Anbindung von 4D Datenbanken an STARFACE Telefonanlagen

Durch die fehlende TAPI-Unterstützung auf dem Mac sind Anbindungen an Telefonanlagen nicht so weit verbreitet wie unter Windows. Mit der zunehmenden Zahl an IP Telefonanlagen hat sich die Situation aber verbessert, da diese über IP und nicht über TAPI angesprochen werden können. Im folgenden wird gezeigt, dass die Anbindung einer 4D Datenbank an eine STARFACE-Anlage nicht sonderlich aufwendig ist. STARFACE ist eine kommerzielle Version der freien Software-Telefonanlage Asterisk.

Anforderung

Die zugrundeliegenden Routinen entstanden aus einem Projekt mit folgenden Anforderungen:

- aus einer 4D Datenbank telefonieren, d.h. Telefongespräche mit den in der Datenbank hinterlegten Telefonnummern initiieren.
- Eingehende Anrufe signalisieren und zugehörige Datensätze wie Adresse, offene Rechnungen etc. anzeigen.
- Faxen
- Mac und Win / Client/Server

Testumgebung

Falls keine STARFACE-Anlage zur verfügung steht, kann man sich mit relativ wenig Aufwand eine Testumgebung schaffen. STARFACE free gibt es als kostenlose reine Softwarelösung. Man kann die Version vorinstalliert als VM-Ware - Datei downloaden. Ich habe die Datei mit Paralles auf dem Mac geöffnet, die Anlage lief auf Anhieb. Probleme gab es lediglich, wenn der Mac bei laufender VM in den Ruhezustand ging: die VM zog 100 % Prozessorlast und musste neu gestartet werden.

Als „Telefon“ habe ich X-Lite auf dem Mac benutzt. Hier hatte ich eines der Hauptprobleme: beim Telefon muss Name und Password des in STARTFACE definierten *Telefons* eingegeben werden und nicht der des *Users.*

Außerdem habe ich bei SIPGate 2 Accounts eingerichtet, um das Raustelefonieren testen zu können. SIPgate-interne Telefonate sind kostenlos. Einen Account habe ich in der STARFACE-Anlage hinterlegt, den anderen in meiner Telefonanlage, so dass „echte“ Telefonate geführt werden konnten

Dokumentation

Die API von STARFACE ist recht gut dokumentiert. Die aktuelle Version der Dokumentation kann im STARFACE Wiki finden. Ich habe das STARFACE Unified Communications Interface (UCI) in der Version 2.1 verwendet.

Außerdem gibt es bei STARFACE ein STARFACE Integration Development Forum , in dem manchmal etwas spät, aber doch kompetent auf Fragen geantwortet wird.

Grundlagen

Die Kommunikation mit der Anlage läuft über HTTP. Bei ausgehenden Telefonaten ist dies in 4D V13 recht einfach über **HTTP Request** zu realisieren.

Bei eingehenden Gesprächen meldet sich die Anlage über HTTP an einer definierten Adresse. Hierfür wäre der 4D Webserver prädestiniert, die Lizenzkosten im genannten

Projekt für 35 Clients in Höhe von 699,- € (Monitor am Server) bzw. 24.465,-€ (Monitor an jedem Client) sind leider aber doch etwas abschreckend. Glücklicherweise lässt sich das auch mit dem 4D Internet-Commands PlugIn lösen.

Für alle Operationen mit der Anlage ist ein Login nötig. Dieser erfolgt mit dem Benutzernamen des STARFACE-Users (nicht des Telefons!) und einem MD5-Digest aus „Username*Password“. In 4D V13:

```
$password_t:=Generate digest($user_t+"*"+$userpass_t;MD5 Digest).
```

In V12 muss man auf die PHP-Funktion hash zurückgreifen.

Für die Callback-Routinen muss die IP des Rechners übergeben werden. Diese erhält man mit den Internet-Commands:

```
$err_l:=IT_MyTCPAddr (<>mytcpadr_t;$subnet_t)
```

Befehle an die Anlage müssen an die URL

http://Adresse_*der_Anlage*/xml-rpc?*Parameter*

erfolgen.

Die Parameter sind im STARFACE Unified Communications Protocol 2.1.pdf S. 40 dokumentiert:

de.vertico.starface.callback.type	HTTP Protocol type to be used when sending server events (either “http” or “https”, default is “http”)
de.vertico.starface.callback.host	Host name to be used when sending server events (default is “127.0.0.1”)
de.vertico.starface.callback.port	Port number to use when sending server events (default is 80)
de.vertico.starface.callback.path	Path to use when sending server events (default is “/”)
de.vertico.starface.user	LoginID of the STARFACE user the message belongs to e.g. 0001
de.vertico.starface.auth	MD5 authentication token of the STARFACE user this message belongs to **in lower case** Token is constructed as MD5(<loginId>*<password>) e.g. MD5(“0001*Password”)= “7c2d62a5c855633b86957729d563b224”

Im Einfachsten Fall also

```
$url_t:="http://"+$adress_t+"/xml-
rpc?"+"de.vertico.starface.user="+$login_t
+"&de.vertico.starface.auth="+$password_t
```

Falls Anrufe signalisiert werden sollen, muss zusätzlich die Callback-Adresse und der zugehörige Port übergeben werden (auch bei ausgehenden Gespächen, bei denen kein Callback erfolgt!).

Die Befehle an die Anlage werden als XML-Text übergeben. Hier für den Login:

```
<methodCall>
      <methodName>ucp.v21.server.connection.login</methodName>
</methodCall>
```

Die Anlage antwortet im Idealfall mit:

```
<methodResponse>
  <params>
    <param>><!-- successful -->
      <value>
        <boolean>1</boolean>
      </value>
    </param>
  </params>
</methodResponse>
```

Für den Login wird also folgende Methode **UCP_DoCommand** (siehe Beispielanwendung) mit dem Parameter „login“ aufrufen:

```
C_LONGINT($0)
C_TEXT($1)  // Command

C_LONGINT($result_l)
C_TEXT($adress_t;$command_t;$login_t;$password_t;$post_t;$ref_t;$ucptext_t;$url_t;$xml_t)

$command_t:=$1

  // Parameter
$adress_t:=[Voreinstellungen]Anlage_Adresse
$login_t:=[Voreinstellungen]Username
$password_t:=[Voreinstellungen]Password

$url_t:="http://"+$adress_t+"/xml-rpc?"\
  +"de.vertico.starface.user="+$login_t\
  +"&de.vertico.starface.auth="+$password_t

  // Command
$xml_t:=DOM Create XML Ref("methodCall")
$ref_t:=DOM Create XML element($xml_t;"methodName")
Case of
: ($command_t="login")
  DOM SET XML ELEMENT VALUE($ref_t;"ucp.v21.server.connection.login")
: ($command_t="logout")
  DOM SET XML ELEMENT VALUE($ref_t;"ucp.v21.server.connection.logout")
End case
DOM EXPORT TO VAR($xml_t;$post_t)
DOM CLOSE XML($xml_t)

$ucptext_t:=""  // Response

$result_l:=HTTP Request(HTTP POST Method;$url_t;$post_t;$ucptext_t)

If ($result_l=200)
   // check response
  $0:=0
Else
  ALERT("Error: "+String($result_l))
  $0:=29999  // Error
End if
```

Vorhandene Telefone

Für ausgehende Gespräche muss festgelegt werden, mit welches Telefon benutzt werden soll. Die zur Verfügung stehenden Telefone bekommt man mit
`ucp.v21.server.communication.call.getPhoneIds`
In der Beispielanwendung ist das in der Methode **UCP_GetPhones** realisiert.

Wählen

Ein Anruf wird über
`ucp.v21.server.communication.call.placeCallWithPhone`
getätigt. Dabei müssen Nummer und Telefon übergeben werden. Siehe Methode **UCP_Dial**.

Anrufe anzeigen

Um eingehende Anrufe anzuzeigen, muss entweder der 4D Webserver oder ein eigener Listener mit den 4D Internet Commands laufen.

Der Anlage muss zuerst mitgeteilt werden, dass die Anrufe signalisiert werden sollen. Dazu dient:
`ucp.v21.server.connection.setProvidedServices`
`ucp.v21.client.connection`
`ucp.v21.client.communication.call`

```
$xml_t:=DOM Create XML Ref("methodCall")
$ref_t:=DOM Create XML element($xml_t;"methodName")
DOM SET XML ELEMENT VALUE($ref_t;"ucp.v21.server.connection.setProvidedServices")
$element_t:=DOM Create XML element($xml_t;"params")
$element_t:=DOM Create XML element($element_t;"param")
$element_t:=DOM Create XML element($element_t;"value")
$element_t:=DOM Create XML element($element_t;"array")
$element_t:=DOM Create XML element($element_t;"data")
$value_t:=DOM Create XML element($element_t;"value")
DOM SET XML ELEMENT VALUE($value_t;"ucp.v21.client.connection")
$value_t:=DOM Create XML element($element_t;"value")
DOM SET XML ELEMENT VALUE($value_t;"ucp.v21.client.communication.call")
DOM EXPORT TO VAR($xml_t;$post_t)

DOM CLOSE XML($xml_t)
$ucptext_t:=""

$result_l:=HTTP Request(HTTP POST Method;$url_t;$post_t;$ucptext_t)
```

Anschließend meldet die Anlage die Anrufe an die angegebene IP-Adresse mit dem angegebenen Port.

4D sollte mit
`<?xml version="1.0"?>\r\n<methodResponse><params><param><value><boolean>1</boolean></value></param></params></methodResponse>\r\n`
antworten.

In der Beispielanwendung ist das in **UCP_Listen** realisiert.

Zusätzlich muss STARFACE mindestens alle 30 Sekunden mitgeteilt werden, dass 4D noch „lebt“, also noch auf Nachrichten lauscht. Dies geschieht durch Senden von
`ucp.v21.server.connection.probe`
alle 25 Sekunden. In der Beispielanwendung erfolgt dies über einen eigenen Prozess.

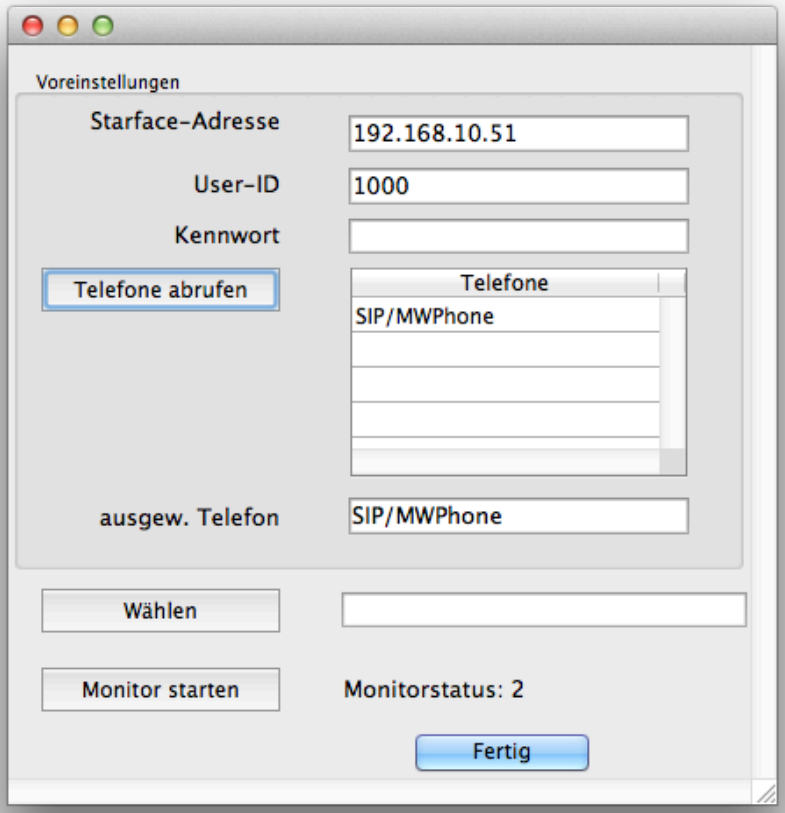
Faxen

Die STARFACE-API bietet auch die Möglichkeit, PDFs als Fax zu verschicken. Dazu muss die PDF-Datei aufgeteilt, jeder Teil mit BinHex umgewandelt und mit einer Prüfsumme versehen werden. Alle Teile werden an die Anlage geschickt, wiederum mit einer

Prüfsumme. Bei meinen Tests war ich damit nicht erfolgreich, STARFACE hat immer aufschlussreich „0“ zurückgemeldet. Daher bin ich auf die wesentlich einfacher zu implementierende Variante über das kostenlose STARFACE–PlugIn Mail2Fax umgestiegen. Die Vorgehensweise ist simpel: ein Mail mit der Faxnummer im Betreff und dem PDF im Anhang an eine Mailadresse schicken, fertig. Den Rest übernimmt das PlugIn.

Beispielanwendung

Die Beispielanwendung liegt für 4D V13 vor. In On Startup wird ein Dialog gezeigt, der eine Eingabemöglichkeit für nötigen Einstellungen, einen Anruf tätigen und den Anrufmonitor starten lässt:



Fazit

Die Anbindung einer 4D Datenbank an STARFACE ist mit überschaubarem Aufwand zu realisieren. Auch hier nicht angesprochene Funktionen wie Anruflisten, weiterverbinden etc. sind problemlos möglich.