

PROACTIVE 

FLIS PORTAL

GENNEMGANG OG STIKPRØVER (SEPTEMBER 2015)

INDHOLD

1. FORMÅL OG SCOPE	1
1.1. TEKNOLOGIER OG RAMMEVÆRKER	1
2. ARKITEKTUR	2
2.1. KOMBIT.FLIS.PORTAL	3
2.2. SCRIPTS	3
2.3. SHAREPOINT TILPASNINGER.....	3
3. KILDEKODE.....	4
3.1. C# KODESTANDARDER	4
3.2. POWERSHELL KODESTANDARDER.....	5
3.3. JAVASCRIPT KODESTANDARDER	5
3.4. ASP.NET WEBFORMS.....	5
3.5. SPMETAL	5
3.6. KOBLING TIL NETCOMPANYS CRM SYSTEM.....	6
4. SEMANTIK	7
4.1. FLIS_LOGGER.....	7
4.2. DISPOSE.....	7
4.3. ELEVATE.....	8
4.4. LOKAL UDVIKLING	8
4.5. SQL OG MDX.....	9
4.6. FEATURE RECEIVERS.....	9
4.7. INTERFACES OG SNITFLADER	9
5. STIKPRØVER	10
5.1. STIKPRØVE "3.1 FLIS STANDARDRAPPORT".....	10
5.2. STIKPRØVE "5.1 KOMMUNE".....	10
5.3. STIKPRØVE "8.1 DATAPAKKEBESTILLING".....	11
6. AUTOMATISK TEST (UNIT/INTEGRATION).....	12
7. KONKLUSION.....	13

1. FORMÅL OG SCOPE

Formålet med dette dokument er at give en gennemgang af FLIS Portal kildekoden og arkitektur i forhold til overdragelse til en ny udviklings- og systemleverandør.

1.1. TEKNOLOGIER OG RAMMEVÆRKER

Følgende teknologier er brugt til udviklings i backend og frontend:

1.1.1. BACKEND:

- SPMetal (Kombit.Flis.Portal.DataAccess.LinqContext)
- CKSDev
- SharePoint 2010
- ASP.Net
- AutoSPInstaller
- PowerShell
- AnalysisServices
- SQL

1.1.2. FRONTEND

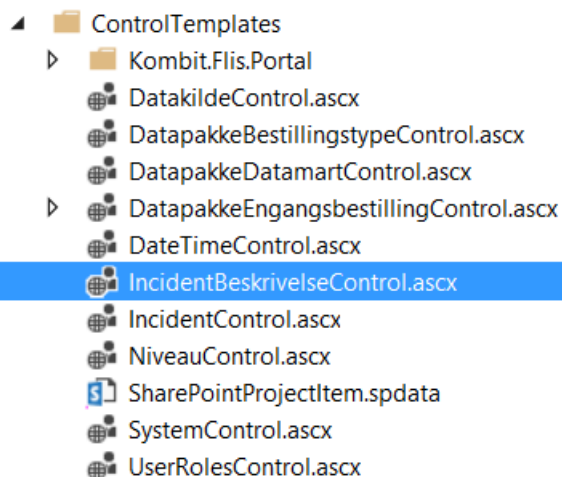
- jQuery
- Grunt
- RequireJS
- Underscore
- SPServices

2. ARKITEKTUR

SharePoint løsningen er struktureret i 3 SharePoint (wsp) pakker:

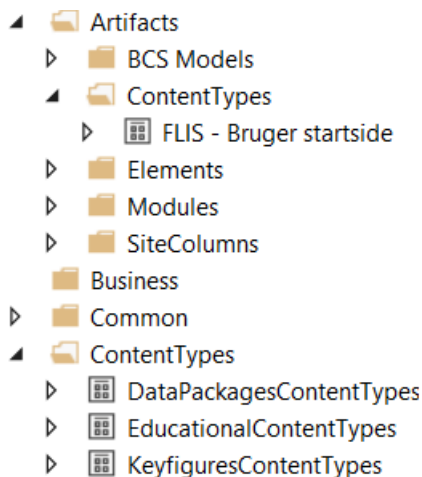
- Kombat.Flis.Portal
- Kombat.Flis.PortalDefinition
- Kombat.Flis.SiteDefinitions

Alle løsningskomponenter fra "D0150 - Softwarearkitektur" afsnit 4.2.1 er implementeret i pakken Kombat.Flis.Portal, løsningen er grupperet efter artefakt type:



Det kan derfor være svært for en ny udvikler direkte og danne sig et overblik over præcis hvor koden til de enkelte komponenter ligger.

Det er også uklart hvordan den præcise opdeling er fordi ting som måske kunne være grupperet sammen også er fordelt under flere foldere, f.eks. indholdstyper:



En mere naturlig opdeling af pakker kunne måske være en pakke per løsningskomponent.

De to sidste pakker implementere portal strukturen fra afsnit 4.3 i softwarearkitektur dokumentet hvor man får indtryk af at der er tale om to portaler. Men i PortalDefinition virker det som at der ligger en ekstra form for portal som ikke er beskrevet i dokumentet – en Kombat.Flis.CommunityPortalDefinition.

2.1. KOMBIT.FLIS.PORTAL

Hele pakken `Komit.Flis.Portal` er tæt koblet, og der er ikke brug koncepter som f.eks. dependency injection. Hvis der var taget udgangspunkt i guidelines fra Microsoft <https://msdn.microsoft.com/en-us/library/ff798371.aspx> ville man kunne skabe en mere afkoblet arkitektur.

I pakken er der ligeledes implementeret en `LoggingService`, servicen skriver til SharePoint's ULS log. Klassen er implementeret på den måde man må forvente af en SharePoint 2010 løsning.

2.2. SCRIPTS

Udover selve pakkerne findes der også en stor mængde scripts. Formålet med disse scripts er deployment af løsningen, men det virker også som at delkomponenter ligger f.eks. i `FLIS41\source\web\sharepoint\Komit.Flis.Portal\Scripts\Modules`. Opdelingen mellem hvad der f.eks. er implementeret som en feature receiver og hvad der er et script er svær at gennemskue.

2.3. SHAREPOINT TILPASNINGER

SharePoint er blevet tilpasset i større grad, løsningen har lavet custom tilpasningerne af typerne:

- `SiteDefinitions`
- `CustomFieldTypes` (`Komit.Flis.Portal.Common.CustomFieldTypes`)
- `DelegateControls`
- `HttpModules`
- `Layouts Pages`
- `EventReceiver`

Mange af disse tilpasninger er tidssvarende for da løsningen er lavet, men hvis man vil op på en nye version af SharePoint, måske med en cloud-first tilgang, så må man forvente at der skal investeres en del tid i en sådan opgradering.

3. KILDEKODE

Der er ikke fundet beskrivelser i dokumentationen for hvilke kode standarder løsningen skal overholde.

Mange steder i koden er sproget en blanding mellem dansk og engelsk, f.eks.:

```
protected void EditComment(object sender, EventArgs e)
{
    var button = sender as Button;
    CurrentCommentId = int.Parse(button.CommandArgument);
}
```

```
protected void Tilbage(object sender, EventArgs e)
{
    var source = HttpContext.Current.Request.QueryString["source"];
}
```

Det er ikke optimalt hvis vedligeholdelse af løsningen skal udbydes til en udenlandsk partner, eller løsningen skal vedligeholdes af en person med mindre gode dansk egenskaber.

3.1. C# KODESTANDARDE

De fleste steder overholdes normale navne og kodestandarder for C#. Flere steder er man dog ikke helt konsekvent med brug af var, f.eks:

```
private static StringBuilder CreateResetPasswordMailBody(string navn, string newPassword)
{
    StringBuilder mailSignature = MailSignature();
    var sb = new StringBuilder();
    sb.Append("Kære " + navn);
}
```

og casing (burde have været stort C):

```
public string createFLISLink(string lastBody)
{
}
```

Der er også forskel på hvilke type mellemrum man har brugt (tabs vs. spaces):

```
namespace Kombit.Flis.Portal.Common.Helpers
{
    |> public class iPadRolloutHelper
    |>     {
    |>     |>     ///

```

Mængden af kommentarer er meget forskellig, nogle klasser indeholder kun standard autogenerated kommentarer, andre ingen. Men der findes også klasser med fint XML documentation comments.

3.2. POWERSHELL KODESTANDARDE

Ser fornuftig ud, flere steder kunne man dog godt have fulgt Verb-Noun standarden (<https://technet.microsoft.com/en-us/library/dd315315.aspx>).

3.3. JAVASCRIPT KODESTANDARDE

Alt javascript kode befinder sig i Kom-bit.Flis.Portal\Kombit.Flis.Portal\LAYOUTS\Kombit.Flis.Portal\UI\js koden standarden ser fornuftig ud.

Men jQuery burde have været sat i noConflict mode for at undgå sammenfald med SharePoints standard definition af \$ (<http://www.dotnetmafia.com/blogs/dotnettipoftheday/archive/2009/08/03/quick-reminder-to-use-noconflict-with-jquery-in-sharepoint.aspx>).

3.4. ASP.NET WEBFORMS

Web kontrollerne er lavet på den klassiske ASP.Net WebForms måde, man kunne måske forvente at en løsning af dette omfang var lavet efter et MVW (Model-View-Whatever) princip (f.eks. <http://www.codeproject.com/Articles/605662/Using-MVP-with-ASP-NET-Web-Forms> og <https://github.com/webformsmvp/webformsmvp>)

3.5. SPMETAL

Der er dannet SPMetal data kontekst som ligger i Kom-bit.Flis.Portal\Kombit.Flis.Portal\DataAccess\LinqContext

Konteksten bliver brugt f.eks.:

Kombit.Flis.Portal.LAYOUTS.Kombit.Flis.Portal.ApplicationPage1

```
var url = KommuneHelper.GetAdminPortalUrl() + "/UddannelseOgVejledning";
using (var entity = ContextHelper.GetAdministrationUddannelseContext(url, true))
{
    var context = HttpContext.Current;
    HttpContext.Current = null;
    var kurser = entity.Kurser.Where(k => k != null && k.Status != Status.Aflyst &&
k.Status != Status.Gennemført);
```

Men alligevel fraviger man i samme klasse også brugen og tilgår SharePoint direkte:

```
SPList listKurser = root.Lists["Kurser"];
SPListItem kursus = listKurser.GetItemById(kursusNavnId);
if (kursus != null)
{
    string lokationTmp = kursus["CourseLocation"].ToString();
    kursusLokationNavn = lokationTmp.Substring(lokationTmp.IndexOf('#') + 1);
    KursusLokationLbl.Text = kursusLokationNavn;
    KursusDatoLbl.Text = DateTime.Parse(kursus["StartDate"].ToString()).ToShortDateString();
```

3.6. KOBLING TIL NETCOMPANYS CRM SYSTEM

Dokumentet "D0130 Logisk Datamodel" beskriver i afnist 7 at fejl og ændringer sendes direkte fra FLIS Portal til Netcompanys CRM system. BCS modellen i Kombit.Flis.Portal\Kombit.Flis.Portal\Artifacts\BCS Models\IncidentModel er derfor stærkt afhængig af dette eksterne system.

4. SEMANTIK

Afsnit beskriver konceptuelle udfordringer ved kildekoden.

4.1. FLIS_LOGGER

Der er fundet en FLIS_Logger klasse hvor alle metoderne er tomme. Løsningen indeholder allerede en anden logger – LoggingService.

Kursus delen i løsningen kalder FLIS_Logger:

```
new FLIS_Logger("FLIS", "FLIS log").WriteToEventLog("Kursusdetaljer: Der fandtes enten ingen eller flere kursusmoduler i kursusmodul-listen med følgende navn: " + kursusModulNavn + ", da " + SPContext.Current.Web.CurrentUser.LoginName + "forsøgte at få vist kursusdetaljer.", "error");
```

Og det virker derfor til at disse fejlbeskeder bliver slugt af systemet.

4.2. DISPOSE

Flere steder i løsningen kan der findes eksempler på at SPWeb og SPSite objekter ikke frigives korrekt.

F.eks.:

```
public class LayoutEventReceiver : SPFeatureReceiver
{
    public override void FeatureActivated(SPFeatureReceiverProperties properties)
    {
        SPSite site = (SPSite)properties.Feature.Parent;

        using (SPWeb web = site.RootWeb)
        {
```

Og

```
public class iPadRolloutHelper
{
    public static void RollOutIPadAppFile(SPWebApplication webApplication, SPFile sourceFile)
    {
        LoggingService.LogVerbose(string.Format("Beginning rollout of iPad-file {0}", sourceFile.Name));
        SPSecurity.RunWithElevatedPrivileges(() =>
        {
            foreach (SPSite site in webApplication.Sites)
            {
```

Er direkte modstridende i forhold til de guidelines som microsoft har lavet:

[https://msdn.microsoft.com/en-us/library/office/ee557362\(v=office.14\).aspx](https://msdn.microsoft.com/en-us/library/office/ee557362(v=office.14).aspx)

Andre steder i løsningen (f.eks. SiteCollectionIteratorService):

```
foreach (SPSite spSite in webApplication.Sites)
{
    try
    {
        ...
    }
    finally
```

```

    {
        spSite.Dispose();
    }
}

```

Virker det dog til at man er udmærket klar over disse guidelines.

Et værktøj som <https://gallery.technet.microsoft.com/office/SharePoint-Dispose-Checker-01da48e8> burde man forvente var brugt til udviklingen, men det virker ikke som tilfældet.

Konsekvensen ved ikke at bruge dispose korrekt er at hukommelsen for Application Pool'en vil blive fyldt og på et tidspunkt recyclet.

4.3. ELEVATE

Når rettigheder skal forhøjes bruges `RunWithElevatedPrivileges`. Mange steder bruges den dog ikke korrekt, f.eks. (fra [ManuelDataLeveranceJob](#)):

```

using (SPSite site = new SPSite(siteCollection.Url))
{
    SPSecurity.RunWithElevatedPrivileges(() =>
    {

```

Andre steder kaldes metoden korrekt, og der er tilmed en kommentar til hvorfor det skal gøres sådan ([AfmeldKursus](#)):

```

protected void AfmeldKursusBtn_Click(Object sender, EventArgs e)
{
    var centralUrl = KommuneHelper.GetAdminPortalUrl();

    // XTZ FLIS-1744: Moved the RunWithElevatedPrivileges to BEFORE we open the Admin-
    // site, hence giving the user the required credentials..
    SPSecurity.RunWithElevatedPrivileges(() =>
    {
        using (var site = new SPSite(centralUrl))
        {

```

Se [https://msdn.microsoft.com/en-us/library/office/aa543467\(v=office.14\).aspx](https://msdn.microsoft.com/en-us/library/office/aa543467(v=office.14).aspx) for korrekt brug (frequently, to perform actions in SharePoint, you must get a new `SPSite` object to effect the changes.).

Når `RunWithElevatedPrivileges` bruges er det også vigtig at validere på om den nuværende bruger faktisk har burde kunne udføre den handling som skal gøres. F.eks. virker det som at der i [AfmeldKursus](#) gives lov til at alle kan slette elementer fra listen `Kursustilmeldinger`:

```

SPSecurity.RunWithElevatedPrivileges(() =>
{
    using (var site = new SPSite(centralUrl))
    {
        using (var root = site.AllWebs["UddannelseOgVejledning"])
        {
            root.AllowUnsafeUpdates = true;

            //Sletter listelementet med det id, som står i query string
            int courseToDeleteId = Convert.ToInt32(Request.QueryString["k"]);
            SPSList listKursustilmeldinger = root.Lists["Kursustilmeldinger"];
            listKursustilmeldinger.Items.DeleteItemById(courseToDeleteId);
            listKursustilmeldinger.Update();

```

4.4. LOKAL UDVIKLING

Der er fundet hardcoding af en IP:

```
// .....[Necessary hack for local develop-
ment].....
var host = SPContext.Current.Web.Url.Contains("10.0.29.43") ? "http://localhost" :
SPContext.Current.Web.Url;
```

Det virker som en IP der bliver brugt i forhold til udvikling, hvis udviklingsservere skal flyttes på et tidspunkt vil man måske også skulle ændre denne IP.

Som koden er skrevet kommer IP'en også med i produktions koden, her kunne man med fordel have brugt compiler direktiver (<https://msdn.microsoft.com/en-us/library/4y6tbswk.aspx>).

4.5. SQL OG MDX

SQL og MDX er hardcoded i løsningen, typisk vil man nok have brugt stored procedures eller et Object-relational mapping framework for at opnå bedre abstraktion.

I klassen `Begrebsbeskrivelse` overføres der direkte argumenter fra query string til mdx kommando. Det har måske ikke den store konsekvens, men virker lidt bekymrende.

4.6. FEATURE RECEIVERS

Features udgøre funktionalitet som kan aktiveres og deaktiveres. Når man aktivere en feature og aktiveringen går godt, vil man som bruger eller udvikler forvente funktionaliteten blev tilføjet. Men flere af de features som er blevet lavet sluger fejl, f.eks. (`CommunityBlogModifierEventReceiver`):

```
try
{
    web.MasterUrl = "/sites/community/_catalogs/masterpage/fliskommune.master";
    web.CustomMasterUrl = "/sites/community/_catalogs/masterpage/fliskommune.master";
    web.Update();
}
catch (Exception ex)
{
    Logging.WriteException(ex, "CommunityBlogModifier.EventReceiver.cs:FeatureActivated() - Customize Branding");
}
```

Burde have været lavet med `throw` efter logging. En konsekvens kan være at man som udvikler skal ind og læse logs for at finde ud af hvad der er gået galt.

4.7. INTERFACES OG SNITFLADER

Kildekoden indeholder kun et enkelt interface (`IActiveDirectory`), og det kan derfor konkluderes at der ikke er lagt meget energi i at beskrive snitflader mellem komponenter.

Dokumentationen beskriver heller ikke snitflader mellem komponenter.

5. STIKPRØVER

Dette afsnit indeholder stikprøver på dokumentet "D0130 Logisk Datamodel" afsnit 3 om FLIS entiteterne.

Stikprøverne indeholder eventuelle forskelle mellem dokumentation og kildekode.

5.1. STIKPRØVE "3.1 FLIS STANDARDRAPPORT"

Navn: "FLIS - Standardrapport" imod "FLIS Standardrapport"

Attributter:

- Standard er tvungen i koden, men ikke i dokumentation
- Ansvarlig Person er tvungen i koden, men ikke i dokumentation
- Rapporttype hedder Type i koden
- Rapporttype valgmuligheder i dokumentation: Analyserapport, Benchmarkrapport vs. koden: Benchmarking, Analyse
- Felt i kode som ikke er i dokumentation: FlisRapportId

5.2. STIKPRØVE "5.1 KOMMUNE"

Navn: " FLIS - Kommune" imod "Kommune"

Attributter:

- Kommunnavn er nedarvet fra Element (navn blot ændret)
- KommuneID hedder Kommune ID i koden
- Valg til Region passer ikke præcis sammen med koden
- Valg til Aktuel tilmeldingsstatus passer ikke præcis med koden.
- Godkendt dato er tvungen i dokumentation med ikke i koden
- Kommentar er tvungen i dokumentation men ikke i koden
- Forretningsansvarlig hedder Forretningsansvarlig – Kommune i koden
- Teknisk ansvarlig hedder Teknisk ansvarlig – Kommune i koden
- Forretningsansvarlig type passer ikke, dokumentation siger Opslag (flere) men er kun enkelt
- Teknisk ansvarlig type passer ikke, dokumentation siger Opslag (flere) men er kun enkelt
- Teknisk kontakt – Netcompany type passer ikke, dokumentation siger Enkelt tekstlinje men er opslag

Dokumentet "D0160 Brugergænseflade administrationsportal" afsnit 2.2 viser:

Det bemærkes at der findes to "Teknisk kontakt - Netcompany" på skærbilledet. Ifølge dokumentationen burde der kun finde et tekstlinje felt, men der findes to, en af type User og en af typen Url.

Følgende script er fundet der tilføjer feltet: Scripts\Administration.ModifyKommuneList.ps1

Både kildekoden (pakkerne) indeholder derfor felter og scripts indeholder felter, det er ikke klart hvorfor noget ligger i scripts og andet ligger i pakker.

5.3. STIKPRØVE "8.1 DATAPAKKEBESTILLING"

Navn: "Datapakkebestilling" imod "FLIS - Datapakkebestilling"

Attributter:

- Område er tvungen i dokumentation men ikke i koden
- Område type er uklar fordi valg i dokumentation men taxonomy i kode
- Felt i kode som ikke er i dokumentation: Status

6. AUTOMATISK TEST (UNIT/INTEGRATION)

Under gennemgangen blev der ikke fundet udpræget grad af automatisk test. Det eneste der blev fundet var en enkelt test i FLIS41\source\test\SeleniumFrontend der omhandlede nøgletal for kommuner.

7. KONKLUSION

- SharePoint er tilpasset i stor grad.
- Det er tydeligt at der ikke er brugt ensartede kodestandarder igennem hele løsningen.
- De klassiske udfordringer omkring SharePoint 2010 server side udvikling i forhold til elevate og dispose er løst tvivlsomt.
- Dokumentation omtaler komponenter men der er ingen direkte relation mellem pakker eller namespaces og disse komponenter.
- Løsningen indeholder ikke begreber som Dependency Injection eller Model View Whatever og er derfor forholdsvis hårdt koblet. Hvis man på et tidspunkt ønsker at skrive automatiske tests til løsningen kan dette være svært fordi der ikke er introduceret disse former for abstraktion.
- Alt er samlet i få SharePoint pakker, det kan derfor være svært at finde rundt i den store pakke hvis man kun skal ændre i en af komponenterne.
- Stikprøver fra "Logisk Datamodel" holdt op imod kildekoden antyder at disse ikke stemmer overens.