

# AopWeb

**Die WEB - Anbindung**

für

**ASAM-ODS V3.x Server**

Typ	Version	Datum	Verfasser
Anwenderhandbuch	2.11	Oktober 2002	Horst Fiedler

## Zusammenfassung

Diese Applikation zeigt Möglichkeiten der Integration von ASAM-ODS in das Internet. Die Schwerpunkte sind:

- HTTP Anbindung von ASAM-ODS (Protokollschicht) unter Verwendung von SSL, Session-Mapping, ...
- Behandlung von Mimetypes wie in ASAM-ODS BLOB-Attributen als via HTTP zugreifbare Ressourcen
- XML Unterstützung für Datenexport und Import
- Formeln
- Ausgabe für Microsoft Excel

Das Programm selbst besteht aus einem Servlet, der Client benötigt nur einen WWW-Browser (IE5, Navigator, ...). Ein im Paket inkludiertes Applet demonstriert die Möglichkeit die Daten nicht nur tabellarisch zu präsentieren sondern im Browser zu verarbeiten. Als getrennt lieferbare Anwendungen sind z.Z. ein Messdateneditor, Datentransferanwendungen und Hilfswerkzeuge (z.B. ein Messdatenvalidierer mit dem Flags abhängig von den Datenwerten gesetzt werden können) integrierbar.

## Hinweise

Dieses Handbuch ist unvollständig und kann Fehler enthalten (Status: Preliminary). Es sollte aber ausreichend sein, um die Anwendung installieren und betreiben zu können.

## Über die Autoren

Hr. H. Fiedler ist via E-Mail erreichbar [horst.fiedler@tiff.com](mailto:horst.fiedler@tiff.com). bzw. über die Internetseiten der Firma <http://www.tiff.com/tiff/>.

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
<b>2</b>	<b>ASAM-ODS Daten Navigator</b>	<b>3</b>
2.1	Aktivierung . . . . .	3
2.2	Authentizierung . . . . .	4
2.2.1	Authorisierung . . . . .	5
2.2.2	Einstellungen . . . . .	6
2.3	Applikationselementseite . . . . .	8
2.3.1	Navigation . . . . .	9
2.3.2	Binäre Objekte . . . . .	10
2.3.3	Zusatzfunktionen . . . . .	10
2.3.4	Auswahl von Sichten . . . . .	10
2.4	Meßdaten . . . . .	12
2.4.1	Tabellenansicht . . . . .	12
2.4.2	Graphische Präsentation . . . . .	13
2.4.3	Messdateneditor . . . . .	14
<b>3</b>	<b>Standard-Erweiterungen</b>	<b>15</b>
3.1	Tabellenformat . . . . .	16
3.2	Anwenderadresse . . . . .	16
3.3	Logdateien . . . . .	16
3.4	Datentransfer/XML . . . . .	16
3.4.1	Datenmodell . . . . .	16
3.4.2	Exportierte Daten holen . . . . .	17
3.4.3	XSL-Transformation . . . . .	17
3.5	Microsoft EXCEL . . . . .	18
3.5.1	Vorlageformat . . . . .	18
3.5.2	Durchführung . . . . .	20
3.5.3	Aktuelle Restriktionen . . . . .	20
3.6	Formeln . . . . .	21
3.6.1	Begriffe . . . . .	21
3.6.2	DTD . . . . .	23
3.6.3	Sprachumfang . . . . .	25
3.6.4	Integration in AopWeb . . . . .	25

<b>4</b>	<b>Integration von Anwendungen</b>	<b>29</b>
4.1	Erweiterung mit SOAP . . . . .	30
4.2	Applets . . . . .	30
4.3	Externe Prozesse . . . . .	31
4.3.1	Registrierung von Anwendungen . . . . .	32
4.3.2	Starten von Anwendungen . . . . .	32
4.3.3	Programmausgaben . . . . .	32
4.3.4	Musteranwendungen . . . . .	32
4.3.5	Erzeugen eigener Anwendungen . . . . .	33
<b>5</b>	<b>Installation</b>	<b>35</b>
5.1	Apache Tomcat . . . . .	35
5.1.1	Starten von Tomcat . . . . .	35
5.1.2	Konfigurieren . . . . .	36
5.1.3	Zugriffsberechtigungen . . . . .	36
5.1.4	Installation von AopWeb . . . . .	36
5.1.5	Parametrierung . . . . .	37
5.1.6	Fehlersuche . . . . .	38
<b>A</b>	<b>Anhang A</b>	<b>39</b>
A.1	Deployment descriptor . . . . .	39

# Kapitel 1

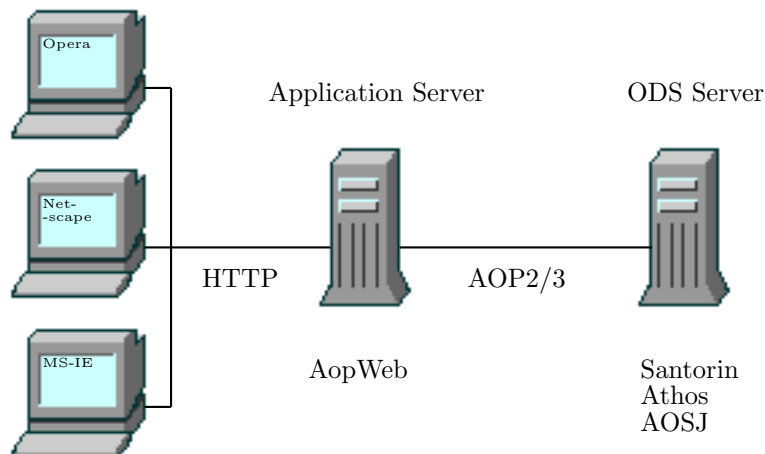
## Einleitung

*AopWeb* dient zur dynamischen Erzeugung von HTML Seiten die ASAM-ODS Daten in Form von Tabellen darstellen. Durch die Verwendung von Applets ist es sogar möglich große Datenmengen graphisch darzustellen.

*AopWeb* ist als Servlet implementiert. Servlets erweitern die Funktionen eines WWW-Servers (wie Apache oder IIS) indem sie Programmfunktionalität hinzufügen.

*AopWeb* kann ASAM-ODS Daten exportieren und diese zum Herunterladen als XML Datei anbieten.

*AopWeb* ist intern und extern erweiterbar. Mittels SOAP lassen sich z.B. Auswertungen oder Editoren anbinden, die *AopWeb* zur Navigation nutzen bzw. statt ONC-RPC oder CORBA das leichtgewichtigeren, über HTTP laufende SOAP verwenden wollen (Firewalltunnel).



**Abbildung 1.1:** ODS und Internetdienste

Die Daten werden von ODS zur Verfügung gestellt, jeder ODS Dienst mit Protokollschicht 2/3 kann von AopWeb verwendet werden.

Es können natürlich alle Dienste auf einem Rechner gestartet sein, in der Skizze sind sie getrennt dargestellt, um die Möglichkeit der über mehrere Rechner verteilten Datenverarbeitung aufzuzeigen.

AopWeb stellt neben der einfachen Datennavigation auch eine Infrastruktur bereit, in die andere Anwendungen eingegliedert werden können (Pluggins).

Aufgrund der Verwendung von normalen HTML Browsern (MS Internet Explorer, Netscape Navigator) ist keine Softwareinstallation auf den Client-Rechnern notwendig.

Eine Grundbedingung für den Erfolg wird neben der Einfachheit auch die Möglichkeit sein, Fremdapplikationen zu integrieren. Näheres zur Einbindung von AopWeb in andere Programme oder zur Verwendung als Browser der externe Applikationen mit ODS Daten versorgt in Kapitel 4.

Des weiteren kann man AopWeb als voll funktionstüchtige Studie für folgende ASAM-ODS Aspekte sehen:

- Glatte Integration von ASAM-ODS in die bestehende Web Architektur wobei die Protokollschicht (AOP V3) unverändert bleibt (zumindest in der ersten Phase bis die Erfahrungen zeigen in welche Richtung die nächste Generation der Schnittstelle sich entwickelt (CORBA-Beans, EJB, SOAP, ... ?)).
- ASAM-ODS und der XML basierten Datenaustausch. Die Hintergründe über die Unterschiede zwischen dem Zugang zu XML hier und dem in der ASAM-ODS Arbeitsgruppe diskutierten sind in der Dokumentation zu AOXS nachzulesen.

## Kapitel 2

# ASAM-ODS Daten Navigator

### 2.1 Aktivierung

AopWeb startet durch Aktivieren (Point-Click) eines auf AopWeb zeigenden Web-Links, der bereits alle notwendigen Angaben enthält, oder durch Eingabe des passenden URL's (z.B. <http://www.esra.com/aopweb>) im Eingabefeld des verwendeten Web-Browsers (IE-Explorer, Navigator oÄ.).

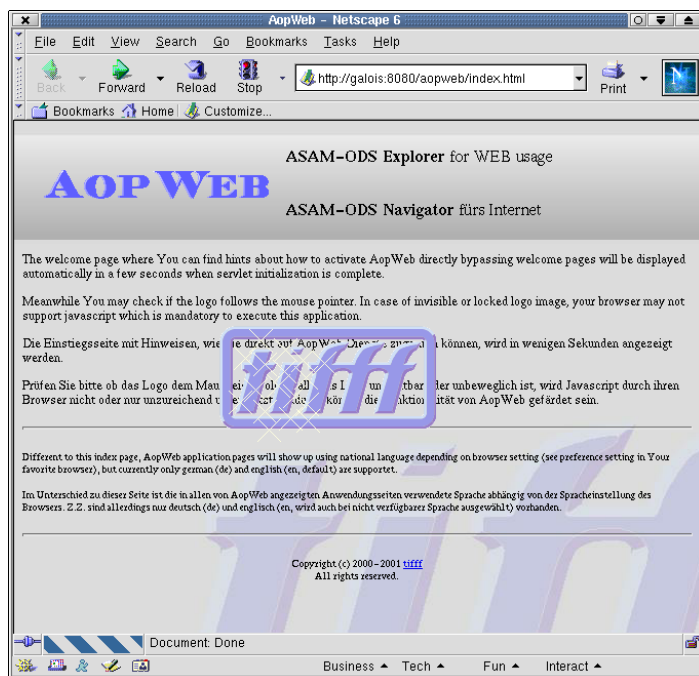


Abbildung 2.1: Begrüßungsseite

wird angezeigt, falls die Anwahl, wie hier angegeben, über den URL der Wurzel erfolgt<sup>1</sup>. Der zu verwendende URL hängt auch davon ab, die das Servlet installiert wurde. Falls z.B. Zonen eingerichtet wurden, kann der Pfad länger sein, oder falls nicht Port 80 (Standard bei http) verwendet wird, kann die Angabe einer Portnummer. z.B. 8080 notwendig sein.

## 2.2 Authentifizierung

Da der Zugriff auf AopWeb (nicht zu verwechseln mit dem Zugriff auf ASAM-ODS - Daten) kontrolliert wird, folgt, abhängig von der gewählten Konfiguration, ein Dialogfenster des Browsers (siehe Abbildung 2.2)

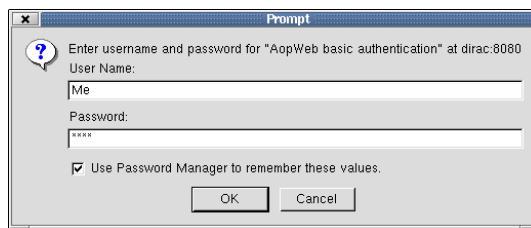


Abbildung 2.2: Authentifizierung (Browser)

oder als Formular des Servers (siehe Abbildung 2.3)

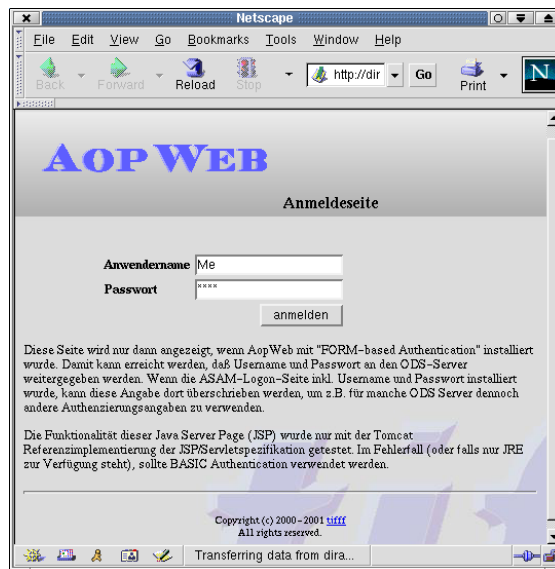


Abbildung 2.3: Authentifizierung (Formular)

Alle folgenden Seiten des Datenbrowsers werden zur Laufzeit erzeugt. Die Spracheinstellung der Benutzeroberfläche ergibt sich automatisch aus der Sprachein-

<sup>1</sup>vordefinierte ODS-Dienste können auch direkt angewählt

Rolle	erlaubt
lookup	Verwendung von AopWeb als Browser (Minimalrecht)
export	Lookup und Exportieren von Daten in XML
launch	Lookup und Aktivieren von ladbaren (JNLP) Diensten (z.B. Editor)
execute	Lookup und Aktivieren von registrierten serverseitigen Anwendungen
register	Lookup und Registrieren von ODS Datendiensten
modify	Lookup und Verwendung des Instanzeditors
admin	Alle Rechte zusammen (Maximalrecht)

**Tabelle 2.1:** Rechte

stellung im Browser. Das Seitenlayout kann unterschiedlich sein, da lizenzierte Versionen von AopWeb einen vom Kunden wählbaren Hintergrund haben und den Text (“Lizenziert für”) enthalten.

### 2.2.1 Authorisierung

Die Authentizierung ist notwendig, um dem Anwender Rechte zuordnen zu können, z.B. ob der Anwender neue Dienste bekanntmachen darf oder ob er Datenanwendungen, z.B. einen Dateneditor, aktivieren darf. Diese Rechte sind unabhängig von Rechten, die vom ASAM-ODS Server auf Datenelemente vergeben werden und beziehen sich auf alle ODS-Dienste, die der User in dieser Rolle verwenden darf. D.h. hier werden AopWeb-Aktivitäten zugelassen oder verhindert, unabhängig von den durch die Aktivität betroffenen ODS-Daten. Die Vergabe der Rechte erfolgt dadurch, daß Anwendern ein oder mehrere Rollen zugeordnet werden (siehe Abschnitt 5.1.3). Die Rollen entsprechen Zusammenfassungen von Rechten. Hinweis: Der Anwender hat z.Z. keine Möglichkeit die Rolle, falls im mehrere zugeordnet wurden, auszuwählen. Er hat alle sich aus diesen Rollen ergebenden Rechte. D.h. wenn einem Anwender alles außer `execute` erlaubt sein soll, müssen ihm die Rollen `export`, `launch`, `register` und `modify` zugeordnet werden.

## 2.2.2 Einstellungen

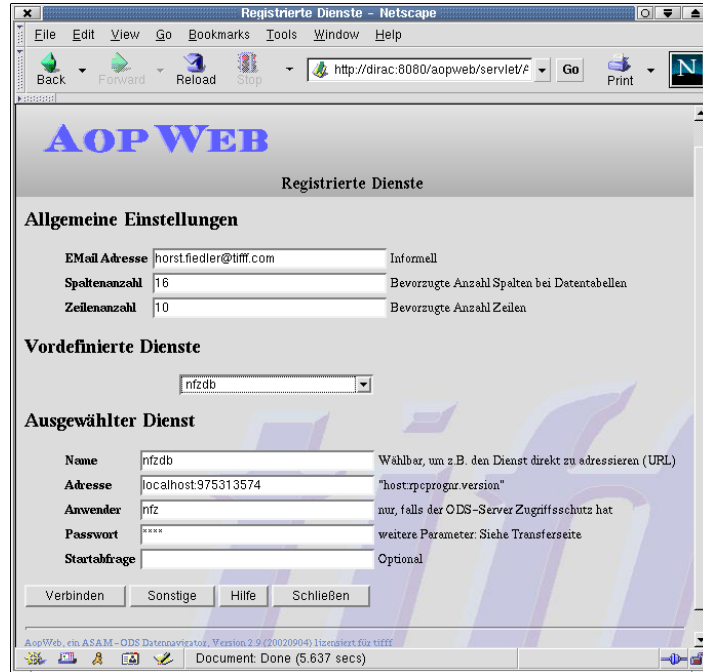


Abbildung 2.4: ASAM-ODS Anmeldeseite

### Allgemeines

Die Angabe einer Anzahl Zeilen/Spalten gibt eine Präferenz betreffend Tabellensichten an, solange keine Sichten festgelegt werden. Die Sichten auf Applikationselemente lassen sich auch über Sitzungen hinweg erhalten, die Anzeige von Teilmatrizen erfolgt zuerst immer entsprechend dieser Angabe.

Die EMail-Adresse sollte, insbesondere wenn ihm kein Administrator-Privileg zusteht, auf den Anwender zeigen, der für diesen User Administrationsrechte hat, z.B. `odsadmin@mycompany.com`.

### ASAM-ODS Dienstauswahl

Die Angaben von User und Passwort (siehe Abbildung 2.4) beziehen sich auf die Zugriffsberechtigung auf ASAM-ODS Daten.

Das Format einer ASAM-ODS (RPC) Adresse ist:

```
<Rechnername>:<RPC-Programmnummer>[.<Protokollversion>]
```

z.B.

```
testbedserver:975313576.3
```

Der Rechner kann sowohl mit seinem Namen als auch seiner IP Nummer angegeben werden. Wenn die Protokollversion nicht explizit genannt ist, wird

3 (für die aktuelle AOP-V3) verwendet. Die Unterstützung für Version 2 von AopWeb erlaubt nicht das Schreiben<sup>2</sup>.

Die vorgegebene Einstellung sowie weitere Felder der Einwahlsseite können durch Änderung des AopWeb Deployment-Deskriptors angepaßt werden. Falls Web-Zugriffskontrolle und ODS-Zugriffsschutz eingeschaltet sind, kann die Login-Information (Username/Passwort) auch übernommen werden und muß hier nicht erneut eingegeben werden (siehe see Appendix A.1).

ACHTUNG: Santorin 3.2 Server mit aktivierter Security benötigen noch einen Parameter PROJECT. Die aktuelle Version unterstützt die Eingabe zusätzlicher Parameter leider nicht mehr. Als Umgehungslösung gehen Sie wie folgt vor: Versuchen Sie einen Logon nur unter Angabe von Name (frei wählbar), RPC-Adresse und User/Passwort. Wenn eine Fehlermeldung erscheint, erzeugen Sie eine XML-Datei

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE nvlst SYSTEM "http://localhost:8080/aopweb/nv.dtd">
<!-- Used for AVL's Santorin Security system -->
<nvlst>
  <nv name="PROJECT">myproject</nv>
</nvlst>
```

wobei `myproject` durch einen gültigen Wert zu ersetzen ist und kopieren diese Datei nach

wobei `TOMCATDIR` das Verzeichnis ist, in dem tomcat installiert wurde, `HTTPUSERNAME` der Ihnen vom AOPWebadministrator zugewiesene Name, den Sie auf der AopWeb-Authorisierungsseite eingegeben haben ist, und `ODSSERVICENAME` der von Ihnen auf der ODS-Logonseite (ober der RPC-Nummer) eingegebene Bezeichner für das ASAM-ODS-Umfeld ist, danach nochmals versuchen.

Falls das klappen sollte: Das muß natürlich nur einmal gemacht werden. Falls sie mehrere unterschiedliche PROJECT-Namen benötigen: Einfach mehrere Umfeldbezeichner verwenden, die alle auf denselben Dienst zeigen.

Falls der gesuchte Dienst bereits mit Name registriert ist, kann die Anmeldeseite auch übersprungen werden, d.h. der registriert Name kann auch im URL angegeben werden:

```
<servletURL>/logon?service=<servicename>
```

z.B.

```
http://www.esra.com/servlet/AopWeb/logon?service=myODS
```

Das Servlet antwortet automatisch mit der ersten Datenseite des ASAM-ODS Dienstes mit der gegebenen RPC Nummer. Der Servlet URL kann auch unterschiedlich sein, da er u.A. auch von der Installation der Servlet Engine abhängt. Wird z.B. Tomcat auf dem Rechner "esra" mit der Standardkonfiguration installiert, so ergibt sich folgender Servlet URL

```
http://www.esra.com/aopweb/servlet/AopWeb
```

Es können sämtliche Parameter, die in der Registrierseite angebar sind, auch im URL angegeben werden, z.B. mit

<sup>2</sup>Version 2 wird meistens vom AVL TRR-Server verwendet, der ebenfalls keine Schreiboperationen erlaubt

<http://www.esra.com/servlet/AopWeb/logon?service=myODS&QUERY=Versuche%20Typ=DL>

kann sofort das Applikationselement Versuche nach den Zeilen gefragt werden, die im Attribut "Typ" den Wert "DL" beinhalten.

Aus Sicherheitsgründen sollte diese Art der Anmeldung nur für nicht passwortgeschützte ASAM-ODS Dienste verwendet werden, da das Eintragen von Klartextpasswörtern in HTML Seiten (als Teil des URL's) ungünstig ist. Andererseits erlaubt dieses kompakte Format das Ablegen des URLs als Bookmark oder Favoriten !!! und das Eintragen in andere statische HTML Seiten.

Es ergibt sich allerdings eine Beschränkung aus der Verknüpfung von ASAM-ODS und HTTP (Web) Session: Um auf ASAM-ODS zuzugreifen wird eine HTTP-Session (Der Besitzer der Session wird durch ein Cookie identifiziert) eröffnet, die die ASAM-ODS-Session enthält (1:1). Dadurch kann innerhalb einer HTTP-Session zu einem bestimmten Server nur ein ASAM-ODS Dienst verwendet werden. Bei erneuter Anmeldung geht die vorherige ASAM-ODS Session verloren. Um Datenaustausch zwischen 2 AODS Diensten, die von nur einem Rechner angeboten werden, zu ermöglichen muß das Sessionmanagement verbessert werden damit mehrere AODS Sessions in einer HTTP Session verwaltet werden können.

## 2.3 Applikationselementseite

Durch das Anmelden wird eine HTTP-Session, die eine AODS-Session enthält, erzeugt. Die HTTP-Session kann zum Zwischenspeichern von Daten verwendet werden, wie z.B. um für eine Tabellenansicht ausgewählte Daten an ein Graphik oder Exportierprogramm weiterzureichen.

Üblicherweise wird das erste Element aus dem Applikationselementindex von ASAM-ODS als erstes angezeigt. Die Seite beinhaltet Titel (Name des Applikationselements), Datentabelle und eine Navigationsbox, in der Angaben zur nächsten Auswahl gemacht werden können.

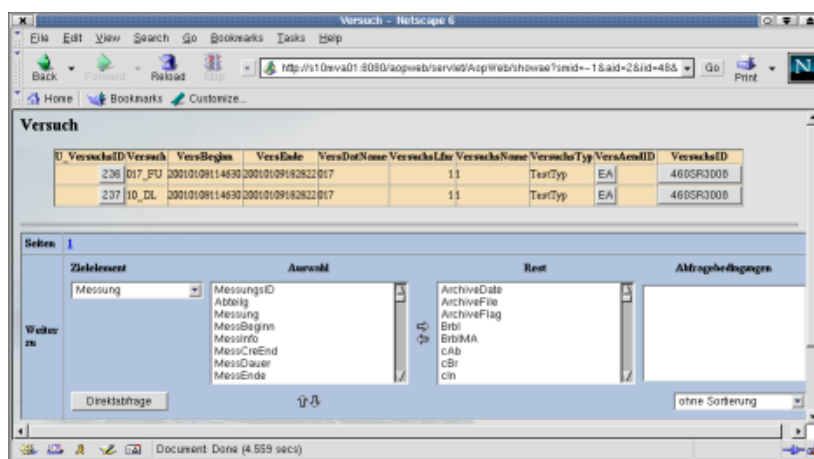


Abbildung 2.5: Applikationselementseite

Die Daten vom ASAM-Server werden im Servlet zwischengespeichert und

falls mehr Datensätze abgefragt werden als die Servletparameter<sup>3</sup> pro Seite erlauben, so wird nur ein Teil der Daten auf der ersten Seite, der Rest auf Fortsetzungsseiten angezeigt, die als Links in der ersten Seite auftreten.

### 2.3.1 Navigation

Meistens werden die Daten nicht direkt abgefragt, sondern der Anwender navigiert in den Daten, indem er bei einem Applikationselement mit dem Zielelementmenü festlegt, welche Daten nach dem nächsten Zugriff angezeigt werden sollen, und aktiviert mit einem Click der linken Maustaste auf einen Knopf in der ID-Spalte der Datentabelle den Zugriff.

Die nächste Seite enthält dann jene Instanzen des Zielapplikationselements, die in Beziehung zum aktivierten Element stehen. Zusätzliche Bedingungen für die auszuwählenden Instanzen können als *Abfrage Text* im Textfeld auf der rechten Seite eingegeben werden. Momentan ist das Abfragefeld ein normales Textfeld, der Benutzer muß die *Name-Operator-Wert* Tripel selbst eingeben oder kann die in den nebenliegenden Listen selektierten Attributnamen durch einen SHIFT-Klick mit der rechten Maustaste in das Abfragefeld einfügen. Die Namen müssen gültige Namen von Attributen im Zielapplikationselement sein.

Die *Direktabfrage* liefert unabhängig von Beziehung alle Instanzen.

Ev. vorhandene weitere Knöpfe in der Datentabelle repräsentieren direkte Referenzen (Fremdschlüssel): Bei Aktivierung wird als Antwort die Instanz angezeigt, die als Text im Knopf genannt ist, ohne andere Abfragebedingungen zu beachten.

**Hinweis zu Navigation** Falls ein Datenmodell nichtzusammenhängende Bereiche von Applikationselementen enthält, kann man durch Navigation nicht alle Elemente erreichen. In diesem Fall kann der das Element repräsentierende URL in der URL-Zeile des Browsers explizit angegeben werden, z.B.

```
http://myhost:8080/aopweb/servlet/AopWeb/showae?appelem=unreachable
```

wobei `unreachable` der Name des anders nicht erreichbaren Elements ist.

---

<sup>3</sup>siehe web.xml für Servletparameter

### 2.3.2 Binäre Objekte

BLOB-wertige Applikationselementattribute (Bilder, Tonspuren, .doc-Dateien usw.) können angezeigt werden, wenn der Inhalt des dem Blob zugeordneten Textfeldes den MIME-Typ enthält und für diesen Typ im Browser ein entsprechender Viewer registriert ist.

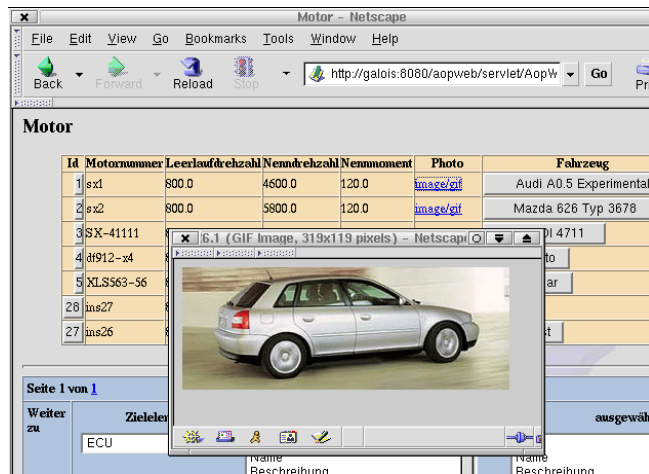


Abbildung 2.6: Embedded Blobs

Anderenfalls kann auf das Browser-Menü "Sichern als ...." zurückgegriffen werden und der Blob auf der lokalen Festplatte gespeichert werden.

### 2.3.3 Zusatzfunktionen

Falls bei der Auswahl einer Instanz (Click auf ID-Knopf) eine Zusatztaste gedrückt wird, können, abhängig von den Rechten des Anwenders, weitere Funktionalitäten erlaubt sein:

- Control-Click: Editiermöglichkeit der Instanz. Achtung: Diese Funktion unterliegt der Authorisierung durch den Administrator, d.h. i.a. ist sie für die meisten Anwender gesperrt.
- Shift-Click: Anzeige registrierter Zusatzanwendungen Siehe 4 betreffend Registrierung.

### 2.3.4 Auswahl von Sichten

Die Anordnung der Attribute kann im Auswahlfenster geändert werden, z.B. können Attribute, die von geringem Interesse sind entfernt werden. Empfehlenswert ist es, die Attribute, die der Navigation dienen, nicht zu entfernen.

Weiters können Bedingungen im Abfragebedingungsfenster formuliert werden. Die Syntax einer Bedingung ist:

Attributenname Vergleichsoperator Wert

Der dem Vergleichsoperator (=, <=, >=, !=, <, >) <sup>4</sup> vorangestellte Text wird als Applikationsattributname verwendet, der Text nach dem Operator bis zum Zeilenende wird als Vergleichswert interpretiert. Vor- und nachlaufende Leerzeichen werden ignoriert.

Das Auswahlmenü "Sortieren" ermöglicht, daß die darauffolgende Abfrage die Zeilen sortiert zurückliefert, wobei stets nach der ersten Datenspalte (die nach der ID-Spalte) sortiert wird, d.h. die zu sortierende Spalte muß an den Anfang geschoben werden.

Wenn ein explizites Logout - verwendet wird, werden die Sichteinstellungen für den User im Server gespeichert und sind beim nächsten Besuch auf dieser Site wieder verfügbar.

**Achtung:** Bei Verwendung der "Zurück-Knopf" und "Vorwärts-Knöpfe" im Browser werden keine neuen Abfragen durchgeführt. D.h. die Einstellungen entsprechen dann oft nicht den Erwartungen. Die Verwendung dieser Art der Navigation ist nur dann zu empfehlen, wenn Sie etwas mit Browser und Proxy-Cache und "Reload"-Verhalten vertraut sind.

---

<sup>4</sup>LIKE not yet

## 2.4 Meßdaten

### 2.4.1 Tabellenansicht

Darstellung in Tabellenform:

Messung 80 Teilmatrix 0

PntNr	PNTNR	\$KEY	ALPHA	AL_STELL	BEFF	BEFFW	BRBLNR	DATUM	LAUFZAUF	MB	MEFFW
1.1	1.0	252.47	1	100.0	199.8887	1	21.0	13-FEB-01	252.4715	548.1	2493.9524
2.1	1.0	252.47	1	100.0	199.8887	1	21.0	13-FEB-01	252.4715	548.1	2493.9524
1.2	2.0	255.59	1	100.0	197.12004	1	21.0	13-FEB-01	255.5945	544.6	2512.707
2.2	2.0	255.59	1	100.0	197.12004	1	21.0	13-FEB-01	255.5945	544.6	2512.707
1.3	3.0	259.12	1	100.0	194.45857	1	21.0	13-FEB-01	259.1215	542.5	2527.6284
2.3	3.0	259.12	1	100.0	194.45857	1	21.0	13-FEB-01	259.1215	542.5	2527.6284
1.4	4.0	262.34	1	100.0	193.79825	1	21.0	14-FEB-01	262.3448	539.9	2533.8072
2.4	4.0	262.34	1	100.0	193.79825	1	21.0	14-FEB-01	262.3448	539.9	2533.8072
1.5	5.0	265.37	1	100.0	194.04426	1	21.0	14-FEB-01	265.3715	542.0	2531.2578
2.5	5.0	265.37	1	100.0	194.04426	1	21.0	14-FEB-01	265.3715	542.0	2531.2578

Anzeige Seite 1 von 1224

Auswahl Teilmatrix: Leichte Spalten: Ausgewählt: Formeln:

\$KEY  
ALPHA  
AL\_STELL  
BEFF  
BEFFW  
BRBLNR  
DATUM  
F

PNTNR  
\$KEY  
ALPHA  
AL\_STELL  
BEFF  
BEFFW  
BRBLNR  
DATUM

F  
P\_MEAN  
P\_MEAN

Netzkarte von: 1 Anzahl: 40 ohne Sortierung Laden

Hilfe Auswertung Zurück

Transferring data from gskis...

Abbildung 2.7: Ein kleiner Bereich aus einer Submatrix

AopWeb unterscheidet zwischen ausgewählten und angezeigten Daten. Es ist möglich, daß eine große Menge an Daten ausgewählt wurde, aber da nur eine beschränkte Menge pro Seite sinnvoll dargestellt werden kann, sich der Großteil auf Fortsetzungsseiten befindet. Wird dann eine graphische Darstellung der Datenmenge durchgeführt werden **alle** selektierten Daten dem Applet zur Verfügung gestellt.

### 2.4.2 Graphische Präsentation

Die Visualisierungsmöglichkeiten im Browser sind beschränkt. Im Auswertungs-menü sind nur x-y Plot (Bild 2.4.2) und Isolinien-darstellung angeführt, die als Applet (Pluggin im Browser) implementiert sind.

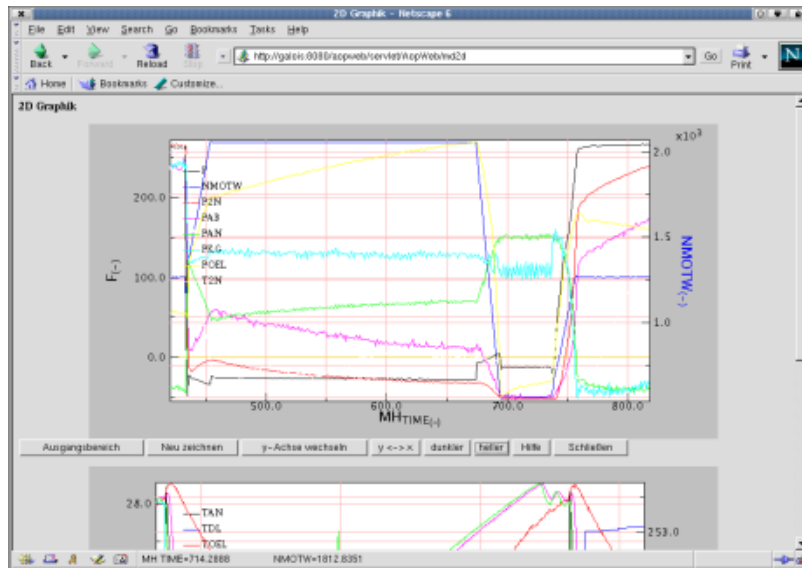


Abbildung 2.8: Graphik statt Wertetabelle

## 2.4.3 Messdateneditor

The screenshot shows the 'Messdateneditor' application window. The title bar reads 'AasEditor 6.3'. The main window contains a table with the following columns: 'Attribut', 'Wert', 'RowID', 'TI\_PSTART', 'PHASE\_NR', 'T\_SO', 'LAM', 'NO\_V\_KAT', 'GO\_V\_KAT', 'HC\_V\_KAT', and 'Q2'. The table contains data for rows 302 through 329. The 'T\_SO' column is highlighted in red for all rows, indicating a value of 200.0. On the left side, there is a list of attributes with their corresponding values, such as 'MESSARTID' with value 1, 'MESSART' with value 'KL 180', and 'SY\_MENR' with value 'S05E53'. The application interface includes a menu bar with 'Datei', 'Bearbeiten', and 'Werkzeuge', and a toolbar with various icons. The status bar at the bottom indicates 'Daten wurden geladen' and shows page navigation buttons 'Blatt 1' and 'Blatt 2'.

Attribut	Wert	RowID	TI_PSTART	PHASE_NR	T_SO	LAM	NO_V_KAT	GO_V_KAT	HC_V_KAT	Q2
NTWK		302	201.13	12.0	200.0	0.9886	1021.0	1005.0	479.0	611
LLK		303	201.35	12.0	200.0	0.98842	1020.0	10062.0	482.0	612
ME_TEXT2		304	201.57	12.0	200.0	0.98831	1021.0	10062.0	479.0	606
MessartID	1	305	201.79	12.0	200.0	0.9881	1021.0	1005.0	479.0	605
MESSART	KL 180	306	202.01	12.0	200.0	0.98786	1021.0	1005.0	482.0	601
VersuchID	1	307	202.23	12.0	200.0	0.98768	1021.0	1005.0	479.0	598
SY_MENR	S05E53	308	202.56	12.0	200.0	0.98754	1021.0	1005.0	479.0	596
LoeschFlag	0	309	202.78	12.0	200.0	0.98728	1021.0	1005.0	479.0	587
SY_MEDAT	24.09.2001	310	203.0	12.0	200.0	0.98706	1021.0	10062.0	479.0	585
SY_MESSPR	Lkennl	311	203.22	12.0	200.0	0.98695	1021.0	10062.0	479.0	581
SY_MEZET	11.25	312	203.44	12.0	200.0	0.98668	1021.0	10062.0	479.0	571
SY_OPER		313	203.66	12.0	200.0	0.9864	1021.0	1005.0	482.0	569
MOTID		314	203.88	12.0	200.0	0.98622	1021.0	10062.0	482.0	567
PRSTNR	SOND	315	204.1	12.0	200.0	0.98602	1021.0	10062.0	482.0	562
VE_BEM1		316	204.32	12.0	200.0	0.98583	1021.0	10062.0	479.0	559
VE_BEM2		317	204.54	12.0	200.0	0.98562	1021.0	1005.0	479.0	554
VE_CREATE		318	204.76	12.0	200.0	0.98543	1021.0	10062.0	482.0	553
MOTTYP		319	205.03	12.0	200.0	0.98524	1021.0	10062.0	482.0	553
SG_MENU		320	205.25	12.0	200.0	0.98511	1021.0	1005.0	482.0	545
		321	205.53	12.0	200.0	0.98487	1020.0	10062.0	479.0	541
MessGroId	25	322	205.75	12.0	200.0	0.9847	1021.0	10073.0	479.0	536
MessGrosse	T_SO	323	205.97	12.0	200.0	0.9844	1020.0	10062.0	482.0	531
Datentyp	7	324	206.19	12.0	200.0	0.98419	1021.0	10062.0	479.0	529
MessStaNr	0	325	206.41	12.0	200.0	0.98398	1021.0	10062.0	482.0	525
Abstrakte	NaN	326	206.63	12.0	200.0	0.98383	1021.0	10062.0	479.0	521
LoeschFlag	0	327	206.85	12.0	200.0	0.98368	1020.0	10062.0	479.0	518
MessGroDim	0	328	207.07	12.0	200.0	0.98345	1021.0	10062.0	479.0	513
MessGroExi...	0	329	207.29	12.0	200.0	0.98329	1021.0	10062.0	479.0	510

Abbildung 2.9: Dateneditor

Wenn die um SOAP erweiterte Version verwendet wird und entsprechende Desktopanwendungen in das Applikationsverzeichnis von AopWeb kopiert wurden (deployed), werden kontextabhängig weitere Werkzeuge angeboten, z.B. ein Messdateneditor (Bild 2.4.3). Auch dieser wird von AopWeb aktiviert (mittels MIME-Typ "application/x-java-jnlp-file"), exekutiert aber am Desktop des Anwenders und ist mittels SOAP-Protokoll mit AopWeb verbunden.

## Kapitel 3

# Standard-Erweiterungen

Die Parametrierung von optionalen Funktionspunkten, z.B. Formelberechnungen oder Ausgabe in XML oder als EXCEL-Spreadsheet erfolgt auf der Seiten "Einstellungen und Datentransfer".

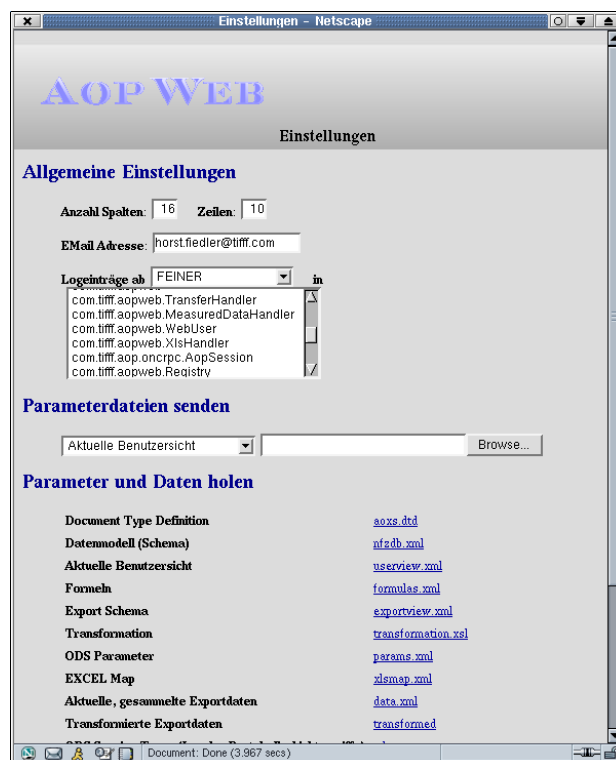


Abbildung 3.1: Einstellungen

Diese Seite kann mittels "Transfer"-Knopf (auf allen Applikationselementseiten vorhanden) angezeigt werden und enthält je nach vorhandenen Parametrierungen und auch abhängig von der Rolle des AopWeb-Anwenders (Adminis-

trator, ...) mehr oder weniger Elemente.

### 3.1 Tabellenformat

Mit Zeilen/Spalten kann die Größe von Tabellen eingestellt werden. Die Angabe der Anzahl Spalten wird allerdings nur dann verwendet, wenn noch keine Sicht festgelegt wurde.

### 3.2 Anwenderadresse

DaAopWeb keine Ausgaben per Mail versendet, ist diese optionale Angabe z.Z. funktionslos.

### 3.3 Logdateien

Dieser Abschnitt wird nur angezeigt, falls der AopWeb-Anwender in Administrator-Rolle ist, d.h. auch das Recht hat, die Logging-Einstellungen des Servers zu ändern (Abbildung 3. Da Logging vor allem dazu dient, zusätzliche Informationen im Fall von Fehlverhalten (z.B. "AopWeb kann sich nicht zu einem ODS-Dienst verbinden") zu erhalten, wird AopWeb zumeist im voreingestellten Modus (Fehler und Warnhinweise werden geloggt) betrieben und beim Reproduktionsversuch von Fehlern wird auf "FEIN" betreffend tiff-Klassen geschaltet. Falls das Logging der HTTP-Schnittstelle (com.tiff.aopweb.AopWeb) nicht enthalten sein soll, kann dieses auch wieder auf einen kleineren Level geschaltet werden.

Die Log-Dateien finden sich unter aopweb\*.log im Arbeitsverzeichnis der Servletengine (z.B. dort, wo Tomcat gestartet wurde). Diese und andere Einstellungen können in der Datei webapps/aopweb/WEB-INF/logconfig.properties mit Texteditor geändert werden.

### 3.4 Datentransfer/XML

AopWeb verwendet für die Parameterierung von Sichten und für Datenexport die XML-Infrastruktur von AOXS. Die Parametrierung von ODS-Parametersätzen, Formeln und EXCEL-Mappings erfolgt mit eigenen DTD's.

#### 3.4.1 Datenmodell

Mit "Datenmodell holen" erfolgt ein Download des Datenmodells des Servers als XML Dokument. Hinweis: Dies ist nicht in XML-Schmalanguage formuliert, sondern entsprechend AOXS-DTD.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE AOXS SYSTEM "http://galois:8080/aopweb/aoxs.dtd">
<AOXS>
  <AE ID="A1" NAME="Umfeld" TYP="ENV">
    <AA ID="A1.0" NAME="UmfeldId" TYP="ID" DTYP="INT"/>
    <AA ID="A1.1" NAME="Umfeldname" TYP="NAME" DTYP="STRING"/>
  </AE>
</AOXS>
```

```

    <AA ID="A1.2" NAME="Umfeldbezeichnung" DTYP="STRING"/>
    <AS ID="S1.6" NAME="Versuchsreihe(Umfeld)" REF="R5.0"/>
    ...
    <AS ID="S1.19" NAME="Fahrzeug(Umfeld)" REF="R21.0"/>
  </AE>
  <AE ID= ....>
  ...
    <AA ID="A7.3" NAME="Messungsbeginn" DTYP="DATE"/>
    <AA ID="A7.4" NAME="Basis" DTYP="INT"/>
    <AS ID="S7.9" NAME="Messungsgroessen(Messung)" TYP="MEAID" REF="R8.0"/>
    <AS ID="S7.35" NAME="Profil(Profil)" REF="R43.1"/>
    <AS ID="S7.13" NAME="Dauerlauf(Messungen)" TYP="BaseClass" REF="R10.0"/>
    <AS ID="S7.12" NAME="Vollast(Messungen)" TYP="BaseClass" REF="R9.0"/>
    <AR ID="R7.0" NAME="Versuchsversion(Versuch)" REF="S6.8"/>
  </AE>
</AOXS>

```

Dieses Datenmodell kann editiert werden (Texteditor, XML-Editor) um eine Export-Sicht zu definieren (Teilmenge aller Elemente und Attribute), die am Server hinterlegt werden kann.

### 3.4.2 Exportierte Daten holen

Auf der “Anwendungen”-Seite (siehe Abbildung 4.3) kann dann für die ausgewählte Instanz ein “Export” gestartet werden, wodurch dieses Schema mit Instanzelementen gefüllt wird. Dies kann wiederholt (mit unterschiedlichen Instanzen) erfolgen. Die angesammelten Daten können dann abgeholt werden.

Alle mit AopWeb exportierten Daten können mit AOXS weiterverarbeitet werden, da immer dasselbe Schema (als DTD) verwendet wird. Dieses System zur dynamischen Erstellung von XML-Elementen aus Daten der Aop Protokollschicht wird in dem Dokument *AoxsG.pdf* behandelt.

Die Funktionalität von AOXS zur Durchführung von Datentransfers ist sowohl mittels SOAP (Beispielkomponente: Transfermonitor) oder mittels ausführbarer (exec Methode) Skripts an AopWeb anbindbar (siehe Abschnitt 4).

### 3.4.3 XSL-Transformation

Wenn ein Stylesheet am Server hinterlegt wird (upload), können die exportierten Daten auch “transformiert” werden.

### 3.5 Microsoft EXCEL

AopWeb kann direkt EXCEL (97-2002) Ausgabe erzeugen <sup>1</sup>. Voraussetzung: Es muß eine Vorlage zum Server hochgeladen worden sein (Transfer-Seite). Diese Vorlage bleibt über Sessions hinweg, bis eine neue Vorlage geladen wird, erhalten. Wenn eine Vorlage vorhanden ist, wird auf der "Auswerte"-Seite nach den Teilmatrizen als zusätzliche Option "Excel" angeboten. Falls man diese Option wählt, wird das gerade aktuelle Workbook um die in der Vorlage angegebenen Sheets erweitert, d.h. in ein Excel-Workbook können Daten aus unterschiedlichen Datenbereichen einer oder mehrerer ODS-Datenbanken geladen werden.

The screenshot shows a Microsoft Excel Viewer window titled 'aopweb.xls'. The spreadsheet contains the following data:

Teilmatrix:	PNTNR	\$KEY	ALPHA	BEFF	BRBLN	DATUM	F	LAUFZAL
1	1	252.47	100.0	199.8887	21.0	13-FEB-01	261.166	252.4715
12	2	252.51	0.0	-8.557207	21.0	13-FEB-01	-28.45852	252.51149
13	3	252.52	0.0	-135.3659	21.0	13-FEB-01	-12.63182	252.5215
14	4	255.59	100.0	197.12004	21.0	13-FEB-01	263.12997	255.5945
15	5	256.03	0.0	-7.521408	21.0	13-FEB-01	-27.55199	256.0345
16	6	256.04	0.0	-99.53538	21.0	13-FEB-01	-12.55174	256.0445
17	7	259.12	100.0	194.45857	21.0	13-FEB-01	264.69254	259.1215
18	8	259.16	0.0	-8.440406	21.0	13-FEB-01	-27.46444	259.1615
19	9	259.17	0.0	-118.7454121	21.0	13-FEB-01	-12.27549	259.1715
20	10	262.24	100.0	193.78825	21.0	14-FEB-01	265.31863	262.24448

Abbildung 3.2: Excel

#### 3.5.1 Vorlageformat

Die Vorlage ist eine XML-Datei, die mindestens ein Sheet definiert, siehe Beispiel 3.5.1.

Die DTD zur Vorlage in von AopWeb ladbar und definiert folgende Elemente und Attribute:

- map  
ist das Wurzelement, das alle weiteren Elemente enthält,
- style  
definiert das Layout von Zellen,
  - id ist ein frei wählbarer Name, unter dem diese Definition vom cell-Element referenziert werden kann,
  - color legt die Farbe der Zelle fest,
  - rotate erlaubt, die Zelle zu drehen,
  - align erlaubt, die Zelle auszurichten,

<sup>1</sup>Für UNIX: OpenOffice kennt dieses Format selbstverständlich auch

`borderXXX` mit `XXX = top, bottom, left` bzw. `right` ermöglicht die Gestaltung des Randes einer Zelle wobei Linienart und Farbe einfach hintereinandergeschrieben werden (z.B. “`ref thick`”),

`dataformat` erlaubt Formatfestlegungen (z.B. Währung, andere Farbe für negative Werte usw.)

`font` legt die Schriftart fest (siehe unten).

- `font`  
legt die Schriftart und zugehörige Attribute fest.

`id` ist ein frei wählbarer Name, unter dem diese Definition vom style-Element referenziert werden kann,

`name` ist der Name eines am Desktop verfügbaren Fonts,,

`size` gibt die Größe in Punkten an,

`color` ist ein Farbname

`weight` erlaubt nur `normal` (default) und `bold`.

- `sheet`  
definiert den Beginn eines Arbeitsblattes. Ein Sheet enthält ausschließlich row-Elemente und hat nur ein Attribut:

`title` ist der Name des Arbeitsblattes (für die Tab's).

- `row`  
definiert den Beginn einer Zeile,

`height` gibt die Höhe der Zeile an (in 1/20 Punkten),

`span` erlaubt die Wiederholung der Zeilendefinition (Default:1).

- `cell`  
definiert den Inhalt einer Zelle und hat auch ein paar Attribute

`type` legt den Zellentyp fest, “`element`” (default) oder “`matrix`”. Element-Zellen enthalten

`ref` legt eine Applikationsattribut (Name) als Datenquelle fest. Es wird der Werte des erste Attributs dieses Namens aus der Messungsinstanz bzw. aus einer von der Messung aus direkt über n:1 - Beziehung erreichbaren Instanz eingesetzt. Nur bei `type=element` verwendbar.

`width` legt die Breite der Spalte, in der diese Zelle vorkommt, fest. Achtung: Falls mehrfach Breiten zugeordnet werden, gilt die letzte Festlegung. Falls der `type` “`matrix`” ist, gilt die Breitenfestlegung nur für die “Zeilenummerierung”

`style` legt den zu verwendenden Stil für diese Zelle fest. Nur für “`element`”-Zellen.

`styles` legt eine Liste zu verwendender Stile für “`matrix`” Elemente fest. Der erste definiert den Stil der Kopfzeile (Messungsgrößennamen), der zweite den Stil für die Einheiten-Kopfzeile. Alle weiteren definieren den Stil von Datenspalten.

span erlaubt es, die Zellendefinition zu wiederholen (nur bei “element”-type, “matrix” zählt von selbst weiter).

Falls ein “ref”-Attribut vorhanden ist, wird versucht, das entsprechende ODS-Attribut zu ermitteln und es wird dessen Wert in die Arbeitsblattzelle gesetzt, anderenfalls der, sofern vorhanden, Inhalt (Content) des Cell-Elements selbst.

Die verfügbaren Farben entsprechen den in Excel definierten:

aqua, black, blue, blue\_grey, bright\_green, brown, dark\_blue, dark\_green, dark\_red, dark\_teal, dark\_yellow, gold, green, grey\_25\_percent, grey\_40\_percent, grey\_50\_percent, grey\_80\_percent, indigo, lavender, light\_blue, light\_green, light\_orange, light\_turquoise, light\_yellow, lime, olive\_green, orange, pale\_blue, pink, plum, red, rose, sea\_green, sky\_blue, tan, teal, turquoise, violet, white, yellow

Falls als Hintergrund für Zellen eine Farbe gewählt wird, wird automatisch das Muster “sparse dots” hinterlegt.

### 3.5.2 Durchführung

Einfach auf den “Durchführen”-Knopf in der “Auswerte”-Seite drücken. Die Beispiel-Vorlage 3.5.1 könnte dann (je nach vorhandenen Daten und getätigter Auswahl) ein Ergebnis entsprechend Abbildung 3.2 liefern.

### 3.5.3 Aktuelle Restriktionen

Der erste Versuch beinhaltet noch einige Restriktionen, die voraussichtlich in einem zukünftigen Release behoben sein werden:

#### Vorlage

- Keine Hintergrundmuster auswählbar
- Es sind nur einige Randstärken verfügbar: hair, medium, thick, double.
- Es gibt keine Formelfelder
- Feld-Alignments werden nicht erkannt
- Zellbreiten/Höhen: In 1/20 von pts

**Durchführung** Z.Z. wird ein Arbeitsbuch nur dann zurückgesetzt, wenn eine neue Anwendersitzung beginnt (IE neu gestartet, Webserver neu gestartet oder Sitzungs-Timeout). Die nächste Version bekommt eine “Reset”-Checkbox.

Da das Arbeitsbuch aber dem Anwender zugeordnet ist, bleibt es auch über unterschiedliche ODS-Sitzungen (auch unterschiedliche Umfelder) hinweg erhalten. Das hat den Vorteil, daß sogar Daten aus unterschiedlichen Umfeldern in ein EXCEL-Dokument gespeichert werden können. Die Vorlagen sind “Umfeld + Anwender” spezifisch. Z.Z. erfolgt aber keine automatische Vorlagenumschaltung, d.h. falls Daten aus unterschiedlichen Umfeldern gemischt werden sollen,

muß jeweils nach dem Öffnen des Umfelds die zu diesem Umfeld passende Vorlage hochgeladen werden <sup>2</sup>.

## 3.6 Formeln

Neben Meßwerten werden auch aus den Meßwerten abgeleitete Werte (Rechenwerte) benötigt. Dieser Formelinterpreter, der auch getrennt von AopWeb eingesetzt werden kann, hat neben der zeilenweisen Evaluierung von "virtuellen Kanälen" (skalare Formelevaluierung) auch die Möglichkeit von Vektorberechnungen, und kann auch, z.B. bei Bindung an ASAM-ODS, auch Applikationselementattribute miteinbeziehen. Falls der ODS-Server Formeln evaluieren kann, kann die Funktion dazu verwendet werden, diese Formelevaluierung anzustoßen.

Das Formelpaket fußt auf 2 Technologien:

- XML zur Speicherung von Formeln, d.h. durch XML sind Zeichensatz, Struktur und auch diverse Attribute zur Integration in Anwendungen, die mathematische Formeln evaluieren, definiert,
- ECMAScript, um die Anweisungen zu formulieren, die berechnet werden sollen.

Dieses Handbuch ist keine Einführung in XML oder ECMAScript. ECMAScript, oft auch als JavaScript <sup>3</sup> bezeichnet, ist die dominierende Skriptsprache bei der Erstellung von dynamischen HTML-Seiten. Anleitung betreffend Verwendung: Siehe entsprechende Literatur.

Formeln werden in XML-Dokumenten abgelegt, siehe Beispiel 3.6.1. In dieser Form können sie transferiert (an Anwendungen weitergegeben) werden. Bei Verwendung von AopWeb kann auf der Transfer-Seite eine lokale, Formeln enthaltende Datei ausgewählt und an den Server transferiert werden. Dort wird die Formeldatei von einem XML-Parser gelesen, kompiliert (inkludiert Syntaxprüfung) und, falls sowohl XML-Syntax als auch ECMA-Skript Syntax der Formeln in Ordnung sind, als "aktueller Formelsatz" hinterlegt. Für diese und nachfolgende Sessions dieses Anwenders, die denselben ODS-Server verwenden, werden diese Formeln bei der Anzeige von Meßdaten ausgewertet.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE fmlmap SYSTEM "http://localhost:8080/aopweb/fml.dtd">
<fmlmap>
  <var name="P" depends="NMOT,MEFFW" >NMOT*MEFFW*Math.PI/30000</var>
  <var name="P_MIN" dispose="summary">min(P_MIN, P)</var>
</fmlmap>
```

### Beispiel 3.6.1: Formelbeispiel

#### 3.6.1 Begriffe

Einige Begriffe benötigen etwas Erläuterungen:

<sup>2</sup>Vorlagen die nur matrix-Zellen enthalten, d.h. keine Applikationsattribute ansprechen, sind von dieser Restriktion nicht betroffen

<sup>3</sup>Die Ähnlichkeit von C# zu Java ist wesentlich ausgeprägter als die von Java zu JavaScript

- Formel bezeichnet sowohl einfache arithmetische Ausdrücke, z.B.  $n * md / const$ , Anweisungen, z.B.  $v = l * t$ , Gruppen/Blöcke von Anweisungen und auch Programmteile inkl. Kontrolllogik, z.B. `var a = newArray();`  
`for(var i = 0; i < ROWCNT; i++)`  
`a[i] = d * Math.pow(v[i], 2) + Math.random();`  
`return a;`  
 Formeln sind der "Inhalt" (Content) von `< var >` - XML-Elementen.
- Als Sonderfall (Server-Formeln) kann der Inhalt der Formel auch leer sein. Darartige Formeln sind ein Hinweis, daß der ODS-Server (mittels z.B. FORINT-Pluggin) die Formel evaluieren soll und das Ergebnis in die Antwort einbauen soll.
- Variable sind benannte Dinge. Auch das Ergebnis der Berechnung einer Formel ist eine Variable. Variable haben auch eine Sichtbarkeit zugeordnet (Scope). Lokale Variable (z.B.  $i$  und  $a$  in obigem Beispiel) sind nur innerhalb der Formel sichtbar, die Verwendung desselben Namens in einer anderen Formel bezieht sich auf eine andere Variable. Globale Variable (z.B.  $ROWCNT$  oder  $d$  in obigem Beispiel) bezeichnen auch bei Vorkommen in einer anderen Formel dieselbe Variable. Die durch das `name`-Attribut eines `< var >` XML-Elements definierten Variablen sind immer global.
- Paket bezeichnet Zusammenfassungen von eingebauten Funktionen, z.B. enthält das *Math* Paket (Teil des Standards) die gebräuchlichsten mathematischen Funktionen (Logarithmen, trigonometrische Funktionen usw.). Pakete können auch vom Anwender erstellt und zum Satz der verfügbaren Pakete hinzugefügt werden. Neben den Java-Standardpaketen ist ein Paket JsMath mit ein paar speziellen Methoden enthalten. Zur Erstellung von zusätzlichen Paketen (oder weiteren Methoden) ist Java-Programmierung nötig.
- Wert ist der aktuelle Inhalt einer Variablen. "undefined" ist ein zulässiger Wert und kann in Formeln auch zugewiesen bzw. abgefragt werden. Innerhalb von Formeln sind `undefined` und `NaN` (IEEE "Not a Number") unterschiedliche Werte.
- XREF bezeichnet "eXternal REference", das sind alle `< var >`-Elemente der XML-Datei, die keine Formeln definieren, d.h. deren "Inhalt" leer ist und deren Inhalt zur Laufzeit aus externen Datenquellen, z.B. ODS Daten, ermittelt werden muß.
- Locator sind Hilfsobjekte, um externe Variable zu binden (Map). Locator werden auf die zugrundeliegende Datenbasis abgestimmt. Z.Z. enthält das Formelpaket Locator-Klassen für Umgebungsvariable (wenn der Formelanwendung z.B. Variable in den Startargumenten der Anwendung mitgegeben werden), ASAM-ODS Applikationsattribute und ASAM-ODS Meßmatrixelemente. Locator sind optional, d.h. eine Anwendung kann das Formelobjekt nach den XREF's fragen, die erhaltene Tabelle mit verfügbaren Werten füllen und danach die Evaluierung anfordern.
- Attribute der XML-Elemente geben Hinweise, wie Formeln abzuarbeiten sind (mode) bzw. Hinweise, wie Ergebnisse von der umgebenden Anwendung zu behandeln sind.

### 3.6.2 DTD

Die festgelegte XML-Struktur von Formeldokumenten ist durch eine DTD (Document Type Definition) festgelegt. Die DTD ist sehr einfach:

```
<!ELEMENT fmlmap (var*)>
<!ATTLIST fmlmap mode (row|matrix) "row">

<!ELEMENT var (#PCDATA)>
<!ATTLIST var name ID #REQUIRED>
<!ATTLIST var dispose (default|summary|utility) "default">
<!ATTLIST var depends CDATA #IMPLIED>
```

#### Beispiel 3.6.2: DTD

d.h. Ein Formelsatz (`fmlmap`) besteht aus einer beliebigen Anzahl von Variablen, die einen Namen haben (müssen) und ein weiteres Attribut (`typ`) haben können. Der Inhalt eines Variablen-elements (XML-Content) ist eine Zeichenfolge (PCDATA, die Formel) oder leer.

Das **mode** - Attribut erlaubt es, die Bindung (Map) der vorkommenden Variablen zu beeinflussen. Das Attribut kann die Werte **row** (dieser Mode ist auch eingestellt, falls das Attribut nicht angegeben wird) und **matrix** annehmen.

- **row**  
bindet Variable, auch wenn diese extern als Arrays (Vektoren) vorliegen, an Skalare. Dies erlaubt zeilenweise Abarbeitung der Formeln, wenn Eingangsdaten als Matrizen vorliegen. D.h. bei der Formelevaluierung werden die Variablen für jede Zeile der Eingangsmatrix gesetzt und dann die Formeln evaluiert. I.a. wird das solange wiederholt, bis die Matrix abgearbeitet ist. Diese Vorgangsweise hat den Vorteil, das Formeln ohne Indexarithmetik formuliert werden können (siehe Beispiel 3.6.1).
- **matrix**  
bewirkt, daß die Variablen, falls sie als Vektoren (Matrixspalten) vorliegen, als Arrays übergeben werden und die Evaluierung nur einmal aufgerufen wird. Das Beispiel 3.6.3 bewirkt ähnliches wie Beispiel 3.6.1, arbeitet allerdings im "matrix"-Modus.

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE fmlmap SYSTEM "http://localhost:8080/aopweb/fml.dtd">
<fmlmap mode="matrix">
  <var name="P"><![CDATA[
    var a = new Array();
    for (var i = 0; i < ROWCNT; i++)
      a[i] = NMOT[i] * MEFFW[i] * Math.PI/30000;
    return a;
  ]]></var>

  <var name="P_MEAN" dispose="summary"><![CDATA[
    if (ROWCNT == 0)
      return undefined;
    var sum=P[0];
    for (var i = 1; i < ROWCNT; i++)
      sum += P[i];
    return sum/ROWCNT;
  ]]></var>
</fmlmap>

```

### Beispiel 3.6.3: Matrixmode

Werte des Attributs **dispose** sind nicht in der DTD, sondern im Formelpaket (Software) festgelegt<sup>4</sup>. Die zulässigen Werte können je nach Bindung an unterschiedliche Datenbasen variieren. Z.Z. (ODS-Bindung) sind folgende Werte für dieses Attribut in Verwendung:

- utility  
wird verwendet, wenn die umgebende Anwendung das Ergebnis nicht berücksichtigen soll. Wird z.B. für Zwischenergebnisse verwendet.
- summary  
wird verwendet, wenn die umgebende Anwendung das Ergebnis nicht als (zusätzliche) Spalte der Matrix anzeigen soll sondern als Summenergebnis der gesamten Berechnung deuten soll.

Das Attribut **depends** kann eine Liste von Variablenamen enthalten. Der Evaluator versucht vor der Durchführung, diese Variablen zu befriedigen. Falls der Ausdruck leer ist, wird automatisch der Name selbst in diese Liste eingetragen. Das kann dazu verwendet werden auf Spalten bzw. Attribute zuzugreifen, die in der Datenquelle (ODS-Server) nicht registriert sind, z.B. ODS-Serverformeln.

Hinweis: Da Formeln sehr oft das Zeichen  $<$  (Vergleichsoperator) enthalten, ist es empfehlenswert, die Formel in einen **CDATA**-Abschnitt zu packen (siehe Beispiel 3.6.3) um diese Zeichen nicht als *&lt;* schreiben zu müssen.

<sup>4</sup>Daher wird fehlerhafte Schreibweise vom XML-Parser nicht erkannt

### 3.6.3 Sprachumfang

Da die Formeln in ECMA-Skript (ISO/IEC 16262) formuliert werden, wird auf entsprechende Einführungen in JavaScript verwiesen. Spezifisch ist allerdings die Definition des Kontexts, in dem die Formeln ausgeführt werden. Ein einfacher Ausdruck  $N * MEF\text{FW}/9549, 3$  ist kein vollständiges Skript. Bevor der Formelintepreter die Formel kompiliert, wird der Ausdruck zu `functiongetP()returnN * MEF\text{FW}/9549, 3` ergänzt und dieser Ausdruck wird dann compiliert. Falls die Formel bereits ein oder mehrere Textstücke `return` enthält, wird kein `return` ergänzt.

Bei Evaluierung der Formeln werden die Funktionen in der Reihenfolge, in der sie im XML-Dokument angeführt sind, evaluiert.

Daraus ergeben sich Restriktionen:

- Sofern die Formel nicht als genau ein arithmetischer Ausdruck formuliert werden kann, muß ein "return" angegeben werden,
- Variablenamen müssen ECMA-Skript geeignet sein, d.h. sie dürfen keine Operatoren (z.B. -) enthalten.
- Verwendete globale Namen müssen eindeutig sein. Falls z.B. eine Variable `d` in mehreren ASAM-ODS Applikationselementen vorkommt, kann (z.Z.) keine Qualifizierung z.B. mittels Applikationselementnamen erfolgen.
- Die Funktionen werden jeweils einmal aufgerufen und das Ergebnis steht dann nachfolgenden Funktionen zur Verfügung. Das gilt auch für globale Variable, die nicht mit `return` (Rückgabewert der Funktion) gesetzt werden. Falls in einer Formel explizit eine andere Formel (nochmals) evaluiert werden soll, muß z.B. `getP()` statt `P` geschrieben werden.

### 3.6.4 Integration in AopWeb

Um das Formelpaket zu nutzen, erzeugt die Anwendung ein Formelsatz-Objekt, wobei als Argument das XML-Dokument (Stream oder File) angegeben werden. Das Formelsatzobjekt liest (im Konstruktor) das XML-Dokument ein, kompiliert die Formeln und erzeugt eine Tabelle für externe Referenzen, d.h. eine Liste von Variablen, für die das Formelobjekt selbst keinen Wert ermitteln kann.

Die umgebende Anwendung nimmt diese Tabelle, die die Namen dieser unbefriedigten Variablen zusammen mit Undefined-Werten enthält, trägt die Werte, die ihr bekannt sind ein und startet die Evaluierung. Danach kann die umgebende Anwendung die Werte der berechneten Variablen abfragen und weiterverwenden.

Für bekannte Datenquellen ist ein Automatisierung des Mappings enthalten. Falls die Formeln an eine ODS-Datenquelle gebunden sind, können aus der MessungsID (Basiselement Messung) automatisch Attribute (aus allen von Messung aus über n:1 - Beziehung erreichbaren Applikationselementen) ermittelt werden und es kann auf Vorhandensein von Spalten mit entsprechenden Namen geprüft werden und damit die XREF's-Tabelle automatisch fertiggestellt werden.

Folgende Variablen werden automatisch gesetzt und können in den Formeln verwendet werden:

- ROWCNT  
gibt bei Vorliegen der externen Daten als Matrix die Anzahl vorhandener Zeilen an (sowohl im row als auch im matrix - Modus),
- ROWNR  
gibt die aktuelle Zeile  $0 \leq \text{ROWNR} < \text{ROWCNT}$  an (nur im row-Modus).

**Messung 80 Teilmatrix 1**

PntNr	PNTNR	\$KEY	ALPHA	BEFF	BRBLNR	DATUM	F	MEFFW	NMOT	P	P
-	-	-	%	g/kWh	-	ddmmyy	-	Nm	l/min	-	-
1	1.0	252.47	100.0	199.8887	21.0	13-FEB-01	261.166	2493.9524	1259.9003	329.08386	329.0432191558902
2	2.0	255.59	100.0	197.12004	21.0	13-FEB-01	263.12997	2512.707	1259.9629	331.5215	331.5341173076777
3	3.0	259.12	100.0	194.45857	21.0	13-FEB-01	264.69254	2527.6284	1260.1091	333.52856	333.5415986276764
4	4.0	262.24	100.0	193.78825	21.0	14-FEB-01	265.31863	2533.6072	1260.0837	334.32733	334.3238091114793
5	5.0	265.37	100.0	194.04408	21.0	14-FEB-01	265.07257	2531.2576	1259.8237	334.0156	333.9448433704932
6	6.0	268.50	100.0	194.22148	21.0	14-FEB-01	264.23465	2523.2559	1260.0925	332.95193	332.9602164337799
7	7.0	270.22	100.0	200.62425	22.0	14-FEB-01	260.86743	2491.1013	1259.9277	328.68924	328.6742234920601
8	8.0	272.10	100.0	197.0836	22.0	14-FEB-01	263.82877	2519.3801	1260.0891	332.39883	332.4478862753132

P_MIN	P_MEAN
328.6742234920601	332.05873922179626

Anzeige: 1      Zeilen: 10

Auswahl: Teilmatrix      Lokale Spalten: PNTNR      Rest: AL\_STELL      Meßpunktbereich: Bereichsbeginn:

Abbildung 3.3: AopWeb-Anzeige

Dieses Beispiel hier zeigt, daß berechnete Spalten zusätzlich zu vorhandenen (mit gemessenen Werten) Spalten gleichen Namens angezeigt werden können.

Nicht beinhaltet sind in Details (API) zur Einbettung des Pakets in andere Anwendungen, die Programmierung eigener Erweiterungspakete (z.B. zur Bereitstellung von Methoden der linearen Algebra (LAPACK) oder Pakete zu Berechnungen mit komplexen Zahlen usw.) und die Adaption an andere Datenquellen/senken. Die ist Teil des für Softwareentwickler und Systemintegratoren bereitgestelltem "Internals"-Handbuch.

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE map SYSTEM "http://localhost:8080/aopweb/xls.dtd">
<map>
  <!-- cell styles -->
  <style id="hdr" color="aqua" font="bold" bordertop="double red"
  />
  <style id="hdr2" color="yellow" font="bold" borderbottom="thick aqua"
  />
  <style id="data" dataformat="($#,##0_);[Red]($#,##0)" />
  <style id="redcell" color="red" />
  <style id="rownr" font="blue"/>

  <!-- fonts used -->
  <font id="blue" color="blue" size="10" />
  <font id="bold" color="yellow" size="12" weight="bold" />

  <!-- sheets -->
  <sheet title="MD Test">
    <row height="400">
      <cell style="hdr" width="4000">Umfeld:</cell>
      <cell ref="UmfeldName" style="data" />
    </row>
    <row height="400">
      <cell style="hdr">Motornummer:</cell>
      <cell ref="MOTID" style="data" />
    </row>
    <row height="400">
      <cell style="hdr">Info:</cell>
      <cell ref="MotInfo" style="data" />
    </row>
    <row height="400">
      <cell style="hdr" width="4000">Versuch:</cell>
      <cell ref="Versuch" style="data" />
    </row>
    <row/>
    <row>
      <cell style="hdr">Durchmesser:</cell>
      <cell ref="d" style="data">24.5</cell>
    </row>
    <row>
      <cell style="hdr">Messbeginn:</cell>
      <cell ref="MessBeginn" style="data"/>
    </row>
    <row>
      <cell style="hdr">Messende:</cell>
      <cell ref="MessEnde" style="data">nicht beendet</cell>
    </row>

    <row height="100">
      <cell span="6" style="redcell" />
    </row>
    <row>
      <cell style="hdr">Teilmatrix:</cell>
      <cell />
      <!-- width applies to rownr column only -->
      <cell type="matrix" styles="hdr hdr2 rownr" width="800"/>
    </row>
  </sheet>
  <sheet title="Teilmatrix only">
    <row>
      <cell type="matrix" styles="hdr hdr2 rownr" width="800"/>
    </row>
  </sheet>
</map>

```



## Kapitel 4

# Integration von Anwendungen

Es gibt mehrere unterschiedliche Möglichkeiten um Fremdanwendungen an AopWeb anzuhängen bzw. um Fremdanwendungen via AopWeb mit ODS Daten zu versorgen. I.A. wird dann mit AopWeb in den Daten navigiert und, wenn ein Datenbereich ausgewählt ist, die Fremdanwendung mit diesen Daten aktiviert, ohne direkten ODS-Zugang

Dazu gibt es mehrere Möglichkeiten (von der standardmäßig vorhandenen EXCEL-Schnittstelle einmal abgesehen):

- Eingebaute oder angefügte Applikationen laufen im selben Prozess der VM wie AopWeb oder zumindest am gleichen Rechner. Das ist der Fall bei der zentralen Auswertung von Daten ohne viel Interaktion mit dem Benutzer, bzw.
- benutzerseitige Anwendungen, die eingebettet (als Pluggin oder Applet) direkt im Browser exekutiert werden. Das trifft vor allem auf interaktive Anwendungen zu, die von AopWeb nur mit Daten versorgt werden. Das mitgelieferte Grafikapplet (vorgestellt in Kapitel ??) ist ein Beispiel für Anwendungen dieser Art.
- benutzerseitige Anwendungen, die mittels SOAP mit AopWeb kommunizieren. Dazu ist die um SOAP erweiterte Version von AopWeb Voraussetzung.

Wahrscheinlich werden auch gemischte Fälle auftreten, wie z.B. Anwendungen im Hintergrund, die aus Daten komplizierte Graphiken errechnen und das Ergebnis dem Browser des Benutzers retournieren, oder das Anzeigen von Kopier-, Backup- und Transformationsoperationen auf Daten auf einen Rechner im Hintergrund um Benutzerinteraktion mit z.B. Drag'n'Drop im Browser zu ermöglichen.

In allen Fällen kann AopWeb zur Navigation in ASAM-Daten verwendet werden, und ähnlich wie beim vielen Dateimanagern (z.B. MS Windows Explorer) kann AopWeb zum Starten bestimmter Applikationen durch Anklicken von Datensätzen (Applikationselementinstanzen) in der momentan angezeigten Seite verwendet werden.

## 4.1 Erweiterung mit SOAP

SOAP (Simple Object Access Protocol) erlaubt den Zugriff auf AopWeb von Zusatzkomponenten aus, i.a. Anwendungen wie spezielle Editoren. Z.B. ist der im Bild ?? gezeigte Meßdateneditor auf diese Art angekoppelt und kann auf von AopWeb selektierte Daten zugreifen.

Ein SOAP-Adapter zum Betrieb des optional erhältlichen Editors ist in AopWeb enthalten. Um Anpassungen an andere Anwendungen zu ermöglichen, sind die Quellen für den Adapter Teil einer AopWeb-Lieferung.

Webservices via SOAP werden registrier, z.B. aus dem DOS-Fenster mit

```
java org.apache.soap.server.ServiceManagerClient
http://galois:8080/aopweb/servlet/rpcrouter
deploy dds/editservice.xml
```

wobei der URL des RPC-Routers anzupassen ist und der classpath die div. Libraries (SOAP, XML) enthalten muß (dds ist das Verzeichnis, in dem bei mir lokal die Deskriptoren sind).

Der Servicemanager sollte sich ca. so wie hier gezeigt präsentieren.

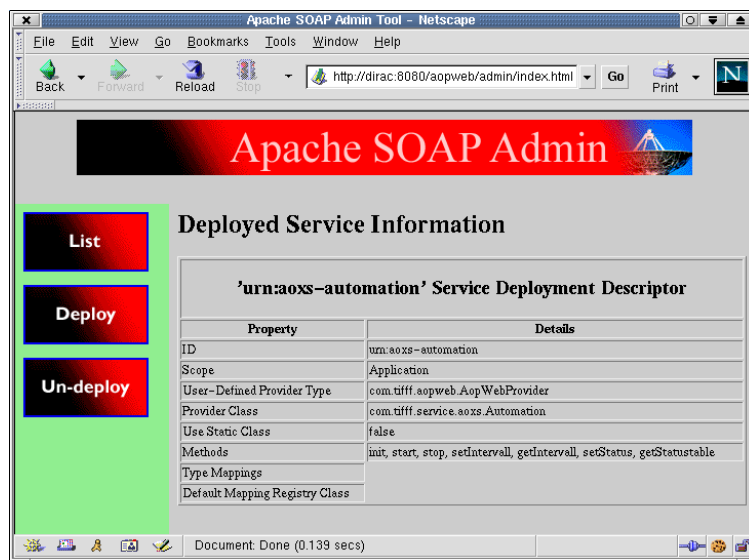


Abbildung 4.1: Servicemanager

Weitere Informationen: Siehe Dokumentation des Editors.

## 4.2 Applets

Zur graphischen Präsentation sind 2 Applets (zur X-Y Präsentation bzw. zur Anzeige als kotierte Projektion<sup>1</sup>) enthalten. Diese Applets fordern nach dem

<sup>1</sup> Ausgabe beliebiger Daten als Isolinien ohne weitere Vorverarbeitung bringt I.A. unbrauchbare Resultate

Laden die Daten über einen TCP-Kanal, der Datenvektoren übermittelt, von AopWeb an. Diesen Kanal können auch andere eingebrachte Applets verwenden. Das Format ist sehr einfach: Pro Request HTTP-GET wird ein Kanal übertragen, der immer aus IEEE Double - Werten besteht, mit 2 UTF encodierten vorlaufenden Strings für Meßgrößenname und Einheit. Falls der Kanal nicht existiert: 0 Bytes.

### 4.3 Externe Prozesse

Dabei können registrierte Anwendungen Informationen über die in AopWeb angewählten Elemente erhalten und dann beliebige Datenverarbeitungsaufgaben mit diesen Parametern durchführen. Die Prozesse laufen getrennt vom AopWeb ab und können daher in beliebiger Programmiersprache implementiert sein. AopWeb besorgt nur die Aktivierung inkl. Bereitstellung von Aufrufargumenten die Umleitung der Standardausgabe des Prozesses (als Text auf die HTML-Seite) sowie die Überprüfung des Ergebnisstatus.

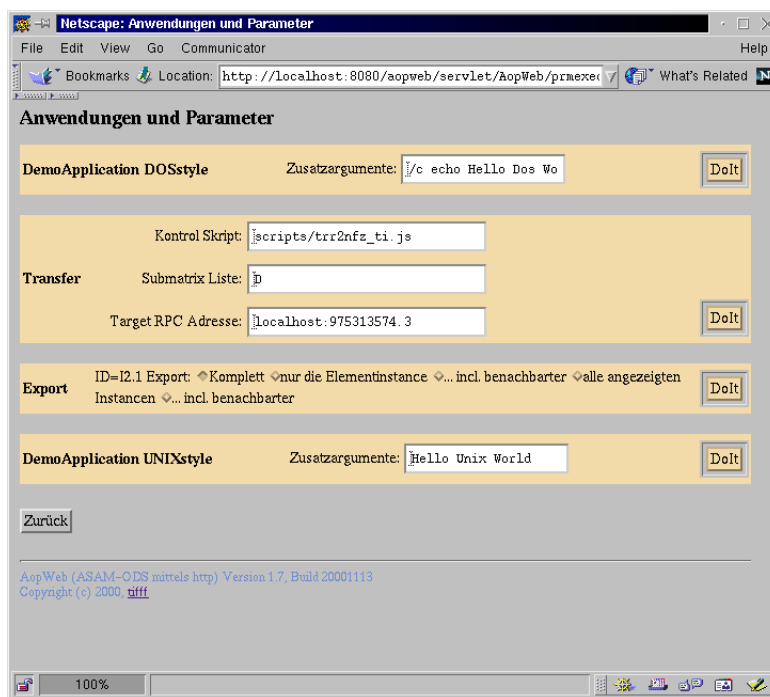


Abbildung 4.2: Applikationsauswahlseite

Die Art der weitergegebenen Information hängt vom Typ (Klasse) der Anwendung ab, so erhält z.B. die Klasse `com/tiff/aopweb/TestApplication`, die zu Demonstrationszwecken mitgeliefert wird, folgende Parameter:

- ASAM-ODS Adresse (Rechnername und RPC-Nummer ),
- Instanzelement-ID

- zusätzliche Argumente direkt vom Benutzer

Normalerweise kann die Testapplikation dazu verwendet werden, um serverseitig Applikationen im Hintergrund zu starten. Es kann auch eine Applikationsklasse mehrfach unter verschiedenen Namen und mit anderen Startparametern registriert werden.

### 4.3.1 Registrierung von Anwendungen

Die Applikationen werden beim Starten vom AopWeb registriert, und zwar unter Verwendung von Parametern aus der Konfigurationsdatei `web.xml`. Die Parameter geben ebenfalls an wann eine Anwendung angeboten werden soll, z.B. nur wenn Instanzen vom Typ *Messung* ausgewählt wurden.

### 4.3.2 Starten von Anwendungen

Aus der Anwendungsseite (shift-click auf die ID einer Instanz) kann der Benutzer die Anwendungen starten, die für den momentanen Zusammenhang registriert wurden. In manchen Fällen wird dann auch die Möglichkeit geboten, zusätzliche Argumente anzugeben.

Während die Anwendung läuft wird die Ausgabe in das Browserfenster umgeleitet, damit der Benutzer den Ablauf von seinem Rechner aus beobachten kann.

### 4.3.3 Programmausgaben

Anwendungen können entweder ihre Ausgaben direkt an den Benutzer weitergeben, z.B. `stdout` in den Browser des Benutzers als einfachen Text senden, oder auch filtern. Im einfachsten Fall wäre dies nur die Umwandlung in HTML, die Anwendung kann aber auch HTML mit Links zu anderen von der Applikation erstellten Daten, wie z.B. Graphiken (gif, jpeg, ..) oder anderen Dokumenten von Belang, erzeugen. Es sind noch viele andere Möglichkeiten denkbar nur deren Diskussion würde den Rahmen dieses Dokuments sprengen.

Die in der Standardversion von AopWeb enthaltene Testanwendung zum Aufruf von externen Programmen steht sowohl kompiliert als auch als Quelltext zur Verfügung. Entwickler können selbst weitere Anwendung (z.B. um spezifisches Filtern der Ausgabe durchzuführen) ableiten, indem sie den Quelltext ändern und ihre neue Anwendung mit einem anderen Klassennamen zu AopWeb hinzufügen (registrieren).

### 4.3.4 Musteranwendungen

Im Paket sind folgende Anwendungen enthalten <sup>2</sup>:

- Eine `TestApplication`, wie zuvor erwähnt, zum Starten externer Programme, meist durch Starten von Scripts, die noch einige Laufzeitparameter anpassen. Der Dateiname der zu startenden Anwendung muß im `web.xml` angegeben werden. Im ausgelieferten `web.xml` sind dafür zwei Beispiele

---

<sup>2</sup>`TestApplication` im Grundpaket, alle anderen im erweiterten AopWebS Paket

zu finden, eines für Servlets, die in einer MS-Windows Umgebung laufen, ein zweites für UNIX. Beide Versionen sind ein Aufruf an das **echo** Kommando.

- **GeneralApplication**, eine Verallgemeinerung der **TestApplication**. Es kann eine komplette Startzeile angegeben werden, in der Variable der Form `%name%` durch den entsprechenden Wert ersetzt werden. Dazu werden die Werte aus der “posted” Form genommen, aus den von **AopWeb** gesetzten Variablen (`aid`, `iid`), aus der ODS-Session (`host`, `progNr`, `user`, `passwd`) oder aus den System-Variablen (z.B. `user.home`). In `Web.xml` können die in der Form zu transportierenden Variablen definiert werden (inkl. Label und ev. Eingabefeldbreite).
- **ExportApplication**, die sehr eng an **AopWeb** angeknüpft ist, da sie das Exportschema verwenden muß, das zuvor zum Servlet hochgeladen wurde (siehe Kapitel ??).
- **TransferApplication**, ist der **TestApplikation** ähnlich, fragt aber nach zusätzlichen Parametern. Diese Anwendung kann nur verwendet werden, wenn zusätzlich **AOXS** installiert wurde. Die Anwendung wird auch nur dann angeboten wenn die Bedingungen aus den Servletparametern (`web.xml`) erfüllt werden (der Browser auf einem bestimmten Applikationselement positioniert wurde). In diesem Fall wird eine **ASAM-** zu **ASAM-Server** Datenübertragung für die ausgewählte Instanz eingeleitet. Die Art der Übertragung (Adresse des Zielsystems, Transformation) wird extern festgelegt, dies kann durch eine Datei mit Steuerparametern oder durch Ändern der Parameter in `web.xml` geschehen.  
Diese Art eines einzelnen Transfers ist nicht zu verwechseln mit dem automatisierten Transfer. Im automatisierten Transfer startet **AopWeb** nur das zum **WebService** “Transfer” bzw. “Automation” gehörende GUI.
- **WebApplication** ist eine **JNLP - Webstart** Interface. Es werden Substitutionen im Lauch-Deskriptor (**JNLP-File**) durchgeführt, dieser wird zum User-Agent geschickt, worauf die Clientanwendung geladen wird.

Alle im Servletdeskriptor enthaltenen Konfigurationen sind Beispiele, die im Bedarfsfall zu adaptieren sind. Falls “**GeneralApplication**” nicht ausreicht, um eine bestimmte Anwendung zu aktivieren, kann auch eine eigene “**Application**”-Klasse integriert werden.

#### 4.3.5 Erzeugen eigener Anwendungen

Das Erstellen neuer Anwendungen ist relativ einfach: Durch Erweiterungen der Klasse `com.tiff.aopweb.Application` (implementieren der abstrakten Methoden) und Hinzufügen der kompilierten Klasse zum `aopweb.jar` wird die Anwendung verfügbar.



# Kapitel 5

## Installation

Der Installationsprozess hängt stark von der verwendeten Servletengine ab. Die Engine sollte den Standard entsprechend Version 2.3 (z.B. Tomcat 4) implementieren.

AopWeb wurde ursprünglich mit Apache JServ (Windows und Unix), Servletexec (Windows) und Apache Tomcat (Windows und Unix) getestet. Neuere Versionen von AopWeb (2.x) wurden nur mit der Referenzimplementierung der Servletspezifikation (Tomcat) getestet. Die nachfolgende Beschreibung bezieht sich daher auf Tomcat.

### 5.1 Apache Tomcat

Tomcat ist die von JavaSoft anerkannte Referenzimplementierung der Servletspezifikation, frei verfügbar (Apache-Lizenzmodell) und ladbar von <http://www.apache.org>. Diese Implementierung ist auch in einigen Applikationsservern vorhanden.

Tomcat kann mit ISAPI (Microsoft) NSAPI (Netscape WEB server) und Apache kombiniert werden oder als Einzelapplikation (standalone) betrieben werden.

Die Installation erfolgt entsprechend der Angaben in der dem Paket beiliegenden Anleitung. Es empfiehlt sich die Umgebungsvariable `CATALINA_HOME` auf das Verzeichnis zeigen zu lassen, in dem das Paket ausgepackt wurde. `JAVA_HOME` sollte auf das zu verwendende Java JDK zeigen.

#### 5.1.1 Starten von Tomcat

In der Standalone-Variante kann Tomcat dann für Windows mit

```
%CATALINA_HOME%\bin\startup
```

bzw. für Unix mit

```
$CATALINA_HOME/bin/startup.sh
```

gestartet werden.

Nach Start von tomcat ist die Dokumentation zu Tomcat online verfügbar: <http://localhost:8080/tomcat-docs/index.html>

**Achtung:** AopWeb kann, z.B. zur Erzeugung von Excel-Spreadsheets einigen virtuellen Speicher benötigen. Empfehlenswert ist, vor dem Starten von Tomcat die Umgebungsvariable `JAVA_OPTS` auf `-server` zu setzen. Falls weiterhin `OutOfMemory` Fehler auftreten, kann der der VM zur Verfügung gestellte Bereich weiter erhöht werden, z.B. mit `-server -Xmx128m`.

### 5.1.2 Konfigurieren

Tomcat wird über die Datei `conf/server.xml` konfiguriert. AopWeb sollte mit der ausgelieferten Grundeinstellung funktionieren, im Bedarfsfall könne aber ein paar Parameter angepasst werden, z.B.

- AopWeb ab 2.11 erzeugt keine Log-Einträge für Felder der HTTP-Requests mehr. Falls diese Informationen z.B. zu Debugzwecken benötigt werden, kann das `RequestDumperValve` und/oder das `AccessLogValve` aktiviert werden, jeweils durch Entfernen der Kommentarmarken um das entsprechende Valve-Element.
- Durch Änderung diverser Debug-Attribute (z.B. statt `debug="0"`: `debug="9"`) kann die Menge der erzeugten Logeinträge erhöht werden, um in Problemfällen mehr Informationen aus den Logdateien zu erhalten.

### 5.1.3 Zugriffsberechtigungen

Nicht vergessen: AopWeb bevorzugt Authentication, d.h. entsprechende User-Einträge sollten nach der Installation von Tomcat gemacht werden, z.B. in `conf/tomcat-users.xml`:

```
<tomcat-users>
<!-- users for examples included with tomcat -->
  <user name="tomcat" password="tomcat" roles="tomcat" />
  <user name="role1" password="tomcat" roles="role1" />
  <user name="both" password="tomcat" roles="tomcat,role1" />
<!-- aopweb users, see aopweb's web.xml for roles used -->
  <user name="Me" password="free" roles="admin" />
  <user name="You" password="too" roles="register,export,launch" />
  <user name="Guest" password="guest" roles="lookup" />
</tomcat-users>
```

Damit haben die User `tomcat`, ... keinen Zugriff auf AopWeb (sie haben keine der Rollen, die AopWeb anerkennt), der User `Guest` hat einfaches Leserecht, der User `You` darf zusätzlich noch neue Datendienste definieren, Exportieren und vorhandene Anwendungen aktivieren, der User `Me` darf alles.

### 5.1.4 Installation von AopWeb

Die Erstinstallation des Servlets in der Basisversion (ohne SOAP) gestaltet sich einfach: Das `.zip` Archiv von AopWeb wird entpackt und die darin enthaltene Datei `aopweb.war` in das `webapps` Verzeichnis der Tomcat Installation kopiert. Danach muß Tomcat neu gestartet werden:

```
>unzip dist/aopweb-2_11.zip
Archive: dist/aopweb-2_11.zip
  creating: aopweb-2_11/
   creating: aopweb-2_11/licenses/
  inflating: aopweb-2_11/ReadMe.html
  inflating: aopweb-2_11/aopweb.war
  inflating: aopweb-2_11/licenses/License-ecs
  inflating: aopweb-2_11/licenses/License-xalan
> cp aopweb-2_11/aopweb.war /opt/apache/jakarta-tomcat/webapps
> /opt/apache/jakarta-tomcat/bin/tomcat.sh start
```

Alle weiteren Installationsschritte sind nur zur Unterstützung erweiterter Funktionalitäten oder bei Updates nötig. Das Vorhandensein der Basisfunktionalität sollte daher jetzt ausprobiert werden (siehe Abschnitt zur Bedienung des Browsers). Die Installation von AopWeb war nur dann vollständig erfolgreich, wenn die Verbindung zu einem ASAM-Server fehlerfrei funktioniert. Die Anzeige der Start- bzw. Anmeldeseite ist alleine noch keine Garantie für eine korrekte Installation.

**Update von AopWeb** Wenn sie AopWeb schon zuvor einmal installiert hatten sind einige zusätzliche Schritte bei der Installation durchzuführen:

- Stellen sie sicher, daß Tomcat nicht mehr läuft (tomcat.sh stop durchführen).
- Falls die Einstellungen der Benutzer in die neue Version übernehmen wollen, müssen sie die .xml-Dateien im Verzeichnis `webapps/aopweb/WEB-INF` (AopWebReg.xml, ...) sichern.
- Entfernen sie die gesamte alte AopWeb-Installation und das alte `aopweb.war` aus dem `webapps` Verzeichnis. Sie können jetzt wie bei einer Neuinstallation fortfahren. Nach dem Neustart von Tomcat wird das `WEB-INF` Verzeichnis neu angelegt. Die dort enthaltene Datei `web.xml` (der Servlet Deploymentdescriptor) sollte mit der gesicherten Version dieser Datei verglichen werden. Alle nicht site-spezifisch sind (Licensekey, registrierte Anwendungen, Authentifizierungstyp, ...) sollten in der gesicherten Version nachgetragen werden.

Um die zuvor gesicherten Benutzereinstellungen wieder einzuspielen brauchen sie jetzt nur noch die gesicherten Dateien ins `WEB-INF` Verzeichnis kopieren<sup>1</sup>.

### 5.1.5 Parametrierung

**Deploymentdescriptor** AopWeb selbst ist ohne Parametrierung funktionstüchtig, alle Parameter befinden sich im Deployment descriptor (`web.xml`) des Servlets (siehe A.1).

Erweiterungen sind spezifisch zu parametrieren. Diesbzgl. enthält der Deploymentdescriptor nur Beispiele.

<sup>1</sup>Bei Upgrade einer Version vor 2.5 auf 2.5 oder höher, sollte nur AopWebReg.xml übertragen werden, Sichten (Views) sind neu anzulegen

**Ladbare Web-Anwendungen** Anwendungen werden im `apps`-Verzeichnis von AopWeb hinterlegt. Diese Anwendung und die Beschreibung der Parametrier-/Konfiguriermöglichkeiten (WebService- und JNLP-Deskriptoren) ist in der Dokumentation der entsprechenden Erweiterung enthalten.

### 5.1.6 Fehlersuche

AopWeb wandelt HTTP-Abfragen in ASAM-ODS-Abfragen um (protocol map). Die primäre Quelle, um Fehler rückzuverfolgen, die die Logging-Dateien, die die Servletengine anlegt. Z.B. finden sich AopWeb Einträge bei Verwendung von Tomcat auf `tomcat/log/catalina.out`.

Probleme bei der Verbindung zu ODS-Servern kann man ev. auch der Sessiontrace (Download von der Transfer-Seite) entnehmen. Meistens wird der Fehler aber "Session timeout" oder ähnlich sein.

# Anhang A

## web.xml

### A.1 Deployment descriptor

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app
PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
"http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>
  <display-name>AopWeb-Apache-SOAP</display-name>
  <description>SOAP accessible AopWeb Application including WebStart</description>

  <!-- to download jnlp files, no jnlp-download-servlet is needed, instead
  WebApplication.java adapts the jnlp file accordingly and jar's
  are always delivered from unprotected area.
  To see how jnlp servlet works, see aopwebapp/test directory
  -->

  <!-- rpcrouter and messagerouter are SOAP supplements -->
  <servlet>
    <servlet-name>rpcrouter</servlet-name>
    <display-name>Apache-SOAP RPC Router</display-name>
    <description>no description</description>
    <servlet-class>
      org.apache.soap.server.http.RPCRouterServlet
    </servlet-class>
    <init-param>
      <param-name>faultListener</param-name>
      <param-value>org.apache.soap.server.DOMFaultListener</param-value>
    </init-param>
  </servlet>

  <servlet>
    <servlet-name>messagerouter</servlet-name>
    <display-name>Apache-SOAP Message Router</display-name>
    <servlet-class>
      org.apache.soap.server.http.MessageRouterServlet
    </servlet-class>
    <init-param>
      <param-name>faultListener</param-name>
      <param-value>org.apache.soap.server.DOMFaultListener</param-value>
    </init-param>
  </servlet>

  <!-- AopWeb is just copy of standalone (soap-less) aopweb -->
  <servlet>
    <servlet-name>AopWeb</servlet-name>
    <servlet-class>com.tiff.aopweb.AopWeb</servlet-class>

  <!-- License, included dynamically when building distribution -->
  <init-param>
    <param-name>LicenseKey</param-name>
```

```

        <param-value>26 DJD4WFR0K7TGCEPWABW0GH9TU22CMB4H24A0ZP4DF1VYVWQKKMKLQVMM27CYHR03ONFGLJ8NF2GA10W165NES467L10KJD9
    </init-param>

    <!-- RPC address default (if no services yet registered) -->
        <init-param>
    <param-name>RpcAddress</param-name>
    <param-value>localhost:975313579.3</param-value>
    </init-param>

    <!-- Only if ASAM server requires unix style authentication and AopWeb
        is executed on Windows platform (default: NONE) -->
        <init-param>
    <param-name>RpcAuthStyle</param-name>
    <param-value>UNIX</param-value>
    </init-param>
        <init-param>
    <param-name>RpcUid</param-name>
    <param-value>777</param-value>
    </init-param>
        <init-param>
    <param-name>RpcGid</param-name>
    <param-value>220</param-value>
    </init-param>

    <!-- RPC request timeout if ODS servers are slow (default: 25000 msec) -->
        <init-param>
    <param-name>RpcTimeout</param-name>
    <param-value>60000</param-value>
    </init-param>

    <!-- Default maximum number of table rows to show -->
        <init-param>
    <param-name>maxrows</param-name>
    <param-value>40</param-value>
    </init-param>

    <!-- Default number of application element table columns (attributes) -->
        <init-param>
    <param-name>maxcolumns</param-name>
    <param-value>16</param-value>
    </init-param>

    <!-- URI set into XML exported data -->
    <!-- is a 'servlet' independent one is available, take that address -->
        <init-param>
    <param-name>aoxs.dtd</param-name>
    <param-value>http://%host%:%port%/aopweb/aoxs.dtd</param-value>
    </init-param>

    <!-- Extra applications to be run at server side may be declared here -->
    <!-- WARNING: All applications registerd here are example only !!! -->
    <!-- There is no guarantee of availibility -->
        <!-- Testapplications (part of AopWeb basic distribution) -->
        <!-- UNIX only, remove section if servlet is installed on other OS -->
        <init-param>
    <param-name>test.class</param-name>
    <param-value>com/tiff/aopweb/TestApplication</param-value>
    </init-param>
        <init-param>
    <!-- the executable -->
    <param-name>test.application</param-name>
    <param-value>/bin/echo</param-value>
    </init-param>
        <init-param>
    <param-name>test.title</param-name>
    <param-value>DemoApplication UNIXstyle</param-value>
    </init-param>
        <init-param>
    <param-name>test.argname</param-name>
    <param-value>Zusatzargumente</param-value>
    </init-param>
        <init-param>
    <param-name>test.argdefault</param-name>

```

```

<param-value>Hello Unix World</param-value>
</init-param>

  <!-- Windows only, remove if installing on other OS -->
  <init-param>
<param-name>testdos.class</param-name>
<param-value>com/tiff/aopweb/TestApplication</param-value>
</init-param>
  <init-param>
<param-name>testdos.application</param-name>
<param-value>cmd</param-value>
</init-param>
  <init-param>
<param-name>testdos.title</param-name>
<param-value>DemoApplication DOSstyle</param-value>
</init-param>
  <init-param>
<param-name>testdos.argname</param-name>
<param-value>Zusatzargumente</param-value>
</init-param>
  <init-param>
<param-name>testdos.argdefault</param-name>
<param-value>/c echo Hello Dos World</param-value>
</init-param>

<!-- ===== FlagChangeutility -->
  <init-param>
<param-name>flg.class</param-name>
<param-value>com/tiff/aopweb/GeneralApplication</param-value>
</init-param>
  <init-param>
  <!-- the executable -->
<param-name>flg.application</param-name>
<param-value>/opt/j2sdk1.3/bin/java -Drpc.host=%host% -Drpc.progrnr=%progrNr% -jar %user.home%/dev/applications/aoputil/dist/aoputil-1.1_1.jar</param-value>
</init-param>
  <init-param>
<param-name>flg.title</param-name>
<param-value>Measured data value flags</param-value>
</init-param>
  <init-param>
<param-name>flg.variables</param-name>
<param-value>OP PAT LOW HIGH</param-value>
</init-param>
  <init-param>
<param-name>flg.OP-label</param-name>
<param-value>Operation</param-value>
</init-param>
  <init-param>
<param-name>flg.OP-value</param-name>
<param-value>set</param-value>
</init-param>
  <init-param>
<param-name>flg.PAT-label</param-name>
<param-value>Bit pattern</param-value>
</init-param>
  <init-param>
<param-name>flg.PAT-value</param-name>
<param-value>0x0001</param-value>
</init-param>
  <init-param>
<param-name>flg.LOW-label</param-name>
<param-value>Lower limit</param-value>
</init-param>
  <init-param>
<param-name>flg.LOW-value</param-name>
<param-value>1.0E10</param-value>
</init-param>
  <init-param>
<param-name>flg.HIGH-label</param-name>
<param-value>Upper limit</param-value>
</init-param>
  <init-param>
<param-name>flg.HIGH-value</param-name>
<param-value>1.0E10</param-value>

```

```

    </init-param>
    <!-- aid-condition -->
    <init-param>
<param-name>fig.aid</param-name>
<param-value>4</param-value>
    </init-param>

<!-- To add further applications which are sufficiently parametrized using
ASAM-Server address, Application element instance ID and optional to
be entered parameters, just repeat section above using different primary
part of names and different values (except xxx.class which has
to point to an existing class implementing "Application").
Adding new application classes requires adding java code.
-->

<!-- Export application -->
<!-- not really external but triggered by same mechanism as exec - types -->
    <init-param>
<param-name>export.class</param-name>
<param-value>com/tiff/aopweb/ExportApplication</param-value>
    </init-param>
</servlet>

<servlet-mapping>
    <servlet-name>rpcrouter</servlet-name>
    <url-pattern>/servlet/rpcrouter</url-pattern>
</servlet-mapping>

<servlet-mapping>
    <servlet-name>messagerouter</servlet-name>
    <url-pattern>/servlet/messagerouter</url-pattern>
</servlet-mapping>

<!-- explicit mapping for default servlet mandatory since tomcat 4.1.12 -->
<servlet-mapping>
    <servlet-name>AopWeb</servlet-name>
    <url-pattern>/servlet/AopWeb/*</url-pattern>
</servlet-mapping>

<!-- 600 min session timeout -->
<session-config>
    <session-timeout>600</session-timeout>
</session-config>

<!-- add security, see conf/tomcat-users.xml for simple realm auth -->
<security-constraint>
    <web-resource-collection>
        <web-resource-name>AopWeb access protection</web-resource-name>
        <url-pattern>/servlet/AopWeb/*</url-pattern>
    </web-resource-collection>
    <auth-constraint>
        <role-name>admin</role-name>
        <role-name>modify</role-name>
        <role-name>register</role-name>
        <role-name>execute</role-name>
        <role-name>launch</role-name>
        <role-name>export</role-name>
        <role-name>lookup</role-name>
    </auth-constraint>
</security-constraint>

<!-- either none, BASIC or FORM authentication can be used (no SSL yet) -->
<!--
<login-config>
    <auth-method>BASIC</auth-method>
    <realm-name>AopWeb basic authentication</realm-name>
</login-config>
-->

<!-- -->
<login-config>

```

```
<auth-method>FORM</auth-method>
<realm-name>AopWeb form-based authentication</realm-name>
<form-login-config>
  <form-login-page>/pages/login.jsp</form-login-page>
  <form-error-page>/pages/error.jsp</form-error-page>
</form-login-config>
</login-config>

<!-- roles referenced by this web application, since tomcat 4.1.12 -->
<security-role>
  <role-name>admin</role-name>
</security-role>
<security-role>
  <role-name>modify</role-name>
</security-role>
<security-role>
  <role-name>register</role-name>
</security-role>
<security-role>
  <role-name>execute</role-name>
</security-role>
<security-role>
  <role-name>launch</role-name>
</security-role>
<security-role>
  <role-name>export</role-name>
</security-role>
<security-role>
  <role-name>lookup</role-name>
</security-role>

<!-- Environment entry example (not used by AopWeb) -->
<!--env-entry>
  <env-entry-description>
    Rows per table (default, may be changed by user)
  </env-entry-description>
  <env-entry-name>maxRows</env-entry-name>
  <env-entry-value>15</env-entry-value>
  <env-entry-type>java.lang.Integer</env-entry-type>
</env-entry-->

</web-app>
```