Protokollbeschreibung

AMBUS® Net
Web Service Integration mit SOAP

VD 9-906 d 01.2009

Inhaltsverzeichnis

1 Einführung

Dieses Dokument beschreibt die Integration des Web Service der M-Bus Zentrale AMBUS® Net mit SOAP. Es erhebt nicht den Anspruch eine Einführung in SOAP oder .Net zu sein.

1.1 Was ist ein Kommunikations-Protokoll

Folgender Fall zeigt am Beispiel einer Vater-Sohn-Kommunikation, um was es bei einem Kommunikationsprotokoll geht:

Vater

1. Was lest ihr denn gerade in der Schule?
3. Aber das ist doch nicht von Karl May, sondern von Karl Marx!

Sohn

2) Das Kapital von Karl May.
4) Ach deshalb! Wir sind schon auf Seite 200, und es kommen immer noch keine Indianer vor!

Auch Menschen verwenden zur Kommunikation unbewusst ein „Protokoll“. Damit eine Verständigung stattfindet, müssen gewisse Regeln eingehalten werden. So sollten beide die gleiche Sprache sprechen. Diese kulturbezogenen Regeln sind allerdings nicht niedergeschrieben, werden uns jedoch schon als Kinder beigebracht. Aus diesem Grund kann es auch dann zu Verständigungsproblemen zwischen zwei Personen führen, wenn sie die gleiche Sprache sprechen aber unterschiedlicher kultureller Herkunft sind.

Aus technischer Sicht legt ein Protokoll fest, zu welchem Zeitpunkt oder in welcher Reihenfolge welche Operation ausgeführt wird. Das Protokoll des Vater / Sohn Beispiels könnte man z.B. so darstellen:

→ ANFRAGE	(Was bearbeitet ihr den grade in der Schule?)
← ANTWORT	(Das Kapital von Karl May.)
→ KORREKTUR	(Aber das ist doch nicht von Karl May, sondern von Karl Marx!)
← BESTÄTIGEN	(Ach deshalb! Wir sind schon auf Seite 200, und noch keine Indianer.)

Mit einem Protokoll dieser Art kann man Informationen abfragen und diese wenn nötig korrigieren.

1.2 SOAP

1.2.1 Definition

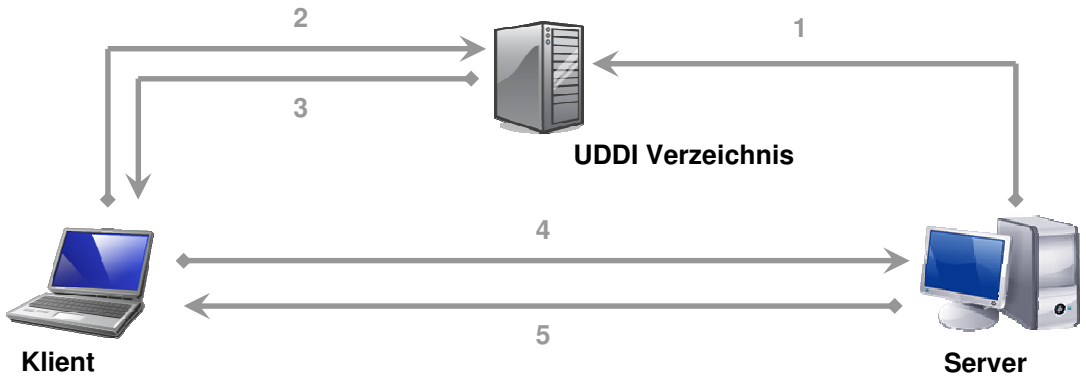
SOAP stellt ein Rahmenwerk zur Verfügung, welches erlaubt, dass beliebige applikationsspezifische Informationen zwischen zwei Rechnern übertragen werden können. Es stellt Regeln für das Nachrichtendesign auf, regelt wie Daten in der Nachricht abzubilden und zu interpretieren sind, und gibt eine Konvention für entfernte Prozeduraufrufe vor. SOAP bildet das Datenaustauschprotokoll eines Web Service[01].

1.2.2 Geschichte

Die Version 1.0 wurde Ende 1999 von Microsoft veröffentlicht. Im Jahr 2000 schloss sich IBM der Entwicklung an was zur Folge hatte dass die SOAP Version 1.1 beim World Wide Web Consortium (W3C) eingereicht wurde. Dieses Konsortium kümmert sich um die Internettechnologiestandardisierung. Die erste vom W3C veröffentlichte Version 1.2 wurde im Juni 2003 freigegeben.

1.2.3 Kommunikationsablauf

An einem klassischen Kommunikationsablauf sind mindestens zwei Komponenten beteiligt. Dies ist zum einen der Server welcher einen Web Service anbietet und der Klient welcher einen Web Service in Anspruch nimmt. Das UDDI (Universal Description, Discovery and Integration) Verzeichnis wird nur in einem Umfeld mit dynamischen Web Services benötigt. Ist dem Klient sowohl der Web Service als auch der Server bekannt, kann er eine direkte Verbindung erstellen. Somit kann Schritt 1 - 3 in der folgenden Abbildung weggelassen werden.



1. Der Server registriert den Web Service beim UDDI Verzeichnisdienst
2. Der Klient fragt den UDDI Verzeichnisdienst, wo sich sein gewünschter Web Service befindet.
3. Als Antwort erhält der Klient den Standort des Servers.
4. Darauf hin baut der Klient eine Verbindung zu dem nun bekannten Server auf und nutzt den Web Service.
5. Der Web Service seinerseits erledigt seine Arbeit und sendet die Antwort and den Klienten zurück.

1.2.4 Eigenschaften

Durch die Verwendung von textbasiertem XML[02] als Repräsentation der Daten ist SOAP sowohl unabhängig von der Plattform als auch von der Programmiersprache. Das bedeutet dass zum Beispiel der Klient auf einer .Net / Windows Plattform basieren kann und der Web Service auf einer Java / Linux Basis implementiert wurde. Ausserdem ist SOAP vom Protokoll unabhängig. Das heisst dass die Protokolle aus der Netzwerk-, Transport- und Anwendungsschicht (Siehe Protokollstapel) ausgetauscht werden können.

1.2.5 Vorteile

- SOAP kann mit geringem Aufwand in Standard-Netzwerken genutzt werden.
- Die Applikation, entscheidet was mit den Daten geschieht
- Sehr einfache und schnelle Integration eines Web Services (z.B. in einem .Net oder Java Umfeld)

1.2.6 Nachteile

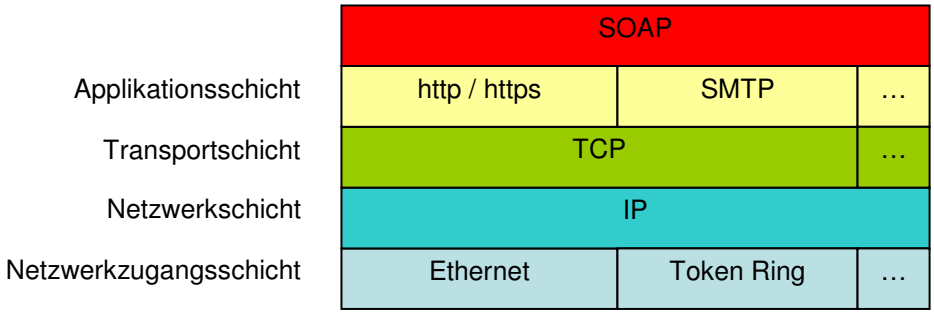
- XML erhöht die Menge der zu übertragenden Daten
- Dadurch ist SOAP/XML langsamer als binäre Netzwerkprotokolle

2 AMBUS® Net und SOAP

Dieses Kapitel richtet sich an Personen, die mit Netzwerkprotokollen vertraut sind und gewisse Vorkenntnisse haben betr. SOAP haben.

2.1 Protokollstapel

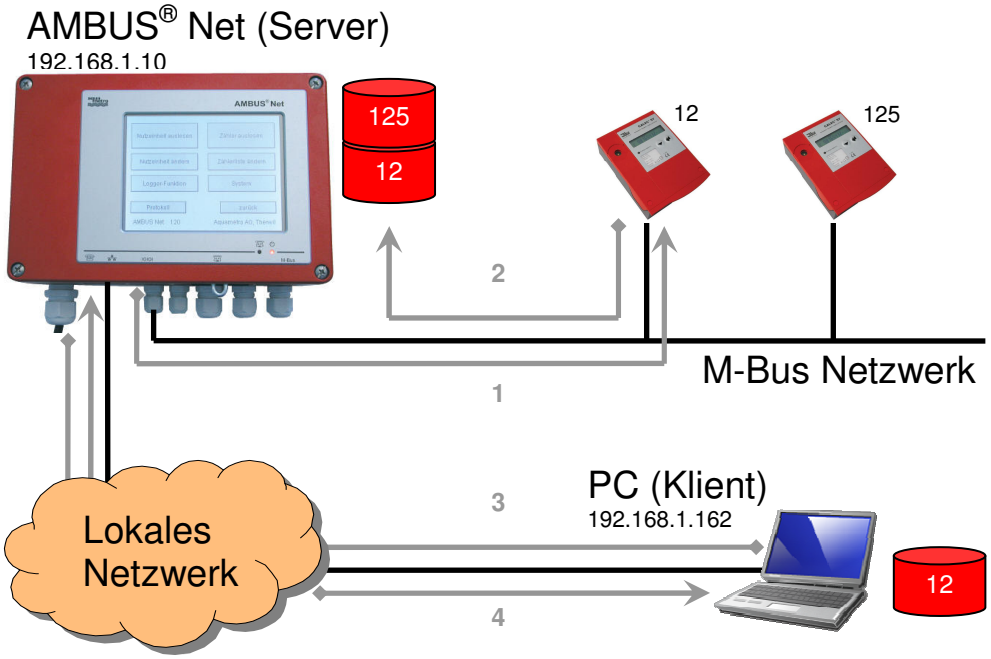
Die folgende Illustration zeigt SOAP im TCP/IP[03] Protokollstapel.



Der TCP/IP Protokollstapel ist auf die Internet-Protokolle zugeschnitten, welche den Datenaustausch über die Grenzen lokaler Netzwerke hinweg ermöglichen. Es wird weder der Zugriff auf ein Übertragungsmedium noch die Datenübertragungstechnik definiert. Vielmehr sind die Internet-Protokolle dafür zuständig, Datenpakete über mehrere Punkt - zu - Punkt - Verbindungen weiterzuvermitteln.

2.2 AMBUS® Net Web Service

Folgende Illustration zeigt an einem Beispiel, wie die Kommunikation in einer AMBUS® Net Anlage abläuft.



1. AMBUS® Net liest in regelmässigen Intervallen die M-Bus Anlage aus. Das Intervall kann im Logger Modul festgelegt werden.
2. Die von den Zählern erhaltenen Daten werden abgelegt.
3. Der Klient ruft „getMeter(12)“ des Web Services von AMBUS® Net auf, um die Zählerdaten anzufordern.
4. AMBUS® Net schickt die angeforderten Daten über SOAP zum Klient zurück.

SOAP übernimmt die Konvertierung und den Transport der Daten zwischen Klient und Server.

3 AMBUS® Net Integration mit SOAP

Dieses Kapitel richtet sich an Softwareentwickler, die den Web Service von AMBUS® Net nutzen möchten. Die folgenden Codeausschnitte sind allesamt in der .Net Programmiersprache C# 2.0 geschrieben.

3.1 Allgemeine Erläuterung zum Web Service

AMBUS® Net stellt einen Web Service mit zwei Funktionen zur Verfügung:

`Meter getMeter(byte PrimAddress)`	
`PrimAddress`	Die Primäradresse des Zählers
`Meter`	Rückgabewert ist der Zähler mit der Primäradresse „PrimAddress“

`UsageUnit getUsageUnit(int UnitNr)`	
`UnitNr`	Die Indexnummer der Nutzeinheit
`UsageUnit`	Rückgabewert ist die Nutzeinheit mit der Indexnummer „UnitNr“

3.2 Eigenschaften eines Zählers (Meter)

Als Rückgabewert der Funktion „getMeter(…)“ erhält man eine Instanz der Klasse Meter. Eine Instanz dieser Klasse ist wie folgt aufgebaut:

```
public class Meter
{
      public byte    primAddr;
      public string  deviceName;
      public string  desc;
      public string  type;
      public string  info;
      public string  supplier;
      public string  serial;
      public string  medium;
      public string  instPoint;
      public string  readout;
      public string  status;
      public Display opTime;
      public Display errorTime;
      public Channel channel_1;
      public Channel channel_2;
      public Channel channel_3;

      public Meter()    {     }
}
```

```
public class Channel
{
      public Display M1;
      public Display M2;
      public Display M3;
      public Display M4;
      public Display P;
      public Display Q;
      public Display Th;
      public Display Tc;
      public Display dT;

      public Channel()  {     }
}
```

```
public class Display
{
      public string value;
      public string unit;

      public Display()  {     }
}
```

3.3 Eigenschaften einer Nutzeinheit (UsageUnit)

Als Rückgabewert der Funktion „getUsageUnit(…)“ erhält man eine Instanz der Klasse UsageUnit. Eine Instanz dieser Klasse ist wie folgt aufgebaut:

```
public class UsageUnit
{
      public int     index;
      public string  name;
      public string  readout;
      public string  deviceName;
      public int     size;
      public Entry[] entries;

      public UsageUnit()      {     }
}
```

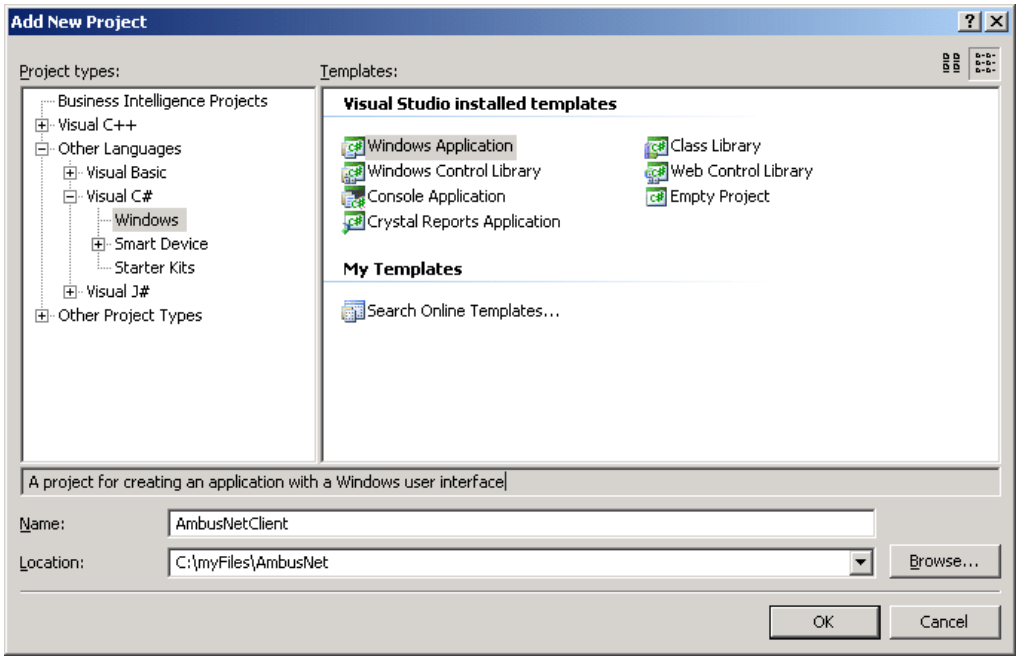
```
public class Entry
{
      public byte primAddr;
      public byte channelNr;
      public byte meterNr;
      public string medium;

      public Entry()    {     }
}
```

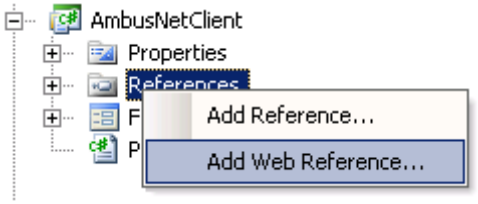
3.4 Einbinden des Web Services in ein C# .Net Projekt

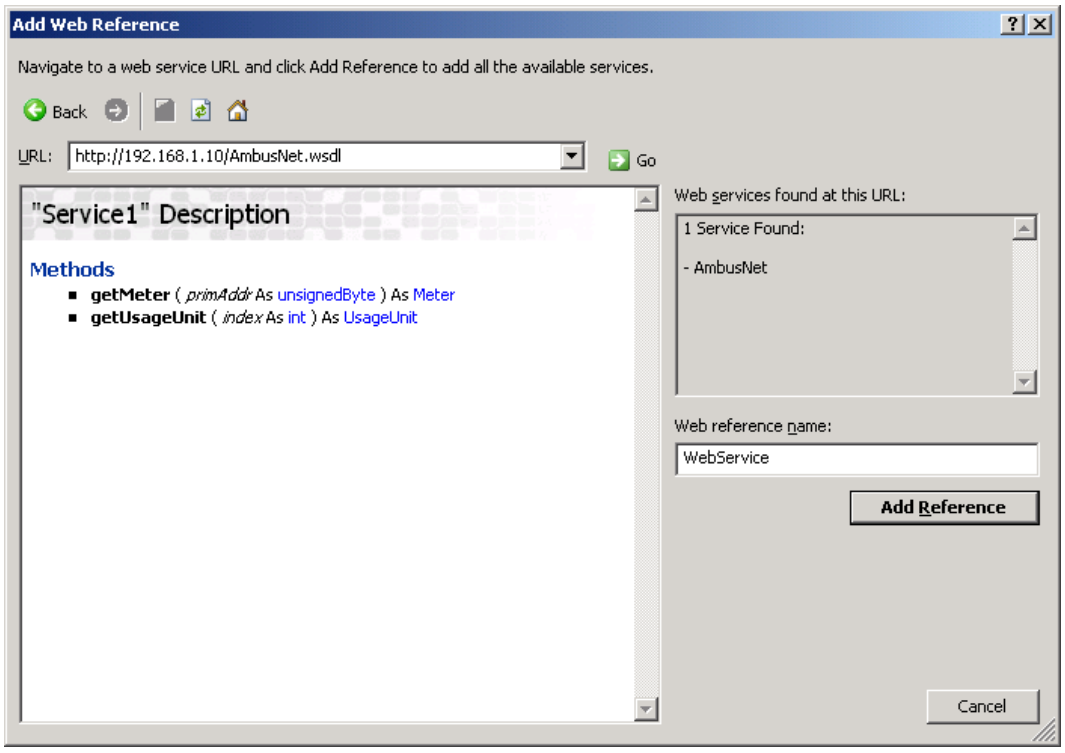
Der Web Service von AMBUS® Net kann in vier Schritten in ein C# Projekt eingebunden werden.

1. Ein neues C# Projekt wird mit MS Visual Studio 2005 .Net angelegt.



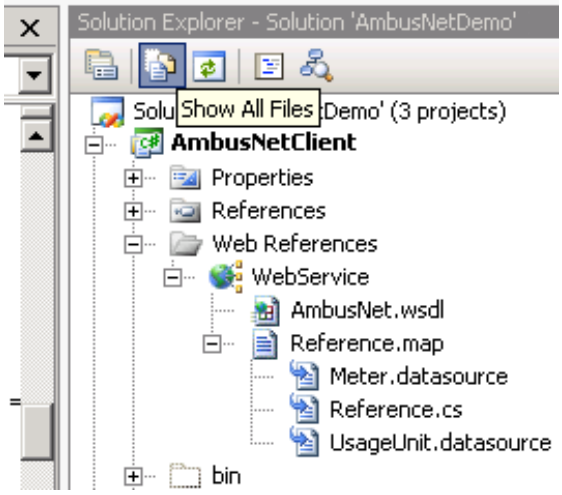
2. Eine Web-Referenz wird mit Hilfe der Datei „AmbusNet.wsdl“[04] erstellt. Diese Datei befindet sich auf AMBUS® Net und beschreibt den Web Service. Der Name der Referenz kann frei gewählt werden, muss jedoch im Code verwendet werden. In diesem Beispiel hat AMBUS® Net die IP Adresse „192.168.1.10“.





In diesem Dialog sehen wir dass der Web Service welcher AMBUS® Net zur Verfügung stellt „Service1“ heisst und die beiden Funktionen „getMeter“ und „getUsageUnit“ beinhaltet. Nach betätigen des „Add Reference“ Knopfs werden im Hintergrund die Klassen und Funktionen automatisch generiert um den Web Service zu nutzen. Dies ist möglich da die WSDL Datei eine genaue Beschreibung der Funktionen, Parameter und Datentypen beinhaltet.

3. Das erstellen einer Web Referenz erzeugt im Projektordner im Verzeichnis „Web References“ ein Unterverzeichnis mit dem Namen der Referenz. In diesem Verzeichnis befindet sich unter anderem die automatisch generierte Datei „Reference.cs“.



Diese Dateien kann man sichtbar machen indem man den Knopf „Show All Files“ drückt. Als nächstes müssen wir eine kleine Modifikation des automatisch generierten Codes vornehmen. Durch die Umstellung vom .Net Framework 1.1 auf das .Net Framework 2.0 hat sich das Standartverhalten eines Web Service Klient verändert. Die folgenden eingerahmten Zeilen Code müssen in der Datei „Reference.cs“ eingefügt werden, damit die Kommunikation problemlos klappt.

```
public partial class Service1 : SoapHttpClientProtocol {
  ...
  protected override System.Net.WebRequest GetWebRequest(Uri uri)
  {
    System.Net.HttpWebRequest webRequest =
                      (System.Net.HttpWebRequest)base.GetWebRequest(uri);
    webRequest.KeepAlive = false;
    return webRequest;
  }
  ...
}
```

4. Als nächstes können wir den Web Service in unserem Projekt einbinden. In diesem Beispiel heisst die Web Referenz „WebService“. Wenn Sie einen anderen Namen gewählt haben, muss dieser im Code verwendet werden. Ihrer Fantasie sind bezüglich der Einsatzmöglichkeiten keine Grenzen gesetzt. Weiter Beispiele finden Sie im Kapitel Anwendungsmöglichkeiten.
5.

```
using System;
using System.Net;
using AmbusNetClient.WebService;
...
namespace AmbusNetClient
{
  // reference of the AMBUS® Net web service instance
  private Service1 ambus = null;
  ...
  public class AmbusNet : System.Windows.Forms.Form
  {
    ...
    public AmbusNet()
    {
      InitializeComponent();
      ambus    = new Service1(); //initialises the AMBUS® Net web service
      ambus.Proxy = WebRequest.DefaultWebProxy; //iexplorer proxy or
      // ambus.Proxy = new WebProxy();          // if you need an empty proxy
      //gets the system credentials of the application
      ambus.Proxy.Credentials = CredentialCache.DefaultCredentials;
      ambus.Url    = "http://192.168.1.10/"; // AMBUS® Net url[05]
      ...
    }
```

```
// function gets the data of a single meter.
private void GetMeter(byte btyNumber)
{
  try
  {
    Meter m = ambus.getMeter(btyNumber); //remote call
    if (m != null)
    {
      string deviceName = m.deviceName;
      ...
    }
  }
  catch (Exception ex)
  {
    // error while loading the meter btyNumber
    ...
  }
}
```

```
// function gets the data of a usage unit.
private void GetUsageUnit(int intNumber)
{
  try
  {
    UsageUnit u = ambus.getUsageUnit(intNumber); //remote call
    if (u != null)
    {
      string unitName = u.name;
      ...
    }
  }
  catch (Exception ex)
  {
    // error while loading the usage unit intNumber
    ...
  }
}
```

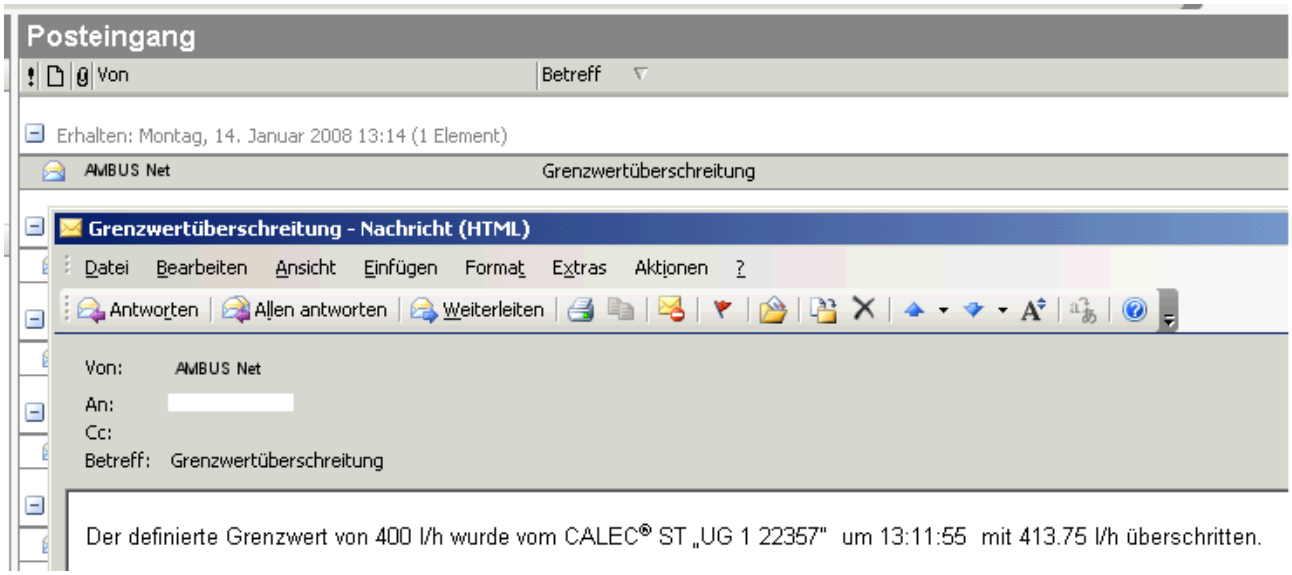
3.5 Anwendungen

Der integrierte Web Service von AMBUS® Net bietet zahlreiche Anwendungsmöglichkeiten. In diesem Kapitel werden einige Beispiele gezeigt.

3.5.1 Anzeigen / Loggen

Zählerstände und oder Momentanwerte können auf dem Bildschirm dargestellt werden. Wenn nötig können die Daten in einer Datenbank hinterlegt werden.

3.5.2 Grenzwertalarmierung



Werden in der Software definierte Grenzwerte überschritten kann das Programm, welches vom Systemintegrator erstellt wurde, darauf reagieren und beispielsweise ein E-Mail oder ein SMS verschicken.

3.5.3 Steuern und Regeln

Anstelle einer Grenzwertalarmierung kann die Software direkt reagieren und z.B. über ein Leitsystem ein Ventil betätigen um den entsprechenden Momentanwert anzupassen.

4 Begriffserklärungen

[01] Web Service: Ein Web Service / Webdienst ist eine Softwareanwendung, die im Internet eindeutig identifizierbar ist und deren Schnittstellen definiert, beschrieben und gefunden werden können.

[02] XML: eXtensible Markup Language ist eine Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten in Form von Textdateien.

[03] TCP/IP: IP: Das Internet Protocol ist für die Verbindung zweier Rechner verantwortlich.

TCP: Das Transmission Control Protocol dient zur Datenübertragung zwischen zwei Rechnern.

[04] WSDL: Die Web Services Definition Language definiert eine plattform-, programmiersprachen- und protokollunabhängige XML-Spezifikation zur Beschreibung von Netzwerkdiensten (Web Services) zum Austausch von Nachrichten.

[05] URL: Uniform Resource Locator: Identifiziert eine Ressource im Computernetzwerk.

Änderungen vorbehalten

Aquametro AG	Aquametro SA	Aquametro Messtechnik GmbH	Aquametro BELGIUM SPRL	Aquametro s.r.o.
Ringstrasse 75	Rue du Jura 10	Zum Panrepel 24	Dallaan, 67	Prosecká 811 / 76a
CH-4106 Therwil	CH-1800 Vevey	D-28307 Bremen	B-1933 Sterrebeek	CZ-190 00 Praha
Tel. 061 725 11 22	Tel. 021 926 77 77	Tel. 0421 / 871 64-0	Tel. 02 / 241 62 01	Tel. 02 / 86 88 77 78
Fax 061 725 15 95	Fax 021 926 77 78	Fax 0421 / 871 64-19	Fax 02 / 216 22 63	Fax 02 / 86 88 95 59
info@aquametro.com	info@aquametro.com	info.amd@aquametro.com	info.amb@aquametro.com	info.amc@aquametro.com