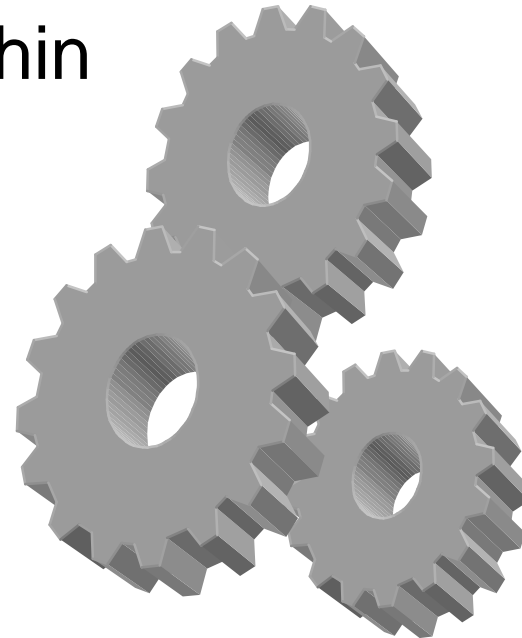


RDS einsetzen

Einleitung

Vom RPC, Remote Procedure
Call 1985, IPC zum RDS,
Remote Data Service 2003 hin
zur SOA, Service Oriented
Architecture 2006

Max Kleiner, Mai 2006



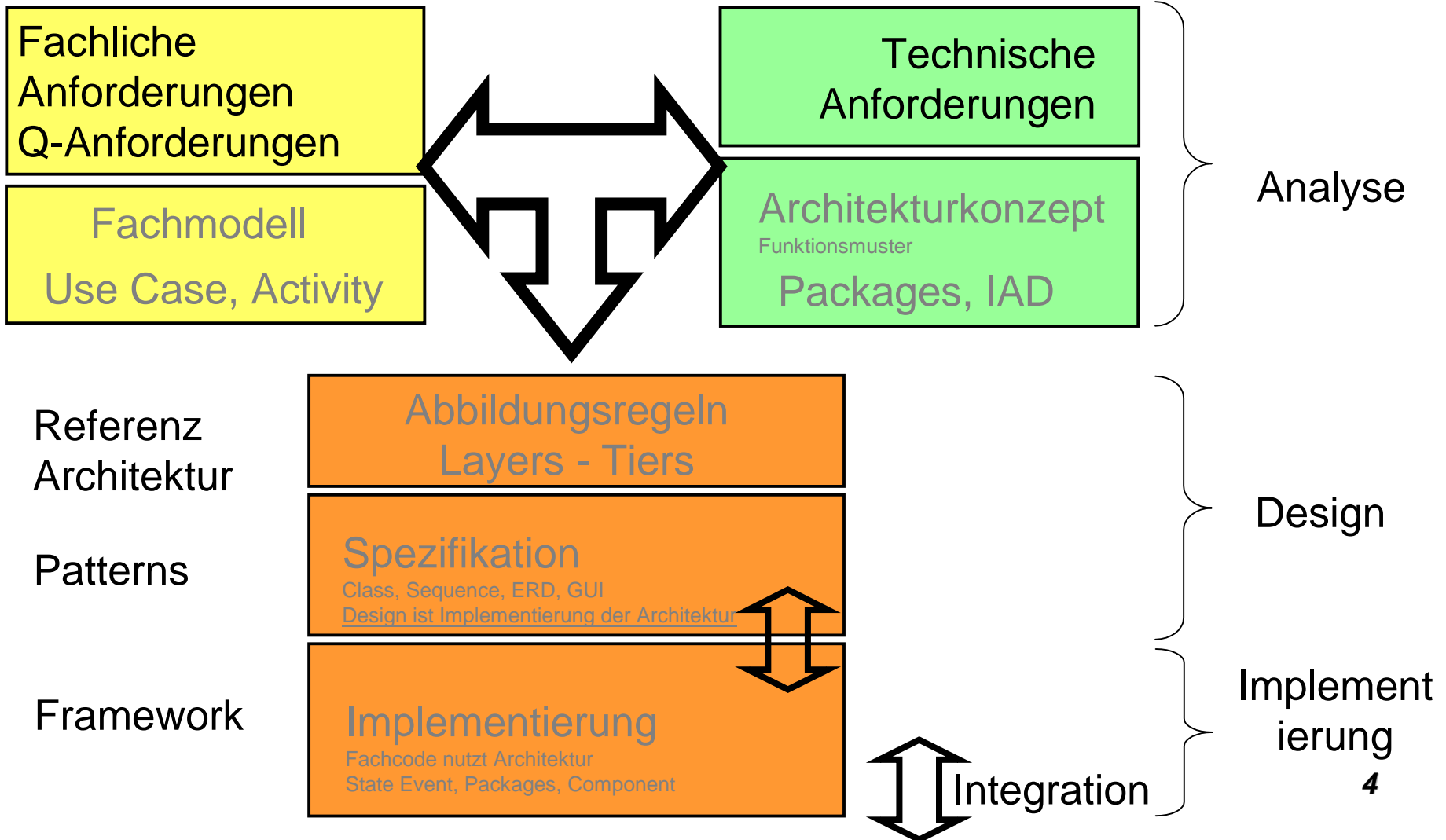
RDS Agenda

- Kernidee (Funktionsmuster) ist, gesammelte Daten aus n-Clients mit einer Applikation oder einem Dienst via SOAP zentral an einen Server zu senden und auszuwerten.
- Die Umsetzung zeigt mit WebServices ein Framework zur entfernten Speicherung von Daten (Storage as Service).
- Warum SOA nur eine Idee und keine Technik ist!

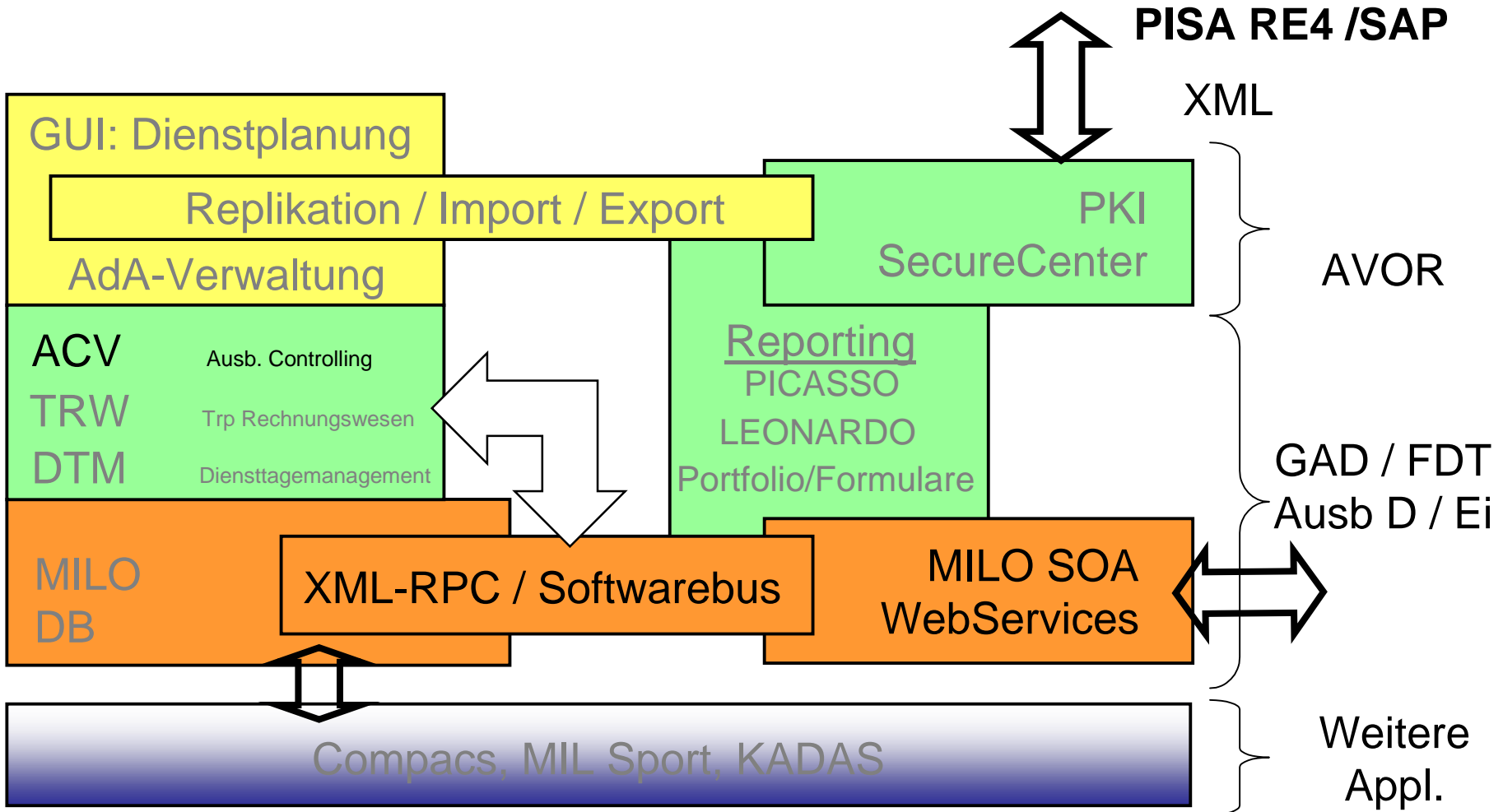
Anforderung

- Ein schweizweites Netz von mobilen Rechnern (Ausbildungscontrolling) sendet periodisch Ausbildungsdaten an einen zentralen Server, der die ausgewerteten Daten dann per Browser für die Instrukturen bereithält.
- In der folgenden Fallstudie möchte ich die Grundzüge dieser Technik näher bringen und vor allem das entfernte Speichern von Daten via Funktionen erklären.

Rahmenbedingung



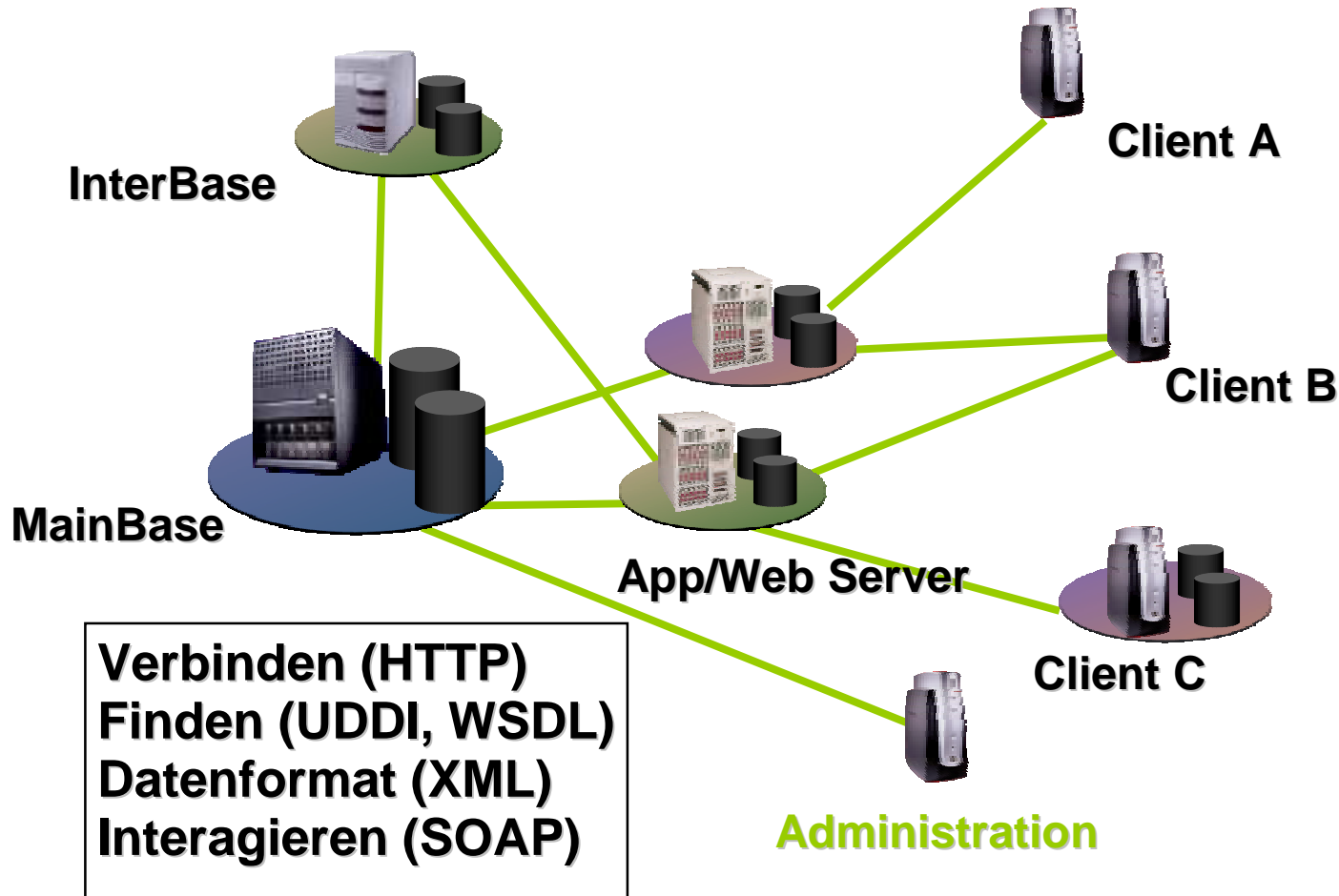
Architektur MILO 4.X



RDS: Worum geht es ?

- VCLScanner.exe als Client (Compilat)
- VCLScannerServer.exe als Webservice
- Apache WebServer
- InterBase DB / DBExpress Connector
- Die Komponenten des RDS:
 - WSDLPublish
 - Dispatcher
 - Invoker
 - DataSetProducer

RDS konkret



WebService Technik

- Dienste im Netz, die über Standard-Protokolle erreichbar sind
- Datenaustausch basiert auf XML
- Plattform- und unternehmensübergreifende Transaktionen kombinierbar
- OO-Technik mit Interfaces realisiert
- HTTP-Port 80 in der Regel offen

Schlüsseltechnologie

Web
Service
Benutzer

Finden: UDDI

<http://FindAService.ch>

XML mit Link zur Web Site

Business
Repos.

Realisieren: WSDL

<http://adb2.ch/scanner.exe/WSDL>

XML mit Servicebeschreibung

Web
Service

Ausführen: SOAP

<http://adb2.ch/serverscanner.exe>

XML als Prozeduraufruf

Webdienst-Katalog UDDI

- Universal Description, Discovery und Integration
- „Gelbe Seiten“ des Internet für Webdienste
- Gremium aus ca. 120 Firmen für die Propagation von Webdiensten
- Bsp.: <http://www.xmethods.net> and click the BabelFish Web Service. Also notice the path to the WSDL file for the service:
<http://www.xmethods.net/sd/2001/BabelFishService.wsdl>.

SOAP Webmodul

- Die Komponente THTTPSoapDispatcher empfängt und beantwortet hereinkommende SOAP-Nachrichten und entscheidet, welches Interface man verwendet.
- Das korrekte Objekt, welches das Interface realisiert, wird an den PascallInvoker weitergeleitet, der die richtig ausgepackte Methode mit den formatierten Argumenten einschliesslich Errorcodes in den eigentlichen OP-Aufruf inklusive Parametern transformiert.
- WSDL ist ein XML-Derivat zur Beschreibung der Schnittstellen von WebServices. Nachrichtenstrom-Formate und Funktionsaufrufe werden definiert.
- HTMLPublish ist verantwortlich für das Veröffentlichen von WSDL (WebServiceDescriptionLanguage)

Das Web Modul

type

```
TWebModule1 = class(TWebModule)
```

```
    WSDLHTMLPublish1: TWSDLHTMLPublish;
```

```
    HTTPSoapDispatcher1: THTTPSoapDispatcher;
```

```
    HTTPSoapPascalInvoker1: THTTPSoapPInvoker;
```

```
    DataSetTableProducer1: TDataSetTableProducer;
```

Das Interface

type

```
{ Invokable interfaces must derive from IInvokable }  
IVCLScanner = interface(IInvokable)  
['{8FFBAA56-B4C2-4A32-924D-B3D3DE2C4EFF}']  
    function PostData(const UserData : WideString; const  
        CheckSum : DWORD) : Boolean; stdcall;  
    procedure PostUser(const Email, FirstName,  
        LastName : WideString); stdcall;  
end;
```

Die Implementation des Interface

```
TVCLScanner = class (TInvokableClass,  
    IVCLScanner)  
public  
    function PostData(const UserData : WideString;  
        const CheckSum : DWORD): Boolean; stdcall;  
    procedure PostUser(const Email, FirstName,  
        LastName : WideString); stdcall;  
end;
```

Implementation ...

Client Aufruf

var

WS: IVCLScanner;

- WS:= HTTPRIO1 as IVCLScanner;
- WS.PostData(reFinalResults.Text,CRC);

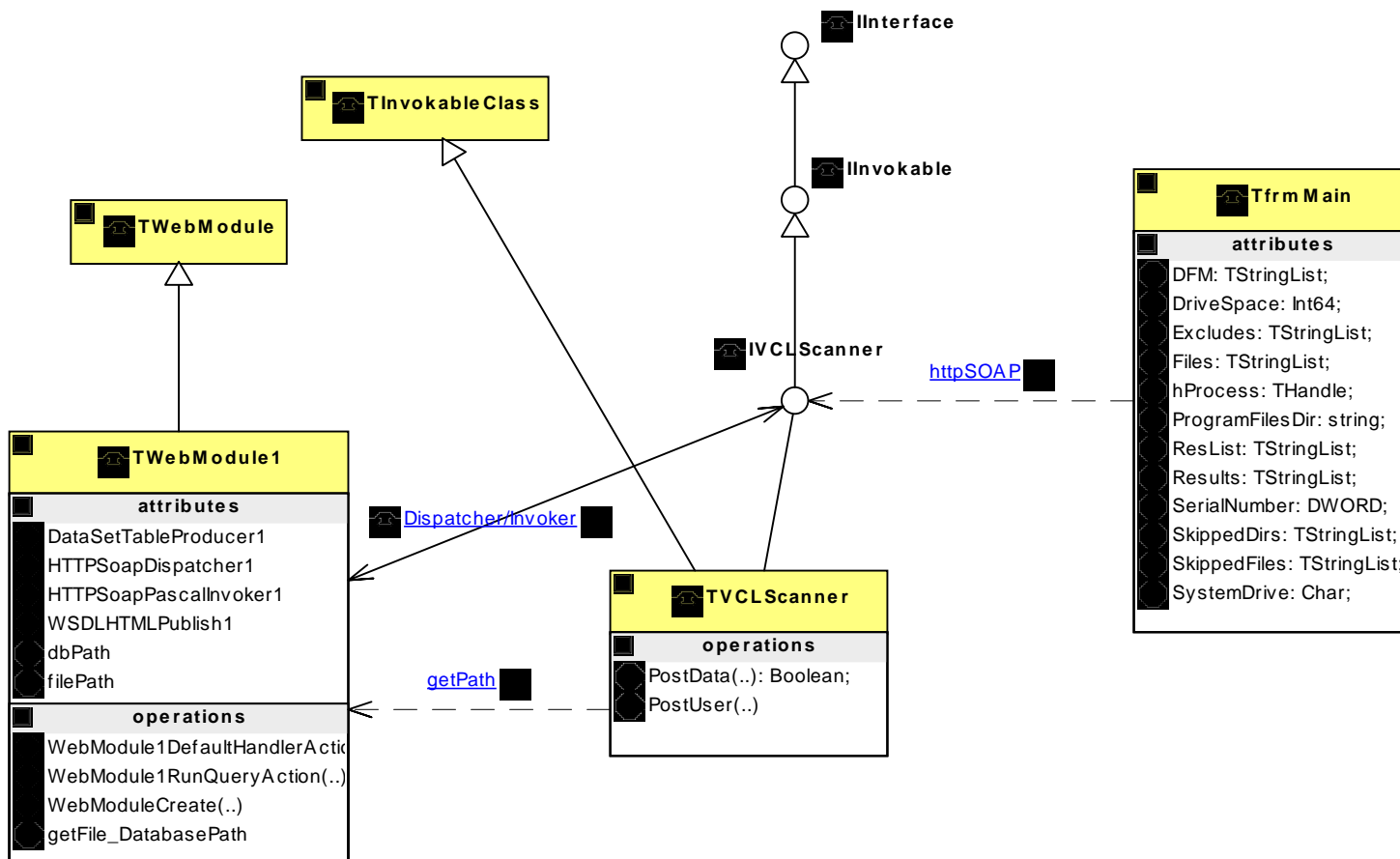
HTTPRIO1:URL

<http://adb2/scripts/vclscannerserver.exe/soap/IVCLScanner>

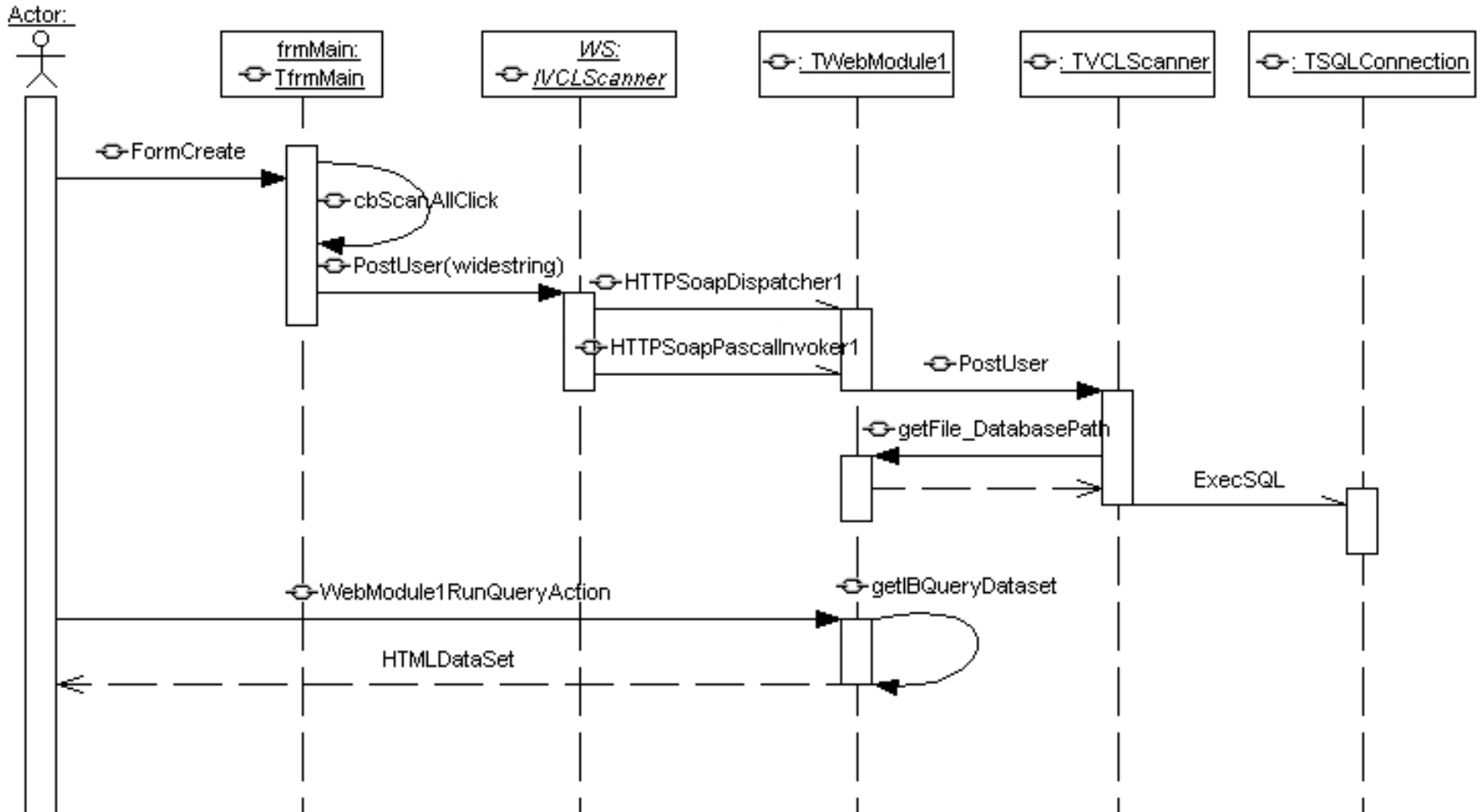
Die sieben Express Schritte

1. `Connection:= TSQLConnection.Create(NIL);`
2. `with Connection do begin ...`
3. `DataSet := TSQLDataSet.Create(NIL);`
4. `with DataSet do begin ...`
5. `SQLConnection:= Connection;`
6. `CommandText:=`
7. `Format('insert into ACL_LN values("%d","%s", "%s", "%s", "%s")',[10, Email, FName, LName, ACLdate]);`

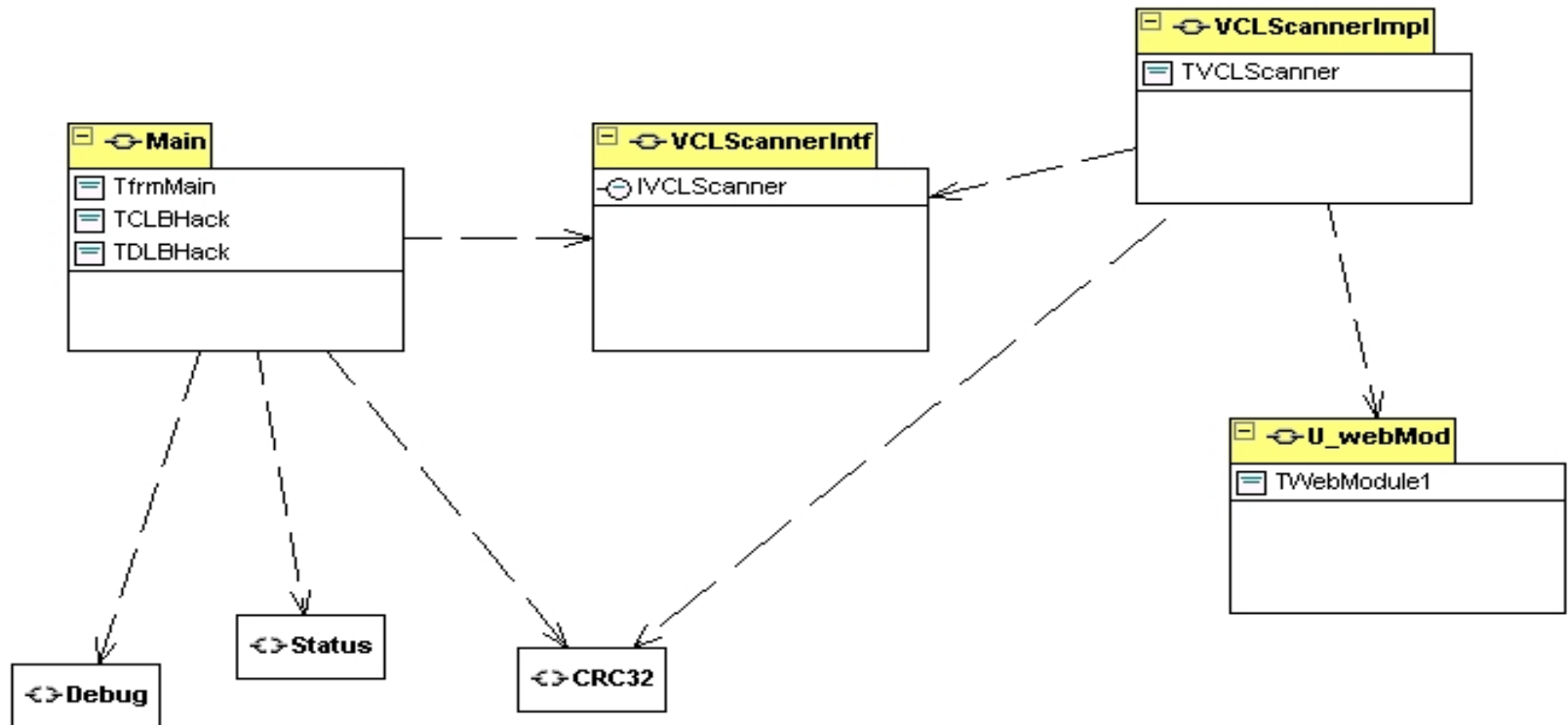
Struktur als Class Diagram



Das Sequenzdiagramm



RDS Pakete



RDS Vorteil gegenüber Scripts

- OO-Technik mit Klassenbildung
- Direkt aus dem Code instanzierbar
 - Auch ohne Browser möglich
 - Microsoft selbst betont den Willen zur Einbindung anderer Hersteller und Plattformen
 - Echte typensichere Compile erlauben eine stabile Nutzung mehrerer mobiler Plattformen und Sprachen
 - Dank des Verzeichnisdienstes immer up-to-date

Warum es vorderhand SOA wenig braucht ?

- Nicht jeder Service oder Prozess muss verteilt, orchestriert und veröffentlicht werden.
- Zu viele nachrichtenorientierte Services erhöhen den Verwaltungsaufwand und vermindern Performance und Stabilität.
- SOA ist wenig typensicher, will heißen, der Hund ist zur Laufzeit begraben!

F1: Fragen und hoffentlich Antworten ?

<http://max.kleiner.com>

