LIAB ApS
Østre Allé 6
DK-9530 Støvring
Tlf: +45 98 37 06 44
http://www.liab.dk

Energinet.dk
Tonne Kjærsvej 65
DK-7000 Fredericia
Att: Steen Kramer

`Journal nr. 2011-02-23/1` Støvring den 23. februar 2011

Programmering med gsoap ifht. varmepumpeprojektet

I forbindelse med demonstrationprojektet vedrørende dataopsamling og styring af varmepumper i private hjem, er der blevet opbygget en grænseflade til udveksling af data med den enkelte varmepumpeinstallation. Tilgang til data sker via en såkaldt "SOAP"-grænseflade, hvorved man blandt andet kan udtrække data for en given installation og for en given tidsperiode. Dette interface er dokumenteret i to tidligere LIAB-rapporter: "Dataudveksling for eksterne aktører i varmepumpeprojektet" [1] af 28. januar 2011 (`Journal nr. 2011-01-28/1`) og "Kommandointerface for eksterne aktører i varmepumpeprojektet" [2] af 21. februar 2011 (verb+Journal nr. 2011-02-21/1+).

"SOAP" er en forkortelse for "Simple Object Access Protocol" og indeholder specifikationer for udveksling af maskinlæsbar, struktureret information, i forbindelse med afvikling af WEB-Services. Formatet for udvekslinger er bygget på XML: "Extensible Markup Language. Selve datatransporten benytter forskellige netværksbaserede appliktionslag, især Remote Procedure Call (RPC) og Hypertext Transfer Protocol (HTTP). De mulige beskeder, ordre og dataindhold for input og output defineres i en såkaldt WSDL-fil (Web Services Description Language), et format der generelt benyttes til at beskrive WEB services.

I det følgende beskrives hvordan man dels installere et `gsoap` udviklingsmiljøet på en Ubuntu Linux-PC, dels hvordan man benytter dette miljø til at udvikle et C-program, der benytter en SOAP-grænseflade.

1 Installation af gsoap

Installation af gsoap på en Linux-PC med Ubuntu 10.04 eller nyere gøres simpelt med værktøjerne fra "Advanced Packaging Tool" eller APT:

```
..$ sudo apt-get install gsoap
```

Som konsekvens af denne installation får to programmer, som benyttes i udviklingen:

wsdl2h og soapccp2

Disse programmer benyttes som følger:

- wsdl2h benyttes til at oversætte en WSDL-fil (Web Services Description Language) til en serie af C header-filer. Disse indeholder datastrukturer og prototyper på funtioner, som overholder de definitioner, som er givet i wsdl-filen.
- soapccp2 benyttes derefter til at skabe skeletter og funktions-templates der kan benyttes i den senere programskrivning.

Derudover installeres relevante header-filer og biblioteker.

2 Benyttelse af gsoap til dataudtræk

Før man kan begynde kodeskrivningen i C i et gsoap-miljø skal have genereret de relevante headerfiler og templates. Dette gøres med udgangspunkt i den for systemet relevante WSDL-fil. Som beskrevet i [1] finder man en WSDL-fil med beskrivelse af datastrukturer og metoder for dataudtræk via gsoap på webadressen: "http://datalogger.liab.dk/rpfchart_flash.wsdl". Efter at have hentet denne benyttes wsdl2h og soapcpp2 som angivet nedenfor:

```
..$ wget http://datalogger.liab.dk/rpfchart_flash.wsdl
    ....
..$ wsdl2h -c rpfchart_flash.wsdl
    ....
..$ soapcpp2 rpfchart_flash.h
```

Disse kald giver anledning til følgende filer i direktoriet:

```
rpfchart_flash.h            rpfchart.getEvents.res.xml  soapH.h
rpfchart_flash.wsdl         rpfchart.nsmap              soapServer.c
rpfchart.getETypes.req.xml  soapC.c                     soapServerLib.c
rpfchart.getETypes.res.xml  soapClient.c                soapStub.h
rpfchart.getEvents.req.xml  soapClientLib.c
```

hvor der således er templates til såvel at skrive en soap-server som en soap-client. For a simplificere kodeskrivningen har vi følgende `Makefile`:

```
#
# Makefile for simpel gsoap-klient:
#   Mikael Dich, LIAB ApS, februar 2011

PROJ            = MyFirstClient
PROJSRC         = MyFirstClient.c soapC.c  soapClient.c
LIB             = -lgsoap
CFLAGS          = -Wall -Wno-unused

all:            .depend $(PROJ)

$(PROJ):        $(PROJSRC:.c=.o)
                $(CC) $^ $(LIB) -o $@

clean:
    rm -f *.o *~ core .depend

depend .depend dep: $(PROJSRC)
    $(CC) $(CFLAGS) -M $^ > .depend

ifeq (.depend,$(wildcard .depend))
include .depend
endif
```

Følgende kodeeksempel viser hvordan man initialisere og sender en gsoap-request til en server. For at passe med ovenstående `Makefile` skal kildeteksten placeres i filen `MyFirstClient.c`:

```
/*
** gsoap demo program til dataudtræk ifbm varmepumpeprojektet
** Mikael Dich, LIAB ApS, februar 2011
*/
#include <stdio.h>
#include <string.h>
#include <unistd.h>
```

```
#include "soapH.h"        // obtain the generated stubs
#include "rpfchart.nsmap" // obtain the namespace mapping table
#define MAXLEN (512)
#ifdef    undef
// Denne funktion, som er defineret i soapClient.c og proto-
// typed i soapStub.h, som er inkluderet i soapH.h modsvarer
// metoden "getETypes" beskrevet i dokumentet "Dataudveksling
// for eksterne aktører i varmepumpeprojektet", J.nr. 2011-01-28/1
        struct soap *soap,
        const char *soap_endpoint,
        const char *soap_action,
        char *box,
        struct ns1__getETypesResponse *_param_1)
#endif /* undef */

#define DEFAULTSOAPSERVERURL     "http://datalogger.liab.dk"
#define DEFAULTSOAPCGINAME       "/cgi-bin/rpfchart.cgi"
#define DEFAULTHEATPUMPLIABSGBOX "12"  // Mikael's testinstallation!

int main(int narg, char *argp[])
{
    int i, rtn, etypeitems;
    char ServerURL[MAXLEN], BoxID[MAXLEN];
    struct soap soap;
    struct ns1__getETypesResponse ETret;
    struct ns1__etype *etypeptr;

    // gsoap initializing with resonable options, allocates an
    // instance of "struct soap"
    soap_init1(&soap, SOAP_C_UTFSTRING | SOAP_IO_STORE);

    // Setup strings with URL and boxid
    strncpy(ServerURL, DEFAULTSOAPSERVERURL, MAXLEN-1);
    strncat(ServerURL, DEFAULTSOAPCGINAME, MAXLEN-1);
    strncpy(BoxID,     DEFAULTHEATPUMPLIABSGBOX, MAXLEN-1);

    // Make the gsoap call ...
    rtn = soap_call_ns1__getETypes(&soap,
              ServerURL, NULL, BoxID, &ETret);
    printf("Soap call returns: ...: %4d, %s\n",
           rtn, rtn == SOAP_OK ? "OK" : "Fail");
    // Bail out if something went wrong !
    if (rtn != SOAP_OK)
        return -1;
    // Find the number EType items ...
    etypeitems = ETret.result->__sizeitem;
    printf("Items returned .......: %4d\n", etypeitems);
    // ... and print them out:
```

```
    for(i = 0; i < etypeitems; i++)
    {
        etypeptr = &(ETret.result->item[i]);
        printf("%2d: %4d   %-20s   %-20s\n", i,
               etypeptr->eid, etypeptr->name,
               etypeptr->description);
    }
    return 0;
}
```

Kompileres ovestående kildetekst med `make` fås en eksekverbare fil: "`MyFirstClient`". Afviklingen af programmet giver følgende resultat:

```
..$ ./MyFirstClient
Soap call returns: ...:    0, OK
Items returned .......:   27
 0: 1133   acc.hflow              Varme
 1: 1134   acc.htfwd             Varme frem
 2: 1130   acc.htret              Varme retur
 3: 1140   acc.pwrh               Varme
 4: 1139   acc.pwrw               Forbrugsvand
 5: 1137   acc.rtind              Indendørs
 6: 1138   acc.rtoutd            Udendørs
 7: 1132   acc.tttop              Tank
 8: 1136   acc.wcin               Koldt vand ind
 9: 1135   acc.wflow              Varmt vand
10: 1131   acc.whout              Varmt vand ud
...
..$
```

Til dokumentet hører en `tar`-arkiv med de nødvendige filer:

`GsoapDemoprogramForVarmepumpeProj24feb2011.tgz`

i arkivet findes et shellscript som bygger `MyFirstClient` og eksekverer det!

Vi står til jeres disposition for at besvare eventuelle spørgsmål.

Med Venlig Hilsen

LIAB ApS
Mikael Dich
midi@liab.dk