Zürcher Hochschule Winterthur

Versenden von MMS aus einem Online-Fotoalbum

Projektarbeit 2 (SS 2004)

Dozent
Prof. Dr. Andreas Steffen

Industriepartner
FutureLAB AG, Winterthur
Fabio Vena

Ausgabe der Aufgabe
17. Mai 2004

Abgabetermin	**Verschobener Abgabetermin**
02. Juli 2004	05. Juli 2004

Autoren

Christian Wettstein IT3a _____

Martin Straumann IT3a _____

Übersicht

Ausgangslage für diese Arbeit war der Wunsch vom Versenden von Bildern an einen beliebigen GSM-Benutzer aus einem Online-Fotoalbum heraus. Ein einfacher Prototyp, in Perl realisiert, stand als Beispiel bereits zur Verfügung. Nun sollte das komplette MM7-Protokoll implementiert und alle Funktionen ausgeschöpft werden. Dazu soll ein Konzept und eine mögliche Demoapplikation erstellt werden.

Das Projekt *Versenden von MMS aus einem Online-Fotoalbum* wurde von der Firma Futurelab in Auftrag gegeben und von Prof. Dr. Andreas Steffen betreut. Es wurde uns am 17. Mai 2004 übergeben und besprochen. Abgabetermin ist der 05. Juli 2004.

Ziel dieser Projektarbeit war es eine API zu schreiben, welche einen einfachen und direkten Zugriff auf die Funktionen des Protokolls MM7 erlaubt. Dabei sollte eine Demoapplikation als Testumgebung dienen.

Fazit zum Projekt
Die Implementierung der MM7-Schnittstelle brachte uns einige Probleme ein. Das Hauptproblem war, dass die IP-Adressen der eingesetzten Rechner erst Mitte 4. Projektwoche bei der Firewall von Sunrise aufgeschaltet waren, was zwingend notwendig war für einen Verbindungsaufbau zum MMS-Center. Zudem wurde das MMSC von Sunrise zur Projektzeit umgerüstet. Nur die wichtigsten Teile der Funktionen, welche der Standard MM7 definiert, konnten benutzt werden. Dies hatte Auswirkungen auf die erstellte Programmierschnittstelle, welche deshalb nur die getesteten Nachrichten bedient.
Trotz diesen Problemen konnten die Zielaufgaben der Projektarbeit erfolgreich abgeschlossen werden. Diese Zielaufgaben bestanden aus dem Einarbeiten in die 3GPP MM7 Schnittstelle, dem Aufstellen eines Realisierungskonzeptes, der Implementierung der Schnittstelle MM7 in einer API sowie die Erstellung einer möglichen Demoapplikation.

Dokumenten- und Dateienübersicht

Dokument	Dateiname
Dokumentation der Projektarbeit	Dokumentationen \ Dokumentation_PA_Sna02.doc Dokumentationen \ Dokumentation_PA_Sna02.pdf
Dokumentation über MM7Schnittstelle	Dokumentationen \ 3GPP_V5.10.0.pdf

Java-Packete	Dateiname
DOM	Packages \ dom.jar
Xerces	Packages \ xercesImpl.jar
Activation	Packages \ activation.jar
Mail	Packages \ mail.jar
MySQL Connector	Packages \ mysql-connector-java-2.0.14-bin.jar
MM7.jar	Packages \ mm7.jar
SAAJ	Packages \ saaj-api.jar Packages \ saaj-impl.jar

Source-Code	Dateiname
PHP-Skript zur Erzeugung der XML-Dateien	Source \ createDataFile.php
PHP Webseiten-Beispiel	Source \ MMS-Formular.html
JSP Webapplikation	Source \ Jsp-Webseite.zip
MM7 API als JCreator-Projekt	Source \ MM7-API
MM7 Input als JCreator-Projekt	Source \ MM7-Input
Beispiel für XML-Datendatei	Source \ mm7_submit_req.100.xml
HTTP-Dump einer gültigen Nachricht	Source \ http-Dump.txt

Tools	Dateiname
Ausführbare Exe-Datei	Programm \ MM7.exe
JavaDoc	JavaDoc \ html \ index.html

Inhaltsverzeichnis

1. Aufgabenstellung

Versenden von MMS aus einem Online-Fotoalbum

Projektarbeit im Sommersemester 2004 - PA2 Sna02

Studierende:

- Martin Straumann, IT3a
- Christian Wettstein, IT3a

Partnerfirma:

- futureLAB AG, Schwalmenackerstrasse 4, CH-8400 Winterthur
  http://www.futurelab.ch

Termine:

- Ausgabe: Dienstag, 17.05.2004, 13.30 Uhr bei FutureLAB
- Abgabe: Freitag, 2.07.2004

Beschreibung:

Seit einiger Zeit ist es möglich Bilder von einem GSM Handy aus über den Multimedia Messaging Service (MMS) zu versenden. Weniger bekannt ist die Möglichkeit, dies aus dem Internet zu tun, wo die im 3GPP TS 23.140 Standard definierte MM7 Schnittstelle zur Anwendung kommt. Dabei werden XML-annotierte Bilder in SOAP gekapselt über das bekannte HTTP Protokoll übertragen.

In dieser Projektarbeit soll der automatische Versand von Bildern aus einem web-basierten Online-Fotoalbum heraus an das MMS Center eines GSM Operators realisiert werden. Eine Rapid-Prototype Realisierung existiert schon, nun soll das komplette MM7 Protokoll implementiert werden.

Aufgaben:

- Einarbeitung in die 3GPP MM7 Schnittstelle
- Aufstellen eines Realisationskonzepts
- Implementation und Test der 3GPP MM7 Schnittstelle
- Erstellen einer Demoapplikation

Infrastruktur / Tools:

- Raum: **E523**
- Rechner: 2 Rechner unter Linux
- SW-Tools: Perl, Java

Literatur / Links:

- 3GPP TS 23.140
  Multimedia Messaging Service (MMS)

Winterthur, 8. März 2004

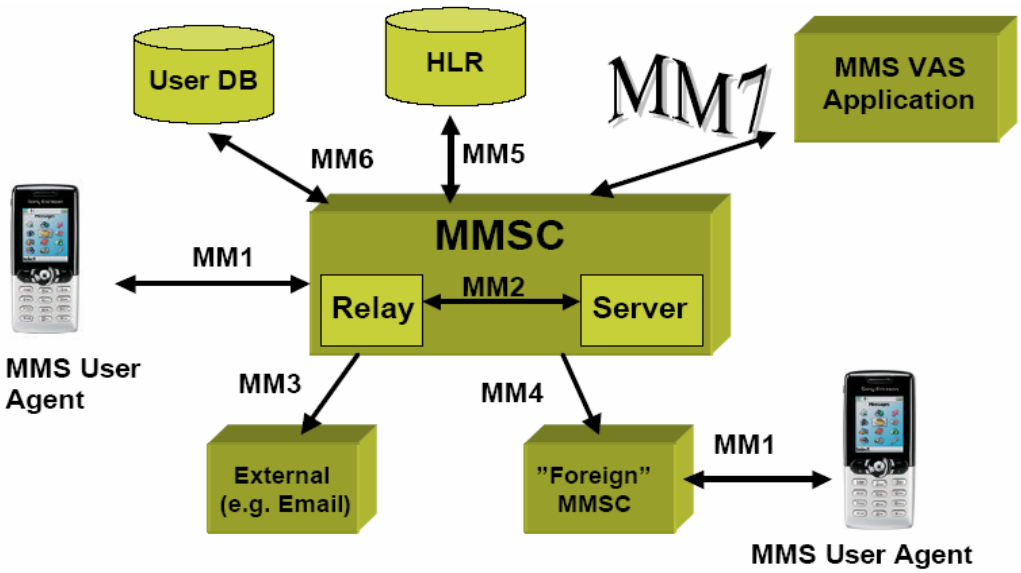
Prof. Dr. Andreas Steffen

2. Vorwort

Wir haben diese Projektarbeit auf Grund der kurzen Beschreibung und vor allem dem Titel ausgewählt. Zu Beginn der Arbeit mussten wir herausfinden was funktioniert und falls Fehler erzeugt wurden herausfinden welche Ursachen sie hatten. Die Schwierigkeit war, dass wir keinerlei gesicherten Dokumentationen bekamen und wir uns hauptsächlich auf einen Prototypen verlassen mussten, von welchem wir sicher sein konnten, dass er funktionierte. Da dieser allerdings in Perl geschrieben wurde, war er uns auch keine grosse Hilfe, denn wir haben uns entschieden, die Arbeit in Java zu realisieren. Was uns die Arbeit allerdings erheblich vereinfachte, war ein HTTP-Dump, den wir erhielten. Bevor wir mit der eigentlichen Hauptaufgabe, der Programmierschnittstelle, beginnen konnten, erstellten wir eine einfache Applikation um alle möglichen Nachrichten-Elemente zu testen und nach dem Prinzip von Try-and-Error herausfinden konnten was der MMS-Gateway akzeptiert und/oder verarbeitet.
Diese Applikation füllte Daten in eine generierte SAOP-Nachricht ab, welche dann über das HTTP-Protokoll an ein gegebenes MMS-Center versandt wurde. Da wir erst sehr spät gesicherte Angaben zum MMS-Center erhielten um damit die definitive Programmierschnittstelle zu erstellen, wurde eigentlich die Demoapplikation rund um die Verwendung der API zuerst fertig gestellt. Schlussendlich bietet diese Arbeit der Firma FutureLAB jedoch die Möglichkeit die MM7-Schnittstelle über eine API zu verwenden, wie es Ericsson als Lieferanten des Sunrise MMS-Centers seit langer Zeit angekündigt hat.

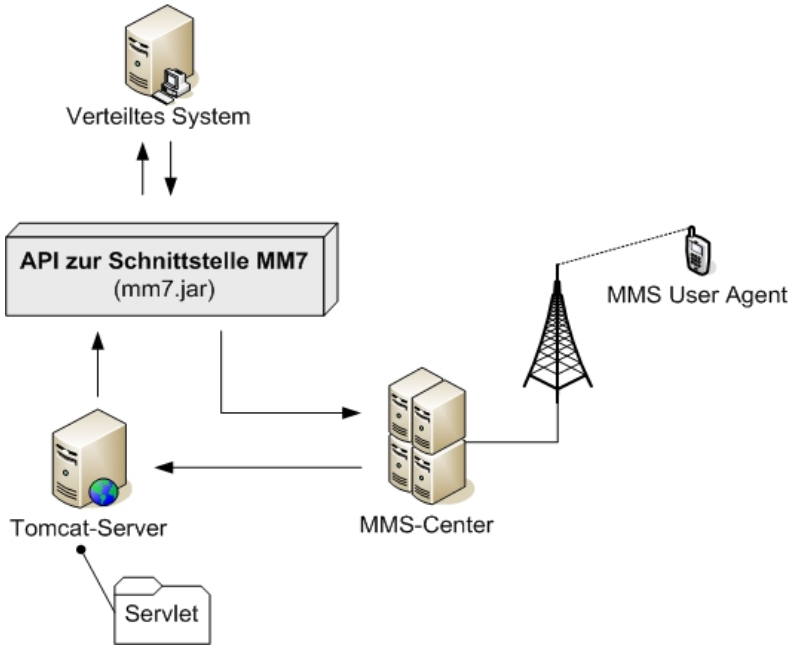
3. Aktuelle Situation und Motivation

Die nachfolgende Grafik zeigt einen Überblick über die Schnittstellen MM1 bis MM7 welche im Zusammenhang mit Multimedia-Nachrichten verwendet werden. Die Anwendung der Schnittstelle MM1 ist bereits weit verbreitet und es existieren immer mehr Dienste, welche darauf basieren. In dieser Projektarbeit wird jedoch eine Implementation der weniger bekannten Schnittstelle MM7 realisiert um später auch über diesen Weg Dienste mit Multimedia-Nachrichten anbieten zu können.



4. Die Schnittstelle MM7

4.1. Grafische Darstellung



4.2. Beschreibung

Die Schnittstelle MM7 definiert die Kommunikation zwischen einem MMS-Center (Gateway) und einem VASP[1]. Ein VAS-Provider stellt die Möglichkeit zur Verfügung von einem Computer aus eine MMS-Nachricht zu versenden. Dabei werden die Daten in eine SOAP-Message verpackt. Diese SOAP-Message wird danach unter anderen über das Protokoll HTTP an das MMS-Center gesendet, welches die Daten ausliest und verarbeitet.
Die SOAP-Nachrichten dienen nur als Transport-Container, allerdings definiert die Schnittstelle MM7 für jede Funktion eine eigene Nachrichtenstruktur, welche mit den geforderten Daten gefüllt werden müssen. Die verschiedenen Nachrichten werden im nächsten Abschnitt genauer erklärt. Wenn ein VASP eine MMS-Nachricht versenden möchte, baut dieser eine Http-Verbindung zum MMS-Center auf, über welche er danach seine SOAP-Nachricht als Anfrage sendet. Der MMS-Gateway sendet über die bestehende Verbindung eine SOAP-Nachricht zurück, in welcher er die Anfrage beantwortet oder eine Fehlermeldung zurückgibt. Danach wird die Verbindung abgebaut.
Falls der VASP in seiner Anfrage eine Lieferbestätigung verlangt hat, baut das MMS-Center zu gegebenem Zeitpunkt eine Verbindung zu einem zuvor angegebenen Servlet auf und sendet eine SOAP-Nachricht, welche vom Servlet eingelesen wird und als Java-Objekt über die API dem VASP übergeben wird. Dasselbe Verfahren kommt auch zur Anwendung falls eine Lesebestätigung angefordert wurde.

[1] VASP: Value Added Service Provider (in obiger Grafik als verteiltes System dargestellt)

4.3. Nachrichten

Da die Firma Sunrise eine MM7-Implementation von Ericsson in Betrieb hat, beziehen wir uns auf die erhaltenen Unterlagen der Firma Ericsson betreffend dem MMS-Center, welches einige Abweichungen vom Standard vorweist. Ergänzend dazu stammen gewisse Informationen allerdings aus der Standard-Definition. Es gibt insgesamt 14 verschiedene Nachrichten, welche vom Ericsson-MMSC unterstützt werden. Sunrise benutzte im Zeitraum der Projektarbeit einen funktionell stark eingeschränkten MMS-Gateway. Nach Abschluss dieser Projektarbeit sollten gemäss Information seitens Sunrise allerdings alle implementierten Funktionen aufgeschaltet sein.
Die folgenden Nachrichtenstrukturen sind nun die definitiv und werden vom MMS-Center unterstützt. Es ist aber zu beachten, dass wir von der Sunrise nur die Beschreibung der Nachrichten *MM7_submit.REQ*, *MM7_submit.RES*, *MM7_delivery_report.REQ*, *MM7_delivery_report.RES*, *MM7_read_reply.REQ* und *MM7_read_reply.RES* erhalten haben. Die definitive Beschreibung der restlichen Nachrichtentypen wurde von Sunrise(Ericsson) noch nicht erstellt.

4.3.1. MM7_submit.REQ

Die SOAP-Nachricht MM7_submit.REQ stellt eine Anfrage an den MMS-Gateway um eine MMS-Nachricht versenden zu dürfen. Falls die enthaltenen Authentifizierungsinformationen bestätigt werden, erstellt der Gateway eine MMS-Nachricht mit den bereits gelieferten Daten.

Mögliche Elemente und Zusatzinformationen

Element Name	auszufüllen	Beispiel	Beschreibung
Transaction-ID	obligatorisch	1547_48596542	Identifikation aller Transaktionen zu einer Übertragung

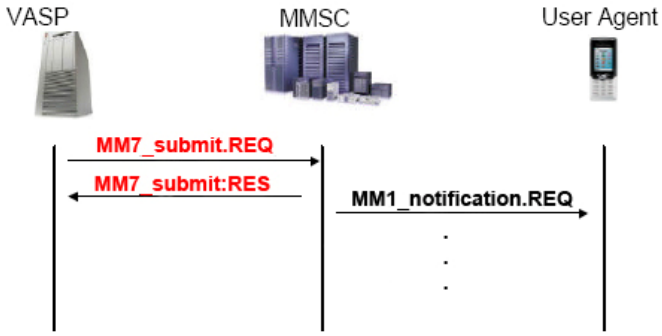
Elemente im Header-Teil der SOAP Nachricht

Element Name	auszufüllen	Beispiel	Beschreibung
Message-Type	obligatorisch	MM7_submit.REQ	Funktionstyp der Nachricht
Version	obligatorisch	5.6.0	Identifikation der verwendeten Protokollversion
VASP-ID	optional	c1Cad9	ID des VASP für den MMS-Gateway
VAS-ID	optional	45	Identifikation der Applikation
From	optional	0787073223	Adresse des Senders
Recipients	obligatorisch	0787073223	Adresse des Empfängers. Mehrere Empfänger sind möglich (Arrayliste)
Service-Code	optional	1001	Ermöglicht den Inhalt der Nachricht einem Dienst zuzuordnen
Message-Class	optional	Informational	Klasse der Nachricht (Advertisement, Informational, Auto)
Report-Address	ja falls Report verlangt wird	http:/www.xy.ch/	Servlet-Adresse an welche der Report gesendet werden soll.

Expiry-Date	optional	20040212143500	Ablaufzeit der Nachricht.
Earliest-Delivery-Time	optional	20040212143500	Zeit um welche die Nachricht frühestens versendet werden soll.
Delivery-Report	optional	true oder false	Forderung einer Lieferbestätigung
Read-Reply	optional	true oder false	Forderung einer Lesebestätigung
Sender-Visibility	optional	true oder false	Ob Absender angezeigt werden soll
Date	optional	20040212143500	Setzen eines Zeitstempels
Priority	optional	Low, Normal oder High	Wichtigkeit des Nachrichteninhalts
Subject	optional	Text	Thema der Nachricht
Free-Text	optional	Text	freie Beschreibung des Inhalts
Content-Type	optional	text/plain	Content-Type des Inhalts
Message-File href:cid	optional	href:cid	Referenz auf erstes Attachment

Elemente im Body-Teil der SOAP Nachricht

Möglicher Ablauf einer Submit-Nachrichten-Übermittlung



4.3.2. MM7_submit.RES

Als Antwort auf eine MM7_submit_REQ wird diese Nachricht geschickt. Sie dient als Bestätigung des Empfangs und liefert Statusinformation oder allfällige Fehlermeldungen.

Mögliche Elemente und Zusatzinformationen

Element Name	auszufüllen	Beispiel	Beschreibung
Transaction-ID	obligatorisch	1547_48596542	Identifikation aller Transaktionen zu einer Übertragung

Elemente im Header-Teil der SOAP Nachricht

Element Name	auszufüllen	Beispiel	Beschreibung
Message-Type	obligatorisch	MM7_submit.RES	Typ der Nachrichtenfunktion
Version	obligatorisch	1	Identifikation der verwendeten Protokollversion
Message-ID	optional	25325	Identifikation der Nachricht
Request-Status	obligatorisch	1000	Statuscode der erhaltenen Nachricht
Request-Status-Text	optional	Success	Textbeschreibung des Status

Elemente im Body-Teil der SOAP Nachricht

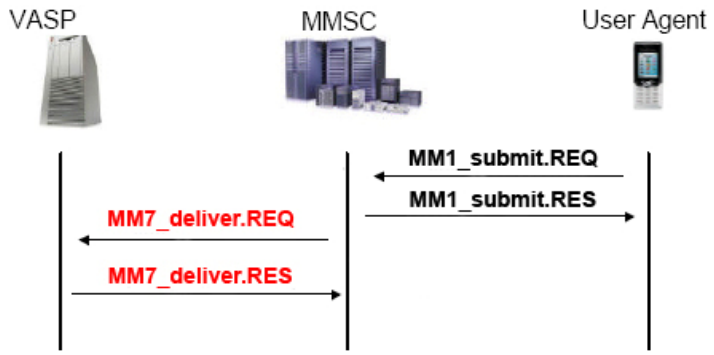
4.3.3. MM7_deliver.REQ

Falls ein User Agent eine MMS-Nachricht an einen VASP sendet, schickt der MMS-Gateway einen MM7_deliver.REQ an den gewünschten VASP, welcher die Informationen und Attachments aus der gesendeten MMS-Nachricht enthält.

Mögliche Elemente und Zusatzinformationen

Element Name	auszufüllen	Ort	Beschreibung
TransactionID	obligatorisch	SOAP Header	Identifikation aller Transaktionen zu einer Übertragung
MessageType	obligatorisch	SOAP Body	Funktionstyp der Nachricht
MM7Version	obligatorisch	SOAP Body	Identifikation der verwendeten Protokollversion
MMSRelayServerID	optional	SOAP Body	Adresse des MMS Relay/Server
LinkedID	optional	SOAP Body	ID welche vom VASP in einer vorhergehenden MM7_submit.REQ Nachricht verwendet wurde.
Sender	obligatorisch	SOAP Body	Adresse des Senders
Recipients	optional	SOAP Body	Adresse des Empfängers. Mehrere Empfänger sind möglich (Arraylist)
TimeStamp	optional	SOAP Body	Zeit und Datum der Übertragung. (time stamp)
ReplyChargingID	optional	SOAP Body	Im Fall eines reply-charging wenn ein reply-MM übertragen wurde ist dies die ID des Original MM an welche geantwortet wird.
Priority	optional	SOAP Body	Priorität der Nachricht
Subject	optional	SOAP Body	Thema der Nachricht
Content-Type	obligatorisch	MIME Header des Anhangs	Der Contenttype des MM's Content
Content	optional	SOAP Body	Der Content der Multimediamessage

Möglicher Ablauf einer Deliver-Nachrichten-Übermittlung



4.3.4. MM7_deliver.RES

Bei einer erfolgreichen Übermittlung der Daten antwortet der VASP mit einer Nachricht vom Typ MM7_deliver.RES und übergibt den Status oder eine Fehlermeldung.

Mögliche Elemente und Zusatzinformationen

Element Name	auszufüllen	Ort	Beschreibung
TransactionID	obligatorisch	SOAP Header	Identifikation aller Transaktionen zu einer Übertragung
MessageType	obligatorisch	SOAP Body	Funktionstyp der Nachricht
MM7Version	obligatorisch	SOAP Body	Identifikation der verwendeten Protokollversion
ServiceCode	optional	SOAP Body	Informationscodes welche vom VASP unterstützt werden. Es gibt vordefinierte Codes.
StatusCode	obligatorisch	SOAP Body	Status über den Stand des Requestes
StatusText & Details	optional	SOAP Body	Textbeschreibung des Status. Sollte dem Requeststatus ensprechen.

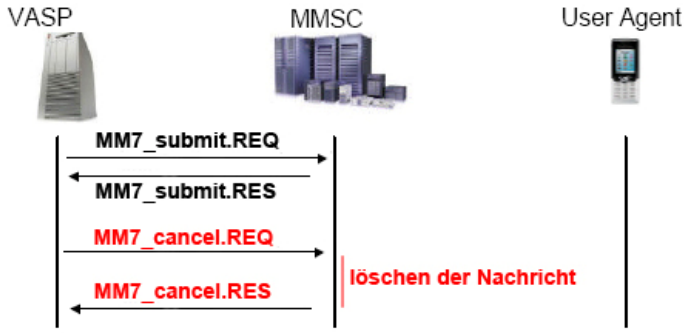
4.3.5. MM7_cancel.REQ

Falls der VASP das Versenden einer Nachricht stoppen bzw. löschen will, kann er eine MM7_cancel.REQ Nachricht an den MMSC senden. Der MMSC überprüft danach mittels der Message-ID den Status der Nachricht und löscht gegebenenfalls alle noch zu versendenden Nachrichten.

Mögliche Elemente und Zusatzinformationen

Element Name	auszufüllen	Ort	Beschreibung
TransactionID	obligatorisch	SOAP Header	Identifikation aller Transaktionen zu einer Übertragung
MessageType	obligatorisch	SOAP Body	Funktionstyp der Nachricht
MM7Version	obligatorisch	SOAP Body	Identifikation der verwendeten Protokollversion
VASPID	optional	SOAP Body	ID des VASP für den MMS Relay server
VASID	optional	SOAP Body	Identifikation der Applikation
SenderAddress	optional	SOAP Body	Adresse des Senders
MessageID	obligatorisch	SOAP Body	Identifikation der ID der Message welche gelöscht werden soll.

Möglicher Ablauf einer Deliver-Nachrichten-Übermittlung



4.3.6. MM7_cancel.RES

Das MMS-Center schickt als Bestätigung des Löschvorgangs eine Nachricht vom Typ MM7_cancel.RES.

Mögliche Elemente und Zusatzinformationen

Element Name	auszufüllen	Ort	Beschreibung
TransactionID	obligatorisch	SOAP Header	Identifikation aller Transaktionen zu einer Übertragung
MessageType	obligatorisch	SOAP Body	Funktionstyp der Nachricht
MM7Version	obligatorisch	SOAP Body	Identifikation der verwendeten Protokollversion
StatusCode	obligatorisch	SOAP Body	Dieses Element beinhaltet den Statuscode der zuvor gesendeten Nachricht.
StatusText & Details	optional	SOAP Body	Textbeschreibung des Status. Sollte dem Requeststatus ensprechen.

4.3.7. MM7_replace.REQ

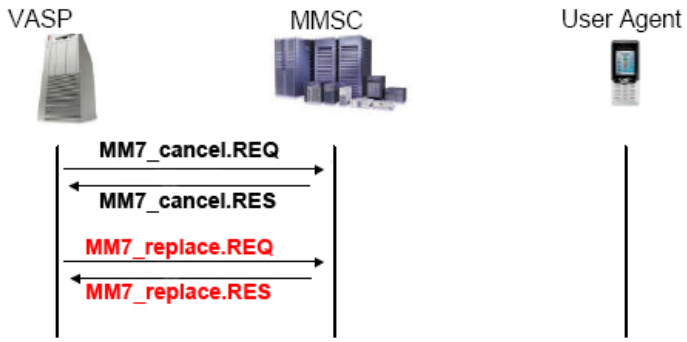
Falls der Inhalt einer MMS-Nachricht verändert werden soll, hat ein Absender die Möglichkeit über einen MM7_replace.REQ die frühere Nachricht zu ersetzen.

Mögliche Elemente und Zusatzinformationen

Element Name	auszufüllen	Ort	Beschreibung
TransactionID	obligatorisch	SOAP Header	Identifikation aller Transaktionen zu einer Übertragung
MessageType	obligatorisch	SOAP Body	Funktionstyp der Nachricht

MM7Version	obligatorisch	SOAP Body	Identifikation der verwendeten Protokollversion
VASPID	optional	SOAP Body	Identifikation des VASP
VASID	optional	SOAP Body	Identifikation der Applikation
MessageID	obligatorisch	SOAP Body	ID der Message, welche welche mit der zuvorgesetzten ID überschreben werden soll.
ServiceCode	optional	SOAP Body	Informationscodes welche vom VASP unterstützt werden. Es gibt vordefinierte Codes.
TimeStamp	optional	SOAP Body	Zeit und Datum der Übertragung (time stamp)
EarliestDeliveryTime	optional	SOAP Body	Die Zeit um welche die MM versendet werden soll. (Timestamp)
Read Reply	optional	SOAP Body	Eine Anfrage für eine Lesebestätigung.
allowAdaptations	optional	SOAP Body	Angabe ob Veränderungen erlaubt
Content-Type	bedingt	MIME part Header	Content-Type der Nachricht

Möglicher Ablauf einer Replace-Nachrichten-Übermittlung



4.3.8. MM7_replace.RES

Als Antwort auf eine Replace-Anfrage sendet der MMS-Gateway eine Nachricht vom Typ MM7_replace.RES.

Mögliche Elemente und Zusatzinformationen

Element Name	auszufüllen	Ort	Beschreibung
TransactionID	obligatorisch	SOAP Header	Identifikation aller Transaktionen zu einer Übertragung
MessageType	obligatorisch	SOAP Body	Funktionstyp der Nachricht
MM7Version	obligatorisch	SOAP Body	Identifikation der verwendeten Protokollversion
StatusCode	obligatorisch	SOAP Body	Dieses Element beinhaltet den Statuscode der zuvor gesendeten Nachricht.
StatusText & Details	optional	SOAP Body	Textbeschreibung des Status. Sollte dem Requeststatus ensprechen.

4.3.9. MM7_delivery_report.REQ

Das MMS-Center sendet als Bestätigung für eine erfolgreiche Weiterleitung an einen User-Agent die Nachricht MM7_delivery_report.REQ. Dies allerdings nur falls in der vorausgehenden Submit-Request-Nachricht eine Lieferbestätigung verlangt wurde.

Mögliche Elemente und Zusatzinformationen

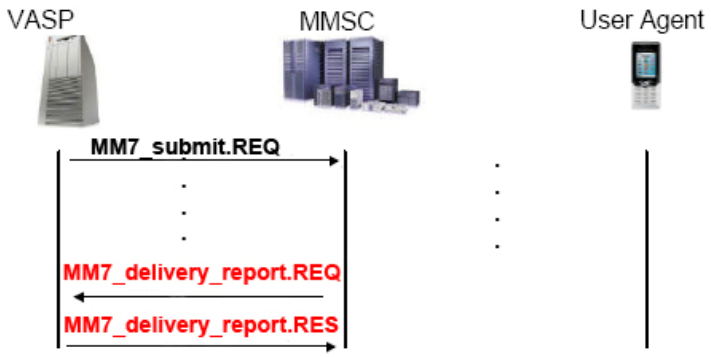
Element Name	auszufüllen	Beispiel	Beschreibung
Transaction-ID	obligatorisch	1547_48596542	Identifikation aller Transaktionen zu einer Übertragung

Elemente im Header-Teil der SOAP Nachricht

Element Name	auszufüllen	Beispiel	Beschreibung
Message-Type	obligatorisch	MM7_delivery_report.REQ	Funktionstyp der Nachricht
Version	obligatorisch	5.6.0	Identifikation der verwendeten Protokollversion
Message-ID	obligatorisch	7897897	Identifikation der zu bestätigenden Nachricht
Recipient	obligatorisch	0787073223	Adresse des Empfängers. Mehrere Empfänger sind möglich (Arraylist)
From	obligatorisch	0787073223	Adresse des Senders
Date	obligatorisch	20040312143500	Zeit und Datum an dem die Nachricht gesendet wurde (retrieved, expired, rejected) (time stamp)
MM-Status	obligatorisch	3000	Status der MM (vordefinierte Codes)
Status-Text	optional	Server Error	Textbeschreibung des Status. Sollte dem Requeststatus ensprechen

Elemente im Body-Teil der SOAP Nachricht

Möglicher Ablauf einer DeliveryReport-Nachrichten-Übermittlung



4.3.10. MM7_delivery_report.RES

Wenn der VASP den MMC oder die Message ID nicht erkennt, sollte der VASP mit einer MM7_Delivery_Report.RES Nachricht antworten. Diese beinhalten einen Status mit dem Grund warum die Nachricht nicht angenommen werden konnte.

Mögliche Elemente und Zusatzinformationen

Element Name	auszufüllen	Beispiel	Beschreibung
Transaction-ID	obligatorisch	1547_48596542	Identifikation aller Transaktionen zu einer Übertragung

Elemente im Header-Teil der SOAP Nachricht

Element Name	auszufüllen	Beispiel	Beschreibung
Message-Type	Obligatorisch	MM7_delivery_report.RES	Funktionstyp der Nachricht
Version	Obligatorisch	5.6.0	Identifikation der verwendeten Protokollversion
Request-Status	Obligatorisch	3000	Dieses Element beinhaltet den Statuscode der zuvor gesendeten Nachricht.
Request-Status-Text	Optional	Server Error	Textbeschreibung des Status. Sollte dem Requeststatus ensprechen.

Elemente im Body-Teil der SOAP Nachricht

4.3.11. MM7_read_reply.REQ

Falls der User-Agent die Anforderung einer Lesebestätigung beantwortet, erkennt das MMS-Center diese MM1-Bestätigung und sendet seinerseits eine Meldung vom Typ MM7_read_reply.REQ an den VASP.

Mögliche Elemente und Zusatzinformationen

Element Name	auszufüllen	Beispiel	Beschreibung
Transaction-ID	obligatorisch	1547_48596542	Identifikation aller Transaktionen zu einer Übertragung

Elemente im Header-Teil der SOAP Nachricht

Element Name	auszufüllen	Beispiel	Beschreibung
Message-Type	obligatorisch	MM7_read_reply.REQ	Funktionstyp der Nachricht
Version	obligatorisch	5.6.0	Identifikation der verwendeten Protokollversion
Message-ID	obligatorisch	7897897	Identifikation der zu bestätigenden Nachricht
Recipient	obligatorisch	0787073223	Adresse des Empfängers. Mehrere Empfänger sind möglich (Arraylist)

From	obligatorisch	0787073223	Adresse des Senders
Date	obligatorisch	20040312143500	Zeit und Datum an dem die Nachricht gesendet wurde (retrieved, expired, rejected) (time stamp)
Read-Status	obligatorisch	3000	Status der MM (vordefinierte Codes)
Status-Text	optional	Server Error	Textbeschreibung des Status. Sollte dem Requeststatus ensprechen

Elemente im Body-Teil der SOAP Nachricht

4.3.12. MM7_read_reply.RES

Als Antwort auf die Nachricht MM7_read_reply.REQ sendet der VASP eine Meldung vom Typ MM7_read_reply.RES und sagt darin ob die Lesebestätigung erfolgreich empfangen wurde oder nicht.

Mögliche Elemente und Zusatzinformationen

Element Name	auszufüllen	Beispiel	Beschreibung
Transaction-ID	obligatorisch	1547_48596542	Identifikation aller Transaktionen zu einer Übertragung

Elemente im Header-Teil der SOAP Nachricht

Element Name	auszufüllen	Beispiel	Beschreibung
Message-Type	obligatorisch	MM7_read_reply.RES	Funktionstyp der Nachricht
Version	obligatorisch	5.6.0	Identifikation der verwendeten Protokollversion
Request-Status	obligatorisch	3001	Dieses Element beinhaltet den Statuscode der zuvor gesendeten Nachricht.
Request-Status-Text	optional	Server Error	Textbeschreibung des Status. Sollte dem Requeststatus ensprechen.

Elemente im Body-Teil der SOAP Nachricht

4.3.13. MM7_RS_error.RES / MM7_VASP_error.RES

Wenn der VASP oder auch das MMS-Center eine Nachricht erhält auf die nicht mit einer entsprechenden Response-Nachricht geantwortet werden kann, sollten diese mit einer Error-Nachricht beantwortet werden.

Mögliche Elemente und Zusatzinformationen zu MM7_RS_error.RES

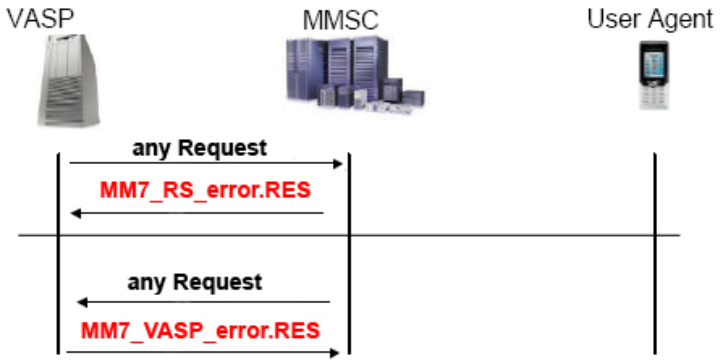
Element Name	auszufüllen	Ort	Beschreibung
TransactionID	obligatorisch	SOAP Header	Identifikation aller Transaktionen zu einer Übertragung
MessageType	obligatorisch	SOAP Body	Funktionstyp der Nachricht
MM7Version	obligatorisch	SOAP Body	Identifikation der verwendeten Protokollversion

StatusCode	obligatorisch	SOAP Body	Error codes (sind zum Teil vordefiniert.)
StatusText & Details	optional	SOAP Body	Textbeschreibung des Errors. Sollte dem Errorcode entsprechen.

Mögliche Elemente und Zusatzinformationen zu MM7_VASP_error.RES

Element Name	auszufüllen	Ort	Beschreibung
TransactionID	obligatorisch	SOAP Header	Identifikation aller Transaktionen zu einer Übertragung
MessageType	obligatorisch	SOAP Body	Funktionstyp der Nachricht
MM7Version	obligatorisch	SOAP Body	Identifikation der verwendeten Protokollversion
StatusCode	obligatorisch	SOAP Body	Error codes (sind zum Teil vordefiniert.)
StatusText & Details	optional	SOAP Body	Textbeschreibung des Errors. Sollte dem Errorcode entsprechen.

Möglicher Ablauf bei Fehlermeldungen



4.4. Request Status und Error Status Codes

Request- und Error Status Codes geben in einer Nachricht vom Typ Response Auskunft über die zuvor übermittelte SOAP-Nachricht. Die Codes sind in 4 Klassen eingeteilt, welche eine Gruppierung der Informationen ergeben. Folgende Auflistung zeigt, welche Art von Informationen die Klassen beschreiben.

- 1xxx : Erfolgreiche Operationen
- 2xxx : Client Fehler (VASP hat eine ungültige SOAP-Nachricht geschickt)
- 3xxx : Server Fehler (MMSC konnte die Nachricht nicht erfolgreich weiterleiten)
- 4xxx : Service Fehler (Falls Dienste nicht verfügbar sind)

Die Status-Codes sind erweiterbar. Die Code-Erweiterungen müssen gegenseitig abgesprochen sein. Erweiterungen sind dann sinnvoll, wenn sie Informationen über zusätzliche Zustände eigener Dienste Auskunft geben. Für die Implementierung von spezifischen Codes sollte die Range x500-x999 eingehalten werden.

Statuscode	Statustext	Bedeutung
1000	Success	Der Request konnte erfolgreich verarbeitet werden
1100	Partial success	Der Request konnte ausgeführt werden, jedoch konnten gewisse Teile des Request nicht verarbeitet werden
2000	Client Error	Ungültiger Request des Clients
2001	Operation restricted	Der Request konnte nicht ausgeführt werden, das die Berechtigung für diesen Command nicht gesetzt sind
2002	Address Error	Mindestens eine Adresse des Request ist vom Format her ungültig bzw. nicht kompatibel mit dem angeschlossenem Netzwerk am MMC
2003	Address Not Found	Mindestens eine Adresse des Request konnte vom MMC nicht gefunden werden.
2004	Multimedia content refused	Der Server konnte den angehängten MIME Content nicht auflösen
2005	Message ID Not Found	Der MMC bzw. VASP konnte die ID der auf die zuvor übertragenen Message nicht finden bzw. erkennen.
2006	LinkedID Not Found	Der MMC konnte nicht die besagte Message, auf das sich die ID bezieht, finden.
2007	Message format corrupt	Ein Elementformat der Nachricht ist ungültig
3000	Server Error	Der Server konnte einen korrekte Request nicht ausführen
3001	Not Possible	Ein Request konnte nicht mehr versendet werden. Dieser Code wird vor allem beim Cancelstatus gebraucht, wenn die gewünschte Message in der Queue vom MMC nicht mehr vorhanden ist.
3002	Message rejected	Der MMC konnte den Request nicht vollständig ausführen
3003	Multiple addresses not supported	Der MMC kann diese Art von mehreren Empfängern nicht ausführen. Der Request sollte mit einem einzelnen Empfänger wiederholt werden.
4000	General service error	Der gewünschte Service kann nicht ausgeführt werden
4001	Improper identification	Der Header des Request entspricht nicht der Identifikation des Clients
4002	Unsupported version	Die übermittelte Version wird nicht unterstützt
4003	Unsupported operation	Der Server unterstützt nicht den angegebenen Messagetyp des Requestes

4004	Validation error	Die SOAP und XML Stuktur konnte nicht aufgelöst werden.
4005	Service Error	Diese Operation verursachte eine Server-fehlfunktion (VASP oder MMC)! Sollte auf keinen Fall wiedergesendet werden
4006	Service unavailable	Der gewünschte Sevice kann zurzeit nicht inAnspruch genommen werden. (Server Busy)
4007	Service denied	Der Client hat keine Berechtigung um diesen Request auszuführen zu lassen.

5. Beschreibung der Programmierschnittstelle

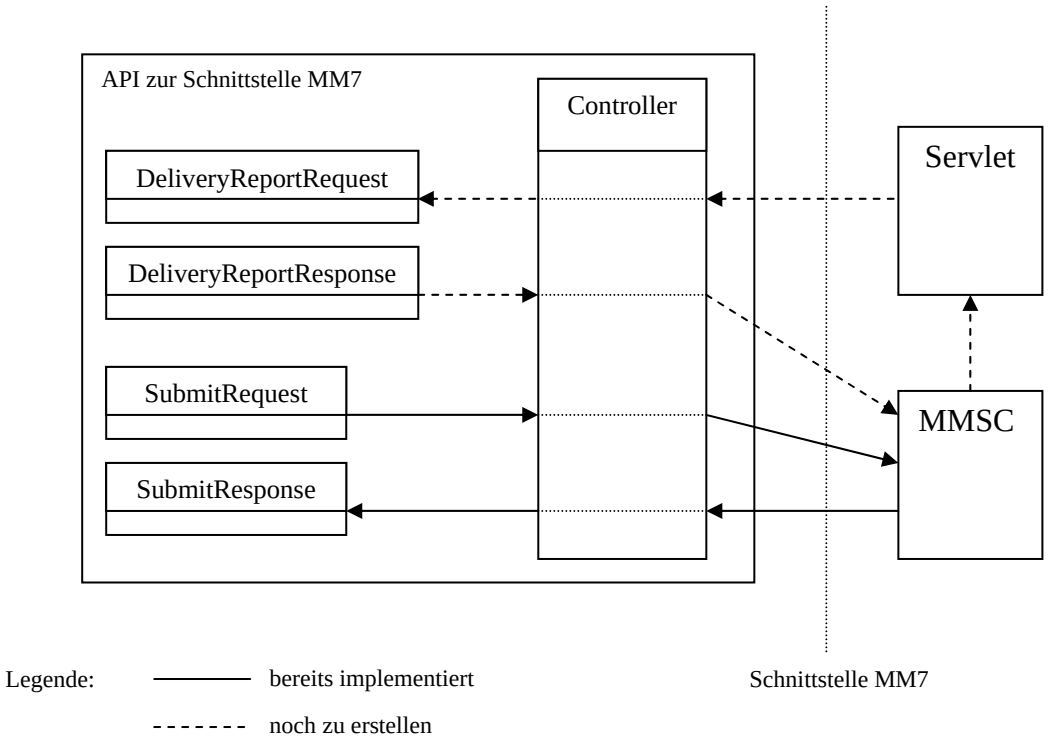
5.1. Idee und Aufbau

Die erstellte API zur Schnittstelle MM7 ist in drei Klassen eingeteilt, welche jede eine klar zugeordnete Funktion übernehmen. Da die gesamte Kommunikation bei MM7 in einzelne Nachrichten aufgeteilt ist, haben wir diese Idee übernommen und für jeden Nachrichten-Typ eine eigene Klasse geschrieben. Im Wesentlichen stellen diese Klassen Methoden zur Verfügung, um die Werte der Nachrichten-Elemente zu setzten oder auszulesen. Die realisierten Klassen sind SubmitRequest und SubmitResponse, welche die entsprechenden Nachrichten MM7_submit.REQ und MM7_submit.RES bearbeiten. Dazu kommt noch die Klasse Controller, welche für den Aufbau der Verbindung zum MMS-Center verantwortlich ist und die Versendung der Nachrichten übernimmt. Zur gesamten Schnittstelle gehören noch weitere Klassen, welche die Nachrichten MM7_delivery_report.REQ und MM7_delivery_report.RES bearbeiten. Auch alle weiteren Nachrichten, welche in der Schittstellendefinition gemäss der Organisation 3GPP beschrieben sind, lassen sich nach dem gleichen Prinzip verarbeiten. Leider standen uns bei der gegebenen Infrastruktur alle Funktionen dieser Nachrichten-Typen nicht zur Verfügung, deshalb haben wir uns hauptsächlich auf die drei implementierten Klassen konzentriert.

5.2. Verwendung der Klassen und deren Methoden

Die Beschreibung der einzelnen Klassen und deren Methoden, liegen diesem Dokument als Javadoc in elektronischer Form bei.

5.3. Grafik zur Idee und Aufbau der API



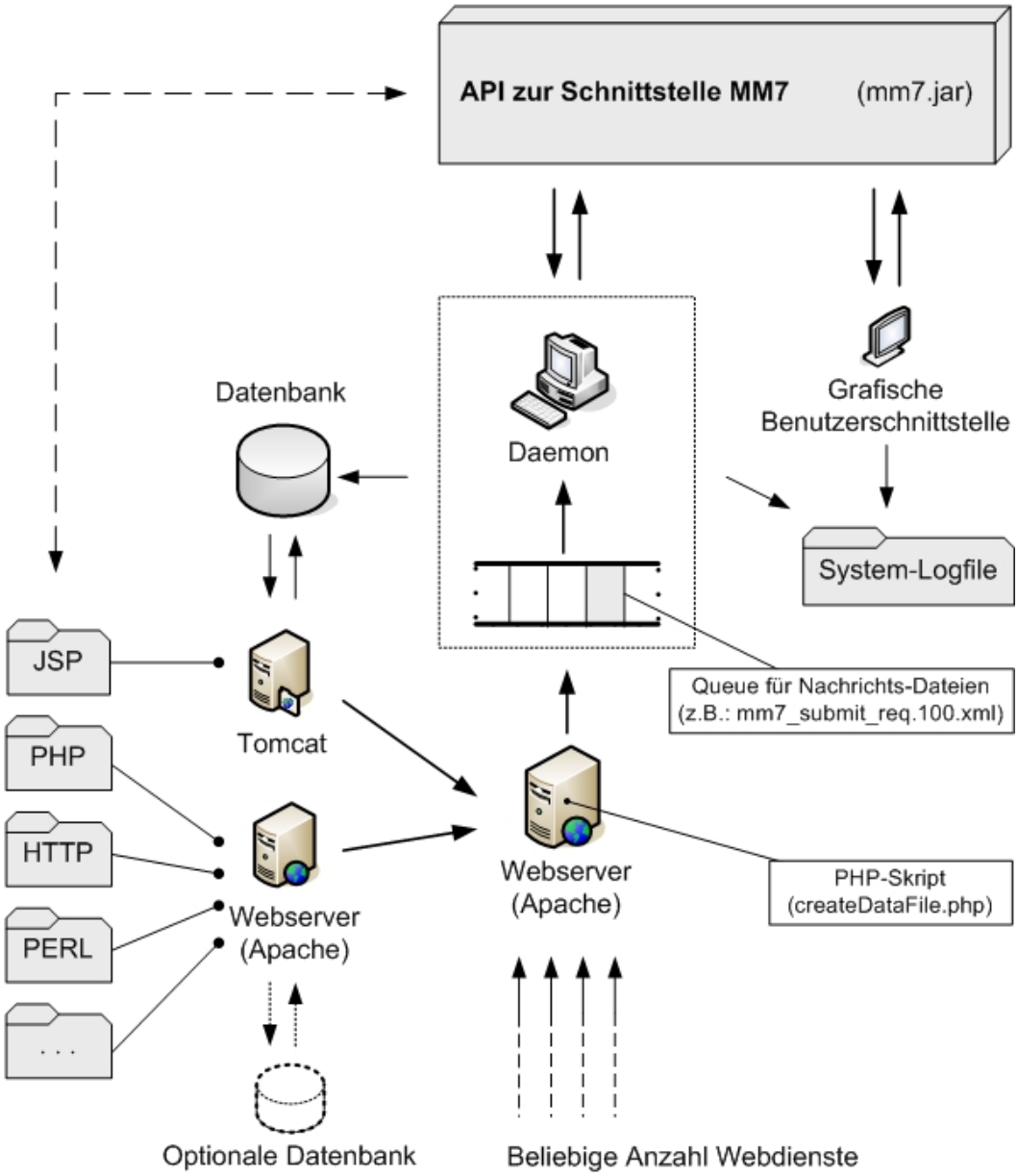
6. Mögliches Anwendungskonzept

6.1. Realisiertes System

Die folgende Grafik zeigt das realisierte verteilte System um von einer beliebigen Webseite oder einer lokalen Applikation eine Multimedia-Nachricht (MMS) zu versenden. Zu beachten ist, dass die dargestellten Geräte logische Einheiten sind und nicht unbedingt physikalischen Einheiten entsprechen müssen. Das System besteht aus den einzelnen Teilen:

- Implementation zur Schnittstelle MM7 (in Form einer API[2])
- Daemon-Tool
- Lokale Applikation mit grafischer Benutzschnittstelle
- MySQL-Datenbank
- Webserver mit PHP-Skript zur Erstellung von temporären Datenspeicher (XML-Datei)
- Webserver Apache und Tomcat
- JSP- und HTML/PHP-Webseiten für die zur Verfügungsstellung des Dienstes an den Endbenutzer.

[2] API: Application Programming Interface



6.2. Systembeschreibung

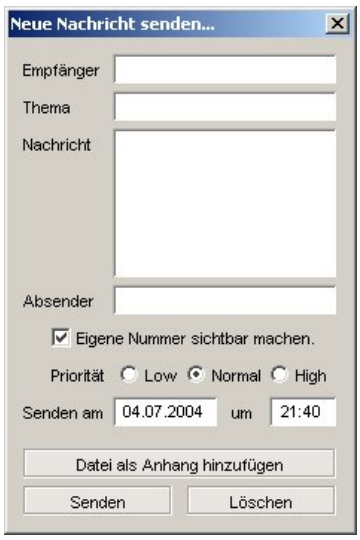
Ein Endbenutzer hat die Möglichkeit ein Formular auszufüllen, das über eine Webseite zur Verfügung gestellt wird, um eine MMS-Nachricht zu versenden. Die erwähnte Webseite kann in einer beliebigen Sprache realisiert sein und wurde zu Demonstrationszwecken als JSP- und PHP-Seite[3] implementiert. Die Webseiten senden die eingegebenen Formulardaten an ein PHP-Skript, das auf einem zentralen Webserver gespeichert ist. Dieses Skript erstellt aus den empfangenen Daten eine temporäre XML-Datei mit vorgegebener Struktur und speichert diese in einem Ordner mit vordefiniertem Pfad ab. Dieser Ordner dient dem System als Reader/Writer-Queue damit zeitliche Aspekte keine Rolle spie-

[3] JSP (Java Server Page) und PHP (PHP: Hypertext Preprocessor) sind beides serverseitige Skriptsprachen zur Erstellung von dynamischen Webseiten.

len. In diesem System spielt das erwähnte PHP-Skript den Writer und das nachfolgend beschriebene Daemon-Tool den Reader.
Auf dem gleichen Rechner, auf welchem der zentrale Webserver mit PHP-Skript läuft, muss auch das Daemon-Tool laufen, um die erstellten XML-Dateien zu verarbeiten. Das Daemon-Tool schaut in einem regelmässigen zeitlichen Abstand ob eine neue Datei vorhanden ist. Solange sich Dateien in der Queue befinden, werden diese nacheinander ausgelesen und die Daten verarbeitet. Nach einer erfolgreichen Verarbeitung werden die ausgelesenen Dateien gelöscht. Die zu versendenden Daten werden über die API des mm7.jar in eine SOAP-Nachricht verpackt, welche danach an das MMS-Gateway (MMS-Center) eines Telekommunikationsdienste-Anbieters gesendet wird.
Falls ein Anbieter dieses MMS-Systems die Möglichkeit einer Nachrichten-History zur Verfügung stellen will, kann er dies über seine eigene Webapplikation tun, falls diese an eine Datenbank angebunden ist, oder er kann diese Aufgabe im Daemon-Tool implementieren. In der vorliegenden Realisierung erstellt die JSP-Webapplikation einen neuen Eintrag in der Datenbank. Falls die Nachricht vom Daemon-Tool erfolgreich abgesetzt werden konnte und es eine Empfangsbestätigung des MMS-Center erhalten hat, wird der Nachrichtenstatus des entsprechenden Eintrages in der Datenbank aktualisiert. Dabei wird die eindeutige Kennzeichnung der Transaktion (Transaction-ID) verglichen um den richtigen Datensatz auszuwählen. Unabhängig von der Verwendung einer Datenbank dokumentiert das Daemon-Tool in einem System-Logfile ob eine Meldung abgesetzt oder eine Bestätigung empfangen wurde, sowie allfällige Fehlermeldungen.
Parallel zur Verwendung des Daemons in Verbindung einer Webseite, kann auch eine Applikation mit grafischer Benutzeroberfläche verwendet werden um eine MMS-Nachricht zu versenden. Die erstellte Applikation bietet die gleichen Funktionen wobei sie direkt auf die API zugreift und nicht zuerst über das Daemon-Tool. Eine weitere Möglichkeit wäre unter anderen die direkte Einbindung des Paketes mm7.jar in die JSP-Webseite.

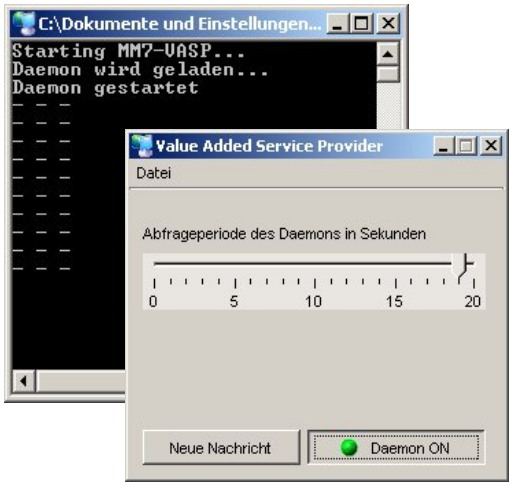
6.3. Benutzerschnittstellen zum System

6.3.1. MMS-Nachricht verfassen einem eigenständigen Tool



Eine Möglichkeit könnte sein eine gewünschte MMS-Nachricht per grafischem Tool zu verfassen. Verschiedene Parameter, allerdings in der gelieferten Version nicht alle, können gesetzt werden. Zusätzlich zu einem Text kann ein Bild, das über einen FileDialog ausgewählt wird, mitgeschickt werden. Dieses Tool verwendet die direkt eingebundene API MM7.jar.

6.3.2. Versenden einer MMS mit Hilfe des Daemon-Tools

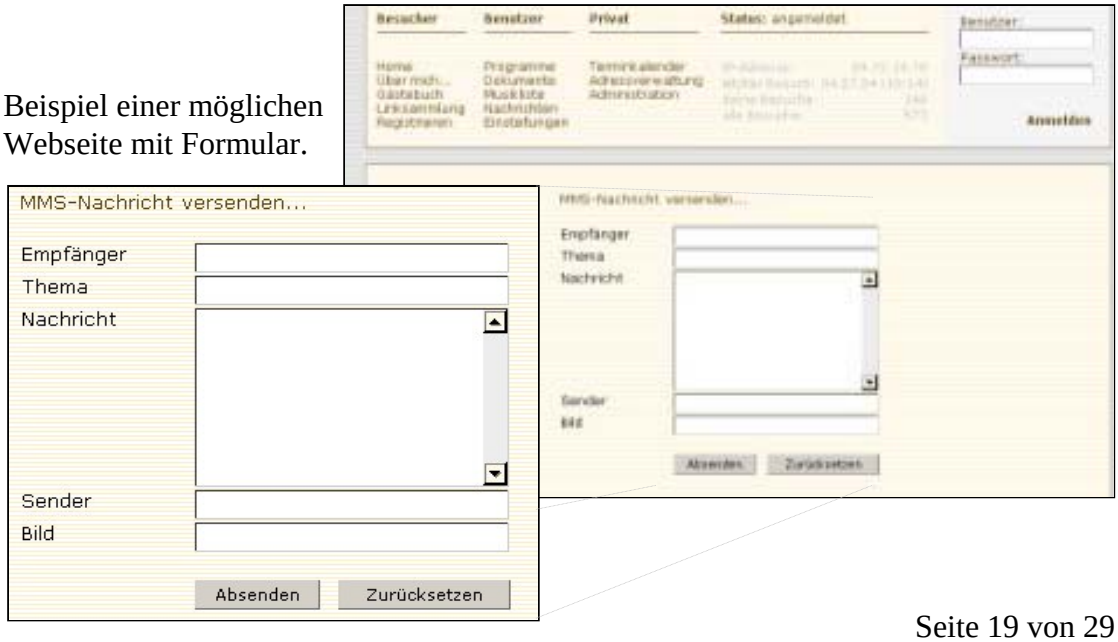


Dieses Daemon-Tool liest die Dateien aus, welche von einem Skript erstellt werden. Das erwähnte Skript kann von einer beliebigen Webseite aufgerufen werden. Die Daten werden mittels HTTP-POST aus einem HTML-Formular übergeben. Wichtig ist dabei, dass die verwendeten Feldnamen (Variablen) des Formulars und des Skriptes übereinstimmen. Der Daemon überprüft periodisch in einem angegebenen Verzeichnis, ob sich dort neu generierte XML-Dateien befinden. Nachdem auslesen wird die Datei gelöscht.

Dem erstellten PHP-Skript können folgende Variablen übergeben werden.

Variablenname	**Gesetztes Element**
senderVis	Sender-Visibility
sender	SenderAddress
empfaenger	Recipients
class	Message-Class
expiryDate	Expiry-Date
delivTime	Earliest-Delivery-Time
delivReport	Delivery-Report
readReply	Read-Reply
priority	Priority
thema	Subject
text	Textnachricht als Attachment
linkToPic	Bild als Attachment (URL)
transId	Transaction-ID

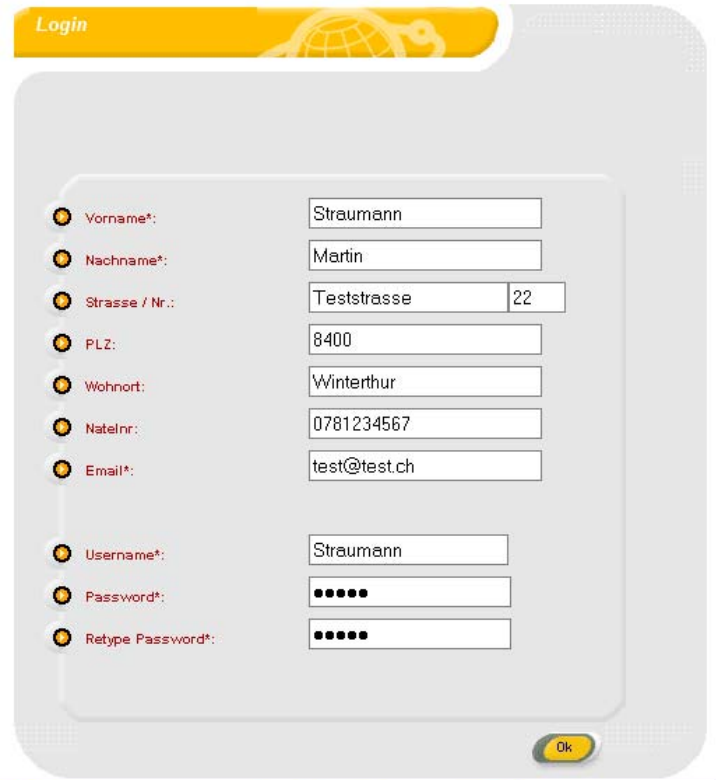
Beispiel einer möglichen Webseite mit Formular.



6.3.3. JSP-Webapplikation zum versenden von MMS-Nachrichten

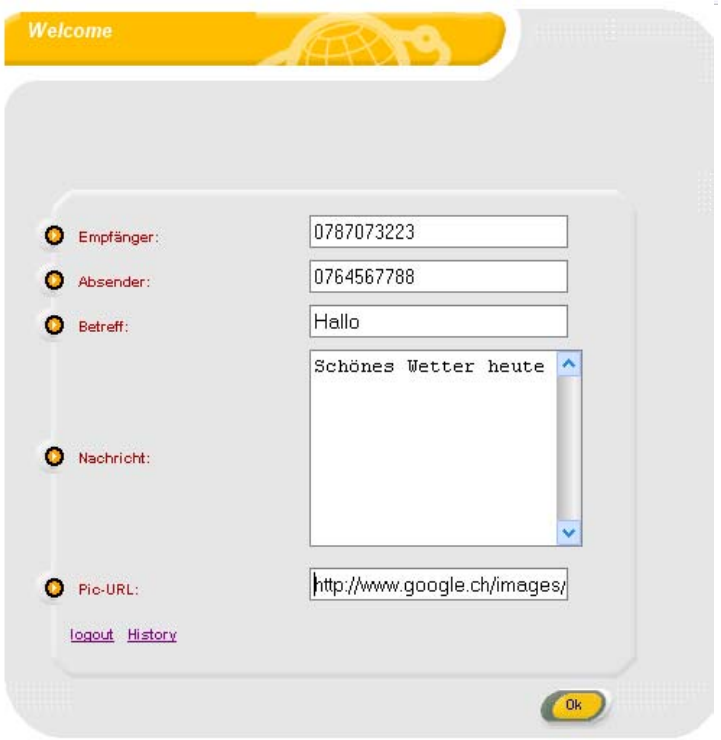
Die erstellte JSP-Webapplikation ist ein drittes mögliches Beispiel, wie ein MMS-Dienst benutzt werden kann. Diese Testseite ist folgendermassen aufgebaut und enthält die wichtigsten Funktionen wie Benutzerregistration, Anmeldung, Übersicht über gesendete Nachrichten.

Der Benutzer kann sich via eine Loginseite anmelden, um auf die Seite zu gelangen, von der er eine MMS-Nachricht versenden kann. Falls der Benutzer nicht registriert ist, so kann er sich über eine Seite der Benutzerverwaltung registrieren.



Die Registrierungsdaten werden in einer MySQL Datenbank gespeichert. Neben den Benutzdaten werden in einer eigenen Relation auch die versendeten Nachrichten gespeichert.

Nach dem erfolgreichem Anmelden wird der Benutzer auf die Seite zur Versendung von MMS weitergeleitet.



Der Benutzer hat wiederum die Möglichkeit ein Formular mit den gewünschten Daten der MMS-Nachricht auszufüllen.
Beim Versenden der Nachricht an ein Skript können die Daten gleichzeitig in die Datenbank übernommen werden.

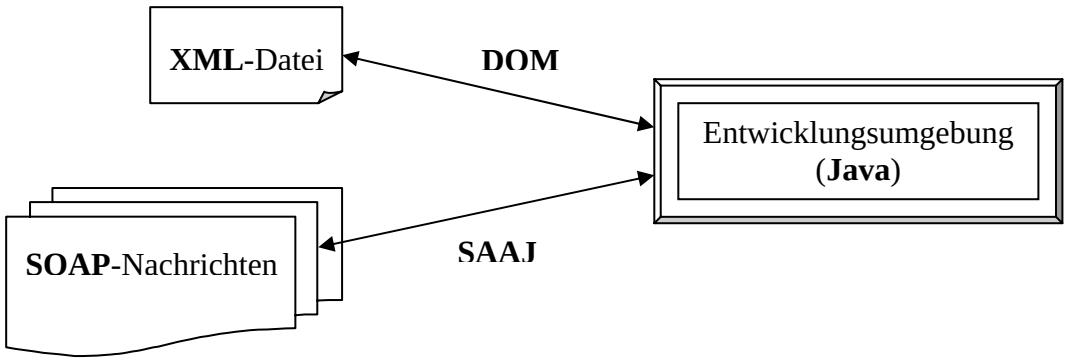
Eine sehr einfache Seite für die Übersicht der versendeten Nachrichten könnte in etwa wie folgt aussehen. Der Status gibt an, ob die MMS-Nachricht bereits gesendet wurde oder sich noch in Bearbeitung befindet. Weitere mögliche Statusänderungen wären möglich sobald der User Agent die Nachricht erhalten hat und eine gegebenenfalls angeforderte Lesebestätigung verschickt hat.

HISTORY

Sender	Emfänger	Thema	Text	Status
0765454545	0787073223	Hallo	Schönes Wetter heute	in bearbeitung

BACK

7. Überblick über die eingesetzten Technologien und Pakete



7.1. XML – Extensible Markup Language

XML ist ein von W3C eingeführter Standard zur Dokumentenauszeichnung. Diese Sprache definiert eine generische Syntax, mit deren Hilfe sich Daten mit einfachen, vom Menschen lesbaren Tags auszeichnen lassen. In der vorliegenden Projektarbeit haben wir uns entschieden XML als Dokumentensyntax für die temporären Dateien einzusetzen, welche die zu sendenden Daten enthalten und vor ihrer Verarbeitung in einer Queue abgelegt werden.

Dazu ein mögliches Beispiel:

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<message type='MM7_sumit.REQ'>
  <Sender-Visibility>true</Sender-Visibility>
  <SenderAddress>0764458312</SenderAddress>
  <Recipients>0787483223</Recipients>
  <Message-Class>Normal</Message-Class>
  <Expiry-Date>2004-12-12T15:25</Expiry-Date>
  <Earliest-Delivery-Time>2004-11-11T12:04</Earliest-Delivery-Time>
  <Delivery-Report>true</Delivery-Report>
  <Read-Reply>false</Read-Reply>
  <Priority>Normal</Priority>
  <Subject>Nachricht</Subject>
  <Content>Hello World!</Content>
  <Attachment>http://www.urlToPicture.com</Attachment>
  <Transaction-ID>2458_20041211154789</Transaction-ID>
</message>
```

Diese temporären Dateien werden von einem Skript automatisch erstellt damit sie immer die gleiche Struktur haben. Ein immer gleicher Aufbau dieser Dateien ist die Voraussetzung um sie später maschinell auslesen zu können. Für das Arbeiten mit XML in Verbindung von Java wurden keine zusätzlichen Pakete benötigt.

Verweise: W3C http://www.w3.org/
XML http://www.w3.org/XML/

7.2. DOM – Document Object Model

Das Document Object Model ist eine sprachneutrale baumorientierte API, die ein XML-Dokument als Menge verschachtelter Objekte mit verschiedenen Eigenschaften behandelt. Wir haben DOM eingesetzt als Parser um die erstellten XML-Dateien auszulesen. Um mit DOM arbeiten zu können muss mit dem DocumentBuilder die gesamte XML-Datei eingelesen und als Document im Speicher abgelegt werden. Danach sind alle Objekte (Nodes) erreichbar.

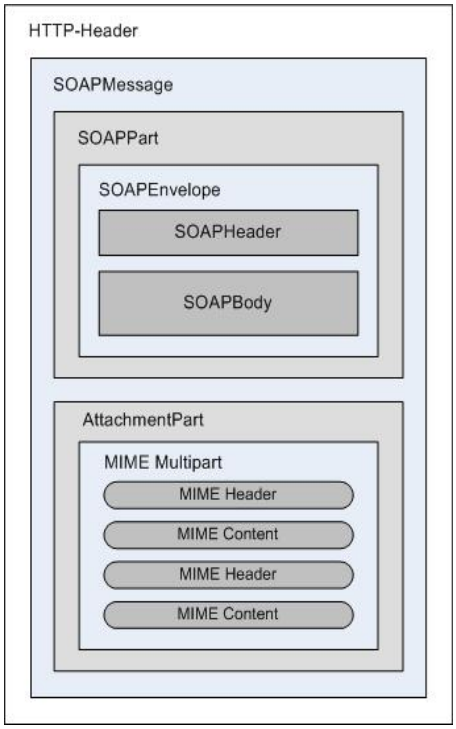
```
DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
DocumentBuilder docBuilder = dbf.newDocumentBuilder();

Document doc = docBuilder.parse(dataSource);
Element root = doc.getDocumentElement();
NodeList children = root.getChildNodes();

String sender = doc.getElementsByTagName("SenderAddress")
                   .item(0).getFirstChild().getNodeValue());
```

Die wichtigste Funktion ist dabei getElementsByTagName, welche eine Liste liefert mit allen Elementen, welche den angegeben Namen haben. Da es in der Struktur immer nur ein Element mit dem gleichen Namen gibt, konnten wir immer den ersten Eintrag dieser Liste verarbeiten. Um DOM in unserer Entwicklungsumgebung zu verwenden, waren die zusätzlichen Pakete *dom.jar* und *xercesImpl.jar* notwendig.

Verweise: DOM http://www.w3.org/DOM/
Xerces http://xml.apache.org/xerces2-j/index.html

7.3. SOAP – Simple Object Access Protocol

SOAP definiert ein Standard-Format für den Transport von XML-Daten via HTTP, SMTP und FTP für den Zugriff auf Web-Services. SOAP ist demnach ein Kommunikationsprotokoll und basiert auf der XML-Struktur. Es ist zu Vergleichen mit RPC, RMI, Corba usw. und wird verwendet um Informationen zwischen Anwendungen auszutauschen. In dem vorliegenden Projekt werden die SOAP-Nachrichten zwischen den Anwendungen MMSC und VASP ausgetauscht. Die nachfolgende Grafik zeigt den Aufbau einer SOAP-Nachricht, wie sie in der vorliegenden Projektarbeit verwendet wurde.



Header:

Im Header befinden sich nur Daten, die relevant sind für die Übertragung der Nachricht.

Body:

Der Body beinhaltet alle Daten welche später in eine MMS-Nachricht abgefüllt werden.

AttachmentPart:

Dieser Teil beinhaltet genau ein Attachment. Dies kann zum Beispiel ein Text, Bild oder ein Multipart-Objekt sein. Nur wenn ein Multipart-Objekt verwendet wird können mehrere Attachments verschickt werden.

Für einen einfachen und direkten Zugriff auf SOAP-Nachrichten setzten wir die Programmierschnittstelle SAAJ ein und benötigten deshalb die Pakete *saaj-impl.jar* und *saaj-api.jar*. Um mehrere Attachments versenden zu können mussten wir ein MIME-Objekt erstellen und benötigten somit noch die Pakete *mail.jar* und *activation.jar*

Verweise:

SOAP http://www.w3schools.com/soap/default.asp
MIME http://java.sun.com/j2ee/sdk_1.3/techdocs/api/javax/mail/internet/package-summary.html

7.4. SAAJ – SOAP with Attachments API for Java

Wie bereits erwähnt ist SAAJ eine Programmierschnittstelle für SOAP-Nachrichten, welche Sun in dem Java Web Service Developer Pack anbietet.

Verweise:

SAAJ Javadoc http://java.sun.com/j2ee/1.4/docs/api/javax/xml/soap/package-summary.html
SAAJ Tutorial http://java.sun.com/webservices/docs/1.3/tutorial/doc/index.html

Bemerkung:

Für die erstellte Programmierschnittstelle API zu MM7 werden nur die Pakete saaj-api.jar, saaj-impl.jar, mail.jar und activation.jar benötigt. Alle weiteren werden von dem erstellten verteilten System (Gesamtapplikation) zusätzlich benötigt, wie zum Beispiel das Paket mysql-connector-java-2.0.14-bin.jar für den Zugriff auf eine MySQL Datenbank über JDBC.

8. Projektplanung und Organisation

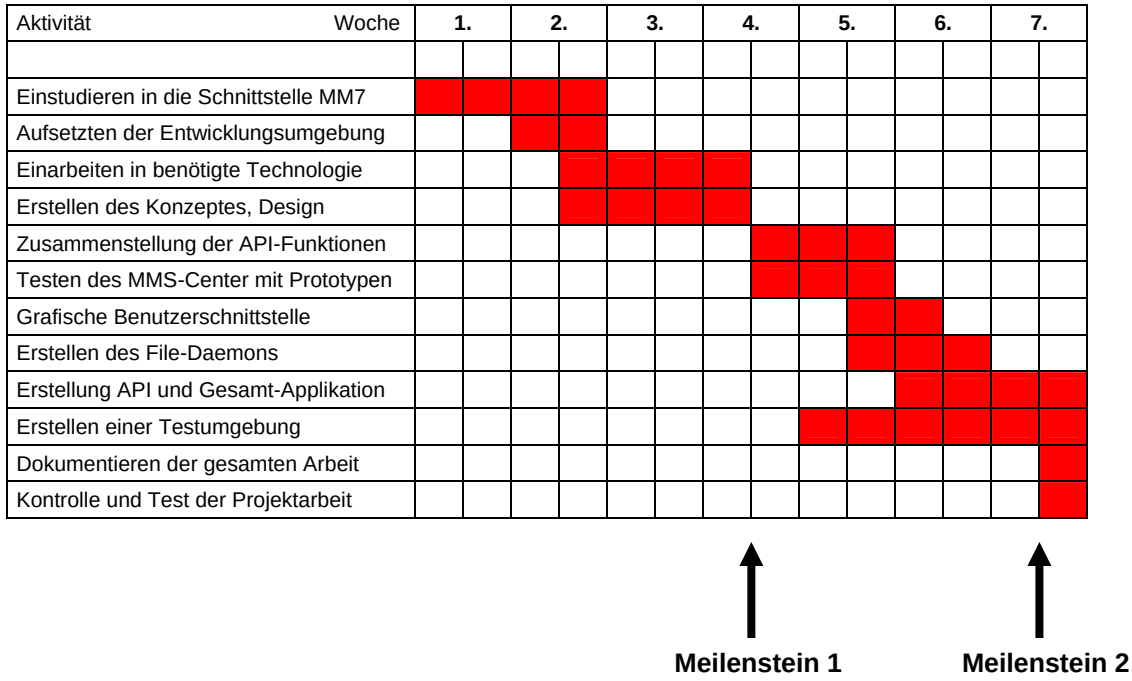
8.1. Vorgehen und Planung

Als erstes ging es darum herauszufinden was als Ziel dieser Projektarbeit überhaupt verlangt wurde. Dies wurde uns von Seiten der Auftraggeber sehr gut erklärt. Auf Grund der bereits erwähnten Probleme betreffend Sunrise gab es immer wieder Änderungen in der zeitlichen Planung. Da wir mit den eigentlichen Projektaufgaben aus technischen Gründen erst Mitte der vierten Projektwoche beginnen konnten, wurde die Zeit am Ende extrem knapp. Jedoch konnten alle verlangten Aufgaben gemäss Fabio Vena erfolgreich erfüllt werden.

Unter anderen gab es die folgenden Aufgaben zu erledigen.

- Einstudieren in die Schnittstelle MM7
- Aufsetzten der benötigten Entwicklungsumgebung (JCreator, Tomcat, etc.)
- Einarbeiten in benötigte Technologien
- Erstellen des Konzeptes, Design
- Zusammenstellung der Funktionen der API
- Testen des MMS-Center mit Prototype
- Erstellen des Daemon-Tools
- Erstellen der Programmierschnittstelle (API)
- Erstellen einer Demoapplikation um die Logik zu testen und um die verfügbaren Optionen und Möglichkeiten zu präsentieren.
- Dokumentieren der gesamten Arbeit
- Kontrolle und Test der gesamten Projektarbeit

8.2. Ganttdiagramm



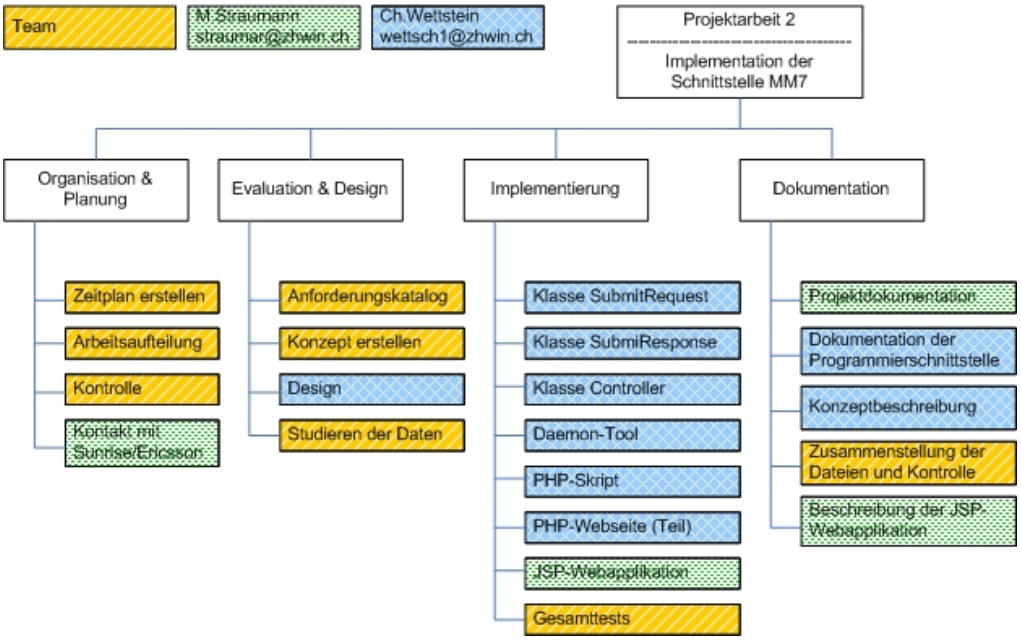
1. Meilenstein:

   Unser Account für den Zugriff auf das Sunrise-MMSC wurde am 09.06.04 aufgeschaltet. Somit konnten wir von unseren Arbeitsstationen ein erstes mal eine Verbindung aufbauen, die nicht von der Firewall geblockt wurde.

2. Meilenstein:

   Das MMS-Center unterstützt keine SOAP-Nachricht mit mehreren Attachments gemäss SOAP-Anleitung. Es musste ein MimeMultipart-Container abgefüllt werden, um mehrere Attachments versenden zu können. Eine erste erfolgreiche Versendung von mehreren Bildern und Texten fand am 30.06.04 statt. Zusammen mit Fabio Vena konnte nach vielen erfolglosen Tests eine Lösung gefunden werden.

8.3. Arbeitsaufteilung und Zuständigkeitsbereiche



8.4. Informationsbeschaffung und Kontakte

FutureLab

Die Zusammenarbeit mit der Firma FutureLAB verlief ohne Probleme. Es wurde fast jede Woche eine Sitzung abgehalten um den aktuellen Stand zu besprechen. Ein gelieferter HTTP-Dump sowie die Unterstützung von Fabio Vena waren sehr hilfreich.

Sunrise

Die Kontaktperson bei der Firma Sunrise (Herr Kolakovic) war sehr selten erreichbar und deshalb keine Hilfe bei Fragen betreffend des gemieteten MMS-Gateway. Nach vielen Telefonanten konnten andere Kontaktpersonen ermittelt werden. Es dauerte dennoch sehr lange bis wir weiterführende Unterlagen über die implementierte MM7-Schnittstellen bekamen.

Wichtige Kontaktadressen von Sunrise:

Herr Marc Blöchliger (Product Manager)
E-Mail: marc.bloechliger@sunrise.net
Tel: +41 76 300 93 08

Herr Dore Satisch (Administrator)
E-Mail: satisch.dore@sunrise.net
Tel: +41 76 300 89 19

Ericsson

Bei Ericsson konnten wir keine Informationen bekommen. Es wurden zwar viele Telefonate geführt und Informationen versprochen, allerdings war die zuständige Kontaktperson nie erreichbar. Eine direkte Telefonnummer konnte uns nicht genannt werden. Auch wurde keine einzige E-Mail Anfrage beantwortet.

Herr Furrer
Tel: + 41 1 807 22 22 (Hauptnummer Ericsson Schweiz)

9. Entwicklungsumgebung

Zur Verfügung standen zwei Rechner aus dem ZHW-Schulnetz. Die Arbeiten konnten nur mit diesen beiden Rechnern ausgeführt werden, da deren IP-Adressen bei der Firewall der Firma Sunrise registriert werden mussten.

Die kompilierten Dateien wurden für die Windows-Betriebssysteme kompiliert, jedoch steht der Source-Code von allen benötigten Teilen zur Verfügung damit auch andere Betriebssysteme unterstützt werden könnten. Dies war ein wesentlicher Grund weshalb das ganze Projekt in Java realisiert wurde.

Arbeitsstation 1	Betriebssystem:	Windows XP SP2
	IP-Adresse:	160.85.170.138
	Java SDK:	j2sdk1.4.2_04
	Java Webservices:	jwsdp-1.3
	Java-Editor:	JCreator 2.5 Pro
Arbeitsstation 2	Betriebssystem:	Windows XP SP2
	IP-Adresse:	160.85.170.139
	Java SDK:	j2sdk1.4.2_04
	Java-Editor:	JCreator 2.5 Pro
MySQL-Datenbank	Version 4.0.18	
Apache Webserver	Version 2.0.49 (Win32) PHP/4.3.7	
Tomcat-Webserver	Version 4.0	

10. Problembeschreibungen

10.1. Allgemeine Probleme

Eines der Hauptprobleme war die Zusammenarbeit mit der Firma Sunrise da sie uns lange keine Hilfreichen Informationen geben konnte. Auch dass sich der MMS-Gateway während der gesamten Projektzeit im Upgrade zu einer neuen Version befand hat die Arbeit erschwert. Jedoch sollten diese beiden Probleme für die Zukunft keine Schwierigkeiten mehr darstellen, denn per Anfangs Juli hat die neue Version des Gateways den Betrieb aufgenommen. Auch die zuvor fehlenden Funktionen seien gemäss Sunrise nun verfügbar.

10.2. Filterung des Netzwerkverkehrs über Port 8888

Sunrise sendet die Request-Nachrichten des MMSC immer an den Destination-Port 8888. Dieser wurde von der ZHW gefiltert und blockiert bevor er uns geöffnet wurde. Danach mussten wir leider feststellen und erfahren, dass diese Dienste noch gar nicht in Betrieb waren und wir deshalb unser vorbereitetes Servlet für die Entgegennahme nie testen konnten.

10.3. Direktes Einbinden des MM7.jar in JSP-Webapplikation

Grundsätzlich ist es möglich ein Jar-Paket direkt in eine JSP-Webapplikation einzubinden und zu verwenden. Dies stellte auch keine Probleme dar, jedoch wurden die gesamte Nachricht doppelt verpackt und löste deshalb beim MMS-Center den Fehler 2007 (Message Format Corrupt) aus.

11. Glosar

HTTP	Hypertext Transfer Protocol
MIME	Multipurpose Internet Mail Extensions
MM	Multimedia Message
MMS	Multimedia Messaging Service
MMSC	Multimedia Messaging Service Center
MMSE	Multimedia Messaging Service Environment
SAAJ	SOAP with Attachments API for Java
SOAP	Simple Object Access Protocol
VAS	Value Added Service
VASP	Value Added Service Provider
XML	Extensible Markup Language

12. Quellen- und Linkverzeichnis

Datenbanken und JDBC

- MySQL Datenbank http://dev.mysql.com/downloads/mysql/4.0.html
- MySQL JDBC-Driver http://www.mysql.com/products/connector/j/

Eingesetzte Software

- Tomcat Webserver 5.0 http://jakarta.apache.org/tomcat
- Apache Webserver http://www.apache.org
- JCreator 2.5 http://www.jcreator.com

Zusätzlich eingesetzte Java-Pakete

- Mail.jar http://java.sun.com/products/javamails/
- Activation.jar http://java.sun.com/products/javabeans/glasgow/jaf.html
- SAAJ-impl.jar http://java.sun.com/webservices/webservicespack.html
- SAAJ-api.jar http://java.sun.com/webservices/webservicespack.html
- Xerces.jar http://xml.apache.org/xerces2-j/index.html

Unterlagen und Tutorials

- JSP http://jva.sun.com/products/jsp/docs.html
- Tomcat http://jakarta.apache.org/tomcat/tomcat-5.0-doc
- JSP-Hilfe http://www.jsptut.com
- Java Web Service Developer Pack http://java.sun.com/webservices/webservicespack.html
- MimeMultipart-Klassen http://java.sun.com/j2ee/sdk_1.3/techdocs/api/javax/mail/Internet/package-summary.html