Australian Government

Australian Digital Health Agency

HIPS
Build Guide

7 June 2019   v7.1.1

Approved for external use

Document information

Key information

Owner   National Health Chief Information Officer, Infrastructure Operations

Contact for enquiries   Australian Digital Health Agency Help Centre

Phone   1300 901 001

Email   help@digitalhealth.gov.au

Table of contents

7 June 2019

1 Introduction

1.1 Purpose

The purpose of this document is to provide straightforward instructions for building the HIPS suite of products from the provided source code.

If using the HIPS product suite, the steps in this build guide must be completed prior to steps from any of the HIPS installation guides.

1.2 Intended audience

The intended audience for this document is software developers and implementers of the HIPS product suite.

1.3 Scope

This document describes the steps and tools required to build the binaries and associated files required for each of the HIPS products. All information that is essential to the build process of version 7.1.1 of the HIPS product suite is provided. This includes how to build and verify the various products, library dependencies, and tools required.

This document does not describe any functional requirements, or the installation of the products into a test environment; these are covered in separate documentation.

1.4 Assumptions

The following assumptions have been made in the development of this document:

- the reader has an understanding of the HIPS products and their use

- the reader is familiar with Microsoft Visual Studio and its associated solution files

- the term "solution" refers to the Microsoft Visual Studio solution file

- the developer/implementer has the correct software and versions to build the HIPS products.

2   Products overview

2.1   Source Code Software Package

The HIPS source code and related artefacts are contained in the *HIPS_SourceCodeSoftwarePackage_v7.1.1* subfolder of the *HIPS Release 7.1.1* zip file.

The folder contents are as follows:

- **HIPS Core Source.zip**: Contains all source code required to build the components of the HIPS Core module.

  o **Build**: Contains Cake Build scripts to build source code.

  o **extern-lib**: Contains external libraries referenced by the module.

  o **HIPS**: Contains the HIPS_AppServer solution source code, which includes the HIPS Core database, web services and queue consumer and alert monitor background processes.

  o **HipsMonitoring**: Contains the HIPS-Monitoring solution source code, providing rudimentary monitoring of HIPS Core functions.

  o **hipstest**: Contains the Demo Harness solution source code, used for evaluation of the HIPS Core functions.

  o **Mirth**: Contains the source code and associated library files required to build the acknowledgement web service library for the optional MirthConnect ESB.

- **HIPS Web UI Source.zip**: Contains all source code required to build the components of the HIPS Web UI module.

  o **Build**: Contains Cake Build scripts to build source code.

  o **extern-lib**: Contains external libraries referenced by the module.

  o **HIPS_Web**: Contains the HIPS Web UI solution source code, providing a web user interface for HIPS functions.

  o **TokenGenerator**: Contains the Token Generator solution source code, used to generate authentication tokens for the HIPS Web UI.

3   Build process

The build process is designed to be self-contained. The build tasks can be run on any machine where Visual Studio and the Java Development Kit (JDK) are available. There should be no need to resolve library dependencies as all of the required libraries are included in the release package or automatically downloaded by the NuGet Package Manager.

The remainder of the section will provide details on the tools, targets, dependencies, and information related to the build process.

3.1   Build target products

The build process should result in the following products being built, and ready for release into a test environment:

- HIPS Core
  - o   HIPS-AppServer: database, web, queue consumer, alert monitor
  - o   HIPS-DemoHarness
  - o   HIPS-Monitoring
  - o   HIPS-Mirth
- HIPS Web UI
  - o   HIPS-Web
  - o   HIPS-Web-TokenGenerator

3.2   Build tools and environment

The following are the prerequisites that a developer will need to compile the source code into executable binaries and deployment packages for HIPS.

The development environment will require the following software packages to be able to build the HIPS source code.

3.2.1   Core web services

Software	Version	Source
Visual Studio Professional or Enterprise	2017	https://www.visualstudio.com/downloads/
NuGet Package Manager	4.5.0+	Included in Visual Studio 2017
Java SE Development Kit	8+	http://www.oracle.com/technetwork/java/javase/overview/index.html

Software	Version	Source
Windows PowerShell	5.0+	https://docs.microsoft.com/en-us/powershell/scripting/setup/installing-windows-powershell?view=powershell-5.1
SvcUtil.exe[1]		Part of the Windows SDK
		https://developer.microsoft.com/en-us/windows/downloads/windows-10-sdk

3.2.2   User interface web application

Software	Version	Source
Visual Studio Professional or Enterprise	2017	https://www.visualstudio.com/downloads/
NuGet Package Manager	4.5.0+	Included in Visual Studio 2017
Windows PowerShell	5.0+	https://docs.microsoft.com/en-us/powershell/scripting/setup/installing-windows-powershell?view=powershell-5.1

3.3   Build target

3.3.1   Visual Studio

Visual Studio includes a Configuration Manager tool that specifies how the projects within the solution are to be built and deployed. The following configurations have been set for each of the solutions.

Solution	Configuration	Description
HIPS_AppServer	Release	Builds a version of the application that can be deployed to a test or production environment. Each project within the solution has the *Target Platform* set to "Any CPU".
	Debug	Supports the debugging of an application. Used in a development environment.
	Puma_Debug	Not used for this HIPS release.
	Puma_Release	Not used for this HIPS release.
HIPS.Web	Release	Builds a version of the application that can be deployed to a test or production environment. Each project within the solution has the *Target Platform* set to "Any CPU".
	Debug	Supports the debugging of an application. Used in a development environment.

---

[1] Required to generate WSDL during the HIPS-AppServer build.

Solution	Configuration	Description
DemoHarness	Release	Builds a version of the application that can be deployed to a test environment. *Target Platform* set to "Any CPU".
	Debug	Supports the debugging of an application. Used in a development environment.
HipsMonitoring	Release	Builds a version of the application that can be deployed to a test or production environment. *Target Platform* set to "x86".
	Debug	Supports the debugging of an application. Used in a development environment.

3.3.2   Mirth Connect

The target for the Mirth Connect component is a java class (.jar file) required for the Mirth Connect component to accept messages from the HIPS Web Services.

3.4   Build steps

The following section will outline the steps for building the HIPS Source Code Software Package components.

IMPORTANT NOTE: The HIPS Source Code Software Package previously required developers to individually open and build each solution within the Package. Starting with HIPS 7.0.0 the HIPS build process has been largely automated using Cake Build. This enables a developer to execute a single build script for each module that will build all module components from their source code. Developers may still individually open and build each solution within the Package, however instructions are no longer provided for doing so.

3.4.1   Prerequisites

3.4.1.1   Debugging uninstall requirement

Visual Studio adds in an XML element <VsDebuggerCausalityData> to the SOAP messages it generates when debugging. This is used by Visual Studio to help it debug the WCF calls using system diagnostics.

This XML element must not be added to the SOAP requests sent to the My Health Record system. This function must be disabled by running the WCF Diagnostics Registration Tool - vsdiag_regwcf.exe to uninstall WCF diagnostics. This is located (on 64 bit Windows) in C:\Program Files (x86)\Microsoft Visual Studio\2017***Professional**\Common7\IDE where ***Professional*** depends on the version you are using. This path is generally included in the path of the Developer Command Prompt for VS 2017.

To uninstall the component, the following command needs to be run from a command prompt as an administrator:

```
vsdiag_regwcf.exe –u
```

3.4.1.2 Source code organisation

The contents of each ZIP file contained in the HIPS Source Code Software Package should be extracted to a common root folder, for example:

- C:\Source\HIPS\<version>

  - HIPS Core: Contents of *HIPS Core Source.zip* file

  - HIPS Web UI: Contents of *HIPS Web UI.zip* file

3.4.1.3 HIPS NuGet package source

The HIPS UI Web module depends upon the HIPS Core module at build time and run time via the `HIPS.Client.Proxy` assembly and its dependencies. Previously this build-time dependency was satisfied by a post-build event in the HIPS Core module's `HIPS_AppServer` solution that would attempt to copy required files from the `HIPS.Client.Proxy` project build output directly into the HIPS Web UI module's `HIPS.Web` solution.

Starting with HIPS 7.0.0 this dependency moves to NuGet instead. The HIPS Core module's `HIPS.Client.Proxy` project build now also generates a `HIPS.Client.Proxy.nupkg` NuGet package, which is pushed to a shared filesystem location. The `HIPS.Web` solution is configured with a local filesystem-based NuGet source based on the same shared filesystem location. The HIPS Web UI module's build process uses this NuGet source at build time to reference the latest `HIPS.Client.Proxy.nupkg` pushed to the source by the HIPS Core module's build process.

A developer should not need to be aware of or change the default behaviour of these build processes. However, in larger team-based development environments it may be desirable to reconfigure this default NuGet source to employ a file share or alternatively a NuGet server.

The default HIPS NuGet source is named HIPS-NuGet and is located in the root of the filesystem location to which the HIPS Source Code Software Package has been extracted, for example: `C:\Source\HIPS\HIPS-NuGet`.

3.4.1.4 WSDL Generation

If it is desired to generate WSDL during the HIPS-AppServer build, you must download the Windows SDK and place `SvcUtil.exe` in the `scripts` subfolder of the `Build` folder in the `HIPS Core` source code, for example `C:\Source\HIPS\<version>\HIPS Core\Build\scripts`.

3.4.2 HIPS Core

1   Open **Windows PowerShell** as Administrator.

2   Use the Set-ExecutionPolicy cmdlet to enable running of PowerShell scripts, if not already enabled:

```
Set-ExecutionPolicy RemoteSigned
```

3   Use the Set-Location cmdlet to move to the folder containing the HIPS Core build scripts, for example:

```
Set-Location "C:\Source\HIPS\<version>\HIPS Core\Build"
```

4   Execute the following command to build all HIPS Core components:

```
.\build.ps1 -Script "build-binaries.cake"
```

This command will build all HIPS Core components using default arguments and output the build results to a subfolder located at `C:\Temp\HIPS`.

The following arguments can optionally be provided to the build script:

`outputPath`: Absolute path to the filesystem location at which build outputs should be generated.

For example, execute the following command to output to `C:\Output\HIPS\HIPS Core`: `.\build.ps1 -Script "build-binaries.cake" -outputPath="C:\Output\HIPS\HIPS Core"`

> For further details of available arguments, refer to the source code of the `Build\build-binaries.cake` **and** `Build\scripts\build-common.cake` scripts.

NOTE: If it is desired to build individual components, simply execute the `build-*.cake` script corresponding to the individual component.

The output of the `build-binaries.cake` script is a folder structure similar to the following:

- `C:\Temp\HIPS\eb9ecd55`
    - o   `HIPS-AppServer`
    - o   `HIPS-DemoHarness`
    - o   `HIPS-Mirth`
    - o   `HIPS-Monitoring`

This is exactly the same structure and contents as are provided in the `HIPS Core Binaries.zip` file contained in the `HIPS_BinarySoftwarePackage_v7.1.1`. To test the build process was successful, deploy the components and artefacts built in the preceding steps to a test environment by following the *HIPS – Initial and Clean Installation Guide (Core) document.*

3.4.3   HIPS Web UI

1   Open **Windows PowerShell** as Administrator.

2   Use the Set-ExecutionPolicy cmdlet to enable running of PowerShell scripts, if not already enabled:

```
Set-ExecutionPolicy RemoteSigned
```

3   Use the Set-Location cmdlet to move to the folder containing the HIPS Web UI build scripts, for example:

```
Set-Location "C:\Source\HIPS\<version>\HIPS Web UI\Build"
```

4   Execute the following command to build all HIPS Web UI components:

```
.\build.ps1 -Script "build-binaries.cake"
```

This command will build all HIPS Web UI components using default arguments and output the build results to a subfolder located at `C:\Temp\HIPS`.

The following arguments can optionally be provided to the build script:

`outputPath`: Absolute path to the filesystem location at which build outputs should be generated.

For example, execute the following command to output to `C:\Output\HIPS\HIPS Web UI`:

```
.\build.ps1 -Script "build-binaries.cake" -outputPath="C:\Output\HIPS\HIPS Web UI"
```

For further details of available arguments, refer to the source code of the `Build\build-binaries.cake` and `Build\scripts\build-common.cake scripts`.

NOTE: If it is desired to build individual components, simply execute the *build-*.cake* script corresponding to the individual component.

The output of the `build-binaries.cake` script is a folder structure similar to the following:

- `C:\Temp\HIPS\eb9ecd55`
    - `HIPS-Web`
    - `HIPS-Web-TokenGenerator`

This is exactly the same structure and contents as are provided in the `HIPS Web UI Binaries.zip` file contained in the `HIPS_BinarySoftwarePackage_v7.1.1`. To test the build process was successful, deploy the components and artefacts built in the preceding steps to a test environment by following the *HIPS – Initial and Clean Installation Guide (UI)* document.

Acronyms

Acronym	Description
CPU	Central processing unit of the computer.
DMZ	Demilitarised zone, exposes external facing services.
JDK	Java Development Kit
MLLP	Minimal Lower Layer Protocol, transport protocol used in the Mirth Connect component to transport HL7 messages.
SOAP	Simple Object Access Protocol, used to exchange structured information through web services.
UI	user interface
WCF	Windows Communication Framework, used in the building of the web services.
WSDL	Web Service Description Language, used to describe web services.
XML	Extensible Markup Language.