|  |  |  |
| --- | --- | --- |
| Q:\PPT\SAMLING\JV\Logo2013\LogoSortDK.gif |  | 13. august 2018  EWM/HGS/HAS/KHO/PGL  IT |

Systemdokumentation af <<NAVN>>

Indholdsfortegnelse

[0 Intro 2](#_Toc523902237)

[1 Indledning 2](#_Toc523902238)

[1.1 Udvikler info 2](#_Toc523902239)

[1.2 Læsevejledning 2](#_Toc523902240)

[1.3 Forretningsbeskrivelse af systemet 3](#_Toc523902241)

[1.4 Forretningsprocesser systemet indgår i 3](#_Toc523902242)

[1.5 Begreber, tilstande og primære usecases 3](#_Toc523902243)

[2 Overordnet IT-arkitektur 3](#_Toc523902244)

[2.1 IT-Arkitektur 4](#_Toc523902245)

[2.2 Platforme og deployment 4](#_Toc523902246)

[2.3 Værktøjsvalg 4](#_Toc523902247)

[2.4 Versionering 4](#_Toc523902248)

[2.5 Øvrige standarder 4](#_Toc523902249)

[3 Overordnet Design 4](#_Toc523902250)

[3.1 Anvendte designprincipper 4](#_Toc523902251)

[3.2 Komponenter 5](#_Toc523902252)

[3.3 Datamodel 5](#_Toc523902253)

[3.4 Integrationer 5](#_Toc523902254)

[3.5 Brugergrænseflader 5](#_Toc523902255)

[3.6 Sikkerhed 5](#_Toc523902256)

[4 Detaljeret design 6](#_Toc523902257)

**Figuroversigt**

1. ["New diagram" dialog i EA 2](#_Toc520458133)

**Versionslog**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Dato** | **Sammenfatning af ændringer** | **Ændringer markeret**  **(ja/nej)** |
|  |  |  |  |
|  |  |  |  |

# 0 Intro

**Systemdokumentationen** er tiltænkt de it udviklere som skal fejlrette og vedligeholde it systemet.

Når der udvikles NYE systemer til blivende brug, skal der afsættes tid til at udarbejde systemdokumentation med udgangspunkt i det udarbejdede it-arkitekturdokument. Ambitionen er at systemets opbygning og struktur skal beskrives, så en udenforstående, med kendskab til DST kan få et overblik over systemet.

Det er IT udviklernes og evt. it projektleders pligt at eventuelle efterfølgende ændringer i it-systemet afspejles i systemdokumentationen.

# Indledning

*Ganske kort information om, hvilket system det drejer sig om.*

*Hvem har været udviklere på projektet.*

*Hvilket kontor ejer systemet.*

## Udvikler info

Se de generelle oplysninger om systemet i skabelon O. Her følger de mest essentielle udvikler-rettede oplysninger om systemet:

Hvilke **teknologier** anvendes (SAS, Oracle DB, PLSQL, Excel, VB mv.):

Hvis der anvendes Oracle, hvad er **projektnummeret**:

Hvis der er data i filer, hvad hedder de filer og hvordan er **katalogstrukturen**:

Hvad skal man have **adgang** til, som **udvikler** (mapper, Oracle):

Hvad skal man have **adgang** til som **bruger**:

Hvilke **miljøer** er der (udvikling, test, produktion mv) – og hvor ’bor’ disse:

Er der **automatiseret test**, fx unit test – hvor og hvordan:

Hvor ligger systemets krav-, design-, driftmanual mv. **dokumenter**:

Hvis der anvendes **Enterprise Architect**, hvad hedder eap filen og hvor er den:

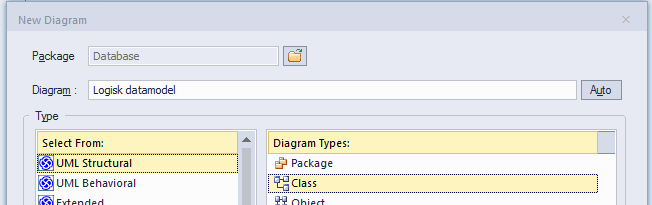
Hvis der anvendes **subversion** til koden, hvor i subversion:

Hvis der anvendes standarder for program-**fil navne** eller brug af **kataloger**, hvilke:

## Læsevejledning

*Ting skrevet i kursiv Ariel (som denne tekst) er vejledning og hjælp til udarbejdelsen og skal derfor slettes i den konkrete færdige systembeskrivelse. Ting skrevet i Normal typografi (som ovenstående tekst) er forslag til tekst, mest indledninger i kapitlerne, og kan omformuleres eller bevares efter behov.*

*Henvisninger til diagramtyper er skrevet som henvisning til hvor i Enterprise Architect (EA) nye diagrammer findes; fx ”UML Structural - Class diagram”, hvor første navn er valget i første kolonne, andet navn er valget i anden kolonne i nedenstående ”New Diagram” dialog fra EA:*



1 "New Diagram" dialog i EA

## Forretningsbeskrivelse af systemet

*Ud over den korte intro til systemet i Skabelon O, kan man her kan skrive en lidt mere udførlig forretningsbeskrivelse af systemet, eller henvises til andre dokumenter som kan være nyttige at læse.*

*Undgå at gentage lange beskrivelser, om nødvendigt skrives kun en opsummering her.*

## Forretningsprocesser systemet indgår i

*Her indsættes kort beskrivelse og arbejdsprocesdiagrammer for de forretningsprocesser som systemet indgår i.*

*Der skal være en henvisning til komplet arbejdsprocesdokumentation for de forretningsprocesser som systemet indgår i (hvis denne er lavet).*

*Anvendte diagramtyper:*

*Arbejdsprocesdiagram: BPMN 2.0 - Business Process*

## Begreber, tilstande og primære usecases

Liste med de primære begreber systemet arbejder med (fx person, familie, uddannelse):

*Logisk datamodel: UML Structural – Class*

Liste over de primære usecases – hvad brugerne bruger systemet til (fx indlæs data, fejlsøg data, analyser data, publicer data):

*Use Case overblik: UML Behavioral – Use Case*

Hvis relevant – vis de tilstande en dataforekomst kan være i og de overgange der kan være:

*Tilstandsdiagram: UML Behavioral – State Machine*

# Overordnet IT-arkitektur

I dette kapitel beskrives systemets overordnede IT-arkitektur.

## IT-Arkitektur

*Her skal gives en beskrivelse af systemets arkitektur. Dette kan indeholde elementer som lagdeling, opdeling i komponenter.*

*Fokus skal være på formidling og forståelighed og ikke formel overholdelse af diagramstandarder. Derfor er der stor frihed i diagrammerne her.*

*Anvendte diagramtyper:*

*Whiteboard diagram: Whiteboard – Whiteboard diagram*

## Platforme og deployment

*I dette afsnit beskrives miljøer for udvikling, test og produktion, samt overvejelser om adskillelse mellem udvikling og drift.*

*Dette er diagrammer af deployment, der viser hvordan systemets komponenter skal placeret på forskellige servere og databaser. Deployment diagrammet kan være et detaljeret UML Deployment diagram, men et simplere Whiteboard diagram kan ofte anvendes i stedet.*

*Anvendte diagramtyper:*

*Deployment diagram: UML Structural – Deployment*

*ELLER Whiteboard diagram: Whiteboard – Whiteboard diagram*

## Værktøjsvalg

*En kort beskrivelse af værktøjer i systemet skrives her. Værktøjer er både programmer (fx udviklingsværktøjer) og programmeringssprog. Begrundelser for værktøjsvalg bør holdes kort; fokus er hvilke værktøjer bruges til hvad.*

## Versionering

*En beskrivelse af hvordan subversion er brugt i dette system. Hvis der er lavet en selvstændig længere beskrivelse skal der henvises til denne og her kun skrives en opsummering af principperne.*

*Hvis det er kildetekster eller scripts uden for subversion (fx af hensyn til drift) skal principperne for hvordan disse ændres beskrives her. Fokus for beskrivelsen er er bevaring af konsistens med subversion.*

## Øvrige standarder

*Her skrives en beskrivelse af øvrige standarder der benyttes i udviklingen og i systemet, herunder udviklingsstandarder, navngivningsstandarder og lignende.*

*Også specifikt hvordan systemet testes. Er der fx unit tests, særlige forhold vedr test servere eller lignende?*

# Overordnet Design

I dette kapitel beskrives systemets overordnede design.

## Anvendte designprincipper

*Her beskrives de designprincipper der er brugt i designet, Dette kan fx være ”Adskillelse af grafiske grænseflader og forretningslogik (lag-model)”, ”Valget af historikmodel” eller ”Placering af data i Oracle database”.*

*Figurtyper her bør vælges så de bedst matcher det område princippet relaterer sig til. Database diagrammer til principper der har fokus på data; Komponent diagrammer eller Whiteboard diagrammer til lag-model illustrationer. Fokus i illustrationerne her er at vise hvilket designprincip der er benyttet og behøver derfor ikke at vise systemet komplet her.*

*Anvendte diagramtyper:*

*Komponent diagram: UML Structural – Component*

*Whiteboard diagram: Whiteboard – Whiteboard diagram*

*Database diagram: Extended – Data Modeling*

## Komponenter

*Komponenter er delsystemer, hvor alle dele skal deployeres samlet. Eksempler på komponenter kan være ”Send Mail komponent”, ”Driftslogger dannelse og administration” og ”Grænseflade mod noget system-eksternt data”. Et veldesignet system består af mange komponenter, således at forbedringer af systemet kan foretages ved erstatning af kun dele af systemet frem for ændring af hele systemet.*

*Her beskrives de komponenter systemet består af og hvordan disse bruger hinanden. Dette kan beskrives i et komponent-diagram*

*Anvendte diagramtyper:*

*Komponent diagram: UML Structural – Component*

## Datamodel

*Her beskrives:*

1. *Den logiske datamodel (hvis man har lavet en sådan i projektet)*
2. *Den fysiske datamodel for de vigtigste data i systemet. Det er ikke nødvendigt at beskrive alle komponenternes mellemdata. For data i Relationel database (fx Oracle) kan dette bedst gøres med Database diagrammer. For data der ikke gemmes relationelt kan benyttes den diagramtype der bedst beskriver datamodellen. SAS data kan ofte med fordel beskrives med Database diagrammer.*

*Anvendte diagramtyper:*

*Database diagram: Extended – Data Modeling*

## Integrationer

*Her beskrives alle grænseflader med andre systemer på et overordnet niveau.*

*Andre systemer kan levere inddata til dette system, kan bruge uddata fra dette system eller kan være tæt integreret i selve behandlingen i dette system; Alle tre typer af integrationer skal beskrives.*

## Brugergrænseflader

*Her kan man beskrive brugergrænsefladerne i systemet, grafiske brugergrænseflader kan vise som skærmdumps eller mockups. Dette kan være meget omfattende i nogle systemer, man kan undlade det eller henvise til anden dokumentation.*

## Sikkerhed

*Nedenstående række af spørgsmål skal besvares. Hvis spørgsmål er irrelevante, besvares de med N/A*

* Krav til it-sikkerhed
  + Hvilken risikovurdering er der foretaget af systemet?
  + Hvordan sker eventuel ekstern adgang til systemet (fx via webgrænsefalder mv)?
  + Hvordan er adgangen sikret mod uautoriseret adgang? Beskriv hvordan
  + Modtager systemet data udefra (fx via XIS eller CEMOS), eller modtages der data ad andre kanaler? Hvis andre kanaler benyttes, skal der redegøres for sikkerheds niveauet i datamodtagelsen?

Hvordan er adgangsstyring på systemet?

* + Hvordan er adgangsstyring administreret?
  + Er der defineret differentierede roller i systemet? Hvilke
  + Hvordan er der logning i systemet etableret, og er logningen tilstrækkelig til at eventuelle uautoriserede adgange vil kunne afdækkes? Beskriv logningen
  + Hvorledes opbevares passwords til systemet? Beskriv hvordan
  + Har man sikret sig at passwords ikke er hardcoded, eller på anden måde kan blive afsløret? Beskriv hvordan
* Krav til datasikkerhed
  + Er det verificeret at eventuelle beregningsalgoritmer regner rigtigt. Hvis ja, så redegør for hvorledes denne verifikation er sket
* Krav til datafortrolighed
  + Hvilke typer data indeholder systemet personfølsomme oplysninger? Hvis systemet indeholder personfølsomme oplysninger så redegør for hvordan de opbevares
  + Er personfølsomme oplysninger pseudonymiseret? Hvis nej, så angiv årsagen
  + Hvilke data eksponeres gennem systemet? Findes der en beskrivelse af disse, henvis til beskrivelsen
  + Hvilke andre systemer i huset anvender systemets data? Hvis ja, angiv hvem/hvilke systemer
  + Afleveres mikrodata til eksterne? Hvis ja, så angiv hvem
  + Anendes ”levende” data til test? Hvis ja, er det så givet tilladelse til det, og er det sikret at disse data behandles fortroligt?

*.*

# Detaljeret design

*Detaljeret design beskrivelser af komponenterne bør kun inkludere specielle eller komplicerede dele af systemet. Selvdokumenterende programmer bør ikke beskrives. Processer kan beskrives med aktivitets diagrammer (ikke BPMN, da det er rettet mod forretningsprocesser og ikke velegnet til it-processer) og beskrivelser af kompliceret samspil og timing mellem komponenter og dele af komponenter kan bruge med sekvens diagrammer.*

*Anvendte diagramtyper:*

*Aktivitets diagram: UML Behaviour – Activity*

*Sekvens diagram: UML Behaviour – Sequence*