Web Services Integration Toolkit for OpenVMS I64 and x86-64

Installation Guide and Release Notes

May 2024

VSI-I64VMS-WSIT-V0305-0F-1.PCSI$COMPRESSED
VSI-X86VMS-WSIT-V0305-0F-1.PCSI$COMPRESSED

VMS Software, Inc. (VSI)
Boston, Massachusetts, USA

1. Introduction 3
2. What's New (cumulative) 3
3. Requirements 4
    3.1. Hardware Requirements 4
    3.2. Software Installation, Development, and Runtime Requirements (OpenVMS) 4
4. Installation 5
5. After Installing the Web Services Integration Toolkit for OpenVMS 6
    5.1. Starting and Stopping the Web Services Integration Toolkit for OpenVMS 7
    5.2. Uninstalling the Web Services Integration Toolkit 7
    5.3. Web Services Integration Toolkit Directory Structure 7
6. Performance Recommendations 8
7. Known Problems and Issues 9

1. Introduction

VMS Software Inc. is pleased to provide you with this new version of the Web Services Integration Toolkit for OpenVMS on Integrity servers.

The Web Service Integration Toolkit for OpenVMS (WSIT) contains a collection of integration tools. These tools are easy to use, highly extensible, based on standards, and built on open-source technology. The toolkit can be used to call OpenVMS applications written in 3GL languages, such as C, BASIC, COBOL, FORTRAN, and ACMS from newer technologies and programming languages such as Java, C#, Python, and JavaScript[1].

2. What's New (cumulative)

V3.5-0F:

This is the first release of WSIT for VSI OpenVMS x86-64 and includes all the functionality provided by the I64 kit, with the exception that the x86-64 kit does not include the `obj2idl.exe` tool to generate interface definitions from object code or the WSIT Python extension (`wispy.py` and `_wsipy.exe`). It is anticipated that these components will be included in future releases of WSIT for VSI OpenVMS x86-64.

V3.5-0E:

- Previous releases of VSI WSIT for VSI OpenVMS included only upper-case symbol names in the symbol vector of the `WSI$COMMON.EXE` shareable image, which caused problems for some users. This issue has now been resolved, with the image now containing both upper- and mixed-case symbol names, consistent with older HPE versions of WSIT. Whilst the order of existing upper-case symbol entries in the `WSI$COMMON.EXE` shareable image symbol vector have not changed, it is recommended that users relink their WSIT server applications to ensure that any potential shareable image mismatch problems are avoided.

- When generating server interface code (C code), WSIT may produce code containing function names that are longer than 31 characters in length, resulting in compiler warnings, and unresolved symbols when linking. To avoid this problem, the code generation template for the server build procedure has been modified to generate build scripts that use the `/names=shortened` option for code compilation.

3.5-0D:

- The new options (XML tags) `<protocol>` and `<CLOPT>` have been introduced to the out-of-process section of server configuration file. The `<protocol>` tag is used to specify the inter-process communication protocol to be used for the application, and the `<CLOPT>` tag can be used to specify command line options to be supplied to the server by `WSI$MANAGER.EXE` at server start-up.

- A generic RabbitMQ consumer component (`wsi$server_amqp.exe`) that allows RabbitMQ and AMQP to be used in place of the traditional WSIT ICC protocol for out-of-process deployments server deployments. The corresponding ICC-based server has been renamed to `wsi$server_icc.exe`.

[1]Use of C#, Python, and JavaScript for client development is supported on Linux and Windows only and requires the use of AMQP (the Advanced Message Queuing Protocol) in conjunction with the RabbitMQ message broker. Java can be used in conjunction with the ICC protocol on OpenVMS and with AMQP on other platforms.

- New sets of Velocity templates have been implemented to facilitate the generation of interfaces in Java, C#, Python, and JavaScript (via Typescript) that can be used in conjunction with the new RabbitMQ AMQP support.
- New Java, Python, C#, and JavaScript libraries have been implemented to support code generated by the new Velocity templates and facilitate communication with OpenVMS-based server components using the associated RabbitMQ language APIs.
- Client kits are available for Linux that include client libraries and tools for building and running client applications that use the new protocol support.
- A simple command line tool for managing WSIT applications is included in the OpenVMS kit. The tool (`wsiman.exe`) may be used by system managers and other suitably privileged users to start, stop, and monitor WSIT applications.
- Support for JavaScript includes a Node.js-based RESTful microservices framework that can be used to implement REST-based web service interfaces to WSIT-wrapped 3GL and ACMS applications.

3. Requirements

The Web Services Integration Toolkit-generated application server component is a native OpenVMS image installed on the system running the wrapped application. The generated source code must be built on the OpenVMS system that hosts the application using command-line driven development tools.

3.1. Hardware Requirements

The Web Services Toolkit for OpenVMS I64 requires approximately 80,000 blocks of disk space for installation.

3.2. Software Installation, Development, and Runtime Requirements (OpenVMS)

Following are the minimum requirements for OpenVMS systems needed to install the Web Services Integration Toolkit and to build and run the generated components. Optional requirements represent requirements that are not needed for every type of application. However, several such requirements will apply to your particular platform/component selection.

- **Installation requirements**
  - OpenVMS 8.4-2L1 or higher (I64), OpenVMS 9.2-1 or higher (x86-64)
  - ODS-5 disk
- **Development requirements**
  - VSI C V7.4-001 or higher for OpenVMS (the C compiler is required to build all server components)
  - OpenJDK 8.0-372C for VSI OpenVMS, or higher
- **Runtime requirements**
  - OpenJDK 8.0-372C (or higher) for VSI OpenVMS (required only if developing Java applications that will run on OpenVMS)

- **Out-of-process account preparation and requirements**

  If you are specifying an account in which to run out-of-process servers, you may want the account to have a minimal amount of privileges. You can specify an account to run out-of-process servers that has only the NETMBX and TMPMBX privileges. To use an account with these privileges, perform the following steps:

  1. Create an identifier within the system UAF with the name WSI$SERVER. Perform this step one time only.
  2. Grant the WSI$SERVER identifier to each account used to run a WSIT out-of-process server.

  If you do not perform these steps, the privileges required by the account are as follows:

  - BYPASS
  - SYSNAM
  - SYSPRV
  - IMPERSONATE
  - DETACH
  - TMPMBX

4. Installation

Installation involves installing and then starting the Web Services Integration Toolkit on OpenVMS.

- **Before you begin**

  If you have previously installed a version of the Web Services Toolkit, make sure the Web Services Toolkit is shut down by entering this command from the SYSTEM account (note that the "ALL" option should be specified to ensure that all out-of-process WSIT-managed application server processes are shutdown in addition to WSIT management processes):

  ```
  $ @SYS$STARTUP:WSI$SHUTDOWN.COM ALL
  ```

  VMS Software Inc. recommends that you perform a system disk backup before proceeding.

- **Installation**

  The kit is provided as an OpenVMS PCSI kit (VSI-I64VMS-WSIT-V0305-0F-1.PCSI or VSI-X86VMS-WSIT-V0305-0F-1.PCSI, depending on platform) that can be installed by a suitably privileged user using the following command:

  ```
  $ PRODUCT INSTALL WSIT/DESTINATION=ods5disk:[directory]
  ```

  The Web Services Toolkit installation procedure accepts a destination directory specified on the PCSI install command line. This location is used as the root directory for the WSIT product. If the destination is not specified, the default is SYS$COMMON:[000000] and the directory created is SYS$COMMON:[WSIT].

  Follow the on-screen installation instructions to complete the installation. To proceed with a default installation, press Enter/Return in response to any installation questions.

- **Installing the WSIT runtime only**

  During the WSIT installation procedure, you are asked if you want to install only the runtime file. If you answer YES, the WSIT tools, documentation and sources will not be installed.

```
Do you want the defaults for all options? [YES] n

 Install the full WSIT kit? [YES] n

 Install the WSIT runtime only? [YES] y

Do you want to review the options? [NO] n

Execution phase starting ...
```

5. After Installing the Web Services Integration Toolkit for OpenVMS

To complete the installation, perform the following steps:

- Modify your system start-up command procedure (SYS$MANAGER:SYSTARTUP_VMS.COM) to include the Web Services Toolkit start-up command:

```
$ @SYS$STARTUP:WSI$STARTUP.COM
```

- Modify your system shutdown command procedure (SYS$MANAGER:SYSHUTDWN.COM) to include the Web Services Toolkit shutdown command:

```
$ @SYS$STARTUP:WSI$SHUTDOWN.COM ALL
```

- Finally, start the Web Services Integration Toolkit with this command:

```
$ @SYS$STARTUP:WSI$STARTUP.COM
```

This completes the Web Services Integration Toolkit for OpenVMS installation.

Note that if you opt not to use the SYSTEM account, the Web Services Toolkit start-up procedure must still be run every time your system starts. It can be run as part of your system start-up or in another account.

It is recommended that you use the SYSTEM account. However, the account in which you run the Web Services Toolkit must have, minimally, the following authorized account privileges:

- BYPASS
- SYSPRV
- TMPMBX
- SYSNAM
- CMKRNL
- DETACH
- LOG_IO

5.1. Starting and Stopping the Web Services Integration Toolkit for OpenVMS

- Use the following command to start the Web Services Integration Toolkit:

  ```
  $ @SYS$STARTUP:WSI$STARTUP.COM
  ```

- Stopping the Web Services Integration Toolkit requires the same privileges as starting it. Ensure that all client connections have been closed and from the SYSTEM account run the following command:

  ```
  $ @SYS$STARTUP:WSI$SHUTDOWN.COM ALL
  ```

5.2. Uninstalling the Web Services Integration Toolkit

To remove the Web Services Toolkit for OpenVMS from your system, enter the following command:

```
$ PRODUCT REMOVE WSIT
```

5.3. Web Services Integration Toolkit Directory Structure

After you install the Web Services Integration Toolkit, browse the WSIT directories to see the location of the tools and sample programs.

- **WSIT root directory**

  The root directory is represented by the logical WSI$ROOT. This logical is created when the WSI $STARTUP.COM procedure is executed.

  ```
  $ directory wsi$root:[000000]

  Directory WSI$ROOT:[000000]

  deploy.DIR;1        lib.DIR;1           LICENSE-APACHE2.txt;1
  LICENSE-GPL2.txt;1  LICENSE-MPL-RabbitMQ.txt;1              logs.DIR;1
  samples.DIR;1       tools.DIR;1         wsi-version.txt;1

  Total of 9 files.
  ```

- **Deploy subdirectory**

  The [.deploy] subdirectory contains the shareable image for the application's server wrapper. The generated JavaBean calls the server wrapper's shareable image at runtime.

- **Docs subdirectory**

  The [.docs] directory contains Web Services Integration Toolkit documentation.

- **LIB subdirectory**

  The [.lib] subdirectory is where the Web Services Integration Toolkit stores the files it uses internally; for example, Velocity template files.

- **Samples subdirectory**

The [.samples] subdirectory contains example programs. This kit includes sample applications written in ACMS, C, COBOL, and FORTRAN.

The .C, .CBL and .FOR programs are the wrappers for the original application. The .JAVA files illustrate calling the WSIT-generated JavaBean for each wrapper. See the Developer's Guide Appendix for the source files of the C sample.

- **Tools subdirectory**

  The tools in this directory are used to develop code that wraps 3GL applications.

- **Source subdirectory**

  The [.source] subdirectory contains objects and classes used by the generated code, including Java exceptions and holder classes. The files in this subdirectory are provided for your reference.

6. Performance Recommendations

- **OpenVMS operating system requirements**

  Increase the PRCLM quota to be high enough to run the maximum number of expected concurrent servers.

- **Requirements for Java on OpenVMS**

  The Java runtime environment was designed to perform optimally on UNIX systems, where each process is given large quotas by default. On OpenVMS, the default behaviour gives each process lower quotas so that many processes can co-exist on a system. To get the best Java performance on OpenVMS, you are recommended to set process quotas to match a typical UNIX system. These are also the recommended minimum quota settings (except where noted).

  These are the recommended UAF and system parameter settings for WSIT with Java on OpenVMS systems:

FILLM	4096
CHANNELCNT	4096
WSDEF	2048
WSQUOTA	4096
WSEXTENT and WSMAX	16384
PGFILQUO	2,097,152 (see comments below)
BYTLM	6,000,000
BIOLM	150
DIOLM	150
TQELM	100
CTLPAGES	16384

  A good number for PGFLQUO is (2 x heap-size), for example, 128 MB (2*128*1024*1024)/512 = 524288. The recommended minimum PGFLQUO is 96 MB when using the WSIT RTE. When you increase the PGFLQUO parameter, you should always increase the system's page file size to accommodate the new PGFLQUO parameter, if needed.

  Note that if you receive a WSIConnectException without a specific message attached to it, the most likely cause is a value for BYTLM that is too low. VMS Software Inc. recommends that you

set this value to at least 6,000,000. It is additionally recommended that CTLPAGES be significantly increased (as shown above), particularly for sites that have greater than 150 concurrent ICC connections to out-of-process application servers. ICC connections consume considerable BYTLM and CTLPAGES resource, and if either of these values are too low then connection exceptions can occur.

7. Known Problems and Issues

- **Known problem in AXIS2 java2WSDL tool affecting case**

  The AXIS2 java2WSDL tool modifies the name of all properties to begin with a lowercase letter. For example, if the IDL XML has a structure with a field named EMPLOYEE_NAME, the AXIS2 java2WSDL tool generates WSDL with the element eMPLOYEE_NAME. Until this matter is resolved, you must choose between the following:

  - Using lowercase first letters in the WSIT IDL where appropriate; or
  - Including your own WSDL file in your AXIS2 archive file (.aar). In this scenario, you can run the tool by hand and then modify the WSDL and add it to the archive.

  The elements that need to be addressed include structure field names and parameter element names when the usage is specified as IN/OUT or OUT. The parameters are included because the SOAP response class is a composite of the returned parameters.

- **Copying WSIRTL.JAR to server classpath (if required)**

  If you plan to call a Web Services Integration Toolkit application from a web server or application server, the file wsi$root:[lib]wsirtl.jar must be in the servers classpath. The details on how to configure the server to include this jar file are specific to each server.

  For example, if you are using CSWS_JAVA (Apache Tomcat for OpenVMS), you should copy the file wsi$root:[lib]wsirtl.jar to the "common/lib" directory. To ensure that the server becomes aware of the file, a restart if often required. However, this is also server-specific.