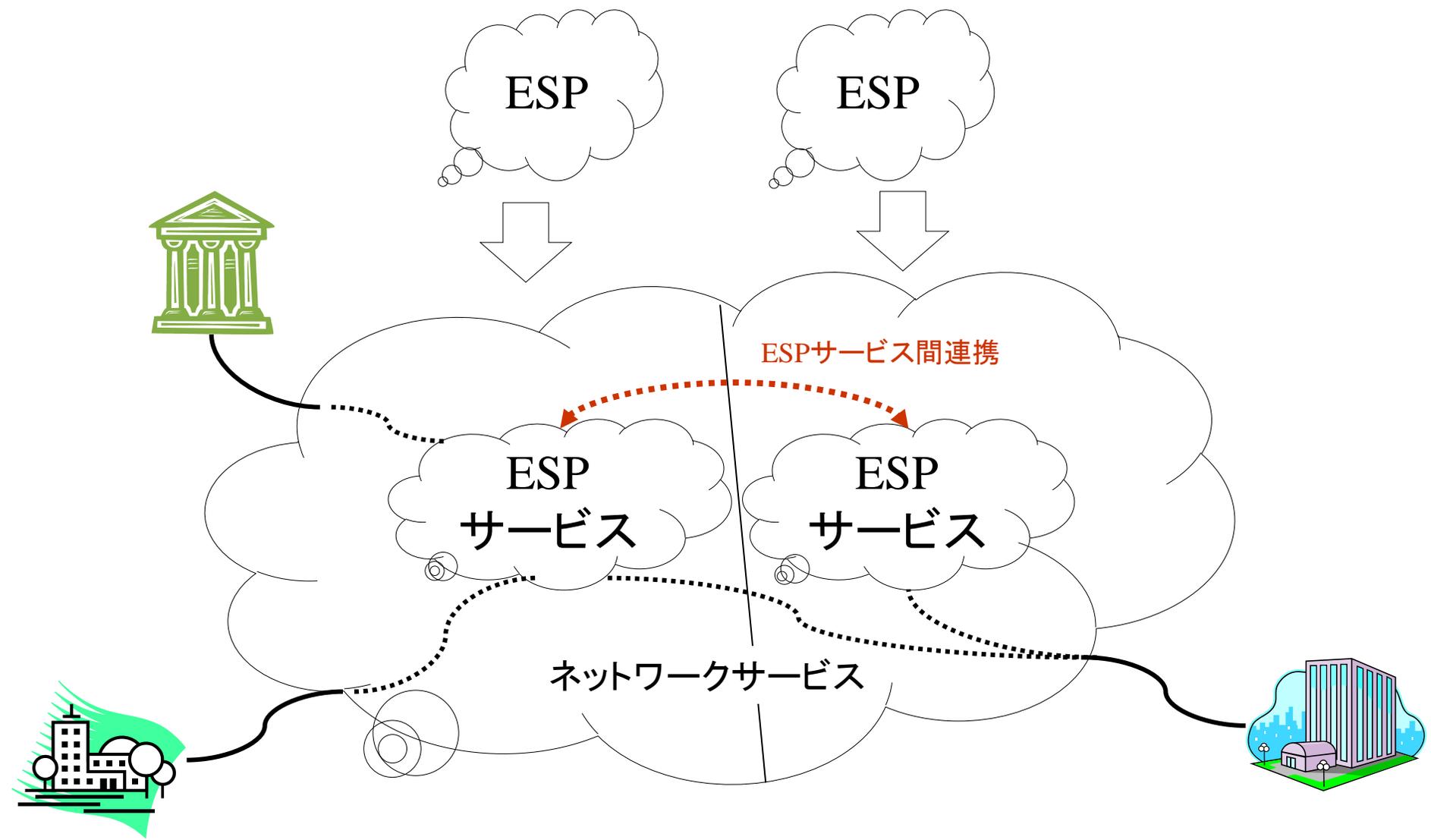


7/24/2017

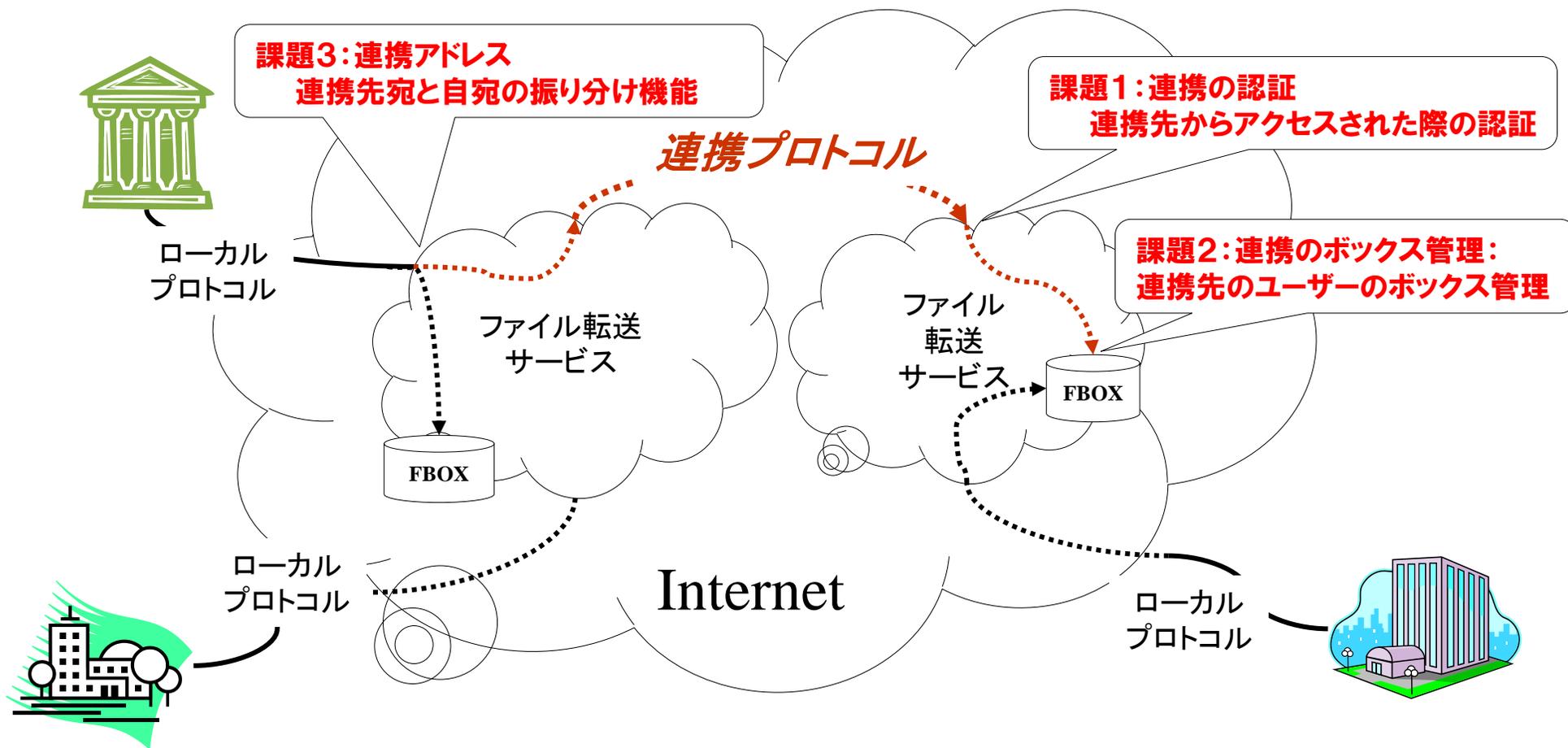
EDI連携/InterVANサービスを目指して

EDIサービスプロバイダ間連携の検討

ESPが連携したネットワークの将来像(コンセプト)

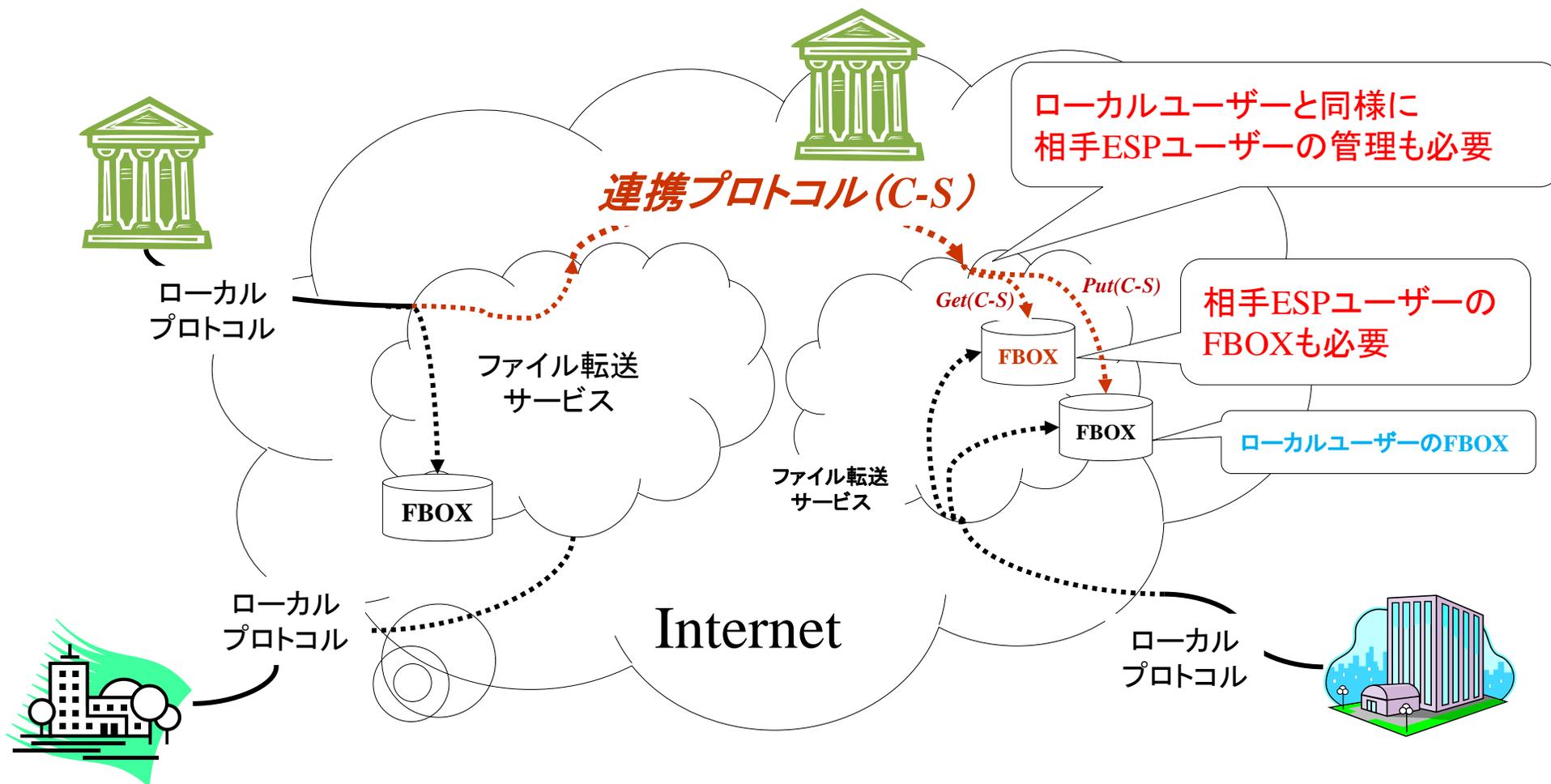


ESP(ファイル転送サービス)の連携



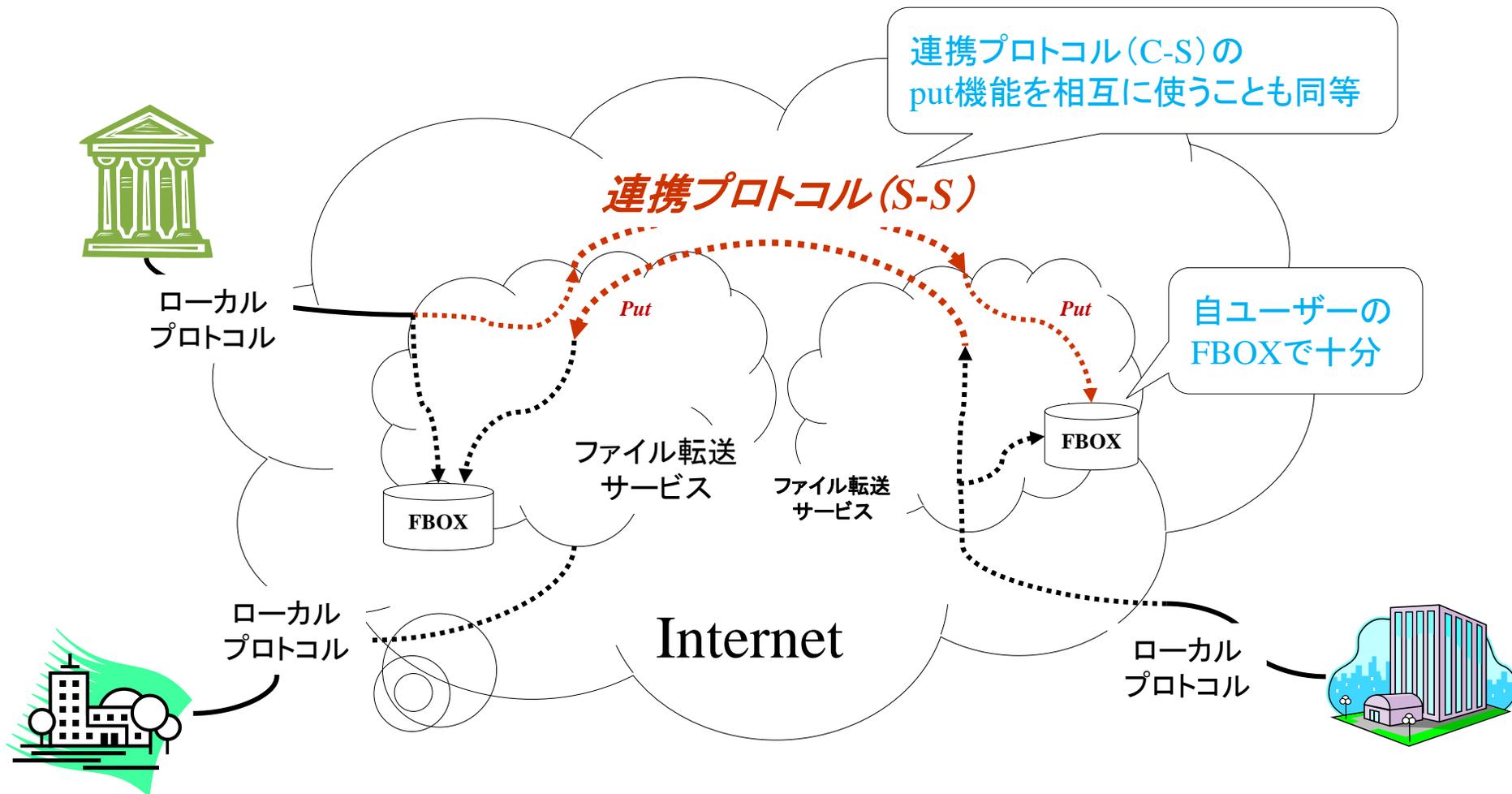
FBOX: ファイルボックス

ESP(ファイル転送サービス)の連携(C-S、代行的認証)



C-S: Client Server Protocol
Get(C-S): Client Server Protocolのファイル取得(Get)
Put(C-S): Client Server Protocolのファイル送付(Put)

ESP(ファイル転送サービス)の連携(S-S)

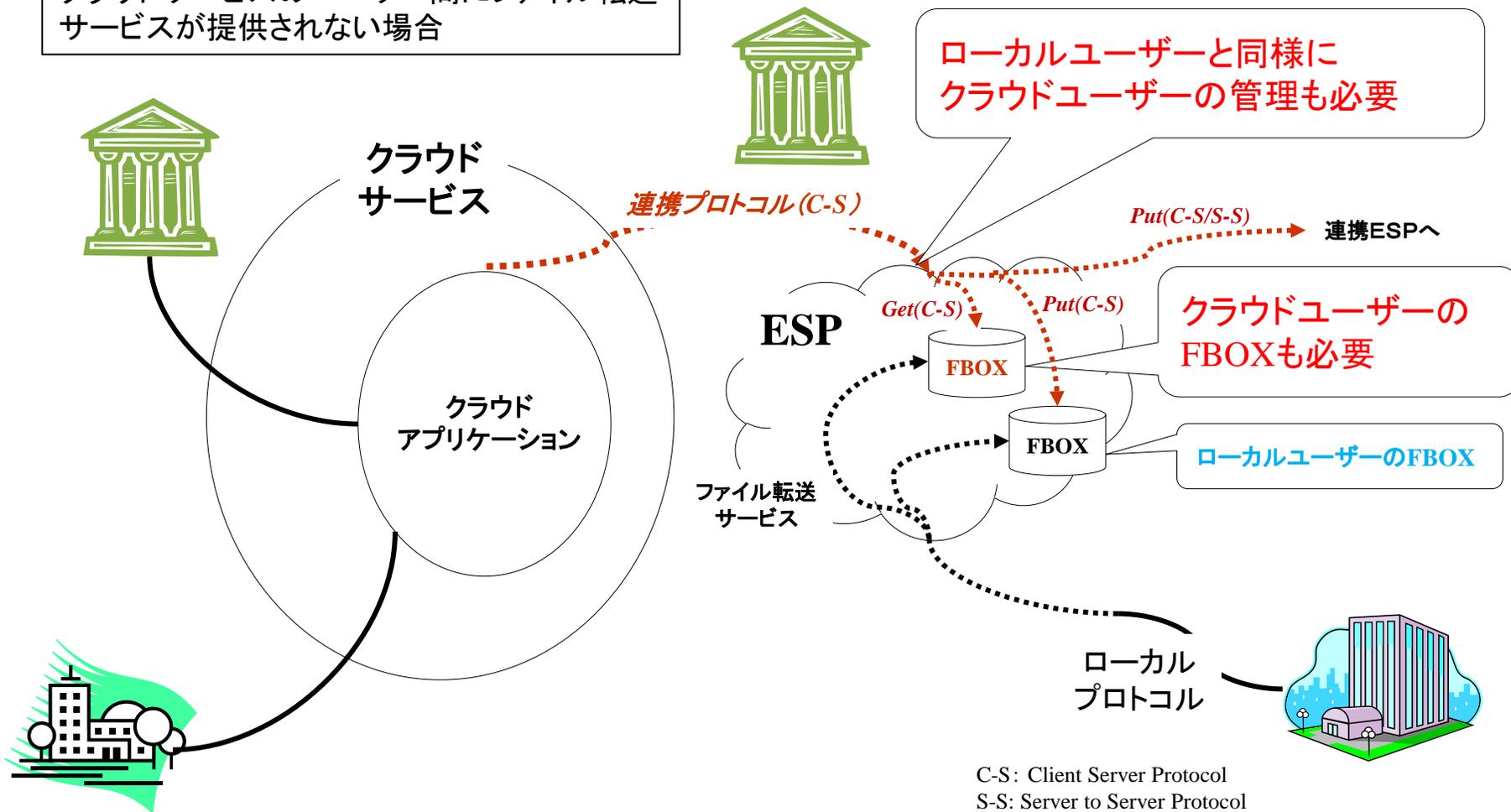


- 代理的認証、代行的認証での違いはない
- アクセスされる側の作業は代理的

S-S: Server to Server Protocol
Put: Server to Server Protocol のファイル送付

クラウド・ESP連携(ファイル転送サービス)(C-S、代行的認証)

クラウドサービスのユーザー間にファイル転送サービスが提供されない場合



C-S: Client Server Protocol

S-S: Server to Server Protocol

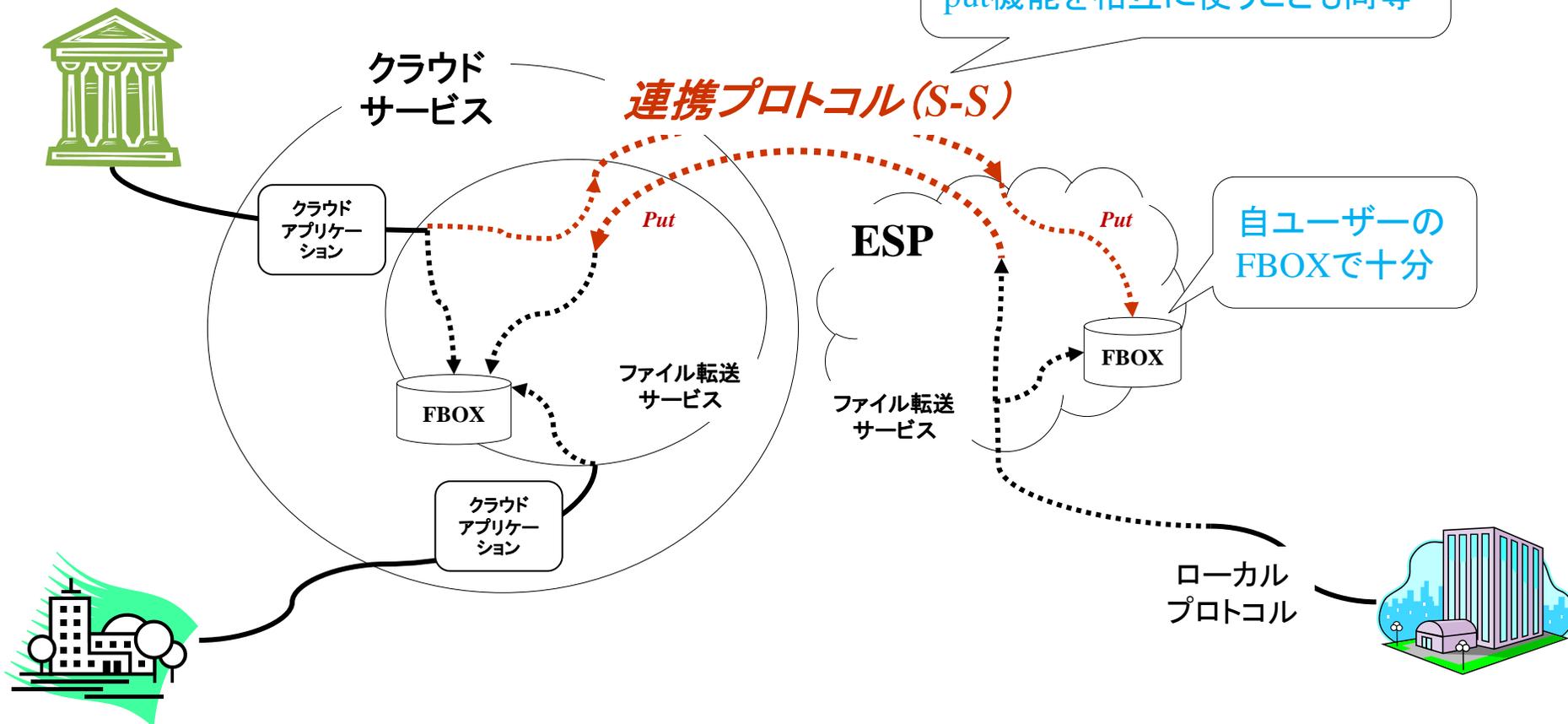
Get(C-S): Client Server Protocolのファイル取得(Get)

Put(C-S): Client Server Protocolのファイル送付(Put)

Put(S-S): Server to Server Protocolのファイル送付(Put)

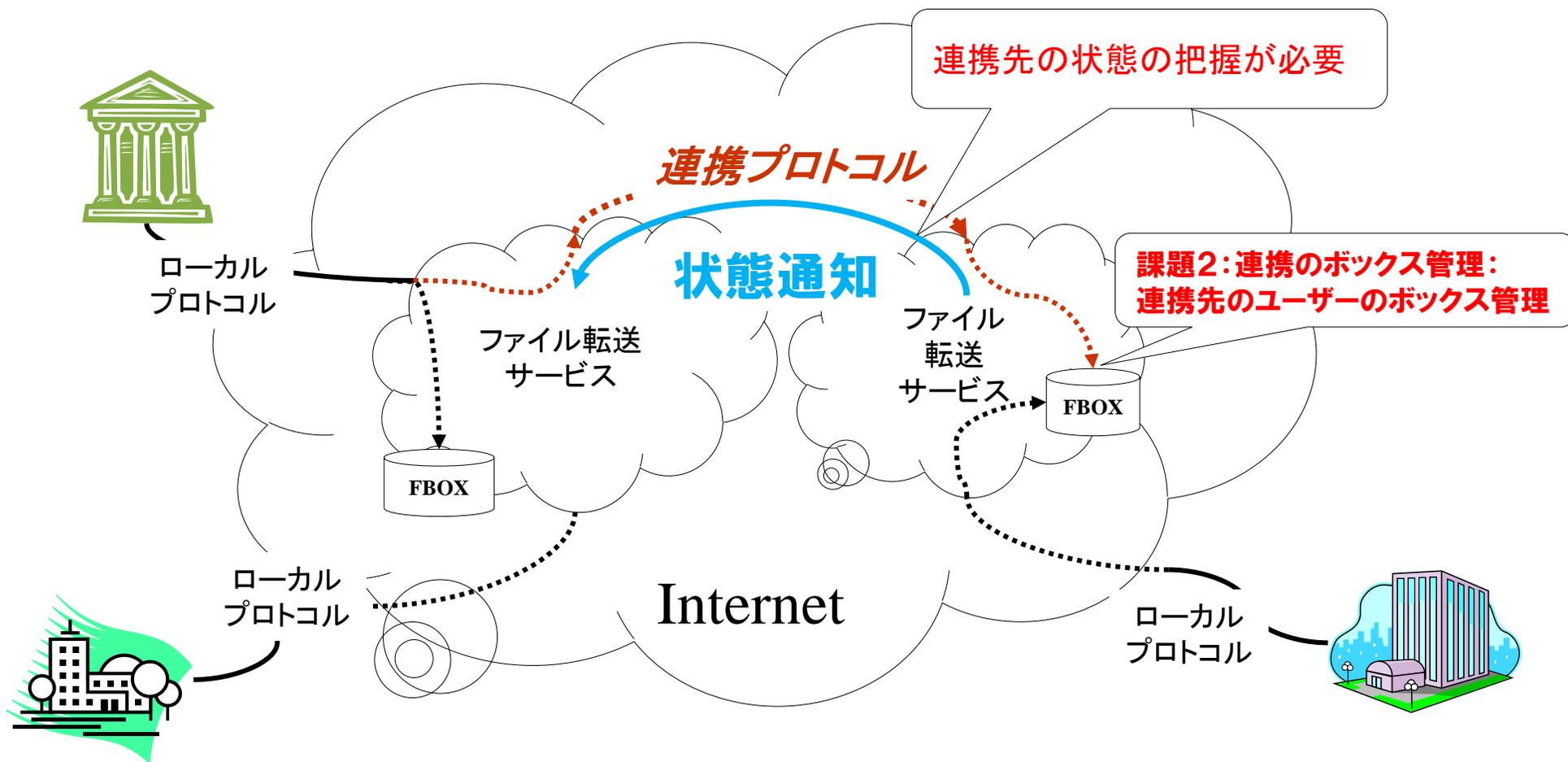
クラウド・ESP連携(ファイル転送サービス)(S-S、代理的認証)

クラウドサービスのユーザー間にクラウドサービスとしてファイル転送サービスが提供される場合



S-S: Server to Server Protocol
Put: Server to Server Protocol のファイル送付

連携先のファイル転送状態把握



FBOX: ファイルボックス

クラウドユーザー向け(C-S)インターフェース

クラウドユーザー・ESP間(C-S)

- (Push) 企業からファイルを送信
 - (Pull) 企業から送信したファイルの状態問合せ
 - (Pull) ESPからファイルを受信
 - (Pull) ESPからファイルを再取得
 - (Push) ファイル受信の完了を通知
-
- ・ 企業から受領したファイルが他ESPのユーザー向けの場合、ESPはESP間インターフェースで転送を行う。
 - ・ 他ESPおよび自ESP内から送付されたファイルを区別なく受信する。
 - ・ 各機能は実装する通信プロトコルが提供する機能にマッピングされ実現される。
 - ・ 認証、信頼性確保、セキュリティ確保は実装する通信プロトコルの機能で実現する。

クラウド内ESPと外部ESP間(S-S)インターフェース

ESP間(S-S)

- (Push) ESPから連携先ESPにファイルを送信
- (Push) 連携先ESPから受信したファイルの状態(送達確認状況など)を通知
- ・ ESP相互にファイルを送信し合う。
- ・ クラウド内ESPがクラウドユーザーから受領したファイルが外部ESPのユーザー向けの場合、ESP間インタフェースで転送を行う。
- ・ クラウドユーザーとクラウド内ESPの間のインタフェースは、クラウドユーザー向け(C-S)インタフェース(案)相当と想定する。
- ・ 各機能は実装する通信プロトコルが提供する機能にマッピングされ実現される。
- ・ 認証、信頼性確保、セキュリティ確保は実装する通信プロトコルの機能で実現する。

ESP連携のためのインターフェース要求(案) 企業・ESP間

企業・ESP間(C-S)

- 企業からファイルを送信
- 企業から送信したファイルの状態問合せ
- ESPから一つのファイルを受信
- ESPから一つのファイルを再取得
- ファイル受信の完了を通知

企業・ESP間(S-S)

- 企業からESPにファイルを送信／ESPから企業にファイルを送信
 - ESPが受信したファイルの状態(送達確認など)を企業に通知／企業のファイルの受信完了をESPに通知
-
- ・ ESP間で伝達された状態は、企業・ESP間がS-Sの場合は状態の通知で、C-Sの場合は状態問合せにより企業に伝達される。
 - ・ 各機能は実装する通信プロトコルが提供する機能にマッピングされて実装される。

ESP連携のためのインターフェース要求 ESP間

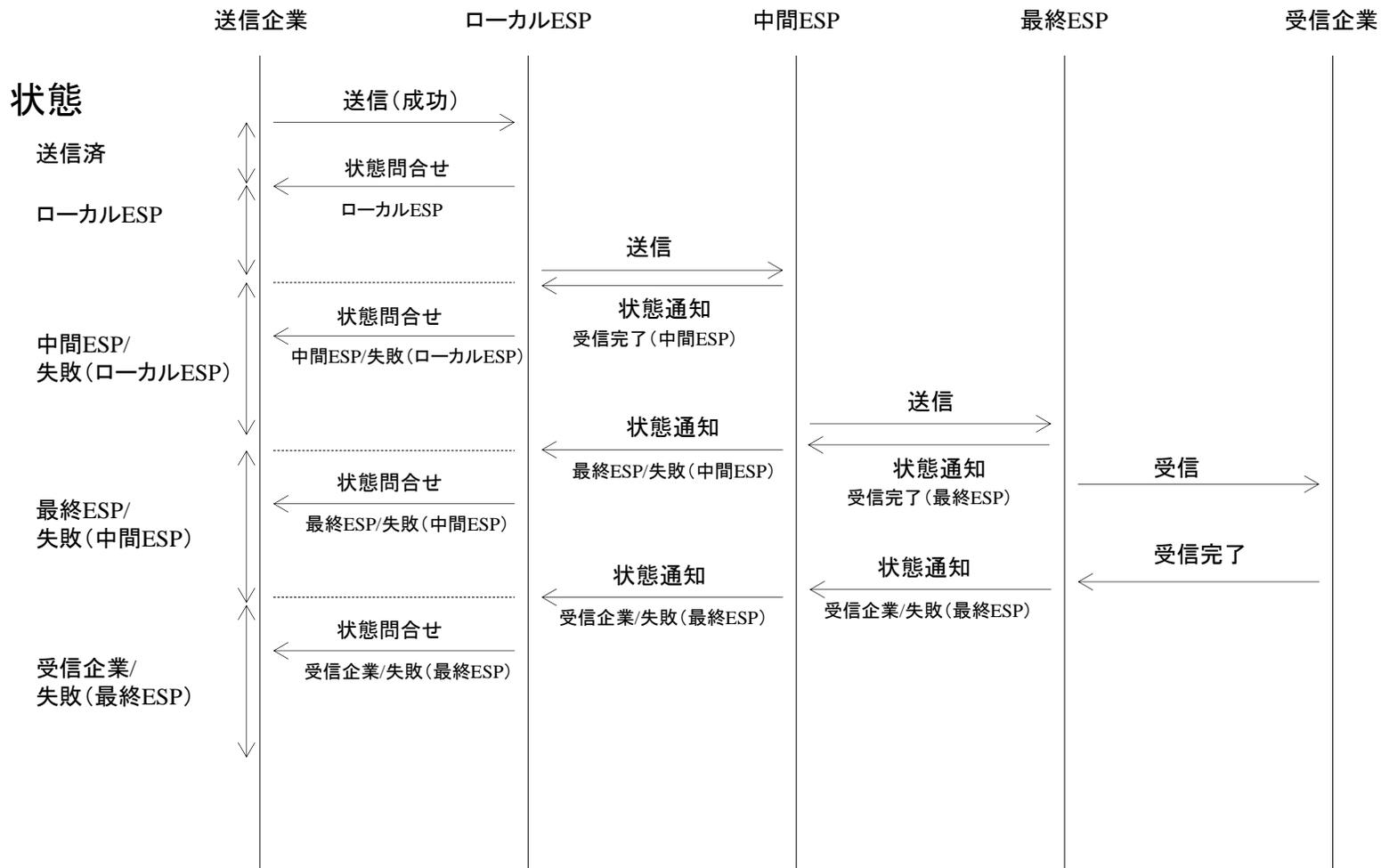
ESP間(C-S)

- ESPからファイルを送信
- ESPから送信したファイルの状態問合せ
- 連携先ESPから一つのファイルを受信
- 連携先ESPからのファイル受信の完了を通知

ESP間(S-S)

- ESPから連携先ESPにファイルを送信
 - 連携先ESPから受信したファイルの状態(送達確認など)を通知
-
- ・ ESP間(C-S)は企業・ESP間(C-S)と同じインタフェースとする。
 - ・ 各機能は実装する通信プロトコルが提供する機能にマッピングされて実装される。

通知する状態(1)



連携アドレスの実現方法

連携アドレス

- username@domain

username : ローカルのESPユーザーアドレス

domain : グローバルに一意になるESPアドレス

- URI acct

acct : username@domain

- 管理システム

username :

- 各ESPが管理

domain :

- (1) 連携ESPアドレスの管理組織が登録・配布

hostsファイル形式

- (2) 連携ESPアドレスの登録システムによる管理

RDAP (次世代WHOISプロトコル RFC7480～7484)を活用

ファイルの識別子の実現方法（1）

ファイルの識別子

- ・ tag URI ~ RFC4151

"tag:" acct "," date ":" sha1

acct : username@domain 連携アドレス

date : year ["-" month ["-" day]]

year : yyyy 西暦数字4桁

month : mm 数字2桁

day : dd 数字2桁

sha1 : US Secure Hash Algorithm 1 (RFC3174)で計算したファイルデータの
ダイジェスト値(16進表記)

ファイルの識別子の実現方法(案)(2)

包括的ファイル識別子

- ・ tag URI ~ RFC4151

"tag:" acct "," date ":" "*"

acct : username@domain 連携アドレス

date : year ["-" month ["-" day]]

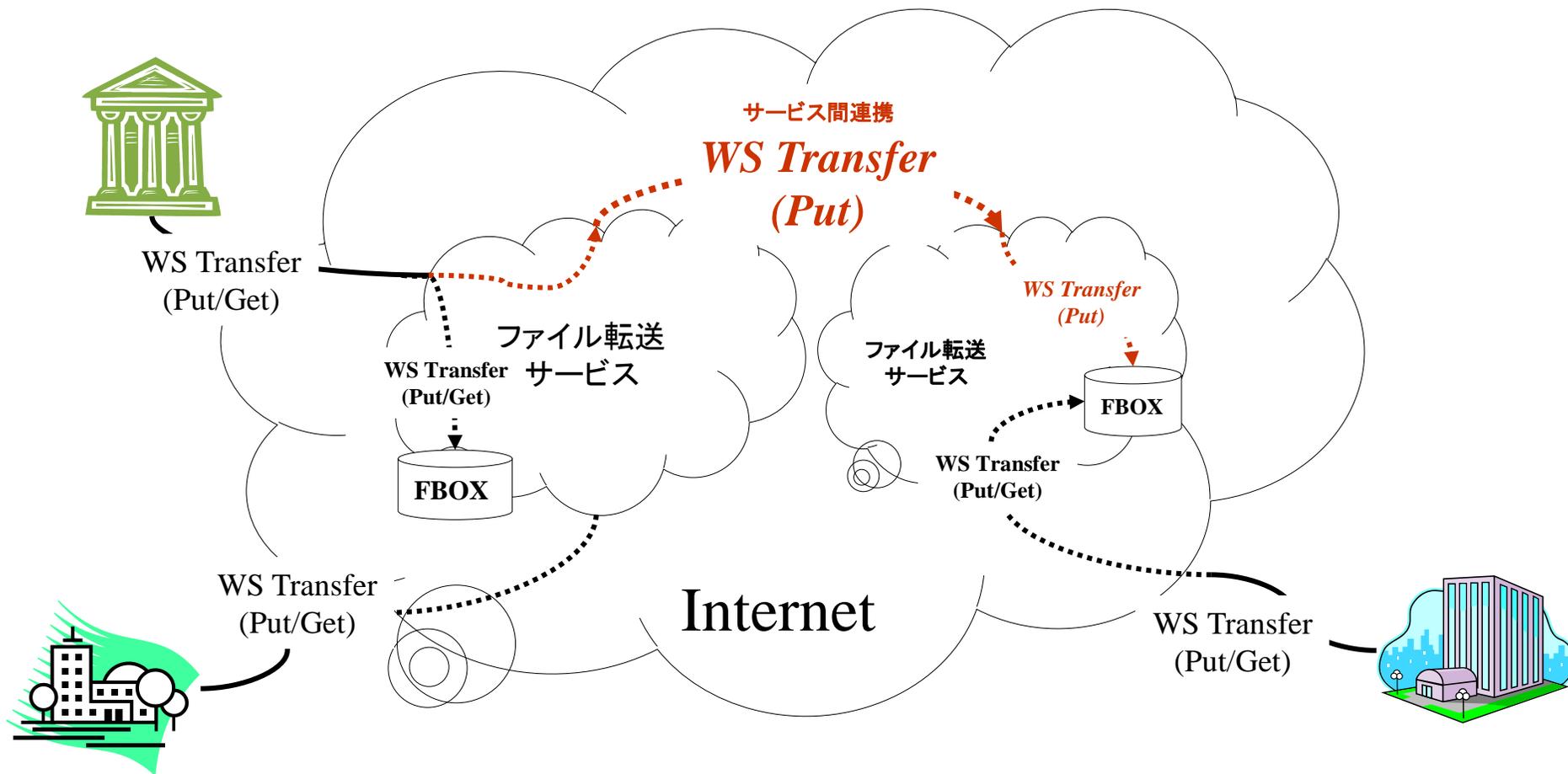
year : yyyy 西暦数字4桁

month : mm 数字2桁

day : dd 数字2桁

実装プロトコルの選定

WS Transferを使ったESP連携プロトコルの実装(案)



WS Transferを使ったESP連携プロトコルの実装(案)

WS Transfer (SOAP 1.1 or 1.2、WS-Addressing 1.1、WS-Policy 1.5)

Put、Get、Delete、Create

- ・ WS TransferはSOAPを使いリモートのXMLリソースを取り扱うための規格
- ・ リモート先にあるXMLリソースの取得、削除、追加、作成を行う

企業・ESP間(C-S)

- | | |
|----------|-------------------------|
| Put | - WS Transfer (Put)を拡張 |
| State | - WS Transfer (Get)を拡張? |
| Get | - WS Transfer (Get)を拡張 |
| Retrieve | - WS Transfer (Get)を拡張? |
| Confirm | - WS Transfer (Put)を拡張? |

ESP間(S-S)

- | | |
|---------|-------------------------|
| Forward | - WS Transfer (Put)を拡張 |
| Notify | - WS Transfer (Put)を拡張? |

ESP連携プロトコルの実装方法

案1) 拡張方式

プロトコルに情報項目やサービス種別を追加(拡張)して実装

- ・ 類似し、かつ拡張余地を持つプロトコルに適する

案2) スタック方式

下位プロトコルのデータとして上位プロトコルのメッセージを授受して実装

- ・ 機能など違いが多いプロトコルに適する
- ・ プロトコルスタックを実現するフレームワークが無いとアプリケーション内での実装となるため複雑なプロトコルは実現が困難
- ・ 既存プロトコル上への実装に向く

ESP連携のためのインターフェース要求・詳細(案)

企業・ESP間(C-S)

企業から送信したファイルの状態問合せ

To - 受信企業のアドレス(acct URI)
From - 送信企業のアドレス(acct URI)
Id - ファイルの識別子(tag URI)

- 状態問合せの結果

Result/Fault - ファイル状態通知／エラー終了通知

ローカルESP/中間ESP/最終ESP/受信企業/失敗(発生場所)

Id - ファイルの識別子(tag URI)

- ・ ESPは企業のアドレスとFromの送信企業のアドレスが同じことを確認し、送信したファイルの状態だけを回答する。

ESP連携のための実装インターフェース詳細(案)

企業・ESP間(C-S)

- 企業から送信したファイルの状態問合せ
(State)

To - 受信企業のアドレス(acct URI)
From - 送信企業のアドレス(acct URI)
Id - ファイルの識別子(tag URI)

- 状態問合せの結果(StateResponse)

Result/Fault - ファイル状態通知/
エラー終了通知

ローカルESP/中間ESP/最終ESP/
受信企業/失敗(発生場所)

Id - ファイルの識別子(tag URI)

| | |
|----------|--|
| Header | wsa:Action http://www.w3.org/2011/03/ws-tra/Get |
| Body | wst:Get/wsa:To acct: xxx@aaa wst:Get/wsa:From/wsa:Address acct: yyy@bbb wst:Get/wsa:MessageID tag:yyy@bbb,2017-06-21:xxxx... wst:Get/wsa:Action http://example.jp/2017/06/ws-tra-fr/ State |
| エラー終了通知 | s:Fault |
| ファイル状態通知 | frtra>Status LocalEsp InterESP DestESP DestUser Fault(URI) wsa:MessageID tag:yyy@bbb,2017-06-21:xxxx... |

StateのRequest定義(案)

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wst="http://www.w3.org/2011/03/ws-tra"
  xmlns:wstfr="http://example.jp/2017/06/ws-tra-fr">
  <s:Header>
    [Headers]
    <wsa:Action>http://www.w3.org/2011/03/ws-tra/Get</wsa:Action>
    ...
  </s:Header>
  <s:Body>
    <wst:Get>
      <wsa:To>acct URK</wsa:To>
      <wsa:From><wsa:Address>acct URK</wsa:Address></wsa:From>
      <wsa:MessageID>tag URK</wsa:MessageID>
      <wsa:Action>http://example.jp/2017/06/ws-tra-fr/State</wsa:Action>
    </wst:Get>
  </s:Body>
</s:Envelope>
```

StateのRequest例

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wst="http://www.w3.org/2011/03/ws-tra"
  xmlns:wstfr="http://example.jp/2017/06/ws-tra-fr">
  <s:Header>
    <wsa:ReplyTo>
      <wsa:Address>http://www.fabrikam123.example.org/pullport</wsa:Address>
    </wsa:ReplyTo>
    <wsa:To>http://www.example.org/repository</wsa:To>
    <wsa:Action>http://www.w3.org/2011/03/ws-tra/Get</wsa:Action>
    <wsa:MessageID>urn:uuid:00000000-0000-0000-C000-000000000046</wsa:MessageID>
  </s:Header>
  <s:Body>
    <wst:Get>
      <wsa:To>acct:xxx@aaa</wsa:To>
      <wsa:From><wsa:Address>acct:yyy@bbb</wsa:Address></wsa:From>
      <wsa:MessageID>tag:yyy@bbb,2017-06-21:xxxx...xx</wsa:MessageID>
      <wsa:Action>http://example.jp/2017/06/ws-tra-fr/State</wsa:Action>
    </wst:Get>
  </s:Body>
</s:Envelope>
```

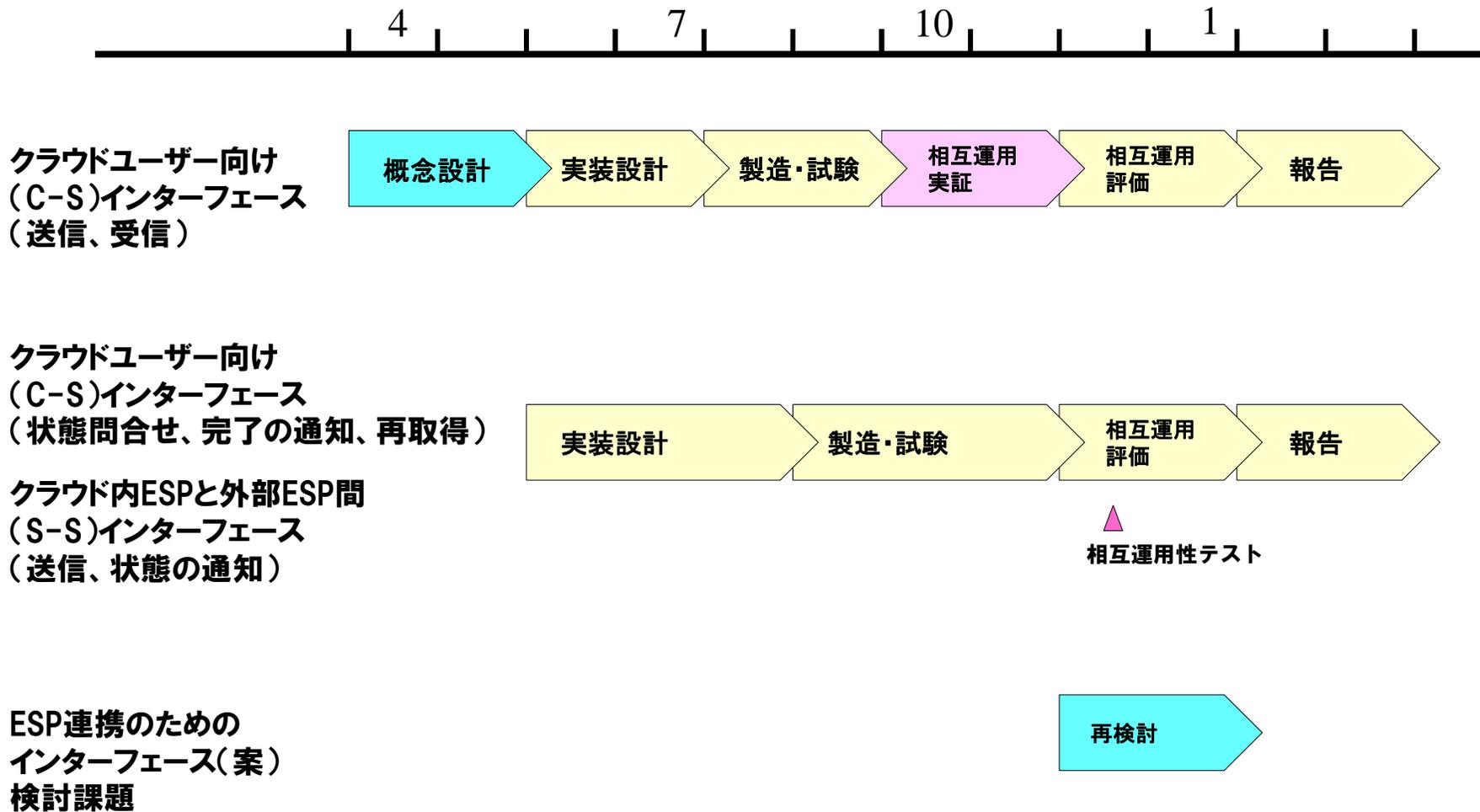
StateのResponse定義(案)

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wst="http://www.w3.org/2011/03/ws-tra"
  xmlns:wstfr="http://example.jp/2017/06/ws-tra-fr">
  <s:Header>
    [Headers]
    <wsa:Action>http://www.w3.org/2011/03/ws-tra/GetResponse</wsa:Action>
    ...
  </s:Header>
  <s:Body>
    <wst:GetResponse ...>
      <wst:Representation ...>
        xs:any?
      </wst:Representation>?
      <wstfr:Status>Status</wstfr:Status>
      <wsa:MessageID>tag URK</wsa:MessageID>
      <wsa:Action>http://example.jp/2017/06/ws-tra-fr/StateResponse</wsa:Action>
      xs:any*
    </wst:GetResponse>
  </s:Body>
</s:Envelope>
```

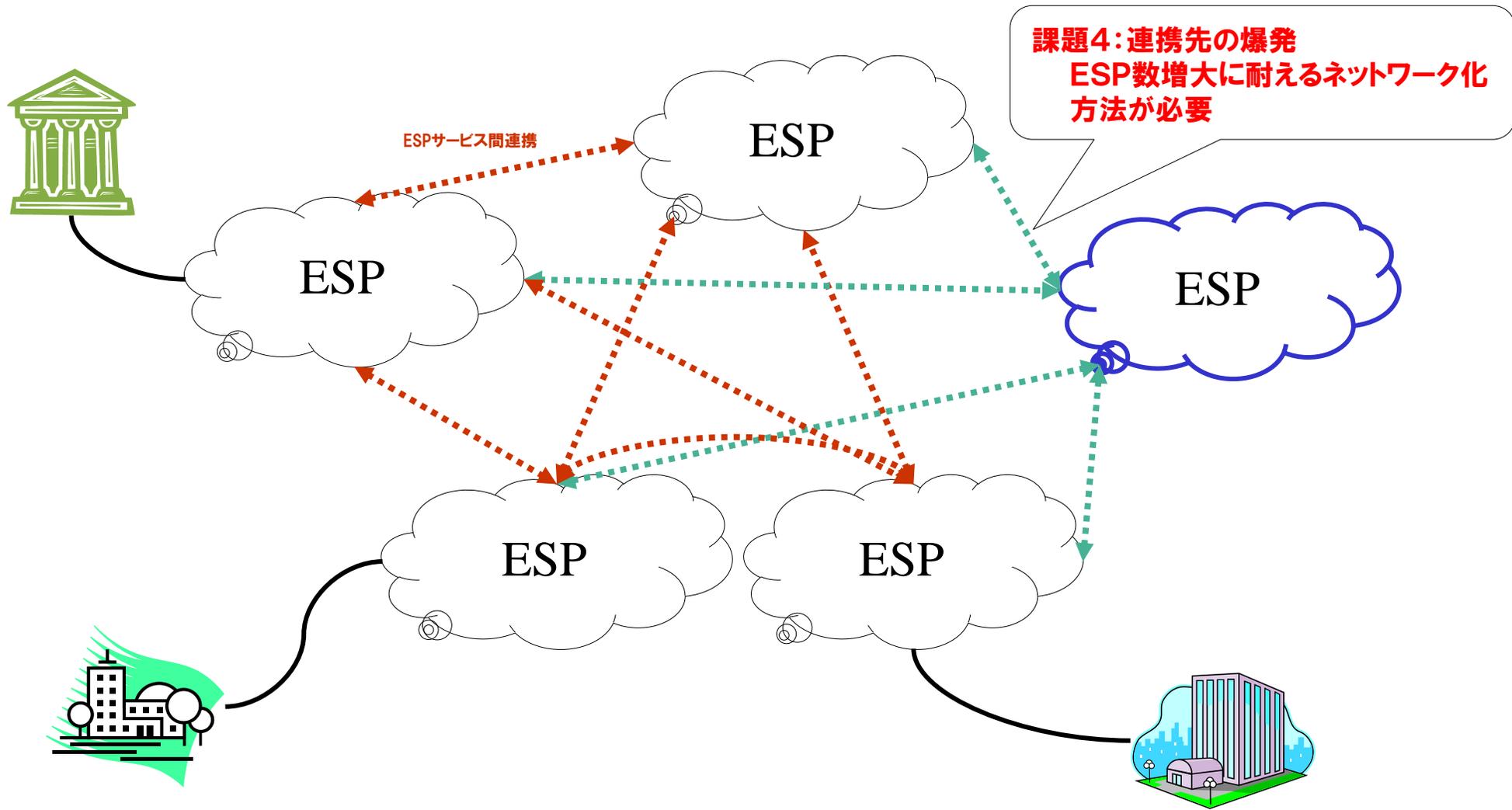
Stateの Response例

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wst="http://www.w3.org/2011/03/ws-tra"
  xmlns:wstfr="http://example.jp/2017/06/ws-tra-fr">
  <s:Header>
    <wsa:To>http://www.fabrikam123.example.org/pullport</wsa:To>
    <wsa:Action>http://www.w3.org/2011/03/ws-tra/GetResponse</wsa:Action>
    <wsa:MessageID>urn:uuid:0000010e-0000-0000-C000-000000000046</wsa:MessageID>
    <wsa:RelatesTo>urn:uuid:00000000-0000-0000-C000-000000000046</wsa:RelatesTo>
  </s:Header>
  <s:Body>
    <wst:GetResponse>
      <wstfr:Status>DistUser<wstfr:Status>
      <wsa:MessageID>tag:yyy@bbb,2017-06-21:xxxx...xx</wsa:MessageID>
      <wsa:Action>http://example.jp/2017/06/ws-tra-fr/StateResponse</wsa:Action>
    </wst:GetResponse>
  </s:Body>
</s:Envelope>
```

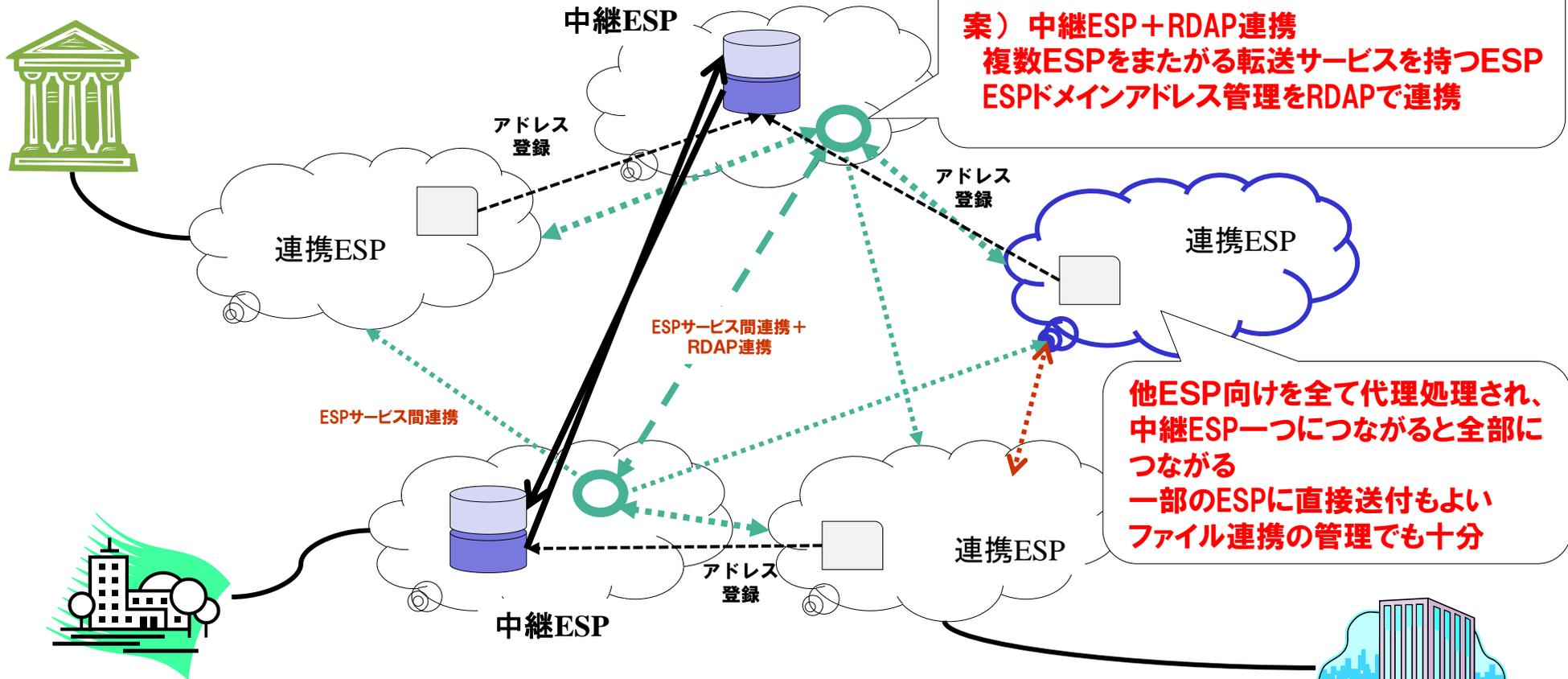
スケジュール(案)



ESPが連携したネットワーク実現の課題



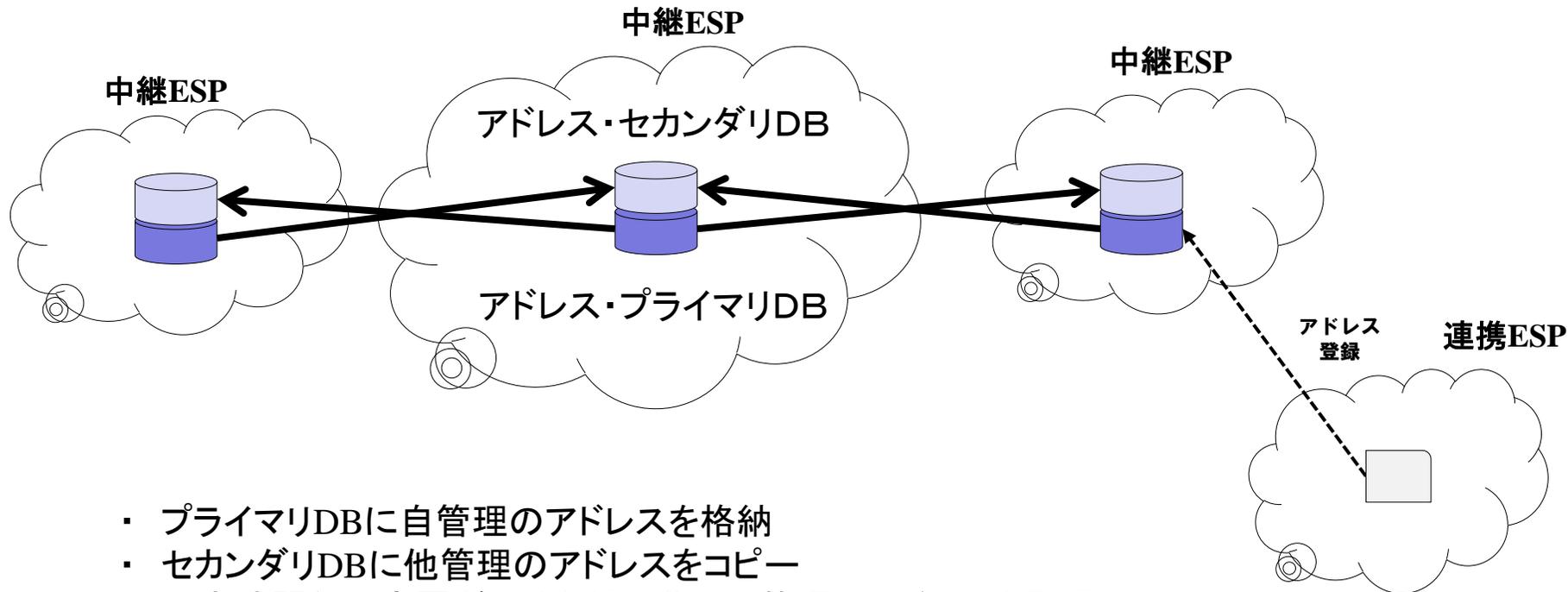
ESPが連携したネットワーク実現に向けて(4)



- 連携ESPは、中継ESPと一部のESPを受付(認証)すればよい
- 連携ESPは、中継ESPが複数あれば、障害時や高負荷時等の代替ルートに使える
- 連携ESPは、ESPドメインアドレス管理のコストが大幅に低減
- 中継ESP間をRDAPで連携し、全てのESPドメインアドレスの管理をシステム化する
- × 中継ESPは、中継するコストが増える
⇒ 中継ESPに依頼する連携ESPが応分の負担をしてはどうか？

ESPドメインアドレス管理

中継ESPのESPドメインアドレス管理も相互分散管理が望ましい



- ・ プライマリDBに自管理のアドレスを格納
- ・ セカンダリDBに他管理のアドレスをコピー
- ・ 一定時間毎に変更が生じた他中継ESP管理のアドレスを取得
- ・ 不明なアドレスは他中継ESPに随時問合せ

注) ESPドメインアドレスとは、ESPのサービスを受付けるサーバーのドメインアドレス

注) アドレス登録は別途検討が必要

検討課題

- ・ 連携アドレス管理
 - 各ESPのユーザーのアカウントアドレスの問合せ方法
 - * 各ESPが管理するアカウントアドレスを他のESP経由でどう問合せするか
- ・ ESP間の認証方法
 - 中継ESPが連携ESPを認証する方法
 - * 原則、中継ESPが認証方法を決定する
 - * 推奨する認証方式を選定するか？
 - 連携ESPが中継ESPを認証する方法
 - * 統一した方が連携ESPに好ましい
 - 中継ESPが他の中継ESPを認証する方法
 - * 統一した方が好ましい
- ・ 既存VANとの連携方法
 - ???