# Design og Arkitektur beskrivelse af Seal.Net til .net 4.5

## Introduktion

At designe og implementere en service der overholder Den Gode Webservice (DGWS), er ikke nødvendigvis en let opgave. Derfor er Seal.Net's formål at indpakke DGWS specifikke detaljer, og abstrahere alle typer fra XML til objektform, for på den måde, at gøre det lettere for udvikleren at overholde standarden.

## Design

Seal.Net bygger på WCF og WIF. Disse er valgt, da de er supporteret af Microsoft, og indgår som standard i .Net Frameworket. I en WSDL der beskriver en snitflade til DGWS indgår skemaer der beskriver de specifikke DGWS klasser. Når der genereres en proxy til hhv. klient eller server, dannes disse klasser på typestærk form i den autogenererede proxy. Eksempler på genererede klasser er Security, Assertion og Header. Dette Api er designet til at benytte disse klasser i videst mulig omfang.

## Objekter

De indgående objekter i Seal.Net kan inddeles i følgende grupper:

### Klienter

Indbefatter Seal2SamlStsClient og Saml2SealStsClient, som kan konvertere en SealAssertion til en SamlAssertion, og vice versa.

### Kort

Indbefatter SealCard of DgwsHeader som benyttes til at indpakke data fra hendholdsvis Security Token Service (STS) og MedCom.

### MessageHeaders

Indbefatter DgwsMessageHeader og SealCardMessageHeader, som kan konvertere henholdsvis DgwsHeader og SealCard til en WCF MessageHeader

### Endpointsbehaviors

Disse kan validere de mest basale DGWS-krav på udkomne og indgående beskeder. De er endvidere i stand til at udpakke fejl fra en service og lave dem til exceptions.

### Hjælpefunktioner

Indbefatter SealUtilities, som indeholder forskellige funktioner, til f.eks. signering og validering.