Using Lutris Enhydra with VoiceXML

Authored 2/14/2001 by Bob Bourbonnais for Lutris Technologies, Inc.

Summary

This white paper provides the information you need to get started using Lutris® Enhydra™ and VoiceXML together. It includes four main sections. The first section provides a general overview of voice applications and technology, and serves as a foundation for the more detailed information that follows. The second section provides more specific information on using Enhydra with VoiceXML. The third section explains how to create a simple VoiceXML application. Finally, the fourth section includes two appendixes: one containing the VoiceXML file, `Welcome.vxml`, used in the simple VoiceXML application, and the other containing the DTD file, `enhydra-voicexml1-0.dtd`, used with the application.

Table of Contents

Overview

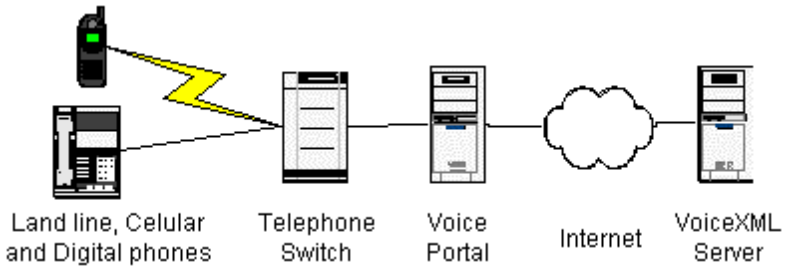
This section provides basic information about voice technology and applications.

History

On March 7, 2000, the VoiceXML Forum released the VoiceXML 1.0 specification. VoiceXML is based on Extensible Markup Language (XML), a metalanguage created by

the World Wide Web Consortium (W3C). XML provides a standardized programming interface for speech and telephony applications.
During its May 10-12, 2000 meetings, the Voice Browser Working Group of the W3C agreed to adopt VoiceXML 1.0 as the basis for the development of a W3C dialog markup language.

Voice Application Architecture



Voice applications typically require multiple components. For example, as depicted in the figure, when a consumer uses a telephone to access a voice application, a range of components are involved in the process. In this case, the telephone dials into a special telephone number that connects through a telephone switch into a voice portal. (A *voice portal* is a hardware device that connects a telephone line to the Internet and provides speech recognition, text-to-speech, speech recording, and digital sound playback functionality.) The voice portal then interfaces over the Internet to a VoiceXML server, which provides the actual application that the user is accessing through the other interfaces. The VoiceXML server provides VoiceXML to the voice portal and the business logic and database access. Multiple machines can replace the single VoiceXML server to distribute the layers, the load, and the functionality.
IBM, Nuance, L&H, and SpeechWorks are just some of the providers of voice portal software. Voice portals are expensive to set up and maintain, which is why so many third-party companies provide voice portal services. Examples include TellMe, BeVocal, and Voxeo.

Grammar

Most speech recognizers match up a recorded speech utterance with a list of words. A *grammar* defines the way the utterances are matched with the speech-recognition rules. Because speech recognizers and their grammars existed prior to the VoiceXML specification, the grammar can map into the specification in different ways, depending on vendor, voice portal service, and VoiceXML emulator. Additional features such as menu operations are dependent on and tied to the grammar. For this reason, writing VoiceXML for Enhydra is different from writing VoiceXML for Voxeo, IBM, or TellMe.

Data Type Definition files

Data Type Definition (DTD) files are used to define the elements and attributes of markup languages. Because of the differences between grammars, their associative changes, and other changes, most voice portal software and sites recommend a specific DTD. For example, the IBM emulator supports `voicexml1-0.dtd`, Voxeo supports `nuancevoicexml.dtd`, and Tellme supports `vxml-tellme.dtd`. The Enhydra DTD, `enhydra-`

`voicexml1-0.dtd`, is available in Appendix B. By examining the files and the online help, you can see that the grammar and even the menu command itself can vary from DTD to DTD. For example, Nuance doesn't have the menu command and wants all grammars in an external file.

Understanding how Enhydra and VoiceXML work together

Enhydra is an application server that can dynamically modify VoiceXML documents at runtime. Enhydra enables a voice application to change VoiceXML documents "on the fly" as it accesses business logic and database information. In addition, Enhydra supports multiple simultaneous protocols, including Hypertext Markup Language (HTML), Wireless Markup Language (WML), compact HTML (cHTML), and Extensible Markup Language (XML). A single application can support all these different protocols, using the same business and data layer code—all you have to do is create markup files for each protocol. In the same way, a voice application can support different types of VoiceXML and thus different voice portals, with the same business and data layer code.

Using Enhydra XMLC to check VoiceXML against a DTD

Enhydra XMLC™ checks the VoiceXML file (or any markup language files) against a specified DTD before translating it to a Java Document Object Model (DOM). This enables VoiceXML developers to check the format before they do their testing. Otherwise, many VoiceXML developers don't discover there are format problems until they run their VoiceXML. The test and fix cycle can be long and arduous with an emulator, and even longer if developers have to upload their file each time they want to test it. Enhydra XMLC solves this problem with its built-in format checking.

Using Enhydra XMLC to translate VoiceXML to a DOM

Enhydra XMLC translates VoiceXML (or any other markup language) files to a DOM Java file. The Document Object Model (DOM) is a World Wide Web Consortium (W3C) specification. Once created, the DOM is easily accessed and modified from any other Java file. At runtime, a request to the application server triggers the DOM file to be converted back into its associated markup language file carrying along any changes the program made to the DOM.

DOM Accessor methods

In addition to creating the DOM, XMLC can sense ID tags in the markup language and provide accessor methods in the DOM Java file. This makes it possible to change the attributes easily. Without the accessor methods, the items can still be modified, using a more complex DOM-tree walking methodology.

Enhydra extension to the DTD

Because ID tags are required in order to create the accessor methods, ID tags were added to the Nuance and VoiceXML forum DTD files. The extended DTDs are provided on the Enhydra website as `enhydra-nuance-voicexml.dtd` and `enhydra-voicexml1-0.dtd` at http://www.enhydra.org/xml/voicexml. (`enydra-voicexml1-0.dtd` is included as

Appendix B.) You can make similar modifications to any DTD when you want XMLC to be able to add accessor methods.

Creating a simple VoiceXML example

The example you will create in this section dynamically updates the file `Welcome.vxml` so that, when accessed, it will tell the current time.

Note: This procedure assumes that you know how to use Enhydra, so it does not explain all the details involved in each step. You should know how to create an Enhydra super-servlet application and a simple Wireless Markup Language (WML) application before you attempt this process. For your convenience, the completed example is provided in the `<enhydra_root>/examples/SimpleVXML` directory. The completed example should match your results for performing steps 1 through 6 in the procedure for creating a simple VoiceXML example below. To use the provided example, follow the procedure below, starting with step 7.

Before you begin

The following example uses IBM's WebSphere Voice Server™ SDK to test the SimpleVXML application. The Voice Server SDK is freely available from IBM with your registration in the IBM developers program. The Voice Server SDK has the following system requirements:

- Pentium 366 MHz processor
- 128 MB RAM
- 200 MB disk space
- Microsoft™ Windows NT™ compatible 16-bit full-duplex sound card
- Microsoft Windows NT Workstation or Server 4.0 (English language version only), with Service Pack 6a or later applied
- Sun Java Runtime Environment (Sun JRE) 1.3.0 (included in the SDK, but must be installed prior to the IBM WebSphere Voice Server SDK software)

For complete system requirements and more information , go to the WebSphere Voice Server SDK page at http://www-4.ibm.com/software/speech/enterprise/ep_12.html.

To create a simple VoiceXML example:

1) Use the Enhydra Application Wizard (`appwizard`) to create a simple WML super-servlet application named SimpleVXML.
   a) Select Enhydra Super-Servlet from the Generator pull-down menu and click Next.
   b) Select WML from the Client type pull-down menu.
   c) Enter S`impleVXML` for the Project directory name.

d) Enter `simplevxml` (note the difference in case) for Package and click Finish.

For more information, refer to Chapter 2, "Using the Application Wizard to create Enhydra applications" in the *Developer's Guide*.

2) Delete the `Welcome.wml` file from the project and add the `Welcome.vxml` file from Appendix A.

3) Update `WelcomePresentation.java` in the `<SimpleVXML_root>/src/simplevxml/presentation` directory.

e) Change all instances of `WelcomeWML` to `WelcomeVoiceXML`.

f) Ensure the `DateFormat` class is using a `SHORT` format.

4) Update `Makefile` in the `<SimpleVXML_root>/src/simplevxml/presentation` directory to include the `VOICEXML_DIR` and the `VOICEXML_CLASSES`.

a) Remove the following lines:

```
WML_CLASSES = WelcomeWML
WML_DIR = .
```

b) Add the following lines:

```
VOICEXML_DIR = .
VOICEXML_CLASSES = WelcomeVoiceXML
```

c) Remove the following line:

```
XMLC_WML_OPTS_FILE = options.xmlc
```

5) Remove `options.xmlc` from the `<SimpleVXML_root>/src/simplevxml/presentation` directory.

6) Add the following lines to the end of the `config.mk` file in `<SimpleVXML_root>` to include the `voicexml.mk` file from Enhydra `lib` directory.

```
ifneq ($(wildcard $(ENHYDRA_DIR)/lib/voicexml.mk),)
    include $(ENHYDRA_DIR)/lib/voicexml.mk
endif
```

7) Enter `make` in `<SimpleVXML_root>` to compile build the application.

8) Enter the following commands to start the application:

```
cd output
./start
```

9) Install the IBM Emulator available from http://www6.software.ibm.com/dl/websphere02/wsvsvrdk-p (62.1 MB). Note: You will need to join the IBM developer program to get the free download.

10) Modify your path to point to the `/bin` directory created when you installed the IBM software.

11) Access the demo from the command line by entering `vsaudio http://localhost:9000/`, where 9000 is the specified port.

Notice that the time changes, showing that the field has been updated from within the application.

Appendix A Welcome.vxml

This appendix contains the `welcome.vxml` file used in the previous section, "Creating a simple VoiceXML example."

```
<?xml version="1.0"?>
<!DOCTYPE vxml
    PUBLIC "-//ENHYDRA//DTD VXML 1.0 + Enhydra Ids//EN"
    "http://www.enhydra.org/xml/voicexml/enhydra-voicexml1-0.dtd">

<vxml version="1.0">
  <form>
    <block>Welcome to voice XML from Lutris Technologies</block>
    <block>The time is now </block>
    <block id="time">00:00:00</block>
  </form>
</vxml>
```

Appendix B enhydra-voicexml1-0.dtd

This appendix contains the `enhydra-voicexml1-0.dtd` file that is referred to in previous sections of this white paper.

```
<!--
 -
 - A DTD for Voice Extensible Markup Language
 - as currently supported by the Lutris Enhydra Platform
 - based on the official v1.0 VXML DTD
 -
 - Copyright (c) 2000 VoiceXML Forum (AT&T, IBM, Lucent Technologies, Motorola)
 - http://www.vxml.org
 -
 - The Lutris VXML DTD is Copyright (C) 2001 Lutris Technologies, Inc.
 - http://www.lutris.com
 -
 - We needed to create this extension to allow ID tagging of entities
 - Last revised 2/12/2001 by Bob Bourbonnais <bob.bourbonnais@lutris.com>
 -
 - Use:
 -   <!DOCTYPE vxml PUBLIC "-//ENHYDRA//DTD VXML 1.0 + Enhydra Ids//EN"
 -             "http://www.enhydra.org/xml/voicexml/enhydra-voicexml1-0.dtd">
 -
 - $Revision: 1.1.2.5 $
 -->

<!ENTITY % enhydraid  "id    ID     #IMPLIED">
<!ENTITY % audio "#PCDATA | audio | enumerate | value" >
```

```
<!ENTITY % boolean  "(true|false)" >
<!ENTITY % content.type "CDATA">
<!ENTITY % duration "CDATA" >
<!ENTITY % event.handler "catch | help | noinput | nomatch | error" >
<!ENTITY % event.name "NMTOKEN" >
<!ENTITY % event.names "NMTOKENS" >
<!ENTITY % executable.content
      "%audio; | assign | clear | disconnect | exit | goto | if | prompt |
      reprompt | return | script | submit | throw | var " >
<!ENTITY % expression "CDATA" >
<!ENTITY % field.name "NMTOKEN" >
<!ENTITY % field.names "NMTOKENS" >
<!ENTITY % integer "CDATA" >
<!ENTITY % item.attrs
      "name        %field.name; #IMPLIED
      cond         %expression; #IMPLIED
      expr         %expression; #IMPLIED " >
<!ENTITY % uri "CDATA" >
<!ENTITY % cache.attrs
      "caching     (safe|fast)  #IMPLIED
      fetchhint    (prefetch|safe|stream)    #IMPLIED
      fetchtimeout %duration;   #IMPLIED " >
<!ENTITY % next.attrs
      "next        %uri;        #IMPLIED
      expr         %expression; #IMPLIED " >
<!ENTITY % submit.attrs
      "method             (get|post)   'get'
      enctype             %content.type;      'application/x-www-form-
urlencoded'
      namelist     %field.names;#IMPLIED" >
<!ENTITY % tts             "break | div | emp | pros | sayas" >
<!ENTITY % variable "block | field | var" >

<!--================================= Root ================================-->

<!ELEMENT vxml
      (%event.handler; | form | link | menu | meta |
      property | script | var)+ >
<!ATTLIST vxml
      application  %uri;        #IMPLIED
      base         %uri;        #IMPLIED
      lang         CDATA        #IMPLIED
      version             CDATA        #REQUIRED >
<!ELEMENT meta             EMPTY >
<!ATTLIST meta
      name         NMTOKEN             #IMPLIED
      content             CDATA        #REQUIRED
      http-equiv   NMTOKEN             #IMPLIED
       %enhydraid; >

<!--================================= Dialogs ============================-->

<!ENTITY % input    "dtmf | grammar" >
<!ENTITY % scope    "(document | dialog)" >
<!ELEMENT form
      (%input; | %event.handler; | filled | initial | object | link | property
      | record | subdialog | transfer | %variable;)* >
<!ATTLIST form
      id           ID           #IMPLIED
```

```
      scope        %scope;             'dialog' >
<!ELEMENT menu
      (%audio; | choice | %event.handler; | prompt | property)* >

<!ATTLIST menu
      id           ID           #IMPLIED
      scope        %scope;             'dialog'
      dtmf         %boolean;    'false' >
<!ELEMENT choice   (%audio; | grammar | %tts;)* >
<!ATTLIST choice
      %cache.attrs;
      dtmf         CDATA        #IMPLIED
      event        %event.name; #IMPLIED
      fetchaudio   %uri;        #IMPLIED
      %next.attrs;
       %enhydraid; >

<!--================================ Prompts ================================-->

<!ELEMENT prompt   (%audio; | %tts;)* >
<!ATTLIST prompt
      bargein            %boolean;    #IMPLIED
      cond         %expression; #IMPLIED
      count        %integer;    #IMPLIED
      timeout            %duration;   #IMPLIED
       %enhydraid; >
<!ELEMENT enumerate (%audio; | %tts;)*>
<!ELEMENT reprompt  EMPTY >

<!--================================ Fields ================================-->

<!ELEMENT field
      (%audio; | %event.handler; | filled | %input; | link | option | prompt |
property)* >
<!ATTLIST field
      %item.attrs;
      type         CDATA        #IMPLIED
      slot         NMTOKEN             #IMPLIED
      modal        %boolean;    'false'
       %enhydraid; >
<!ELEMENT option   (#PCDATA)* >
<!ATTLIST option
      dtmf         CDATA        #IMPLIED
      value        CDATA        #IMPLIED
       %enhydraid;>
<!ELEMENT var      EMPTY >
<!ATTLIST var
      name         %field.name; #REQUIRED
      expr         %expression; #IMPLIED
       %enhydraid;>
<!ELEMENT initial  (%audio; | %event.handler; | link | prompt | property)* >
<!ATTLIST initial
      %item.attrs;
       %enhydraid; >
<!ELEMENT block    (%executable.content;)* >
<!ATTLIST block
      %item.attrs;
       %enhydraid; >
<!ELEMENT assign   EMPTY >
```

```
<!ATTLIST assign
       name          %field.name; #REQUIRED
       expr          %expression; #REQUIRED
        %enhydraid; >
<!ELEMENT clear      EMPTY >
<!ATTLIST clear
       namelist      %field.names;#IMPLIED
        %enhydraid; >
<!ELEMENT value      EMPTY >
<!ATTLIST value
       class         CDATA        #IMPLIED
       expr          %expression; #REQUIRED
       mode          (tts|recorded)      "tts"
       recsrc        %uri;        #IMPLIED
        %enhydraid; >

<!--================================== Events ============================-->

<!ENTITY % event.handler.attrs
       "count        %integer;    #IMPLIED
       cond          %expression; #IMPLIED">
<!ELEMENT catch      (%executable.content;)* >
<!ATTLIST catch
       event         %event.names;#REQUIRED
       %event.handler.attrs;
        %enhydraid; >
<!ELEMENT error      (%executable.content;)* >
<!ATTLIST error
       %event.handler.attrs;
        %enhydraid; >
<!ELEMENT help             (%executable.content;)* >
<!ATTLIST help
       %event.handler.attrs;
        %enhydraid; >
<!ELEMENT link             (dtmf | grammar)* >
<!ATTLIST link
       %cache.attrs;
       %next.attrs;
       fetchaudio    %uri;        #IMPLIED
       event         %event.name; #IMPLIED
        %enhydraid; >
<!ELEMENT noinput    (%executable.content;)* >
<!ATTLIST noinput
       %event.handler.attrs;
        %enhydraid; >
<!ELEMENT nomatch    (%executable.content;)* >
<!ATTLIST nomatch
       %event.handler.attrs;
        %enhydraid; >
<!ELEMENT throw      EMPTY >
<!ATTLIST throw
       event         %event.name; #REQUIRED
        %enhydraid; >

<!--============================== Audio Output ==========================-->

<!ELEMENT audio      (%audio; | %tts;)* >
<!ATTLIST audio
       src           %uri;        #IMPLIED
```

```
       %cache.attrs;
        %enhydraid; >
<!ELEMENT break     EMPTY >

<!ATTLIST break
       msecs        %integer;    #IMPLIED
       size         (none|small|medium|large) #IMPLIED
        %enhydraid; >
<!ELEMENT div       (%audio; | %tts;)* >
<!ATTLIST div
       type         CDATA  #IMPLIED
        %enhydraid; >
<!ELEMENT emp       (%audio; | %tts;)* >
<!ATTLIST emp
       level        (strong | moderate | none | reduced) "moderate" >
<!ELEMENT pros             (%audio; | %tts;)* >
<!ATTLIST pros
       rate         CDATA        #IMPLIED
       vol          CDATA        #IMPLIED
       pitch        CDATA        #IMPLIED
       range        CDATA        #IMPLIED
        %enhydraid; >
<!ELEMENT sayas     (#PCDATA)* >
<!ATTLIST sayas
       sub          CDATA        #IMPLIED
       class        CDATA        #IMPLIED
       phon         CDATA        #IMPLIED
        %enhydraid; >

<!--============================== Audio Input ==============================-->

<!ENTITY % key      "CDATA" >
<!ENTITY % grammar.attrs
       "%cache.attrs;
       scope        %scope;             #IMPLIED
       src          %uri;        #IMPLIED
       type         CDATA        #IMPLIED " >
<!ELEMENT dtmf             (#PCDATA)* >
<!ATTLIST dtmf
       %grammar.attrs;
        %enhydraid; >
<!ELEMENT grammar   (#PCDATA)* >
<!ATTLIST grammar
       %grammar.attrs;
        %enhydraid; >
<!ELEMENT record
       (%audio; | %event.handler; | filled | grammar | prompt | property)* >
<!ATTLIST record
       %item.attrs;
       type         CDATA        #IMPLIED
       beep         %boolean;    'false'
       maxtime             %duration;   #IMPLIED
       modal        %boolean;    'true'
       finalsilence %duration;   #IMPLIED
       dtmfterm     %boolean;    'true'
        %enhydraid; >

<!--============================ Call Control ============================-->
```



```
<!ELEMENT disconnectEMPTY >
<!ELEMENT transfer
      (%audio; | %event.handler; | dtmf | filled | grammar | prompt |
property)* >
<!ATTLIST transfer
      %item.attrs;
      dest          %uri;         #IMPLIED
      destexpr      %expression;  #IMPLIED
      bridge        %boolean;     'false'
      connecttimeout      %duration;    #IMPLIED
      maxtime             %duration;    #IMPLIED
       %enhydraid; >

<!--============================ Control Flow ============================-->

<!ENTITY % if.attrs
      "cond         %expression;  #REQUIRED" >
<!ELEMENT if        (%executable.content; | elseif | else)* >
<!ATTLIST if
      %if.attrs;
       %enhydraid; >
<!ELEMENT elseif    EMPTY >
<!ATTLIST elseif
      %if.attrs;
       %enhydraid; >
<!ELEMENT else      EMPTY >
<!ELEMENT exit              EMPTY >
<!ATTLIST exit
      expr          %expression;  #IMPLIED
      namelist      %field.names;#IMPLIED >
<!ELEMENT filled    (%executable.content;)* >
<!ATTLIST filled
      mode          (any|all)     "all"
      namelist      %field.names;#IMPLIED
       %enhydraid; >
<!ELEMENT goto              EMPTY >
<!ATTLIST goto
      %cache.attrs;
      %next.attrs;
      fetchaudio    %uri;         #IMPLIED
      expritem      %expression;  #IMPLIED
      nextitem      %field.name;  #IMPLIED
       %enhydraid; >
<!ELEMENT param     EMPTY >
<!ATTLIST param
      name          NMTOKEN             #REQUIRED
      expr          %expression;  #IMPLIED
      value         CDATA         #IMPLIED
      valuetype     (data|ref)    'data'
      type          CDATA         #IMPLIED
       %enhydraid; >
<!ELEMENT return    EMPTY >
<!ATTLIST return
      namelist      %field.names;#IMPLIED
      event         %event.name;  #IMPLIED
       %enhydraid; >
<!ELEMENT subdialog
      (%audio; | %event.handler; | filled | param | prompt | property)* >
<!ATTLIST subdialog
```

```
       %item.attrs;
       src          %uri;         #REQUIRED
       %cache.attrs;
       fetchaudio   %uri;         #IMPLIED
       %submit.attrs;
        %enhydraid; >
<!ELEMENT submit    EMPTY >
<!ATTLIST submit
       %cache.attrs;
       %next.attrs;
       fetchaudio   %uri;         #IMPLIED
       %submit.attrs;
        %enhydraid; >

<!--========================== Miscellaneous ==============================-->

<!ELEMENT object
       (%audio; | %event.handler; | filled | param | prompt | property)* >
<!ATTLIST object
       %item.attrs;
       %cache.attrs;
       classid             %uri;         #IMPLIED
       codebase     %uri;         #IMPLIED
       data         %uri;         #IMPLIED
       type         CDATA         #IMPLIED
       codetype     CDATA         #IMPLIED
       archive             %uri;         #IMPLIED
        %enhydraid; >
<!ELEMENT property  EMPTY >
<!ATTLIST property
       name         NMTOKEN             #REQUIRED
       value        CDATA         #REQUIRED
        %enhydraid; >
<!ELEMENT script    (#PCDATA)* >
<!ATTLIST script
       src          %uri;         #IMPLIED
       charset             CDATA         #IMPLIED
       %cache.attrs;
        %enhydraid; >
```