

## S18 – Workshop - SOA

Wilhelm Mild, IBM Labor

Heinz Peter Maassen, Lattwein

# Agenda

- SOAP – SOA – ESB
- Was ist ein Webservice
- Beispiel
- Workshop

# Was ist SOA ?

SOA sollte folgende Regeln erfüllen:

- Ein Dienst ist in sich abgeschlossen und kann eigenständig genutzt werden.
- Dienste sind in einem Netzwerk verfügbar.
- Jeder Dienst hat eine veröffentlichte Schnittstelle. Für die Nutzung reicht es, die Schnittstelle zu kennen. Kenntnisse über die Details der Implementierung sind hingegen nicht erforderlich.
- Die Benutzung von Diensten ist **Plattform unabhängig**, d.h. Anbieter und Nutzer eines Dienstes können in unterschiedlichen **Programmiersprachen** auf verschiedenen Plattformen realisiert sein.
- Alle Dienste sind in einem Verzeichnis registriert.
- Die Dienste werden **dynamisch gebunden**, d.h. bei der Erstellung einer Anwendung, die einen Dienst nutzt, muss der Dienst nicht vorhanden sein. Er wird erst bei der Ausführung lokalisiert und eingebunden.

\* Aus Wikipedia.de

## SOAP ist ...nicht ->




- Der Begriff Soap (engl., für Seife) hat mehrere Bedeutungen:
- die Seifenoper (vom englischen: „Soap-Opera“)
- eine US-amerikanische Comedy-Serie, die in Deutschland unter dem Titel Soap – Trautes Heim im Fernsehen lief.
- Snakes on a Plane, ein Film
- Subjective, Objective, Assessment, Plan, ein Dokumentationsschema in der Medizin
- das Grafikprogramm Soap der Firma MetaCreations (Kai Krause)

\* Aus Wikipedia.de

## SOAP ist ...

Die Abkürzung **SOAP** bzw. **Soap** steht für:

- Simple Object Access Protocol (plattformunabhängiges Kommunikationsprotokoll bei Web Services)



**Serviceorientierte Architektur (SOA)**, engl. **service oriented architecture**, auch **dienstorientierte Architektur**, ist ein Ansatz der **Informationstechnik**, Dienste von Mitarbeitern und Organisationen zu strukturieren und zu nutzen.

\* Aus Wikipedia.de

## Was ist SOAP ?

- Steht für: Simple Object Access Protokoll.
- Ist ein Protokollstandard des W3C.
- Macht Anwendungen webfähig und ermöglicht die Kommunikation von verteilten Anwendungen und Objekten.
- SOAP ist ein RPC (Remote Procedure Call), der Daten in XML darstellt und meist HTTP als Übertragungsprotokoll verwendet.

## Was ist SOAP ? (2)

- SOAP ist unabhängig von Betriebssystem, Programmiersprache und Objektmodell und kann leicht in verschiedene Plattformen implementiert werden.
- SOAP arbeitet sowohl mit Objekt orientierten als auch mit nicht-objektorientierten Programmiersprachen. Die von SOAP aufgerufenen Funktionen sind eher statische Methoden.
- Vorteile von SOAP sind Standardisierung, Plattformunabhängigkeit, sowie die robuste und skalierbare Implementierung.

## Was ist SOAP ? (3)

- SOAP ist ein XML-basiertes Protokoll, das aus drei Teilen besteht:
  - der Spezifikation für einen Umschlag (Envelope), der ein Regelwerk definiert, welches beschreibt, was in einer Nachricht enthalten ist, von wem es wie verarbeitet werden soll, und ob einzelne Daten optional sind oder enthalten sein müssen
  - ein Satz von Kodier- und Ordnungsregeln (Serialization), welcher Instanzen von anwendungsspezifischen Datentypen beschreibt
  - eine Konvention, um Remote Procedure Calls und eventuelle Antworten auf diese zu repräsentieren.



## Ziele von SOAP

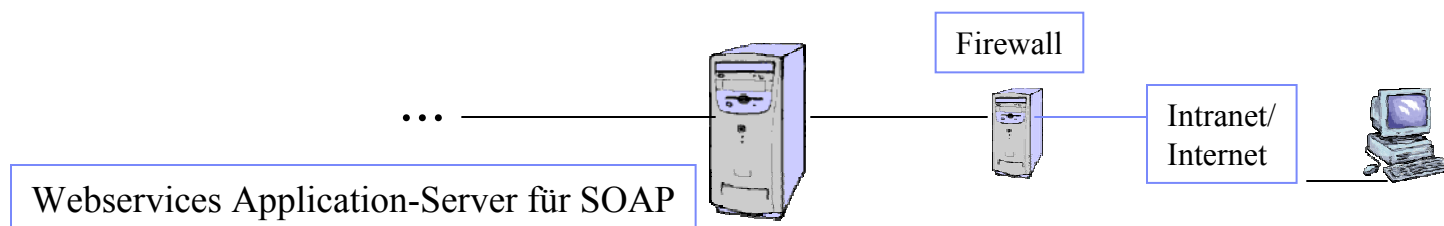
- **E**rweiterbarkeit
- **E**infachheit
- **E**insatz auf verteilten Systemen, auch durch Firewalls hindurch
- Das Rad nicht neu zu erfinden, sondern aktuelle Standards (HTTP und XML) zu nutzen

## Was ist ein Web Service ?

- Web Services sind aufrufbare Dienste im Internet.
- In Web Services sind Kommunikation und Methode standardisiert.
- Web Services sind registriert.
- Sie werden verwendet um einen Datenaustausch zwischen Programmen auf verschiedensten Rechnern auszuführen.
- Verwenden intern meist XML Format und HTTP Protokoll.

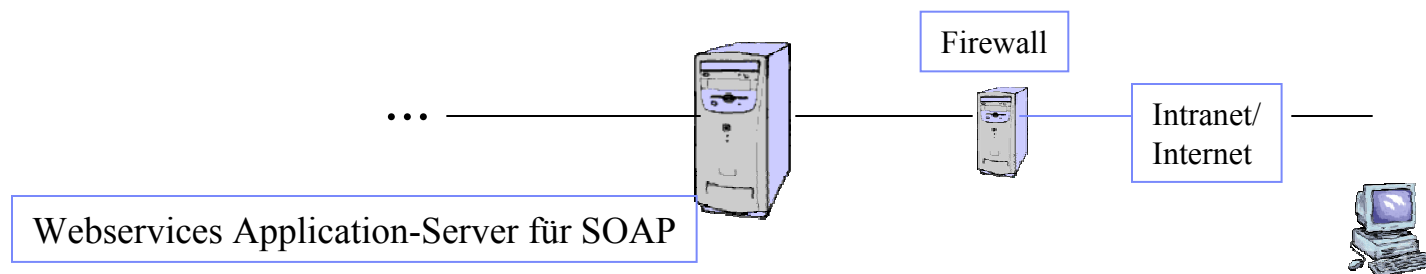
# SOAP: Webservices

- **Web Services** sind automatisierbare Dienste per Internet (oder Intranet)
- Dabei kommunizieren Applikationen, es werden also **nicht** HTML-Seiten zu einem Webbrowser geschickt, die von einem Menschen betrachtet werden, sondern Programme tauschen Daten und starten auf entfernten Rechnern Funktionen.
- Die weltgrößten Softwarehersteller haben sich auf Prozeduren und Protokolle für Web Services geeinigt:
- **SOAP** (Simple Object Access Protocol)
- **WSDL** (Web Services Description Language)
- **UDDI** (Universal Description, Discovery, and Integration)



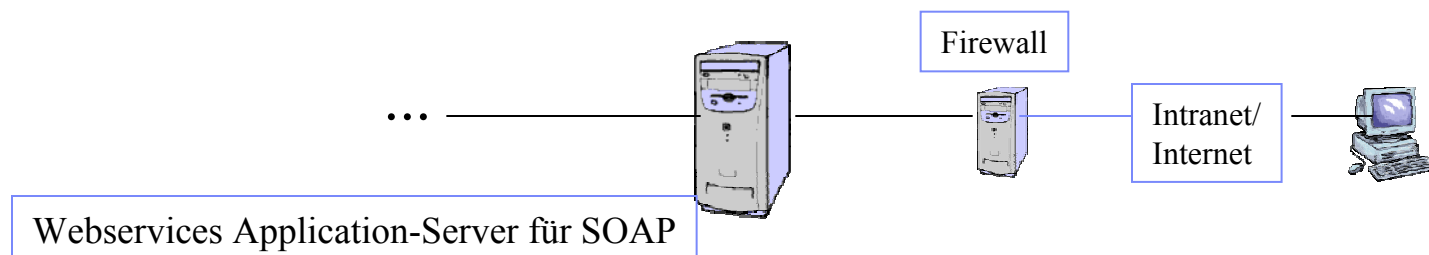
## SOAP: Webservices (2)

- Dadurch wird standardisiert, wie auf entfernten Rechnern Funktionen aufgerufen werden und wie automatisierbarer Datenaustausch erfolgt
- Das Besondere an SOAP Web Services ist die allgemeine Akzeptanz, Kommunikation ist sogar zwischen den beiden konträren Plattformen Sun / Java und Microsoft / .NET möglich
- Anwendungen mit SOAP-Schnittstelle lassen sich viel leichter kombinieren, als dies früher möglich war
- Kosten für Integrations-Middleware und EAI (Enterprise Application Integration) werden gesenkt
- Viele Web Services per SOAP werden bereits heute im Internet angeboten



## SOAP: Webservices (3)

- Viele große Anwendungen, wie zum Beispiel mySAP®, aber auch viele Office-Programme, wie zum Beispiel Microsoft Project 2002 unterstützen Web Services per SOAP.
- Per Web Services als Dienste angebotene Geschäftsprozesse sind Voraussetzung für moderne IT-Konzepte, die zur Realisierung des agilen RTE (Real-time Enterprise) eine flexible SOA (Service Oriented Architecture) benötigen.

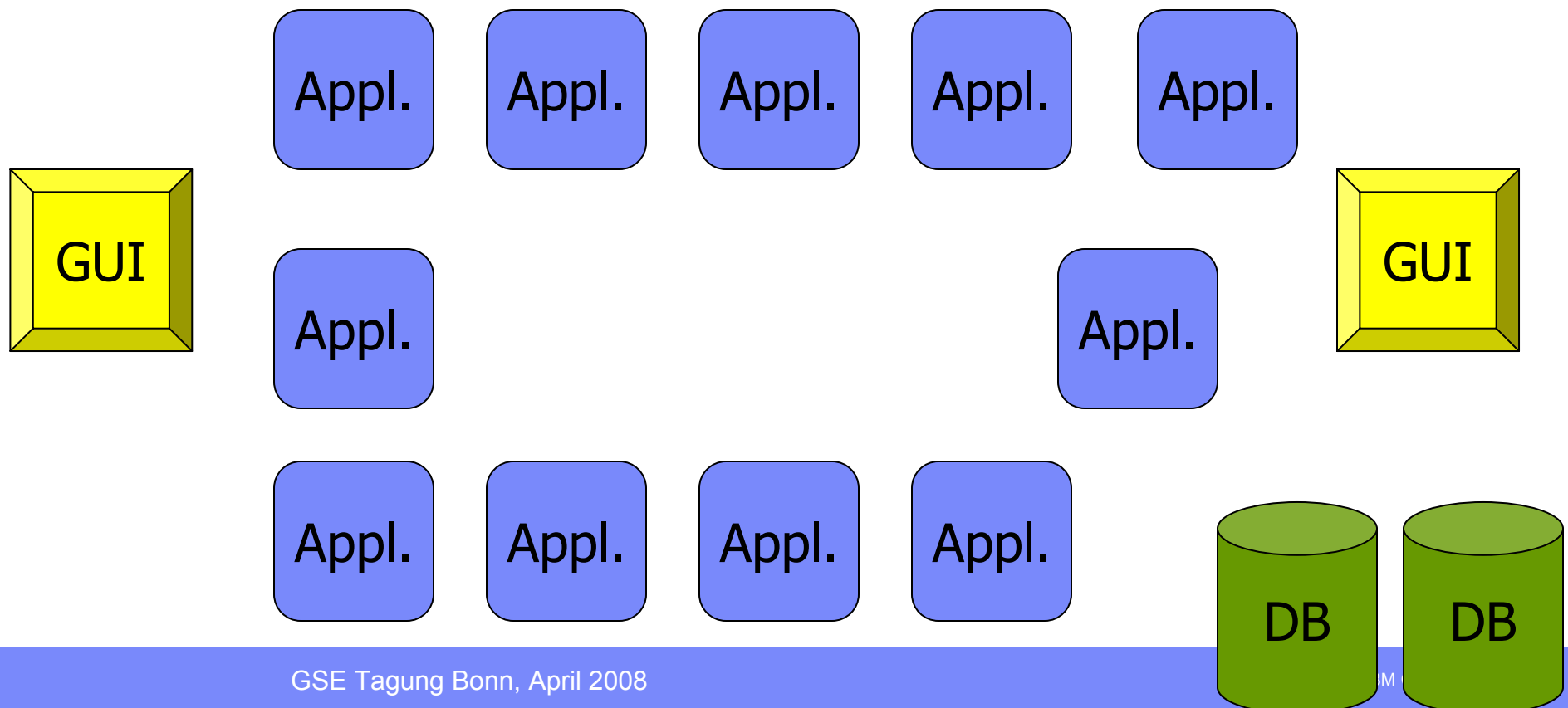


# WebService: Beispiele

- Beispiele für Webservices:
  - **eBay Price Watcher** Checks current bid price of an eBay auction.
  - **Currency Exchange Rate** Exchange rate between any two currencies
  - **Domain Name Checker** Checks whether a domain name is available
  - **BabelFish** Interface for AltaVista's Babelfish service.
  - **FedEx Tracker** Access to FedEx Tracking information
  - **SOAP Web Search SOAP** Interface to major search engines
  - **Text-To-Speech (TTS)** A Text-To-Speech (TTS) web service.
  - **Airline Fare Checker** Returns airfare/flight information.
  - **German Bank Code Lookup (BLZ)** Lookups German bank codes for name, city and zipcode
  - **Stock Quote** Stock quote service which actually provides more than just quote.
  - Und viele mehr ... siehe Internet.

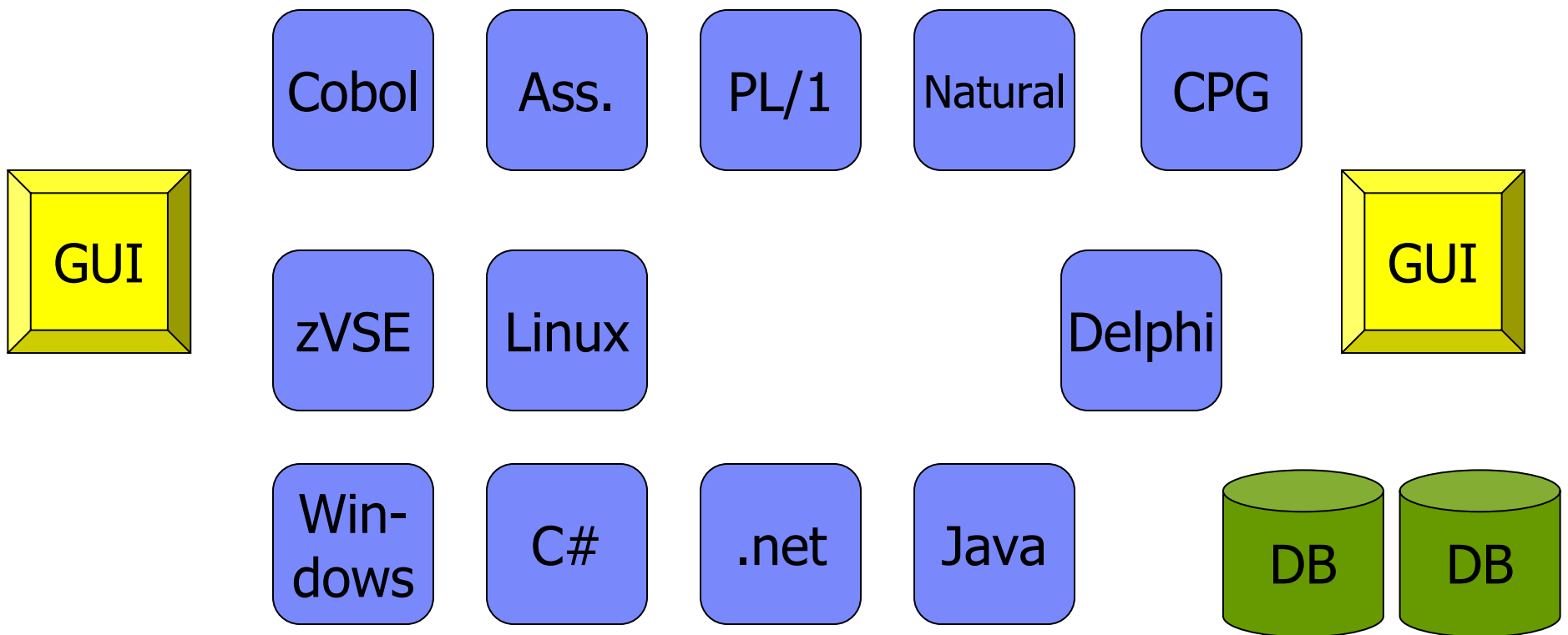
## Warum soll man SOA machen?

- Systeme sind über Jahre gewachsen



## Warum soll man SOA machen?

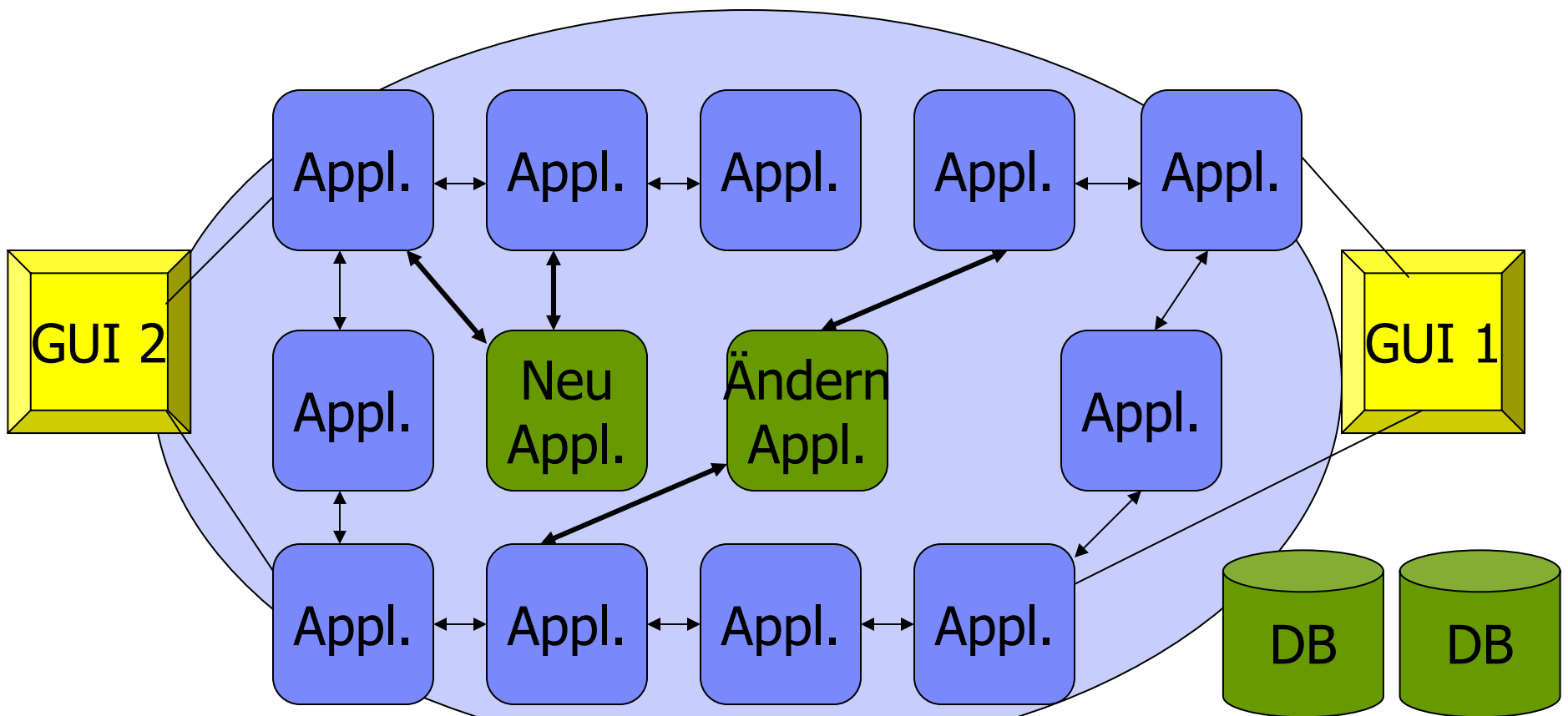
- Verschiedene Plattformen und Programmiersprachen





## Warum soll man SOA machen?

- Neue Anwendungen kommen hinzu und verändern bestehende Applikationen

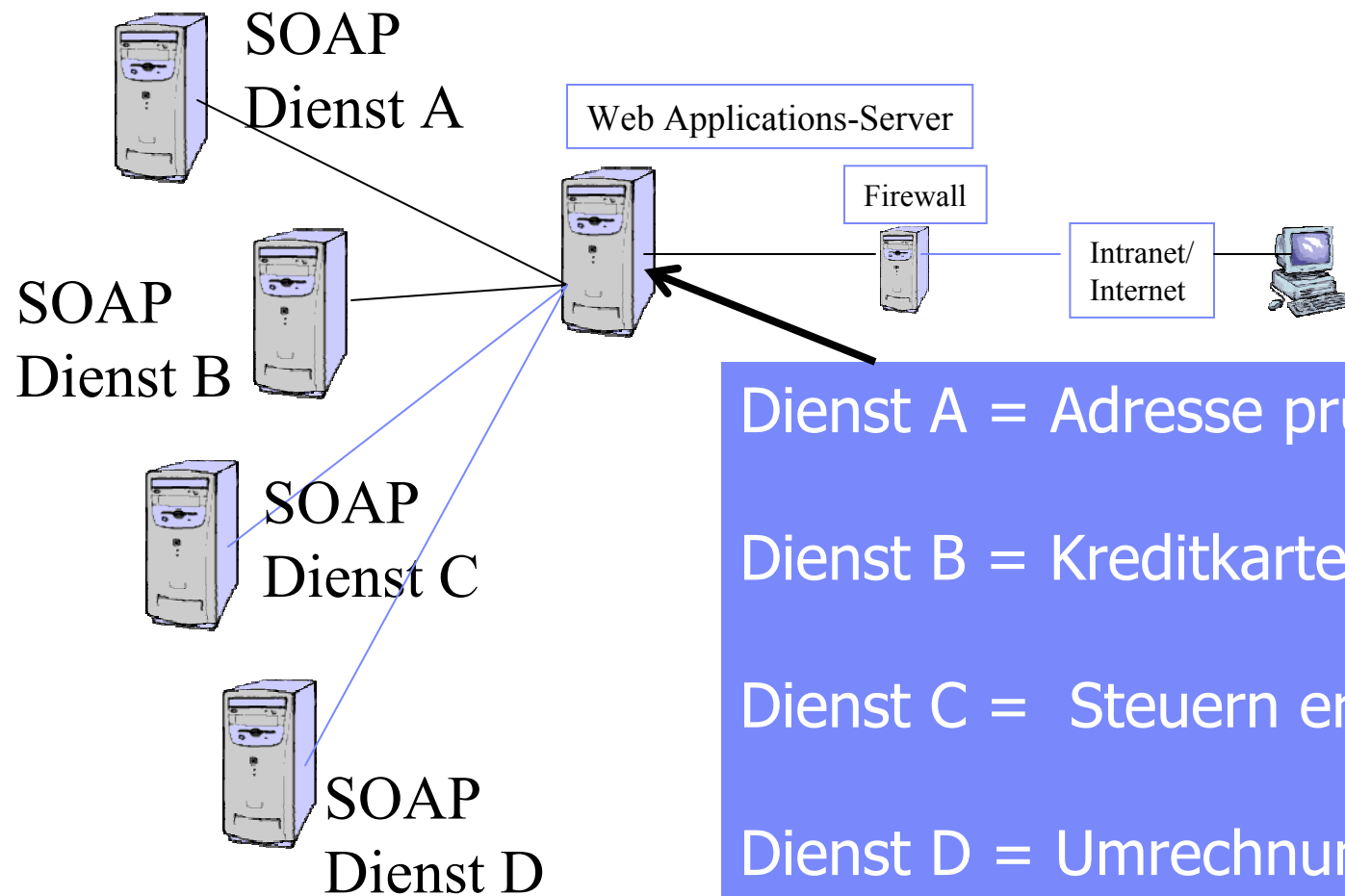


## SOAP Beispiel: Onlineshop

**Ein Kunde möchte in einem Onlineshop Artikel bestellen. Der Onlineshop ist in einen Applikationsserver realisiert.**

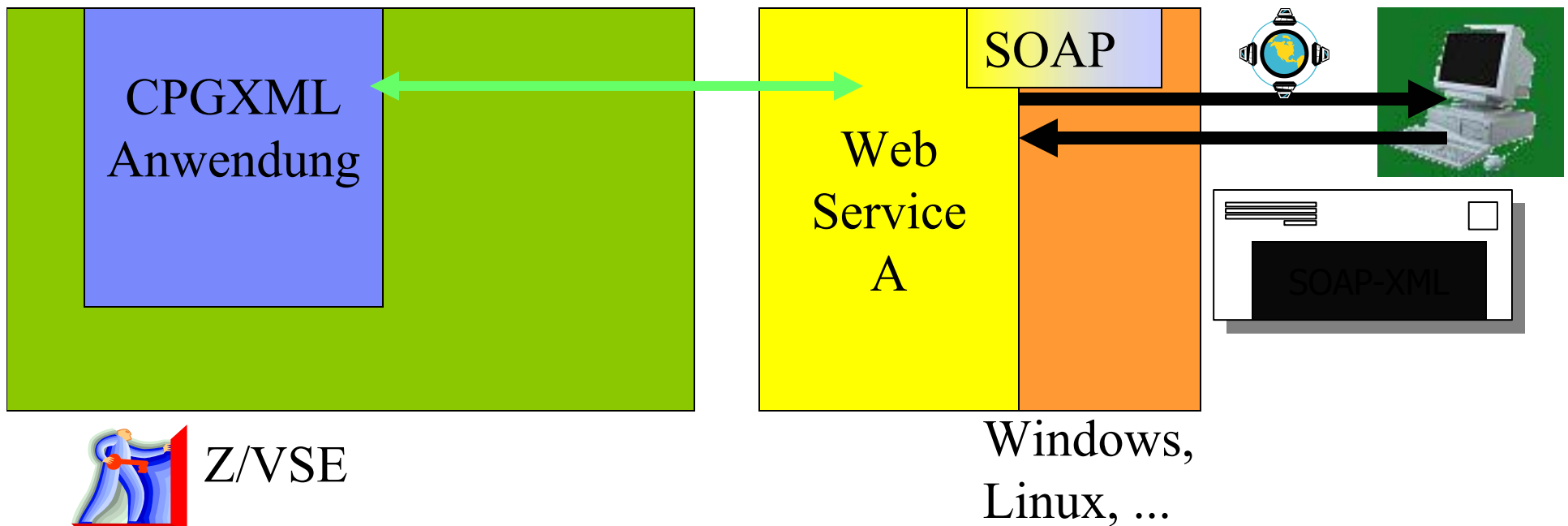
- ❖ Dieser Applikationsserver überprüft über den SOAP Web Service des **Servers A**, ob die angegebene Adresse gültig ist,
- ❖ verifiziert über den SOAP-**Dienst B** die Kreditkartennummer,
- ❖ ermittelt über den SOAP-**Dienst C** die für das jeweilige Land zu berechnenden Steuern (Umsatzsteuer, Luxussteuer, ...),
- ❖ erfragt beim SOAP-**Dienst D** tagesaktuelle Währungsumrechnungskurse, um den Endpreis korrekt berechnen zu können und
- ❖ zeigt dem Benutzer alle Ergebnisse gesammelt im Webbrowser an.

## SOAP Beispiel: Onlineshop

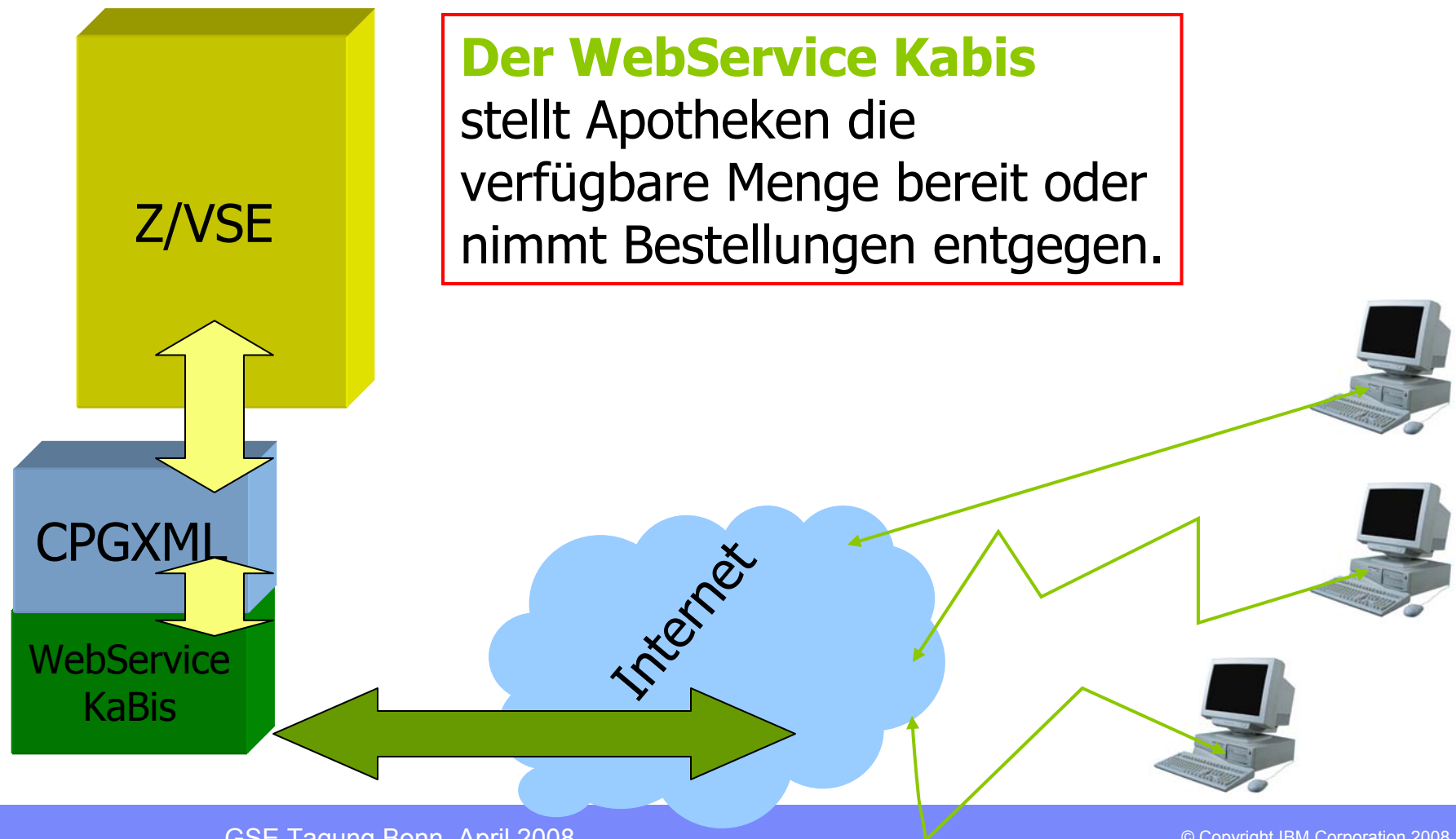


# SOAP und CPG5

SOAP Anwendungen können heute schon einfach mit CPGXML VSE Anwendungen implementieren. Der Host wird als Server Komponente in die SOAP Anwendung eingebunden.



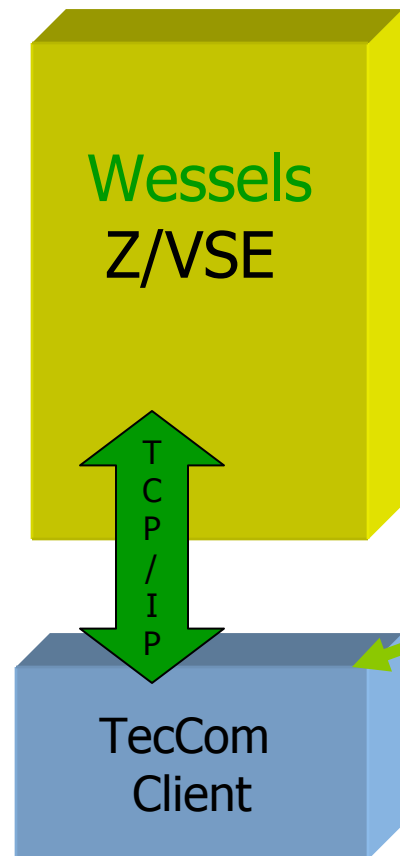
## Z/VSE als Webservice Provider



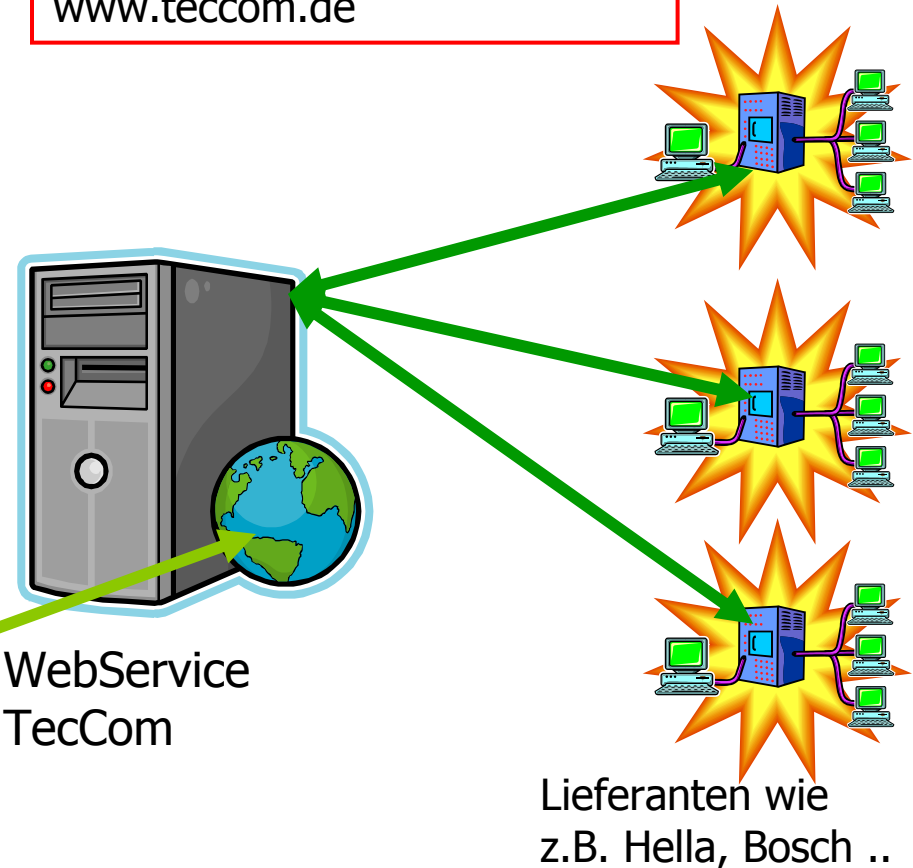
## z/VSE als Webservice Client

Aus mehreren Anwendungen können Bestellungen über TecCom erfolgen.

z/VSE spricht über TCP/IP den TecCom Client an, der über Webservices bei den verschiedenen Lieferanten Bestellungen auslöst.



**TecCom** stellt einen zentralen Bestellservice für die Automobil Großhändler zur Verfügung.  
[www.teccom.de](http://www.teccom.de)



# Wie gehe ich ein SOA Projekt an

## 1. Zukünftige Geschäftsziele definieren

*Geschäftsziele auflisten*

*Geschäftsziele priorisieren*

## 2. Die IT Prozesse analysieren

*IT Prozesse erfassen*

*Programm Struktur pro Prozess aufschlüsseln*

## 3. Zukünftige Dienste definieren

*Geschäfts Ziele und IT Prozesse zusammen bringen*

*Geschäftsziele über IT Dienste definieren*

## 4. Dienste Realisieren und implementieren

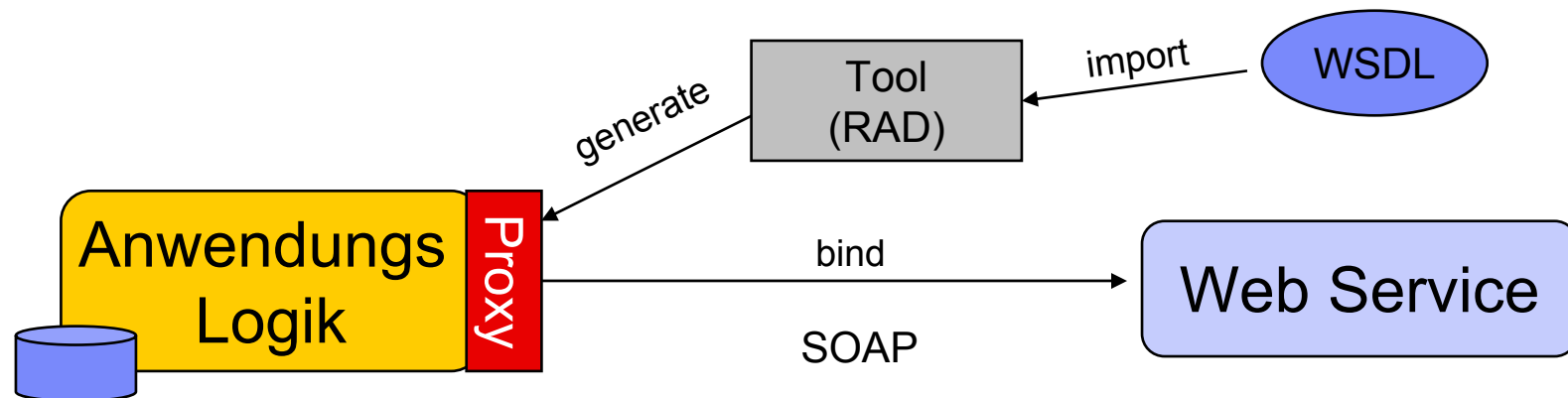
*Service Orientierte Architektur definieren*

*System z Ankopplung*

*IT Infrastruktur Anpassungen definieren*

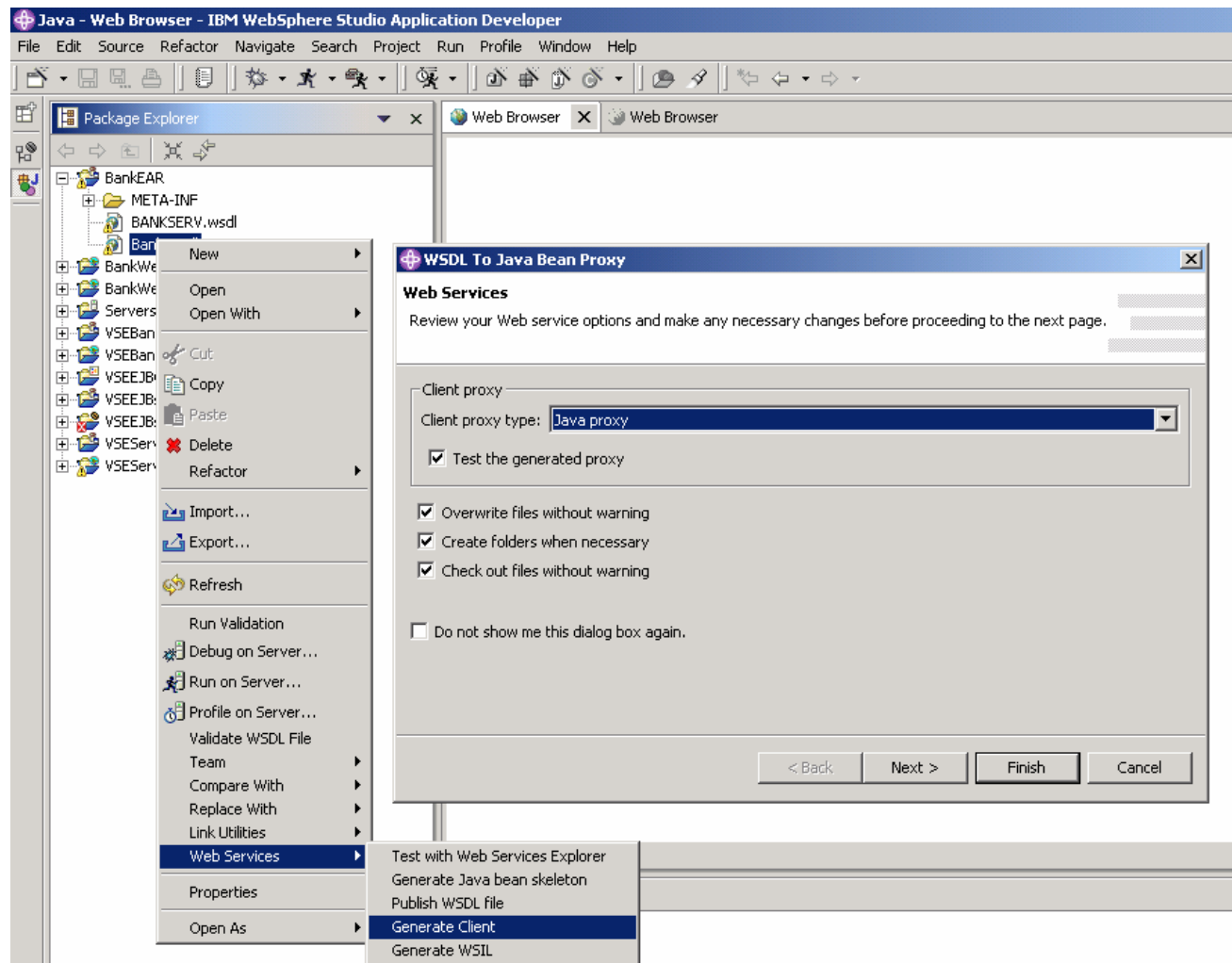
## Existierende Web Services verwenden

- **Aufrufen** eines bestehenden Web Service
  - Die Web Service Description (WSDL) für den gewünschten Web Service finden/importieren
  - In einem Tool wie **Rational Application Developer** (RAD,WDz) oder Microsoft Visual Studio wird die WSDL importiert
    - Generieren des “Proxy-Codes”
    - Anwendungen rufen die Logik des Proxy-Codes auf also ob es eine lokale Funktion wäre



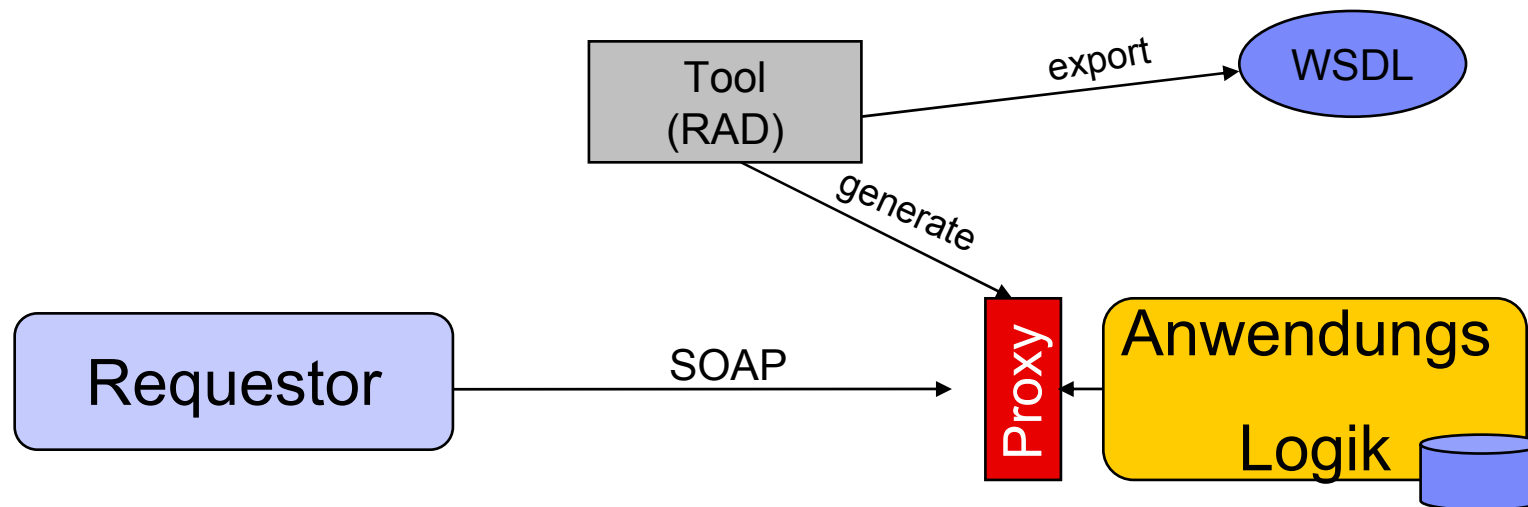


## Web Services (Proxy) mit Java oder MS .Net generieren

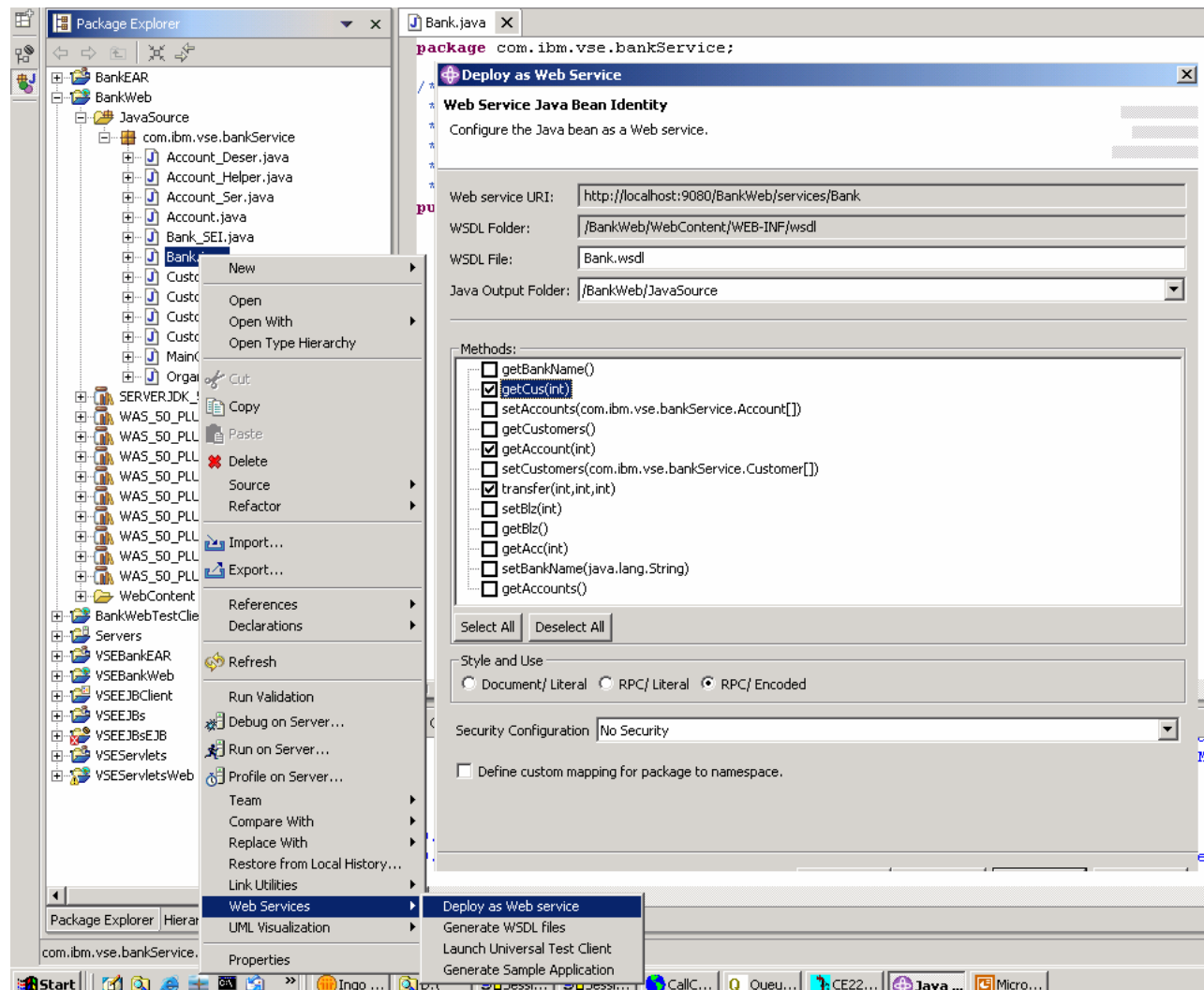


## Erstellen eines Web Services von existierender Logik

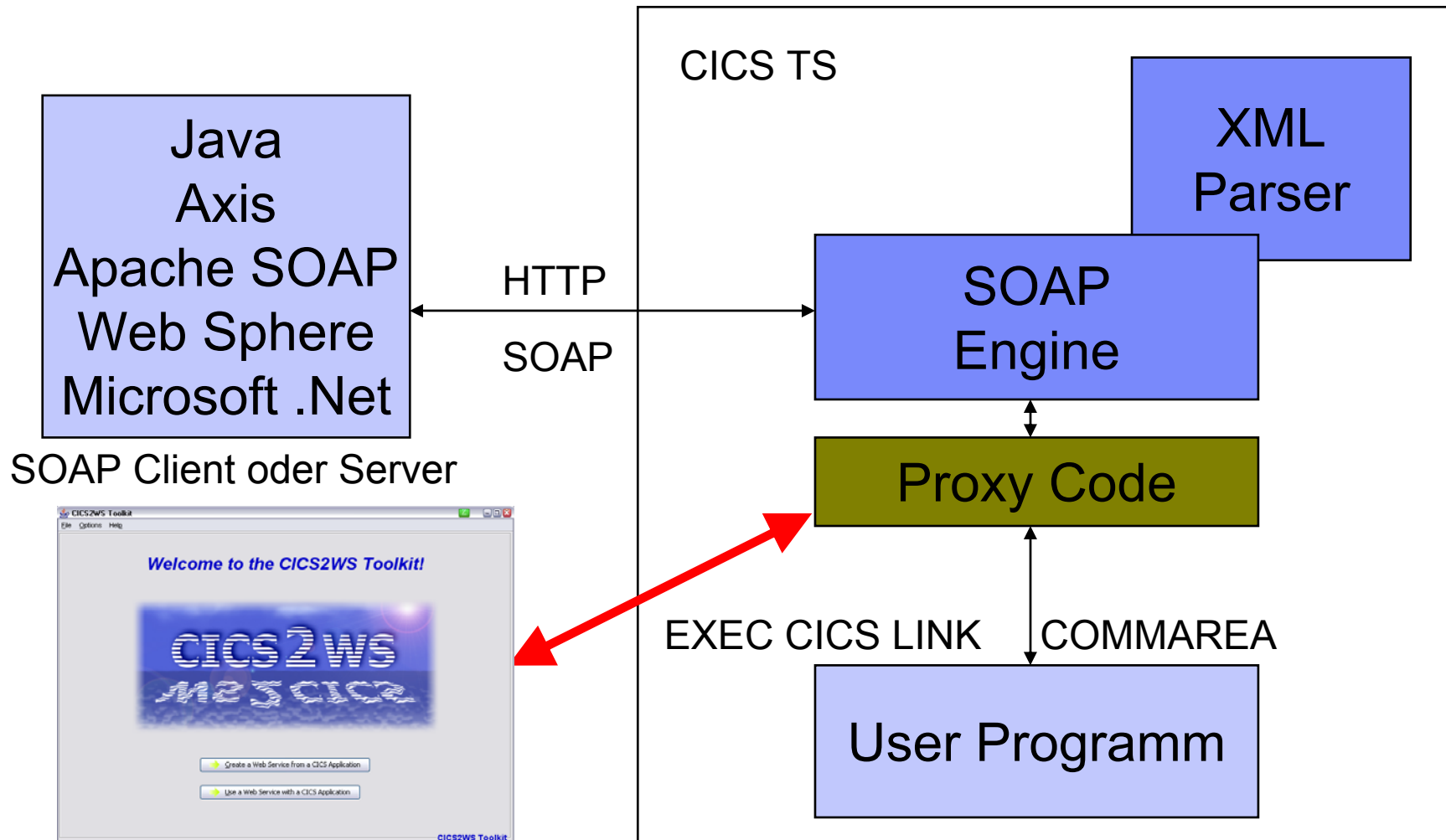
- **Erstellen** eines neuen Web Services
  - Eine Logik/Funktion, die aufrufbar ist, soll als Web Service zur Verfügung stehen
  - In einem Tool wie **Rational Application Developer** (RAD/WSAD) oder Microsoft Visual Studio wird der Web Service modelliert
    - Erzeugen der Web Service Description (WSDL) und evtl veröffentlichen in einem UDDI
    - Generieren des “Proxy-Codes” welcher die Funktion oder Methode aufrufbar über SOAP als Web Service macht (die existierende Logik wird nicht verändert)
    - Den Code In einen Web Application Server ,deployen‘



# Web Services mit Java oder MS .Net verwenden



## Web Services in und mit VSE verwenden



**Tool zur Generierung des Proxy Codes**

## Wie wird der Proxy-Code erzeugt?

- Sie können den Proxy-Code “von Hand” schreiben
  - Ist nicht sehr schwierig, anhand von Beispielen
  - COBOL Beispiel - in zJournal.com beschrieben von Rich Smrcina, mit Source Code unter: [ftp://ftp.software.ibm.com/eserver/zseries/zos/vse/download/xmps/soap\\_cobol\\_rsmrcina.zip](ftp://ftp.software.ibm.com/eserver/zseries/zos/vse/download/xmps/soap_cobol_rsmrcina.zip)
  - Empfohlen bei komplexen Schnittstellen
  
- CICS2WS Tool
  - Generiert den Proxy-Code und WSDL Dateien
  - Der Proxy-Code wird in Assembler erzeugt
    - Es wird kein zusätzlicher kostenpflichtiger Compiler benötigt (z.B. COBOL oder PL/I)
    - Der Code ist sehr einfach
    - Normalerweise muss der Code nicht angepasst werden
  - <http://www.ibm.com/servers/eserver/zseries/zvse/downloads/>

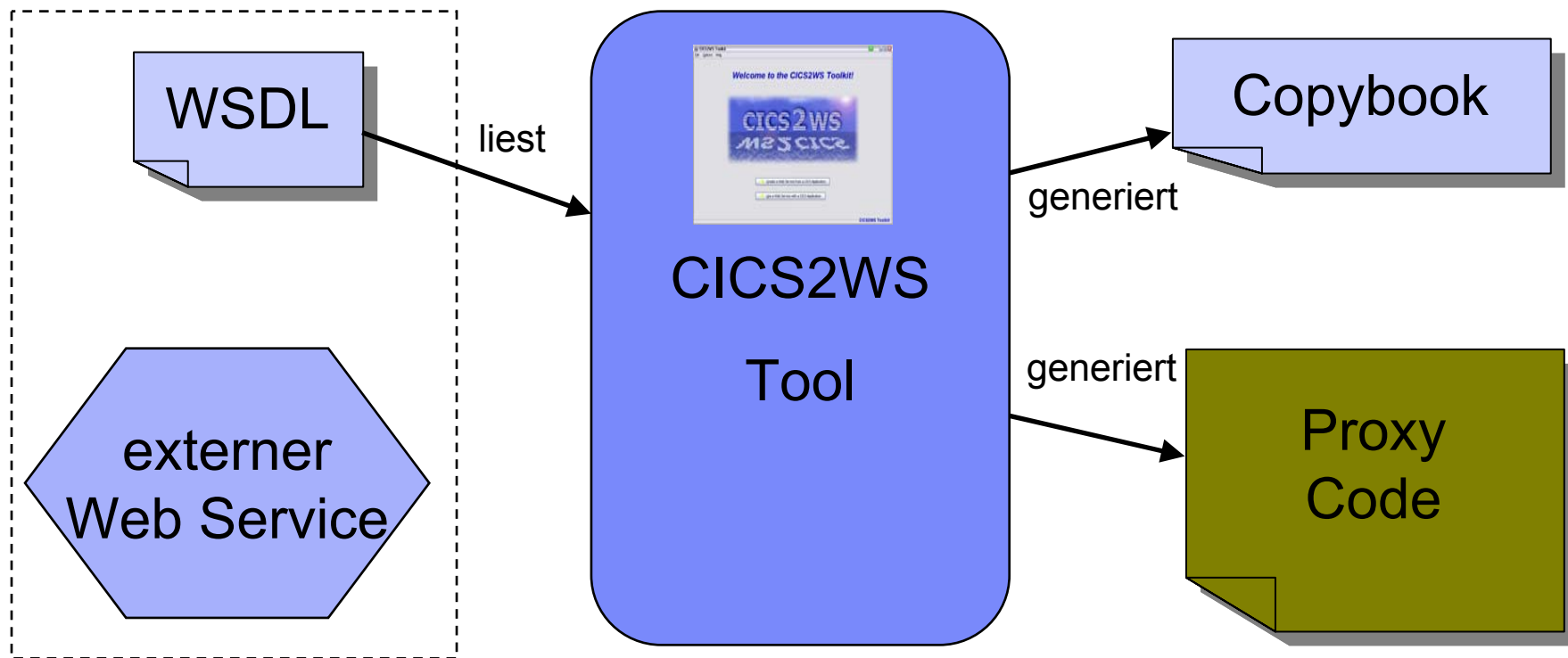


## CICS2WS Tool

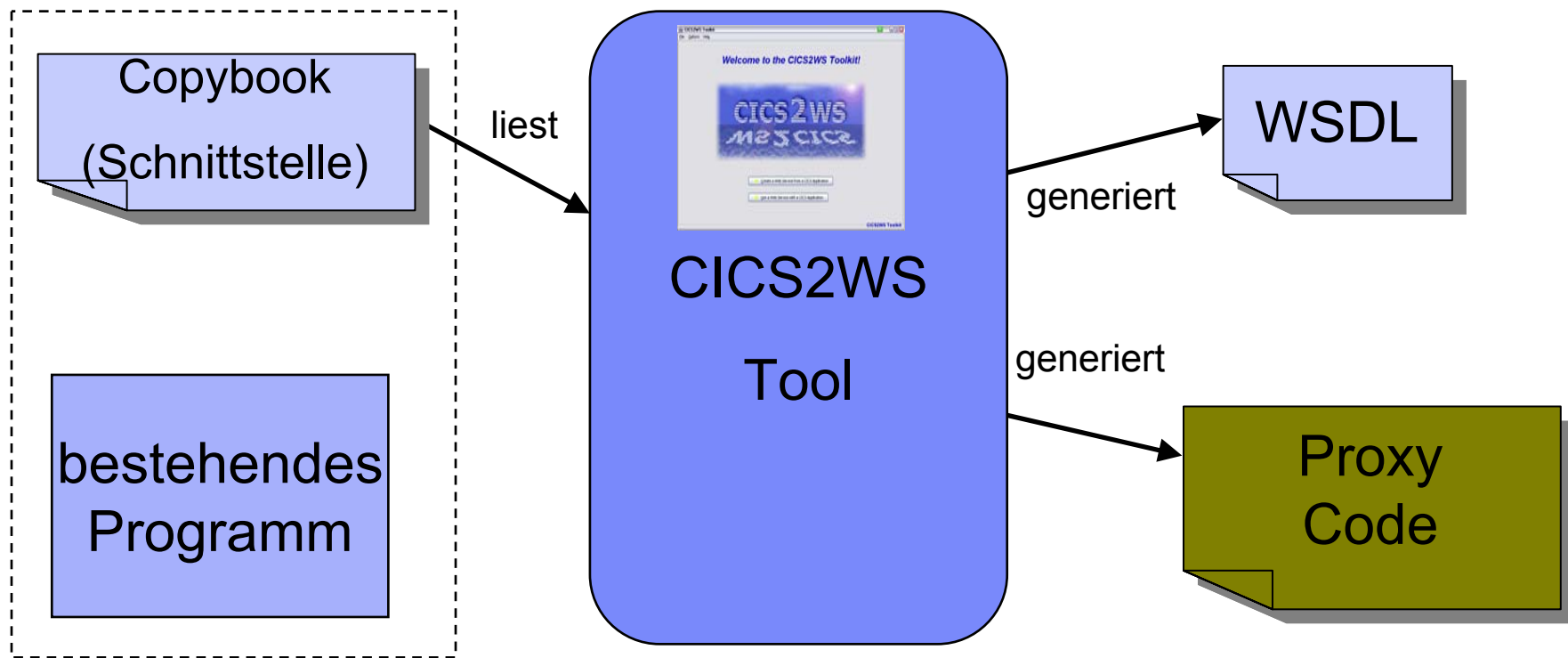
- Das Tool läuft auf einer Java Plattform
- Implementiert in Java
- VSE als SOAP Server (Service Provider)
  - Liest ein vorhandenes COMMAREA Mapping ein (Copybook)
    - in COBOL, PL/I oder Assembler
  - Generiert den Proxy-Code (Assembler)
  - Generiert die WSDL Datei
- VSE als SOAP Client (Service Requestor)
  - Liest die WSDL Datei ein
  - Generiert den Proxy-Code (Assembler)
  - Generiert ein COMMAREA Mapping (Copybook)
    - in COBOL, PL/I oder Assembler



## VSE als SOAP Client (Service Requestor)



## VSE als SOAP Server (Service Provider)





# Workshop

- Wir erstellen eine Web Service Anwendung im VSE
  
- Wie fangen wir an ?
  - Entweder eine bestehende Anwendung aussuchen
  
  - Oder eine CICS Common Area definieren – und daraus eine WSDL und Soap Anwendung generieren – dann den Service programmieren.

## Workshop

- Mit Methode 2)
  - Ein Copy Buch mit einer Common Area Definition erstellen und auf PC speichern als: **WSADDCOM.A**

```

COMMAREA DSECT
FUNC DS CL4
I DS F
J DS F
RESULT DS F
END
    
```

Methode soll sein:

```

ADD
I + J = RESULT
    
```

## Workshop

- Diese Definition für CICS2WS als Eingabe bereitstellen:



# Workshop

- Daten eingeben:

The screenshot shows the 'CICS2WS Toolkit' dialog box with the following fields and annotations:

- Path to Source:** D:\ibm\411\CICS2WS\WSADD\WSADD.COM.asm (Red arrow pointing to the field)
- Source Code Language:** ASM Source (Red arrow pointing to the dropdown menu)
- Skip control characters:**  (Red arrow pointing to the checkbox)
- Parse Source:** Button (Red arrow pointing to the button)
- Basic Service Data:**
  - Service Name:** CICS Calc (Red arrow pointing to the field)
  - Service URL:** http://192.168.3.1:1087/cics/CWBA\IESSOAPS (Red box around the field with text: "Hier z/VSE IPAdress +Port eingeben")
  - URN:** urn:IESSOAPD:SOAPCD
- Program Names:**
  - Proxy Name:** SOAPCD (Red arrow pointing to the field)
  - User Program Name:** MYPROG (Red arrow pointing to the field)

Buttons at the bottom: Back, Next, Cancel, Help.

# Workshop

- Service definieren:

**CICS2WS Toolkit**

File Options Help

Create Service Add Operation to Service Service Summary

**New Operation**

Name: CAdd

Description:  
Cics Add Webservice Call. \$This code add 2 Variables und the result is the sum.\$

**Input/Output Parameter Mapping**

Content	Instance	Type	Length	Offset
COMMAREA Variables				
├─ S T FUNC	Field	STRING	4	0
├─ S T I	Field	INTEGER	4	4
├─ S T J	Field	INTEGER	4	8
└─ S T RESULT	Field	INTEGER	4	12

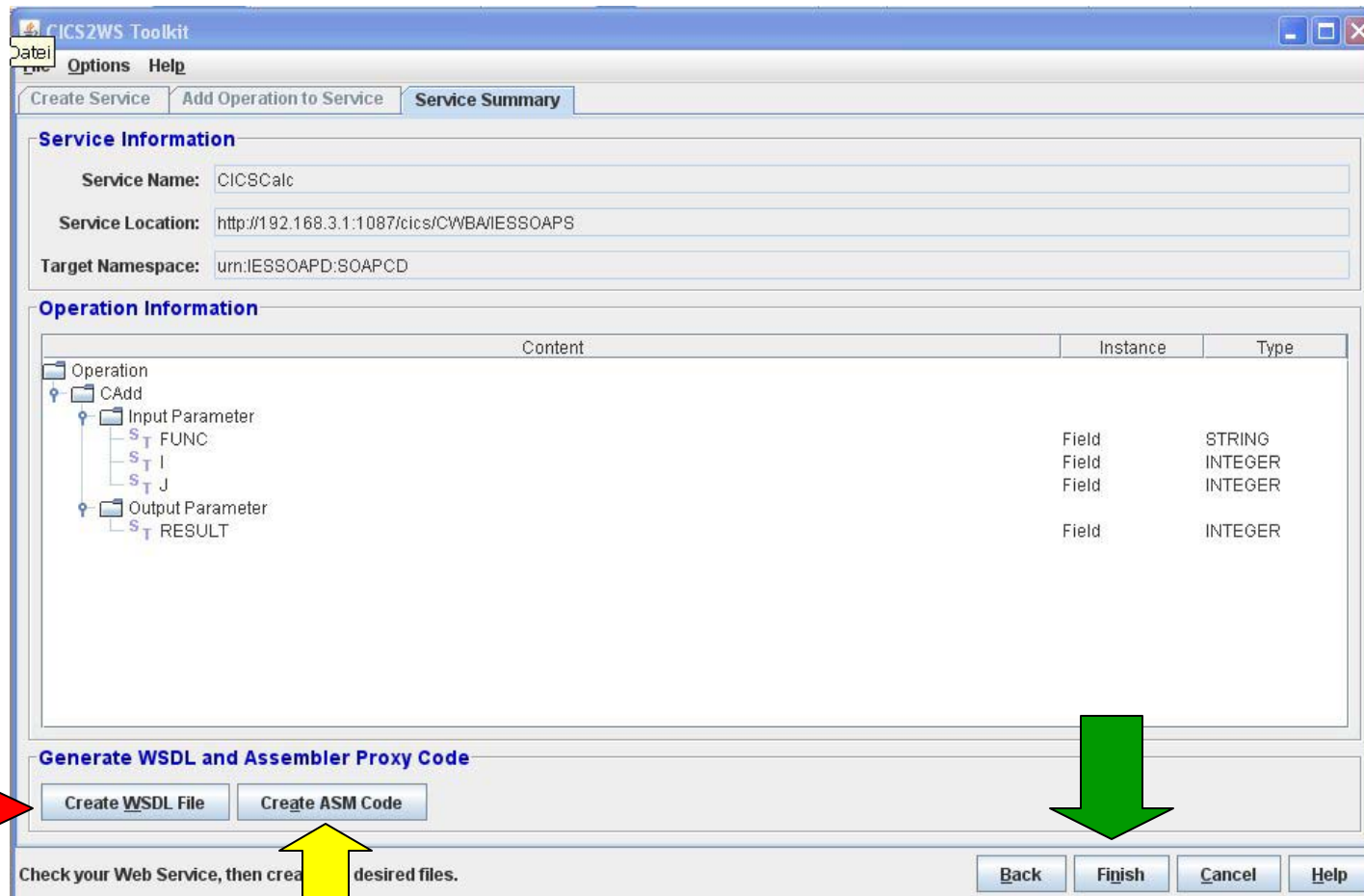
Content	Instance	Type	COMMAREA Name
Mapped Variables			
├─ Input Parameter			
├─ S T FUNC	Field	STRING	FUNC
├─ S T I	Field	INTEGER	I
├─ S T J	Field	INTEGER	J
└─ Output Parameter			
└─ S T RESULT	Field	INTEGER	RESULT

Provide an operation name, then select source and target in the trees to use the buttons.

Back Next Cancel Help

# Workshop

- WSDL und Proxy Code generieren:



# Workshop

- Code im VSE als Phase erstellen.
- Hierzu unbedingt die JCL verwenden, die generiert wurde.
- LibDef Phase ergänzen
- Den TCPIPService definieren
- Eigenes Programm schreiben.

# Workshop CEDA Definitionen

EXP G (GSEHTTP)

ENTER COMMANDS

NAME	TYPE	GROUP	DATE	TIME
MYPROG	PROGRAM	GSEHTTP	08.094	21.58.42
SOAPCD	PROGRAM	GSEHTTP	08.094	21.58.42
HTTPLATT	TCPIPSERVICE	GSEHTTP	08.094	21.58.42

SYSID=CICT APPLID=CICSTEST

RESULTS: 1 TO 3 OF 3

TIME: 21.59.00 DATE: 08.094

PF 1 HELP            3 END 4 TOP 5 BOT 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL



# Workshop

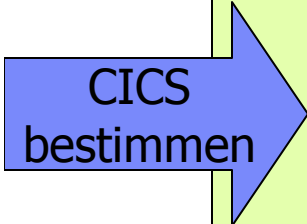
**OBJECT CHARACTERISTICS**

CICS RELEASE = 0411

CEDA View TCpipservice( HTTPLATT )

```

TCpipservice      : HTTPLATT
Group              : GSEHTTP
Description        : CICS Web TCPIPSERVICE
Urm                : DFHWBADX
Portnumber         : 01087           1-65535
Certificate        :
STatus             : Open           Open ! Closed
SSL                : No             Yes ! No ! Clientauth
Attachsec         : Local           Local ! Verify
TRansaction        : CWXN
Backlog            : 00005           0-32767
TSqprefix          : DFHWEB
Ippaddress         :
SOketclose         : No             No ! 0-240000
    
```



## Workshop

- Programm Code erstellen für MYPROG:

```

OPTIONS   PHASE MYPROG MAIN.
-D.
          CRESLT      9 0.
-I.
  FIELD CPGCOM.
          1   4   FUNC.
  BIN    5   8 0 CI.
  BIN    9  12 0 CJ.
-C.
  SELCT CPGCOM.
  CRESLT = CI + CJ.
  EDIT CPGCOM.
-O.
  FIELD CPGCOM
  CRESLT 16 BIN.

```

\* Programm Name

\* Daten Struktur

\* Common Area

\* Funktion !

\* Daten Übergabe in

\* Common Aea

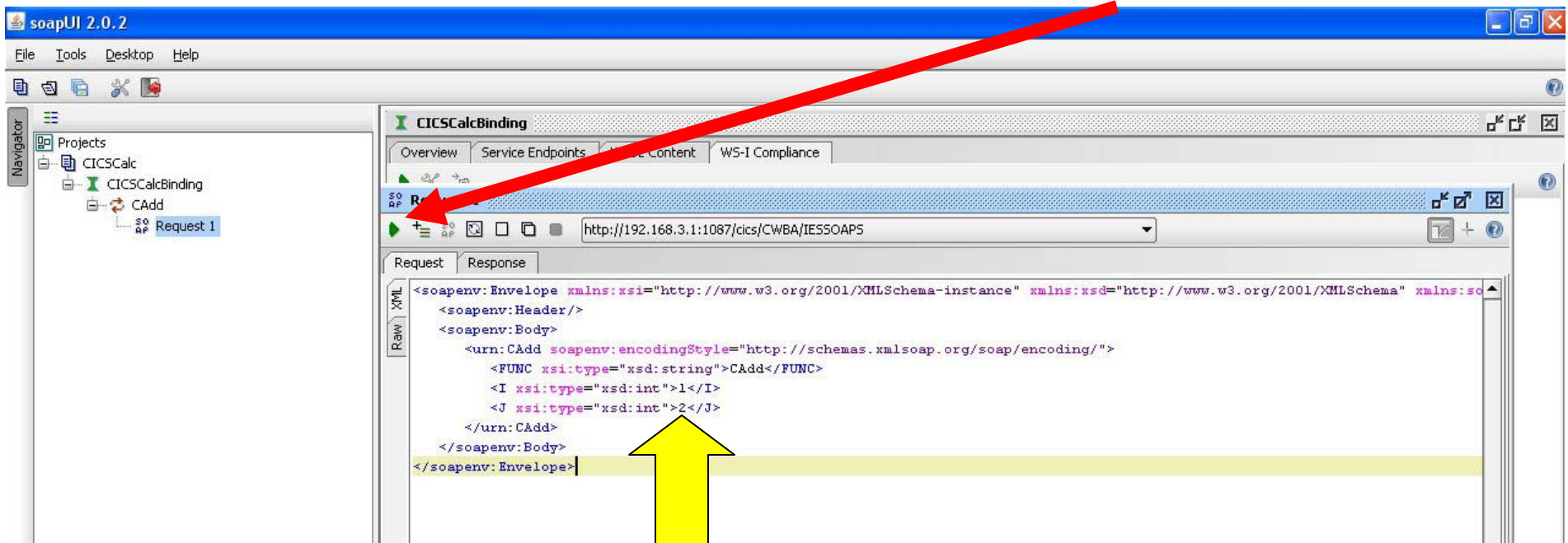
## Workshop

- Webservice mit SOAPUI testen:
- Hierzu von [www.soapui.org](http://www.soapui.org) die Anwendung Download und installieren.
- Den WSDL Code als Eingabe zur Verfügung stellen und Webservice testen.

# Workshop

- SOAPUI starten:

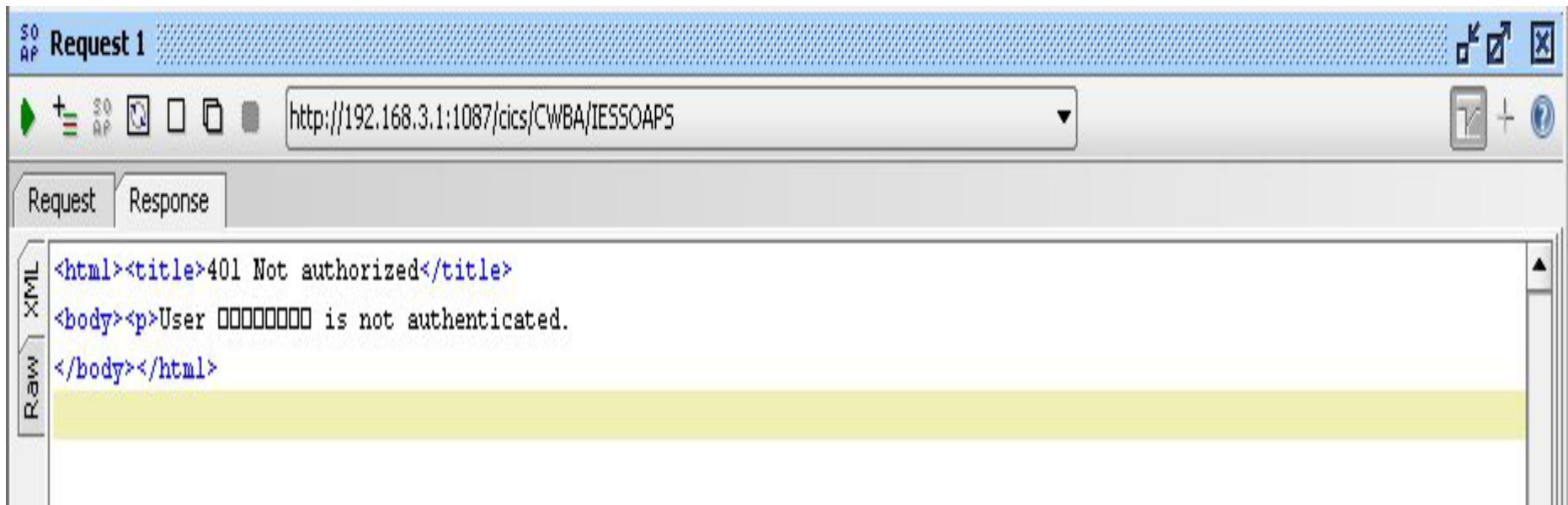
Dieses Icon  
klicken !



Hier Daten eingeben

# Workshop

- SOAPUI starten:



Erster Versuch ! Hier muss sich jemand anmelden .

# Workshop

- Passwort setzen!

The screenshot shows the soapUI 2.0.2 application window. On the left, the 'Request Properties' panel is visible, with a blue arrow pointing to the 'Password' field which contains the value 'GEHEIM'. The 'Request Properties' table is as follows:

Property	Value
Name	Request 1
Description	
Message Size	511
Encoding	iso-8859-1
Endpoint	http://192.168.3.1...
Bind Address	
Username	SYSA
<b>Password</b>	<b>GEHEIM</b>
Domain	
WSS-Password Type	
WSS TimeToLive	
Skip SOAP Action	false
Enable MTOM	false
Force MTOM	false
Inline Response At...	false
Expand MTOM Atta...	false
Disable multipart...	true
Encode Attachments	false
Enable Inline Files	false

The main panel shows the 'WSDL Definition' for 'CICSCalcBinding'. The 'Operations' section contains the following table:

Name	Use	One-Way	Action
CAdd	SOAP En...	false	

# Workshop

- Nochmal

Klick !

The screenshot shows the soapUI 2.0.2 interface. On the left, the Navigator pane shows a project tree with 'CICSCalc' containing 'CICSCalcBinding' and 'CAdd', which in turn contains 'Request 1'. A green arrow points from the text 'Klick !' to the 'Request 1' icon. The main window displays the XML view of the selected request, showing a SOAP envelope with a body containing a 'CAdd' operation with parameters 'I' (11) and 'J' (22). The address bar shows the URL 'http://192.168.3.1:1087/cics/CWBA/IESSOAPS'.

```

<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:CAdd soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <FUNC xsi:type="xsd:string">CAdd</FUNC>
      <I xsi:type="xsd:int">11</I>
      <J xsi:type="xsd:int">22</J>
    </urn:CAdd>
  </soapenv:Body>
</soapenv:Envelope>
  
```

# Workshop

- Ergebnis aus MYPROG

The screenshot shows the soapUI 2.0.2 interface. On the left, the Navigator pane shows a project structure with 'CICSCalc' containing 'CICSCalcBinding' and 'CAdd', with 'Request 1' selected. The main window displays the raw XML of the response for 'Request 1' at the URL 'http://192.168.3.1:1087/cics/CWBA/IESSOAPS'. The XML is as follows:

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:CAddResponse soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <RESULT xsi:type="ns1:int" xmlns:ns1="http://www.w3.org/2001/XMLSchema">33</RESULT>
    </urn:CAddResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

A yellow arrow points from the '33' value in the XML to the right. Below the arrow, a hand icon is shown with its palm facing up, as if presenting the result.



# Workshop

- Dieser Datenstrom wird zum VSE gesendet.

```

Dokument3 *
POST /cics/CWBA/IESSOAPS HTTP/1.1
Content-Type: text/xml;charset=iso-8859-1
SOAPAction: ""
User-Agent: Jakarta Commons-HttpClient/3.0.1
Host: 192.168.3.1:1087
Content-Length: 511

<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w
  <soapenv:Header/>
  <soapenv:Body>
    <urn:CAdd soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <FUNC xsi:type="xsd:string">CAdd</FUNC>
      <I xsi:type="xsd:int">12</I>
      <J xsi:type="xsd:int">22</J>
    </urn:CAdd>
  </soapenv:Body>
</soapenv:Envelope>

```

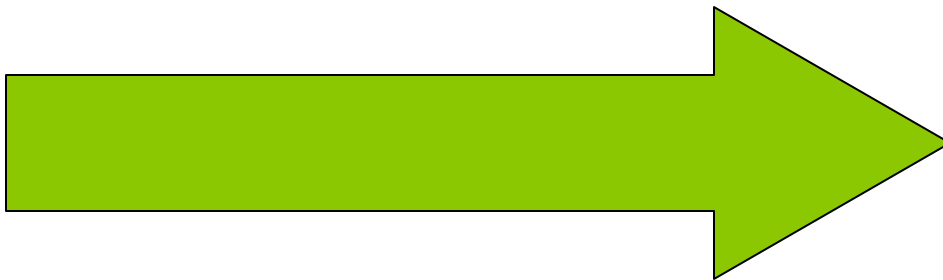
## Workshop Teil 2

- So nun wissen wir wie es geht.
- Wir lesen Adressdaten.
- Funktion: **READ**
- Eingabe Parameter: **KDNRA**
- Ausgabe Parameter:
  - **FIRMA, PLZ, ORT, STRASSE**

## Workshop Teil 2

- Das Copy für die Common Area: **CAKUND.ASM**

Wir erstellen ein neues  
Assembler Copy Buch  
mit der Struktur der  
Common Area.



```

Commarea DSECT
FUNC DS CL5
KDNRA DS CL5
PLZ DS CL5
FIRMA DS CL30
ORT DS CL20
STRASSE DS CL25
RESULT DS F
END
    
```

# Workshop Teil 2

- CAKUND.A ist Eingabe für CICS2WS

The screenshot shows the 'CICS2WS Toolkit' application window. It has a menu bar with 'File', 'Options', and 'Help'. Below the menu bar are three tabs: 'Create Service', 'Add Operation to Service', and 'Service Summary'. The 'Create Service' tab is active.

The interface is divided into several sections:

- CICS Application Data:**
  - Path to Source:** A text box containing 'D:\ibm\411\CICS2WS\KDREAD\CAKUND.asm', highlighted with a red box. A 'Browse...' button is to its right.
  - Source Code Language:** A dropdown menu set to 'ASM Source'.
  - Skip control characters:** An unchecked checkbox.
  - Parse Source:** A button that has been clicked, resulting in a green message box that says 'Successfully parsed the source file.', highlighted with a blue box.
- Basic Service Data:**
  - Service Name:** 'ReadKunde'
  - Service URL:** 'http://192.168.3.1:1087/cics/CWBA/IESSOAPS'
  - URN:** 'urn:IESSOAPD:SKDREAD'
  - Service Description:** A large empty text area.
- Program Names:**
  - Proxy Name:** 'SKDREAD', highlighted with a red box.
  - User Program Name:** 'KDREAD', highlighted with a red box.

At the bottom of the window, there is a status bar with the text: 'Browse for a source file, parse it and then set Basic Service Data and Program Names.' To the right of the status bar are four buttons: 'Back', 'Next', 'Cancel', and 'Help'.

# Workshop Teil 2

- Auswahl Input / Output Parameter

**New Operation**

Name:

Description:

**Input/Output Parameter Mapping**

Content	Instance	Type	Length	Offset
COMMAREA Variables				
-S T FUNC	Field	STRING	5	0
-S T KDNRA	Field	STRING	5	5
-S T PLZ	Field	STRING	5	10
-S T FIRMA	Field	STRING	30	15
-S T ORT	Field	STRING	20	45
-S T STRASSE	Field	STRING	25	65
-S T RESULT	Field	INTEGER	4	90

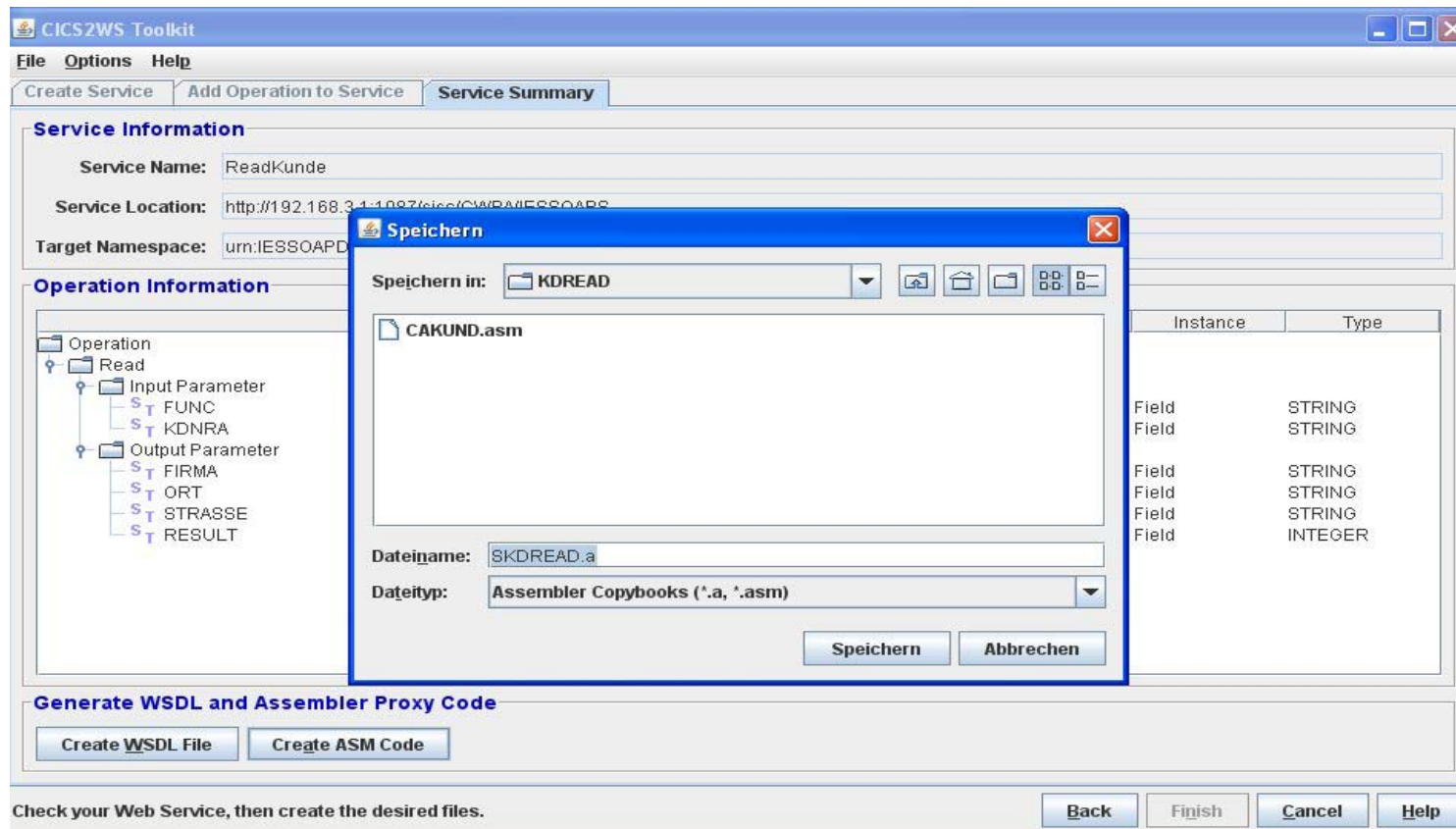
Content	Instance	Type	COMMAREA Name
Mapped Variables			
Input Parameter			
-S T FUNC	Field	STRING	FUNC
-S T KDNRA	Field	STRING	KDNRA
Output Parameter			
-S T FIRMA	Field	STRING	FIRMA
-S T ORT	Field	STRING	ORT
-S T STRASSE	Field	STRING	STRASSE
-S T RESULT	Field	INTEGER	RESULT

Provide an operation name, then select source and target in the trees to use the buttons.

Buttons: Back, **Next**, Cancel, Help

# Workshop Teil 2

- Speichern des generierten Proxy Codes



## Workshop Teil 2

- Wie gehabt: Proxy Code erstellen
- Im CEDA definieren und Installieren.
- WSDL im SoapUI einlesen
  
- **T e s t e n !**

## Workshop Teil 2

- Anwendung erstellen:
- Dieses Programm bereitet die Common Area so auf, wie es der Webservice bzw. das Proxy Programm es erwartet.

```

OPTIONS MAIN PHASE KDREAD.
FILE CPGKDN.
-D. RESULT 9 0.
-I. FILE CPGKDN DD.
   FIELD CPGCOM.
       1 5 FUNC.
       6 10 KDNRA.
-C. SELCT CPGCOM.
   KDNRA CHAIN CPGKDN.
   IF CPGFRC = ' '.
       RESULT = 0. * FOUND.
   ELSE.
       RESULT = 8. * NOT FOUND.
   ENDIF.
   EDIT CPGCOM.
-O. FIELD CPGCOM
   PLZ 15.
   FIRMA 45.
   ORT 65.
   STR1 90.
   RESULT 94 BIN.

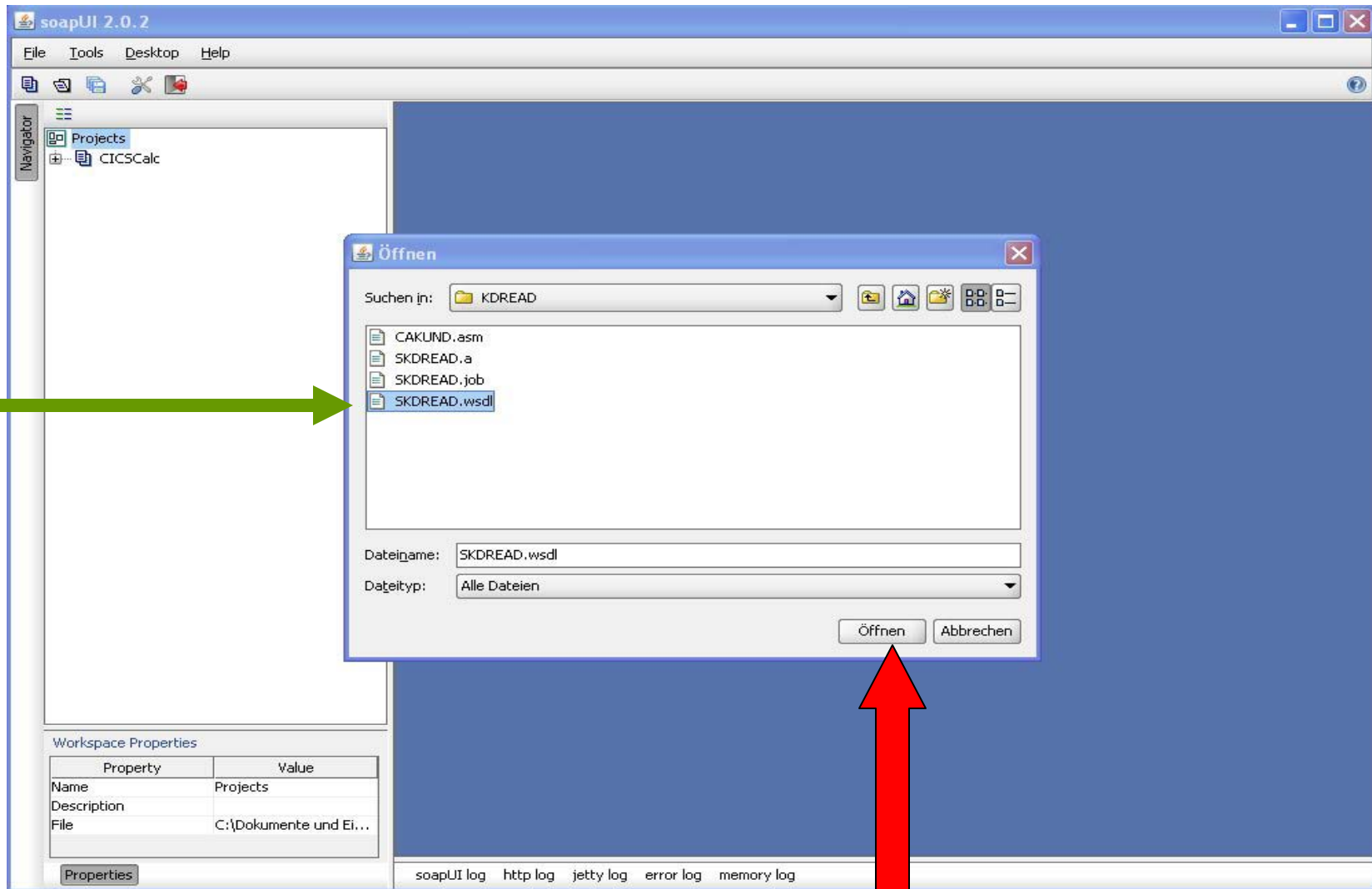
```

Input

Output

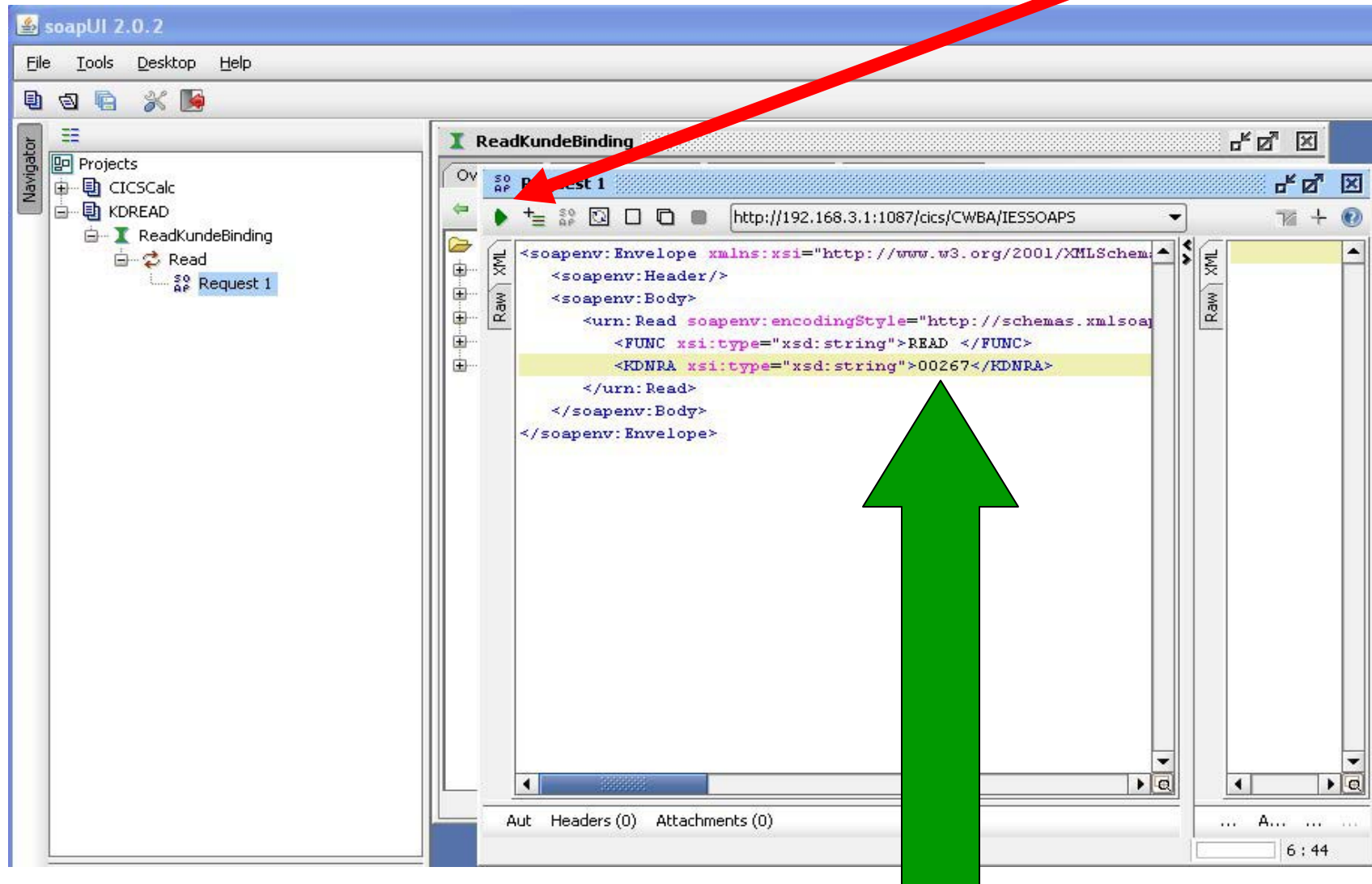


## Workshop 2 SoapUI starten und SKDREAD aufrufen



# Workshop 2 - Eingabe KDNR

Klick!



# Workshop 2 Ergebnis !

http://192.168.3.1:1087/cics/CWBA/IESSOAPS

```

<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:ReadResponse soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <FIRMA xsi:type="ns1:string" xmlns:ns1="http://www.w3.org/2001/XMLSchema">Wessels + M}ller AG</FIRMA>
      <ORT xsi:type="ns1:string" xmlns:ns1="http://www.w3.org/2001/XMLSchema">Osnabr}ck</ORT>
      <STRASSE xsi:type="ns1:string" xmlns:ns1="http://www.w3.org/2001/XMLSchema">Pagenstecherstr. 121</STRASSE>
      <RESULT xsi:type="ns1:int" xmlns:ns1="http://www.w3.org/2001/XMLSchema">0</RESULT>
    </urn:ReadResponse>
  </soapenv:Body>
</soapenv:Envelope>
  
```

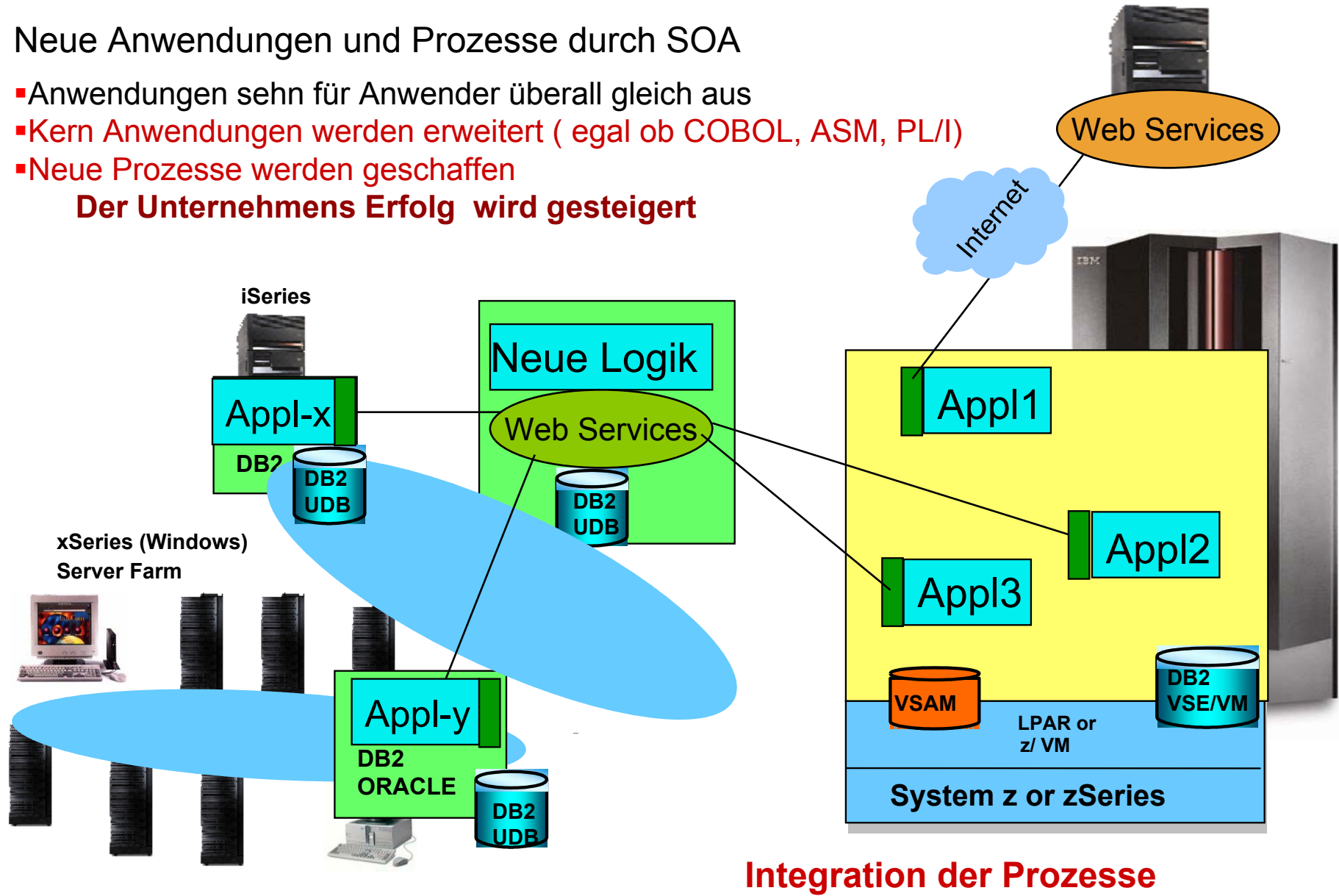
Headers (3) Attachments (0) SSL Info WSS (0)

es) 1:1

## Neue Anwendungen und Prozesse durch SOA

- Anwendungen sehen für Anwender überall gleich aus
- Kern Anwendungen werden erweitert ( egal ob COBOL, ASM, PL/I)
- Neue Prozesse werden geschaffen

**Der Unternehmens Erfolg wird gesteigert**



Noch Fragen ???

**Vielen Dank**  
**für Ihre**  
**Aufmerksamkeit**