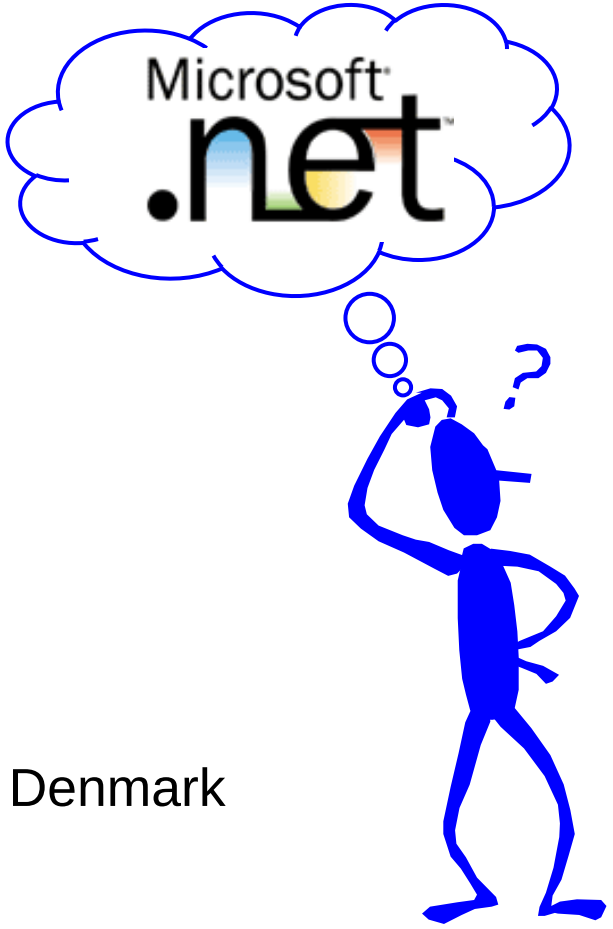
A pragmatic approach to creating services using Windows Communication Foundation



Captator
Tlf: +45 8620 4242
www.captator.dk

Henrik Lykke Nielsen
Softwarearkitekt, Microsoft Regional Director for Denmark
lykke@captator.dk
Mobile: +45 2237 3311

Agenda

Captator

- **Goals**
- **WCF based communication**
- **Requests and responses**
- **Service Implementation**
  - ServiceExecutor
  - Multitenancy
  - Authentication
  - Validation
  - Logging
- **Test**
- **Documentation**

Our Goals

Captator

- **The service model…**
  - should make it easy to reuse service implementations
  - should make it easy to implement centralized logic
  - should support a strict separation of domain and generic logic
  - should only impose a minimal overhead when implementing new service operations
  - should make it easy to validate requests
  - must be secure - the services must be easily securable
  - should be scalable
  - should make the services easily testable
  - should support (automatically generated) service documentation

Communication

Captator

- **Communication Patterns**
  - SOAP
  - XML/JSON over HTTP - URLs denotes operations
  - Simple .NET method calls
- **SOAP and HTTP headers (and other transport specific mechanisms) are only used for transport related issues**
- **Request/response based service definitions**

Technologies

Captator

- **WCF (Windows Communication Foundation)**
- **Various Clients such as**
  - ASP.NET, Windows clients, test clients
    - Network access / simple method calls
  - Silverlight, mobile clients
    - Network access
- **Hosting**
  - IIS / self hosting
  - Standard Windows Server / Windows Azure / …

Communication using WCF

Captator

- **Service definition**
  - A service contract is specified by defining an interface decorated by attributes
- **Service implementation**
  - A service is implemented by implementing the contract (the interface)
- **WCF supports**
  - ServiceHost: SOAP
  - WebServiceHost: XML/JSON over HTTP
    - We primarily use POST (WebInvoke)
    - We occasionally use GET (WebGet) for manual browser execution and for limited clients

WCF Contracts and Implementation

- **Services are specified by the ServiceContract-attribute**
- **Operations are specified by the OperationContract-attribute and the WebInvoke-/WebGet-attributes**
- **Contract**

```
[System.ServiceModel.ServiceContract(Name = "SystemService")]
public interface ISystemService
{
   [System.ServiceModel.OperationContract()]
   [WebInvoke(UriTemplate = "GetCountries")]
   GetCountriesResponse GetCountries(GetCountriesRequest request);
```

SOAP

XML/JSON over HTTP

- **Implementation**

```
GetCountriesResponse ISystemService.GetCountries(
                                        GetCountriesRequest request)
{ /* ... */ }
```

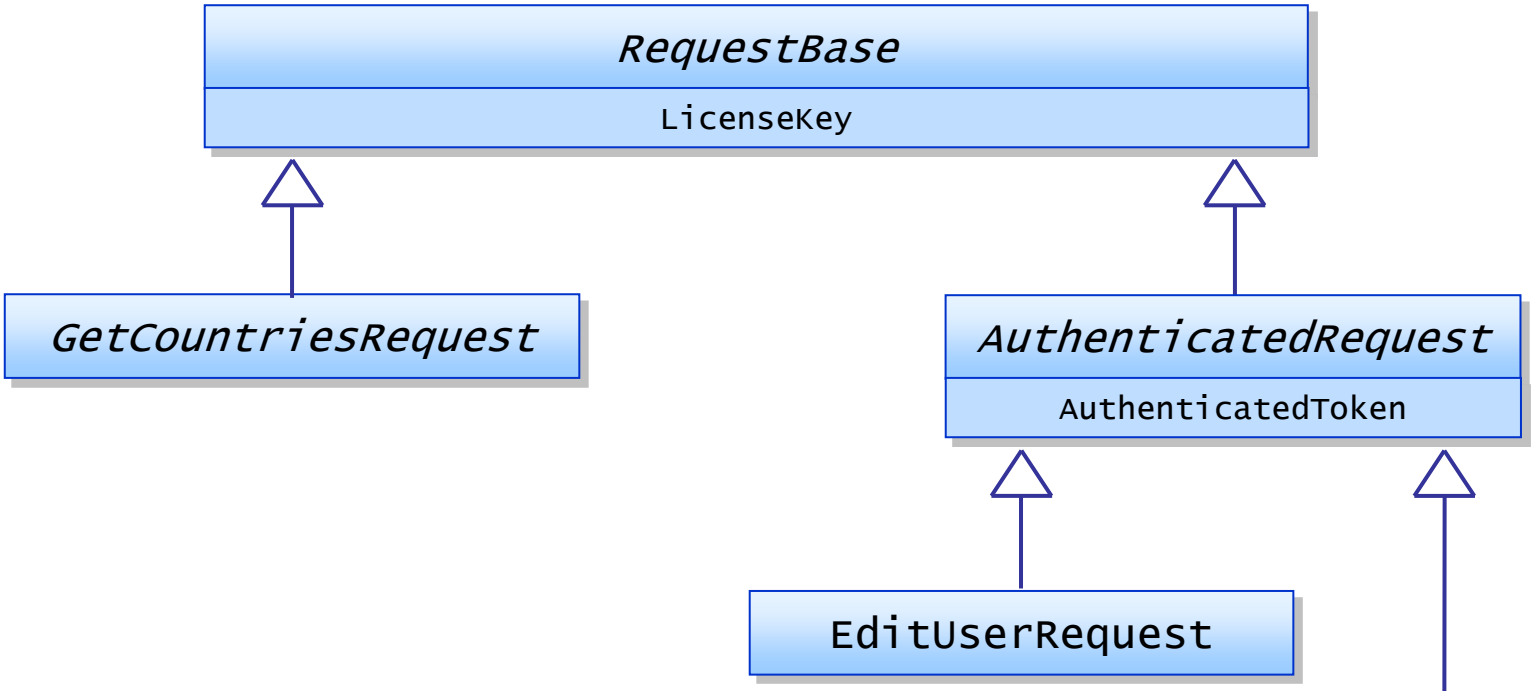
WCF Web Message Formats

Captator

- **WebServiceHost defines a**
  - WebHttpEndpoint.AutomaticFormatSelectionEnabled property
- **We set the response format using our alternative SetWebMessageFormat-method based on**
  1. the “format” query string parameter
     http://captator.com/Services/1/SystemService/GetCountries?format=json
     http://captator.com/Services/1/SystemService/GetCountries?format=xml
  2. the client request’s HTTP accept header
  3. the client request’s HTTP content type
  4. the default format set on the WCF host

Requests

- **Input values are wrapped in a request-object**

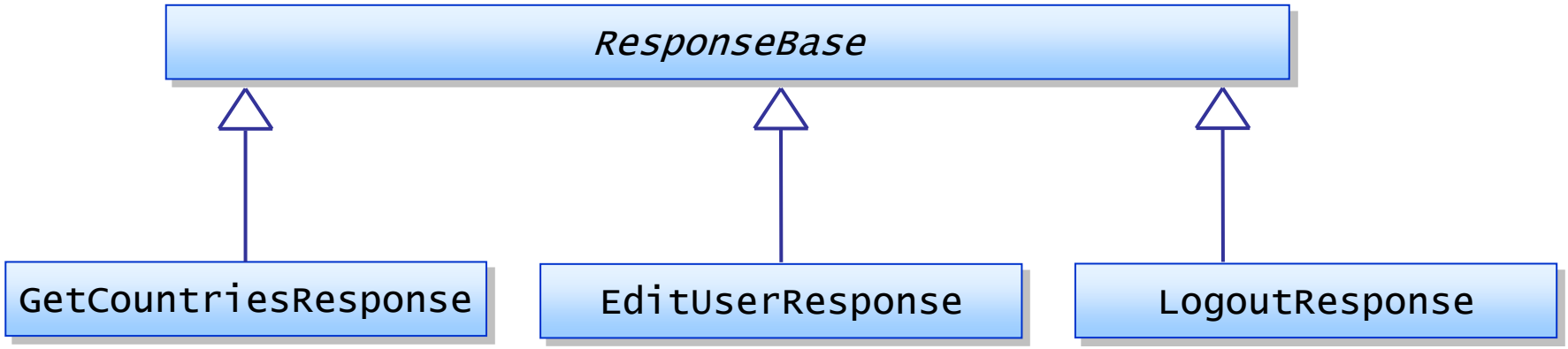



```
public class EditUserRequest : AuthenticatedRequest
{
  public int FirstName { get; set; }

  // ...
}
```

Responses

- **Return values are wrapped in a response-object**

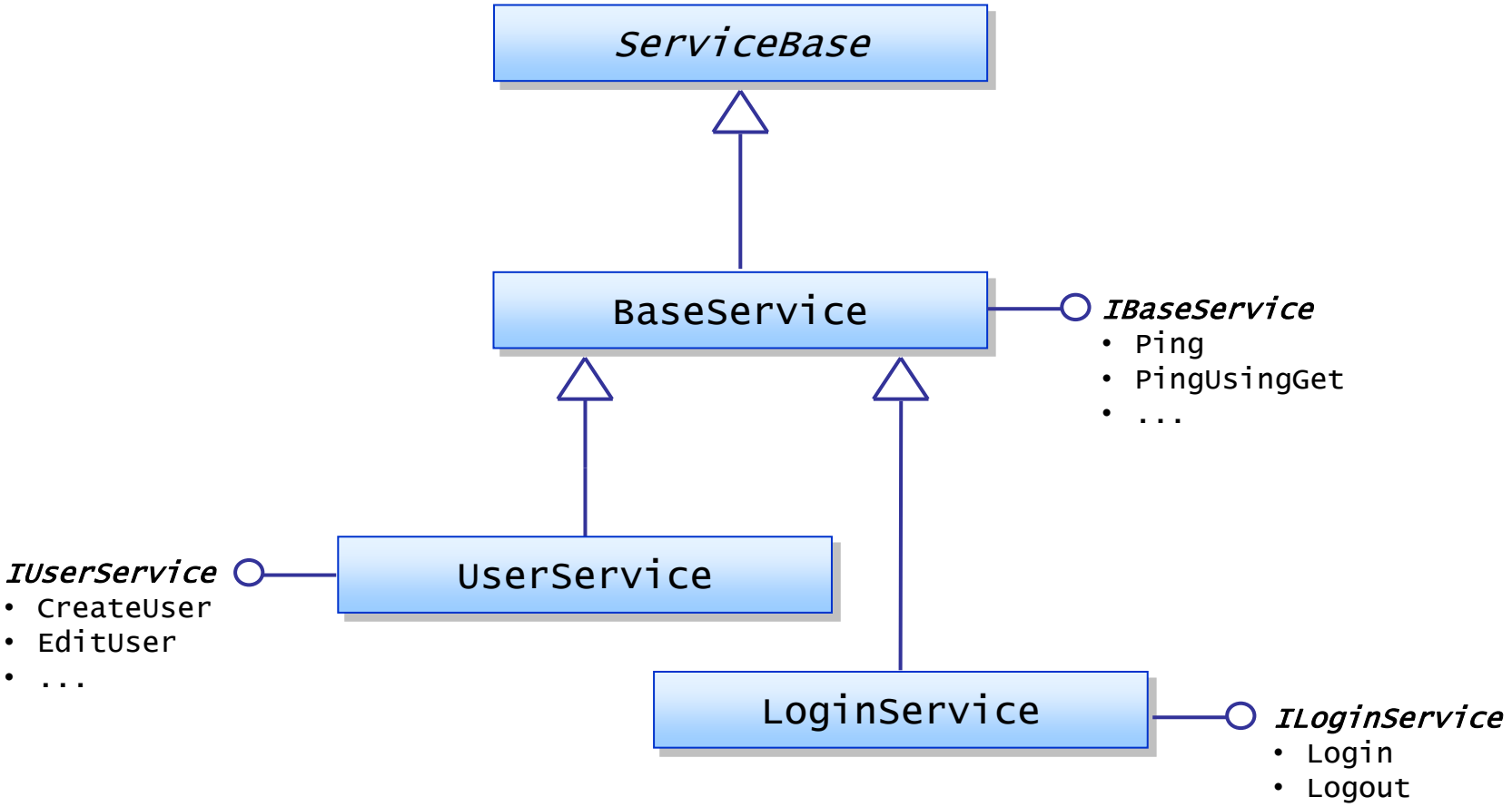



```
public class EditUserResponse : ResponseBase
{
}
```

- **All operations have an associated pair of specific request- and response-objects**
  - GetCountriesRequest, GetCountriesResponse
  - RemoveFriendRequest, RemoveFriendResponse

Implementing Services

- **Diagnostic Ping-operations are available to all services inheriting from BaseService**



Service Implementation

Captator

- Operations are typically simple DAL calls
- ServiceExecutor is defined in ServiceBase

```
public class SystemService : BaseService, ISystemService
{
  private Data.SystemDalBase _systemDal;

  public SystemService() {
    _systemDal = ...
  }

  GetCountryByIdResponse ISystemService.GetCountryById
                                        (GetCountryByIdRequest request)
  {
    return ServiceExecutor.Execute(request, () =>
    {
      Country country = _systemDal.GetCountryById(request.Id);

      return new SystemServiceEntities.GetCountryByIdResponse()
        { Country = country };
    });
  }
```

ServiceExecutor

- **The ServiceExecutor executes the service code**
  - With or without a system transaction
  - Authenticated or not

Carries call specific info such as login, language, tenant, call time etc.

```
public class ServiceExecutor
{
  public ServiceCallContextBase CallContext { get; private set; }

  public T ExecuteInTransaction<T>(AuthenticatedRequest request,
    System.Func<T> func) where T : ResponseBase, new()

  public T Execute<T>(AuthenticatedRequest request,
    System.Func<T> func) where T : ResponseBase, new()

  public T ExecuteInTransaction<T>(RequestBase request,
    System.Func<T> func) where T : ResponseBase, new()

  public T Execute<T>(RequestBase request,
    System.Func<T> func) where T : ResponseBase, new()
```

ServiceExecutor

Captator

- **Implements the general service code**

```
public class ServiceExecutor
{
  public T Execute<T>(AuthenticatedRequest request,
              System.Func<T> func) where T : ResponseBase, new()
  {
    // Validate request.AuthenticatedToken

    return Execute((RequestBase)request, func);
  }

  public T Execute<T>(RequestBase request,
              System.Func<T> func) where T : ResponseBase, new()
  {
    // Check validation attributes on the request object etc.

    T result = func();

    // Log the service call

    return result;
  }
```

Very small excerpt of the code

ServiceExecutor

- **The ServiceExecutor class centralizes all general aspects of executing a service operation**
  - Transactions
  - Multitenancy
  - Authentication
  - Service authorization based on user roles and/or tenant
  - Validation
    - Domain oriented validation
    - Validation that data in request and response objects is allowed for the authenticated user (belongs to its tenant)
  - ExceptionHandling
  - Logging

Multitenancy

Captator

Multitenancy refers to a principle in software architecture where a single instance of the software runs on a server, serving multiple client organizations (tenants). Multitenancy is contrasted with a multi-instance architecture where separate software instances (or hardware systems) are set up for different client organizations. With a multitenant architecture, a software application is designed to virtually partition its data and configuration so that each client organization works with a customized virtual application instance.

wikipedia

- **Tenants and AuthenticatedTokens are stored in a HostingMaster database common for all tenants**
- **Domain data and users are stored in domain databases that are specified in HostingMaster**
- **All tables with tenant specific data has a TenantId column**
  - All tenant specific queries must have a TenantId-predicate as part of the WHERE clause

Multitenancy

- **Tenancy database modes**
  - Shared database and shared schema
    - Tenant shares database and database schema with other tenants

dbo.MyTable

Id	TenantId	Name

  - Shared database and separate schema
    - Tenant shares database with other tenants but the database user is associated with a tenant specific schema

TenantXX.MyTable

Id	Name

  - Separate database
    - Tenant has a separate database

  - Separate server
    - Tenant has a separate database server

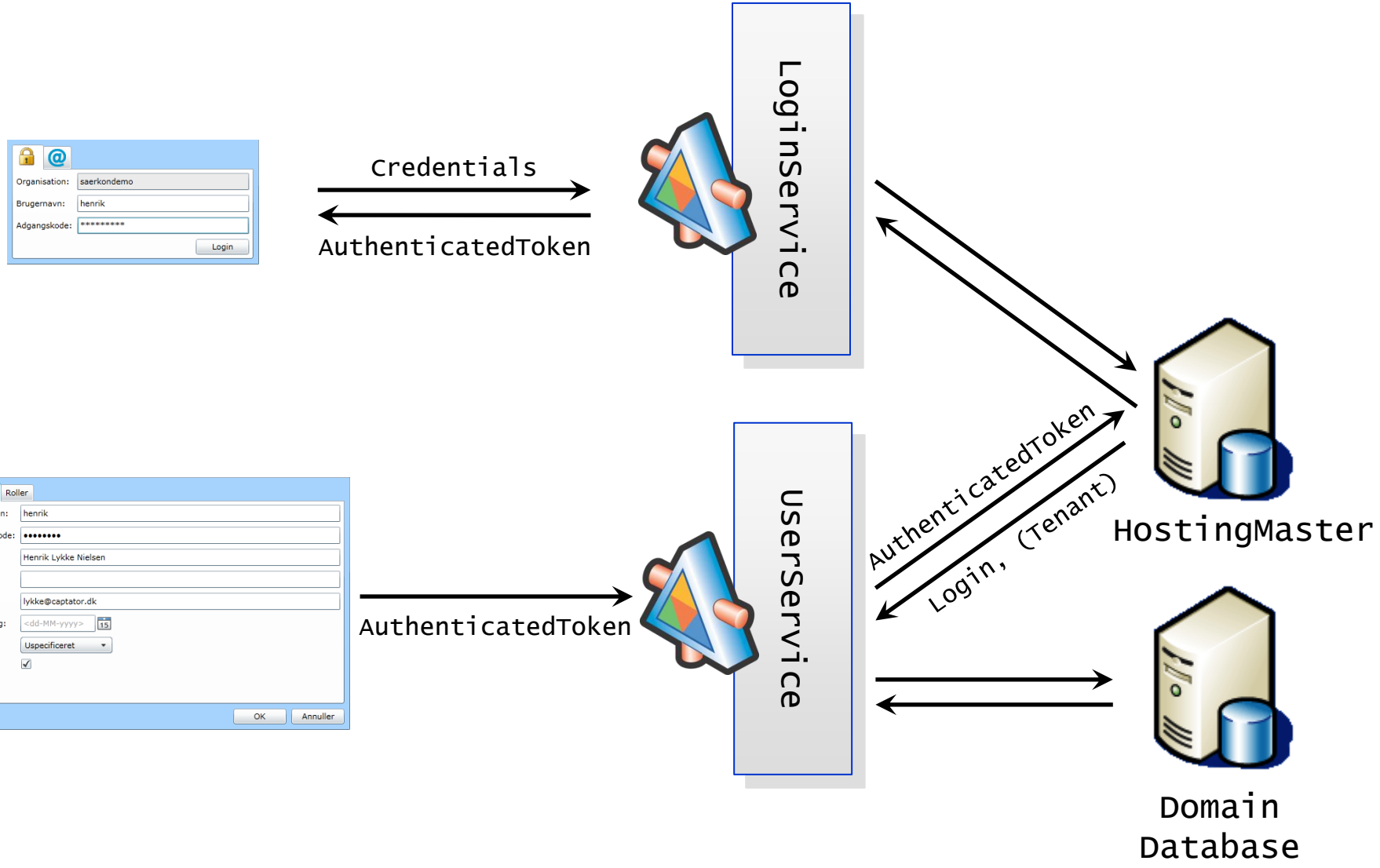
- Implementing the “Shared database and shared schema” mode enables all four modes

Authentication

Captator

- **Various login operations**
  - User name and password
  - Login on behalf of another user
  - Login Link – typically in email
  - Federated login / single sign-on
  - Optional IP lock
- **Successful authentication results in an AuthenticatedToken**
  - If the AuthenticatedToken is not recognized or has timed out an exception is thrown
- **The AuthenticatedToken must be passed in at each operation that takes an AuthenticatedRequest**

Authentication



Validation

Captator

- **Properties of request types are annotated with validation attributes**
  - System.ComponentModel.DataAnnotations.ValidationAttribute
- **Can automatically be included in documentation**
- **General purpose examples:**
  - AcceptedStrings, Maximum, Minimum, Range, RegEx, Required, StringLength, ValidEmail etc.

```
[ValidEmail] [UniqueEmail()]
public string Email { get; set; }

[RegEx(@"^\S{4,}$")]
public string ClearTextPassword { get; set; }

[StringLength(3)] [UniqueNickname()]
public string Nickname { get; set; }
```

Validation

Captator

- **ServiceExecutor validates the request object by validating all validation attributes**
- **Validation often require access to external data**
  - FriendshipExists, FriendshipNotExists, TableEntryExists, UniqueEmail, UniqueNickname
- **Attributes can implement an interface that**
  - signals that the validation is performed by executing a SQL query
  - can return the query for bundled execution (used for optimizing validation)

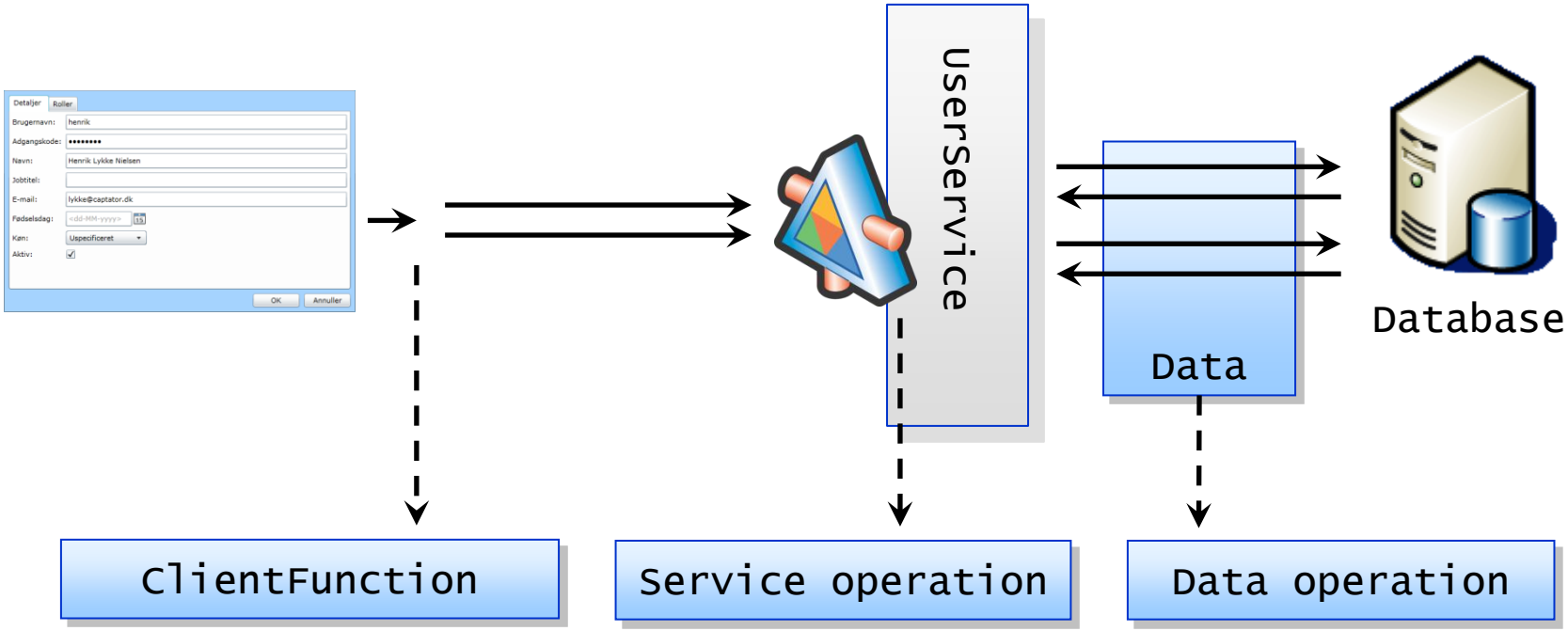
Logging

Captator

- **Purposes of logging**
  - Debugging, performance tuning, statistics, auditing
- **Various information is logged**
  - (Client) FunctionLog
  - ServiceLog
    - Request and response objects can optionally be logged
  - DataLog
    - Parameters / the actual SQL can optionally be logged
  - ActivityLog
  - ExceptionLog
- **Service call log entries are linked to make a call trackable**

Logging

- **Logging to a separate DataLog database**



- **A string dictionary is used for reducing log size**
- **Logging is asynchronous to enhance performance**

Options for Calling Services from .NET

Captator

- **1) Use standard network APIs**
  - Rather cumbersome
- **2) Use a WCF channel**

```
var uri = new Uri("http://mydemo.cloudapp.net/SystemService.svc");

var factory = new WebChannelFactory<ISystemService>(uri);
ISystemService systemService = factory.CreateChannel();

GetCountriesResponse response = systemService.GetCountries
  ( new GetCountriesRequest() { SystemKey = _systemKey });
```

- **3) Use standard .NET method calls**
  - Local execution, tests etc…

```
ISystemService systemService = new SystemService();

GetCountriesResponse response = systemService.GetCountries
  ( new GetCountriesRequest() { SystemKey = _systemKey });
```

Testing

Captator

- **Automatically repeatable tests**
  - Uses MS Test in Visual Studio
- **Testing of**
  - communication by calling the services using WCF
    - Only a few operations need to be tested with respect to WCF communication and generic service model implementation
  - service functionality by calling the services as regular .NET classes
    - All service operations should be tested

Testing

Captator

- **Code exclusively against the interface!**
  - The same code whether calling an XML/JSON over HTTP service, a SOAP service or a .NET component

```
var request = new GetCountryByIdRequest()
{
  Id = 1
}.AddLicenseKey();

GetCountryByIdResponse response =
  systemService.GetCountryById(request);

Assert.AreEqual("DK", response.Country.CountryCode);
```

  - Builder extension-methods such as AddLicenseKey, AddAuthenticatedToken, ...
  - CreateTestData utility-methods

Service Browser

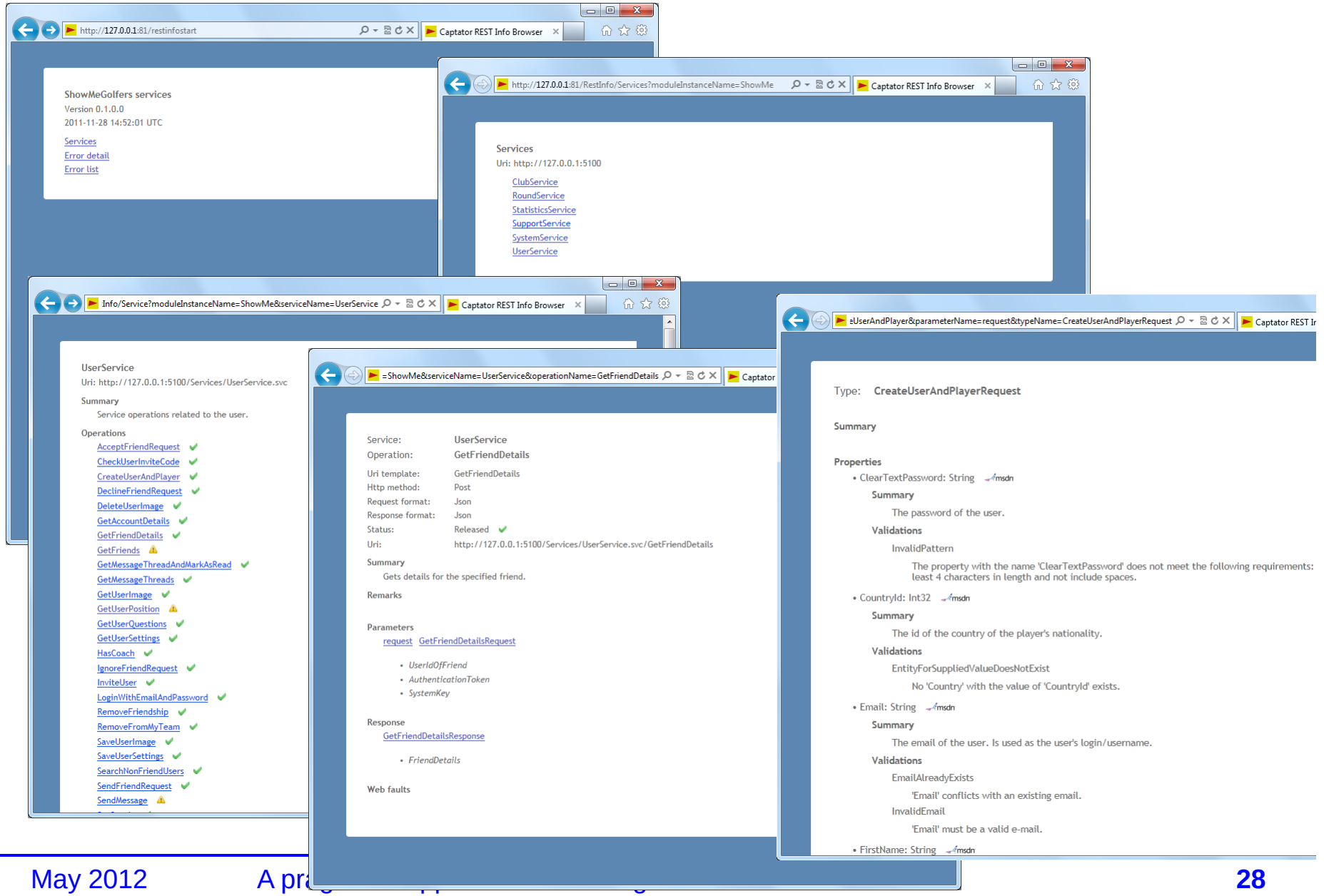
Captator

- Alternative for WCF Web HTTP Help Page
- ASP.NET MVC component used for showing metadata for XML/JSON over HTTP services
- Reflection for finding services, operations, datatypes and validation rules
- Leverages XML comments
- Custom DevelopmentInfo-attribute

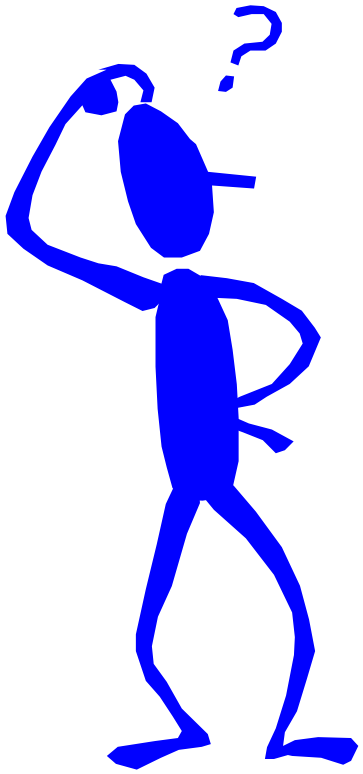
```
[WebInvoke(UriTemplate = "EditUser")]
[DevelopmentInfo(DevelopmentStatus.Released, TestStatus = TestStatus.Acceptable)]
EditUserResponse IUserService.EditUser(EditUserRequest request);
```

  - DevelopmentStatus: Undefined, Planned, InDevelopment, Released, Internal
  - TestStatus: Undefined, Planned, InDevelopment, Acceptable

Service Browser



Questions?



www.captator.dk

training, consulting, software development, ...