Versjon: 2.1.0
Dato: 28.01.2015

evry.com

Cerebrum2Ephorte – en Ephorte web-tjeneste for Cerebrum

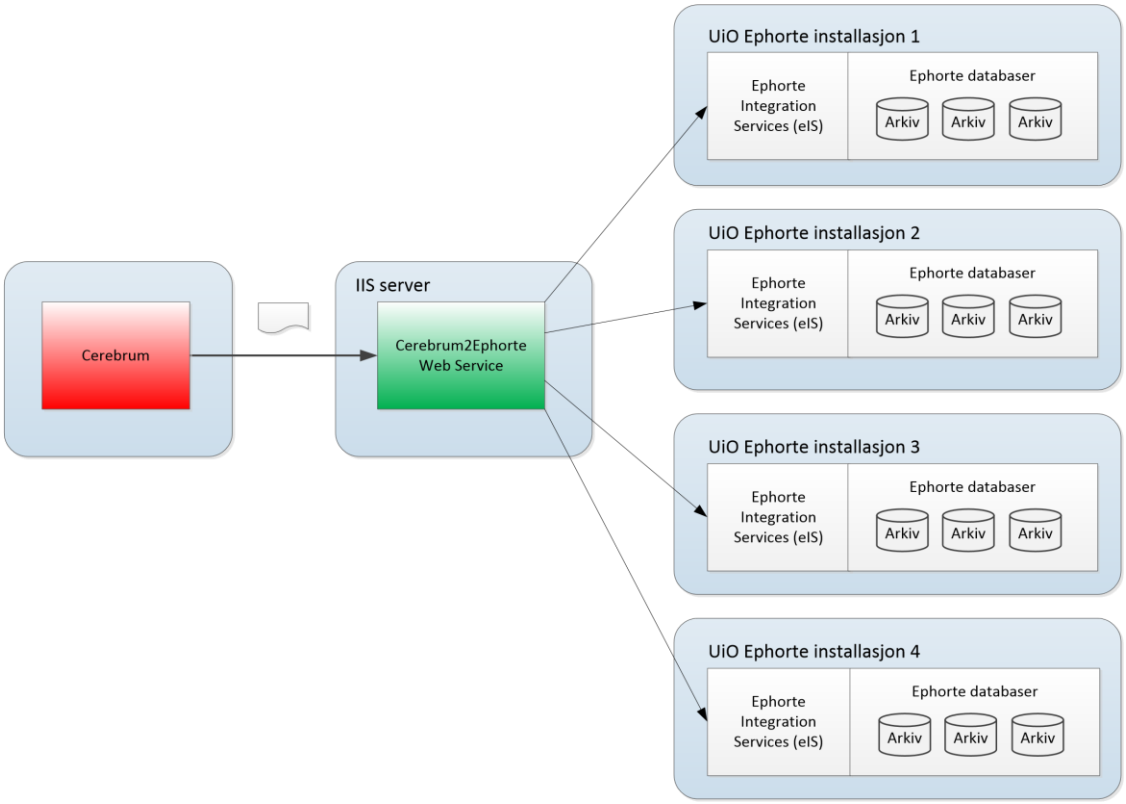
Implementert hos Universitet i Oslo.

Robin Berg Pettersen / Frank Sandersen
Systemutviklere
Product Development & Maintenance robin.berg.pettersen@evry.com
/ frank.sandersen@evry.com

Innhold

1 Introduksjon

Cerebrum2Ephorte er en Ephorte web service skreddersydd for Cerebrum. Cerebrum benyttes for sentralisert tilgangsadministrasjon hos UiO. Cerebrum2Ephorte har web tjenester som er nødvendig for at Cerebrum kan utføre relevant brukeradministrasjon i flere Ephorte-miljøer slik som vist i skissen nedenfor.



Hver metode i web servicen har kunde som input parameter for å skille på de forskjellige Ephorte installasjonene hos UiO. I tillegg må man alltid oppgi navn på Ephorte database innenfor den enkelt Ephorte installasjon. Det vil dermed være Cerebrum som holder orden på hvilken kunde, database og bruker som skal behandles.
Web Servicen gir alltid tilbakemelding på om operasjoner går bra og sender med eventuelle feilmeldinger slik at Cerebrum kan håndtere disse med eventuelt mailvarsling til UiO om dette er ønskelig. Web Servicen er laget mest mulig generisk slik at regler (forretningslogikk) vedlikeholdes av Cerebrum samtidig som det er minimalt konfigurasjon av web servicen. Nødvendig konfigurasjon vil alltid være en del av input parameterne til web metodene.

1.1 Bruk av Ephorte Integration Services (EIS)

Cerebrum2Ephorte fungerer mot én eller flere Ephorte-installasjoner/servere samt én eller flere databaser innenfor den enkelte installasjon. Nye installasjoner og databaser må legges inn i konfigurasjonen med tilhørende Ephorte-bruker som skal benyttes mot disse.

Cerebrum2Ephorte benytter EIS (Ephorte Integration Services) for kommunikasjon med Ephorte. Følgende EIS-tjenester benyttes:

Service	Relative service url	WCF endpoint navn	Kommentar
DocumentService v3	…/services/documents/v3/DocumentService.svc	DocumentServiceV3_BasicHttpBinding	
ObjectModelService v3 No	…/services/objectmodel/v3/No/ObjectModelService.svc	ObjectModelServiceV3No_WSHttpBinding	
ObjectModelService v1	…/services/objectmodel/v1/ObjectModelService.svc/ws	ObjectModelServiceV1_WSHttpBinding	Benyttet unntaksvis på grunn av Ephorte4 v2.0.19 på enkelte baser.
FunctionService v1	…/services/Functions/v1/FunctionsService.svc/ws	FunctionsServiceV1_WSHttpBinding	

ObjectModelService v3 har ikke vært skikkelig støttet i Ephorte før Ephorte4 v2.0.23 og senere. Om man skulle ha gamle baser på Ephorte4 v2.0.19 kan enkelte funksjoner implementers med ObjectModelService v1.

1.2 Kildekode

Kildekode til Cerebrum2Ephorte finnes hos EVRY med følgende referanse:
TFS Server: http://tfs.egdev.net:8080/tfs/
TFS Collection: DefaultCollection
TFS Project: UiO.CerebrumWS

2 Web-metoder

Dette kapitelet inneholder de forskjellige web metoder i Cerebrum2Ephorte – metodenavn, beskrivelse og .net kode-eksempel som viser bruk.

```
[ServiceContract(Name = "Cerebrum2EphorteService", Namespace = "http://Cerebrum2Ephorte/Service")]
public interface IService
{
    [OperationContract]
    DTO.TestUser Test(string username, string password, string customer, string userId);
    [OperationContract]
    DTO.TestUser TestWithEphorte(string username, string password, string customerId, string database, string userId);
    [OperationContract]
    DTO.EphorteOrgUnits GetAllOrgUnits(string username, string password, string customerId, string database);
    [OperationContract]
    DTO.EphorteRoles GetAllRoles(string username, string password, string customerId, string database);
    [OperationContract]
    DTO.EphorteAccessCodes GetAllAccessCodes(string username, string password, string customerId, string database);
    [OperationContract]
    DTO.EphorteUsers GetAllUsers(string username, string password, string customerId, string database);
    [OperationContract]
    DTO.EphorteUsers GetUserList(string username, string password, string customerId, string database, string userSearch);
    [OperationContract]
    DTO.EphorteUserDetails GetUserDetails(string username, string password, string customerId, string database, string userId);
    [OperationContract]
    DTO.Response EnsureUser(string username, string password, string customerId, string database, DTO.EphorteUser user);
    [OperationContract]
    DTO.Response DisableUser(string username, string password, string customerId, string database, string userId);
    [OperationContract]
    DTO.Response EnsureRoleForUser(string username, string password, string customerId, string database, string userId,
                                   string jobTitle, string roleId, string orgId, string fondsSeriesId,
                                   string registryManagementUnitId, bool? setAsDefaultRole);
    [OperationContract]
    DTO.Response EnsureAccessCodeAuthorizationForUser(string username, string password, string customerId, string database,
                                                      string userId, string accessCodeId, string orgId, bool isAuthorizedForAllUnits);
    [OperationContract]
    DTO.EphorteUserBacklog GetUserBacklog(string username, string password, string customerId, string database,
                                          string userId);
    [OperationContract]
    DTO.Response DisableRolesAndAuthorizationsForUser(string username, string password, string customerId,
                                                      string database, string userId);
    [OperationContract]
    DTO.Response DisableUserRole(string username, string password, string customerId, string database, string userId, string roleId,
                                 string orgId, string fondSeriesId, string registryManagementUnitId);
    [OperationContract]
    DTO.Response DisableUserAuthorization(string username, string password, string customerId, string database, string userId,
                                          string accessCodeId, string orgId);
}
```

2.1 Generelle parametere

2.1.1 username

Dette er brukernavnet til Ephorte-brukeren som servicen er linket til. Denne er brukt for autentisering og er obligatorisk for at service-kallene skal fungere. Dersom det ikke er match mellom username og password og det som er spesifisert som Service-bruker, vil man få returnert «Authentication failure!».

2.1.2 password

Passordet til Ephortebrukeren som er linket til servicen.

2.1.3 customerId

Dette er en referanse til gyldige ephortebrukere som er oppgitt i konfigurasjonsfilen (Web.Config).
Under <ephorteConfigurations>-tagen kan denne informasjonen finnes, og "customer"-parameteret må inneholde en av id-ene i denne listen, f.eks "UiO2".
Merk at id-en ikke gjenspeiler et Ephorte-brukernavn, men men en referanse til et fullt oppsett spesifisert i config med alt som trengs for å bruke en spesifikk Ephorte-bruker.

2.1.4 database

Navnet på Ephorte-databasereferansen som skal brukes. f.eks "uiotest2", som peker til "EPHTEST".

2.1.5 userId

En gyldig bruker-id på en Ephorte-bruker.

2.2 Metode: Test

2.2.1 Funksjonsbeskrivelse

```
DTO.TestUser Test(string username, string password, string customer, string userId);
```

Hensikten med metoden er å teste at web servicen fungerer uten at man oppdaterer noe i Ephorte. Metoden forventer userId=Dummy. Hvis userId angis til noe annet vil det returneres en feilmelding.

2.2.2 Parametere

customer: Også oppgitt som customerId i andre funksjoner. Dette er en referanse til gyldige ephortebrukere som er oppgitt i konfigurasjonsfilen (Web.Config). Under <ephorteConfigurations>-tagen kan denne informasjonen finnes, og "customer"-parameteret må inneholde en av id-ene i denne listen, f.eks "UiO2".
userId: Normalt ville det bli brukt en valgfri Ephorte-bruker her, men i dette kallet brukes en "dummy"-bruker med en spesifikk hardkodet verdi.

2.2.3 Eksempel

```
var ephorteUserTest = proxy.Test("ephsys", "*******",
«UiO2», "Dummy");
if (!ephorteUserTest.HasError)
{
    Console.WriteLine(string.Format("User {0} found", ephorteUserTest.UserId));
} else
    Console.WriteLine(string.Format("{0}", ephorteUserTest.ErrorMessage));
```

2.3 Metode: TestWithEphorte

2.3.1 Funksjonsbeskrivelse

```
DTO.TestUser TestWithEphorte(string username, string password, string customerId, string database, string userId);
```

Hensikten med denne metoden er å teste web servicen og samtidig gjøre et reelt oppslag mot brukertabellen i Ephorte. Her må man angi en gyldig Ephorte database og et gyldig Ephorte brukernavn.

2.3.2 Parametere

username: <Se "generelle parametre">

password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">

2.3.3 Eksempel

```
var ephorteUserTest = proxy.TestWithEphorte("ephsys", "*******", «UiO2»,
«UIOTEST2», "BJOJO");
if (!ephorteUserTest.HasError)
{
    Console.WriteLine(string.Format("User {0} found", ephorteUserTest.UserId));
} else
    Console.WriteLine(string.Format("{0}", ephorteUserTest.ErrorMessage));
```

Metode:

2.4 GetAllOrgUnits

2.4.1 Funksjonsbeskrivelse

```
DTO.EphorteOrgUnits GetAllOrgUnits(string username, string password, string customerId, string database);
```

Denne metoden returnerer alle aktive (ikke nedlagte) organisasjonsenheter fra Ephorte. Brukere er ikke direkte koblet til organisasjonsenheter men deres rolletilganger og autorisasjoner kan være det.

2.4.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">

2.4.3 Eksempel

```
var ephorteOrgUnits = proxy.GetAllOrgUnits("ephsys", "*******",
«UiO2», «UIOTEST2»);
if (!ephorteOrgUnits.HasError)
{
    foreach (EphorteOrg org in ephorteOrgUnits.OrgUnits)
        Console.WriteLine(string.Format("Org found: {0} ({1}) - {2}", org.OrgId, org.Name, org.IsTop ? "Top org" : "Not top org"));
} else
    Console.WriteLine(string.Format("{0}", ephorteOrgUnits.ErrorMessage));
```

2.5 Metode: GetAllRoles

2.5.1 Funksjonsbeskrivelse

```
DTO.EphorteRoles GetAllRoles(string username, string password, string customerId, string database);
```

Metode:

Denne metoden returnerer alle roller. Disse benyttes sammen med organisasjonsenheter når man skal tilordne rolletilganger til en bruker.

2.5.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">

2.5.3 Eksempel

```
var ephorteRoles = proxy.GetAllRoles("ephsys", "*******",
«UiO2», «UIOTEST2»);
if (!ephorteRoles.HasError)
{
    foreach (EphorteRole role in ephorteRoles.Roles)
        Console.WriteLine(string.Format("Role found: {0} - {1}", role.RoleId, role.Description));
} else
    Console.WriteLine(string.Format("{0}", ephorteRoles.ErrorMessage));
```

2.6 GetAllAccessCodes

2.6.1 Funksjonsbeskrivelse

```
DTO.EphorteAccessCodes GetAllAccessCodes(string username, string password, string customerId, string database);
```

Denne metoden returnerer alle aktive tilgangskoder. Disse benyttes sammen med organisasjonsenheter når man skal gi en bruker en autorisasjon.

2.6.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">

Metode:

2.6.3. Eksempel

```
var ephorteAccessCodes = proxy.GetAllAccessCodes("ephsys", "*******",
«UiO2», «UIOTEST2»);
if (!ephorteAccessCodes.HasError)
{
    foreach (EphorteAccessCode accesscode in ephorteAccessCodes.AccessCodes)
        Console.WriteLine(string.Format("Access code found: {0} - {1}", accesscode.AccessCodeId, accesscode.Description));
} else
    Console.WriteLine(string.Format("{0}", ephorteAccessCodes.ErrorMessage));
```

2.7 Metode: GetAllUsers

2.7.1 Funksjonsbeskrivelse

```
DTO.EphorteUsers GetAllUsers(string username, string password, string customerId, string database);
```

Denne metoden returnerer en liste med alle aktive (gyldige) brukere. Hver bruker inneholder kontaktinformasjon. Skal man ha mer informasjon om den enkelte bruker må man videre kalle GetUserDetails() for den enkelte bruker. Merk at dette kallet vil bruke lang tid om det er mange brukere i Ephorte, og timeout-parameteret for klienten som bruker servicen bør settes deretter.

2.7.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">

Metode:

2.7.3 Eksempel

```
var ephorteUsers = proxy.GetAllUsers("ephsys",
"********", «UiO2», «UIOTEST2»);
if (!ephorteUsers.HasError) {
    Console.WriteLine(string.Format("Users    found:    {0}",    ephorteUsers.Users.Count()));
foreach (EphorteUser user in ephorteUsers.Users)
    {
        Console.WriteLine(string.Format("User found: {0} - {1}", user.UserId, user.FullName));

        var ephorteUser = proxy.GetUserDetails(«UiO2», «UIOTEST2», user.UserId);
if (!ephorteUser.HasError)
        {
            Console.WriteLine(string.Format("Details for user {0} found: {1} roles and {2} authorizations", ephorteUser.User.UserId,
ephorteUser.UserRoles.Count(), ephorteUser.UserAuthorizations.Count()));
        }
else
            Console.WriteLine(string.Format("{0}", ephorteUser.ErrorMessage));
    }
} else
    Console.WriteLine(string.Format("{0}", ephorteUsers.ErrorMessage))
```

2.8 GetUserList

2.8.1 Funksjonsbeskrivelse

```
DTO.EphorteUsers GetUserList(string username, string password, string customerId, string database, string
userSearch);
```

Denne metoden henter alle brukere med et brukernavn som inneholder den teksten som blir satt i søkeparameteret. Så lenge brukernavnet har deler av teksten i søkeparameteret i seg et eller annet sted, vil brukeren bli returnert. Det er hverken støtte for eller behov for «wildcards» i søket, da søkestringen søkes opp i hele brukernavnet. Et søk med userSearch satt til «ril» ville f.eks finne en bruker med brukernavn «arild». Hver bruker inneholder kontaktinformasjon. Skal man ha mer informasjon om den enkelte bruker må man videre kalle GetUserDetails() for den enkelte bruker.

Metode:

2.8.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">
userSearch: Søkekriterier for brukernavn. Her skriver man inn en tekst som blir mappet opp mot brukernavn i Ephorte. Dersom teksten som blir puttet inn er en del av et brukernavn, vil brukeren bli lagt til i en liste som inneholder alle brukere med brukernavn som matcher mot dette søket.

2.8.3 Eksempel

```
var ephorteUserList = proxy.GetUserList("ephsys", "*******", customerId, database, "arild");
if (!ephorteUserList.HasError)
{
    Console.WriteLine(string.Format("Users found: {0}", ephorteUserList.Users.Count()));
    foreach (EphorteUser user in ephorteUserList.Users.Take(5))
    {
        Console.WriteLine(string.Format("User found: {0} - {1}", user.UserId, user.FullName));


        var ephorteUser = proxy.GetUserDetails(customerId, database, user.UserId);
        if (!ephorteUser.HasError)
        {
            Console.WriteLine(string.Format("Details for user {0} found: {1} roles and {2} authorizations", ephorteUser.User.UserId,
ephorteUser.UserRoles.Count(), ephorteUser.UserAuthorizations.Count()));
        }
        else
            Console.WriteLine(string.Format("{0}", ephorteUser.ErrorMessage));

    }
}
else
    Console.WriteLine(string.Format("{0}", ephorteUserList.ErrorMessage));
```

2.9 GetUserDetails

Metode:

2.9.1 Funksjonsbeskrivelse

```
DTO.EphorteUserDetails GetUserDetails(string username, string password, string customerId, string database, string userId);
```

Denne metoden returnerer kontaktinformasjon om bruker samt lister med brukers rolletilganger og autorisasjoner.

2.9.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">
userId: <Se "generelle parametre">

2.9.3 Eksempel

```
var ephorteUser = proxy.GetUserDetails("ephsys", "*******", «UiO2»,
«UIOTEST2», "BJOJO");
if (!ephorteUser.HasError)
{
    Console.WriteLine(string.Format("Details for user {0} found: {1} roles and {2} authorizations", ephorteUser.User.UserId,
ephorteUser.UserRoles.Count(), ephorteUser.UserAuthorizations.Count()));
} else
    Console.WriteLine(string.Format("{0}", ephorteUser.ErrorMessage));
```

2.10 Metode: EnsureUser

2.10.1 Funksjonsbeskrivelse

```
DTO.Response EnsureUser(string username, string password, string customerId, string database, DTO.EphorteUser user);
```

Denne metoden oppretter bruker hvis denne ikke finnes fra før. Hvis bruker finnes fra før vil kontaktinformasjon oppdateres.

Metode:

2.10.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">
user: Ephorte-bruker som skal endres eller legges til. Om brukeren ikke finnes fra før, blir brukeren automatisk lagt til, ellers vil brukerinformasjon bli oppdatert.

2.10.3 Eksempel

```
var newUser = new EphorteUser
{
    UserId = "OLANOR5",
    FirstName = "Ola",
    LastName = "Nordmann",
    FullName = "Ola Nordmann",
    EmailAddress = "ola.nordmann@norge.no",
    Telephone = "12345678",
    Mobile = "99911999",
    StreetAddress = "Postveien 1",
    ZipCode = "3960",
    City = "Stathelle",
};
var response = proxy.EnsureUser("ephsys", "*******", «UiO2»,
«UIOTEST2», newUser);
if (!response.HasError)
    Console.WriteLine(string.Format("User created/updated: {0}", newUser.UserId));
else
    Console.WriteLine(string.Format("{0}", response.ErrorMessage));
```

2.11 EnsureRoleForUser

Metode:

2.11.1 Funksjonsbeskrivelse

```
DTO.Response EnsureRoleForUser(string username, string password, string customerId, string database, string userId,
string jobTitle, string roleId, string orgId, string fondsSeriesId, string registryManagementUnitId, bool?
setAsDefaultRole);
```

Denne metoden oppretter/oppdaterer en personlig rolle for en bruker. Personlige roller kobles blant annet mot til organisasjonsenhet, journalenhet og rolle.

2.11.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">
userId: <Se "generelle parametre">
jobTitle: Jobbtittel
roleId: ID på rollen som skal endres.
orgId: Organisasjons-ID. Også kalt administrativ enhet.
fondSeriesId: Dette tilsvarer "arkivdel" i Ephorte.
registryManagementUnitId: JournalEnhet
setAsDefaultRole: Skal denne rollen settes som standardrolle? (true eller false)

2.11.3 Eksempel

```
var response = proxy.EnsureRoleForUser("ephsys", "*******", «UiO2», «UIOTEST2», "KARNOR", "Arkivansvarlig", "AR1", "000000",
"SAK1", "JOURNAL", false);
if (!response.HasError)
    Console.WriteLine(string.Format("User role created/updated: {0}", "Arkivansvarlig")); else
    Console.WriteLine(string.Format("{0}", response.ErrorMessage));
```

2.12 Metode: EnsureAccessCodeAuthorizationForUser

Metode:

2.12.1 Funksjonsbeskrivelse

```
DTO.Response EnsureAccessCodeAuthorizationForUser(string username, string password, string customerId, string database, string userId, string accessCodeId, string orgId, bool isAuthorizedForAllUnits);
```

Denne metoden oppretter/oppdaterer en autorisasjon for en bruker. Autorisasjon er en klarering for en bestemt tilgangskode i Ephorte. Autorisasjonen kan enten gjelde for hele virksomheten, kun egne saker eller saker under en bestemt orgenhet.

- For kun egne saker settes orgId=null og isAuthorizedForAllUnits=false
- For hele virksomheten settes orgId=null og isAuthorizedForAllUnits=true
- For saker under bestemt orgenhet settes orgId=<kortnavn for administrativ enhet> og isAuthorizedForAllUnits=false

2.12.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">
userId: <Se "generelle parametre">
accessCodeId: Tilgangskode-ID.
orgId: Organisasjons-ID. Også kalt administrativ enhet.
isAuthorizedForAllUnits: Er bruker autorisert for hele virksomheten?

2.12.3 Eksempel

Metode:

```
//Ola Nordmann: UO for saker i hele virksomheten (kommunen) – NIVÅ 2
var response1 = proxy.EnsureAccessCodeAuthorizationForUser("ephsys", "*******", «UiO2», "EPHORTE-FK",
"OLANOR", "UO", null, true);
if (response1.HasError) Console.WriteLine(string.Format("{0}", response1.ErrorMessage));
//Ola Nordmann: E for saker i egen enhet (FSK) – NIVÅ 1
response1 = proxy.EnsureAccessCodeAuthorizationForUser("ephsys", "*******", «UiO2», «UIOTEST2»,
"OLANOR", "E", "000000", false);
if (response1.HasError) Console.WriteLine(string.Format("{0}", response1.ErrorMessage));
//Ola Nordmann: P for egne saker – NIVÅ 0
response1 = proxy.EnsureAccessCodeAuthorizationForUser("ephsys", "*******", «UiO2», «UIOTEST2», "OLANOR",
"P", null, false);
if (response1.HasError) Console.WriteLine(string.Format("{0}", response1.ErrorMessage));
```

Metode:

2.13 GetUserBacklog

2.13.1 Funksjonsbeskrivelse

```
DTO.EphorteUserBacklog GetUserBacklog(string username, string password, string customerId, string database, string userId);
```

Denne metoden benyttes for å hente informasjon om åpne saker for en bruker. Informasjonen kan f.eks presenteres i en mail til vedkommende med kopi til han/hennes leder en tid før man skal bytte jobb eller slutte. Metoden henter følgende informasjon om bruker:

- alle åpne saker der bruker er saksansvarlig og saksstatus er R, B eller V
- alle åpne journalposter der bruker er saksbehandler og journalstatus er R eller M

Ingen meldinger betyr ingen backlogg (HasBacklog = false). Metoden returnerer også email adresse til bruker og leder (= den som har LD rolle mot admin-enhet tilknyttet brukers standard rolle) som kan benyttes til utsendelse av mail.

2.13.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">
userId: <Se "generelle parametre">

2.13.3 Eksempel

```
var userBackLog = proxy.GetUserBacklog("ephsys", "*******", «UiO2»,
«UIOTEST2», "BJOJO");
if (!userBackLog.HasError)
{
    if (userBackLog.HasBacklog)
    {
        foreach (string logMsg in userBackLog.BacklogMessage)
Console.WriteLine(logMsg);

        Console.WriteLine(string.Format("User email: {0}", userBackLog.UserEmail));
        Console.WriteLine(string.Format("Leader email: {0}", userBackLog.LeaderEmail));
} } else
    Console.WriteLine(string.Format("{0}", userBackLog.ErrorMessage));
```

Output fra denne metoden kan være slik:

```
"BJOJO er saksansvarlig for sak 2010/1719 som har status B"
"BJOJO er saksansvarlig for sak 2010/1620 som har status B"
"BJOJO er saksansvarlig for sak 2010/1597 som har status B"
"BJOJO er saksansvarlig for sak 2010/133 som har status B"
"BJOJO er saksbehandler for journalpost 127/2012 i sak 2012/25. Journalstataus er R"
"BJOJO er saksbehandler for journalpost 122/2012 i sak 2010/133. Journalstataus er R"
"BJOJO er saksbehandler for journalpost 120/2012 i sak 2011/241. Journalstataus er R"
"BJOJO er saksbehandler for journalpost 119/2012 i sak 2011/19. Journalstataus er R"
"BJOJO er saksbehandler for journalpost 118/2012 i sak 2011/19. Journalstataus er R"
"BJOJO er saksbehandler for journalpost 589/2011 i sak 2010/1965. Journalstataus er R"
"BJOJO er saksbehandler for journalpost 235/2011 i sak 2010/2013. Journalstataus er R"
"BJOJO er saksbehandler for journalpost 4651/2010 i sak 2010/1719. Journalstataus er R"
"BJOJO er saksbehandler for journalpost 958/2011 i sak 2011/314. Dokumenttype er I og journalposten er ikke avskrevet!"
"BJOJO er saksbehandler for journalpost 677/2011 i sak 2011/241. Dokumenttype er I og journalposten er ikke avskrevet!"
"BJOJO er saksbehandler for journalpost 609/2011 i sak 2011/19. Dokumenttype er I og journalposten er ikke avskrevet!"
```

2.14 Metode: DisableUser

2.14.1 Funksjonsbeskrivelse

```
DTO.Response DisableUser(string username, string password, string customerId, string database, string userId);
```

Denne metoden vil deaktivere angitt bruker slik at denne ikke kan benyttes for innlogging.

2.14.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">
userId: <Se "generelle parametre">

2.14.3 Eksempel

```
var response = proxy.DisableUser("ephsys", "*******", «UiO2»,
«UIOTEST2», "KARNOR");
if (!response.HasError)
    Console.WriteLine(string.Format("User deactivated: {0}", "KARNOR"));
else
    Console.WriteLine(string.Format("{0}", response.ErrorMessage));
```

2.15 Metode: DisableRolesAndAuthorizationsForUser

2.15.1 Funksjonsbeskrivelse

```
DTO.Response DisableRolesAndAuthorizationsForUser(string username, string password, string customerId, string
database, string userId);
```

Denne metoden vil deaktivere alle personroller og autorisasjoner på angitt bruker.

2.15.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">

userId: <Se "generelle parametre">

2.15.3 Eksempel

```
var response = proxy.DisableRolesAndAuthorizationsForUser("ephsys", "*******", «UiO2», «UIOTEST2», "OLANOR");
if (response.HasError) Console.WriteLine(string.Format("{0}", response.ErrorMessage));
```

2.16 Metode: DisableUserRole

2.16.1 Funksjonsbeskrivelse

```
public DTO.Response DisableUserRole(string username, string password, string customerId, string database, string userId, string roleId, string orgId, string fondSeriesId, string registryManagementUnitId)
```

Denne metoden vil deaktivere en rolle som er knyttet til en bruker. roleId består av to bokstaver, og orgId kan enten være tomt eller ha en orgId på seks tegn.

2.16.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">
userId: <Se "generelle parametre">
fondSeriesId: Dette tilsvarer "arkivdel" i Ephorte.
registryManagementUnitId: JournalEnhet.

2.16.3 Eksempel

```
var response = proxy.DisableUserRole("ephsys", "*******", «UiO2», «UIOTEST2», "OLANOR", "AR1", "000000", "SAK1", "JOURNAL");
if (response.HasError) Console.WriteLine(string.Format("{0}", response.ErrorMessage));
```

2.17 Metode: DisableUserAuthorization

2.17.1 Funksjonsbeskrivelse

```
DTO.Response DisableUserAuthorization(string username, string password, string customerId, string database, string userId, string accessCodeId, string orgId);
```

Denne metoden vil deaktivere en tilgang som er knyttet til en bruker. accessCodeId består som regel av to bokstaver, og orgId kan enten være tomt eller ha en orgId på seks tegn.

2.17.2 Parametere

username: <Se "generelle parametre">
password: <Se "generelle parametre">
customerId: <Se "generelle parametre">
database: <Se "generelle parametre">
userId: <Se "generelle parametre">
accessCodeId: Tilgangskode-ID.
orgId: Organisasjons-ID.

2.17.3 Eksempel

```
var response = proxy.DisableRolesAndAuthorizationsForUser("ephsys", "*******", «UiO2»,
«UIOTEST2», "OLANOR", "SD", "000000");
if (response.HasError) Console.WriteLine(string.Format("{0}", response.ErrorMessage));
```

3 Dataklasser for serviceresponser

3.1 Innledning

Alle service-kall returnerer data av forskjellige typer avhengig av hvilket kall som blir kjørt. Disse dataene er definert i egne klasser i servicen. Disse klassene brukes for å lagre data som blir returnert fra Ephorte via eIS og sende dem som respons til Cerebrum, eller hva som enn benytter seg av servicen. Informasjonen som står her er rettet mot tekniske brukere. Merk at «DTO.»-delen av navnene bare er «namespaces» og reflekteres kun i koden. Dette er ikke noe som vil dukke opp ved bruk av servicen, eller gjenspeiles i dataene som returneres. «DTO.Response» vil f.eks kun vises som «Response».

3.2 Dataklasse: DTO.Response

```
public class Response
{
    public Response()
    {
        this.HasError = false;
        this.ErrorMessage = "";
        this.OccurencesFound = null;
    }
    public bool HasError { get; set; }
    public string ErrorMessage { get; set; }
    public int? OccurencesFound { get; set; }
}
```

3.3 Dataklasse: DTO.TestUser

```
public class TestUser : DTO.Response
{
    public string  UserId { get; set; }
    public string FullName { get; set; }
}
```

3.4 Dataklasse: DTO.EphorteOrg

```
public class EphorteOrg
{
    public string  OrgId { get; set; }
    public string ParentOrgId { get; set; }
    public bool IsTop { get; set; }
    public string Name { get; set; }
}
```

3.5 Dataklasse: DTO.EphorteOrgUnits

```
public class EphorteOrgUnits : DTO.Response
{
    public List<DTO.EphorteOrg> OrgUnits { get; set; }
}
```

3.6 Dataklasse: DTO.EphorteRole

```
public class EphorteRole
{
    public string RoleId { get; set; }
```

```
    public string Description { get; set; }

    public static string GetDescription(Rolle rolle)
    {
        string description = "";
        if (rolle.Systemansvarlig != null && rolle.Systemansvarlig == true) description += "Systemansvarlig, ";
        if (rolle.Arkivleder != null && rolle.Arkivleder == true) description += "Arkivleder, ";
        if (rolle.Arkivpersonell != null && rolle.Arkivpersonell == true) description += "Arkivpersonell, ";
        if (rolle.Leder != null && rolle.Leder == true) description += "Leder, ";
        if (rolle.Saksbehandler != null && rolle.Saksbehandler == true) description += "Saksbehandler, ";
        if (rolle.Utvalgssekretaer != null && rolle.Utvalgssekretaer == true) description += "Utvalgssekretaer, ";
        return description.Trim().TrimEnd(',');
    }
}
```

3.7 Dataklasse: DTO.EphorteRoles

```
public class EphorteRoles : DTO.Response
{
    public List<DTO.EphorteRole> Roles { get; set; }
}
```

3.8 Dataklasse: DTO.EphorteAccessCode

```
public class EphorteAccessCode
{
    public string AccessCodeId { get; set; }
    public string Description { get; set; }
}
```

3.9 Dataklasse: DTO.EphorteAccessCodes

```
public class EphorteAccessCodes : DTO.Response
{
    public List<DTO.EphorteAccessCode> AccessCodes { get; set; }
}
```

3.10 Dataklasse: DTO.EphorteUser

```
public class EphorteUser //: DTO.Response
{
    public string UserId { get; set; }
    public string Initials { get; set; }
    public string FirstName { get; set; }
```

```
    public string MiddelName { get; set; }
    public string LastName { get; set; }
    public string FullName { get; set; }
    public string EmailAddress  { get; set; }
    public string Telephone { get; set; }
    public string Mobile { get; set; }
    public string StreetAddress { get; set; }
    public string ZipCode { get; set; }
    public string City { get; set; }
}
```

3.11 Dataklasse: DTO.EphorteUsers

```
public class EphorteUsers : DTO.Response
{
    public List<DTO.EphorteUser> Users { get; set; }

}
```

3.12 Dataklasse: DTO.EphorteUserRole

```
public class EphorteUserRole
{
    public EphorteRole Role { get; set; }
    public EphorteOrg Org { get; set; }
    public string FondsSeriesId { get; set; }
    public string RegistryManagementUnitId { get; set; }
    public bool IsDefault { get; set; }
    public string RoleTitle { get; set; }
    public string JobTitle { get; set; }

}
```

3.13 Dataklasse: DTO.EphorteUserAutorization

```
public class EphorteUserAuthorization
{
    public string AccessCodeId { get; set; }
    public bool IsAutorizedForAllOrgUnits { get; set; }
    public string OrgId { get; set; }
}
```

3.14 Dataklasse: DTO.EphorteUserDetails

```
public class EphorteUserDetails : DTO.Response
```

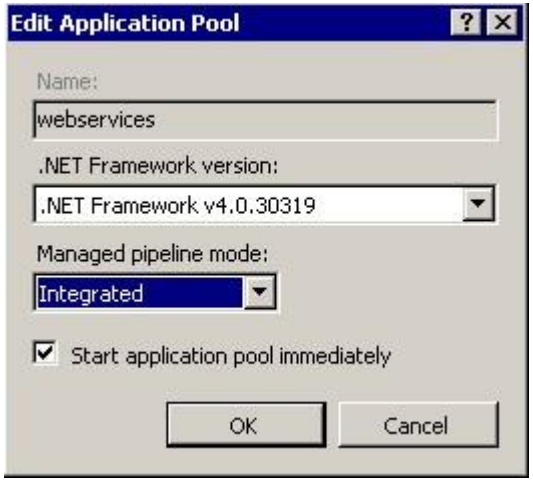
```
{
    public DTO.EphorteUser User { get; set; }
    public List<DTO.EphorteUserRole> UserRoles { get; set; }
    public List<DTO.EphorteUserAuthorization> UserAuthorizations { get; set; }
}
```

3.15 Dataklasse: DTO.EphorteUserBacklog

```
public class EphorteUserBacklog : DTO.Response
{
    public EphorteUserBacklog()
    {
        this.HasBacklog = false;
        this.BacklogMessage = new List<string>();
    }
    public string UserEmail { get; set; }
    public string LeaderEmail { get; set; }
    public bool HasBacklog { get; set; }
    public List<string> BacklogMessage { get; set; }
}
```

4 Installasjon

Her følger en kort beskrivelse av installasjon av web servicen Cerebrum2Ephorte. Den betinger at man har web servicen tilgjengelig som pakket web applikasjon (zip-fil). Den nye servicen skal bruke .Net 4.0 app pool:



Authentication: Anonymous Authentication Enabled og resten av alternativene kan være Disabled.

Installasjonsprosedyre

1. Opprett ny katalog for web site
   Eksempel på plassering: c:\inetpub\wwwroot\Cerebrum2EphorteService
2. Kopier zip-filen Cerebrum2EphorteWebApp.zip til server. Start InetMgr og naviger ned til Sites -> Default web . Høyreklikk på denne og velg «Deploy->Import Application»
3. Test ny web site ved å legge inn gjeldende url i browser og se at det åpnes en web service beskrivelse i browseren. F.eks denne url: http://ephuiotest.trofast.uio.no/Cerebrum2Ephorte/Service.svc

5 Konfigurasjon

Konfigurasjon av servicen gjøres i filen web.config.

Konfig av Ephorte installasjon per kunde

```
  <customConfiguration>
    <commonConfig smtpserver="smtp.uio.no"
                  tempfolder="%TEMP%\Cerebrum2EphorteService\_tempFiles"
                  databasesWithNoPersonAddressSupport="EPHORTE-OLD" />
    <ephorteConfigurations>
      <clear/>
      <!--serviceBaseUrl ends with /ePhorteWeb/ or /NCore/-->
      <add id="UiO" desc="Universitetet i Oslo" user="***" password="*******" role=""
serviceBaseUrl="http://uiotest.ephorte.uninett.no/ePhorteWeb/"/>
      <add id="UiO2" desc="Universitetet i Oslo2" user="***" password="*******" role=""
serviceBaseUrl="http://uiotest.ephorte.uninett.no/ePhorteWeb/"/>
      <add id="UiO3" desc="Universitetet i Oslo3" user="***" password="*******" role=""
serviceBaseUrl="http://uiotest.ephorte.uninett.no/ePhorteWeb/"/>
    </ephorteConfigurations>
  </customConfiguration>
```

Forklaring til `commonConfig->databasesWithNoPersonAddressSupport`: Unntasvis kan det finnes gamle baser som ikke er riktig oppgradert. Disse støtter ikke registrering av personadresser via EIS

Forklaring til `commonConfig-> smtpserver`: Denne brukes foreløpig ikke.

Forklaring til `commonConfig-> tempfolder`: Denne brukes foreløpig ikke men den ikke være tom.

Forklaring til `ephorteConfigurations-> add`: Her legger man inn ny kunde. `id` er kundeid som også brukes ved kall mot web metodene. `user` /`password` /`role` angir den Ephorte brukeren som benyttes av web servicen ved arbeid i Ephorte. `serviceBaseUrl` er root url som sammen med EIS service url vil danne en komplett url.

Logging med standard komponent Log4Net

```
<log4net>
    <appender name="RollingFile" type="log4net.Appender.RollingFileAppender">
      <!-- App Pool exec account need write access to web site folder to create log files-->
      <!--<file value="${TEMP}\Cerebrum2EphorteService\Service.log"/>-->
      <file value="c:\windows\temp\Cerebrum2EphorteService\Service.log"/>
      <appendToFile value="true"/>
      <maximumFileSize value="10000KB"/>
      <maxSizeRollBackups value="25"/>
      <layout type="log4net.Layout.PatternLayout">
        <conversionPattern value="%date %C{1}.%M - %message%newline"/>
        <!--<conversionPattern value="%-5p [%t] %C{1}.%M - %m%n"/>-->
      </layout>
    </appender>
    <appender name="SmtpAppender" type="log4net.Appender.SmtpAppender">
      <to value="robin.berg.pettersen@evry.com"/>
      <from value="Cerebrum2EphorteService@usit.uio.no"/>
      <subject value="Cerebrum2EphorteService Error"/>
      <smtpHost value="159.171.252.75"/>
      <bufferSize value="1"/>
      <lossy value="true"/>
      <evaluator type="log4net.Core.LevelEvaluator">
        <threshold value="ERROR"/>
      </evaluator>
      <layout type="log4net.Layout.PatternLayout">
        <conversionPattern value="%newline%date [%thread] %-5level %logger [%property{NDC}] - %message%newline%newline%newline"/>
      </layout>
</appender>
    <appender name="ConsoleAppender" type="log4net.Appender.ConsoleAppender">
      <layout type="log4net.Layout.PatternLayout">
        <conversionPattern value="%date [%thread] %-5level %logger [%ndc] - %message%newline"/>
      </layout>
</appender>
    <appender name="EventLogAppender" type="log4net.Appender.EventLogAppender">
      <applicationName value="Cerebrum2Ephorte Service"/>
      <layout type="log4net.Layout.PatternLayout">
        <conversionPattern value="%date [%thread] %-5level %logger [%property{NDC}] - %message%newline"/>
      </layout>
    </appender>
    <root>
      <!-- use WARN in production -->
      <!--<level value="WARN"/>-->
```

```
        <level value="DEBUG"/>
        <appender-ref ref="RollingFile"/>
```

```
      <appender-ref ref="EventLogAppender"/>
      <appender-ref ref="SmtpAppender"/>
    </root>
  </log4net>
```

6 Driftsinformasjon

Loggfilene kan etter hvert bli mange så lenge logg-level = DEBUG. Det er mulig å justere level ned til f.eks WARN (warning) men da mister man en del informasjon i loggen som vil være nyttig den dagen man måtte ha behov for å feilsøke.
Plassering av loggfiler er angitt i konfig-filen under `log4net-> appender-> file.`
Mest vanlig plassering er dette: c:\windows\temp\Cerebrum2EphorteService\Service.log

7 Tilleggsinformasjon

7.1 Disabling av enkeltroller

Det er per i dag ingen funksjonalitet for å disable enkeltroller, men dette er gjort av en grunn.
Normalt vil roller og tilganger bli disablet som et resultat at noen enten bytter stilling internt eller slutter, og hvis man vil gjøre enkeltroller utilgjengelige burde Cerebrum til enhver tid vite hva som ligger i Ephorte. Dette er fordi man også på Ephortesiden kan gi spesialtilganger ved behov, som kan være vanskelige å håndtere for Cerebrum.

Når man ønsker å rydde i tilganger er det naturlig å resette alle tilgangene for en bruker slik at man får ryddet opp i eventuelle spesialtilganger. Dette vil da være en sikkerhetsmekanisme som gjør at spesialtilganger ikke bare ligger igjen i Ephorte. Man kan så legge inn rollene og tilgangene man ønsker på nytt.

Skulle det allikevel være ønske om funksjonalitet for å disable enkeltroller, kan dette ganske enkelt legges til.