Kommunikationen med eksterne aftagere via webservices - Vejledning

- 1 Indledning
- 2 Generelt om webserviceaftagere af DFDG
  - 2.1 Generelt aftagemønster for aftagere af DFDG
  - 2.2 Endpoints og konfiguration
  - 2.3 Hændelser eller statusændring fra DFDG til aftager via WSRM
  - 2.4 WSRMService (WS-ReliableMessaging)
    - 2.4.1 CreateSequence
      - 2.4.1.1 Ingen beskeder ved kald af CreateSequence
    - 2.4.2 CloseSequence
    - 2.4.3 TerminateSequence
  - 2.5 SequenceAcknowledgement
  - 2.6 Web Service Header elementer
    - 2.6.1 SequenceRequest
    - 2.6.2 SequenceResponse
    - 2.6.3 AckRequested
    - 2.6.4 WSRM Policy
- 3 Sikkerhed
  - 3.1 Krav til web service aftagerne
  - 3.2 Policy og sikkerhedsmodeller.
  - 3.3 Certifikatkommunikation
  - 3.4 Certifikater under TLS
  - 3.5 Serviceaftagers forpligtigelser
  - 3.6 Kodelister
- 4 Eksempel på konfiguration til et testmiljø

1 Indledning

Indeværende dokument beskriver overordnet hvorledes eksterne aftagere kalder DFDG webservices, herunder håndtering af sikkerhed.

2 Generelt om webserviceaftagere af DFDG

Eksterne aftageres adgang til DFDG-Webservices inddeles i tre dele:

1. Hændelser fra DFDG til den eksterne aftager
2. Tilbagemeldinger og registreringer fra den eksterne aftager til DFDG
3. Opslagsservices som den eksterne aftager kan anvende

Hændelser fra DFDG til den eksterne aftager

Alle beskeder til den eksterne aftager udsendes via WSRM kø. Dette er en kø af beskeder som aftagerne selv skal tømme med jævne mellemrum. Hver besked vil indeholde en webservicemetode, der skal kaldes. Den eksterne aftager skal bruge WsrmService for at gennemløbe køen og derefter forskellige beskedservices for hver enkelt webservicemetode.

For brug af WSRM køen se senere i afsnittet.

Tilbagemeldinger og registreringer fra den eksterne aftager til DFDG

Den eksterne aftager vil kunne tilbagemelde og registrere til DFDG ved at kalde en række registreringsmetoder i diverse webservices. Som udgangspunkt kan den eksterne aftager kun foretage registreringer på borgere, der er tilknyttet den pågældende myndighed som aftageren repræsenterer. Det eksempelvis kan være medlemmer af en a-kasse eller en borgere i en kommune.

Opslagsservices som den eksterne aftager kan anvende

Den eksterne aftager vil kunne hente data for personer ved en kalde en række forskellige opslagsmetoder i et antal webservices, herunder, PersonStatusService, PersonHistoryService.

Desuden skal den eksterne aftager kalde CodeListService en gang i døgnet for at hente de aktuelle kodelister, der benyttes på tværs af webservices.

Det udvikles løbende nye, konsoliderede og opdaterede versioner af en række webservices i forbindelse med moderniseringen af det samlede webservicekatalog. Se den samlede dokumentationen her.

Det er op til den enkelte aftager, at sikre begrundelsesstyret adgang til data i det fælles datagrundlag samt gennemføre den nødvendige logning ved opslag i data som er omfattet af kravet herom.

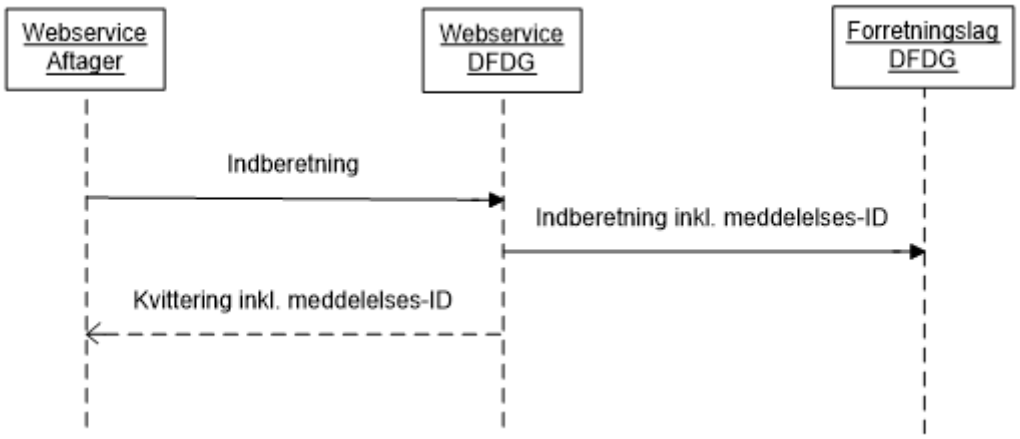
2.1 Generelt aftagemønster for aftagere af DFDG

Et enkelt webservice-kald består af en forespørgsel og et følgende svar sendt over https med SOAP sikret med TLS.

Eksempel i produktion:

https://service.bm.dk/PjakTassWCFtService/9/PersonRegistrationService.svc

Når en aftager skal sende en meddelelse sker det ved at aftageren kalder en af DFDG's webservicemetoder, der som input tager en meddelelse i SOAP/XML-format. Webservicen checker, at formatet af meddelelsen er korrekt i forhold til det XML-Schema, der er defineret for meddelelsestypen. Hvis dette check passeres, checkes om dataindholdet i meddelelsen er validt i forhold til den ønskede forretningsoperation. Herefter udføres forretningsoperationen og et ID (en GUID) returneres til kalder som illustreret i figuren nedenfor.



Figur 1: Kommunikationsmønster for meddelelse fra aftaler til web service

2.2 Endpoints og konfiguration

Endpoint for WCF-T services er https://service.bm.dk/PjakTassWCFtService. I bindings til servicen, skal security mode sættes til Transport:

```
<system.serviceModel>
  <behaviors>
    <endpointBehaviors>
      <behavior name="CommonDFDGServiceBehavior">
        <clientCredentials>
          <clientCertificate storeName="My"
                             storeLocation="LocalMachine"
                             findValue="[Jeres certifikat]"
                             x509FindType="FindBySubjectDistinguishedName"/>
          <serviceCertificate>
            <defaultCertificate storeName="My"
                                storeLocation="LocalMachine"
                                findValue="*.bm.dk"
                                x509FindType="FindBySubjectName"/>
            <authentication certificateValidationMode="None"/>
```

```
        </serviceCertificate>
      </clientCredentials>
    </behavior>
  </endpointBehaviors>
 </behaviors>
 <bindings>
  <basicHttpBinding>
   <binding name="TransportDFDGServiceBinding"
            allowCookies="true"
            bypassProxyOnLocal="true"
            closeTimeout="00:01:00"
            hostNameComparisonMode="StrongWildcard"
            maxBufferPoolSize="65536"
            maxReceivedMessageSize="10000000"
            messageEncoding="Text"
            openTimeout="00:02:00"
            receiveTimeout="00:02:00"
            sendTimeout="00:02:00"
            textEncoding="Utf-8"
            useDefaultWebProxy="true">
     <security mode="Transport">
      <transport clientCredentialType="Certificate" />
     </security>
    </binding>

  </basicHttpBinding>
 </bindings>
 <client>
  <endpoint address="https://service.bm.dk/pjaktasswcftservice/3/WSRMService.svc"
            binding="basicHttpBinding"
            bindingConfiguration="TransportDFDGServiceBinding"
            behaviorConfiguration="CommonDFDGServiceBehavior"
            contract="IWsrmService"
            name="WSRMservice">
    <identity>
     <dns value="bm.dk"/>
    </identity>
  </endpoint>
 </client>
</system.serviceModel>
```

2.3 Hændelser eller statusændring fra DFDG til aftager via WSRM

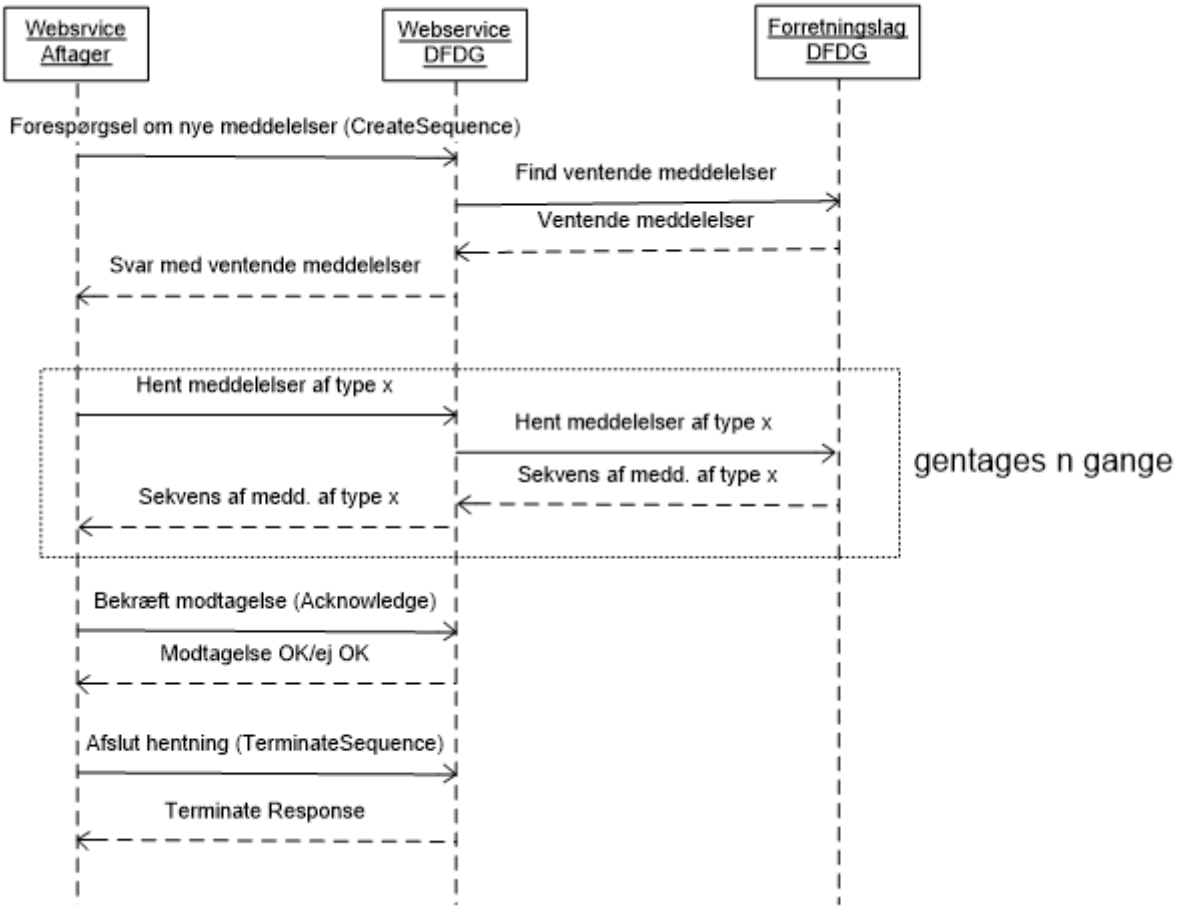
Når der er registreret en hændelse eller et statusskift i DFDG udsendes en notifikation til relevante aftagere.

Hændelser opsamles således ikke pr. dag. Dette betyder derfor også som udgangspunkt at hændelser f.eks. sygdom eller fravær udsendes når det registreres og ikke først når det indtræffer. Aftager skal altså som udgangspunkt selv stå for dette bogholderi.

Notifikationen sendes via webserviceprotokol, der tager afsæt i WSRM – Webservice Reliable Messaging standarden. Køen er implementeret sådan at aftager selv skal tage initiativ til at tømme køen (polle), f.eks. hvert 10. minut. Aftager modtager en liste af webservice endpoints og et operationsnavn, der peger på metoder i WsrmMessageService - typisk af formen GetXXX, hvor XXX er navnet på en forretningshændelse.

Operationen returnerer en liste af hændelsesobjekter af en specifik type. Samlet returneres der p.t. højest 500 objekter ad gangen, uagtet hvor mange hændelser, der ligger i køen.

Hvert objekt er forsynet med en EventDate og den modtagne liste af objekter skal sorteres så ældste objekt behandles først. Denne sortering er meget vigtig da der sagtens kan modtages flere hændelser for samme person.

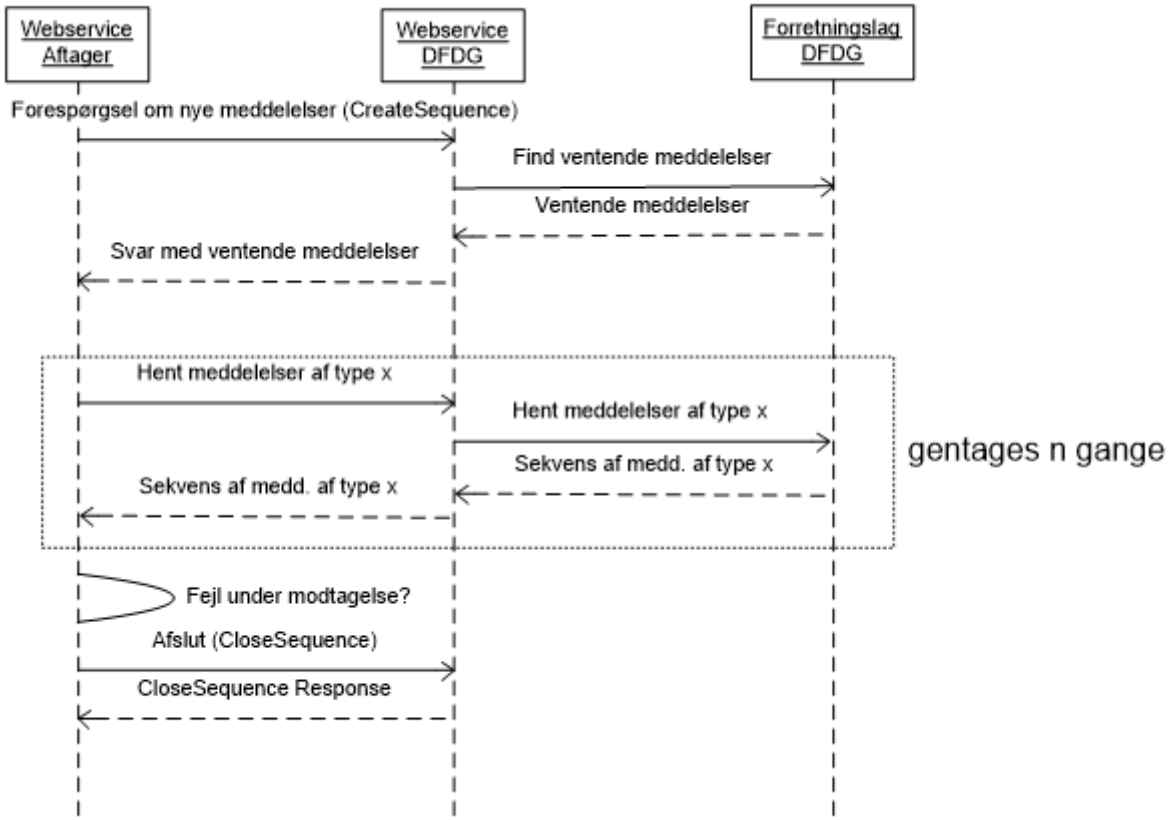


Figur 2: Kommunikationsmønster for meddelelser fra webservice til aftag

Ovenstående viser sekvensen af operationer, der skal til for at tømme køen. Hændelser i f.eks. Jobnet eller jobcenteret, modtages af DFDG og lægges på en kø som en XML besked. Herefter starter ovenstående sekvens.

Når aftageren har hentet alle meddelelser, foretager aftageren derefter et afsluttende kald, som bekræftelse på modtagelse. Hvis DFDG også har registreret alle meddelelser som korrekt afhentet, vil DFDG herefter fjerne de meddelelser, der netop er hentet fra køen.

Hvis der sker fejl under afhentning, kan aftager afbryde under vejs. Hvis der afbrydes under vejs vil beskederne fortsat ligge i køen indtil næste gang aftager forsøger at hente dem.



Figur 3: Kommunikationsmønster for meddelelser fra webservice til aftager ved fejl

2.4 WSRMService (WS-ReliableMessaging)

Følgende afsnit beskriver service elementerne i WS-ReliableMessaging i forhold til denne løsning.

Implementeringen af WS-ReliableMessaging er udarbejdet på baggrund af OASIS specifikationen Web Service Reliable Messaging (WS-Reliable Messaging) – 14 marts 2006.

WSRM gør det muligt for aftagerne at hente og kvittere for modtagelsen af en række beskeder fra flere services på en ensartet måde.

De services, der er en del af WSRM kommunikationen (dvs. de services, der leverer beskeder til aftageren), som er indeholdt i en WSRM Sequence, benytter specifikke WSRM Soap Header beskeder til input og output.

Det er nødvendigt, at aftageren benytter disse headere, da det ellers ikke vil være muligt at benytte WSRM servicen.

WSRM servicen indeholder fire webservicemetoder, der er nødvendige for at implementere WS-Reliable Messaging protokollen, som beskrevet i foregående afsnit.

- **CreateSequence:** initierer en pålidelig overførsel af meddelelser og returnerer en beskrivelse af hvilke meddelelser, der er klar til afhentning.
- **Acknowledge:** bekræftelse på modtagne beskeder fra services.
- **CloseSequence:** bruges til at afbryde en pålidelig overførsel inden normal afslutning. Hele overførslen betragtes som annulleret, og serveren vil derfor ikke slette beskeder, som er blevet sendt i den pågældende overførsel, ligesom aftageren vil få disse sendt beskeder igen ved næste overførsel.
- **TerminateSequence:** afslutter en korrekt gennemført overførsel efter det er konstateret, at alle meddelelser i overførslen er modtaget korrekt.

2.4.1 CreateSequence

CreateSequence metoden benyttes til at initiere en WS-ReliableMessage overførsel af meddelelser og returnerer en beskrivelse af hvilke meddelelser, der er klar til afhentning (serviceinventory).

Serviceaftageren skal derefter kalde alle de services, der er angivet i serviceinventorydelen af CreateSequenceRespons meddelelsen.

Samtlige beskeder i inventory beskeden skal hentes og accepteres i form af en Acknowledgement for hver besked. Det vil ikke være muligt at afslutte WSRM sequencen korrekt med mindre, at aftageren har hentet samtlige beskeder.

Når alle services er kaldt og meddelelserne er accepteret ”Acknowledged” skal ”Sequencen” lukkes via ”TerminatSequence”. Hvis aftageren ikke har hentet og accepteret alle meddelelser i sequencen, kan sequencen ikke termineres, og det er ikke muligt at oprette nye sequences.

Input:

Element	Type
CreateSequence	CreateSequenceType
- AcksTo	EndpointReferenceType
- - Address	AttributedURI Base: anyURI
- - ReferenceProperties	ReferencePropertiesType Base: any
- - - BaseType	any
- - ReferenceParameters	ReferenceParametersType Base: any
- - - BaseType	any
- - PortType	AttributedQName Base: QName
- - ServiceName	ServiceNameType Base: QName
- Expires	Expires
- Offer	OfferType
- - Identifier	guid Base: string
- - Expires	Expires
- SubQueueIdentifier	int

Xml eksempel:

```
<wsrm:CreateSequence>
  <wsrm:SubQueueIdentifier> ... </wsrm:SubQueueIdentifier>
</wsrm:CreateSequence>
```

Output:

Element	Beskrivelse
Identifier	Unikt sekvens-id, der skal benyttes i forbindelsen med al øvrig kommunikation inden for denne sequence. Format: GUID

Inventory	Extension element, som ikke er indeholdt i WS-RM. Består af underelementer: <Inventory> <Service name=”UnemployedRegistration” count=”2”/> <Service name=”Appropriation” count=”1”/> </Inventory> Dette eksempel fortæller, at der venter to meddelelser af typen ”UnemployedRegistration” og én af typen ”Appropriation”. Der kan være ligeså mange <Service> tags som der er service metoder til afhentning af meddelelser.

Xml eksempel:

```
<wsrm:CreateSequenceResponse>
  <wsrm:Identifier> .. </wsrm:Identifier>
  <ams:Inventory>
    <ams:Service name=”UnemployedRegistration” count=”2”/>
    <ams:Service name=”Appropriation” count=”1”/>
  </ams:Inventory>
</wsrm:CreateSequenceResponse>
```

2.4.1.1 Ingen beskeder ved kald af CreateSequence

Hvis kaldet til CreateSequence returnerer en ServiceInventoryCollection med 0 ServiceItems, er der ikke nogen WSRM beskeder at hente for den kaldende aftager.

Sekvensen med Identifier som identifikation lukkes i denne situation automatisk af DFDG, og afhentning kan startes forfra med CreateSequence.

2.4.2 CloseSequence

CloseSequence: bruges til at afbryde en pålidelig overførsel inden normal afslutning. Hele overførslen betragtes som annulleret, og serveren vil derfor ikke slette beskeder, som er blevet sendt i den pågældende overførsel, ligesom aftageren vil få disse sendt beskeder igen ved næste overførsel.

Input:

Element	Beskrivelse
Identifier	Unikt sekvens id, leveret som output af kaldet til ”CreateSequence” Format: GUID

Xml eksempel:

```
<wsrm:CloseSequence>
  <wsrm:Identifier> .. </wsrm:Identifier>
</wsrm:CloseSequence>
```

Output:

Element	Beskrivelse
Identifier	Unikt sekvens-id leveret som output af kaldet til ”CreateSequence” Format: GUID

Xml eksempel:

```
<wsrm:CloseSequenceResponse>
  <wsrm:Identifier> .. </wsrm:Identifier>
</wsrm:CloseSequenceResponse>
```

2.4.3 TerminateSequence

TerminateSequence afslutter en korrekt gennemført overførsel efter det er konstateret, at alle meddelelser i overførslen er modtaget korrekt.

Input:

Element	Beskrivelse
Identifier	Unikt sekvens-id leveret som output af kaldet til ”CreateSequence” Format: GUID

Xml eksempel:

```
<wsrm:TerminateSequence>
  <wsrm:Identifier> .. </wsrm:Identifier>
</wsrm:TerminateSequence>
```

Output:

Element	Beskrivelse
Identifier	Unikt sekvens-id leveret som output af kaldet til ”CreateSequence” Format: GUID
TerminateSequenceCorrect	Servicens svar på bekræftelsen i input. Hvis bekræftelsen var korrekt, returneres ”true”, ellers ”false”. Format: boolean

MissingRange	Dette element angiver hvilke sekvensnumre, som er afsendt, men ikke kvitteret for. Elementet er kun til stede hvis AckCorrect=false. Format: <MissingRange Lower=”2”Upper=”4”/> for sammenhængende interval, eller: <MissingRange Lower=”2”Upper=”3”/> <MissingRange Lower=”6”Upper=”6”/> ... for ikke-sammenhængende intervaller.

Xml eksempel:

```
<wsrm:TerminateSequenceResponse>
  <wsrm:Identifier> .. </wsrm:Identifier>
  <wsrm:TerminateSequenceCorrect> .. </wsrm:TerminateSequenceCorrect>
  <wsrm:MissingRangeCollection>
    <wsrm:MissingRange Upper="int" Lower="int" />
    <wsrm:MissingRange Upper="int" Lower="int" />
  </wsrm:MissingRangeCollection>
</wsrm:TerminateSequenceResponse >
```

2.5 SequenceAcknowledgement

SequenceAcknowledgement sendes som bekræftelse på modtagne beskeder.

Input:

Element	Beskrivelse
Identifier	Unikt sekvens-id leveret som output af kaldet til ”CreateSequence” Format: GUID
AcknowledgementRange *eller* None	Angivelse af modtagne meddelelser på aftagersiden. None angiver at ingen meddelelser er modtaget. Er der modtaget mindst én, angives det på følgende måde: <AcknowledgementRange Lower=”1”Upper=”3”> som fortæller, at meddelelser med sekvensnumre 1, 2, 3 er modtaget. For ikke sammenhængende intervaller, f.eks. hvis sekvensnumre 1,3,4 er modtaget, bruges flere AcknowledgementRange: <AcknowledgementRange Lower=”1”Upper=”1”> <AcknowledgementRange Lower=”3”Upper=”4”>

Xml eksempel:

```
<wsrm:SequenceAcknowledgement>
  <wsrm:Identifier> .. </wsrm:Identifier>

  <AcknowledgementRange Lower=”1”Upper=”1”>

  <AcknowledgementRange Lower=”3”Upper=”4”>

</wsrm:SequenceAcknowledgement>
```

Output:



Vigtigt!:
Denne besked er ikke defineret i WS-RM, men er nødvendig i det aktuelle setup, hvor meddelelser sendes modsat i fo

Element	Beskrivelse
Identifier	Unikt sekvens-id leveret som output af kaldet til ”CreateSequence” Format: GUID
AckCorrect	Servicens svar på bekræftelsen i input. Hvis bekræftelsen var korrekt, returneres ”true”, ellers ”false”. Format: boolean

Xml eksempel:

```
<wsrm:SequenceAcknowledgementResponse>
  <wsrm:Identifier> .. </wsrm:Identifier>

  <wsrm:AckCorrect> .. </wsrm:AckCorrect>

</wsrm:SequenceAcknowledgementResponse>
```

2.6 Web Service Header elementer

De web services, der kræver WS-RM, benytter en række SoapHeader blokke, der benyttes til at formidle information relateret til WS-RM sekvensen

2.6.1 SequenceRequest

SequenceRequest header elementet skal sendes med i Soap Headeren til services, der kræver WS-ReliableMessaging

Xml eksempel:

```
<wsrm:SequenceRequest>
  <wsrm:Identifier> .. </wsrm:Identifier>

</wsrm:SequenceRequest>
```

2.6.2 SequenceResponse

SequenceResponse header elementet sendes af de services, der er den del af WS-ReliableMessaging sekvensen.

Header elementet indeholder information om det messagenumber, der skal benyttes til Acknowledgement meddelelsen.

Identifier-elementet og Messagenumber-elementet skal benyttes, når Acknowledgement meddelelsen skal sendes retur til WSRM servicen.

Xml eksempel:

```
<wsrm:SequenceResponse>
  <wsrm:Identifier> .. </wsrm:Identifier>

  <wsrm:MessageNumber> .. </wsrm:MessageNumber>

</wsrm:SequenceResponse>
```

2.6.3 AckRequested

AckRequested header elementet sendes af de services, der er den del af WS-ReliableMessaging sekvensen.

Elementet fortæller serviceaftageren, at der skal sendes en Acknowledgement for modtagelsen af meddelelsen, så snart meddelelsen er modtaget.

Xml eksempel:

```
<wsrm:AckRequested>
  <wsrm:Identifier> .. </wsrm:Identifier>

  ..

</wsrm:AckRequested>
```

2.6.4 WSRM Policy

Følgende beskriver WS-RM policyen for alle de services der benytter WSRM.

- ***Max Antal gensendelser: 3***
  Det maksimale antal accepterede gensendelser er 3, hvis der herefter ikke er modtaget en Ack, termineres sekvensen.
- ***Der skal sendes en Acknowledgement for alle beskeder***
  Der skal sendes en Acknowledgement for alle beskeder i en sequence før WSRM sekvensen kan afsluttes.
- ***Alle beskeder i WSRM service inventory skal hentes***
  Alle beskeder i WSRM service inventory beskeden skal hentes af aftageren, det er ikke muligt at afslutte en sequence, hvor ikke alle beskeder er hentet.

3 Sikkerhed

Følgende afsnit beskriver de overordnede informationer i forhold til sikkerhed og de i dette dokument beskrevne web services.

3.1 Krav til web service aftagerne

Webservice aftageren skal etablere og opretholde fornødne sikkerhedsmæssige tiltag til sikring af, at meddelelser, der udveksles via Webservices, ikke kommer til uvedkommendes kendskab, forvanskes eller går til grunde.

Det er web service aftagernes eget ansvar at sikre, at dennes IT-systemer er konfigureret og eventuelt tilrettet i nødvendigt omfang til at kunne få adgang til de i dette dokument beskrevne Web services.

Ligeledes er web service aftagernes ansvar at sikre, at der er knyttet forsvarlige sikkerhedsforanstaltninger til de applikationer og systemer, denne benytter for at kunne anvende de i dette dokument beskrevne Webservices.

Yderligere specifikation af de krav der stilles til web service aftageren findes i tilslutningsaftalen for de i dette dokument beskrevne services.

3.2 Policy og sikkerhedsmodeller.

Sikkerheden i de udarbejdede web services er baseret på brugen af OCES-certifikater og understøtter en sikkerhedsmodel **Sik_3** baseret på funktionscertifikater.

Der tilbydes adgang til TLS via https (transportbaseret sikkerhed).

3.3 Certifikatkommunikation

Generelt baserer datakommunikation, der er sikret med certifikater, sig på, at både afsender og modtager har et certifikat og at hvert certifikat består af en privat og en offentlig del. Den private del udleveres aldrig til andre, den offentlige del udleveres til parter, man ønsker at kommunikere sikkert med.

STAR stiller et certifikat til rådighed pr. miljø (udviklingstestmiljø, produktionsmiljø), aftager ønsker at kommunikere sikkert med.

Aftager skal tilsvarende anskaffe sig et mindst et OCES-certifikat via Nets. I langt de fleste tilfælde vil en aftager have en lokal certifikatadministrator (LRA) som kan udstede certifikater på vegne af Nets. For alle certifikater, uanset typen, gælder at de skal være udstedt under samme CVR-nummer som oplyst i tilslutningsaftalen med STAR.

Der skal bestilles et funktionscertifikat pr. myndighed dvs. een myndighed skal bruge netop et funktionscertifikat. Leverandører med flere myndigheder som kunder skal således også bruge et funktionscertifikat pr. myndighed.

Forudsætningen for kunne anvende de bestilte certifikater i forbindelse med adgang til Arbejdsmarkedsportalens webservices er:

- Aftager skal have etableret tilslutningsaftale med STAR.
- Aftager skal have godkendt samtlige certifikater hos STAR. Dette sker ved fremsendelse (pr e-mail) af de offentlige dele af certifikaterne til systemforvalter. Hvis der udelukkende er tale om medarbejdercertifikater kan godkendelsen også ske ved at forsøge pålogning på http://amportal.bm.dk

Aftager skal bruge:

- Et eller flere funktionscertifikater idet der som minimum skal anvendes et funktionscertifikat pr aftagende system($F_{1...}$ F_n)
- Styrelsen for arbejdsmarked og rekrutterings offentlige servercertifikat til det miljø, hvorfra der skal aftages webservices, f.eks. service.bm.dk ($STAR_1$)

Aftager skal:

- Importere/installere ($STAR_1$) i sit lokale Certificate Store, så det kan ses af det system, der skal foretage forespørgslen.
- Kryptere sin forespørgsel med ($STAR_1$)
- Signere sin forespørgsel med den private del af (F_x)
- Afsende sin forespørgsel mod det pågældende miljø og webservice.

Aftager vil modtage et svar:

- Krypteret med (F_x)
- Signeret med ($STAR_1$)

Aftager skal dekryptere svaret med den private del af F_x

Ved anvendelse af funktionscertifikater til webservice kald på vegne af en bruger skal der medsendes et metadataobjekt i SOAP-beskeden, der unikt identificerer, hvilken bruger der har udløst forespørgslen:

Metadatamodellen er baseret på følgende oplysninger:

- User Identifier - Unik bruger identifikation
- Local User Identifier - Lokal bruger identifikation (optionel)
- UserName (GivenName, MidleName, SurName) - Bruger navn (baseret på OIO XML navne standar-den)
- User Email - Bruger e-mail adresse (optionelt)

3.4 Certifikater under TLS

Som udgangspunkt anvendes samme certifikater som beskrevet i det foregående afsnit, Certifikatkommunikation. Forskellen er, at certifikaterne ikke anvendes til kryptering og signering af de enkelte forespørgsler, men i stedet til 1) sikring af en TLS-forbindelse som forespørgsler og svar kan sendes over 2) til logning og autentificering på webservicesiden af aftager.

Aftager skal:

- Importere/installere (AMS_1) som trusted root certificate, så det kan ses af det system, der skal foretage forespørgslen.
- Installere aftagers certifikat (F_x) pr. TLS-forbindelse, der skal opsættes.
- Sende sin forespørgsel i klartekst mod det pågældende miljø og webservice.
- Aftager vil modtage et svar i klartekst.

3.5 Serviceaftagers forpligtigelser

Følgende beskriver de forpligtelser serviceaftageren påtager sig i forbindelse med en hver form for brug af de i dette dokument beskrevne services.

Serviceaftageren forpligter sig til følgende:

- Serviceaftageren er forpligtet til at logge de kvitteringer der returneres af servicen. Loggen skal opbevares i 6 måneder hvorefter den skal slettes.
- Ved anvendelsen af metadatamodellen er det aftager systemet der er ansvarlig for authentication og authorisation af den enkelte bruger, samt at sikre at den enkelte bruger alene kan tilgå data i henhold til kravene i persondata-loven.
- Serviceaftageren forpligter sig til at logge alle SOAP faults incl. fejl koden som en del af deres fejl log.

3.6 Kodelister

Mange af webservicekaldene anvender lister, det kan være jobcentre, a-kasser, klientkategori eller lign., som en del af request eller response parametrene. For ikke at skulle opdatere WSDL grænsefladen hver gang, der ændres i disse lister, er der oprettet en række kodelister med disse værdier som udbydes parallelt med de services, der anvender dem.

Kodelisterne er etableret som selvstændige services, som de aftagende systemer skal kalde for at hente gældende kodelister for enkelte værdier.

Kodelisterne skal max. hentes en gang i døgnet, og skal derfor ikke kaldes i forbindelse med hver request til en service. Det er serviceaftagers ansvar til en hver tid at anvende de aktuelle kodelister. Ændringer i kodelister varles i hh. varslingsfrister i webserviceaftalerne.

Det er aftagerapplikationens opgave at udstille kodelisterne til brugerne på en sigende måde. Det kan således ikke i alle tilfælde anbefales at benytte beskrivelserne direkte i aftagerapplikationerne i forbindelse med indberetninger, da beskrivelserne kan indeholde referencer til felter i webservicebeskeden, som ikke modsvarer en semantik eller begrebsmodel, der giver mening for brugeren af den enkelte applikation. Det tilstræbes dog at kolonnen ”Name” i kodelisten udtrykker et for brugeren sigende navn for kodeværdien.

Der er start- og slutdatoer for alle kodeværdier. Der må kun registreres kodeværdier, der er aktuelle - det vil sige, hvor slutdatoen er mindre end dags dato. Hvis slutdatoen er overskredet skal værdien kun bruges ved læsning af gamle værdier og eventuelle opdateringer af forhold der allerede eksisterer.

Kodelisterne er opbygget efter en fælles struktur:

Element	Beskrivelse
Identifier	Identifier elementet der angiver den id der skal benyttes til servicen for dette kode liste element. **Type:** int
Name	Et sigende navn for kodeværdien. Min. Length: 1 Max. Length: 100 **Type:** string
Description	En kort beskrivelse af kodes betydning Min. Length: 1 Max. Length: 500 **Type:** string

StartDate	Dato fra hvornår kodeværdien gælder **Format:** xsd:dateTime
EndDate	Dato for hvornår kodeværdien udløber **Format:** xsd:dateTime

4 Eksempel på konfiguration til et testmiljø

Eksempel på konfiguration til T10 testmiljø:

```
<system.serviceModel>
  <behaviors>
    <endpointBehaviors>
      <behavior name="CommonDFDGServiceBehavior">
        <clientCredentials>
          <clientCertificate storeName="My"
                             storeLocation="LocalMachine"
                             findValue="CVR:55568510-RID:9100000000023"
                             x509FindType="FindBySubjectDistinguishedName"/>
          <serviceCertificate>
            <defaultCertificate storeName="My"
                                storeLocation="LocalMachine"
                                findValue="*.startest.dk"
                                x509FindType="FindBySubjectName"/>
            <authentication certificateValidationMode="None"/>
          </serviceCertificate>
        </clientCredentials>
      </behavior>
    </endpointBehaviors>
  </behaviors>
  <bindings>
    <basicHttpBinding>
      <binding name="TransportDFDGServiceBinding"
               allowCookies="true"
               bypassProxyOnLocal="true"
               closeTimeout="00:01:00"
               hostNameComparisonMode="StrongWildcard"
               maxBufferPoolSize="65536"
               maxReceivedMessageSize="10000000"
               messageEncoding="Text"
               openTimeout="00:02:00"
               receiveTimeout="00:02:00"
               sendTimeout="00:02:00"
               textEncoding="Utf-8"
```

```
            useDefaultWebProxy="true">
        <security mode="Transport">
          <transport clientCredentialType="Certificate" />
        </security>
      </binding>

    </basicHttpBinding>
  </bindings>

  <client>
    <endpoint address="https://servicet10.amstest.dk/pjaktasswcftservice/18/PersonStatusService.svc"
              binding="basicHttpBinding"
              bindingConfiguration="TransportDFDGServiceBinding"
              behaviorConfiguration="CommonDFDGServiceBehavior"
              contract="IWsrmService"
              name="WSRMservice">
      <identity>
        <dns value="*.startest.dk"/>
      </identity>
    </endpoint>
  </client>
</system.serviceModel>
```