



7363 - Web-basierte Anwendungen: **Übung 07, Projektbeginn**

WSDL-Vorbereitung
Einstieg in's Projekt



Ziele



- **Übungen**
 - Wo notwendig: Installation der für WSDL erforderlichen Komponenten
 - Elementare WSDL-Tests
 - Projektarbeit: Design und Schema-Entwicklung für die Anmeldung
- **Ziele**
 - Vorbereitung für die Projektarbeit mit WSDL
 - Einstieg in's Projekt



WSDL-Tests



WSDL-Tests



- Hintergrund
 - Die Internet-Suchmaschine Google kann auch mittels SOAP als Web Service verwendet werden - wenn auch noch im Beta-Test.
 - Die Schnittstelle wird per WSDL beschrieben.
 - Zur Nutzung mit bis zu 1000 Anfragen pro Tag (!) ist eine (kostenlose) Anmeldung bei Google erforderlich.
 - Im Rahmen der Übung 06 haben Sie sich bereits bei <http://www.google.com/apis> registriert, und das Entwicklerpaket liegt Ihnen vor.
- **Aufgabe**
 - Generieren Sie eine einfache Client-Anwendung für Google mittels der bereitstehenden WSDL-Datei.
 - Führen Sie einige Suchanfragen für Testzwecke durch!
 - Schneiden Sie den HTTP-Datenstrom mit (tcpmon) und überprüfen Sie den Zusammenhang zwischen WSDL-Beschreibung und Aufbau der SOAP Body-Elemente.

Sie sollten dies bei Nachfrage vorführen können!



- Hinweise
 - Verwenden Sie möglichst einen Code-Generator aus Ihrer Entwicklungsumgebung, der aus der WSDL-Datei den Rumpf einer Client-Anwendung erzeugen kann.
 - Im Fall von Ruby ist das z.B. "wsdl2ruby.rb", zu finden unter `~/lv/wba/07/soap4r-1_5_5/bin/wsdl2ruby.rb`
 - Ruby-Alternative: Factory-Klasse `SOAP:WSDLDriverFactory` verwenden
 - Einstellungen für den HTTP-Proxy beachten!
 - Im Fall von Ruby:

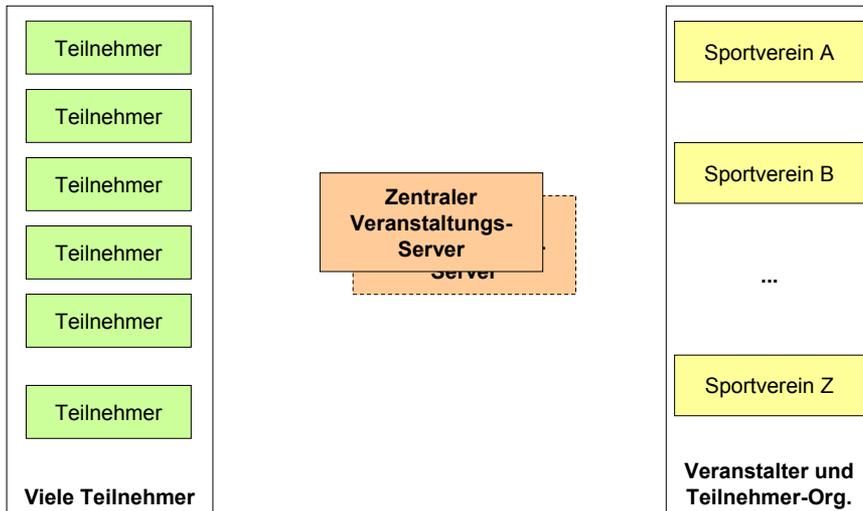
```
$ export HTTP_PROXY=$http_proxy  
$ export SOAP_USE_PROXY=on
```



Einstieg in's Projekt



- Die Akteure



- Die Akteure

- Teilnehmer

- Sind (natürliche) Personen
- Besitzen Chip-ID
- Erhalten bei Anmeldung eine Startnummer
- kommunizieren (nur) per Web Browser + E-Mail
- Sind möglicherweise Mitglieder in Sportvereinen

- Sportvereine

- Besitzen bzw. erhalten eine Vereins-Kennung
- Besitzen weitere Stammdaten (Anschrift, Kontaktperson, ...)
- Verwalten ihre Mitglieder (Mitglieds-Stammdaten)
- Betreiben eine WS-gestützte Anwendung zur Wettkampfverwaltung
- Treten manchmal als **Veranstalter** auf
- Melden oftmals Vereinsmitglieder als Teilnehmer bei Veranstaltungen an
- Interessieren sich für die Ergebnislisten der von ihnen besuchten Veranstaltungen und speziell für die Ergebnisse ihrer Mitglieder



- **Die Akteure**

- **Zentraler Veranstaltungs-Service / -Server**

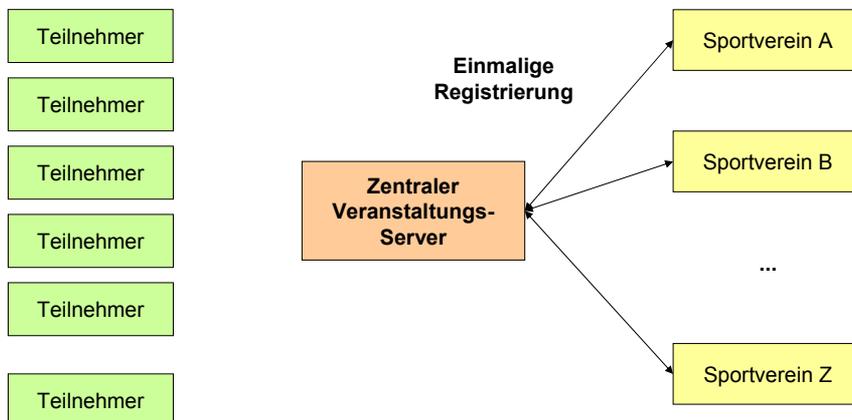
- Registriert Sportvereine (keine Einzelpersonen)
- Registriert neue Veranstaltungen
- Verwaltet Stammdaten und Ergebnisdaten von Veranstaltungen
- Sammelt & speichert Ergebnislisten, auch inkrementell
- Ermöglicht Abruf von Ergebnislisten bzw. Teilen davon
- Unterstützt Urkundendruck und Listendruck
- Berechnet Statistiken, stellt Auswertungsgrafiken bereit
- Verfügt über WS-Schnittstellen (für die Anwendungen der Vereine) als auch über Web *Front-ends* (für die Teilnehmer)

- Warum "zentral"?

- Vorteile zentraler Datenhaltung und statistischer Auswertung
- Einheitliche Kennungsvergabe für Vereine
- Einheitliche Schnittstelle für alle Anwendungen
- Falls mehrere "Zentralen": Auf kompatible Schnittstellen achten...

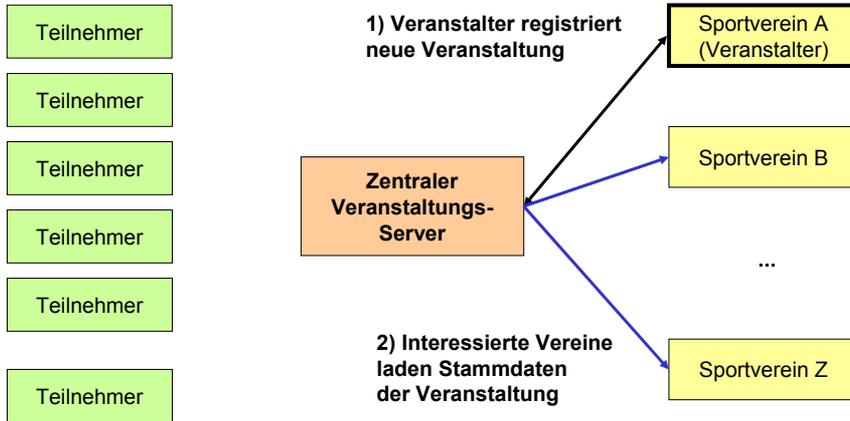


- **Das Szenario: Registrierung der Vereine**

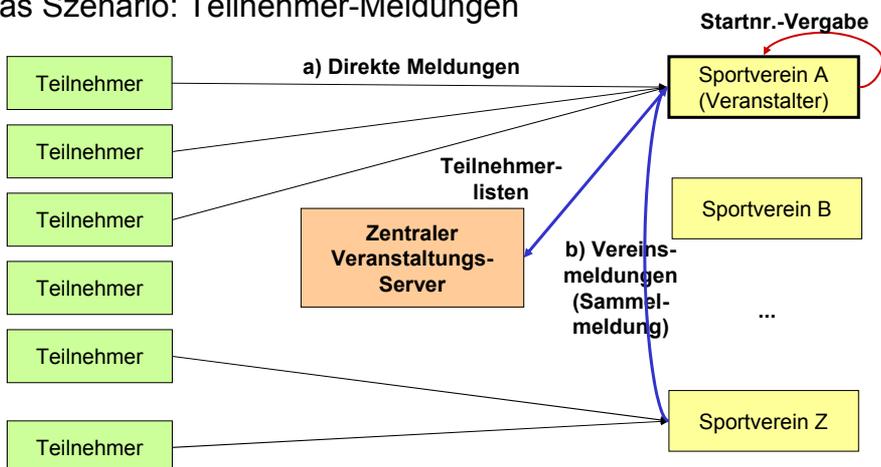




- Das Szenario: Ausschreibung einer Veranstaltung



- Das Szenario: Teilnehmer-Meldungen

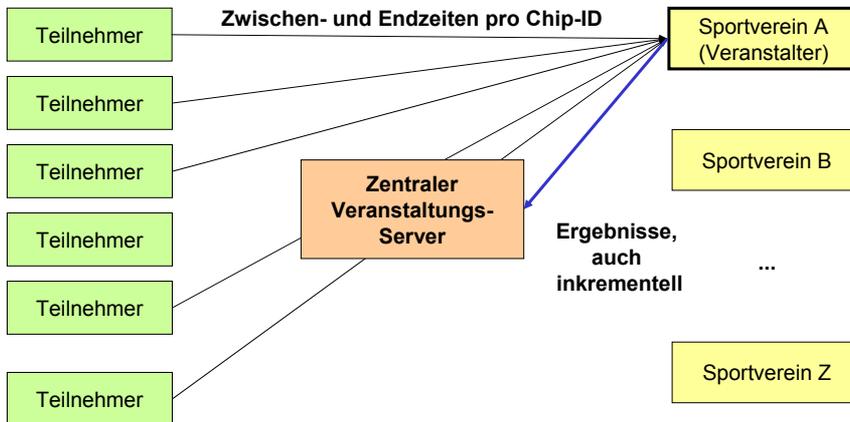




Einstieg in's Projekt



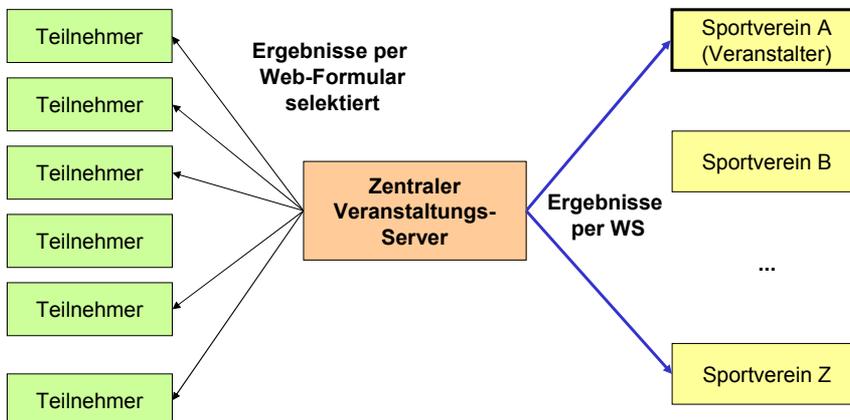
- Das Szenario: Wettkampf (-Simulation)



Einstieg in's Projekt

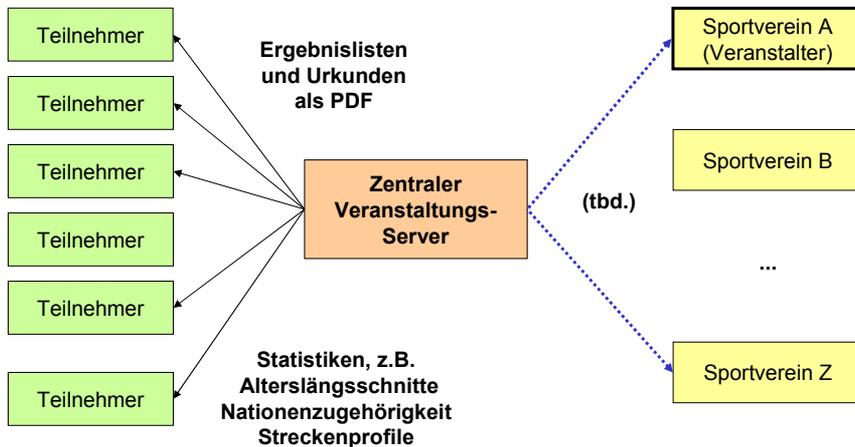


- Das Szenario: Ergebnisabruf





- Das Szenario: Weitere Auswertungen



- Komponenten der Akteure
 - Teilnehmer
 - Nur Internetzugang, UA (Browser) mit SVG-Unterstützung/-Plugin, PDF-Reader
 - Sportvereine
 - Software zur Mitgliederverwaltung und Anmeldung
 - Hält Mitglieds-Stammdaten vor
 - WS-Interface "Vereinsmeldungen" zu Organisatoren von Veranstaltungen
 - Web-Interface für Meldungen zu Teilnahmen über Vereinsmeldung
 - WS-Interface "V-Registrierung" von Vereinen beim Veranstaltungs-Server
 - WS-Interface "Ergebnisabruf" zum Veranstaltungs-Server
 - » ruft die Ergebnisse aller zuvor gemeldeten Vereinsmitglieder ab
 - Software zur Organisation von Veranstaltungen
 - Web-Interface für Einzelmeldungen
 - WS-Interface (Receiver) für Vereinsmeldungen
 - Startnummernvergabefunktion
 - WS-Interface "Teilnehmer": Übergabe von Teilnehmerlisten an V-Server
 - WS-Interface "Ergebnisse": Übergabe von Ergebnislisten an V-Server



- **Komponenten**
 - Zentraler Veranstaltungs-Server
 - WS-Interfaces
 - Registrierung eines Sportvereins
 - Anmeldung einer Veranstaltung
 - Teilnehmerlisten
 - (Mess-) Ergebnisse (Zwischen- und Endzeiten)
 - Ergebnislisten (Abruf/Selektion)
 - Web-Interfaces
 - Abruf / Selektion: Registrierte Veranstaltungen
 - Abruf: Ergebnislisten
 - » nach unterschiedlichen Kriterien ausgewählt, je 25 ... 50 pro Seite
 - » PL: PDF-Liste, via XSLT & XSL-FO
 - Abruf: Einzel-Ergebnisse (PL: PDF-Urkunde, via XSLT & XSL-FO)
 - Abruf: Histogramm (SVG) - % finisher vs. Endzeit, int. oder differenziert
 - Datenhaltung:
 - tbd. Filesystem? Persistente Hashes? RDBMS? ORM/Rails? XML/Tamino?!



- **Aufgaben für Übung 07:**
 - Datenmodellierung mittels XML Schema für die WS-Schnittstellen**
 - Mitgliederverwaltung
 - Registrierungs-Interface. Mitteilen:
 - Vereinsname, Anschrift (incl. Land), E-Mail, ... (was fehlt?)
 - Veranstaltungs-Server
 - Registrierungs-Interface. Antworten:
 - Kennung (Vereins-ID), Kennwort. Als CGI über HTTPS aufsetzen?
 - Datenmodellierung
 - "Veranstaltungsmeldung"
 - "Veranstaltungsliste" (Abrufergebnis)
 - "Teilnehmerliste"
 - "Ergebnisse" (für die Meldung der Zeiten)
 - "Ergebnisanfrage"
 - "Ergebnisliste" (für die ausführlichen Abrufe der Ergebnisse)



- Datenmodellierung, "Veranstaltungsmeldung"
 - Kopfdaten
 - Name der Veranstaltung
 - Ort und Datum der Veranstaltung (auch Zeitraum möglich)
 - Organisator (ein registrierter Verein, evtl. per Referenz)
 - Liste der ausgetragenen Wettkämpfe
 - Wettkampf (mehrere!)
 - Typ:
 - Etwa Hauptlauf, Skater, Rollis, Handbiker, Power-Walker, Bambini
 - Wertungen:
 - ggf. Frauen / Männer getrennt bzw. egal: M, W, X
 - Altersklassen (z.B. MJA, MHK, M30, ..., M75; WJA, WHK, W30, ..., W75)
 - Sonderwertungen (optional):
 - Vereinsmeisterschaften
 - Dt. Meisterschaften etc.
 - [Mannschaftswertungen (je 3 oder 4 eines Vereins, m/w/mixed)]
 - (s. Forts.)



- Datenmodellierung, "Veranstaltungsmeldung"
 - Wettkampf (Forts.)
 - Datum
 - Startzeit
 - Ortsangabe Start (Freitext)
 - Streckenlänge (in km)
 - Liste der Zwischenzeit-Messpunkte
 - gemeint sind km-Angaben
 - Die Pflicht-Zeitmessungen bei Start und Ziel hier nicht angeben.
 - Beispiel Marathon: 5, 10, 15, 20, 21.1, 25, 30, 35, 40
- Datenmodellierung, "Veranstaltungsliste"
 - Liste von "Veranstaltungsmeldung"-Elementen
 - (Selektierbar über Zeitraum und/oder Ortsangabe)



- Datenmodellierung, "Teilnehmerliste"
 - Veranstaltungs- und Wettkampf-Schlüssel
 - Vereine
 - Verein
 - Name
 - Anschrift
 - E-Mail
 - Kennung
 - Teilnehmer
 - Name, Vorname, MI, Geschlecht (M, W)
 - Anschrift, E-Mail
 - Jahrgang oder Geburtstag
 - AK (Altersklasse, vom Veranstalter aus Geschlecht und Jahrgang bzw. Geburtstag abzuleiten)
 - Verein (Kennung / Referenz)
 - Chip-ID
 - Startnummer (vom Veranstalter zuzuweisen)



- **Gruppenarbeit** beim zentralen Server:
 - Die ersten drei WS-Schnittstellen des zentralen Veranstaltungs-Servers sind interoperabel zu halten: Die Veranstalter-Software der Gruppe X muss also mit dem zentralen Server der Gruppe Y zusammen funktionieren können.
 - Alle Teams müssen sich daher über die Schnittstellen einig werden und erarbeiten dazu einen gemeinsamen Satz von WSDL-Dateien!
 - Dazu gehört auch die Abstimmung über RPC- oder Dokumentenmodus (SOAP- bzw. WSDL-Ebene).
 - Bem.: Natürlich differieren die Lösungen bezüglich der URL-Angaben...
 - Falls erforderlich bzw. sinnvoll: XML Schema-Anteile in WSDL verwenden!
 - Bearbeiten Sie die Schnittstellen in ihrer chronologischen Folge.
- Diskutieren Sie die Schnittstellen und Anforderungen rege!
 - Jetzt erkannte Schwachstellen ersparen Ihnen später viel Mühe...
 - **Verständnisfragen zu den Spezifikationen sollten *jetzt* geklärt werden, nicht erst in ein paar Wochen!**



- Datenmodellierung, "Ergebnisse",
Datenmodellierung, "Ergebnisanfrage",
Datenmodellierung, "Ergebnisliste"

Diese werden ebenfalls benötigt. Wir klären nächste Woche, ob hierzu WSDL- und/oder Schema-Dateien zu erarbeiten sind.

- **Allgemeine Hinweise**

- Legen Sie geeignete Namensräume fest, insb. für eigene Datentypen und Elemente!
- Diese Namensräume sollten in den SOAP-Dokumenten später auch benutzt werden.
- Modellieren Sie möglichst präzise, z.B. unter Verwendung von XML-Schema-Datentypen oder auch von eigenen.
- Tipp: Installieren Sie Ihre Schemadatei unter dem Namensraum-URI