



Enhanced Speech Processing (ESP) for **maximiser**

An Introduction



Table of Contents

1. Overview	3
Enhanced Speech Processing (ESP)	3
2. Sample Scripts	5
Simple Extended Auto Attendant (mainmenu.php)	7
info.php	9
speak_cli.php	10
speak_number.php	11
transfer.php	13
record.php	15
record2.php	18
Multi Level Auto Attendant (multilevel.php)	19
multiinfo.php	23
CLI and DTMF Input Capture (capture.php)	25
submit.php	27
Text to Speech Conversion (vxmlmartin.php)	29
Wake-Up/ Alarm/Reminder Calls (wakeup_vxml.php)	31
Appendices	49
Appendix i - Further Reading	49

1. Overview

Enhanced Speech Processing (ESP)

ESP is available as a value added addition to the voicemail & auto-attendant facilities provided as standard on **maximiser**. Developed under VoiceXML (VXML), ESP provides multi-level auto-attendant, IVR/Voice Form and Text-to-Speech facilities and can be run on the 4140 Remote Call Server to support up to 4 concurrent calls/sessions and the 4100 Call Server to support up to 8 concurrent calls/sessions. For larger requirements where a higher density of concurrent calls is required, ESP can be run on a standalone Linux PC or Server where the only limiting factor will be the processing power and disc performance of the chosen platform. ESP is licensed on a per concurrent session basis.





Standard Voicemail

Voicemail and a simple, single level, ten-entry auto-attendant are supplied as standard with each **maximiser** system. Running on the 4100 Call Server module, which supports up to 8 concurrent calls, the voicemail application is provisioned with 30 voicemail boxes free of charge - extra mailboxes can be enabled by license as and when required. The identical voicemail application running on the 4140 Remote Call Server supports up to 4 concurrent calls and comes supplied with 10 voicemail boxes as standard. Again, extra mailboxes can be added as required. If more than 8 concurrent calls are required voicemail “responsibilities” can be distributed to multiple Call Servers, or, alternatively, the voicemail application can be run “off-switch” on a Linux PC or Server running SuSE Linux version 8.1 to version 9.3. In the latter case the number of concurrent calls is limited only by the processing capability of the chosen platform. As an example, today’s entry-level PCs are capable of supporting over 30 concurrent calls.

VoiceXML

Building upon eXtensible Mark-up Language (XML), VoiceXML is the open standard programming language that was created to simplify the development of interactive voice applications. Originally defined by the VoiceXML Forum, a consortium of over 500 companies, in 2000, VXML is now under the control of the World Wide Web Consortium (W3C) where some 600+ companies support it in their products and use it to develop applications. By using the same underlying network infrastructure, Hyper Text Transfer Protocol (HTTP) communications and mark-up language programming model, VXML harnesses the existing investment in skills and massive infrastructure developed for web based applications and content, making it easy to create and deploy advanced voice applications. Taking advantage of several trends; the growth of the World Wide Web, it’s capabilities and increased reach beyond the desktop PC, alongside great improvements in computer-based speech recognition and text-to-speech synthesis, the goal of VXML is to bring the advantages of Web-based development and content delivery to interactive voice response applications. As such and like HTML, VXML opens up huge business opportunities for developers.

Whilst HTML pages are navigated using a browser, display, keyboard and mouse, interaction with VXML pages utilises audio output, audio input and keypad input, i.e. via a standard telephone. As a language, VXML describes the human-machine interaction provided by voice response systems, which includes:

-  Output of synthesized speech (text-to-speech).
-  Output of audio files.
-  Recognition of DTMF input (touch tone).
-  Recording of spoken input.

- § Telephony features such as call transfer and disconnect.
- § Recognition of spoken input (speech recognition - not currently supported on ESP)

VXML provides the means for collecting DTMF and/or spoken input, assigning this input to document-defined request variables, and making decisions that affect the interpretation of VXML documents. A VXML document may be linked to other documents through Universal Resource Identifiers (URIs) – as per URLs in an HTML context. In plain English, this means that the DTMF or spoken input from a phone can be captured, acted upon or forwarded to external databases or other applications.

Because of its open nature, close relationship to HTML, and ability for resellers to provide value-add through customisation and development, SpliceCom have chosen to use VoiceXML as the language under which their Enhanced Speech Processing application will be developed. For those wishing to learn more about VXML and its ability to add-value to **maximiser** based systems, it's one of the areas that will be covered in SpliceCom's Advanced Integrators Course in the very near future.

Multi-Level Auto-Attendant

ESP provides support for a multi-level auto-attendant. There are no limits to the number of branches or levels that can be programmed for each auto-attendant enabled DDI, nor in the number of DDIs running different auto-attendant configurations. As ESP can be licensed on a per concurrent session basis, and each Remote Call Server supports up to 4 concurrent sessions, with Call Server supports up to 8 concurrent sessions. A purchase of 4 ESP concurrent licenses allows 4 multi-level and 4 single-level auto-attendant sessions to be active at the same time on a 4100 Call Server. Configuration and text to speech generation of voice prompt (or call recording of voice prompts) are all made under VXML for which a selection of templates is available.

Text-to-Speech Conversion

Software that allows text to be converted into audible speech has been licensed from US specialists Cepstral and added as an integral feature to ESP. Support of text-to-speech provides a very valuable and flexible alternative to the use of pre-recorded .wav files in cases where:

- § Recordings are too expensive.
- § The application does not know ahead of time what it will need to speak.
- § The information varies too much to record and store all the alternatives.

Examples of implementation include wake-up/alarm/scheduled calls and the announcement of changes in company/ staff DDI numbers to callers with an automatic re-direct.

IVR/Voice Forms

ESP provides an IVR service based on Voice Forms. This allows a series of voice prompts to be generated using Text-to-Speech, or recorded as .wav files, responses captured - be they spoken or DTMF tones - and the results emailed as .wav files or text. This type of application is ideal for competition lines or telephony services through which votes or orders are placed without the need of human intervention.

Future Direction

The use of VXML as the language that underpins ESP will allow future releases to provide speech recognition services. These will include the ability to "speak" the internal directory or external contact name you wish to dial, "talk" to your phone to change your personal settings and have your emails "read" to you remotely.

2. Sample Scripts

VoiceXML will allow you to answer calls, play files, collect DTMF digits, record files and transfer calls. The scripts can also, using PHP, examine and perhaps modify the **maximiser**'s configuration database - for example, allowing you to restrict which extensions the public can dial, or redirect out of hours calls to another mobile.

The following are examples of sample scripts for you to copy and use as a starting point for your own scripts. Although initially they may look complex, you will find that they can be easily customised using "cut & paste".

By visiting the VoiceXML page <http://support.splicecom.com/forums/showthread.php?s=&threadid=639> located on the Web/PHP area of the Technical Forum you can download vxmldemo.tar (for those of you without access to the SpliceCom Technical Forum please send an email request to info@splicecom.com).

This file can be expanded to provide a VXML folder containing the following;

- 📄 mainmenu.php
- 📄 record.php
- 📄 record2.php
- 📄 info.php
- 📄 speak_cli.php
- 📄 speak_number.php
- 📄 transfer.php
- 📄 afternoon.wav
- 📄 evening.wav
- 📄 goodbye.wav
- 📄 info_intro.wav
- 📄 main_intro.wav
- 📄 main_intro2.wav
- 📄 main_sorry.wav
- 📄 main_transfer.wav
- 📄 morning.wav
- 📄 thankyou.wav

These provide you with a main menu Auto Attendant script (mainmenu.php) and associated files that will allow you to;

- 📄 Dial by extension number
- 📄 Dial a pre-recorded message (.wav)
- 📄 Dial the Sales department
- 📄 Have your CLI read back to you
- 📄 Dial, or wait to be transferred to Reception

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

After downloading vxmldemo.tar you need to place it in the ftp area of the Call Server/Remote Call Server.

i.e. Transferring vxmldemo.tar from the PC Desktop to the ftp area of the Call Server with IP address 192.168.3.1 using the Command Prompt. Please note that the process documented below will also require you to login to the Call Server as an intermediate step.

```
>cd Desktop/  
> ftp 192.168.3.1  
ftp> put vxmldemo.tar
```

Then telnet in to the Call Server and run the following command to move and expand the file into the /Web/vxml directory: -

```
>cd /  
>tar -xvf /ftp/vxmldemo.tar
```

Next you will need to create DDI or Dial Plan entries so you can dial the following destinations: -

!EA:http://127.0.0.1/vxmlmainmenu.php - To provide a number where the Auto Attendant can be accessed from. i.e. Using “*41” :

Number Match	Action	Translate To	Time Plan	Translate CLI To	Least Cost Call Plan
*41	Dial	!EA:http://127.0.0.1/	Standard		Standard

!EA:http://127.0.0.1/vxmlrecord.php - Should you wish to re-record the menus in a more acceptable voice (the default .wav files were recorded by our very own Bob Geddes)! i.e. using *42

Number Match	Action	Translate To	Time Plan	Translate CLI To	Least Cost Call Plan
*42	Dial	!EA:http://127.0.0.1/	Standard		Standard

Please remember that an ESP Session Licence is required for each active session/concurrent call required. Once you've done that don't forget to enter a number in the “ESP Capacity” field on the “Voicemail Ports” menu.

Name	Builtin
Description	Internal Voicemail Ports
Location	Demo Suite Call Server
H323 Address	CallServer 00-07-d9-00-03-cc
Capacity	2
Email Smart Host	192.168.3.101
Email Source Address	vmail@maximiser.local
Product Version	2.2(744)
Dial Plan	Standard
Remote Location	
ESP Capacity	1
CapabilityAA	<input checked="" type="checkbox"/>
CapabilityVM	<input checked="" type="checkbox"/>
CapabilityMusic	<input checked="" type="checkbox"/>
CapabilityQueue	<input checked="" type="checkbox"/>
CapabilityRecording	<input checked="" type="checkbox"/>
CapabilityConference	<input checked="" type="checkbox"/>
CapabilityEA	<input checked="" type="checkbox"/>

When testing a script the voicemail log can give many clues in the problem (either tail -f /Voicemail.log or cat /dev.emlog2) also you can look at a script via your browser (<http://192.168.0.1/vxmlmainmenu.php>).

Simple Extended Auto Attendant (mainmenu.php)

```

<?
// this is the basics for a simple Extended Auto Attendant which we hope you can cut and
// paste to your requirements.
//     ini_set("url_rewriter.tags","");
//     session_start();

// the following can be used to select messages dependant on the time of day
    $date = getdate();
    $hour = $date['hours'];
    $greeting = "morning.wav";
    if ($hour > 11) $greeting = "afternoon.wav";
    if ($hour > 17) $greeting = "evening.wav";

// the following line is printed from php to avoid php thinking its script
    print '<.'?xml version="1.0" encoding="ISO-8859-1"?.'>';
?>
<vxml version="2.0">
    <var name="operator" expr="8000"/>
    <form id="main">
        <field name="extn" type="digits?">
            <prompt timeout="5s">
<!-- the following is commented out so it does not play
                <audio src="<?=$greeting?">/>
-->

                <audio src="main_intro.wav"/>
                <audio src="main_intro2.wav"/>
            </prompt>
            <catch event="noinput">
                <var name="extn" expr="operator"/>
                <submit next="transfer.php"/>
            </catch>
            <filled>
                <if cond="extn=='0'">
                    <var name="extn" expr="operator"/>
                </if>
                <if cond="extn=='1'">
                    <goto next="info.php"/>
                </if>
                <if cond="extn=='2'">
                    <goto next="speak_cli.php"/>
                </if>
                <if cond="extn=='3'">
                    <var name="extn" expr="8000"/>

```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
</if>
    <submit next="transfer.php" />
</filled>
</field>
</form>
</vxml>
```

Notes:

This .php script runs when *41 is dialed on the **maximiser** system.

After entering the AutoAttendant you will hear the following;

"Main Menu, you may dial your extension now ([main_intro.wav](#)) or press 0 or wait for Reception ([main_intro2.wav](#))"

When the caller dials the extension we check if its valid and not ex-directory before transferring the call.

If you dial "1" you will be played a message after being routed to info.php.

If you dial "2" your CLI will be read back to you after being routed to speak_cli.php

If you dial "3" you will be transferred to extension 8000 (the Sales Department) after being routed to transfer.php

If you dial "0" or there is no input, you will be transferred to the Operator after being routed to transfer.php

Note: In this example the Operator has also been defined as extension 8000 - `<var name="operator" expr="8000" />` You can change this to suit your customer's numbering scheme by simply substituting a different number.

The "Good Morning/Afternoon/Evening" greeting has been edited out;

```
<!-- the following is commented out so it does not play
    <audio src="<?=$greeting?" />
-->
```

This can be activated by simply changing it to;

```
// the following now plays
    <audio src="<?=$greeting?" />
```

info.php

```
<?
    print '<' . '?' . 'xml version="1.0" encoding="ISO-8859-1"?' . '>';
?>
<vxml version="2.0">
    <form>
        <block>
            <audio src="info_intro.wav"/>
        </block>
    </form>
</vxml>
```

Notes:

This script is executed when "1" is dialed after entering the AutoAttendant system.

It causes the [info_intro.wav](#) file to be played and you will hear the following;

"This is a demonstration of the main information."

The call is then disconnected.

speaking_cli.php

```
<?
//  ini_set("url_rewriter.tags","");
//  session_start();
    print '<.'?xml version="1.0" encoding="ISO-8859-1"?.'.'>';
    print '<vxml version="2.0">';
?>
<!-- set number as callers CLI -->
    <var name="number" expr="session.telephone.number"/>
    <form>
        <subdialog src="speak_number.php" namelist="number"/>
    </form>
</vxml>
```

Notes:

This script is executed when "2" is dialed after entering the AutoAttendant system.

It captures the caller's CLI and then invokes the "speak_number.php" script.

speak_number.php

```

<?
//  ini_set("url_rewriter.tags","");
//  session_start();
print '<.'.'?xml version="1.0" encoding="ISO-8859-1"?'.>';
print '<vxml version="2.0">';

function digits($n) {
    switch ($n) {
        case 0:
            print '<audio src="/SpliceCom/Wavs/en/number00.wav"/>';
            break;
        case 1:
            print '<audio src="/SpliceCom/Wavs/en/number01.wav"/>';
            break;
        case 2:
            print '<audio src="/SpliceCom/Wavs/en/number02.wav"/>';
            break;
        case 3:
            print '<audio src="/SpliceCom/Wavs/en/number03.wav"/>';
            break;
        case 4:
            print '<audio src="/SpliceCom/Wavs/en/number04.wav"/>';
            break;
        case 5:
            print '<audio src="/SpliceCom/Wavs/en/number05.wav"/>';
            break;
        case 6:
            print '<audio src="/SpliceCom/Wavs/en/number06.wav"/>';
            break;
        case 7:
            print '<audio src="/SpliceCom/Wavs/en/number07.wav"/>';
            break;
        case 8:
            print '<audio src="/SpliceCom/Wavs/en/number08.wav"/>';
            break;
        case 9:
            print "<audio src=\""/SpliceCom/Wavs/en/number09.wav\""/>";
            break;
    }
}

print "<form><block>";
$number = $_GET['number'];

```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
for ($n=0;$n<strlen($number);$n++) {  
    digits($number[$n]);  
}  
print "<return/></block></form></vxml>";  
?>
```

Notes:

This script is executed when "2" is dialed after entering the AutoAttendant system and after "speak_cli.php" has run.

The caller's CLI details were previously captured in "speak_cli.php" and are now read back using the default **maximiser** numeric .wav files (female voice).

The call is then disconnected.

transfer.php

```

<?
//  ini_set("url_rewriter.tags","");
//  session_start();
print '<.'.'?xml version="1.0" encoding="ISO-8859-1"?.'.'>';
print '<vxml version="2.0">';

// pickup extention to dial
    $extn=$_GET['extn'];
    $dial = 0;

// connect to local database (no error checking)
    $ds = ldap_connect("127.0.0.1",4000);
    ldap_bind($ds,"INTERNAL","001122334455");

// search for users or departments that are not ex-directory
    $a =array("telephonenumber","exdirectory");
    $sr = ldap_search($ds,"","globaltelephonenumber=$extn", $a,0,1);
    $info = ldap_get_entries($ds,$sr);
    if ($info['count'] != 0) {
        if (($info[0]['exdirectory'][0] == 0) && ($info[0]['telephonenumber'][0] ==
$extn)) $dial = 1;
    }
    if ($dial==1) {

// do transfer
?>
    <form id="transfer">
        <transfer name="call" dest="<?=$extn?>">
            <audio src="main_transfer.wav"/>
        </transfer>
    </form>
<?
    } else {

//go back
?>
<form id="goback">
    <block>
        <audio src="main_sorry.wav"/>
        <goto next="mainmenu.php"/>
    </block>
</form>

```

```
<?  
    }  
?>  
</vxml>
```

Notes:

This script is executed when either “3” or “0” is dialed after entering the AutoAttendant system, or if there is no input.

In all three examples above the number to be transferred to is “8000.”

The script is also executed when the caller has input the extension number to be dialed via the telephone keypad.

The required extension number is captured and then checked against **maximiser**’s internal directory for Users and Departments to ensure that it is a valid number AND it is not an ex-Directory listing. If all is ok the correct User/Department number is dialed, the “main_transfer.wav” is played to the caller (who will hear “Transferring you now”) and the call is then transferred.

If the required number is not a recognised extension OR it is an ex-director listing the “main_sorry.wav” will be played to the caller (who will hear ‘Sorry it has not been possible to connect you”). The “mainmenu.php” script will then be executed again and the caller will be returned to the start of the main AutoAttendant menu again.

record.php

```

<?
//ini_set("url_rewriter.tags","");
//session_start();
print '<.'?xml version="1.0" encoding="ISO-8859-1"?.'.'>';
?>
<vxml version="2.0">
  <var name="file"/>
  <menu id="greetings">
    <property name="inputmodes" value="dtmf"/>
    <prompt timeout="3s">
      Main Recording Menu.
      Press 1 for Main Menu.
      Press 2 for Information.
      Press 9 for general.
    </prompt>
    <choice dtmf="1" next="#main"/>
    <choice dtmf="2" next="#info"/>
    <choice dtmf="9" next="#general"/>
  </menu>

  <menu id="main">
    <property name="inputmodes" value="dtmf"/>
    <prompt timeout="3s">
      Main Menu.
      Press 1 for introduction.
      Press 2 for introduction 2.
      Press 3 for transfer.
      Press 4 for sorry.
    </prompt>
    <choice dtmf="1" next="#main_intro"/>
    <choice dtmf="2" next="#main_intro2"/>
    <choice dtmf="3" next="#main_transfer"/>
    <choice dtmf="4" next="#main_sorry"/>
  </menu>

  <form id="main_intro"><block><assign name="file" expr="'main_intro.wav'"/><goto
next="#record"/></block></form>
  <form id="main_intro2"><block><assign name="file" expr="'main_intro2.wav'"/><goto
next="#record"/></block></form>
  <form id="main_transfer"><block><assign name="file"
expr="'main_transfer.wav'"/><goto next="#record"/></block></form>
  <form id="main_sorry"><block><assign name="file" expr="'main_sorry.wav'"/><goto
next="#record"/></block></form>

```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
<menu id="info">
  <property name="inputmodes" value="dtmf"/>
  <prompt timeout="3s">
    Information Menu.
    Press 1 for main information.

  </prompt>
  <choice dtmf="1" next="#info_intro"/>
</menu>
<form id="info_intro"><block><assign name="file" expr="'info_intro.wav'"/><goto
next="#record"/></block></form>

<menu id="general">
  <property name="inputmodes" value="dtmf"/>
  <prompt timeout="3s">
    General Menu.
    Press 1 for good morning.
    Press 2 for good afternoon.
    Press 3 for good evening.
    Press 4 for thankyou.
    Press 5 for goodbye.

  </prompt>
  <choice dtmf="1" next="#morning"/>
  <choice dtmf="2" next="#afternoon"/>
  <choice dtmf="3" next="#evening"/>
  <choice dtmf="4" next="#thankyou"/>
  <choice dtmf="5" next="#goodbye"/>
</menu>
<form id="morning"><block><assign name="file" expr="'morning.wav'"/><goto
next="#record"/></block></form>
<form id="afternoon"><block><assign name="file" expr="'afternoon.wav'"/><goto
next="#record"/></block></form>
<form id="evening"><block><assign name="file" expr="'evening.wav'"/><goto
next="#record"/></block></form>
<form id="thankyou"><block><assign name="file" expr="'thankyou.wav'"/><goto
next="#record"/></block></form>
<form id="goodbye"><block><assign name="file" expr="'goodbye.wav'"/><goto
next="#record"/></block></form>

<form id="record">
  <record name="msg" dtmfterm="true" beep="true" maxtime="60s"
finalsilence="1500ms" type="audio/x-wav">
    <property name="timeout" value="5s"/>
    <property name="bargain" value="false"/>
    <prompt>
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
        Record message after the beep.
    </prompt>
    <noinput>
        I didn't hear anything, please try again.
    </noinput>
</record>
<filled>
    <prompt>Thank you.</prompt>
    <submit next="record2.php" enctype="multipart/form-data"method="post"
namelist="msg file"/>
    </filled>
</form>
</vxml>
```

Notes:

This script is executed when *42 is dialed on a **maximiser** extension and allows you to record any, or all, of the pre-recorded .wav files supplied with vxmldemo.tar

The voice prompts you will hear are synthesized using ESP's in-built Text To Speech (TTS) engine.

After dialing you will first hear " [Main Recording Menu. Press 1 for Main Menu. Press 2 for Information. Press 9 for general.](#)" If you do not make an entry the menu will be spoken to you again.

If you press "1" you will be played, "[Main Menu. Press 1 for introduction. Press 2 for introduction 2. Press 3 for transfer. Press 4 for sorry.](#)" These relate to the re-recording of main_intro.wav, main_intro2.wav, main_transfer.wav and main_sorry.wav respectively.

If you press "2" you will be played, "[Information Menu. Press 1 for main information.](#)" This relates to the re-recording of the info_intro.wav file.

If you press "3" you will be played, "[General Menu. Press 1 for good morning. Press 2 for good afternoon. Press 3 for good evening. Press 4 for thankyou. Press 5 for goodbye.](#)" These relate to the re-recording of morning.wav, afternoon.wav, evening.wav, thankyou.wav and goodbye.wav respectively.

Pressing a valid key whilst in any of the three menus above moves you on to the recording stage. You will hear "[Record message after the beep.](#)" If you say nothing and silence is detected you will hear "[I didn't hear anything, please try again.](#)" After a successful recording you will hear "[Thank you](#)" after which record2.php will be invoked to handle the file management tasks.

record2.php

```

<?php
ini_set("url_rewriter.tags","");
session_start();
print '<?xml version="1.0" encoding="ISO-8859-1"?'.>';
print "<vxml version=\"2.0\"><form><block>\n";
if (isset($_POST['file'])) $destination = "/Web/vxml/" . $_POST['file'];
else $destination = "/Web/vxml/file_name_missing.wav";
if (move_uploaded_file($_FILES['msg']['tmp_name'], $destination)) {
    print "<prompt>Recording saved.</prompt>";
}
chmod($destination,0666);
print "</block>";
print "<block>";
print "<goto next=\"record.php\" />";
print "</block>";
print "</form>";
print "</vxml>";
?>

```

Notes:

This script is executed when a .wav file has been re-recorded using the record.php script and handles file management tasks.

When a .wav file has been successfully re-recorded and used to replace the existing file you will hear “Recording saved.” record.php will then be re-invoked allowing a second - or subsequent - .wav file to be re-recorded if required.

Multi Level Auto Attendant (multilevel.php)

This script builds on the Simple Auto Attendant Script to deliver Multi-Level capabilities.

```

<?

// this is the basics for a simple Extended Auto Attendant which we hope you can cut and
// paste to your requirements.

//  ini_set("url_rewriter.tags","");

//  session_start();

// the following can be used to select messages dependant on the time of day

    $date = getdate();

    $hour = $date['hours'];

    $greeting = "morning.wav";

    if ($hour > 11) $greeting = "afternoon.wav";

    if ($hour > 17) $greeting = "evening.wav";

// the following line is printed from php to avoid php thinking its script

    print '<.'?xml version="1.0" encoding="ISO-8859-1"?.'>';

?>

<vxml version="2.0">

    <var name="operator" expr="5000"/>

    <form id="main">

        <field name="extn" type="digits?">

            <prompt timeout="5s">

<!-- the following is commented out so it does not play

                <audio src="<?=$greeting?>"/>

-->

```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
<audio src="main_intro.wav"/>
```

```
<audio src="main_intro2.wav"/>
```

```
</prompt>
```

```
<catch event="noinput">
```

```
    <var name="extn" expr="operator"/>
```

```
    <submit next="transfer.php"/>
```

```
</catch>
```

```
<filled>
```

```
    <if cond="extn=='0'">
```

```
        <var name="extn" expr="operator"/>
```

```
    </if>
```

```
    <if cond="extn=='1'">
```

```
        <var name="extn" expr="5000"/>
```

```
    </if>
```

```
    <if cond="extn=='2'">
```

```
        <var name="extn" expr="5004"/>
```

```
    </if>
```

```
    <if cond="extn=='3'">
```

```
        <var name="extn" expr="5018"/>
```

```
    </if>
```

```
    <if cond="extn=='4'">
```

```
        <var name="extn" expr="5009"/>
```

```
    </if>
```

```
    <if cond="extn=='5'">
```

```
        <var name="extn" expr="5014"/>
```

```
    </if>
```

```
    <if cond="extn=='6'">
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
<var name="extn" expr="5001"/>

</if>

<if cond="extn=='7'">

    <var name="extn" expr="5013"/>

</if>

<if cond="extn=='8'">

    <var name="extn" expr="5010"/>

</if>

<if cond="extn=='9'">

    <goto next="multiinfo.php"/>

</if>

<submit next="transfer.php"/>

</filled>

</field>

</form>

</vxml>
```

Notes:

When a call is made into the Extended Auto Attendant there are two messages played. The first (**main_intro.wav**) is the initial introduction which should state something along the following lines;

‘If you know which extension number you require please dial it now.’

There is a slight pause before the next message (**main_intro2.wav**) , which by default will give a short menu - as per options 0-9 below - concluding with:

‘Please hold for the operator’.

If the call is not answered within **5** seconds the call will be forwarded to the operator - defined as extension **5000** in the example above.

The default script is set with the following options:-

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

0. Transfers call to extension 5000
1. Transfers call to 5000 (This is due to the original list starting at option 2 and you would be surprised at how many people just hit 1)
2. Transfers call to 5004
3. Transfers call to 5018
4. Transfers call to 5009
5. Transfers call to 5014
6. Transfers call to 5001
7. Transfers call to 5013
8. Transfers call to 5010
9. Transfers call to the multiinfo.php script

Calls are then transferred using the previously defined **transfer.php** script (see page 13).

Please note that the line `<goto next="mainmenu.php" />` at the bottom of the **transfer.php** script should be changed to read `<goto next="multilevel.php" />` when used with this particular script.

Additionally, all .wav messages used with this script will need recording.

multiinfo.php

```
<?
    print '<'. '?xml version="1.0" encoding="ISO-8859-1"?'. '>';
?>

<vxml version="2.0">

    <menu id="main">

        <property name="inputmodes" value="dtmf"/>

            <prompt timeout="5s">

                <audio src="info_intro.wav"/>

            </prompt>

            <choice dtmf="1" next="#address"/>

            <choice dtmf="2" next="#directions"/>

        </menu>

        <form id="address">

            <block>

                <audio src="address.wav"/>

                <goto next="multilevel.php"/>

            </block>

        </form>

        <form id="directions">

            <block>

                <audio src="directions.wav"/>

                <goto next="multilevel.php"/>

            </block>

        </form>

    </vxml>
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

</block>

</form>

</vxml>

Notes:

If the caller has selected option 9 in multilevel.php (above) they are presented with a further menu as follows:-

1. Company Address (Plays the [address.wav](#) message)
2. Directions (Plays the [directions.wav](#) message)

If no key input is detected after 5 seconds then the [info_intro.wav](#) message is played.

After listening to either of the Company Address or Directions messages the caller is sent back to the top level of multilevel.php again.

Please note that all three .wav messages above need to be recorded.

CLI and DTMF Input Capture (capture.php)

This script captures the CLI of the incoming call (it also checks for those callers that just hang up), then asks the caller to enter a number. These two bits of information are then passed onto a second script that generates a file and places it on the Call Server. This can then be collected by an external application for processing.

```
<vxml application="capture.vxml" version="2.0">

  <form id="test">

    <property name="termtimeout" value="2s"/>

<!-- Setting up variable to collect CLI information -->

    <var name="number" expr="session.telephone.number"/>

<!-- Check if caller hangs up -->

    <catch event="telephone.disconnect.hangup">

      <goto nextitem="submit"/>

    </catch>

<!-- Collect customer information this will accept input of 7 digits -->

    <field name="custno" type="digits?length=7">

      <prompt>

        <audio src="customerno.wav"/>

      </prompt>

    </field>

    <block name="thankyou">

      <audio src="end.wav"/>

    </block>

    <block name="submit">

      <submit next="submit.php"

        namelist="number custno" />

    </block>

```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

</form>

</vxml>

Notes:

This script captures the CLI of an incoming call (in addition it also checks for those callers that just hang up), then asks the caller to enter a customer or PIN number. (In this example it is looking for a seven digit number `<field name="custno" type="digits?length=7">` so please amend accordingly). These two items of information are then passed onto a second script ([submit.php](#)) that actually generates a file and places it on the Call Server, where it can be accessed or collected by external applications.

This script does not “standalone” so it needs to be wrapped around/included with another.

Please note that you will also need to record the required .wav files for the scrip to operate correctly. These should be along the lines of:

“Thank you for calling. Please enter your seven digit customer number or PIN.” ([customerno.wav](#))

“Thank you for entering your details. Please hold whilst we verify” ([end.wav](#))

submit.php

```

<?php

# we always write to file1
# other apps always read from file2 and delete it when finished

$file1 = "/tmp/file1";
$file2 = "/tmp/file2";

$s = $_GET['number'].", "
    .$_GET['custno'].", "

$fp = fopen($file1, "a");
if ($fp)
{
    fputs($fp, $s);
}

if (!file_exists($file2))
{
    rename($file1, $file2);
}

?>
<vxml application="submit.php" version="1.0">
<!--
    <form id="thankyou">
        <block>
            <audio src="thankyou.wav"/>
            <exit/>
        </block>
    </form>
-->
</vxml>

```

Notes:

This script collects the two variables from previous script (capture.php) - customer's CLI and PIN no. - and outputs these to a file on the Call Server in the /ftp/tmp directory (you may need to create the tmp directory under the ftp directory).

The script always writes to file1 and will check if file2 exists, provided file2 is NOT there file1 will be copied to file2.

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

To utilise this information you will need a script that can ftp onto the Call Server and download file2 from the /ftp/tmp directory, and transfer it to your external database or application. Once it has downloaded your script needs to delete file2.

The audio acknowledgment of the file submission has been edited out using the arrow symbols on the script above `<!-- & -->`. If you wish to use this in your script remove these arrows. Once you have done this the .wav file associated with this part of the script will need to be recorded. We'd recommend something along the lines of:

“Thank you. Your information has now been submitted and you may hang-up.” ([thankyou.wav](#))

Text to Speech Conversion (vxlmartin.php)

The following script uses ESP's text to speech capabilities to announce lots of new numbers for staff who had changed offices & numbers, and then forward the call.

```
<?
```

```
// bobg 25may05
```

```
// place this file (martin.php) in /Web/vxml
```

```
// to test create a dial plan with an entry of 33. Dial
!EA:http://127.0.0.1/vxml/martin.php?n=2001
```

```
// and dial 33 to test it.
```

```
// or a DDI entry with a translate to of !EA:http://127.0.0.1/vxml/martin.php?n=2001
```

```
// changing the 2001 as appropriate for the DDI
```

```
// The first time you call the text may be a little slow, however on future attempts it
should be ok as the
text/wav is now in cache.
```

```
// ini_set("url_rewriter.tags","");
```

```
// session_start();
```

```
print '<?xml version="1.0" encoding="ISO-8859-1"?'.>';
```

```
print '<vxml version="2.0">';
```

```
// pickup extension to dial
```

```
$number=$_GET['n'];
```

```
for ($n=0; $n<strlen($number); $n++) $number_with_spaces .= " ".$number[$n];
```

```
// do transfer
```

```
?>
```

```
<form id="transfer">
```

```
<block>
```

```
<prompt>
```

```
Thank you for calling the maximiser.
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

</prompt>

</block>

<transfer name="call" dest="<?=\$number?>">

The person you have called has a new number <?=\$number_with_spaces?>.

You will shortly be transferred to <?=\$number?>.

Please dial it direct next time.

</transfer>

</form>

</vxml>

Wake-Up/Alarm/Reminder Calls (wakeup_vxml.php)

This script allows Wake-Up/Alarm and/or Reminder Calls to be set-up (and cancelled if required) from a **maximiser** handset. ESPs in built Text To Speech engine is used to generate all the voice prompts and acknowledgments in this example.

```
<?php

    // All script inside these brackets (less than question mark and question mark
greater than) is PHP script

    // all other script is VXML

    // ini_set("url_rewriter.tags", "");

    // session_start();

    // error_reporting(0);

    print "<?xml version=\"1.0\" ?>\n";

?>

<!DOCTYPE vxml PUBLIC "SpeechBrowser/2.0/DTD" "ignore" >

<vxml version="2.0">

<?

    // *****

    // Change the following to match your CONFIGURATION

    $filename    = "wakeup_vxml.php";

    $vmpportname = "BuiltIn";

    $vmpportlocation = "Demo Suite Call Server";

    // The following can be used if hosted on the actual call server if not proper
authentication is required

    $server = "127.0.0.1";

    $user_login = "INTERNAL";
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
$user_password = "123456";

$failedtelephonenumber="8000";

$retrymax = "1";

$retryinterval = "30";

// *****

// Now pickup the standard always present parameters (source number and command)

if ($_SERVER['SERVER_PORT']) // We are running from a web server
{
    $cmd = $_REQUEST['cmd'];
    $num = $_REQUEST['n'];
}
else
{
    // We are running from a command line (e.g. crontab)
    // Example use:  php /Web/wakeup_vxml.php submit 307 0700
    // Example use:  php /Web/wakeup_vxml.php cancel 307

    $cmd = $argv[1];
    $num = $argv[2];
    $starttime = $argv[3];

    print "cmd=$cmd num=$num starttime=$starttime\n";
}
}
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
// Now interpret the specific command

if ($cmd == "")
  // DEFAULT Command to Ask what to do - create or cancel

{

  // Lets lookup the name of the calling party

  $ds = LdapConnect($server,$user_login,$user_password);

  $name = LdapGetName($ds,$num);

  $vmport = LdapGetVmPort($ds,$vmportname,$vmportlocation);

  $sr=ldap_search($ds, "guid=".$vmport,
"(&(objectclass=ScheduledCall)(telephoneNumber=".$num."))");

  $o = ldap_get_entries($ds, $sr);

  if ($o['count'] != 0)
  {

    $cur_starttime = $o[0]['starttime'][0];

    $cur_startdate = $o[0]['startdate'][0];

    $date = getdate(time() + (60*60*24)); // Check if set
for tomorrow

    $cur_mday = $date['mday'];

    $cur_month= $date['mon'];

    $cur_year = substr($date['year'],-2,2);

    $startdate = sprintf("%02d%02d%02d",$cur_mday,$cur_month,$cur_year);

    $tomorrow=0;

    if( $startdate == $cur_startdate )
    {

      $tomorrow=1;

    }

  }

}
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
}  
  
    LdapClose($ds);  
  
?>  
  
    <menu id="menu">  
  
        <property name="inputmodes" value="dtmf"/>  
  
        <prompt>hello</prompt>  
  
<?  
  
        SpeakRoom($name,$num);  
  
?>  
  
        <prompt>Welcome to the Splice com White Rock Hotel demonstration  
system.</prompt>  
  
<?  
  
        if( $cur_starttime )  
  
        {  
  
            print "<prompt>You have an alarm set for</prompt>\n";  
  
            SpeakTime($cur_starttime);  
  
            if( $tomorrow )  
  
            {  
  
                print "<prompt>tomorrow</prompt>\n";  
  
            }  
  
        }  
  
        print "<prompt>Press one to setup an alarm call, or two to  
cancel.</prompt>\n";  
  
        print "<choice dtmf=\"1\"  
next=\"\".$filename.\"?cmd=create& n=\".$num.\"\"/>\n";  
  
        print "<choice dtmf=\"2\"  
next=\"\".$filename.\"?cmd=cancel& n=\".$num.\"\"/>\n";  
  
        print "<choice dtmf=\"0\" next=\"#mydisconnect\"/>\n";  
  
?>
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
</menu>
```

```
<form id="mydisconnect">
```

```
  <block>
```

```
    <disconnect/>
```

```
  </block>
```

```
</form>
```

```
<?
```

```
  }
```

```
    else if ($cmd == "create")
```

```
  // CREATE and wakeup call prompt in 24hr format
```

```
  {
```

```
?>
```

```
  <form id="setup">
```

```
    <catch event="telephone.disconnect.hangup">
```

```
      <exit/>
```

```
    </catch>
```

```
    <var name="cmd" expr="'submit'"/>
```

```
<?
```

```
  print "<var name=\"n\" expr=\"'\".\"$num.\"'\"/>\n";
```

```
?>
```

```
  <field name="starttime" type="digits?length=4">
```

```
    <property name="timeout" value="10s"/>
```

```
    <property name="bargain" value="true"/>
```

```
    <prompt>Enter the time in 24 hour format</prompt>
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

<filled>

<?

```
        print '<submit next="'. $filename. '" enctype="multipart/form-data"
method="post" namelist="cmd n starttime"/>\n';
```

?>

</filled>

</field>

</form>

<?

}

```
    else if ($cmd == "submit") // SUBMIT
this form to the database after user enter correct format
```

{

```
    // check for existing scheduled call
```

```
    // if found, delete it
```

```
    if( !isset($starttime) )
```

```
        $starttime = $_REQUEST['starttime'];
```

```
    $startdate="";
```

```
    $start_hour = substr($starttime, -4, 2);
```

```
    $start_min  = substr($starttime, -2, 2);
```

```
    if( $start_hour < 0 || $start_hour > 23 || $start_min < 0 || $start_min > 59 )
```

```
    {
```

?>

```
    <form id="submitfailed">
```

```
        <block>
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
<prompt>I am sorry. That is not a valid time</prompt>
```

```
<?
```

```
print "<goto next=\"\${filename?cmd=create& n=${num}\"/>\n";
```

```
?>
```

```
</block>
```

```
</form>
```

```
<?
```

```
}
```

```
else
```

```
{
```

```
  $ds = LdapConnect($server,$user_login,$user_password);
```

```
  if ($ds)
```

```
  {
```

```
    $vmport = LdapGetVmPort($ds,$vmportname,$vmportlocation);
```

```
    $sr=ldap_search($ds, "guid=".$vmport,  
"(&(objectclass=ScheduledCall)(telephoneNumber=".$num."))");
```

```
    $o = ldap_get_entries($ds, $sr);
```

```
    if ($o['count'] != 0)
```

```
    {
```

```
      $guid = $o[0]['guid'][0];
```

```
      ldap_delete($ds, "guid=".$guid);
```

```
    }
```

```
    // add new scheduled call
```

```
    $date = getdate();
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
$cur_hour = $date['hours'];

$cur_min  = $date['minutes'];

    if( $start_hour < $cur_hour || ( $start_hour == $cur_hour &&
$start_min < $cur_min ) ) // Before now

    {

        $tomorrow = 1;

        $date = getdate( time() + (60*60*24) ); // Get now + 24 hours

        $cur_mday = $date['mday'];

        $cur_month= $date['mon'];

        $cur_year = substr($date['year'],-2,2);

        $startdate =
sprintf("%02d%02d%02d",$cur_mday,$cur_month,$cur_year);

    }

    $attr['objectclass']           = "ScheduledCall";

    $attr['organiser']             = $num;

    $attr['startdate']            = $startdate;

    $attr['starttime']            = $starttime;

    $attr['telephonenumber']      = $num;

    $attr['command']              =
"!EA:http://127.0.0.1/.$filename."?cmd=alarm&n=".$num."&success=1";

    $attr['failtelephonenumber']   = $failedtelephonenumber;

    $attr['failcommand']           =
"!EA:http://127.0.0.1/.$filename."?cmd=alarm&n=".$num."&success=0";

    $attr['retryinterval']        = $retryinterval;

    $attr['retrymax']             = $retrymax;

    $dn = "datetime=".gmdate("ymdHis").",guid=$vimport";
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
        ldap_add($ds, $dn, $attr);

        LdapClose($ds);

    }

?>

<form id="submit">

    <block>

        <prompt>alarm set for</prompt>

<?

        SpeakTime($starttime);

        if( $tomorrow )

        {

            print "<prompt>tomorrow</prompt>\n";

        }

        print "<prompt>thank you</prompt>\n";

?>

    </block>

</form>

<?

    }

}

else if ($cmd == "cancel")
    // CANCEL alarm call on this extension

{

    // check for existing scheduled call

    // if found, delete it

    $ds = LdapConnect($server,$user_login,$user_password);
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
if ($ds)

{

    $name = LdapGetName($ds,$num);

    $vmport = LdapGetVmPort($ds,$vmportname,$vmportlocation);

    $sr=ldap_search($ds, "guid=".$vmport,
"(&(objectclass=ScheduledCall)(telephoneNumber=$num))");

    $o = ldap_get_entries($ds, $sr);

    if ($o['count'] != 0)

    {

        $guid = $o[0]['guid'][0];

        ldap_delete($ds, "guid=$guid");

    }

    LdapClose($ds);

}

?>

<form id="cancel">

    <block>

        <prompt>alarm cancelled for</prompt>

    <?

        SpeakRoom($name,$num);

    ?>

    </block>

</form>

<?

}

else if ($cmd == "alarm")
    // ALARM has gone off
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
{  
  
    $status = $_REQUEST['success'];          // check if good or bad  
  
    $ds=LdapConnect($server,$user_login,$user_password);  
  
    $name = LdapGetName($ds,$num);  
  
    LdapClose($ds);  
  
    if ( $status == "1")  
    {  
  
?>  
  
        <form id="alarm">  
  
            <block>  
  
?<?>  
  
                for($i=1; $i < 5; $i++)  
  
                {  
  
                    print "<prompt> this is your requested alarm call for </  
prompt>\n";  
  
                    SpeakRoom($name,$num);  
  
                }  
  
?>  
  
                <prompt>Goodbye</prompt>  
  
                <disconnect/>  
  
            </block>  
  
        </form>  
  
?<?>  
  
    }  
  
    else  
  
    {  
  
?>
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
<form id="alarmfail">
```

```
  <block>
```

```
<?
```

```
    for($i=1; $i < 5; $i++)
```

```
    {
```

```
      print "<prompt> alarm call for </prompt>\n";
```

```
        SpeakRoom($name,$num);
```

```
        SpeakRoom("", $num);
```

```
      print "<prompt> has not answered </prompt>\n";
```

```
    }
```

```
?>
```

```
      <prompt>Goodbye</prompt>
```

```
    <disconnect/>
```

```
  </block>
```

```
</form>
```

```
<?
```

```
  }
```

```
}
```

```
else
```

```
{
```

```
?>
```

```
  <form>
```

```
    <block>
```

```
      <prompt>I am sorry I do not understand the command</prompt>
```

```
<?
```

```
        print "<prompt>${cmd}</prompt>";
```

```
?>
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
</block>
```

```
</form>
```

```
<?
```

```
}
```

```
?>
```

```
</vxml>
```

```
<?
```

```
// The following code is pure PHP and does not contain any VXML
```

```
// Helpful functions to make the above code easier
```

```
function LdapConnect($server,$user_login,$user_password)
```

```
{
```

```
    $ds=ldap_connect($server.":4000"); // must be a valid LDAP server!
```

```
    if ($ds)
```

```
    {
```

```
        $r=ldap_bind($ds, $user_login, $user_password);
```

```
        if (!$r)
```

```
        {
```

```
            ldap_close($ds);
```

```
            $ds = 0;
```

```
        }
```

```
    }
```

```
    return $ds;
```

```
}
```

```
function LdapClose($ds)
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
{
    ldap_close($ds);
}

function LdapGetName($ds,$num)
{
    $name = "";

    if ($num != "" && $ds)
    {
        $myAttr = array("cn");          // The specific attributes we need - this will
increase performance since a user can have a lot of attributes

        $sr=ldap_search($ds, "", "(GlobalExtnNumber=".$num.")",$myAttr,0,1); //
Optimized special key search

        $o = ldap_get_entries($ds, $sr);

        if ($o['count'] != 0)          // If we have some answers
        {
            $name = $o[0]['cn'][0];    // Extract the name (common name
= cn) attribute
        }
    }

    return $name;
}

function LdapGetVmPort($ds,$vmportname,$vmportlocation)
{
    if( $vmportlocation )
    {
        $sr=ldap_read($ds,
"cn=$vmportlocation,cn=Modules", "(objectclass=Module)", array("guid","cn"),0,1);
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
$o = ldap_get_entries($ds, $sr);

if ($o['count'] != 0)

{

    $location = $o[0]['guid'][0];

}

}

else

{

    $location = "00000000-0000-0000-0000-000000000000";

}

if( $location )

{

    $sr=ldap_list($ds,
"cn=VoicemailPorts", "(&(location=$location)(cn=$vmportname))", array("guid", "cn"), 0, 0);

    $o = ldap_get_entries($ds, $sr);

    if ($o['count'] != 0)

    {

        $vmport = $o[0]['guid'][0];

        return $vmport;

    }

}

return "notfound";

}

function SpeakRoom($name,$num)

{
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
if( $name!="" )
{
    print "<prompt>$name</prompt>\n";
}
else
{
    print "<prompt>Room number </prompt>\n";
    for ( $i=0; $num[ $i ] != ""; $i++)
    {
        print "<prompt>". $num[ $i ]. "</prompt>\n";
    }
}
}
```

```
function SpeakTime($t)
{
    if( strlen($t) == 3 )
    {
        $t="0".$t;
    }
    else if( strlen($t) == 2 )
    {
        $t="00".$t;
    }
    $hour = substr($t, -4, 2);
    $min = substr($t, -2, 2);
    if( $min == "00" )
```

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

```
{  
  
    $min = " hundred hours";  
  
}  
  
print "<prompt>$hour</prompt>\n";  
  
print "<prompt>$min</prompt>\n";  
  
}  
  
?>
```

Notes:

Although this script is quite long, it can be easily customised to meet your customers' specific requirements. Lots of comments have been included to show you exactly what is going on.

The following three items (within the quotation marks) must match the name of the vxml.php file, the Voicemail Port "Name" and the "Location" where the Voicemail application is running, respectively:

The screenshot shows a configuration window for a Voicemail Port. The window has a title bar with 'Builtin' and several icons. The main area is a table with the following fields:

Name	Builtin
Description	Internal Voicemail Ports
Location	Demo Suite Call Server
H323 Address	CallServer 00-07-d9-00-03-cc
Capacity	2
Email Smart Host	192.168.3.101
Email Source Address	vmail@maximiser.local
Product Version	2.2(744)
Dial Plan	Standard
Remote Location	
ESP Capacity	1

Below the table, there are several checkboxes, all of which are checked:

CapabilityAA	<input checked="" type="checkbox"/>	CapabilityVM	<input checked="" type="checkbox"/>
CapabilityMusic	<input checked="" type="checkbox"/>	CapabilityQueue	<input checked="" type="checkbox"/>
CapabilityRecording	<input checked="" type="checkbox"/>	CapabilityConference	<input checked="" type="checkbox"/>
CapabilityEA	<input checked="" type="checkbox"/>		

```
$filename = "wakeup_vxml.php";  
  
$vmportname = "BuiltIn";  
  
$vmportlocation = "Demo Suite Call Server";
```

The failed telephone number is the extension where you want the Guest/User's call to be directed to if the alarm call is not answered. The retry max parameter determines the maximum number of attempted retries

Enhanced Speech Processing (ESP) for **maximiser** - An Introduction

before the call is routed to the number above, whilst the retry interval in (seconds) determines the delay before the system attempts to place the alarm call again.;

```
$failedtelephonenumber="8000";
```

```
$retrymax = "1";
```

```
$retryinterval = "30";
```

Change the following prompt to the greeting of your choice;

```
<prompt>Welcome to the Splice com White Rock Hotel demonstration system.</prompt>
```

Appendices

Appendix i - Further Reading

You can find more details on the VoiceXML language at <http://www.w3.org/TR/voicexml20/>

Also recommend is the book "Definitive Voice XML by Adam Hocek and David Cuddihy ISBN 0-13-046345-0.



SpliceCom Limited The Hall Business Centre, Berry Lane, Chorleywood, Hertfordshire WD3 5EX
Tel: 01923 287700 Fax: 01923 287722 Email: info@splicecom.com Website: www.splicecom.com