Javaklient for Altinns webservicer. Utviklet for formidlingstjenestene

Referanser

Nr	Dokumentnavn	Forklaring
1	https://altinn.github.io/docs/guides/sluttbrukersystem/	Implementasjonsguide for sluttbrukersystemer
2	https://altinn.github.io/docs/guides/tjenesteeier/	Implementasjonsguide for tjenesteeier

Definisjoner

Betegnelse	Beskrivelse
Avgiver	Rolle i forbindelse med formidlingstjenesten: Den person eller virksomhet (offentlig eller privat) som avgir informasjon til formidlingstjenesten.
Mottaker	Rolle i forbindelse med formidlingstjenesten: Den person eller virksomhet (offentlig eller privat) som mottar informasjon fra formidlingstjenesten.
Tjenesteeier	Rolle i forbindelse med formidlingstjenesten: Den offentlige virksomhet som er ansvarlig for en formidlingstjeneste og gir tilganger til avgivere og mottakere.
Reportee	Ved instansiering og opplasting til formidlingstjenesten settes det til avgiver. Ved søking etter tilgjengelige formidlinger settes det til mottaker.
Receipt	Kvittering. Altinn ivaretar kvittering for hver formidling gjennom formidlingstjenesten. Dette gir sporbarhet for både avgiver, mottaker og tjenesteeier. Kvitteringer slettes ikke fra Altinn selv om øvrige formidlingsdata kan være slettet ved at formidling er fullført. Kvitteringene kan oppdateres i ettertid av både avgiver og mottaker.
SRR	Det tjenesteeierstyrte rettighetsregisteret. Dette register kan brukes til å bestemme for hver formidlingstjeneste hvilke enheter angitt ved personnummer/organisasjonsnummer som har lese/skrive (motta/avgi) rettigheter.
payload	Essensen av en formidling bestående av en komprimert datafil som skal formidles fra avgiver til mottaker. Altinn gjør ingen endring eller validering av innholdet i filen utenom en virusskanning.

Innledning

Dette er dokumentasjon for en Java referanseklient som integrerer med Altinn formidlingstjenesten. Formidlingstjenesten er tilgjengelig som et sett av webservicer. Java klienten har til hensikt å gi et enkelt eksempel på hvordan integrasjon mellom et sluttbrukersystem og Altinn formidlingstjenesten kan gjøres. Siden fokus er på bruk av Altinn webservices søkes det å unngå bruk av tredjeparts bibliotek og rammeverk som

ikke er direkte relevant for integrasjonen. Det legges dessuten vekt på å gi informasjon av teknisk karakter som kan være av verdi for en utvikler av sluttbrukersystem som skal integreres mot Altinn sin webservice.

Altinn formidlingstjenesten har i likhet med øvrige webservices som Altinn tilbyr 3 ulike nivå sikkerhet (ref /1/ avsnitt 5.1 og 8.3):

1. **Basic** (SOAP 1.1) Tradisjonell web service
2. **WS-Security** (SOAP 1.2 med WS-Security username token) Støtte for nye web service standarder WS*.
3. **Enterprise Certificate** (SOAP 1.2 med WS-Security X.509 token) Støtte for nye web standarder WS*. Sertifikat ligger i SOAP headeren mens brukernavn og passord ligger i meldingen.

I dette dokumentet beskrives integrasjon mot Altinn webservice basert på det høyeste sikkerhetsnivå: nivå 3 Enterprise Certificate.

Selv om det er formidlingstjenestene som er hovedfokus for denne klienten, har vi utvidet klienten med metoder for å sende inn innsendingstjenester til Altinn. Årsaken er at klienten kan da benyttes som eksempel som utvikler i Java.

Teknisk guide

Java Cryptography Extension

Altinn bruker en type kryptering som går ut over det som standard Java installasjon tilbyr. Det er derfor nødvendig å laste ned en utvidelse til Java installasjonen som gjør det mulig med ubegrenset styrke på kryptering. Avhengig av Java versjon kan den nødvendige utvidelse lastes ned fra www.oracle.com:

- Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 6
- Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 7 Download
- Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 8 Download

Nedlastingen er en zip fil som inneholder to jar filer, local_policy.jar og US_export_policy.jar. Disse må erstatte tilsvarende filer i lib\security katalogen for **både** JRE og JDK katalogen, for eksempel:

- C:\Program Files\Java\jdk1.8.0_121\jre\lib\security
- C:\Program Files\Java\jre1.8.0_121\lib\security

Enterprise Certificate

Gitt nivå 3 for sikkerhet er det en forutsetning for tilgang til Altinn formidlingstjenesten at man har et "Enterprise Certificate" for virksomheten. Dette kan for eksempel være utgitt av Commfides eller BuyPass. I dette eksempel er det tatt utgangspunkt i sertifikat på format pkcs12 (file suffix *.p12).

Med en sertifikatfil og tilhørende passord kan man bruke flere alternative verktøy til å bekrefte at sertifikatet er gyldig. Av spesiell interesse er attributtene *valid from*, *valid to* og *friendly name*. Den sistnevnte attributt må angis som *keystore.alias* i java klienten.

Her er eksempel på kommando med verktøyet openssl (kommer med Cygwin standardinstallasjon):

```
$ openssl pkcs12 -in COMMFIDES_BAREKSTAD_OG_YTTERVÅG_REGNSKAP.p12 -info
```

```
Enter Import Password:
MAC Iteration 2048
MAC verified OK
PKCS7 Encrypted data: pbeWithSHA1And40BitRC2-CBC, Iteration 2048
Certificate bag
Bag Attributes
    localKeyID: 7B 6A EC 05 86 75 86 D2 39 3B 0B 58 4F 39 5F 16 1C 6B DB 91
    friendlyName: Authentication certificate
subject=/CN=BAREKSTAD OG YTTERV\xC3\x85G
REGNSKAP/serialNumber=810514442/O=BAREKSTAD OG YTTERV\xC3\x85G REGNSKAP -
810514442/                                    L=Carl Kjelsens vei 20, 0874
Oslo/C=NO
issuer=/CN=Commfides CPN Enterprise-Norwegian SHA256 CA - TEST/OU=Commfides
Trust Environment(C) 2014 Commfides Norge AS - TE
ST/OU=CPN Enterprise-Norwegian SHA256 CA- TEST/O=Commfides Norge AS - 988 312
495/C=NO
-----BEGIN CERTIFICATE-----
MIIGWTCCBUGgAwIBAgIIDBHVTZoqbWMwDQYJKoZIhvcNAQELBQAwgfExPDA6BgNV
BAMTM0NvbW1maWRlcyBDUE4gRW50ZXJwcmlzZS1Ob3J3ZWdpYW4gU0hBMjU2IENB
IC0gVEVTVDFGMEQGA1UECxM9Q29tbWZpZGVzIFRydXN0IEVudmlyb25tZW50KEMp
IDIwMTQgQ29tbWZpZGVzIE5vcmdlIEFTIC0gVEVTVDExMC8GA1UECxMoQ1BOIEVu
dGVycHJpc2UtTm9yd2VnaWFuIFNIQTI1NiBDQS0gVEVTVDEpMCcGA1UEChMgQ29t
bWZpZGVzIE5vcmdlIEFTIC0gOTg4IDMxMiA0OTUxCzAJBgNVBAYTAk5PMB4XDTE2
```

Tilsvarende kan man bruke Java keytool (skal finnes på %JAVA_HOME%\bin):

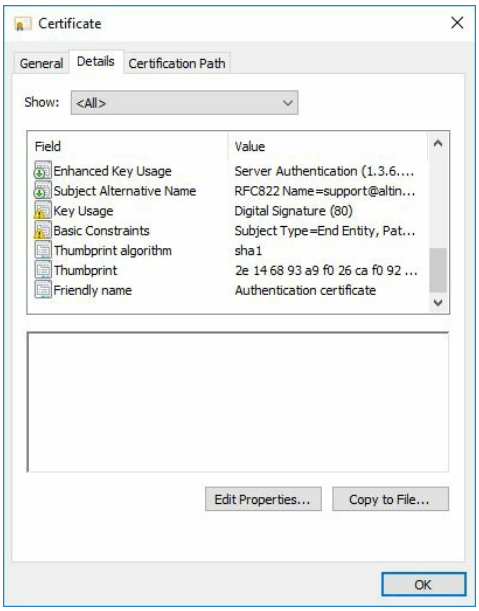
```
d:\>keytool -list -keystore TORNES_I_ROMSDAL_OG_BJERKA_REGNSKAP.p12 -storepass
test123 -storetype PKCS12 -v

Keystore type: PKCS12
Keystore provider: SunJSSE

Your keystore contains 1 entry

Alias name: authentication certificate
Creation date: 15.mar.2017
Entry type: PrivateKeyEntry
Certificate chain length: 3
Certificate[1]:
Owner: C=NO, L="Berggata 20, 1606 Fredrikstad", O=TORNES I ROMSDAL OG BJERKA
REGNSKAP - 910514350, SERIALNUMBER=910514350, CN=TORNES I ROMSDAL OG BJERKA
REGNSKAP
Issuer: C=NO, O=Commfides Norge AS - 988 312 495, OU=CPN Enterprise-Norwegian
SHA256 CA- TEST, OU=Commfides Trust Environment(C) 2014 Commfides Norge AS -
TEST, CN=Commfides CPN Enterprise-Norwegian SHA256 CA - TEST
Serial number: 4420af8342a040f2
Valid from: Thu Oct 20 00:00:00 CEST 2016 until: Wed Oct 20 23:59:59 CEST 2021
Certificate fingerprints:
         MD5:  A7:C4:3E:AB:50:E1:F6:30:4D:F4:49:32:CC:72:3B:16
         SHA1: 2E:14:68:93:A9:F0:26:CA:F0:92:6A:44:C6:36:1C:64:F2:54:72:F0
```

De to nevnte kommandolinjebaserte verktøy har også muligheter til å omskrive et sertifikat fra en form til en annen eller gjøre visse endringer i sertifikatet. For eksempel sette eller endre attributten *friendly name* (aka. alias):

```
keytool -importkeystore -srckeystore C:\Temp\certs\registerenheten.pfx -
srcstoretype pkcs12 -srcstorepass test123 -srcalias {31dae1eb-d2e7-4b1c-a29a-
478c45db8a20} -destkeystore c:\temp\cert\output.jks -deststoretype jks -
deststorepass test123 -destalias bronno
```

Vi anbefaler å ikke bruke ÆØÅ i aliasene/filnavnene.

En annen mulighet er å importere sertifikatet på lokal Windows PC ved å høyreklikke på sertifikatfilen og velge *install PFX* og følge *certificate import wizard*. Velg *store location* som *current user* og åpne *Microsoft management console* **mmc**. I mmc velg *File->add/remove snap ins*, *Certificates* og velg *My user account*. Sertifikatet skal nå være å finne under *Personal->Certificates*. Høyreklikk sertifikatet i listen og velg *open* og deretter *details*:



SoapUI

Under arbeid med en ny integrasjon mot Altinn formidlingstjenesten kan det være et nyttig første steg å gjøre noen enklere tester for å se at man oppnår kontakt med tjenesten og at den tildelte testbruker har de nødvendige tilganger til den formidlingstjenesten man skal bruke. Til dette formål kan verktøyet SoapUI brukes. Gratisversjonen av SoapUI er tilstrekkelig i denne sammenheng.

Til et slikt første steg bruker man da Altinn tjenestene på sikkerhetsnivå *basic*. Man oppretter et nytt prosjekt i SoapUI og starter med å importere wsdl filen for en av tjenestene, ref. /1/ avsnitt 8.3 og etterfølgende oversikt over aktuelle tjenester og wsdl filer. For enkelhets skyld, her er lenker til de relevante wsdl filer for bruk med SoapUI:

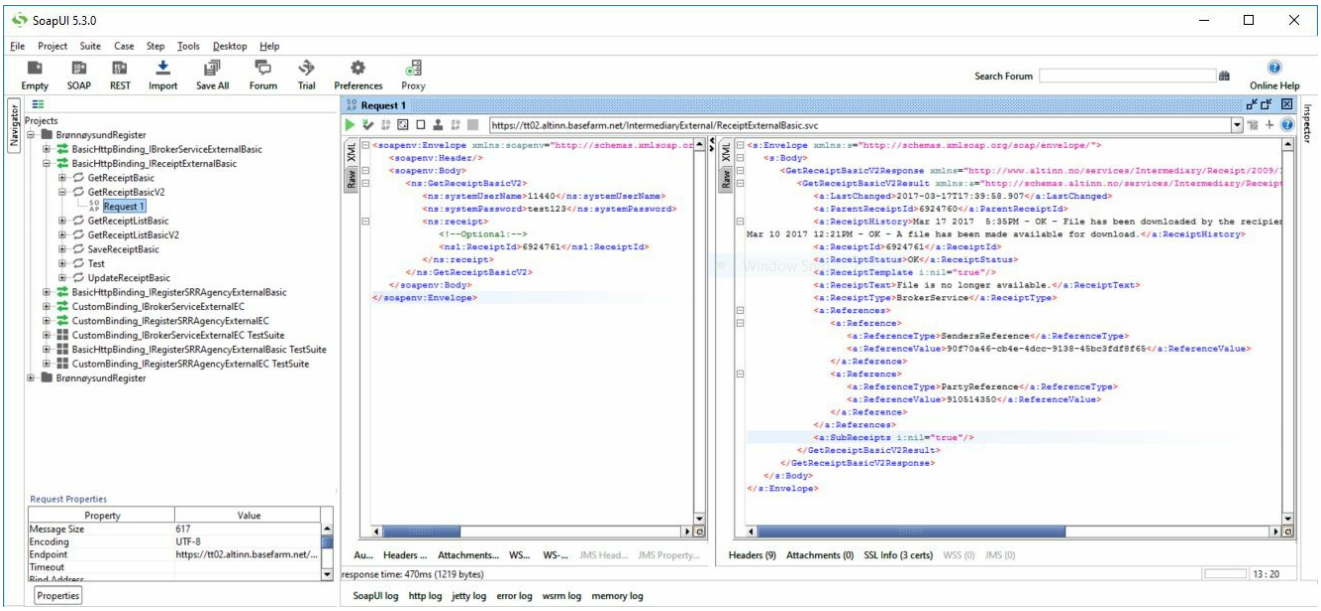
- https://tt02.altinn.no/ServiceEngineExternal/BrokerServiceExternalBasic.svc?wsdl

- https://tt02.altinn.no/ServiceEngineExternal/BrokerServiceExternalStreamedBasic.svc?wsdl
- https://tt02.altinn.no/IntermediaryExternal/ReceiptExternalBasic.svc?wsdl

Det er viktig å være oppmerksom på at etter import av wsdl fil til sitt SoapUI prosjekt så må URI for hvert service endpoint endres. Man må huske å endre fra **http** til **https**. Av tekniske grunner knyttet til Altinn infrastruktur (load balancing) vil hver wsdl angi service endpoint med feil protokoll.

Vi gjør oppmerksom på at de ulike tjenestene i Altinn har forskjellige sikkerhetsnivåer. For eksempel kan det til en spesiell innsendingstjeneste være tilstrekkelig å logge inn med passord, mens det for en annen meldingstjeneste er påkrevd med autentisering på linje med BankId. Vi har ennå ikke lyktes med å benytte SoapUI med virksomhetssertifikater, så her kan det by på en utfordring å få testet "sin" formidlingstjeneste med SoapUI siden den krever sikkerhetsnivå 3.

Her er et eksempel på bruk av SoapUI mot Altinn webservice:



Webservice oversikt

Dokumentet /1/ gir en oversikt over alle Altinn webservices som er tilgjengelige for integrasjon med sluttbrukersystemer. Et subsett av disse webservices er relevante for bruk med Altinn formidlingstjenesten.

For formidlingstjenesten er det først og fremst webservices som er beskrevet i /1/ avsnitt 6.14:

Webservice	**Beskrivelse**
BrokerServiceExternalEC.InitiateBrokerService	Initialisering av formidlingstjenesten.
BrokerServiceExternalEC.GetAvailableFiles	Hent ut tilgjengelige filer fra formidlingstjenesten.
BrokerServiceExternalEC.ConfirmDownloaded	Bekrefte at fil er nedlastet fullstendig.
BrokerServiceExternalECStreamed.UploadFileStreamed	Opplasting av filer til formidlingstjenesten.
BrokerServiceExternalECStreamed.DownloadFileStreamed	Nedlasting av filer fra formidlingstjenesten.

I tillegg er følgende webservices relevant for å hente ut eller oppdatere kvittering fra formidlingstjenesten. Beskrevet i /1/ avsnitt 6.4:

Webservice	Beskrivelse
ReceiptExternalEC.GetReceiptV2	Hente kvittering fra formidlingtjenesten.
ReceiptExternalEC.GetReceiptListV2	Hente alle kvitteringer tilhørende en kvitteringstype og/eller fra et gitt tidsrom
ReceiptExternalEC.UpdateReceipt	Oppdatere en kvittering, for eksempel ved mottak av data fra formidlingstjenesten.

WSDL filer

Wsdl for de nevnte tjenester er tilgjengelig på følgende URI:

- https://tt02.altinn.no/ServiceEngineExternal/BrokerServiceExternalEC.svc?wsdl
- https://tt02.altinn.no/ServiceEngineExternal/BrokerServiceExternalECStreamed.svc?wsdl
- https://tt02.altinn.no/IntermediaryExternal/ReceiptExternalEC.svc?wsdl

De angitte adresser er til Altinn testmiljø tt02. For produksjon byttes tt02.altinn.no med www.altinn.no.

Altinn webservices er utviklet med Microsoft .Net teknologi. Ved arbeid med Java klient mot disse webservices er det avdekket noen tekniske problemer som kan omgås ved å modifisere wsdl filen for Altinn tjenester. Den medfølgende Java referanseklienten har derfor kopier av Altinn wsdl filer med noen mindre endringer.

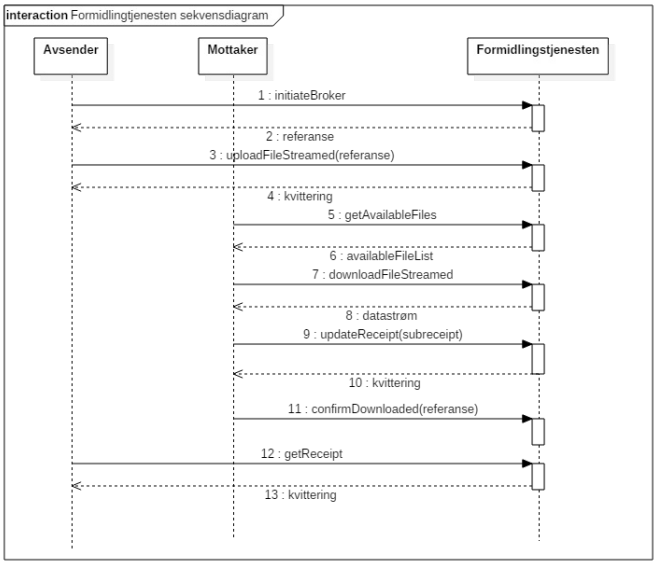
Ett slikt wsdl problem er en konflikt i forbindelse med hvordan WS-SecurityPolicy «RequireDerivedKeys» håndteres på. I Altinn konfigurasjonen er denne attributten satt til true. Dersom informasjonen om «RequireDerivedKeys» utelates fra klienten, fungerer likevel integrasjonen tilsynelatende helt fint, og konklusjonen er derfor at dette elementet ikke har noen praktisk effekt. Dessuten er også "HttpToken" kommentert ut fra wsdl kopi med samme begrunnelse som for "RequireDerivedKeys".

Under arbeid med apache-cxf klienten var det nødvendig å redigere wsdl filen for tjenesten ReceiptExternalEC fordi verktøyet wsdl2java ga feilmelding "Two declarations cause a collision in the ObjectFactory class." Den duplikate deklarasjon ble da utkommentert.

Det er ikke ønskelig å måtte gjøre de beskrevne tilpassninger av wsdl filene for å få Java klienten til å fungere, men inntil de tekniske forskjeller mellom .Net og Java blir rettet opp i eventuelle fremtidige versjoner av rammeverkene så er det dessverre nødvendig.

Bruk av formidlingstjenesten

Tjenestene som inngår i formidlingstjenesten brukes i en bestemt rekkefølge for både avgiver og mottaker. Dette kan illustreres med et sekvensdiagram som følger:



Med henvisning til nummerering i sekvensdiagrammet ovenfor så skjer følgende:

1. InitiateBrokerService

Kallet til tjenesten InitiateBrokerService tar et *Manifest* og en liste av mottakere som parameter. Ref. /1/ avsnitt 6.14.2 så inneholder et manifest:

- ServiceCode+ServiceEditionCode, angir hvilken formidlingstjeneste som skal brukes
- SendersReference, en unik referanse for avgiver
- Reportee, avgiver sitt organisasjonsnummer/fødselsnummer
- FileList, hvilke filer som er inkludert
- PropertyList, en liste av properties, nøkkel og verdi, etter avtale mellom avgiver og mottaker

2. Formidlingsreferanse

Formidlingstjenesten vil validere manifestet og dersom validering er ok så opprettes en formidling med en formidlingsreferanse som returneres.

3. UploadFileStreamed

Formidlingsreferansen må deretter brukes av avgiver ved opplasting av selve datafilen (payload) som skal formidles. Dette gjøres ved kall til tjenesten UploadFileStreamed.

4. Kvittering

For hver formidling holder formidlingstjenesten en kvittering oppdatert, ref. /1/ avsnitt 6.4.1. En kvittering er en datastruktur med en hovedkvittering og liste av underkvitteringer. En underkvittering kan igjen holde en liste av underkvitteringer. Tjenesten UploadFileStreamed oppretter en *hovedkvittering* med en *underkvittering* for hver angitt mottaker. Avgiver får tilbake *hovedkvittering*. Mottakere oppdaterer hver sin egen underkvittering. Enhver oppdatering av en kvittering eller underkvittering vil kun akkumulere informasjon. Kvitteringer i Altinn slettes ikke.

5. GetAvailableFiles

Kallet til tjenesten GetAvailableFiles tar et sett med søkekriterier som parameter, ref. /1/ avsnitt 6.14.1:

- ServiceCode+ServiceEditionCode, angir hvilken formidlingstjeneste det gjelder
- Reportee, mottaker sitt organisasjonsnummer/fødselsnummer
- FileStatus, om filen er *uploaded* (ikke lastet ned) eller *downloaded* (lastet ned men fortsatt tilgjengelig)
- MinSentDateTime, begrenser søket til filer sent etter gitt dato
- MaxSentDateTime, begrenser søket til filer sent før gitt dato

6. Available file list

Tjenesten GetAvailableFiles returnerer en liste av BrokerServiceAvailableFile som er tilgjengelig for angitt mottaker.

7. DownloadFileStreamed

Tjenesten DownloadFileStreamed kalles med enkeltvise referanser hentet fra listen som ble returnert fra kallet til GetAvailableFiles.

8. Datastrøm

Tjenesten DownloadFileStreamed returnerer en datastrøm som kan lagres på fil hos mottaker.

9. UpdateReceipt

Dersom filen lastes ned og verifiseres ok av mottaker så er det mulig for mottaker å meddele avsender status ved å kalle tjenesten UpdateReceipt. Dette steg kan også gjøres etter ConfirmDowloaded hvis ønskelig.

Hver mottaker av en gitt formidling har sin egen underkvittering tilknyttet hovedkvitteringen som ble opprettet da formidling ble instansiert. UpdateReceipt kallet skal da bruke ReceiptID hentet fra listen returnert fra GetAvailableFiles. Denne ReceiptID tilhører mottakers underkvittering.

Kallet til tjenesten UpdateReceipt tar parametere som følger, ref. /1/ avsnitt 6.4.3:

- ReceiptId, unik identifikator for kvittering
- ArchiveReference, hvis det finnes flere kvitteringer med samme arkivreferanse så vil den nyeste bli valgt
- ReceiptText, oppdateringstekst for kvittering
- ReceiptStatus, Status for forsendelse som kvitteringen gjelder: OK, UnExpectedError, ValidationFailed, Rejected
- References, liste med referanser man ønsker legge til, bør begrenses til ReceiversReference
- SubReceipts, liste med barnekvitteringer som også ønskes oppdatert i tillegg til hovedkvitteringen. Barnekvitteringer MÅ identifiseres med ReceiptId. Dette kan ikke benyttes til å lage nye barnekvitteringer.

10. Kvittering

Kallet til UpdateReceipt vil returnere mottakers underkvittering med oppdateringen.

11. ConfirmDownloaded

Det avsluttende steg for mottaker er å kalle tjenesten ConfirmDownloaded med den samme filreferanse som brukt i kallet til tjenesten DownloadFileStreamed. Tjenesten benyttes til å bekrefte at man har fått lastet ned en formidlingsfil i sin helhet. Resultatet er at avsender kan se hvem av filens mottakere som har fått hentet filen. Når alle mottakere av en formidling har kalt ConfirmDownloaded for å bekrefte at data er mottatt så blir datafilen slettet fra Altinn server. Kallet til ConfirmDownloaded skal derfor ikke gjøres før man som mottaker har kontrollert at alle data er mottatt og verifisert ok.

12. GetReceipt

Avgiver kan få vite status for sin formidling ved å kalle tjenesten GetReceipt. Underkvitteringer vil da vise status for hver mottaker hvorvidt disse har fått hentet sine filer.

13. Kvittering

For avgiver returneres formidlingens hovedkvittering.

Java referanseklient

Verktøy/teknologi

Java referanseklienten er basert på seneste versjon av Apache CXF rammeverket som gir støtte for relevante standarder for integrasjon mot Altinn webservices. Programmeringsspråket er Java versjon 8, men det er ikke brukt noen nye java features spesifikt for denne java versjon i koden så eldre versjoner skulle også kunne brukes like godt.

Videre er følgende verktøy/teknologi brukt:

- Maven versjon 3, for bygging
- Spring-context, kjernedelen av spring for *dependency injection*
- Junit, for testing
- Hamcrest, for testing
- Spring-test, for testing
- IntelliJ IDEA Community edition, Java IDE

Generert kode

En viktig del av Apache CXF rammeverket er verktøyet *wsdl2java* som brukes til å generere java klasser basert på wsdl filene for formidlingstjenesten. De genererte klasser kan etter bygging med maven finnes i katalogen /target/generated-sources under prosjektets rotkatalog. Følgende kommando vil bygge hele prosjektet, deriblant kjøre *wsdl2java*, men uten å kjøre noen tester:

```
mvn clean install -DskipTests
```

Referanseklientens grensesnitt

I referanseklienten brukes ikke de automatisk genererte klasser direkte for tilgang til formidlingstjenesten, men isteden via klasser i pakken no.asf.formidling.client.ws.client. Klassen **BrokerECClient** i denne pakken danner et forenklet grensesnitt mot hele formidlingstjenesten og klassen **ReceiptECClient** et tilsvarende forenklet grensesnitt mot Altinn tjenester for kvitteringer. Pakken no.asf.formidling.client.vo inneholder *value objects* klasser som holder data som er input eller output til tjenestene i de to nevnte Client klasser. Det anbefales å se på javadoc dokumentasjonen for disse pakker.

Javadoc

Ved å kjøre følgende maven kommando fra prosjektets rotkatalog får man generert javadoc dokumentasjon for referanseklienten som etterpå kan finnes under katalogen /target/site/apidocs/index.html:

```
mvn javadoc:javadoc
```

Tester

Typisk bruk av de to sentrale klasser i referanseklienten er illustrert i testklasser som finnes i pakken no.asf.formidling.client.ws.client i katalogen /src/test/java. Referanseklienten har en testbruker for hver av rollene avgiver og mottaker og disse er beskrevet i properties filer i katalogen /src/test/resources/properties.

Klassen **FormidlingAvgiverTest** tester opplasting av en fil til formidlingstjenesten. Ettersom initiateService og uploadFile alltid gjøres etter hverandre og returverdi fra den første er input til den neste så er dette forenklet med ett kall til metoden uploadFile i klassen BrokerECClient.

Klassen **FormidlingMottakerTest** tester alle stegene beskrevet ovenfor som utføres i sekvens av en mottaker av en formidling. Det innebærer kall til metoder i BrokerECClient og ReceiptECClient i riktig rekkefølge. Disse testene vil feile dersom det ikke finnes noen filer å hente ned så man bør eventuelt kjøre FormidlingAvgiverTest først.

Klassen **ReceiptECTest** tester kvitteringstjenestene for avgiver. Den bruker en fast kvitteringsid i testen og det burde virke ok ettersom kvitteringer i Altinn ikke slettes.

Konfigurasjon

Klassen **ECClientConfig** i pakken no.asf.formidling.client.config brukes som spring *ContextConfiguration*. Det betyr at den setter opp **spring context** ved oppstart av testene. Dette er angitt i toppen av testklassene med annotation:

```
@ContextConfiguration(classes = {ECClientConfig.class})
```

Alternativt kan tradisjonell xml basert spring konfigurasjon brukes (cxf.xml).

Under katalogen src/main/resources/properties finnes følgende xml konfigurasjonsfiler:

- formidling-tjeneste, Servicekode/versjon for formidlingstjenesten

- formidling-uri, URI addresser for Altinn webservices
- logging-properties, setter nivå på logging for rammeverk og klientkode

Tilsvarende under katalogen src/test/resoures/properties finnes følgende xml konfigurasjonsfiler:

- avgiver-properties, inneholder alle nødvendige data om formidlingens avgiver
- mottaker-properties, inneholder alle nødvendige data om formidlingens mottaker

Interceptors

En viktig del av spring konfigurasjon er instansiering av **Bus** hvor det settes opp webservice **interceptors** som er avgjørende for at kommunikasjonen med Altinn webservices skal fungere. Følgende *interceptors* registreres:

- CookiesInInterceptor, henter attributten *Set-Cookie* fra http header i inngående melding (RequestSecurityTokenResponse) fra webservice og lagrer i minnet (ThreadLocal, dvs. knyttet til en thread).
- CookiesOutInterceptor, setter attributten *Cookie* i http header på utgående melding til webservice hvis den finnes i minnet.
- HeaderInterceptor, setter attributten *Connection* til verdi *Keep-Alive* i header på utgående melding til webservice.
- BadContextTokenInFaultInterceptor, håndterer feilmelding fra webservice av type *BadContextToken* og prøver rette opp.

I tillegg setter ECClientConfig opp *Bus* med logging av alle webserice meldinger (LoggingFeature prettyLogging=true). Følgende er et utsnitt fra loggen ved kjøring av test som viser at interceptors har lagt til headers attributtene *Connection Keep-Alive* og *Cookie*:

```
ID: 3
Address:
https://tt02.altinn.no/ServiceEngineExternal/BrokerServiceExternalECStreamed.s
vc
Encoding: UTF-8
Http-Method: POST
Content-Type: application/soap+xml;
action="http://schemas.xmlsoap.org/ws/2005/02/trust/RST/SCT"
Headers: {Accept=[*/*], Connection=[Keep-Alive], Cookie=[BIGipServerai2v2-
tt02-intweb-80=139074058.20480.0000; path=/]}
Payload: <soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
```

Den samme *Cookie* brukes altså ved flere etterfølgende meldinger til webservice inntil en ny inngående melding med header attributten *Set-Cookie* dukker opp. Ettersom referanseklienten lagrer denne *Cookie* i minnet for en thread så kan det skape problemer å blande tjenestekall med flere virksomhetsbrukere i en og samme test. Et kall til Altinn webservice med ugyldig sikkerhetskonfigurasjon for en virksomhetsbruker vil føre til at denne bruker stenges ute fra Altinn i et visst tidsrom (1 time).