Assignment No: C2

Roll No: 4430

1 Title

Cross compilation using XMLVM..

2 Problem Definition

Cross compilation using XMLVM.

3 Objective

1 . To understand the need for cross compilation 2. To learn about XMLVM

4 Pre-requisite

1. Basic knowledge of platform independence 2. Knowledge of compier

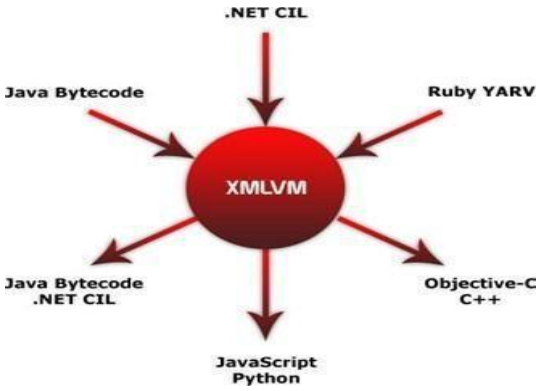
5 Theory XMLVM

The goal of XMLVM is to offer a flexible and extensible crosscompiler toolchain. Instead of cross-compiling on a source code level, XMLVM cross-compiles byte code instructions from Sun Microsystem's virtual

machine and Microsoft's Common Language Runtime. The benefit of this approach is that byte code instructions are easier to cross-compile and the difficult parsing of a high-level programming language is left to a regular compiler. In XMLVM, byte codebased programs are represented as XML documents. This allows manipulation and translation of XMLVM-based programs using advanced XML technologies such as XSLT, XQuery, and XPath.

XMLVM serves as an umbrella for several projects. For all projects, a Java class file or a .NET executable is first translated to an XMLdocument. Based on the XML-document generated by the frontend, various transformations are possible. The first transformation crosscompiles from .NET to JVM byte code. Another transformation enables Java or .NET applications to be cross-compiled to JavaScript so that they can run as AJAX applications in any browser. Yet another transformation allows to cross-compile a Java program to Objective-C to creat e a native iPhone application.

You can even cross compile an Android application to run on the iPhone



Compiling XMLVM

XMLVM depends on numerous third-party libraries such as BCEL, JDOM, and Saxon. All these libraries are also released under an Open Source license. To facilitate the compilation process, XMLVM contains binary versions (i.e., jars) of all required libraries. All thirdparty libraries are contained in the lib directory. Building XMLVM from sources requires Java 1.6 as well as ant. In order to compile XMLVM from command line, simply run ant in the XMLVM root directory:

cd xmlvm ant ant demos

After a successful run of ant, there should be a dist/ directory. The ant script packages all dependent libraries and XMLVM's own class files into one jar file. The only file needed to run XMLVM is the jar file

dist/xmlvm.jar. This jar file can be copied to a convenient location with the ant install target, as presented below

Install and uninstall XMLVM

he install procedure is simple and it will ease the user interface with XMLVM libraries

Edit file properties/local.properties (or create it if it does not exist). Define there the destination prefix of XMLVM in the variable xmlvm.install If you do not define a specific location, the default installation path would be /usr/local Type ant install For example, to install it under your home directory, add the following entry to the properties/local.properties file:
xmlvm.install=*user.home/xmlvmTheantinstallcommandwill* and *XMLV MINSTALL/bin/xmlvmMakesurethatyouhavewritepermissionsonthedesti nat*
Make sure to add the *XMLV MINSTALL/binlocationtoyour*PATH

The uninstall procedure is similar. Make sure again that you have write permissions and then type ant uninstall

Invoking XMLVM

Java 1.6 is needed to run XMLVM. Invoking XMLVM can be done after installing in the following way: xmlvm ARGUMENTS...

Make sure that you have the installation location to your path, with a command similar to this: export
PATH=PATH:/installation/loc ation/of/xmlvm/bin

XMLVM supports various command line options that drive the cross-compilation process. The general pattern of using XMLVM to cross -compile an application is:

xmlvm –in=¡input¿ –out=¡output-directory¿ –target=¡target¿

The –in option defines the input files to be cross-compiled whereas the –out parameter designates the output directory of the crosscompiled input. The –target option finally defines for which platform XMLVM should generate code. These as well as other options are explained in detail in the following section.

6 Conclusion:

We have thus understood the concept of cross compilation using XMLVM.