FD – Application and Media Content Package Manager

Content

History

Version	Date	Adjustments
A	2006-10-05	Initial revision

1 Introduction

The purpose of this document is to explain the internal structure and functions of the Application and Media Content Package Manager component (AMCPM). See ref. [1] for specifications of the Application and Media Content Package Manager functions.

2 Function Structure

The AMCPM consists of the following modules

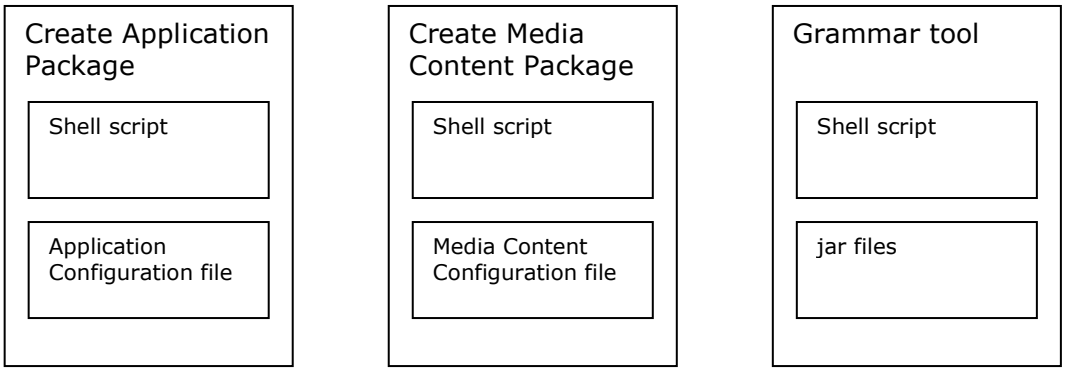

Figure 1 Function structure

The modules are described in subsections below.

2.1 Create Application Package

This part consists of a shell script and a configuration file.

The configuration file has the following information:

- VoiceXML file location
- CCXML file location
- ECMA file location
- Configuration file location
- Event definition file location
- Event template file location
- Number analysis file location
- The name of the Application
- The name of the customer
- Product identity
- R-state
- For each service supported by this Application
    - Service name
    - The CCXML file that should be used for this
    - The protocol to use to access the service

- The protocol port to use to access the service

The shell script does the following:

1. Interpret the configuration file.
2. Copy Application specific files i.e. VoiceXML, CCXML, ECMA, Application configuration, Number analyses, Event definitions and Event template files.
3. Create a delivery file that MAS can install. This file is a simple tar-file with well defined directory structure.

2.2 Create Media Content Package

This part consists of a shell script and a configuration file.

The configuration file has the following information:

- Media files location
- Media Content files location
- Media Object files location
- Grammar file location
- The name
- The package type
- The name of the customer
- Product identity
- R-state
- Language
- Video variant (if any)
- Voice variant (if any)
- Audio encoding
- Video encoding

The shell script does the following:

1. Interpretation of the configuration file.
2. Copy Media specific files i.e. MediaContent definition, MediaObject definition, media and grammar files.
3. Create a delivery file that MAS can install. This file is a simple tar-file with well defined directory structure.

2.3 Grammar tool

This part consists of a shell script and jar-files.

The following jar-files are included:

- *mobeon_media_content_manager.jar* - The grammar tool class and classes for number decomposition.
- *mobeon_logging.jar* - Classes for logging.
- *log4j-1.2.9.jar* - Classes for logging.

The shell script does the following:

1. Set up classpath
2. Start the JVM with the grammar tool

3 Function Behavior

3.1 Creation of an Application Package

When the shell script for creation of an Application Package is used the following happens:

1. The configuration file is opened and read. Either the file is provided as a command line argument or if not a default filename is used. The configuration file contains the characteristics about the application to be created such as the path to where VoiceXML and CCXML files are located.
2. A temporary directory structure is created in the filesystem and the Application files are copied to that structure.
3. Relevant parts for the installation on MAS described in the configuration file, is copied to a file in the directory structure
4. The entire directory structure is archived using tar and the delivery filename is created based on the information in the configuration file.

3.2 Creation of a Media Content Package

When the shell script for creation of a Media Content Package is used the following happen:

1. The configuration file is opened and read. Either the file is provided as a command line argument or if not a default filename is used.
2. A temporary directory structure is created in the file system and the Media content related files are copied to that structure.
3. Relevant parts for the installation on MAS described in the configuration file, is copied to a file in the directory structure
4. The entire directory structure is archived using tar and the delivery filename is created based on the information in the configuration file.

3.3 Test a grammar file

The grammar tool is used to verify the number rules in a Media Content Package grammar file. The tool takes two parameters, one grammar file and one data input file:

grammarFileTool.sh <grammarfile> <input data file>

The syntax for the data input file is shown in Figure 2.

```
// Grammar tool input file example
// syntax: <number>;<rule>;<gender>
102;Number;Male
2006-09-29;DateDM;None
23:59:00;Time12;None
```

Figure 2 Grammar tool data input file

For each line in the data input file, the grammar tool decomposes the number using the corresponding rule from the grammar file and the number decomposition Java classes from *mobeon_media_content_manager.jar*. The result is printed on standard output. Lines beginning with "//" are treated as comments and are ignored by the parser.

An example of the output from the input file in Figure 2 is shown in Figure 3.

```
102          1 100 2
2006-09-29   sep dom29
23:59:00     11 50 9 pm
```

Figure 3 Example output

4 References

[1] FS – Application and Media Content Package Manager 1/FS-SWU0044 Uen

5 Terminology

Intentionally left blank.