spostare da una o più cartelle ad una o più cartelle all'interno del parametro “update”, contenuto a sua volta all'interno dell'oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell'array “methodCalls” della richiesta;
- **Scenario principale:**

1. Il client specifica all'interno del parametro “update”, contenuto nell'oggetto di argomenti della chiamata di metodo, gli oggetti “PatchObject” necessari per lo spostamento delle email da una o più cartelle ad una o più cartelle. Questo parametro è un array di stringhe, il quale rappresenta un insieme non ordinato di patch, dove le chiavi sono un percorso nel formato JSON Pointer, con un “/” posto all'inizio implicitamente, che stanno ad indicare una particolare proprietà. In questo caso ci interessa solamente la proprietà “mailboxIds” che verrà modificata per cambiare gli identificativi di tutte le cartelle che il client intende spostare.

Tutti i percorsi utilizzati come chiave devono essere conformi a specifiche restrizioni, altrimenti

la modifica verrà rifiutata rispondendo con un errore “invalidPatch”.

Per quanto riguarda il valore associato a ciascun percorso:

- se è nullo allora imposta il valore predefinito, se specificato, per la proprietà corrispondente; se non specificato, rimuove la proprietà dalla cartella modificata. Se la chiave non è presente nel padre, questa è un’operazione senza effetto;
- se non è nullo allora specifica il valore da impostare per la proprietà corrispondente (questa può essere una sostituzione o un’aggiunta alla cartella che viene modificata).

Eventuali proprietà impostate dal server possono essere incluse nella modifica se il loro valore è identico al valore corrente del server (prima di applicare le modifiche alla cartella). In caso contrario, la modifica DEVE essere respinta con un errore di tipo “invalidProperties”.

Questa definizione consente di apportare modifiche sia inviando l’intero nuovo oggetto “Email” come patch, che inviando solo le differenze. Entrambi i metodi sono validi, e il server elabora le modifiche allo stesso modo in entrambi i casi;

### 3.5.55.2.1.5) UC55.2.1.5 - Inserimento del parametro “destroy” del metodo “Email/set” necessario alla gestione dei contenuti di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di gestire i contenuti di una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/set”, necessario per la gestione dei contenuti di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “destroy”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Email/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “destroy”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “destroy”, il quale viene impostato a null, dato che nella gestione dei contenuti di una cartella nessun oggetto deve essere distrutto;

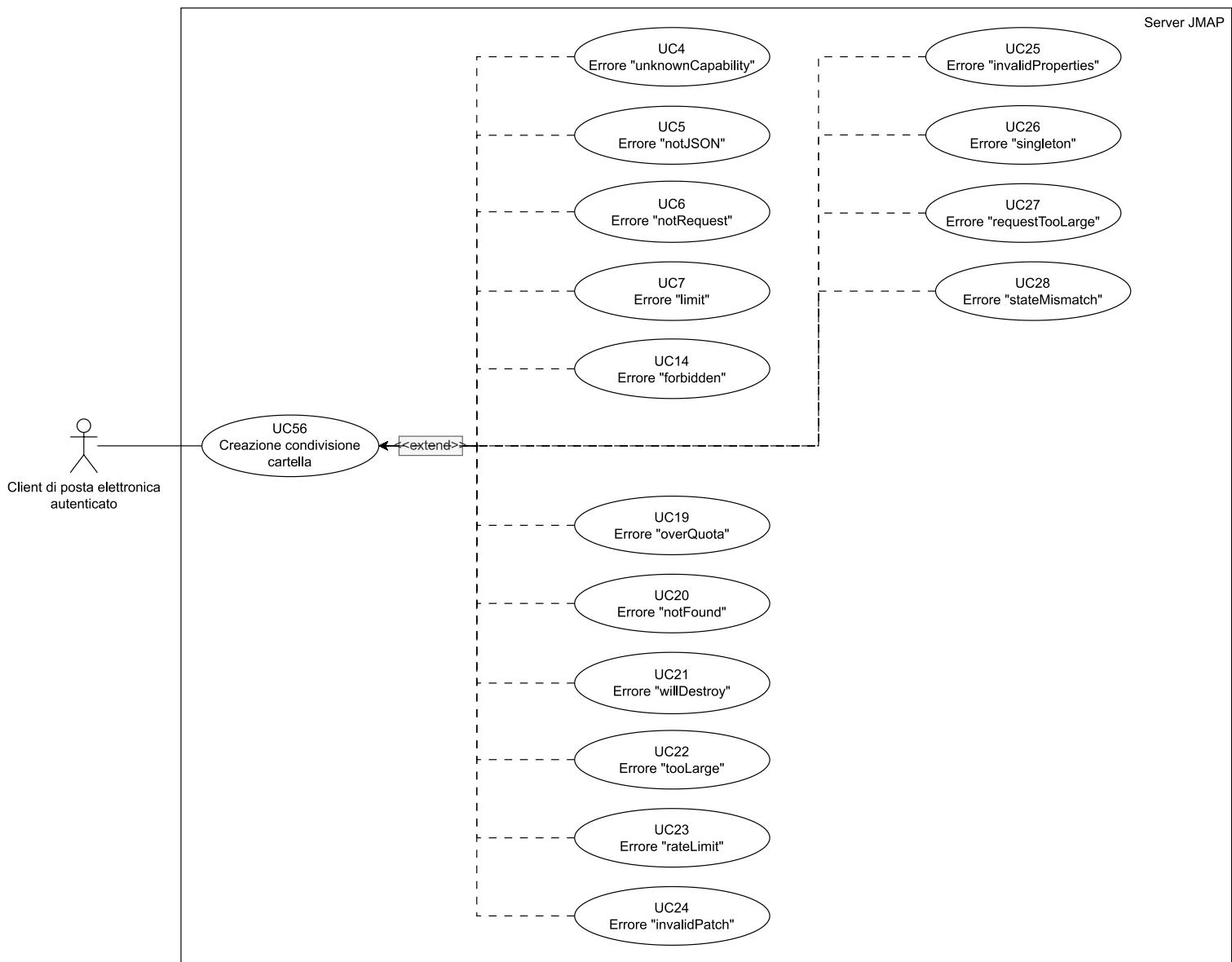


Figura 41: UC56 - Creazione condivisione cartella

### 3.5.56) UC56 - Creazione condivisione cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** La cartella è stata condivisa con successo ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per condividere una cartella;
  2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
  3. Il server elabora la richiesta API e invia una risposta JSON che contiene l'esito dell'operazione di condivisione della cartella oppure una risposta che indica il motivo del fallimento;

- **Estensioni:**

- Errore “unknownCapability”;
- Errore “notJSON”;
- Errore “notRequest”;
- Errore “limit”;
- Errore “forbidden”;
- Errore “overQuota”;
- Errore “tooLarge”;
- Errore “rateLimit”;
- Errore “notFound”;
- Errore “invalidPatch”;
- Errore “willDestroy”;
- Errore “invalidProperties”;
- Errore “singleton”;
- Errore “requestTooLarge”;
- Errore “stateMismatch”;

- **Inclusioni:** /

- **Generalizzazioni:** /

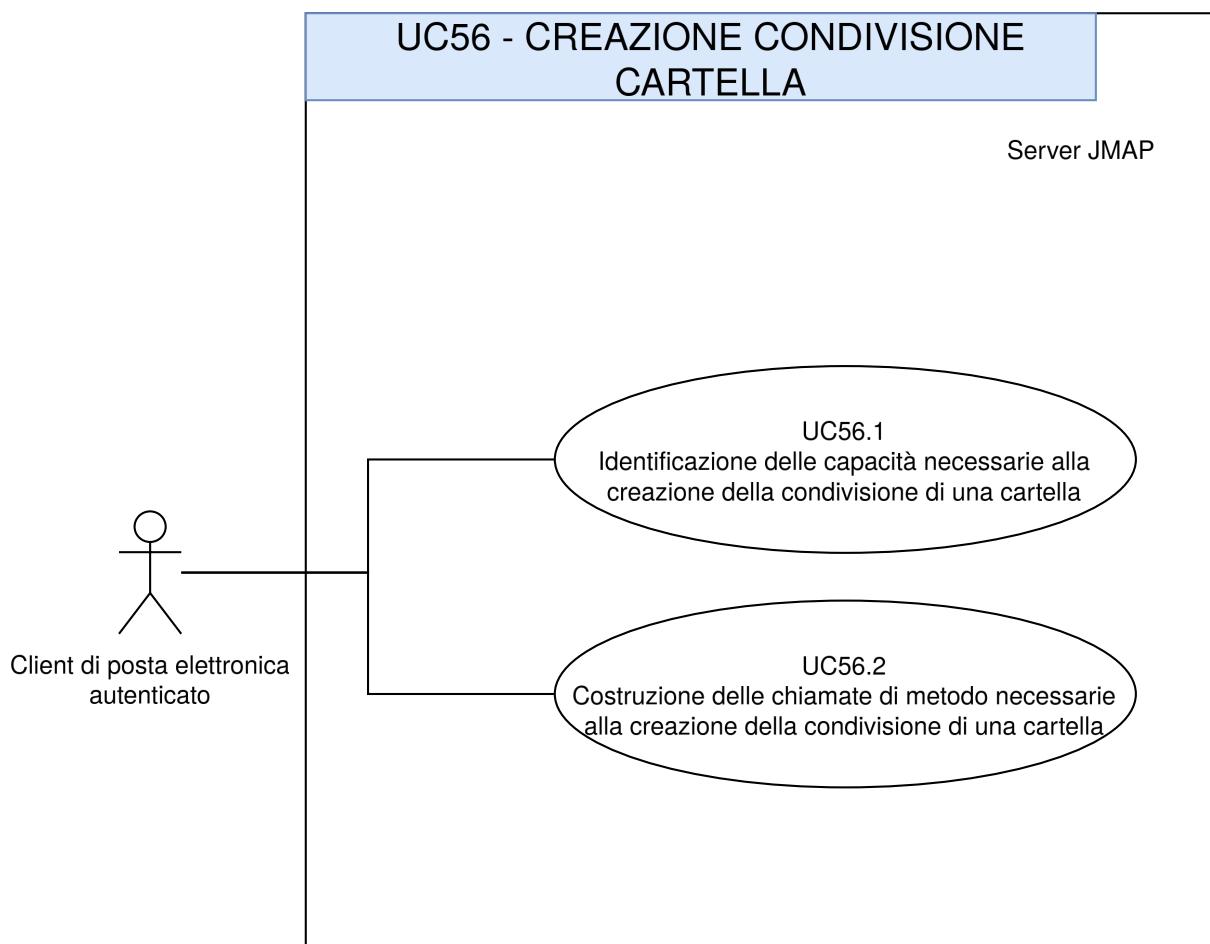


Figura 42: Sottocasi UC56 - creazione condivisione cartella

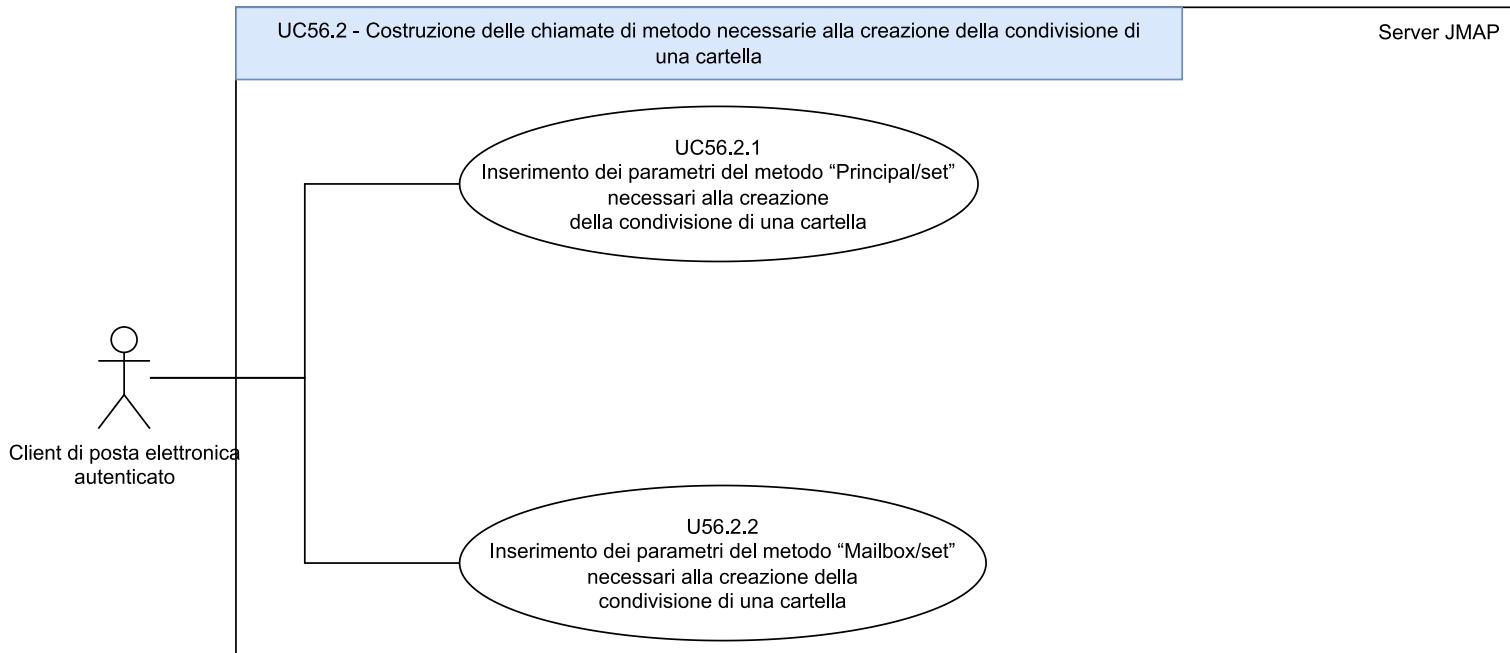
**3.5.56.1) UC56.1 - Identificazione delle capacità necessarie alla creazione della condivisione di una cartella**

- **Attore principale:** Client di posta elettronica autenticato;

- **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per condividere una cartella;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all'interno della richiesta le capacità JMAP necessarie per la condivisione di una cartella;
- **Scenario principale:**
  1. Il client include nella sezione "using" dell'oggetto Request le capacità JMAP "urn:ietf:params:jmap:core", "urn:ietf:params:jmap:mail", "urn:ietf:params:jmap:principals" e "urn:ietf:params:jmap:principals:owner";

### 3.5.56.2) UC56.2 - Costruzione delle chiamate di metodo necessarie alla creazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per condividere la cartella desiderata;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'array "methodCalls" della richiesta le chiamate di metodo necessarie per condividere la cartella desiderata;
- **Scenario principale:**
  1. Il client crea un array di chiamate di metodo denominato "methodCalls" internamente all'oggetto Request, all'interno del quale inserisce il metodo "Principal/set", un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo, ed il metodo "Mailbox/set", un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;



---

Figura 43: Sottocasi UC56.2 - Costruzione delle chiamate di metodo necessarie alla creazione della condivisione di una cartella

**3.5.56.2.1) UC56.2.1 - Inserimento dei parametri del metodo “Principal/set” necessari alla creazione della condivisione di una cartella**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Principal/set”, necessario per la creazione del Principal da associare alla condivisione di una cartella, per il quale vanno specificati i parametri necessari;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, la quale è contenuta nell’array di chiamate di metodo denominato “methodCalls”, contenuto a sua volta all’interno dell’oggetto Request;

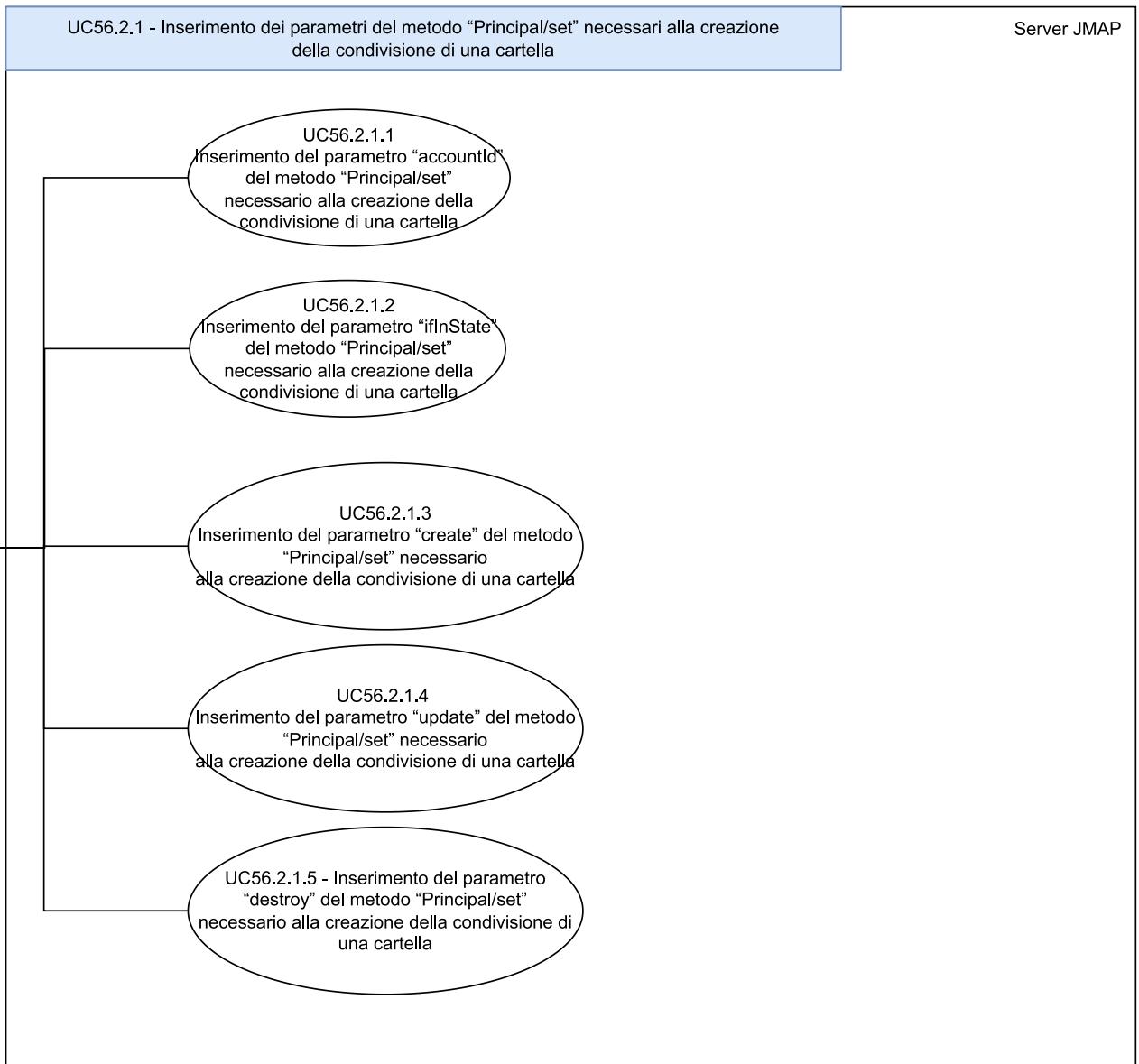


Figura 44: Sottocasi UC56.2.1 - Inserimento dei parametri del metodo “Principal/set” necessari alla creazione della condivisione di una cartella

### 3.5.56.2.1.1) UC56.2.1.1 - Inserimento del parametro “accountId” del metodo “Principal/set” necessario alla creazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Principal/set”, necessario per la creazione del Principal da associare alla condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “accountId”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “accountId”;
- **Scenario principale:**

- 
1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "accountId", il quale rappresenta l'identificativo dell'account da utilizzare;

### **3.5.56.2.1.2) UC56.2.1.2 - Inserimento del parametro "ifInState" del metodo "Principal/set" necessario alla creazione della condivisione di una cartella**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "Principal/set", necessario per la creazione del Principal da associare alla condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è "ifInState";
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Principal/set", contenuto nell'array "methodCalls" della richiesta, il parametro "ifInState";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "ifInState", il quale è un meccanismo di controllo che impedisce la sovrascrittura di dati obsoleti o incoerenti sul server. Esso garantisce che la richiesta venga eseguita solo se il client è a conoscenza dello stato attuale degli oggetti e che tale stato sia valido al momento della richiesta;

### **3.5.56.2.1.3) UC56.2.1.3 - Inserimento del parametro "create" del metodo "Principal/set" necessario alla creazione della condivisione di una cartella**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "Principal/set", necessario per la creazione del Principal da associare alla condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è "create";
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Principal/set", contenuto nell'array "methodCalls" della richiesta, il parametro "create";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "create", il quale è utilizzato per specificare l'intenzione di creare un Principal per la condivisione di una cartella. Questo parametro è una mappa, in cui le chiavi sono identificatori temporanei di creazione (creation id) assegnati dal client ed i valori sono oggetti Principal da creare. La definizione di un oggetto Principal può contenere valori predefiniti per le proprietà. Qualsiasi proprietà con un valore predefinito può essere omessa dal client. Inoltre Il client DEVE omettere qualsiasi proprietà che può essere impostata solo dal server;

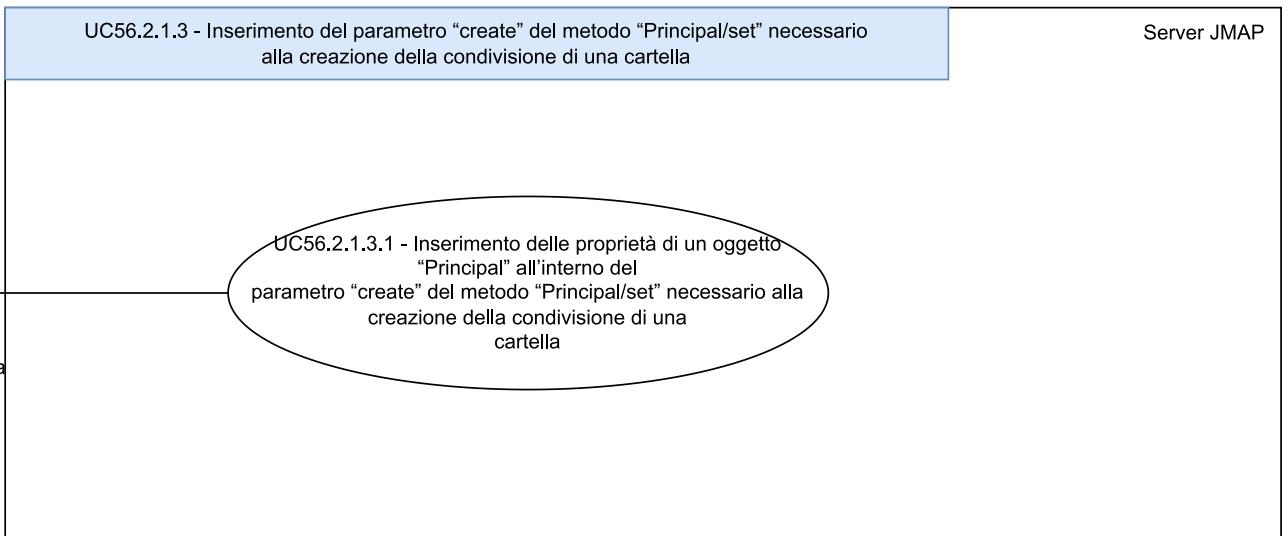


Figura 45: Sottocasi UC56.2.1.3 - Inserimento del parametro "create" del metodo "Principal/set" necessario alla creazione della condivisione di una cartella

**3.5.56.2.1.3.1) UC56.2.1.3.1 - Inserimento delle proprietà di un oggetto "Principal" all'interno del parametro "create" del metodo "Principal/set" necessario alla creazione della condivisione di una cartella**

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "Principal/set", necessario per la creazione del Principal da associare alla condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è "create". All'interno di questo parametro vanno definite le proprietà del Principal necessario per la condivisione da creare;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito le proprietà di un oggetto "Principal" all'interno del parametro "create", contenuto a sua volta all'interno dell'oggetto di argomenti della chiamata di metodo "Principal/set", contenuto nell'array "methodCalls" della richiesta;
- **Scenario principale:**
  1. Il client specifica all'interno del parametro "create", contenuto nell'oggetto di argomenti della chiamata di metodo, le proprietà dell'oggetto "Principal" da creare, tra le quali troviamo le seguenti:
    - **id:** L'identificatore del principale, ovvero un individuo, un gruppo, una posizione (come una stanza), una risorsa (come un proiettore) o qualsiasi altra entità in un ambiente collaborativo. Nell'ambiente JMAP, la condivisione è generalmente configurata assegnando diritti su determinati dati (in questo caso una cartella) all'interno di un account ad altri principali;
    - **type:** Questo deve essere uno dei seguenti valori:
      - **individual:** Rappresenta una singola persona;
      - **group:** Rappresenta un gruppo di persone;
      - **resource:** Rappresenta una risorsa, ad esempio un proiettore;
      - **location:** Rappresenta una posizione;
      - **other:** Rappresenta un altro principale non definito;
    - **name:** Il nome del principale, ad esempio "Jane Doe" o "Sala 4B";

- **description**: Una descrizione più lunga del principale, ad esempio dettagli sulle strutture di una risorsa, o null se non è disponibile alcuna descrizione;
- **email**: Un indirizzo email per il principale, o null se non è disponibile alcun indirizzo email;
- **timeZone**: Il fuso orario per questo principale, se noto. Se non è nullo, il valore deve essere un identificativo del fuso orario dalla base di dati dei fusi orari IANA (TZDB);
- **capabilities**: Una mappa di URI di capacità JMAP a informazioni specifiche del dominio sul principale in relazione a quella capacità, come definito nel documento che ha registrato la capacità;
- **accounts**: Una mappa dell’identificativo degli account a cui vogliamo condividere la cartella all’oggetto Account proprietario della cartella.

#### 3.5.56.2.1.4) UC56.2.1.4 - Inserimento del parametro “update” del metodo “Principal/set” necessario alla creazione della condivisione di una cartella

- **Attore principale**: Client di posta elettronica autenticato;
- **Descrizione**: Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Principal/set”, necessario per la creazione del Principal da associare alla condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “update”;
- **Precondizioni**: Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni**: Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “update”;
- **Scenario principale**:
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “update”, il quale viene impostato a null, dato che nella la creazione del Principal da associare alla condivisione di una cartella nessun oggetto deve essere aggiornato;

#### 3.5.56.2.1.5) UC56.2.1.5 - Inserimento del parametro “destroy” del metodo “Principal/set” necessario alla creazione della condivisione di una cartella

- **Attore principale**: Client di posta elettronica autenticato;
- **Descrizione**: Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Principal/set”, necessario per la creazione del Principal da associare alla condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “destroy”;
- **Precondizioni**: Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni**: Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “destroy”;
- **Scenario principale**:
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “destroy”, il quale viene impostato a null, dato che nella creazione del Principal da associare alla condivisione di una cartella di una cartella nessun oggetto deve essere distrutto;

#### 3.5.56.2.2) UC56.2.2 - Inserimento dei parametri del metodo “Mailbox/set” necessari alla creazione della condivisione di una cartella

- **Attore principale**: Client di posta elettronica autenticato;

- **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "Mailbox/set", necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/set", contenuto nell'array "methodCalls" della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/set", la quale è contenuta nell'array di chiamate di metodo denominato "methodCalls", contenuto a sua volta all'interno dell'oggetto Request;

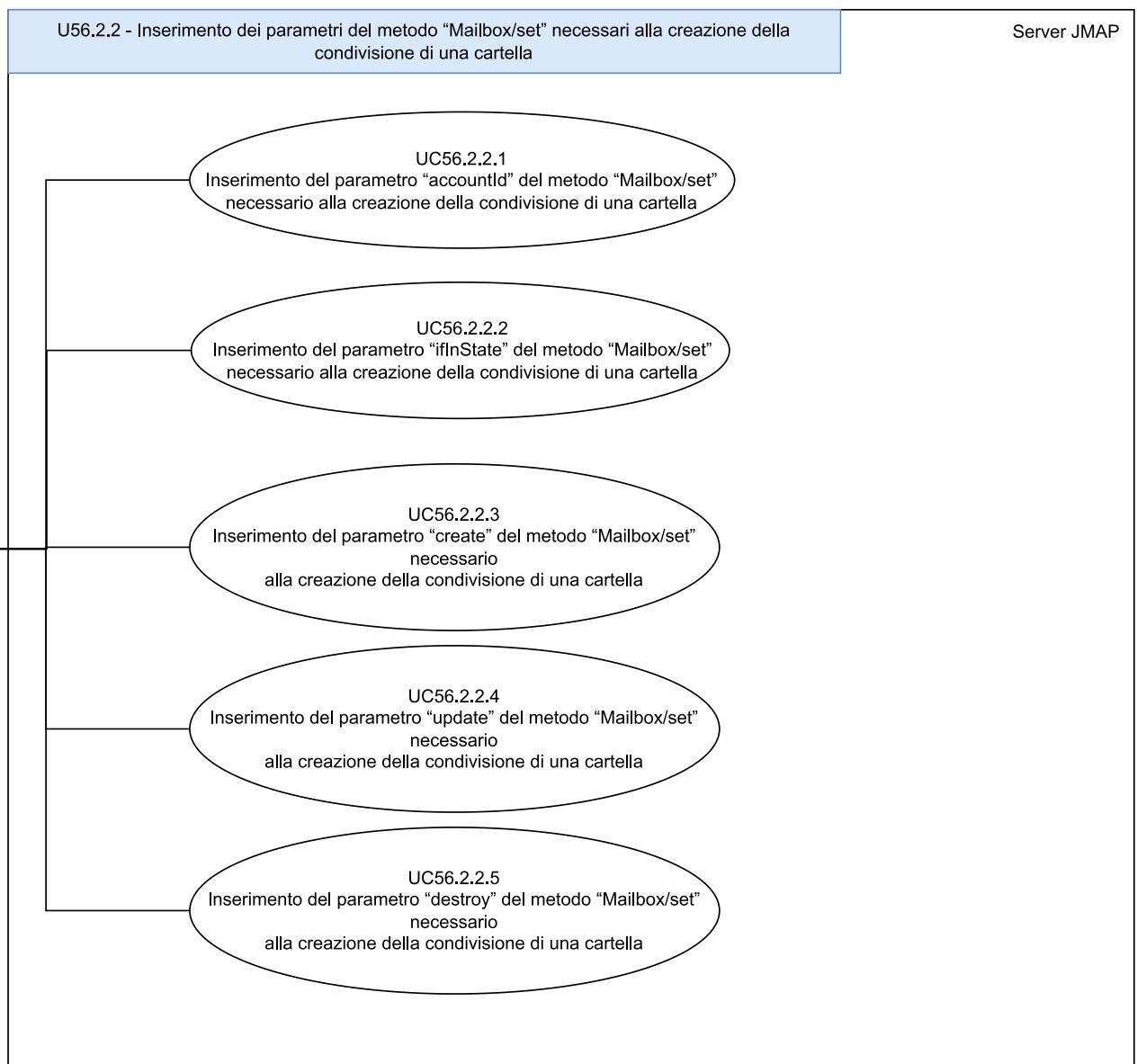


Figura 46: Sottocasi UC56.2.2 - Inserimento dei parametri del metodo "Mailbox/set" necessari alla creazione della condivisione di una cartella

### 3.5.56.2.2.1) UC56.2.2.1 - Inserimento del parametro “accountId” del metodo “Mailbox/set” necessario alla creazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Mailbox/set”, necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “accountId”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “accountId”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “accountId”, il quale rappresenta l’identificativo dell’account da utilizzare;

### 3.5.56.2.2.2) UC56.2.2.2 - Inserimento del parametro “ifInState” del metodo “Mailbox/set” necessario alla creazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Mailbox/set”, necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “ifInState”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “ifInState”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “ifInState”, il quale è un meccanismo di controllo che impedisce la sovrascrittura di dati obsoleti o incoerenti sul server. Esso garantisce che la richiesta venga eseguita solo se il client è a conoscenza dello stato attuale degli oggetti e che tale stato sia valido al momento della richiesta;

### 3.5.56.2.2.3) UC56.2.2.3 - Inserimento del parametro “create” del metodo “Mailbox/set” necessario alla creazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Mailbox/set”, necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “create”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “create”;
- **Scenario principale:**

- Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "create", il quale viene impostato a null, dato che nella modifica della proprietà di condivisione di una cartella nessun oggetto deve essere creato;

#### 3.5.56.2.2.4) UC56.2.2.4 - Inserimento del parametro "update" del metodo "Mailbox/set" necessario alla creazione della condivisione di una cartella

- Attore principale:** Client di posta elettronica autenticato;
- Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "Mailbox/set", necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è "update";
- Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/set", contenuto nell'array "methodCalls" della richiesta, il parametro "update";
- Scenario principale:**
  - Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "update", il quale è utilizzato per specificare l'intenzione di modificare una cartella per cambiare la proprietà di condivisione di quest'ultima. Questo parametro è una mappa che associa un identificativo a un oggetto di tipo patch (PatchObject) da applicare all'oggetto Mailbox corrente con quell'identificativo. Un PatchObject rappresenta un insieme non ordinato di modifiche;

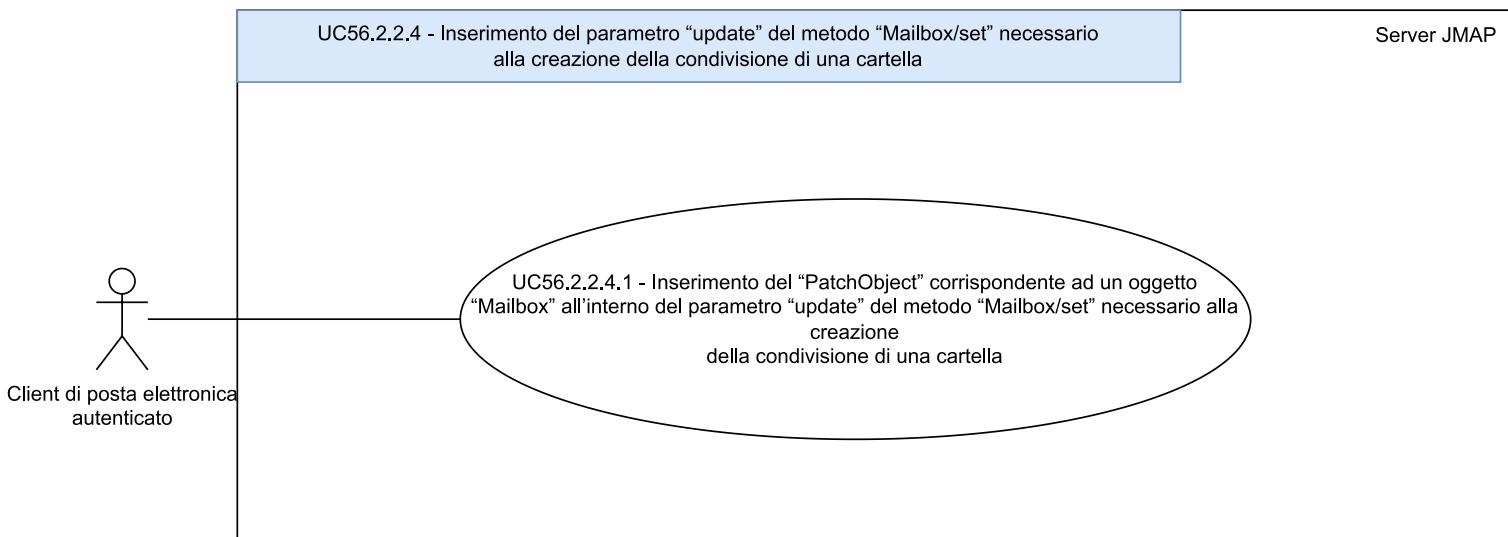


Figura 47: Sottocaso UC56.2.2.4 - Inserimento del parametro "update" del metodo "Mailbox/set" necessario alla creazione della condivisione di una cartella

#### 3.5.56.2.2.4.1) UC56.2.2.4.1 - Inserimento del "PatchObject" corrispondente ad un oggetto "Mailbox" all'interno del parametro "update" del metodo "Mailbox/set" necessario alla creazione della condivisione di una cartella

- Attore principale:** Client di posta elettronica autenticato;
- Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "Mailbox/set", necessario per la modifica della proprietà di condivisione di una cartella,

per il quale vanno specificati i parametri necessari. Uno di questi è “update”. All’interno di questo parametro va definito un “PatchObject”, il quale rappresenta un insieme non ordinato di modifiche;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito il “PatchObject” di un oggetto “Mailbox” all’interno del parametro “update”, contenuto a sua volta all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta;
- **Scenario principale:**

1. Il client specifica all’interno del parametro “update”, contenuto nell’oggetto di argomenti della chiamata di metodo, l’oggetto “PatchObject” necessario per la modifica della proprietà di condivisione di una cartella. Questo è un array di stringhe, il quale rappresenta un insieme non ordinato di patch, dove le chiavi sono un percorso nel formato JSON Pointer, con un “/” posto all’inizio implicitamente, che stanno ad indicare una particolare proprietà. In questo caso è interessata solamente la proprietà “shareWith”, la quale serve a rappresentare che Principal hanno accesso alla cartella e quali diritti hanno. Essa è una mappa in cui la chiave è l’identificativo del principale con cui si sta condividendo e il valore è un oggetto che rappresenta i diritti concessi a quel principale. La chiave corrisponderà al Principal creato in precedenza.

Tutti i percorsi utilizzati come chiave devono essere conformi a specifiche restrizioni, altrimenti la modifica verrà rifiutata rispondendo con un errore “invalidPatch”.

Per quanto riguarda il valore associato a ciascun percorso:

- se è nullo allora imposta il valore predefinito, se specificato, per la proprietà corrispondente; se non specificato, rimuove la proprietà dalla cartella modificata. Se la chiave non è presente nel padre, questa è un’operazione senza effetto;
- se non è nullo allora specifica il valore da impostare per la proprietà corrispondente (questa può essere una sostituzione o un’aggiunta alla cartella che viene modificata).

Eventuali proprietà impostate dal server possono essere incluse nella modifica se il loro valore è identico al valore corrente del server (prima di applicare le modifiche alla cartella). In caso contrario, la modifica DEVE essere respinta con un errore di tipo “invalidProperties”.

Questa definizione consente di apportare modifiche sia inviando l’intero nuovo oggetto “Mailbox” come patch, che inviando solo le differenze. Entrambi i metodi sono validi, e il server elabora le modifiche allo stesso modo in entrambi i casi;

### 3.5.56.2.2.5) UC56.2.2.5 - Inserimento del parametro “destroy” del metodo “Mailbox/set” necessario alla creazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
  - **Descrizione:** Un client di posta elettronica, con lo scopo di condividere una cartella, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Mailbox/set”, necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “destroy”;
  - **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
  - **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “destroy”;
  - **Scenario principale:**
1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “destroy”, il quale viene impostato a null, dato che nella modifica della proprietà di condivisione di una cartella nessun oggetto deve essere distrutto;

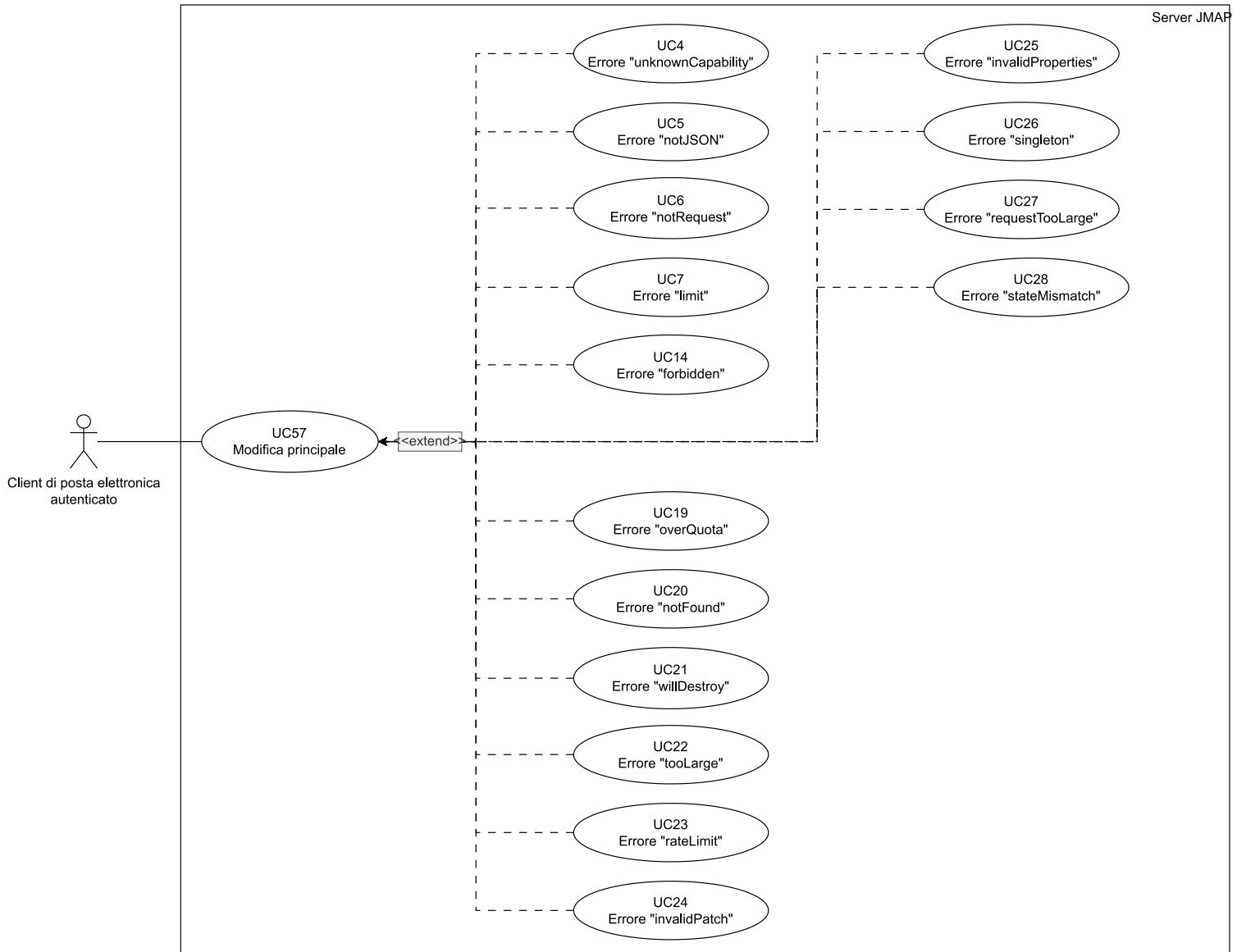


Figura 48: UC57 - Modifica principale

### 3.5.57) UC57 - Modifica principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare un principale, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il principale è stato modificato con successo ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per modificare un principale;
  2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
  3. Il server elabora la richiesta API e invia una risposta JSON che contiene l'esito dell'operazione di modifica del principale;

- **Estensioni:**
  - Errore “unknownCapability”;
  - Errore “notJSON”;
  - Errore “notRequest”;
  - Errore “limit”;
  - Errore “forbidden”;
  - Errore “overQuota”;
  - Errore “tooLarge”;
  - Errore “rateLimit”;
  - Errore “notFound”;
  - Errore “invalidPatch”;
  - Errore “willDestroy”;
  - Errore “invalidProperties”;
  - Errore “singleton”;
  - Errore “requestTooLarge”;
  - Errore “stateMismatch”;
- **Inclusioni:** /
- **Generalizzazioni:** /

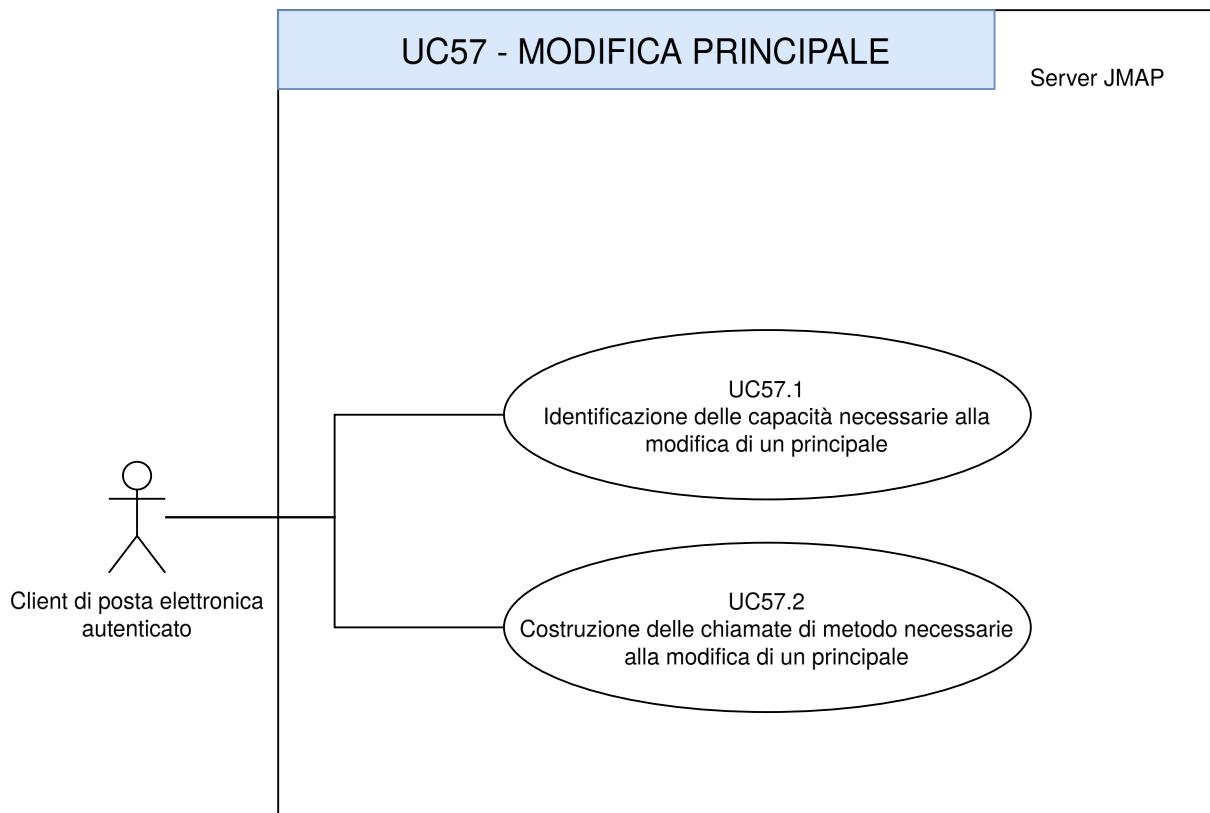


Figura 49: Sottocasi UC57 - Modifica principale

### 3.5.57.1) UC57.1 - Identificazione delle capacità necessarie alla modifica di un principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare un principale, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per la modifica di un principale;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all'interno della richiesta le capacità JMAP necessarie per la modifica di un principale;
- **Scenario principale:**
  1. Il client include nella sezione “using” dell’oggetto Request le capacità JMAP “urn:ietf:params:jmap:core”, “urn:ietf:params:jmap:mail”, “urn:ietf:params:jmap:principals” e “urn:ietf:params:jmap:principals:owner”;

### 3.5.57.2) UC57.2 - Costruzione delle chiamate di metodo necessarie alla modifica di un principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare un principale, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per la modifica di un principale;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’array “methodCalls” della richiesta le chiamate di metodo necessarie per la modifica di un principale;
- **Scenario principale:**
  1. Il client crea un array di chiamate di metodo denominato “methodCalls” internamente all’oggetto Request, all’interno del quale inserisce il metodo “Principal/set”, un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

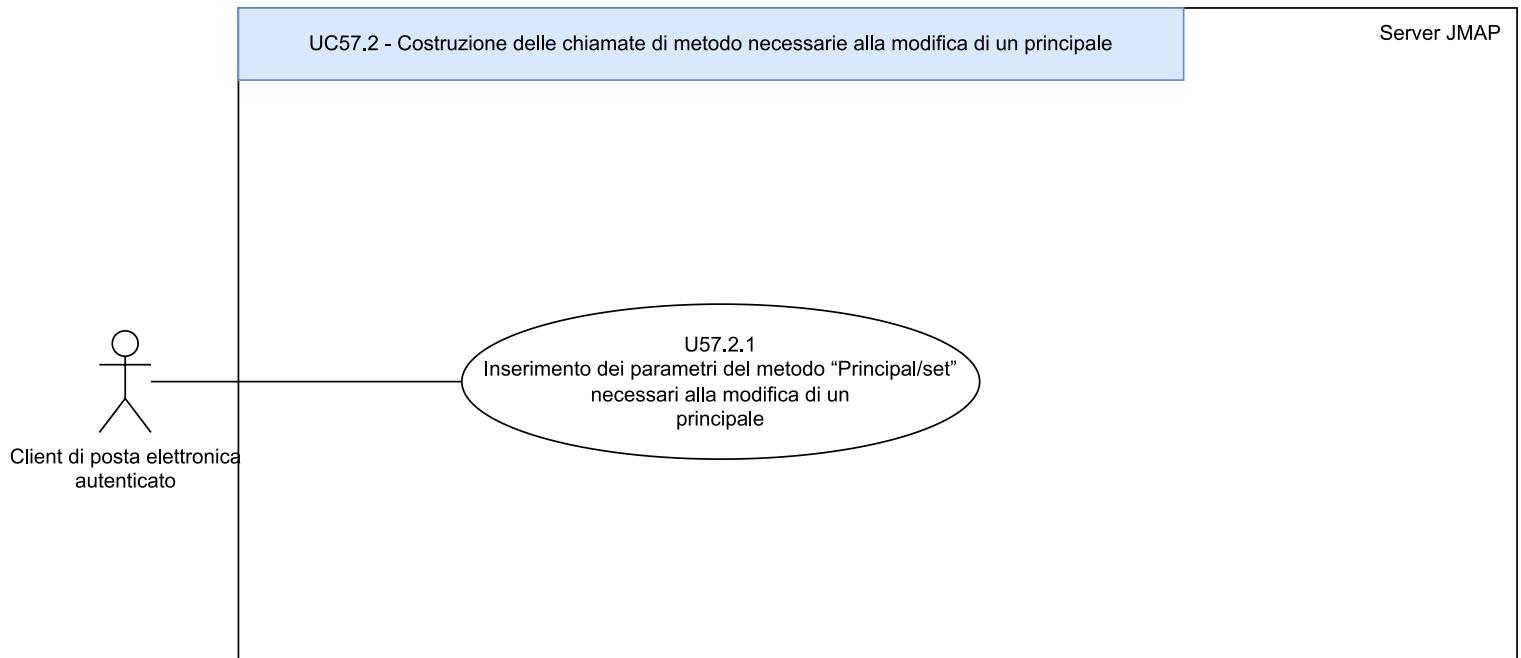


Figura 50: Sottocasi UC57.2 - Costruzione delle chiamate di metodo necessarie alla modifica di un principale

#### 3.5.57.2.1) UC57.2.1 - Inserimento dei parametri del metodo “Principal/set” necessari alla modifica di un principale

- **Attore principale:** Client di posta elettronica autenticato;

- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare un principale, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Principal/set", necessario per la modifica di un principale, per il quale vanno specificati i parametri necessari;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Principal/set", contenuto nell'array "methodCalls" della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all'interno dell'oggetto di argomenti della chiamata di metodo "Principal/set", la quale è contenuta nell'array di chiamate di metodo denominato "methodCalls", contenuto a sua volta all'interno dell'oggetto Request;

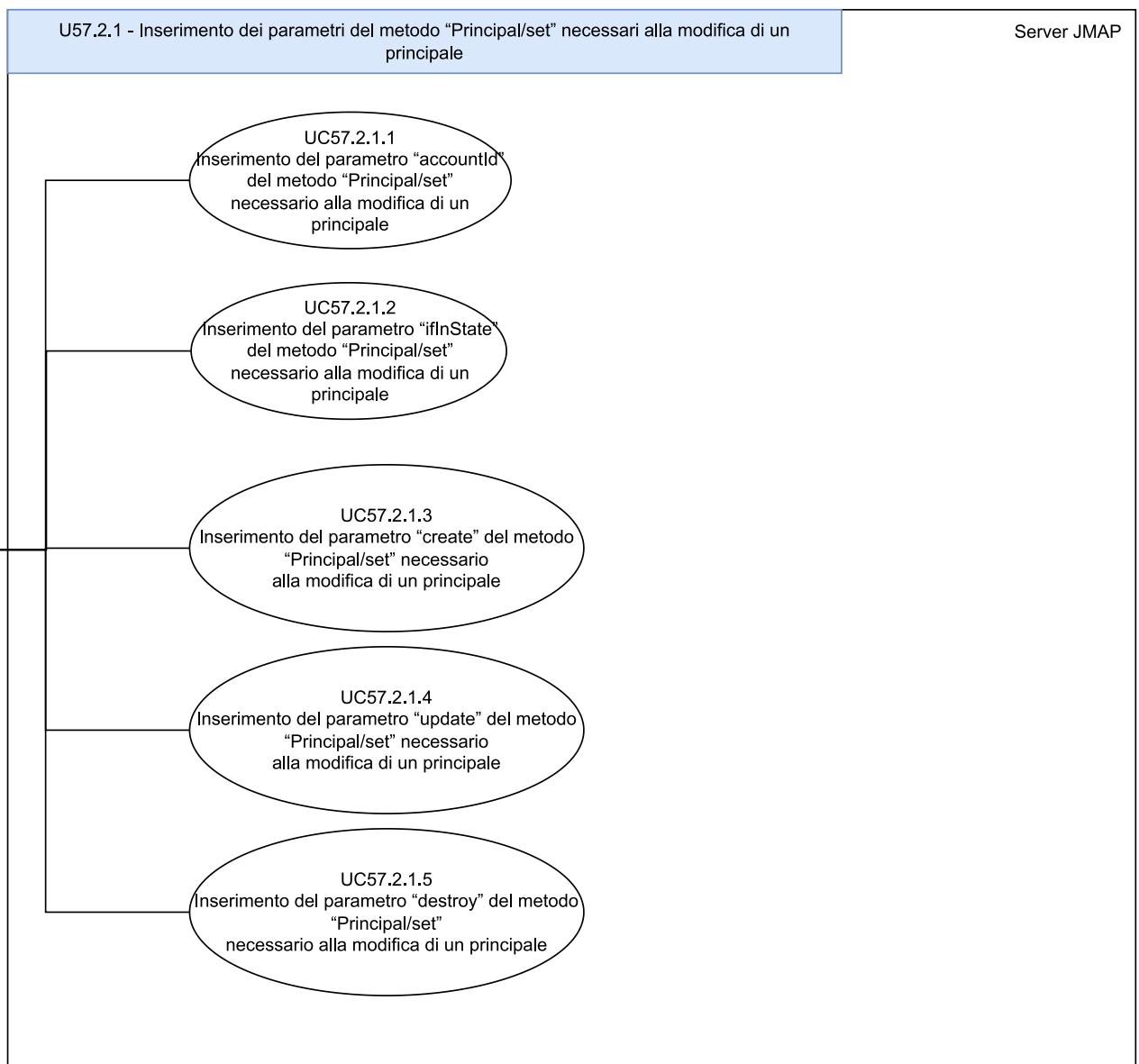


Figura 51: Sottocasi UC57.2.1 - Inserimento dei parametri del metodo "Principal/set" necessari alla modifica di un principale

### 3.5.57.2.1.1) UC57.2.1.1 - Inserimento del parametro “accountId” del metodo “Principal/set” necessario alla modifica di un principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare un principale, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Principal/set”, necessario per la modifica di un principale, per il quale vanno specificati i parametri necessari. Uno di questi è “accountId”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “accountId”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “accountId”, il quale rappresenta l’identificativo dell’account da utilizzare;

### 3.5.57.2.1.2) UC57.2.1.2 - Inserimento del parametro “ifInState” del metodo “Principal/set” necessario alla modifica di un principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare un principale, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Principal/set”, necessario per la modifica di un principale, per il quale vanno specificati i parametri necessari. Uno di questi è “ifInState”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “ifInState”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “ifInState”, il quale è un meccanismo di controllo che impedisce la sovrascrittura di dati obsoleti o incoerenti sul server. Esso garantisce che la richiesta venga eseguita solo se il client è a conoscenza dello stato attuale degli oggetti e che tale stato sia valido al momento della richiesta;

### 3.5.57.2.1.3) UC57.2.1.3 - Inserimento del parametro “create” del metodo “Principal/set” necessario alla modifica di un principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare un principale, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Principal/set”, necessario per la modifica di un principale, per il quale vanno specificati i parametri necessari. Uno di questi è “create”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “create”;
- **Scenario principale:**

- Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "create", il quale viene impostato a null, dato che nella modifica di un principale nessun oggetto deve essere creato;

#### 3.5.57.2.1.4) UC57.2.1.4 - Inserimento del parametro "update" del metodo "Principal/set" necessario alla modifica di un principale

- Attore principale:** Client di posta elettronica autenticato;
- Descrizione:** Un client di posta elettronica, con lo scopo di modificare un principale, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Principal/set", necessario per la modifica di un principale, per il quale vanno specificati i parametri necessari. Uno di questi è "update";
- Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Principal/set", contenuto nell'array "methodCalls" della richiesta, il parametro "update";
- Scenario principale:**
  - Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "update", il quale è utilizzato per specificare l'intenzione di modificare un principale. Questo parametro è una mappa che associa un identificativo a un oggetto di tipo patch (PatchObject) da applicare all'oggetto Principal corrente con quell'identificativo. Un PatchObject rappresenta un insieme non ordinato di modifiche;

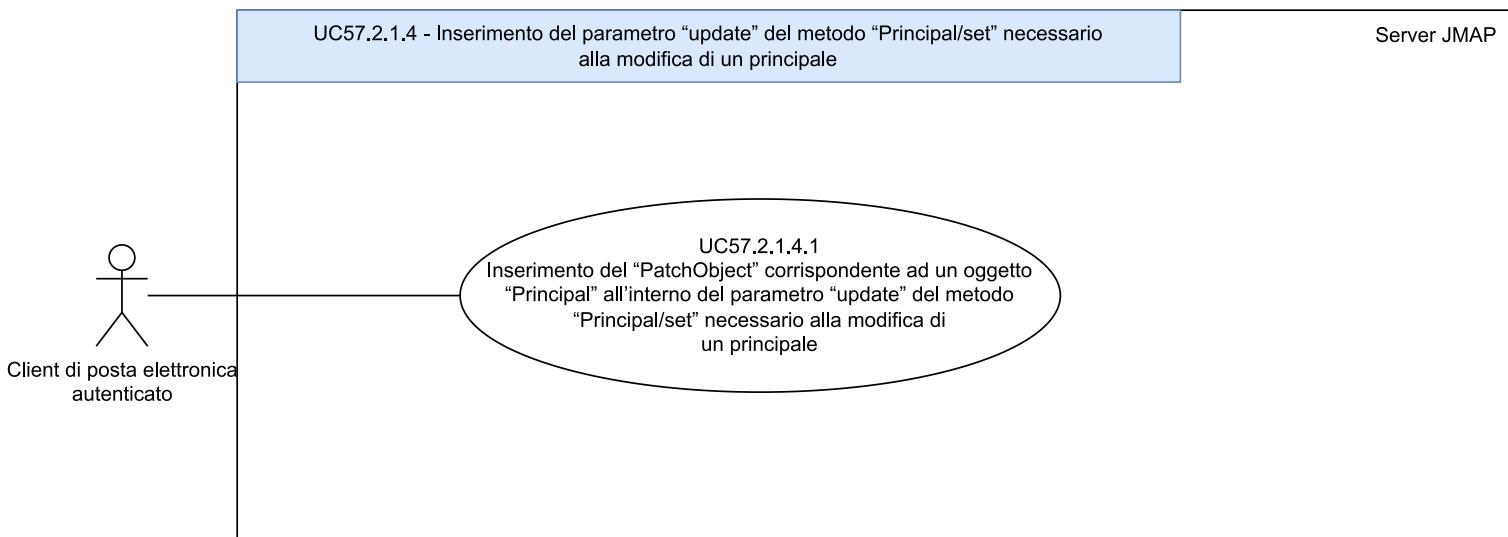


Figura 52: Sottocaso UC57.2.1.4 - Inserimento del parametro "update" del metodo "Principal/set" necessario alla modifica di un principale

#### 3.5.57.2.1.4.1) UC57.2.1.4.1 - Inserimento del "PatchObject" corrispondente ad un oggetto "Principal" all'interno del parametro "update" del metodo "Principal/set" necessario alla modifica di un principale

- Attore principale:** Client di posta elettronica autenticato;
- Descrizione:** Un client di posta elettronica, con lo scopo di modificare un principale, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Principal/set", necessario per la modifica di un principale, per il quale vanno specificati

i parametri necessari. Uno di questi è “update”. All’interno di questo parametro va definito un “PatchObject”, il quale rappresenta un insieme non ordinato di modifiche;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito il “PatchObject” di un oggetto “Principal” all’interno del parametro “update”, contenuto a sua volta all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta;
- **Scenario principale:**

1. Il client specifica all’interno del parametro “update”, contenuto nell’oggetto di argomenti della chiamata di metodo, l’oggetto “PatchObject” necessario per la modifica di un principale. Questo è un array di stringhe, il quale rappresenta un insieme non ordinato di patch, dove le chiavi sono un percorso nel formato JSON Pointer, con un “/” posto all’inizio implicitamente, che stanno ad indicare una particolare proprietà.

Tutti i percorsi utilizzati come chiave devono essere conformi a specifiche restrizioni, altrimenti la modifica verrà rifiutata rispondendo con un errore “invalidPatch”.

Per quanto riguarda il valore associato a ciascun percorso:

- se è nullo allora imposta il valore predefinito, se specificato, per la proprietà corrispondente; se non specificato, rimuove la proprietà dalla cartella modificata. Se la chiave non è presente nel padre, questa è un’operazione senza effetto;
- se non è nullo allora specifica il valore da impostare per la proprietà corrispondente (questa può essere una sostituzione o un’aggiunta alla cartella che viene modificata).

Eventuali proprietà impostate dal server possono essere incluse nella modifica se il loro valore è identico al valore corrente del server (prima di applicare le modifiche alla cartella). In caso contrario, la modifica DEVE essere respinta con un errore di tipo “invalidProperties”.

Questa definizione consente di apportare modifiche sia inviando l’intero nuovo oggetto “Principal” come patch, che inviando solo le differenze. Entrambi i metodi sono validi, e il server elabora le modifiche allo stesso modo in entrambi i casi;

### 3.5.57.2.1.5) UC57.2.1.5 - Inserimento del parametro “destroy” del metodo “Principal/set” necessario alla modifica di un principale

- **Attore principale:** Client di posta elettronica autenticato;
  - **Descrizione:** Un client di posta elettronica, con lo scopo di modificare un principale, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Principal/set”, necessario per la modifica di un principale, per il quale vanno specificati i parametri necessari. Uno di questi è “destroy”;
  - **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
  - **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “destroy”;
  - **Scenario principale:**
1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “destroy”, il quale viene impostato a null, dato che nella modifica di un principale nessun oggetto deve essere distrutto;

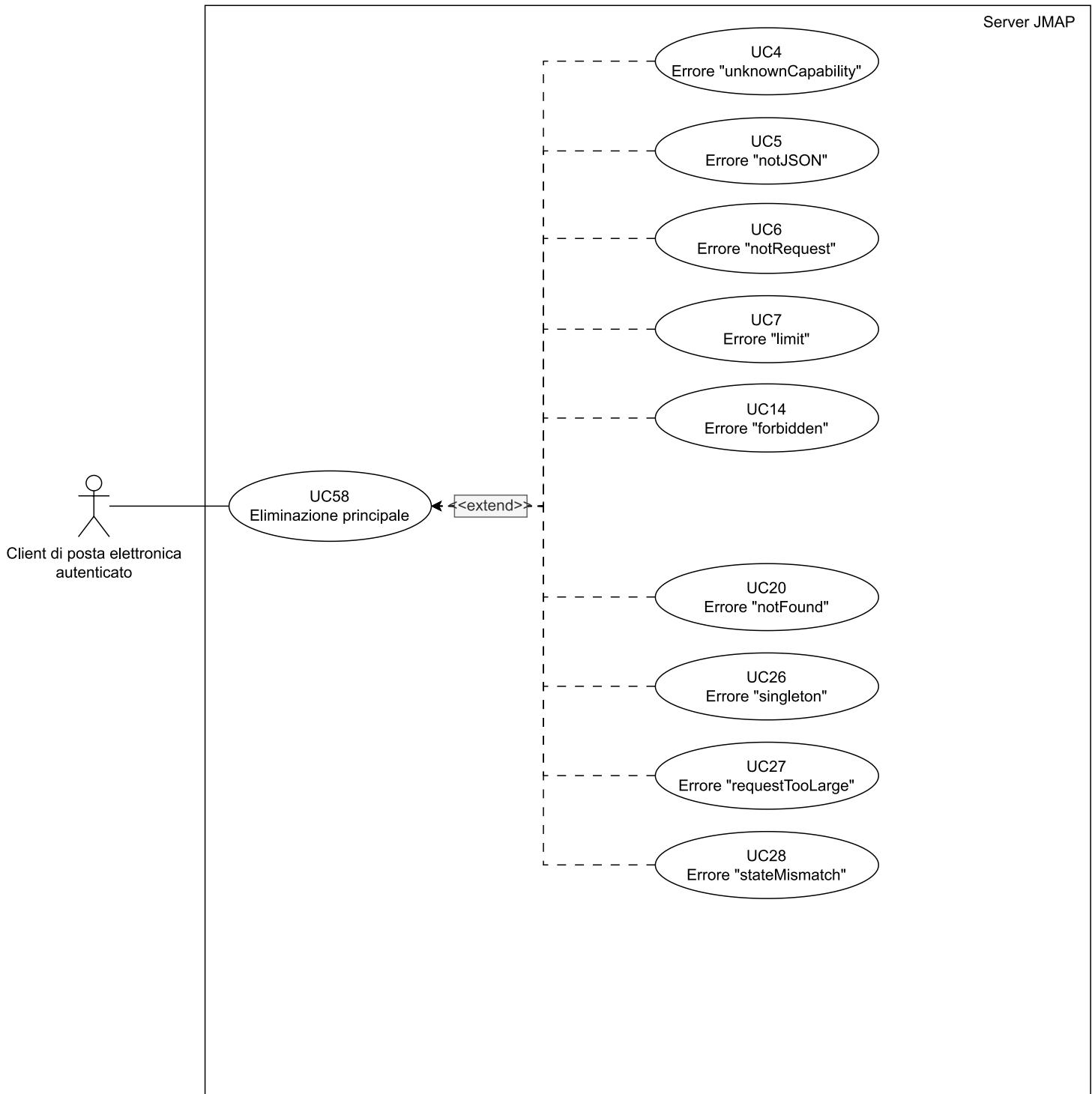


Figura 53: UC58 - Eliminazione principale

### 3.5.58) UC58 - Eliminazione principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare un principale, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il principale è stato eliminato con successo ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;

- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per eliminare un principale;
  2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
  3. Il server elabora la richiesta API e invia una risposta JSON che contiene l'esito dell'operazione di eliminazione del principale;
- **Estensioni:**
  - Errore “unknownCapability”;
  - Errore “notJSON”;
  - Errore “notRequest”;
  - Errore “limit”;
  - Errore “forbidden”;
  - Errore “notFound”;
  - Errore “singleton”;
  - Errore “requestTooLarge”;
  - Errore “stateMismatch”;
- **Inclusioni:** /
- **Generalizzazioni:** /

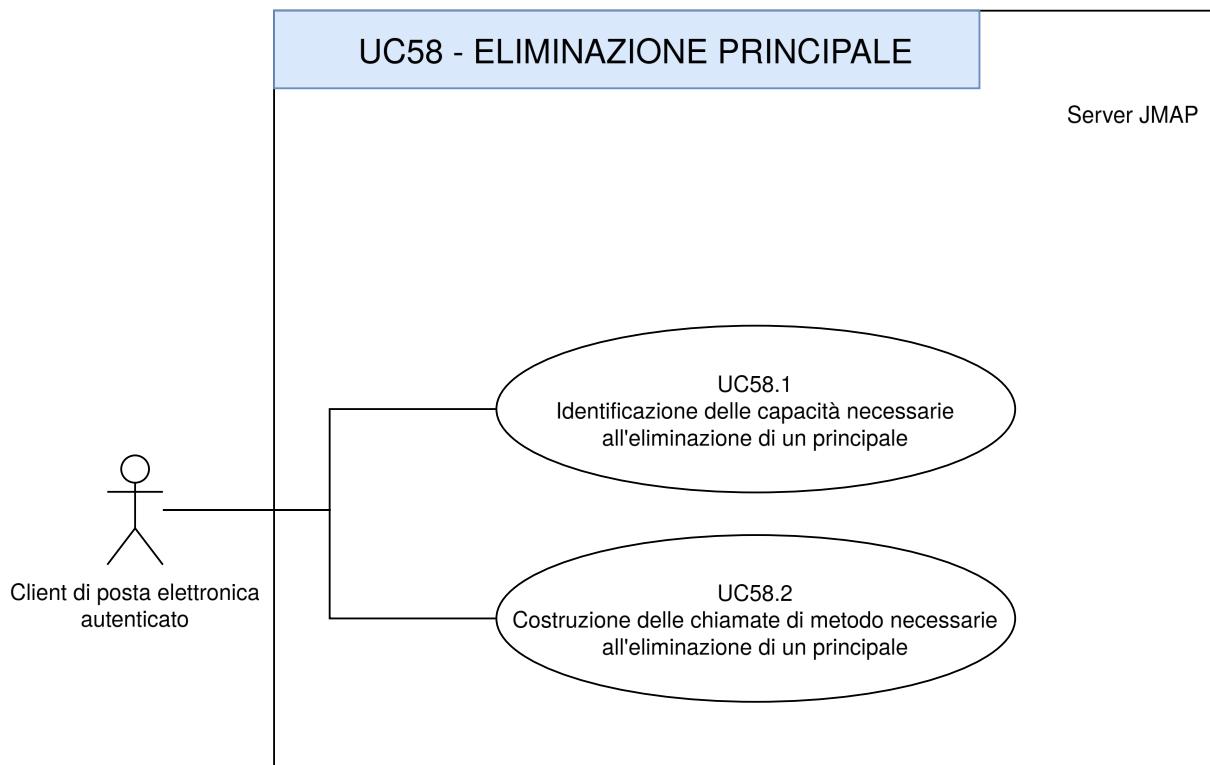


Figura 54: Sottocasi UC58 - Eliminazione principale

### 3.5.58.1) UC58.1 - Identificazione delle capacità necessarie all'eliminazione di un principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare un principale, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per la eliminazione di un principale;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all'interno della richiesta le capacità JMAP necessarie per la eliminazione di un principale;
- **Scenario principale:**
  1. Il client include nella sezione “using” dell’oggetto Request le capacità JMAP “urn:ietf:params:jmap:core”, “urn:ietf:params:jmap:mail”, “urn:ietf:params:jmap:principals” e “urn:ietf:params:jmap:principals:owner”;

### 3.5.58.2) UC58.2 - Costruzione delle chiamate di metodo necessarie all’eliminazione di un principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare un principale, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per la eliminazione di un principale;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’array “methodCalls” della richiesta le chiamate di metodo necessarie per la eliminazione di un principale;
- **Scenario principale:**
  1. Il client crea un array di chiamate di metodo denominato “methodCalls” internamente all’oggetto Request, all’interno del quale inserisce il metodo “Principal/set”, un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

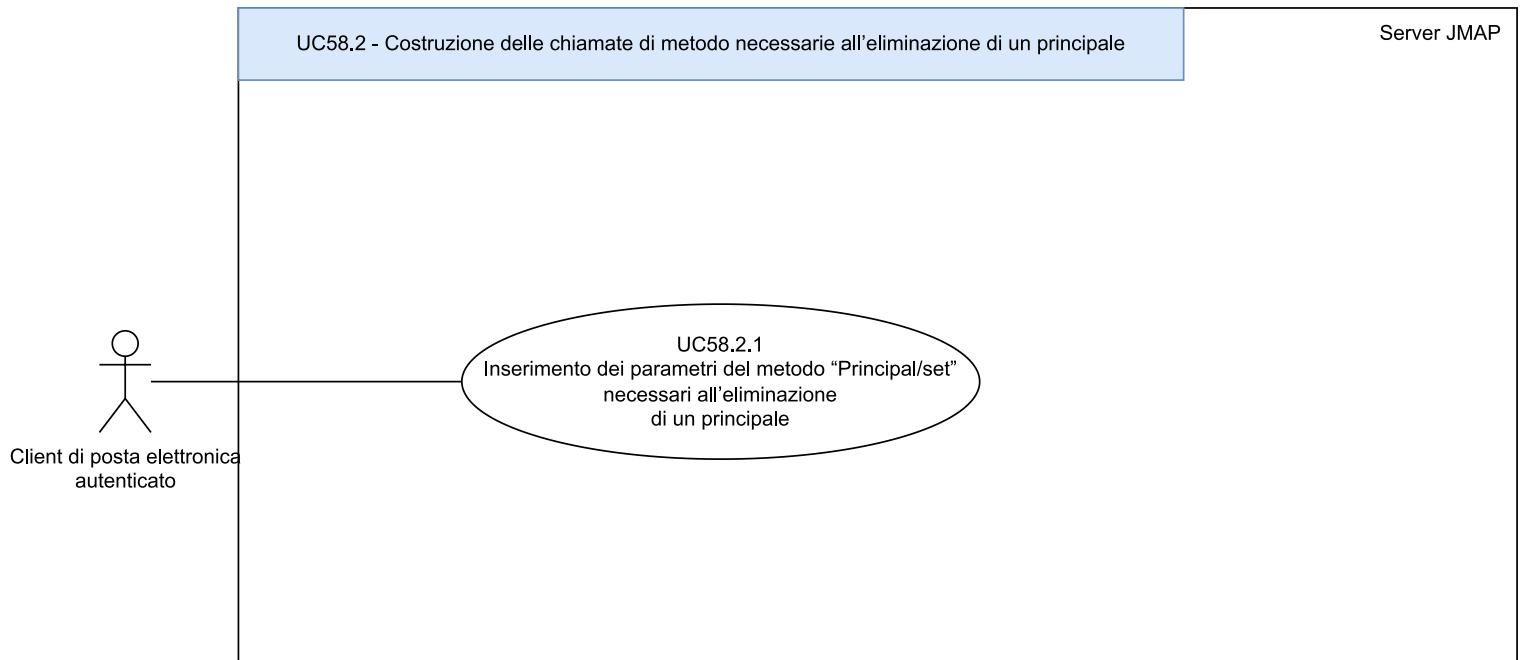


Figura 55: Sottocaso UC58.2 - Costruzione delle chiamate di metodo necessarie all’eliminazione di un principale

#### 3.5.58.2.1) UC58.2.1 - Inserimento dei parametri del metodo “Principal/set” necessari all’eliminazione di un principale

- **Attore principale:** Client di posta elettronica autenticato;

- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare un principale, deve eseguire una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Principal/set", necessario per la eliminazione di un principale, per il quale vanno specificati i parametri necessari;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Principal/set", contenuto nell'array "methodCalls" della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all'interno dell'oggetto di argomenti della chiamata di metodo "Principal/set", la quale è contenuta nell'array di chiamate di metodo denominato "methodCalls", contenuto a sua volta all'interno dell'oggetto Request;

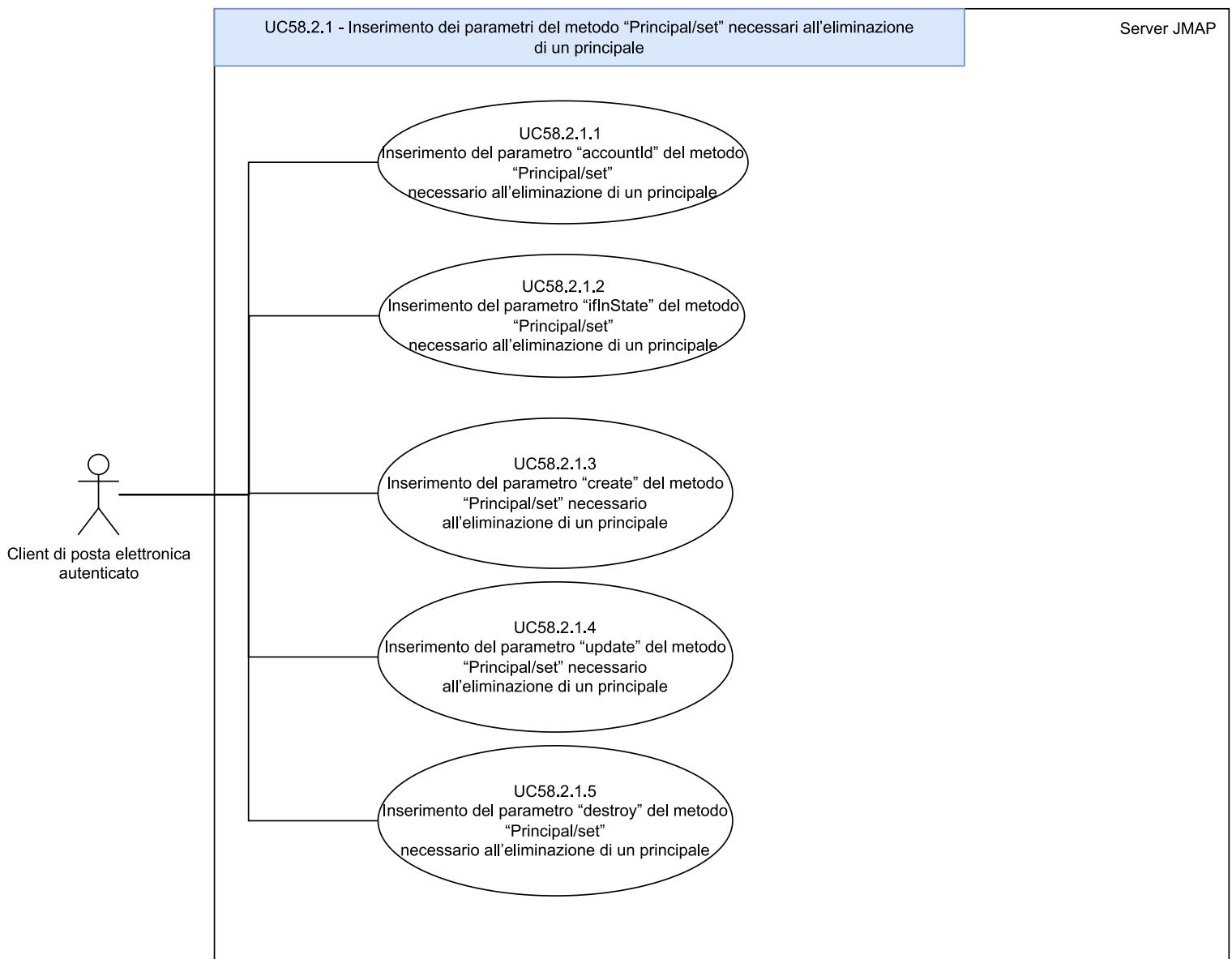


Figura 56: Sottocasi UC58.2.1 - Inserimento dei parametri del metodo "Principal/set" necessari all'eliminazione di un principale

### 3.5.58.2.1.1) UC58.2.1.1 - Inserimento del parametro “accountId” del metodo “Principal/set” necessario all’eliminazione di un principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare un principale, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Principal/set”, necessario per la eliminazione di un principale, per il quale vanno specificati i parametri necessari. Uno di questi è “accountId”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “accountId”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “accountId”, il quale rappresenta l’identificativo dell’account da utilizzare;

### 3.5.58.2.1.2) UC58.2.1.2 - Inserimento del parametro “ifInState” del metodo “Principal/set” necessario all’eliminazione di un principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare un principale, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Principal/set”, necessario per la eliminazione di un principale, per il quale vanno specificati i parametri necessari. Uno di questi è “ifInState”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “ifInState”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “ifInState”, il quale è un meccanismo di controllo che impedisce la sovrascrittura di dati obsoleti o incoerenti sul server. Esso garantisce che la richiesta venga eseguita solo se il client è a conoscenza dello stato attuale degli oggetti e che tale stato sia valido al momento della richiesta;

### 3.5.58.2.1.3) UC58.2.1.3 - Inserimento del parametro “create” del metodo “Principal/set” necessario all’eliminazione di un principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare un principale, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Principal/set”, necessario per la eliminazione di un principale, per il quale vanno specificati i parametri necessari. Uno di questi è “create”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “create”;
- **Scenario principale:**

- 
1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “create”, il quale viene impostato a null, dato che nella eliminazione di un principale nessun oggetto deve essere creato;

#### 3.5.58.2.1.4) UC58.2.1.4 - Inserimento del parametro “update” del metodo “Principal/set” necessario all’eliminazione di un principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare un principale, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Principal/set”, necessario per la eliminazione di un principale, per il quale vanno specificati i parametri necessari. Uno di questi è “update”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “update”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “update”, il quale viene impostato a null, dato che nella eliminazione di un principale nessun oggetto deve essere modificato;

#### 3.5.58.2.1.5) UC58.2.1.5 - Inserimento del parametro “destroy” del metodo “Principal/set” necessario all’eliminazione di un principale

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di eliminare un principale, deve eseguire una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Principal/set”, necessario per la eliminazione di un principale, per il quale vanno specificati i parametri necessari. Uno di questi è “destroy”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Principal/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “destroy”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “destroy”, il quale è utilizzato per specificare l’intenzione di eliminare un principale. Questo parametro è un array che contiene solamente l’identificativo del principale da eliminare;

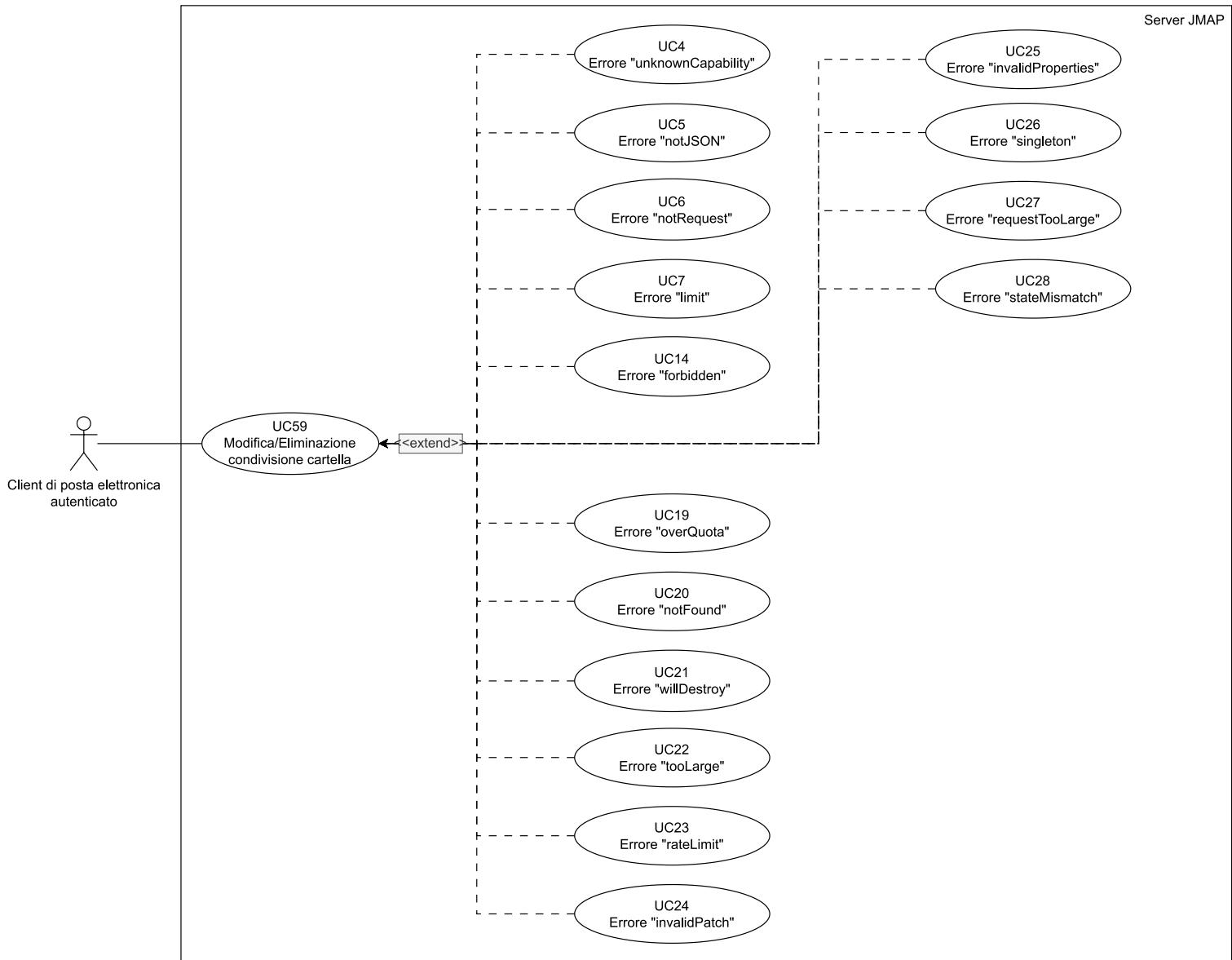


Figura 57: UC59 - Modifica/Eliminazione condivisione cartella

### 3.5.59) UC59 - Modifica/Eliminazione condivisione cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare la condivisione di una cartella (compresa l'eliminazione di quest'ultima), esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** La condivisione della cartella è stata modificata con successo ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per modificare la condivisione di una cartella;
  2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
  3. Il server elabora la richiesta API e invia una risposta JSON che contiene l'esito dell'operazione di modifica della condivisione della cartella oppure una risposta che indica il motivo del fallimento;

- **Estensioni:**
  - Errore “unknownCapability”;
  - Errore “notJSON”;
  - Errore “notRequest”;
  - Errore “limit”;
  - Errore “forbidden”;
  - Errore “overQuota”;
  - Errore “tooLarge”;
  - Errore “rateLimit”;
  - Errore “notFound”;
  - Errore “invalidPatch”;
  - Errore “willDestroy”;
  - Errore “invalidProperties”;
  - Errore “singleton”;
  - Errore “requestTooLarge”;
  - Errore “stateMismatch”;
- **Inclusioni:** /
- **Generalizzazioni:** /

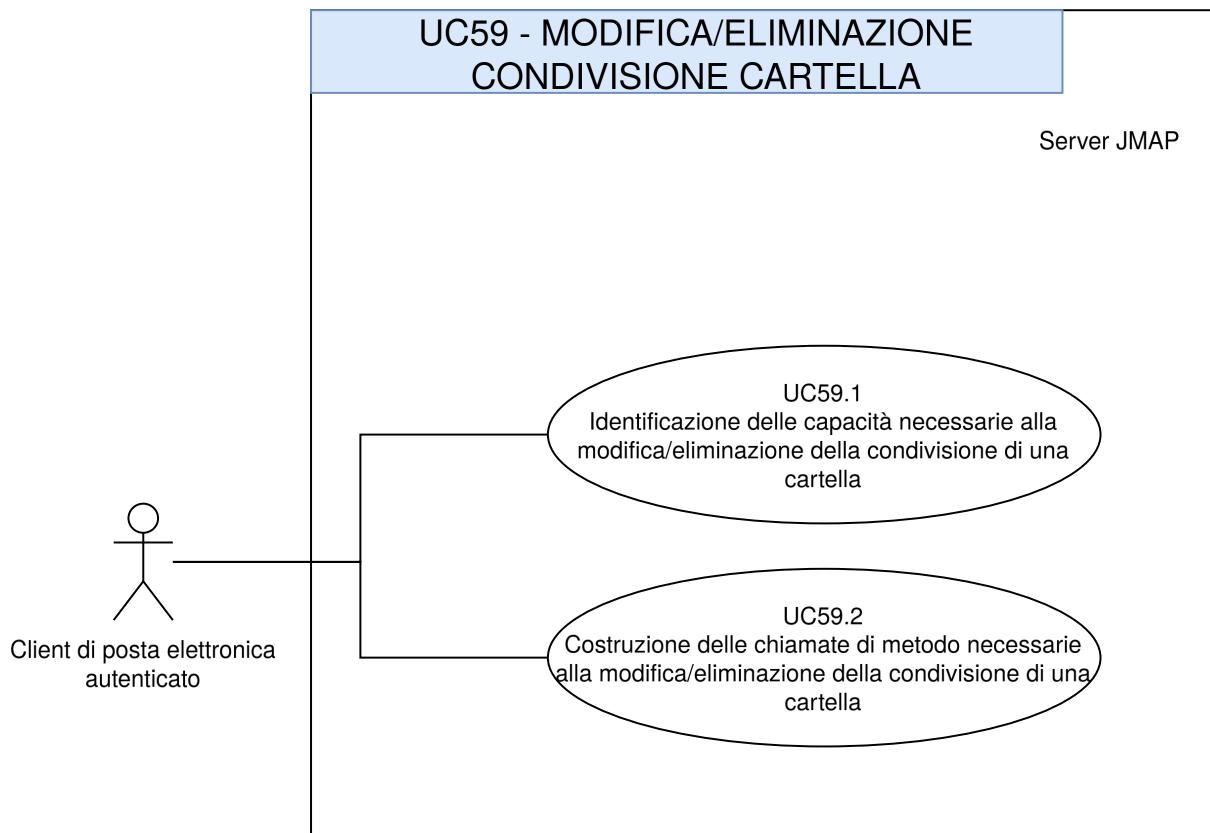


Figura 58: Sottocasi UC59 - Modifica/eliminazione condivisione cartella

### 3.5.59.1) UC59.1 - Identificazione delle capacità necessarie alla modifica/eliminazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare la condivisione di una cartella (compresa l'eliminazione di quest'ultima), esegue una richiesta POST autenticata all'URL dell'API,

il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per modificare la condivisione di una cartella;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all'interno della richiesta le capacità JMAP necessarie per modificare la condivisione di una cartella;
- **Scenario principale:**
  1. Il client include nella sezione “using” dell'oggetto Request le capacità JMAP “urn:ietf:params:jmap:core”, “urn:ietf:params:jmap:mail”, “urn:ietf:params:jmap:principals” e “urn:ietf:params:jmap:principals:owner”;

### 3.5.59.2) UC59.2 - Costruzione delle chiamate di metodo necessarie alla modifica/eliminazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare la condivisione di una cartella (compresa l'eliminazione di quest'ultima), esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per modificare la condivisione della cartella desiderata;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'array “methodCalls” della richiesta le chiamate di metodo necessarie per modificare la condivisione della cartella desiderata;
- **Scenario principale:**
  1. Il client crea un array di chiamate di metodo denominato “methodCalls” internamente all'oggetto Request, all'interno del quale inserisce il metodo “Mailbox/set”, un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

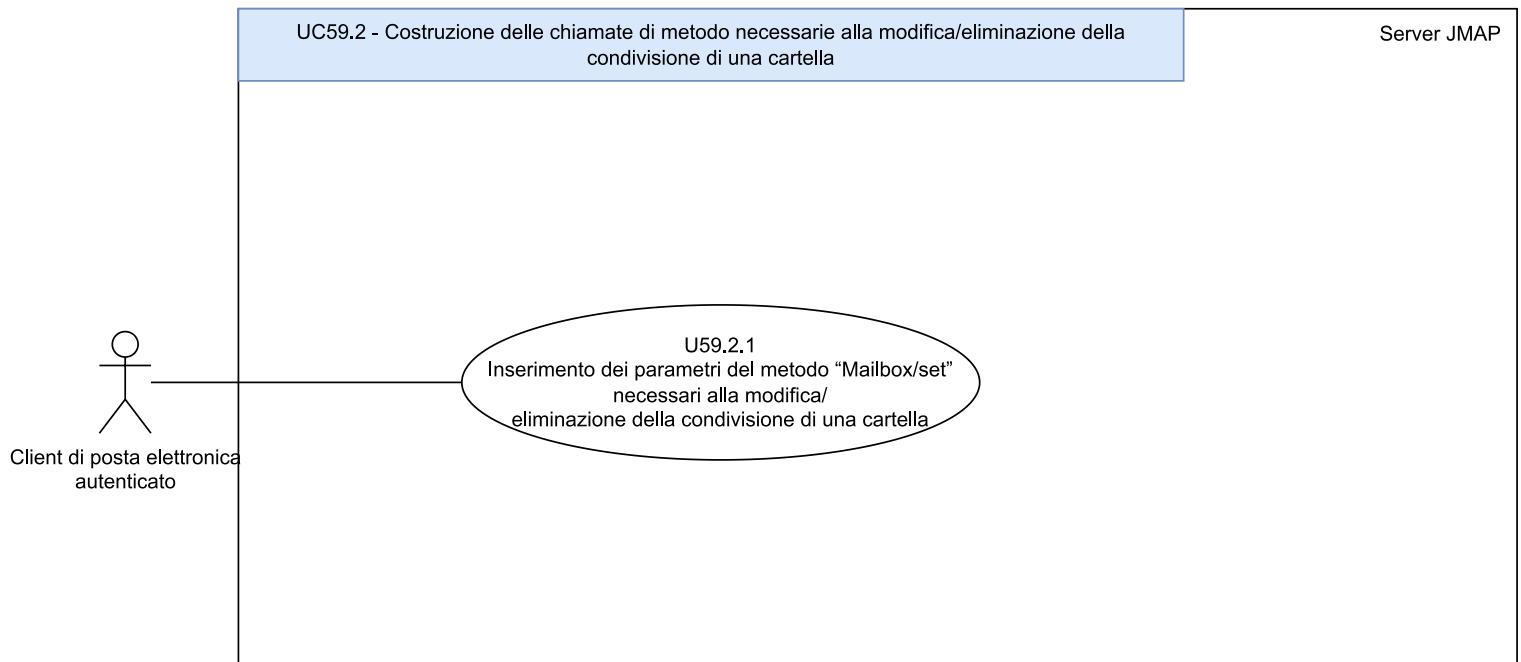


Figura 59: Sottocaso UC59.2 - Costruzione delle chiamate di metodo necessarie alla modifica/eliminazione della condivisione di una cartella

#### 3.5.59.2.1) UC59.2.1 - Inserimento dei parametri del metodo “Mailbox/set” necessari alla modifica/eliminazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare la condivisione di una cartella (compresa l'eliminazione di quest'ultima), esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "Mailbox/set", necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/set", contenuto nell'array "methodCalls" della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/set", la quale è contenuta nell'array di chiamate di metodo denominato "methodCalls", contenuto a sua volta all'interno dell'oggetto Request;

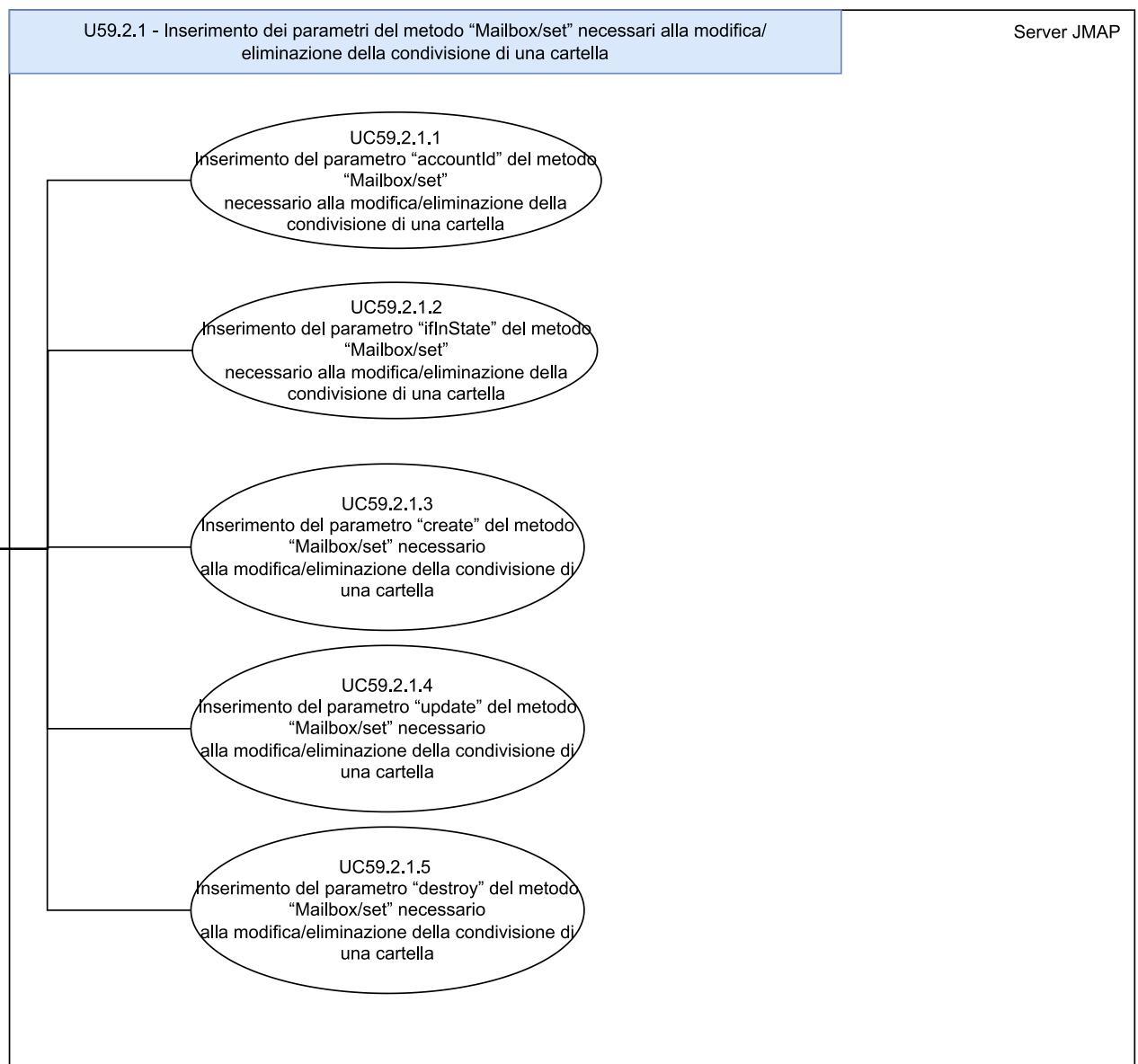


Figura 60: Sottocasi UC59.2.1 - Inserimento dei parametri del metodo "Mailbox/set" necessari alla modifica/eliminazione della condivisione di una cartella

### 3.5.59.2.1.1) UC59.2.1.1 - Inserimento del parametro “accountId” del metodo “Mailbox/set” necessario alla modifica/eliminazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare la condivisione di una cartella (compresa l’eliminazione di quest’ultima), esegue una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Mailbox/set”, necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “accountId”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “accountId”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “accountId”, il quale rappresenta l’identificativo dell’account da utilizzare;

### 3.5.59.2.1.2) UC59.2.1.2 - Inserimento del parametro “ifInState” del metodo “Mailbox/set” necessario alla modifica/eliminazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare la condivisione di una cartella (compresa l’eliminazione di quest’ultima), esegue una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Mailbox/set”, necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “ifInState”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all’interno dell’oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell’array “methodCalls” della richiesta, il parametro “ifInState”;
- **Scenario principale:**
  1. Il client specifica all’interno dell’oggetto di argomenti della chiamata di metodo il parametro “ifInState”, il quale è un meccanismo di controllo che impedisce la sovrascrittura di dati obsoleti o incoerenti sul server. Esso garantisce che la richiesta venga eseguita solo se il client è a conoscenza dello stato attuale degli oggetti e che tale stato sia valido al momento della richiesta;

### 3.5.59.2.1.3) UC59.2.1.3 - Inserimento del parametro “create” del metodo “Mailbox/set” necessario alla modifica/eliminazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare la condivisione di una cartella (compresa l’eliminazione di quest’ultima), esegue una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Mailbox/set”, necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “create”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;

- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “create”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “create”, il quale viene impostato a null, dato che nella modifica della proprietà di condivisione di una cartella nessun oggetto deve essere creato;

#### 3.5.59.2.1.4) UC59.2.1.4 - Inserimento del parametro “update” del metodo “Mailbox/set” necessario alla modifica/eliminazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare la condivisione di una cartella (compresa l'eliminazione di quest'ultima), esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo “Mailbox/set”, necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è “update”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Mailbox/set”, contenuto nell'array “methodCalls” della richiesta, il parametro “update”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “update”, il quale è utilizzato per specificare l'intenzione di modificare una cartella per cambiare la proprietà di condivisione di quest'ultima. Questo parametro è una mappa che associa un identificativo a un oggetto di tipo patch (PatchObject) da applicare all'oggetto Mailbox corrente con quell'identificativo. Un PatchObject rappresenta un insieme non ordinato di modifiche;

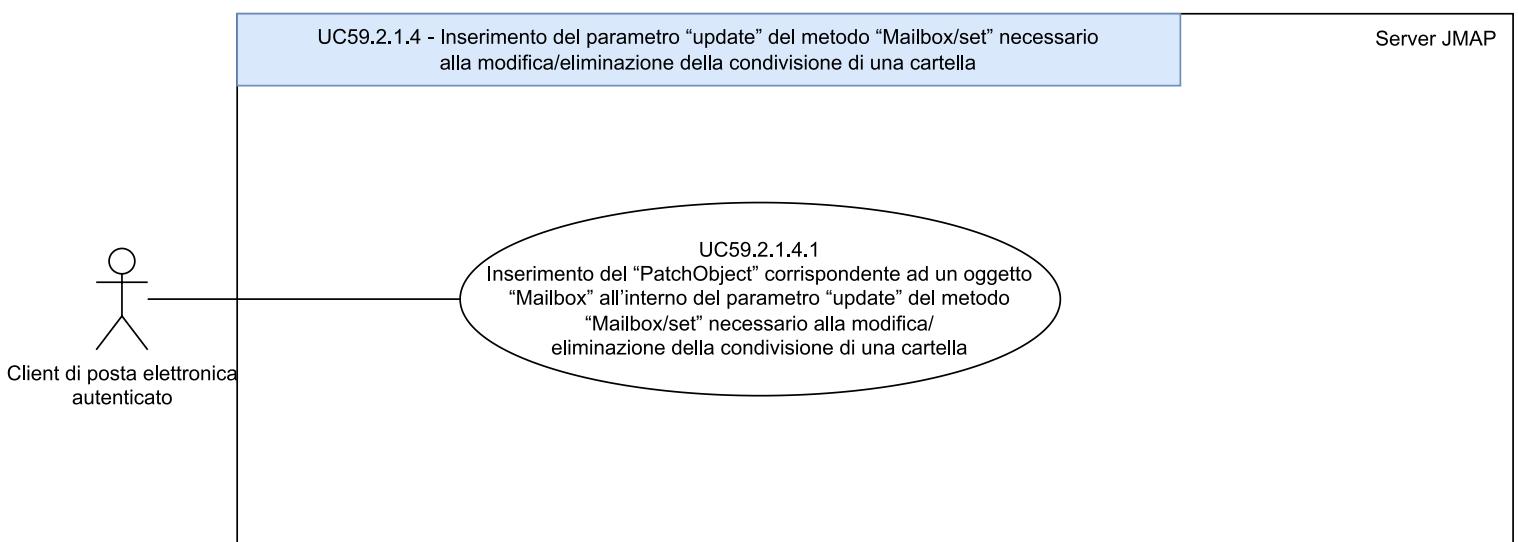


Figura 61: Sottocaso UC59.2.1.4 - Inserimento del parametro “update” del metodo “Mailbox/set” necessario alla modifica/eliminazione della condivisione di una cartella

#### 3.5.59.2.1.4.1) UC59.2.1.4.1 - Inserimento del “PatchObject” corrispondente ad un oggetto “Mailbox” all’interno del parametro “update” del metodo “Mailbox/set” necessario alla modifica/eliminazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;

- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare la condivisione di una cartella (compresa l'eliminazione di quest'ultima), esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "Mailbox/set", necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è "update". All'interno di questo parametro va definito un "PatchObject", il quale rappresenta un insieme non ordinato di modifiche;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito il "PatchObject" di un oggetto "Mailbox" all'interno del parametro "update", contenuto a sua volta all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/set", contenuto nell'array "methodCalls" della richiesta;
- **Scenario principale:**
  1. Il client specifica all'interno del parametro "update", contenuto nell'oggetto di argomenti della chiamata di metodo, l'oggetto "PatchObject" necessario per la modifica della proprietà di condivisione di una cartella. Questo è un array di stringhe, il quale rappresenta un insieme non ordinato di patch, dove le chiavi sono un percorso nel formato JSON Pointer, con un "/" posto all'inizio implicitamente, che stanno ad indicare una particolare proprietà. In questo caso è interessata solamente la proprietà "shareWith", la quale serve a rappresentare che Principal hanno accesso alla cartella e quali diritti hanno. Essa è una mappa in cui la chiave è l'identificativo del principale con cui si sta condividendo e il valore è un oggetto che rappresenta i diritti concessi a quel principale. Se la richiesta è di eliminazione allora il client provvederà a rimuovere dalla mappa i principali che devono essere rimossi dalla condivisione. Altrimenti il client specificherà le modifiche ai diritti o ai principali che desidera apportare.  
Tutti i percorsi utilizzati come chiave devono essere conformi a specifiche restrizioni, altrimenti la modifica verrà rifiutata rispondendo con un errore "invalidPatch".  
Per quanto riguarda il valore associato a ciascun percorso:
    - se è nullo allora imposta il valore predefinito, se specificato, per la proprietà corrispondente;  
se non specificato, rimuove la proprietà dalla cartella modificata. Se la chiave non è presente nel padre, questa è un'operazione senza effetto;
    - se non è nullo allora specifica il valore da impostare per la proprietà corrispondente (questa può essere una sostituzione o un'aggiunta alla cartella che viene modificata).Eventuali proprietà impostate dal server possono essere incluse nella modifica se il loro valore è identico al valore corrente del server (prima di applicare le modifiche alla cartella). In caso contrario, la modifica DEVE essere respinta con un errore di tipo "invalidProperties".  
Questa definizione consente di apportare modifiche sia inviando l'intero nuovo oggetto "Mailbox" come patch, che inviando solo le differenze. Entrambi i metodi sono validi, e il server elabora le modifiche allo stesso modo in entrambi i casi;

### 3.5.59.2.1.5 - Inserimento del parametro "destroy" del metodo "Mailbox/set" necessario alla modifica/eliminazione della condivisione di una cartella

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di modificare la condivisione di una cartella (compresa l'eliminazione di quest'ultima), esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il metodo "Mailbox/set", necessario per la modifica della proprietà di condivisione di una cartella, per il quale vanno specificati i parametri necessari. Uno di questi è "destroy";

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/set", contenuto nell'array "methodCalls" della richiesta, il parametro "destroy";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "destroy", il quale viene impostato a null, dato che nella modifica della proprietà di condivisione di una cartella nessun oggetto deve essere distrutto;

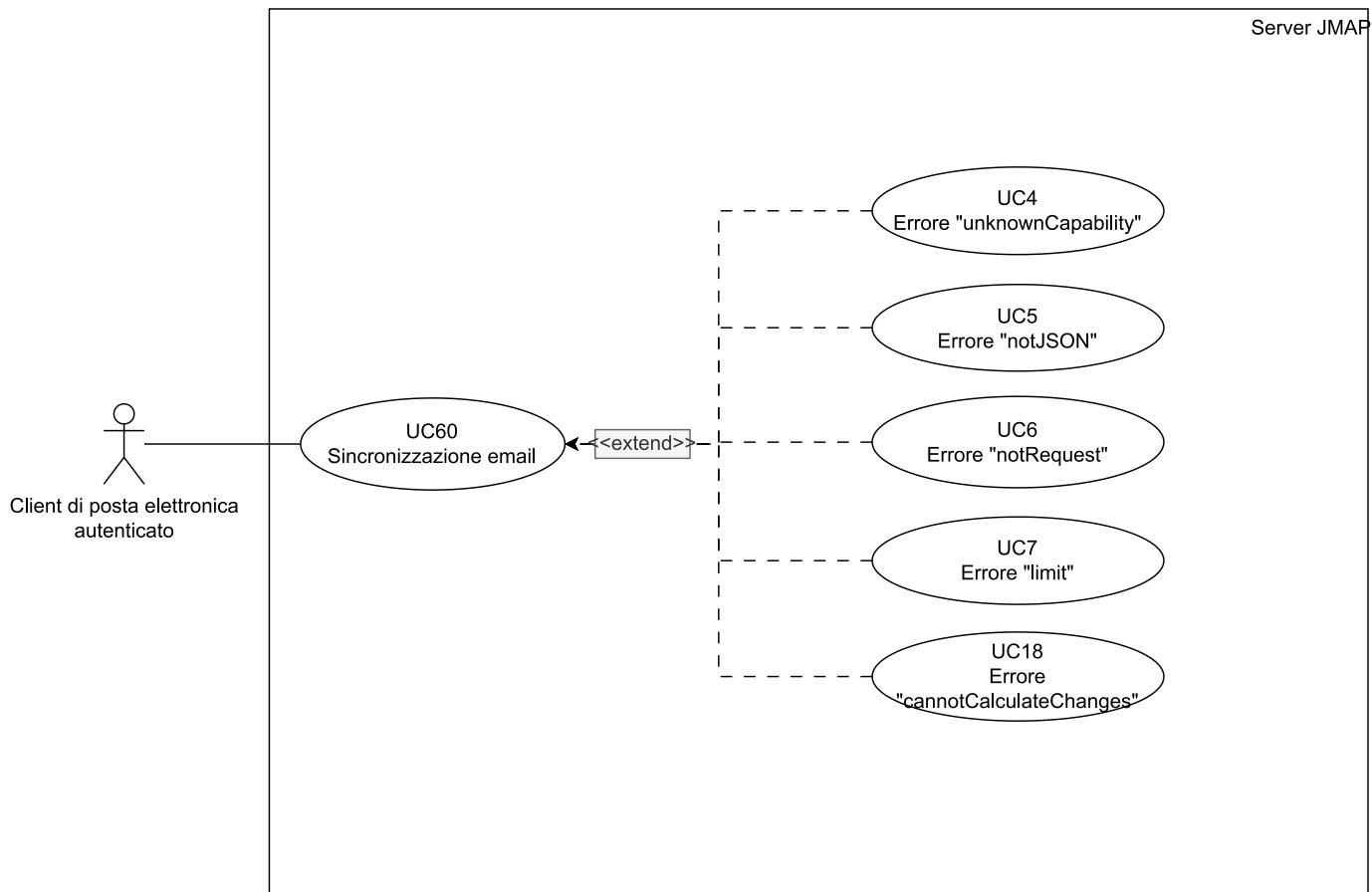


Figura 62: UC60 - Sincronizzazione email

### 3.5.60) UC60 - Sincronizzazione email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le email, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client è riuscito a sincronizzarsi con gli ultimi aggiornamenti per quanto riguarda le email ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per sincronizzarsi con gli ultimi aggiornamenti per quanto riguarda le email;

2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
3. Il server elabora la richiesta API e invia una risposta JSON che contiene le informazioni necessarie al client per sincronizzarsi;

- **Estensioni:**

- Errore “unknownCapability”;
- Errore “notJSON”;
- Errore “notRequest”;
- Errore “limit”;
- Errore “cannotCalculateChanges”;

- **Inclusioni:** /

- **Generalizzazioni:** /

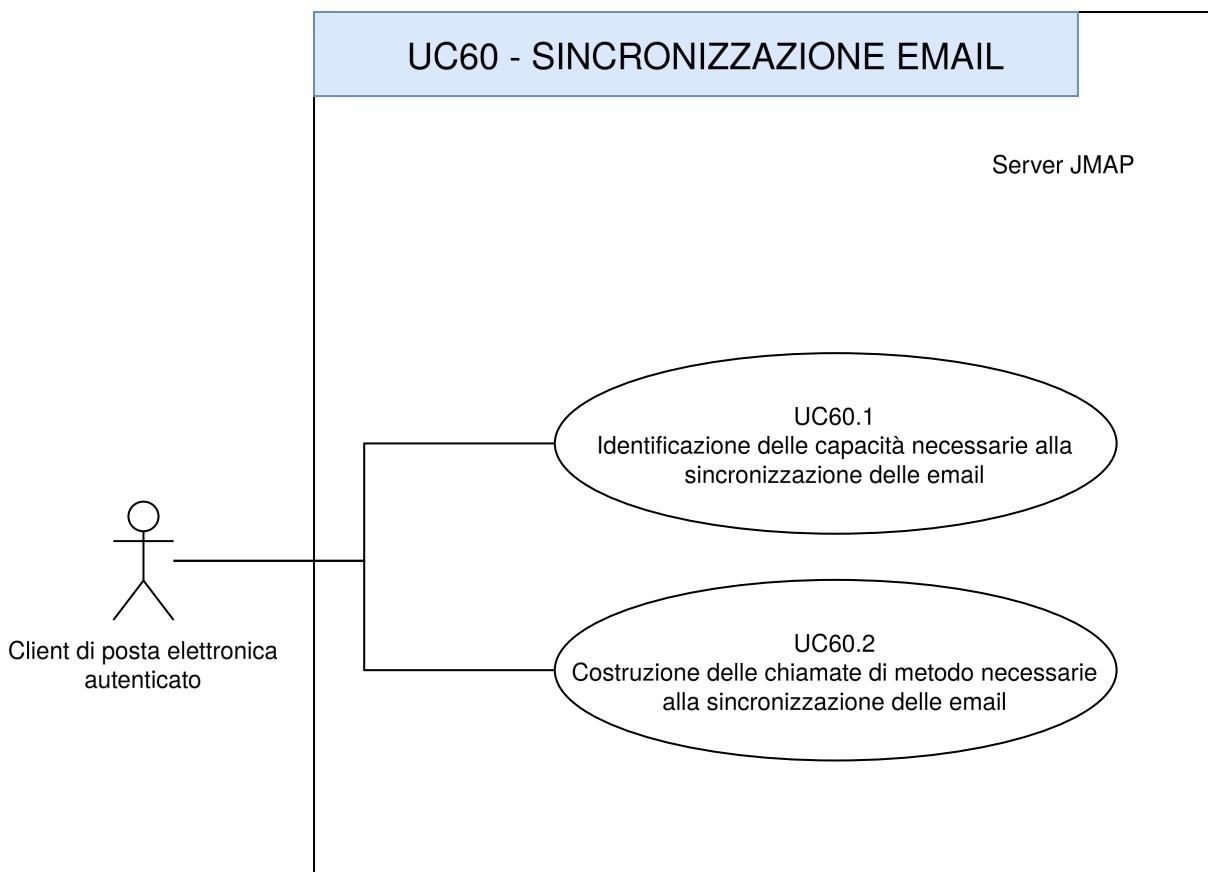


Figura 63: Sottocasi UC60 - Sincronizzazione email

#### 3.5.60.1 UC60.1 - Identificazione delle capacità necessarie alla sincronizzazione delle email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le email, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per la sincronizzazione delle email;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all'interno della richiesta le capacità JMAP necessarie per la sincronizzazione delle email;
- **Scenario principale:**

- Il client include nella sezione “using” dell’oggetto Request le capacità JMAP “urn:ietf:params:jmap:core” e “urn:ietf:params:jmap:mail”;

### 3.5.60.2) UC60.2 - Costruzione delle chiamate di metodo necessarie alla sincronizzazione delle email

- Attore principale:** Client di posta elettronica autenticato;
- Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le email, esegue una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per la sincronizzazione delle email;
- Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- Postcondizioni:** Il client ha inserito all’interno dell’array “methodCalls” della richiesta le chiamate di metodo necessarie per la sincronizzazione delle email;
- Scenario principale:**
  - Il client crea un array di chiamate di metodo denominato “methodCalls” internamente all’oggetto Request, all’interno del quale inserisce il metodo “Email/changes”, un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

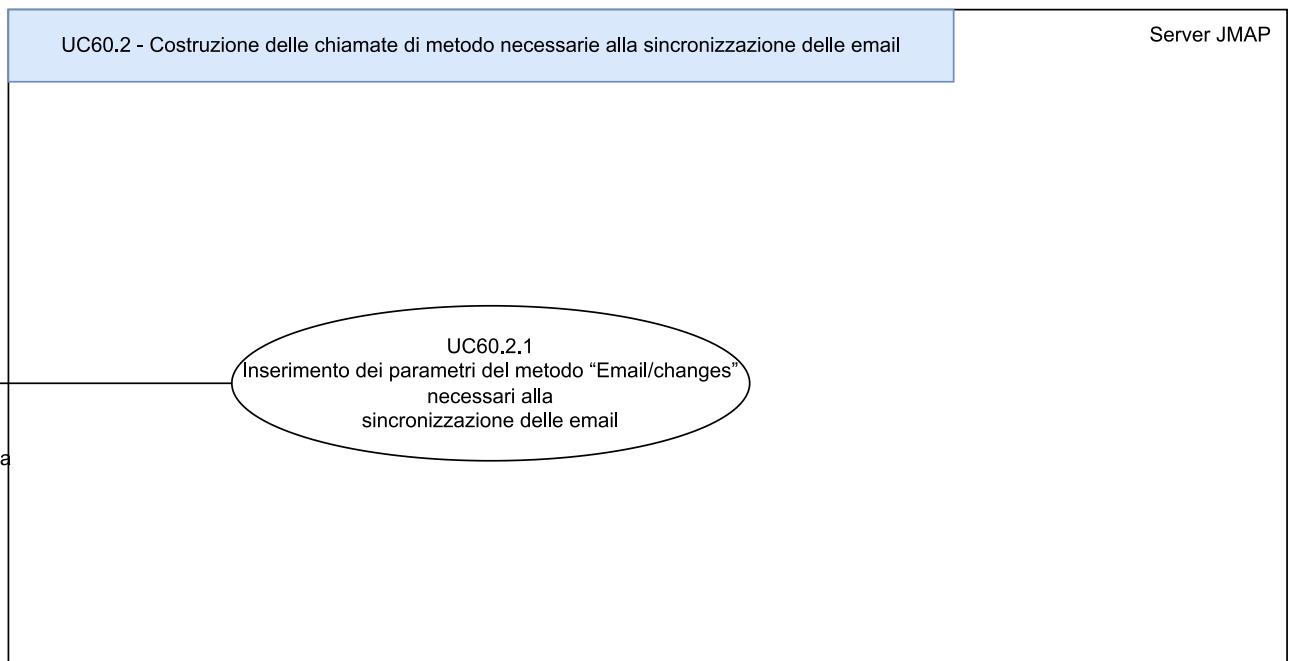


Figura 64: Sottocasi UC60.2 - Costruzione delle chiamate di metodo necessarie alla sincronizzazione delle email

#### 3.5.60.2.1) UC60.2.1 - Inserimento dei parametri del metodo “Email/changes” necessari alla sincronizzazione delle email

- Attore principale:** Client di posta elettronica autenticato;
- Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le email, esegue una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/changes”, necessario per la sincronizzazione delle email, per il quale vanno specificati i parametri necessari;
- Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;

- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Email/changes”, contenuto nell'array “methodCalls” della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all'interno dell'oggetto di argomenti della chiamata di metodo “Email/changes”, la quale è contenuta nell'array di chiamate di metodo denominato “methodCalls”, contenuto a sua volta all'interno dell'oggetto Request;

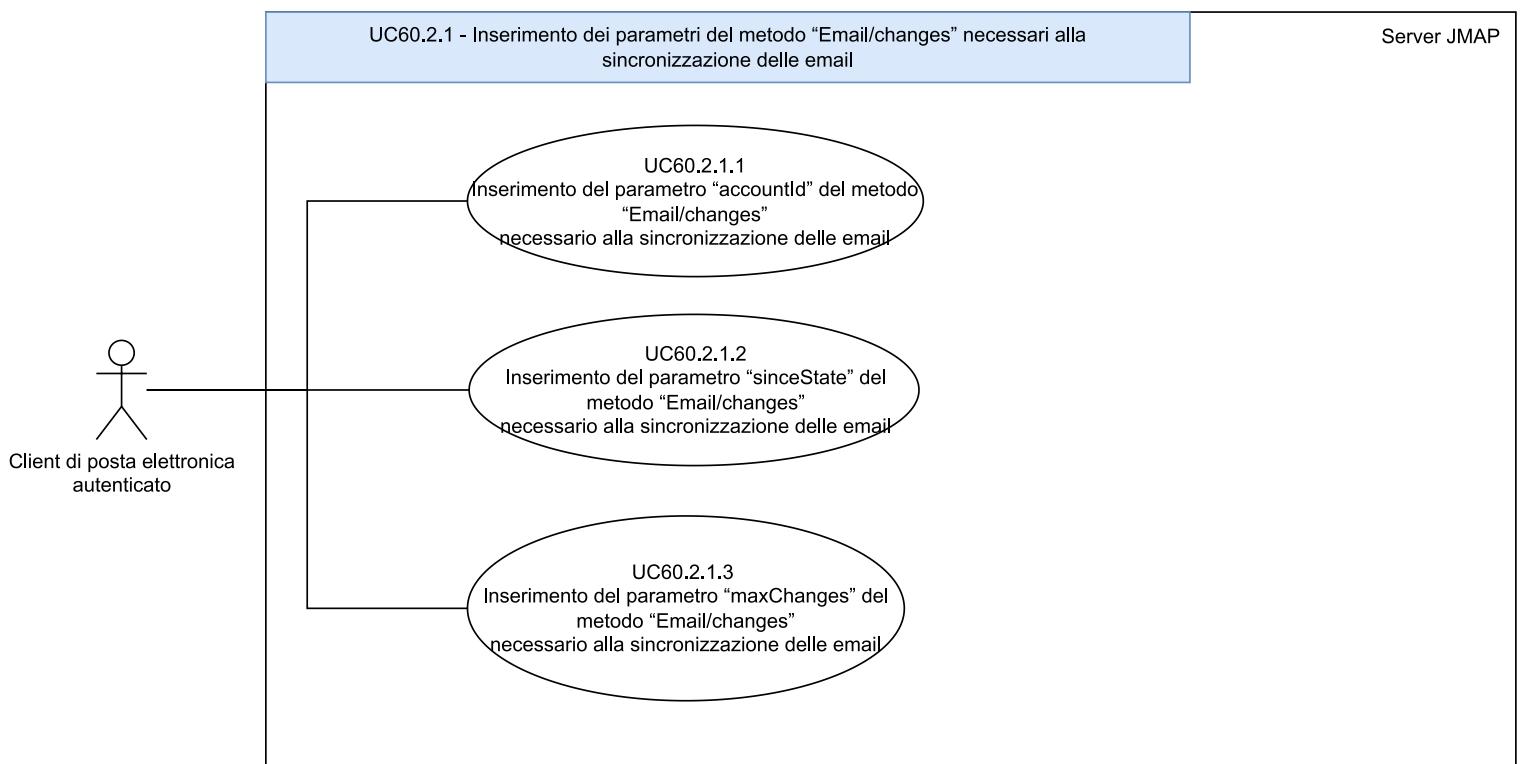


Figura 65: Sottocasi UC60.2.1 - Inserimento dei parametri del metodo “Email/changes” necessari alla sincronizzazione delle email

### 3.5.60.2.1.1) UC60.2.1.1 - Inserimento del parametro “accountId” del metodo “Email/changes” necessario alla sincronizzazione delle email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le email, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo “Email/changes”, necessario per la sincronizzazione delle email, per il quale vanno specificati i parametri necessari. Uno di questi è “accountId”;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo “Email/changes”, contenuto nell'array “methodCalls” della richiesta, il parametro “accountId”;
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro “accountId”, il quale rappresenta l'identificativo dell'account da utilizzare;

### 3.5.60.2.1.2) UC60.2.1.2 - Inserimento del parametro “sinceState” del metodo “Email/changes” necessario alla sincronizzazione delle email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le email, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Email/changes", necessario per la sincronizzazione delle email, per il quale vanno specificati i parametri necessari. Uno di questi è "sinceState";
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Email/changes", contenuto nell'array "methodCalls" della richiesta, il parametro "sinceState";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "sinceState", il quale rappresenta lo stato corrente del client. In pratica, è la stringa che è stata restituita come argomento state nella risposta alle chiamate precedenti del metodo Email/get. Questo stato è ciò che il client ha memorizzato come suo stato corrente;

### 3.5.60.2.1.3) UC60.2.1.3 - Inserimento del parametro "maxChanges" del metodo "Email/changes" necessario alla sincronizzazione delle email

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le email, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Email/changes", necessario per la sincronizzazione delle email, per il quale vanno specificati i parametri necessari. Uno di questi è "maxChanges";
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Email/changes", contenuto nell'array "methodCalls" della richiesta, il parametro "maxChanges";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "maxChanges", il quale rappresenta il numero massimo di identificatori che il client desidera ricevere come risposta. Il server ha la facoltà di restituire un numero inferiore a questo valore, ma non deve superarlo. Se il client non fornisce questo parametro, il server può decidere autonomamente quanti identificatori restituire. Se il client lo specifica, il valore deve essere un numero intero positivo maggiore di zero. Se viene fornito un valore al di fuori di questo intervallo, il server deve respingere la chiamata con un errore di tipo invalidArguments;

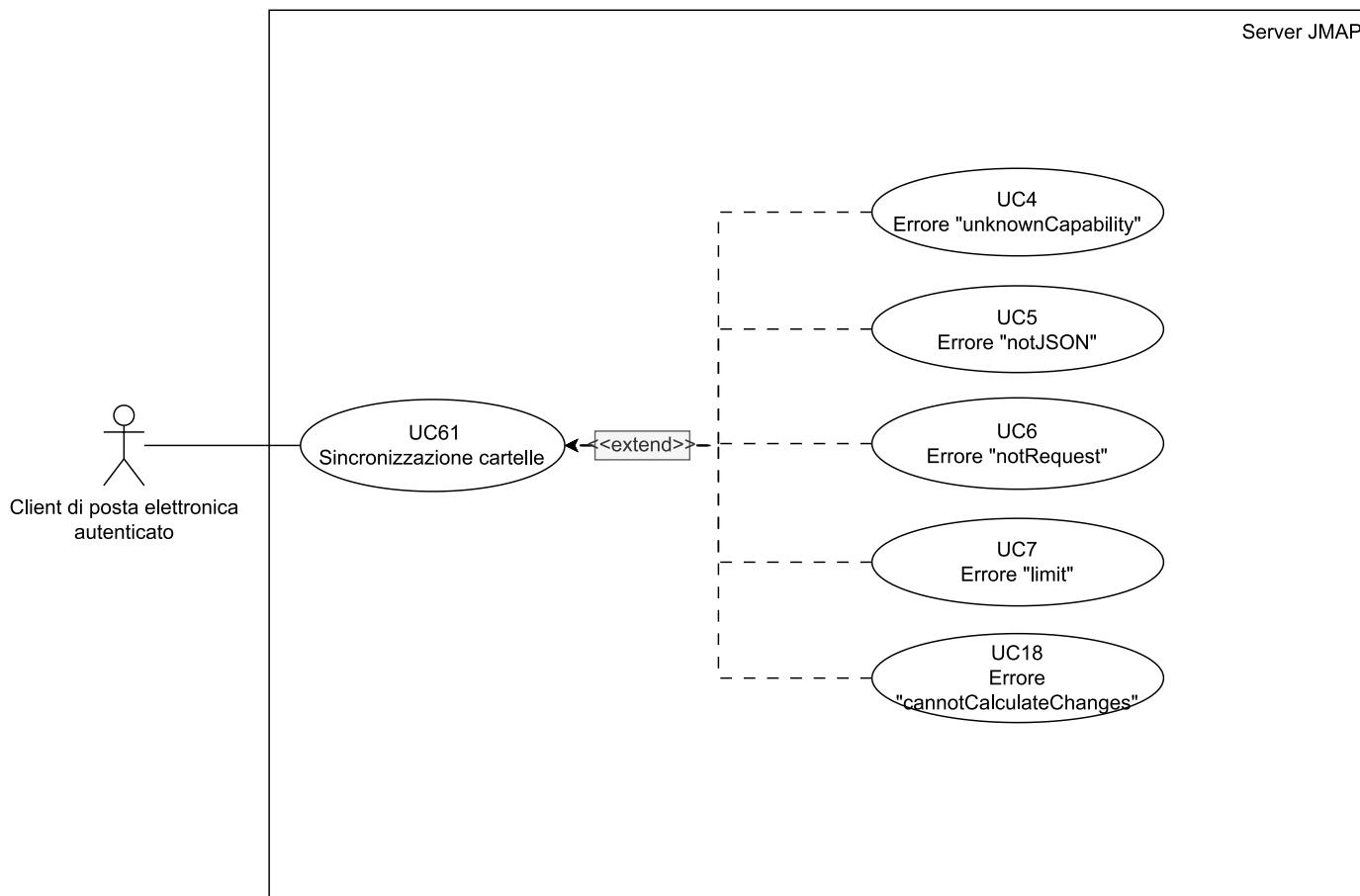


Figura 66: UC61 - Sincronizzazione cartelle

### 3.5.61) UC61 - Sincronizzazione cartelle

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le cartelle, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client è riuscito a sincronizzarsi con gli ultimi aggiornamenti per quanto riguarda le cartelle ed il sistema è nello stato desiderato. Altrimenti il server ha restituito una risposta che indica il motivo del fallimento;
- **Scenario principale:**
  1. Il client prepara la richiesta contenente le chiamate di metodo necessarie per sincronizzarsi con gli ultimi aggiornamenti per quanto riguarda le cartelle;
  2. Il client esegue una richiesta POST autenticata all'URL dell'API, inviando l'oggetto Request JSON con le chiamate di metodo;
  3. Il server elabora la richiesta API e invia una risposta JSON che contiene le informazioni necessarie al client per sincronizzarsi;
- **Estensioni:**
  - Errore "unknownCapability";
  - Errore "notJSON";
  - Errore "notRequest";
  - Errore "limit";

- Errore “cannotCalculateChanges”;
- **Inclusioni:** /
- **Generalizzazioni:** /

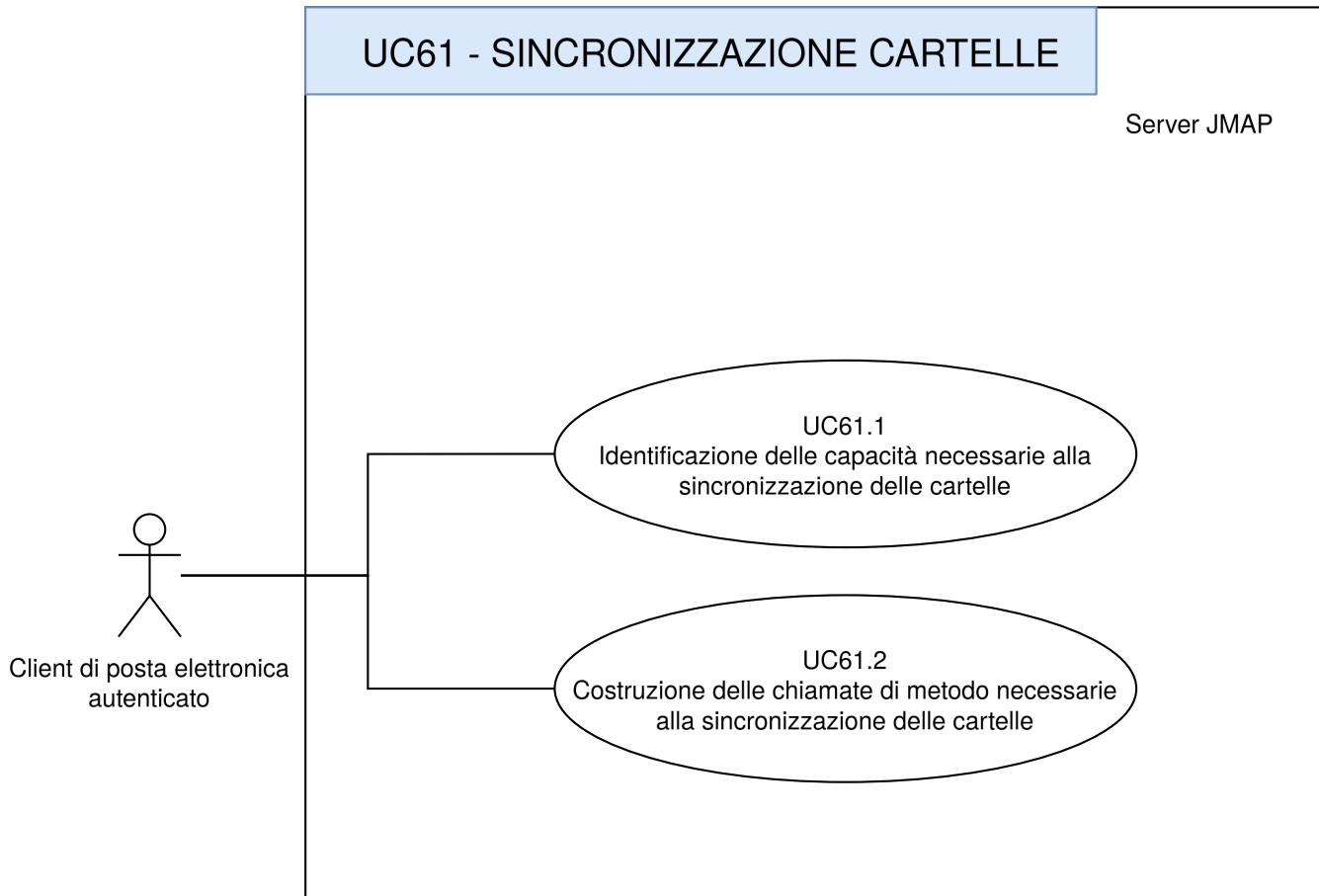


Figura 67: Sottocasi UC61 - Sincronizzazione cartelle

### 3.5.61.1) UC61.1 - Identificazione delle capacità necessarie alla sincronizzazione delle cartelle

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le cartelle, esegue una richiesta POST autenticata all’URL dell’API, il quale è definito all’interno della risorsa JMAP Session. Internamente a questa richiesta il client identifica le capacità JMAP necessarie per la sincronizzazione delle cartelle;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l’oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha identificato all’interno della richiesta le capacità JMAP necessarie per la sincronizzazione delle cartelle;
- **Scenario principale:**
  1. Il client include nella sezione “using” dell’oggetto Request le capacità JMAP “urn:ietf:params:jmap:core” e “urn:ietf:params:jmap:mail”;

### 3.5.61.2) UC61.2 - Costruzione delle chiamate di metodo necessarie alla sincronizzazione delle cartelle

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le cartelle, esegue una richiesta POST autenticata all’URL dell’API,

l'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo necessarie per la sincronizzazione delle cartelle;

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'array "methodCalls" della richiesta le chiamate di metodo necessarie per la sincronizzazione delle cartelle;
- **Scenario principale:**
  1. Il client crea un array di chiamate di metodo denominato "methodCalls" internamente all'oggetto Request, all'interno del quale inserisce il metodo "Mailbox/changes", un oggetto contenente i parametri del metodo e un identificatore univoco associato a quella chiamata di metodo;

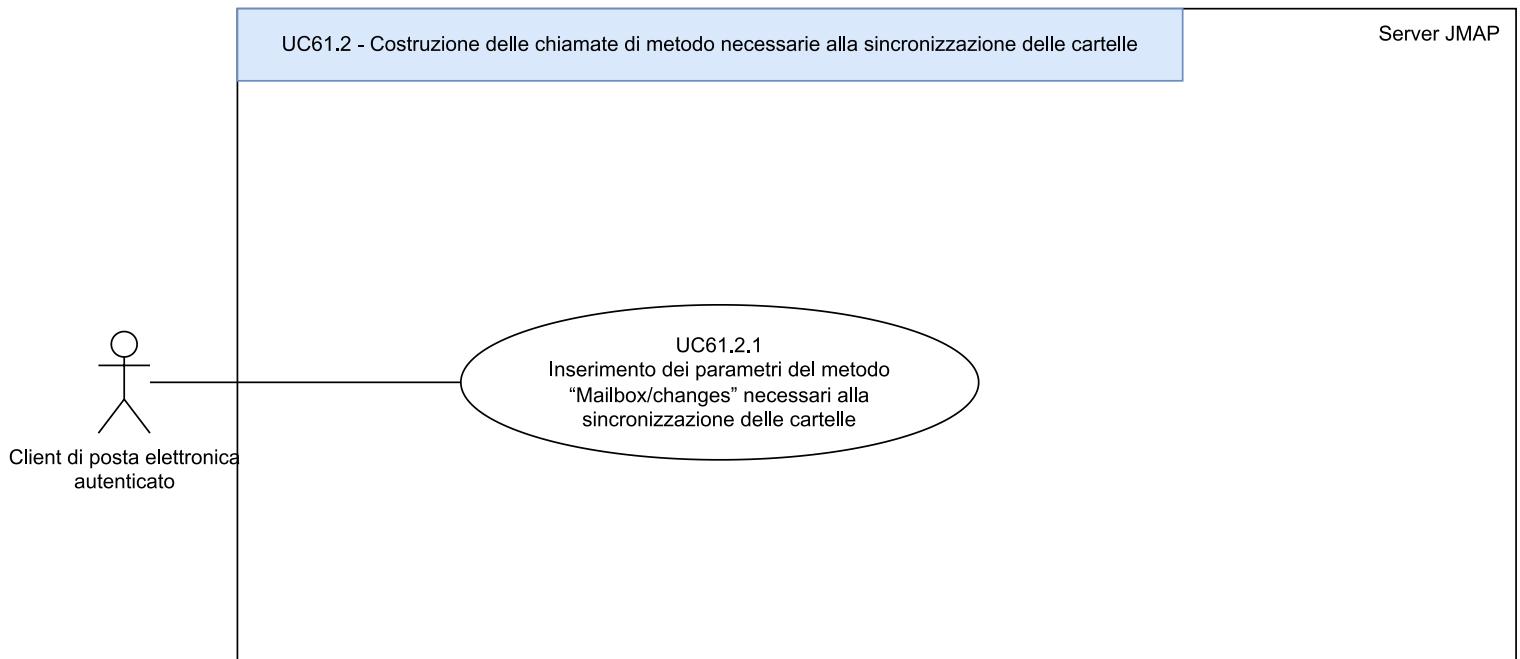


Figura 68: Sottocasi UC61.2 - Costruzione delle chiamate di metodo necessarie alla sincronizzazione delle cartelle

### 3.5.61.2.1) UC60.2.1 - Inserimento dei parametri del metodo "Mailbox/changes" necessari alla sincronizzazione delle cartelle

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le cartelle, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Mailbox/changes", necessario per la sincronizzazione delle cartelle, per il quale vanno specificati i parametri necessari;
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/changes", contenuto nell'array "methodCalls" della richiesta, i parametri necessari;
- **Scenario principale:**
  1. Il client specifica gli argomenti all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/changes", la quale è contenuta nell'array di chiamate di metodo denominato "methodCalls", contenuto a sua volta all'interno dell'oggetto Request;

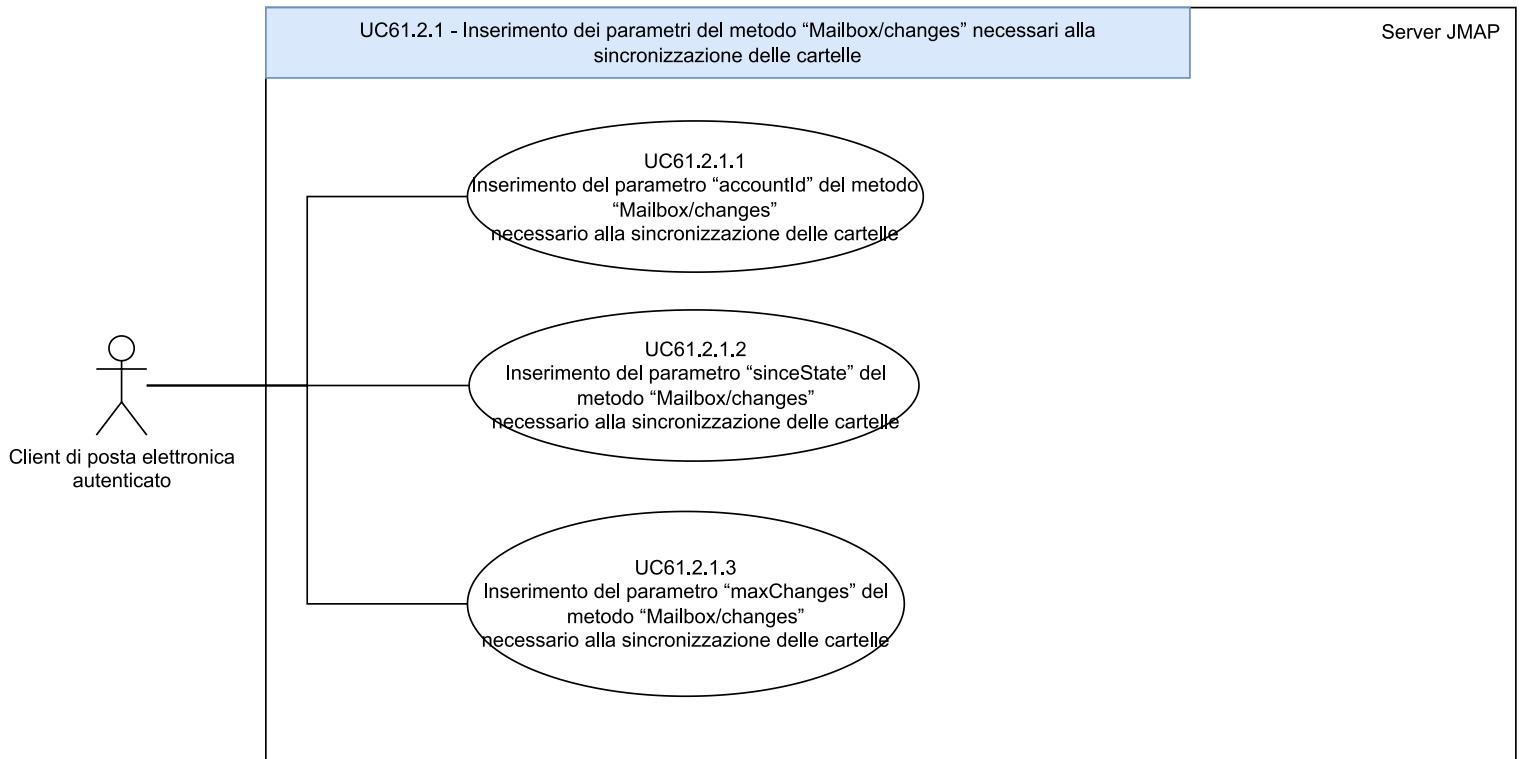


Figura 69: Sottocasi UC60.2.1 - Inserimento dei parametri del metodo "Mailbox/changes" necessari alla sincronizzazione delle cartelle

### 3.5.61.2.1.1) UC61.2.1.1 - Inserimento del parametro "accountId" del metodo "Mailbox/changes" necessario alla sincronizzazione delle cartelle

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le cartelle, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Mailbox/changes", necessario per la sincronizzazione delle cartelle, per il quale vanno specificati i parametri necessari. Uno di questi è "accountId";
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/changes", contenuto nell'array "methodCalls" della richiesta, il parametro "accountId";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "accountId", il quale rappresenta l'identificativo dell'account da utilizzare;

### 3.5.61.2.1.2) UC61.2.1.2 - Inserimento del parametro "sinceState" del metodo "Mailbox/changes" necessario alla sincronizzazione delle cartelle

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le cartelle, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Mailbox/changes", necessario per la sincronizzazione delle cartelle, per il quale vanno specificati i parametri necessari. Uno di questi è "sinceState";

- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/changes", contenuto nell'array "methodCalls" della richiesta, il parametro "sinceState";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "sinceState", il quale rappresenta lo stato corrente del client. In pratica, è la stringa che è stata restituita come argomento state nella risposta alle chiamate precedenti del metodo Mailbox/get. Questo stato è ciò che il client ha memorizzato come suo stato corrente;

### 3.5.61.2.1.3) UC61.2.1.3 - Inserimento del parametro "maxChanges" del metodo "Mailbox/changes" necessario alla sincronizzazione delle cartelle

- **Attore principale:** Client di posta elettronica autenticato;
- **Descrizione:** Un client di posta elettronica, con lo scopo di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le cartelle, esegue una richiesta POST autenticata all'URL dell'API, il quale è definito all'interno della risorsa JMAP Session. Internamente a questa richiesta il client crea un array di chiamate di metodo contenente il solo metodo "Mailbox/changes", necessario per la sincronizzazione delle cartelle, per il quale vanno specificati i parametri necessari. Uno di questi è "maxChanges";
- **Precondizioni:** Il client è riuscito ad autenticarsi ed ha ottenuto l'oggetto JSON che rappresenta la sessione;
- **Postcondizioni:** Il client ha inserito all'interno dell'oggetto di argomenti della chiamata di metodo "Mailbox/changes", contenuto nell'array "methodCalls" della richiesta, il parametro "maxChanges";
- **Scenario principale:**
  1. Il client specifica all'interno dell'oggetto di argomenti della chiamata di metodo il parametro "maxChanges", il quale rappresenta il numero massimo di identificatori che il client desidera ricevere come risposta. Il server ha la facoltà di restituire un numero inferiore a questo valore, ma non deve superarlo. Se il client non fornisce questo parametro, il server può decidere autonomamente quanti identificatori restituire. Se il client lo specifica, il valore deve essere un numero intero positivo maggiore di zero. Se viene fornito un valore al di fuori di questo intervallo, il server deve respingere la chiamata con un errore di tipo invalidArguments;

## 4) Requisiti

In questa sezione andremo ad elencare tutti i requisiti del capitolo, raggruppandoli per tipologia. Ogni requisito è identificato univocamente da un codice il cui formato è definito qui di seguito:

**R-[numero]-[tipo]-[priorità]**

dove:

- **Numero:** un numero progressivo a tre cifre che parte dal numero 001.
- **Tipo:** la tipologia di requisito tra le seguenti:
  - **F (funzionale):** indica una funzione del sistema, definendone la tipologia degli ingressi e delle uscite, nonché il comportamento.
  - **Q (qualità):** definisce le caratteristiche di qualità del prodotto software.
  - **V (vincolo):** specifica i limiti e le restrizioni imposte dal capitolo, che il nostro sistema deve rispettare.
- **Priorità:** il grado di priorità assegnato al requisito con un numero da 1 a 3:
  1. Requisito obbligatorio che deve essere tassativamente soddisfatto dalla soluzione informatica proposta.
  2. Requisito desiderabile il cui soddisfacimento è apprezzato dal committente.
  3. Requisito facoltativo, la cui realizzazione è totalmente a discrezione del team in base all'andamento del progetto.

In alcuni casi verrà esplicitato nella colonna relativa alle fonti se il requisito è stato chiaramente dettato dal Capitolo o se è stato dedotto implicitamente da altri requisiti obbligatori (in questo caso parleremo di requisito interno).

### 4.1) Requisiti di funzionalità

ID Requisito	Descrizione	Fonti
	AUTENTICAZIONE	
R-001-F-2	L'utente che utilizza un client di posta elettronica per interagire con il server deve autenticarsi all'interno del sistema.	UC1 Interno
R-002-F-2	È necessario che il client fornisca all'interno della richiesta l'indirizzo email personale dell'utente per procedere con l'autenticazione.	UC1.1 Interno
R-003-F-2	È necessario che il client fornisca all'interno della richiesta la password associata all'indirizzo email personale dell'utente per procedere con l'autenticazione.	UC1.2 Interno
R-004-F-2	Se la fase di autenticazione è fallita allora è necessario che il client riceva dal server una risposta con eventuali dettagli che ne indicano il motivo.	UC2 Interno
R-005-F-1	Il client deve essere in grado di reperire la risorsa JMAP Session, contenente informazioni sulle capacità del server, dettagli sull'account dell'utente e le URL per le richieste API future, in modo da poter interagire con dati e servizi offerti dal server.	UC3 Interno

	ERRORI	
R-006-F-1	È necessario che il client riceva in risposta l'errore “unkownCapability” in caso di esecuzione di una richiesta con proprietà “using” non supportata dal server.	UC4 Interno
R-007-F-1	È necessario che il client riceva in risposta l'errore “notJSON” se il contenuto di una richiesta inviata al server non era application/json o se la richiesta non è stata interpretata dal server come I-JSON.	UC5 Interno
R-008-F-1	È necessario che il client riceva in risposta l'errore “notRequest” se una richiesta JSON non ha corrisposto alla firma di tipo dell'oggetto di richiesta (Request).	UC6 Interno
R-009-F-1	È necessario che il client riceva in risposta l'errore “limit” in caso di inserimento di una richiesta che supera uno dei limiti definiti sull'oggetto di capacità, come maxSizeRequest, maxCallsInRequest o maxCurrentRequests.	UC7 Interno
R-010-F-1	È necessario che il client riceva in risposta l'errore “serverUnavailable” in caso di inserimento di una richiesta che necessita di alcune risorse interne del server momentaneamente non disponibili.	UC8 Interno
R-011-F-1	È necessario che il client riceva in risposta l'errore “serverFail” in caso si verifichi un errore inaspettato o sconosciuto durante l'elaborazione di una sua richiesta dal server.	UC9 Interno
R-012-F-1	È necessario che il client riceva in risposta l'errore “serverPartialFail” e proceda risincronizzando i dati in caso si verifichi un errore inaspettato o sconosciuto durante l'elaborazione di una sua richiesta dal server.	UC10 Interno
R-013-F-1	È necessario che il client riceva in risposta l'errore “unkownMethod” in caso di inserimento di una richiesta contenente un metodo non riconosciuto dal server.	UC11 Interno
R-014-F-1	È necessario che il client riceva in risposta l'errore “invalidArguments” se uno degli argomenti di un metodo fornito all'interno di una richiesta al server è di tipo errato, non valido o, nel caso in cui sia obbligatorio, è assente.	UC12 Interno
R-015-F-1	È necessario che il client riceva in risposta l'errore “invalidResultReference” se uno degli argomenti di un metodo fornito all'interno di una richiesta al server ha utilizzato un riferimento di risultato che non è stato possibile risolvere da parte del server.	UC13 Interno
R-016-F-1	È necessario che il client riceva in risposta l'errore “forbidden” in caso utilizzi, all'interno di una richiesta al server, un metodo la cui esecuzione violerebbe una Access Control List (ACL) o un'altra policy di autorizzazione.	UC14 Interno

R-017-F-1	È necessario che il client riceva in risposta l'errore "accountNotFound" se l'"accountID" fornito all'interno di una richiesta al server non corrisponde a un account valido.	UC15 Interno
R-018-F-1	È necessario che il client riceva in risposta l'errore "accountNotSupportedByMethod" se all'interno di una richiesta al server è presente un metodo o tipo di dato non supportato dall'"accountID" fornito.	UC16 Interno
R-019-F-1	È necessario che il client riceva in risposta l'errore "accountReadOnly" se all'interno di una richiesta al server è presente un metodo che tenta di modificare lo stato nonostante l'account sia in sola lettura.	UC17 Interno
R-020-F-1	È necessario che il client riceva in risposta l'errore "cannotCalculateChanges" se, in seguito all'inserimento di una richiesta, il server non possa calcolare le modifiche dello stato dalla stringa di stato fornita dal client.	UC18 Interno
R-021-F-1	È necessario che il client riceva in risposta l'errore "overQuota" se una richiesta inserita nel server richiede la creazione di oggetti che per dimensione o quantità superano il limite imposto dal server.	UC19 Interno
R-022-F-1	È necessario che il client riceva in risposta l'errore "notFound" se una richiesta inserita nel server fornisce degli ID che non possono essere trovati.	UC20 Interno
R-023-F-1	È necessario che il client riceva in risposta l'errore "willDestroy" se ha richiesto che un oggetto fosse sia aggiornato che distrutto all'interno della stessa richiesta al server.	UC21 Interno
R-024-F-1	È necessario che il client riceva in risposta l'errore "tooLarge" se una richiesta inserita nel server richiede la creazione di un oggetto che supera il limite definito dal server per la dimensione massima per un oggetto di quel tipo.	UC22 Interno
R-025-F-1	È necessario che il client riceva in risposta l'errore "rateLimit" se una richiesta inserita nel server comporta la creazione di un oggetto per cui sono stati creati troppi oggetti quel tipo di recente, raggiungendo un limite di frequenza definito dal server.	UC23 Interno
R-026-F-1	È necessario che il client riceva in risposta l'errore "invalidPatch" se una richiesta inserita nel server fornisce un PatchObject non valido per modificare il record.	UC24 Interno
R-027-F-1	È necessario che il client riceva in risposta l'errore "invalidProperties" se una richiesta inserita nel server fornisce un record non valido.	UC25 Interno
R-028-F-1	È necessario che il client riceva in risposta l'errore "singleton" se una richiesta inserita nel server tentasse di agire erroneamente su un tipo singleton.	UC26 Interno

R-029-F-1	È necessario che il client riceva in risposta l'errore “requestTooLarge” se una richiesta inserita nel server contiene un numero di azioni che supera il massimo che il server è disposto a elaborare in una singola chiamata di metodo interna alla richiesta.	UC27 Interno
R-030-F-1	È necessario che il client riceva in risposta l'errore “stateMismatch” se una richiesta inserita nel server contiene un argomento ifInState e questo non corrisponde allo stato attuale.	UC28 Interno
R-031-F-1	È necessario che il client riceva in risposta l'errore “blobNotFound” se una richiesta inserita nel server contiene almeno un ID blob fornito per una parte del corpo dell'email che non esiste.	UC29 Interno
R-032-F-1	È necessario che il client riceva in risposta l'errore “tooManyKeywords” se una richiesta inserita nel server modifica un numero di parole chiave dell'email superiore al limite massimo definito dal server.	UC30 Interno
R-033-F-1	È necessario che il client riceva in risposta l'errore “tooManyMailboxes” se una richiesta inserita nel server modifica un numero di cartelle a cui appartiene l'email superiore al limite massimo definito dal server.	UC31 Interno
R-034-F-1	È necessario che il client riceva in risposta l'errore “alreadyExists” se una richiesta inserita in un server che vieta i duplicati contiene un record già esistente nell'account di destinazione.	UC32 Interno
R-035-F-1	È necessario che il client riceva in risposta l'errore “fromAccountNotFound” se una richiesta inserita nel server contiene un fromAccountId che non corrisponde a nessun account valido.	UC33 Interno
R-036-F-1	È necessario che il client riceva in risposta l'errore “fromAccountNotSupportedByMethod” se una richiesta inserita nel server contiene un fromAccountId che non supporta un tipo di dato utilizzato.	UC34 Interno
R-037-F-1	È necessario che il client riceva in risposta l'errore “anchorNotFound” se una richiesta inserita nel server contiene un argomento di ancoraggio che non è stato trovato nei risultati della query.	UC35 Interno
R-038-F-1	È necessario che il client riceva in risposta l'errore “unsupportedSort” se una richiesta inserita nel server presenta una clausola di ordinamento non supportata o un metodo di collezione non riconosciuto dal server.	UC36 Interno
R-039-F-1	È necessario che il client riceva in risposta l'errore “unsupportedFilter” se una richiesta inserita nel server contiene un filtro che il server non è grado di elaborare.	UC37 Interno

R-040-F-1	È necessario che il client riceva in risposta l'errore “tooManyChanges” se una richiesta inserita nel server contiene un numero di modifiche superiore all'argomento maxChanges inserito del client.	UC38 Interno
R-041-F-1	È necessario che il client riceva in risposta l'errore “mailboxHasChild” se una richiesta inserita nel server desidera rimuovere una cartella(Mailbox) che ha ancora almeno una cartella figlia.	UC39 Interno
R-042-F-1	È necessario che il client riceva in risposta l'errore “mailboxHasEmail” se una richiesta inserita nel server, con l'argomento onDestroyRemoveEmails impostato su false, desidera rimuovere una cartella(Mailbox) che ha al suo interno almeno una email.	UC40 Interno
R-043-F-1	È necessario che il client riceva in risposta l'errore “invalidEmail” se una richiesta inserita nel server contiene un'email da inviare non valida.	UC41 Interno
R-044-F-1	È necessario che il client riceva in risposta l'errore “tooManyRecipients” se una richiesta inserita nel server contiene un envelope (insieme di destinatari) che ha più destinatari di quanti il server consenta.	UC42 Interno
R-045-F-1	È necessario che il client riceva in risposta l'errore “noRecipients” se una richiesta inserita nel server contiene un envelope (insieme di destinatari) che non presenta alcun destinatario.	UC43 Interno
R-046-F-1	È necessario che il client riceva in risposta l'errore “invalidRecipients” se una richiesta inserita nel server contiene un envelope (insieme di destinatari) con almeno un indirizzo email destinatario non valido.	UC44 Interno
R-047-F-1	È necessario che il client riceva in risposta l'errore “forbiddenMailFrom” se una richiesta è inserita in un server che non consente all'utente di inviare un messaggio con quel indirizzo mittente nell'envelope (From address).	UC45 Interno
R-048-F-1	È necessario che il client riceva in risposta l'errore “forbiddenFrom” se una richiesta è inserita in un server che non consente all'utente di inviare un messaggio con il campo di intestazione From del messaggio da inviare.	UC46 Interno
R-049-F-1	È necessario che il client riceva in risposta l'errore “forbiddenToSend” se una richiesta è inserita in un server che non consente all'utente di inviare un messaggio in quel momento.	UC47 Interno
	INVIO EMAIL	
R-050-F-1	L'utente che utilizza un client di posta elettronica per interagire con il server deve avere la possibilità di inviare email.	UC48 Capitolato

R-051-F-1	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie all'invio di una email.	UC48.1 Interno
R-052-F-1	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie all'invio di una email, con relativi parametri (sia quelli comuni, come l'identificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC48.2 UC48.2.1 UC48.2.1.1 UC48.2.1.2 UC48.2.1.3 UC48.2.1.4 UC48.2.1.5 UC48.2.2 UC48.2.2.1 UC48.2.2.2 UC48.2.2.3 UC48.2.2.4 UC48.2.2.5 UC48.2.2.6 UC48.2.2.7 Interno
R-053-F-1	È necessario che il client inserisca all'interno della richiesta le proprietà dell'oggetto Email da creare (cartelle in cui è contenuta, mittente, destinatari, oggetto, corpo del messaggio ed altri dettagli definiti dall'RFC5322).	UC48.2.1.3.1 UC48.2.1.3.1.1 UC48.2.1.3.1.2 UC48.2.1.3.1.3 Interno
R-054-F-1	È necessario che il client inserisca all'interno della richiesta le proprietà dell'oggetto EmailSubmission da creare (l'identificativo dell'email creata in precedenza ed ora da inviare, le informazioni necessarie per l'invio ed altri dettagli).	UC48.2.2.3.1 Interno
R-055-F-1	È possibile che il client inserisca all'interno della richiesta eventuali azioni da compiere in seguito al corretto invio dell'email.	UC48.2.2.6 UC48.2.2.7 Interno
R-056-F-1	È necessario che il client riceva una risposta che contiene l'esito dell'operazione di invio dell'email con relativi parametri (come il nuovo stato, gli oggetti creati ed eventuali errori)	UC48 Interno
	RICEZIONE EMAIL	
R-057-F-1	L'utente che utilizza un client di posta elettronica per interagire con il server deve avere la possibilità di ricevere email e visualizzarne il dettaglio.	UC49 Capitolato
R-058-F-1	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie alla ricezione di una email.	UC49.1 Interno
R-059-F-1	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie alla ricezione di una email, con relativi parametri (sia quelli comuni, come l'identificati-	UC49.2 UC49.2.1 UC49.2.1.1

	vo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC49.2.1.2 UC49.2.1.3 UC49.2.1.4 UC49.2.1.5 UC49.2.1.6 UC49.2.1.7 UC49.2.1.8 Interno
R-060-F-1	È necessario che il client inserisca all'interno della richiesta l'identificativo delle email da ricevere	UC49.2.1.2 Interno
R-061-F-1	È possibile che il client inserisca all'interno della richiesta le proprietà specifiche delle email che è interessato a ricevere	UC49.2.1.3 UC49.2.1.4 UC49.2.1.5 UC49.2.1.6 UC49.2.1.7 UC49.2.1.8 Interno
R-062-F-1	È necessario che il client riceva una risposta che contiene l'esito dell'operazione di ricezione dell'email con relativi parametri (come lo stato corrente dei dati di tipo Email sul server, la lista delle email richieste ed eventuali errori)	UC49 Interno
	ELIMINAZIONE EMAIL	
R-063-F-1	L'utente che utilizza un client di posta elettronica per interagire con il server deve avere la possibilità di eliminare email.	UC50 Capitolato
R-064-F-1	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie all'eliminazione di una email.	UC50.1 Interno
R-065-F-1	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie all'eliminazione di una email, con relativi parametri (sia quelli comuni, come l'identificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC50.2 UC50.2.1 UC50.2.1.1 UC50.2.1.2 UC50.2.1.3 UC50.2.1.4 UC50.2.1.5 Interno
R-066-F-1	È necessario che il client inserisca all'interno della richiesta la lista degli identificativi delle email da eliminare.	UC50.2.1.5 Interno
R-067-F-1	È necessario che il client riceva una risposta che contiene l'esito dell'operazione di eliminazione delle email con relativi parametri (come il nuovo stato, gli identificativi degli oggetti eliminati ed eventuali errori)	UC50 Interno
	RICEZIONE CARTELLA	

R-068-F-1	L'utente che utilizza un client di posta elettronica per interagire con il server deve avere la possibilità di ricevere cartelle e visualizzarne il dettaglio.	UC51 Capitolato
R-069-F-1	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie alla ricezione di una cartella.	UC51.1 Interno
R-070-F-1	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie alla ricezione di una cartella, con relativi parametri (sia quelli comuni, come l'identificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC51.2 UC51.2.1 UC51.2.1.1 UC51.2.1.2 UC51.2.1.3 Interno
R-071-F-1	È necessario che il client inserisca all'interno della richiesta l'identificativo delle cartelle da ricevere	UC51.2.1.2 Interno
R-072-F-1	È possibile che il client inserisca all'interno della richiesta le proprietà specifiche delle cartelle che è interessato a ricevere	UC51.2.1.3 Interno
R-073-F-1	È necessario che il client riceva una risposta che contiene l'esito dell'operazione di ricezione della cartella con relativi parametri (come lo stato corrente dei dati di tipo Mailbox sul server, la lista delle cartelle richieste ed eventuali errori)	UC51 Interno
	CREAZIONE CARTELLA	
R-074-F-1	L'utente che utilizza un client di posta elettronica per interagire con il server deve avere la possibilità di creare cartelle.	UC52 Capitolato
R-075-F-1	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie alla creazione di una cartella.	UC52.1 Interno
R-076-F-1	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie alla creazione di una cartella, con relativi parametri (sia quelli comuni, come l'identificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC52.2 UC52.2.1 UC52.2.1.1 UC52.2.1.2 UC52.2.1.3 UC52.2.1.4 UC52.2.1.5 Interno
R-077-F-1	È necessario che il client inserisca all'interno della richiesta le proprietà dell'oggetto Mailbox da creare (nome, eventuale genitore, ruolo ed altri dettagli).	UC52.2.1.3.1 Interno
R-078-F-1	È necessario che il client riceva una risposta che contiene l'esito dell'operazione di creazione della cartella con relativi parametri (come lo stato corrente del server, la lista delle cartelle create ed eventuali errori)	UC52 Interno

	MODIFICA CARTELLA	
R-079-F-1	L'utente che utilizza un client di posta elettronica per interagire con il server deve avere la possibilità di modificare cartelle esistenti.	UC53 Capitolato
R-080-F-1	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie alla modifica di una cartella.	UC53.1 Interno
R-081-F-1	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie alla modifica di una cartella, con relativi parametri (sia quelli comuni, come l'identificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC53.2 UC53.2.1 UC53.2.1.1 UC53.2.1.2 UC53.2.1.3 UC53.2.1.4 UC53.2.1.5 Interno
R-082-F-1	È necessario che il client inserisca all'interno della richiesta le modifiche da apportare all'oggetto Mailbox che l'utente desidera modificare.	UC53.2.1.4.1 Interno
R-083-F-1	È necessario che il client riceva una risposta che contiene l'esito dell'operazione di modifica della cartella con relativi parametri (come lo stato corrente del server, la lista delle cartelle modificate ed eventuali errori)	UC53 Interno
	ELIMINAZIONE CARTELLA	
R-084-F-1	L'utente che utilizza un client di posta elettronica per interagire con il server deve avere la possibilità di eliminare cartelle esistenti.	UC54 Capitolato
R-085-F-1	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie alla eliminazione di una cartella.	UC54.1 Interno
R-086-F-1	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie alla eliminazione di una cartella, con relativi parametri (sia quelli comuni, come l'identificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC54.2 UC54.2.1 UC54.2.1.1 UC54.2.1.2 UC54.2.1.3 UC54.2.1.4 UC54.2.1.5 UC54.2.1.6 Interno
R-087-F-1	È necessario che il client inserisca all'interno della richiesta la lista degli identificativi delle cartelle da eliminare.	UC54.2.1.5 Interno
R-088-F-1	È necessario che il client specifichi all'interno della richiesta il comportamento desiderato da parte del server quando si	UC54.2.1.6 Interno

	cerca di eliminare una cartella che contiene ancora delle email.	
R-089-F-1	È necessario che il client riceva una risposta che contiene l'esito dell'operazione di eliminazione della cartella con relativi parametri (come lo stato corrente del server, la lista degli identificativi delle cartelle eliminate ed eventuali errori)	UC54 Interno
	<b>GESTIONE CONTENUTI CARTELLA</b>	
R-090-F-1	L'utente che utilizza un client di posta elettronica per interagire con il server deve avere la possibilità di gestire i contenuti di una cartella.	UC55 Capitolato
R-091-F-1	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie alla gestione dei contenuti di una cartella.	UC55.1 Interno
R-092-F-1	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie alla gestione dei contenuti di una cartella, con relativi parametri (sia quelli comuni, come l'identificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC55.2 UC55.2.1 UC55.2.1.1 UC55.2.1.2 UC55.2.1.3 UC55.2.1.4 UC55.2.1.5 UC55.2.1.6 Interno
R-093-F-1	È necessario che il client inserisca all'interno della richiesta le modifiche da apportare agli oggetti Email che l'utente desidera aggiungere ad una o più cartelle.	UC55.2.1.4.1 Interno
R-094-F-1	È necessario che il client inserisca all'interno della richiesta le modifiche da apportare agli oggetti Email che l'utente desidera rimuovere da una o più cartelle.	UC55.2.1.4.2 Interno
R-095-F-1	È necessario che il client inserisca all'interno della richiesta le modifiche da apportare agli oggetti Email che l'utente desidera spostare da una o più cartelle ad una o più cartelle.	UC55.2.1.4.3 Interno
R-096-F-1	È necessario che il client riceva una risposta che contiene l'esito delle operazioni di gestione dei contenuti di una cartella con relativi parametri (come lo stato corrente del server, la lista delle email modificate ed eventuali errori)	UC55 Interno
	<b>CREAZIONE CONDIVISIONE CARTELLA</b>	
R-097-F-2	L'utente che utilizza un client di posta elettronica per interagire con il server deve avere la possibilità di condividere le sue cartelle con altri utenti.	UC56 Capitolato

R-098-F-2	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie alla creazione della condivisione di una cartella.	UC56.1 Interno
R-099-F-2	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie alla creazione della condivisione di una cartella, con relativi parametri (sia quelli comuni, come l'identificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC56.2 UC56.2.1 UC56.2.1.1 UC56.2.1.2 UC56.2.1.3 UC56.2.1.4 UC56.2.1.5 UC56.2.2 UC56.2.2.1 UC56.2.2.2 UC56.2.2.3 UC56.2.2.4 UC56.2.2.5 Interno
R-100-F-2	È necessario che il client inserisca all'interno della richiesta le proprietà dell'oggetto Principal da creare (nome, tipo, descrizione, gli identificativi degli account a cui vogliamo condividere le cartelle e altri dettagli).	UC56.2.1.3.1 Interno
R-101-F-2	È necessario che il client inserisca all'interno della richiesta le modifiche da apportare all'oggetto Mailbox che l'utente desidera condividere, specificando quali diritti hanno su quest'ultimo i membri del principale a cui si sta condividendo.	UC56.2.2.4.1 Interno
R-102-F-2	È necessario che il client riceva una risposta che contiene l'esito delle operazioni di creazione della condivisione di una cartella con relativi parametri (come lo stato corrente dei dati sul server, la lista dei principali creati, la lista delle cartelle modificate ed eventuali errori)	UC56 Interno
	MODIFICA PRINCIPALE	
R-103-F-2	L'utente che utilizza un client di posta elettronica per interagire con il server deve avere la possibilità di modificare principali esistenti.	UC57 Interno
R-104-F-2	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie alla modifica di un principale.	UC57.1 Interno
R-105-F-2	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie alla modifica di un principale, con relativi parametri (sia quelli comuni, come l'identificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC57.2 UC57.2.1 UC57.2.1.1 UC57.2.1.2 UC57.2.1.3 UC57.2.1.4 UC57.2.1.5 Interno

R-106-F-2	È necessario che il client inserisca all'interno della richiesta le modifiche da apportare all'oggetto Principal che l'utente desidera modificare.	UC57.2.1.4.1 Interno
R-107-F-2	È necessario che il client riceva una risposta che contiene l'esito dell'operazione di modifica del principale con relativi parametri (come lo stato corrente del server, la lista dei principali modificati ed eventuali errori)	UC57 Interno
	ELIMINAZIONE PRINCIPALE	
R-108-F-2	L'utente che utilizza un client di posta elettronica per interagire con il server deve avere la possibilità di eliminare principali esistenti.	UC58 Interno
R-109-F-2	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie alla eliminazione di un principale.	UC58.1 Interno
R-110-F-2	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie alla eliminazione di un principale, con relativi parametri (sia quelli comuni, come l'identificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC58.2 UC58.2.1 UC58.2.1.1 UC58.2.1.2 UC58.2.1.3 UC58.2.1.4 UC58.2.1.5 Interno
R-111-F-2	È necessario che il client inserisca all'interno della richiesta la lista degli identificativi dei principali da eliminare.	UC58.2.1.5 Interno
R-112-F-2	È necessario che il client riceva una risposta che contiene l'esito dell'operazione di eliminazione del principale con relativi parametri (come lo stato corrente del server, la lista degli identificativi dei principali eliminati ed eventuali errori)	UC58 Interno
	MODIFICA/ELIMINAZIONE CONDIVISIONE CARTELLA	
R-113-F-2	L'utente che utilizza un client di posta elettronica per interagire con il server deve avere la possibilità di modificare la condivisione di una cartella (compresa l'eliminazione di quest'ultima).	UC59 Capitolato
R-114-F-2	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie alla modifica della condivisione di una cartella (compresa l'eliminazione di quest'ultima).	UC59.1 Interno
R-115-F-2	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie alla modifica della condivisione di una cartella (compresa l'eliminazione di quest'ultima), con relativi parametri (sia quelli comuni, come l'iden-	UC59.2 UC59.2.1 UC59.2.1.1 UC59.2.1.2 UC59.2.1.3

	tificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC59.2.1.4 UC59.2.1.5 Interno
R-116-F-2	È necessario che il client inserisca all'interno della richiesta le modifiche da apportare all'oggetto Mailbox di cui l'utente desidera modificare/eliminare la condivisione, specificando i nuovi diritti che hanno su quest'ultimo i membri del principale a cui si sta condividendo.	UC59.2.1.4.1 Interno
R-117-F-2	È necessario che il client riceva una risposta che contiene l'esito dell'operazione di modifica della condivisione di una cartella (compresa l'eliminazione di quest'ultima) con relativi parametri (come lo stato corrente del server, la lista degli identificativi delle cartelle modificate ed eventuali errori)	UC59 Interno
	SINCRONIZZAZIONE EMAIL	
R-118-F-3	Un client di posta elettronica utilizzato da un utente per interagire con il server deve avere la possibilità di mantenersi sincronizzato con gli ultimi aggiornamenti per quanto riguarda le email.	UC60 Capitolato
R-119-F-3	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie alla sincronizzazione delle email.	UC60.1 Interno
R-120-F-3	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie alla sincronizzazione delle email, con relativi parametri (sia quelli comuni, come l'identificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC60.2 UC60.2.1 UC60.2.1.1 UC60.2.1.2 UC60.2.1.3 Interno
R-121-F-3	È necessario che il client inserisca all'interno della richiesta il suo stato corrente per quanto riguarda le email, con lo scopo di sincronizzarsi.	UC60.2.1.2 Interno
R-122-F-3	È necessario che il client inserisca all'interno della richiesta il numero massimo di identificatori di email che desidera ricevere come risposta, con lo scopo di sincronizzarsi.	UC60.2.1.3 Interno
R-123-F-3	È necessario che il client riceva una risposta che contiene le informazioni di cui ha bisogno per sincronizzarsi (come lo stato corrente del server, un flag booleano che indica se ci sono ulteriori cambiamenti nel server relativi ai dati di tipo Email, oltre a quelli già restituiti nella risposta corrente, e la lista delle email da creare/modificare/eliminare per sincronizzarsi)	UC60 Interno
	SINCRONIZZAZIONE CARTELLE	
R-124-F-3	Un client di posta elettronica utilizzato da un utente per interagire con il server deve avere la possibilità di mantenersi	UC61 Capitolato

	sincronizzato con gli ultimi aggiornamenti per quanto riguarda le cartelle.	
R-125-F-3	È necessario che il client inserisca all'interno della richiesta l'identificazione delle capacità necessarie alla sincronizzazione delle cartelle.	UC61.1 Interno
R-126-F-3	È necessario che il client inserisca all'interno della richiesta le chiamate di metodo necessarie alla sincronizzazione delle cartelle, con relativi parametri (sia quelli comuni, come l'identificativo dell'account da utilizzare, sia quelli specifici del caso) e un identificatore univoco associato.	UC61.2 UC61.2.1 UC61.2.1.1 UC61.2.1.2 UC61.2.1.3 Interno
R-127-F-3	È necessario che il client inserisca all'interno della richiesta il suo stato corrente per quanto riguarda le cartelle, con lo scopo di sincronizzarsi.	UC61.2.1.2 Interno
R-128-F-3	È necessario che il client inserisca all'interno della richiesta il numero massimo di identificatori di cartelle che desidera ricevere come risposta, con lo scopo di sincronizzarsi.	UC61.2.1.3 Interno
R-129-F-3	È necessario che il client riceva una risposta che contiene le informazioni di cui ha bisogno per sincronizzarsi (come lo stato corrente del server, un flag booleano che indica se ci sono ulteriori cambiamenti nel server relativi ai dati di tipo Mailbox, oltre a quelli già restituiti nella risposta corrente, e la lista delle cartelle da creare/modificare/eliminare per sincronizzarsi)	UC61 Interno

Tabella 1: requisiti di funzionalità

## 4.2) Requisiti di qualità

ID Requisito	Descrizione	Fonti
R-001-Q-1	Occorre realizzare e consegnare un documento di analisi dei requisiti con un diagramma dei casi d'uso in formato UML.	Capitolato
R-002-Q-1	Occorre realizzare e consegnare uno schema della base di dati utilizzata in formato di diagramma entità relazione.	Capitolato
R-003-Q-1	Occorre implementare ed esporre ai fruitori del nostro sistema una lista di endpoint cosicché un client possa fruire dei vari servizi.	Interno
R-004-Q-1	Occorre realizzare e consegnare una lista dei bug risolti durante la fase di sviluppo.	Interno
R-005-Q-1	Occorre realizzare e consegnare un docker file contenente la componente applicativa del server.	Capitolato
R-006-Q-1	Occorre realizzare il codice sorgente con un sistema di versamento.	Capitolato
R-007-Q-2	Occorre effettuare una serie di stress test (test di carico) per alcuni casi d'uso implementati e riportare con un apposito documento i benchmark del sistema in ogni situazione: normale, di carico e di sovraccarico. Questi andranno specificati all'interno del <b>Piano di Qualifica</b> in una sezione dedicata.	Capitolato
R-008-Q-1	La codifica dell'intera soluzione informativa deve essere conforme con quanto riportato nella sezione <b>Obiettivi metrici di qualità</b> del documento <b>Piano di Qualifica v1.0.0</b> .	Interno
R-009-Q-1	Nello creazione del prodotto software devono essere rispettate tutte le norme definite nelle sezioni <b>Sviluppo, Verifica e Validazione</b> del documento <b>Norme di progetto v1.0.0</b> .	Interno

Tabella 2: requisiti di qualità

### 4.3) Requisiti di vincolo

ID Requisito	Descrizione	Fonti
R-001-V-1	Per implementare il prodotto dovrà essere utilizzato il linguaggio di programmazione Java.	Capitolato
R-002-V-1	Per l'implementazione del protocollo JMAP è preferibile l'utilizzo della libreria iNPUTmice/jmap. Alternativamente va utilizzata l'implementazione presente in uno dei progetti elencati nelle JMAP Software Implementations.	Capitolato
R-003-V-1	Il servizio sviluppato deve essere eseguibile in un sistema container, come Docker.	Capitolato
R-004-V-3	Il servizio sviluppato deve essere scalabile mediante l'inizializzazione di più nodi stateless, ovvero alla richiesta di uno specifico client, fatta ad un'architettura contenente più di un'istanza del servizio dato, deve poter rispondere una qualsiasi istanza del servizio, poiché nessuna istanza deve contenere dati specifici di stato rispetto alle richieste dei client	Capitolato

Tabella 3: requisiti di vincolo

## 5) Tracciamento Requisiti

Fonte	ID Requisiti
Capitolato	R-050-F-1 R-057-F-1 R-063-F-1 R-068-F-1 R-074-F-1 R-079-F-1 R-084-F-1 R-090-F-1 R-097-F-2 R-113-F-2 R-118-F-3 R-124-F-3 R-001-Q-1 R-002-Q-1 R-005-Q-1 R-006-Q-1 R-007-Q-2 R-001-V-1 R-002-V-1 R-003-V-1 R-004-V-3
Interno	R-001-F-2 R-004-F-2 R-005-F-1 R-006-F-1 R-007-F-1 R-008-F-1 R-009-F-1 R-010-F-1 R-011-F-1 R-012-F-1 R-013-F-1 R-014-F-1 R-015-F-1 R-016-F-1 R-017-F-1 R-018-F-1 R-019-F-1 R-020-F-1 R-021-F-1 R-022-F-1 R-023-F-1 R-024-F-1 R-025-F-1 R-026-F-1 R-027-F-1 R-028-F-1 R-029-F-1

	R-030-F-1 R-031-F-1 R-032-F-1 R-033-F-1 R-034-F-1 R-035-F-1 R-036-F-1 R-037-F-1 R-038-F-1 R-039-F-1 R-040-F-1 R-041-F-1 R-042-F-1 R-043-F-1 R-044-F-1 R-045-F-1 R-046-F-1 R-047-F-1 R-048-F-1 R-049-F-1 R-003-Q-1 R-004-Q-1 R-008-Q-1 R-009-Q-1
UC48.1	R-051-F-1
UC48.2 e sottocasi	R-052-F-1
UC48.2.1.3.1 e sottocasi	R-053-F-1
UC48.2.2.3.1	R-054-F-1
UC48.2.2.6 e UC48.2.2.7	R-055-F-1
UC48	R-056-F-1
UC49.1	R-058-F-1
UC49.2 e sottocasi	R-059-F-1
UC49.2.1.2	R-060-F-1
da UC49.2.1.3 a UC49.2.1.8	R-061-F-1
UC49	R-062-F-1
UC50.1	R-064-F-1

UC50.2 e sottocasi	R-065-F-1
UC50.2.1.5	R-066-F-1
UC50	R-067-F-1
UC51.1	R-069-F-1
UC51.2 e sottocasi	R-070-F-1
UC51.2.1.2	R-071-F-1
UC51.2.1.3	R-072-F-1
UC51	R-073-F-1
UC52.1	R-075-F-1
UC52.2 e sottocasi	R-076-F-1
UC52.2.1.3.1	R-077-F-1
UC52	R-078-F-1
UC53.1	R-080-F-1
UC53.2 e sottocasi	R-081-F-1
UC53.2.1.4.1	R-082-F-1
UC53	R-083-F-1
UC54.1	R-085-F-1
UC54.2 e sottocasi	R-086-F-1
UC54.2.1.5	R-087-F-1
UC54.2.1.6	R-088-F-1
UC54	R-089-F-1
UC55.1	R-091-F-1
UC55.2 e sottocasi	R-092-F-1
UC55.2.1.4.1	R-093-F-1
UC55.2.1.4.2	R-094-F-1

UC55.2.1.4.3	R-095-F-1
UC55	R-096-F-1
UC56.1	R-098-F-2
UC56.2 e sottocasi	R-099-F-2
UC56.2.1.3.1	R-100-F-2
UC56.2.2.4.1	R-101-F-2
UC56	R-102-F-2
UC57.1	R-104-F-2
UC57.2 e sottocasi	R-105-F-2
UC57.2.1.4.1	R-106-F-2
UC57	R-107-F-2
UC58.1	R-109-F-2
UC58.2 e sottocasi	R-110-F-2
UC58.2.1.5	R-111-F-2
UC58	R-112-F-2
UC59.1	R-114-F-2
UC59.2 e sottocasi	R-115-F-2
UC59.2.1.4.1	R-116-F-2
UC59	R-117-F-2
UC60.1	R-119-F-3
UC60.2 e sottocasi	R-120-F-3
UC60.2.1.2	R-121-F-3
UC60.2.1.3	R-122-F-3
UC60	R-123-F-3
UC61.1	R-125-F-3

UC61.2 e sottocasi	R-126-F-3
UC61.2.1.2	R-127-F-3
UC61.2.1.3	R-128-F-3
UC61	R-129-F-3

Tabella 4: tracciamento requisiti

## 5.1) Riepilogo

Tipologia	Obbligatori	Desiderabili	Opzionali
Funzionali	93	25	11
Qualità	8	1	0
Vincolo	3	0	1

Tabella 5: riepilogo

## 5.2) Conclusioni

I requisiti del prodotto possono essere modificati durante il suo ciclo di vita per apportare miglioramenti o aggiornamenti. In caso di disponibilità di risorse sufficienti, il gruppo considererà di aggiungere nuovi requisiti per migliorare e aumentare il valore del prodotto. Pertanto, eventuali aggiornamenti saranno presi in considerazione in futuro.