class **Ensemble.Tests.ProductionTest** extends UnitTest.ProductionTestHelper

Inventory

Summary

Parameters

- parameter **PRODUCTION** = "Ensemble.Production";

  Determines what production this test is running against.

Properties

- property **Sender** as UnitTest.HL7Sender;

Methods

- method **OnAfterAllTests()** as %Status

  1. Restore router validation to its original state.
  2. Restore file service's path to its orginal state.
  3. Stop the production.
  4. Remove test directory.

```
        Do ..ChangeSetting(,"MsgRouter","Validation","dm-z")
        Do ..ChangeSetting(,"HL7FileService","FilePath", "/tmp/")

        Do ..CleanUpDirectory(..MainDir, 1)
        Do ..ForceStopProduction()

        Quit $$$OK
```

- method **OnBeforeAllTests()** as %Status

  1. Instantaties cache class that will help us send HL7 messages over TCP and to a file.
  2. Sets main testing directory to a random directory name under the cache temp directory.
  3. Creates directory structure for testing.
  4. Update HL7FileService configuration item to look for HL7 files in our test directory.
  5. Loosen the validation restrictions on the message router to ease the overhead of setting up a message.
  6. Start the production.

```
        Set ..Sender = ##class(UnitTest.HL7Sender).%New()
        Set i%MainDir = ##class(%File).SubDirectoryName(##class(%SYS.System).TempDirectory(),$ZTIMESTA
        Set i%HL7InputDir = ..SubMainDir("HL7/In")
        Set i%HL7OutputDir = ..SubMainDir("HL7/Out")
        Set i%HL7WorkDir = ..SubMainDir("HL7/Work")
        Set i%HL7ArchiveDir = ..SubMainDir("HL7/Archive")
        Do ..CreateMainDirTree()
        Do ..ChangeSetting(,"HL7FileService","FilePath", ..HL7InputDir)
        Do ..ChangeSetting(,"MsgRouter","Validation", "d")
        Do ..StartProduction()
        Quit $$$OK
```

- method **OnBeforeOneTest()** as %Status

  Destroys all HL7 messages and message trace information.

```
	Do ##class(EnsLib.HL7.Message).%KillExtent()
	Do ##class(Ens.MessageHeader).%KillExtent()
	Do ##class(Ens.MessageBody).%KillExtent()

	Quit $$$OK
```

- method **TestFileMessage()**
  - Creates a basic HL7 message that should make it through the router and to the "HL7FileOperation".
  - Generates a temp filename with a "hl7" suffix that will be written out to a directory that the "HL7FileService" is monitoring.
  - Writes the message out to the directory using the UnitTest.HL7Sender helper.
  - Lastly, we assert the path the message took through the production; HL7FileService -> MsgRouter -> HL7FileOperation.

```
	Set message = ..CreateMessage("2.1:ORM_O01")
	Do message.SetValueAt("2.1", "MSH:VERSIONID")
	Do message.SetValueAt("ORM^O01", "MSH:MESSAGETYPE")
	Do message.SetValueAt("O01", "EVN:EVENTTYPECODE")

	Set outputFile = ..HL7InputDir_"/"_##class(%IO.FileStream).NewTempFilename("hl7")
	Do $$$AssertStatusOK(..Sender.SendFile(outputFile,message))

	Do ..WaitForCallInterval("HL7FileService")

	Do ..AssertMessageTrace("HL7FileService","MsgRouter","HL7FileOperation")
```

- method **TestTCPMessageFromGrandRapids()**
  - Creates a HL7 message that has "GRAND RAPIDS" as the Admit Source which means it should be sent to the "HL7FTPOperation" configuration item.
  - Using the UnitTest.HL7Sender we send the message over TCP to the "HL7TCPService".
  - We wait for the call interval for that configuration item.
  - Lastly, we assert the path the message took through the production; HL7FileService -> MsgRouter -> HL7FTPOperation

```
	Set message = ..CreateMessage("2.1:ADT_A12")
	Do message.SetValueAt("2.1", "MSH:VERSIONID")
	Do message.SetValueAt("ADT^A12", "MSH:MESSAGETYPE")
	Do message.SetValueAt("A12", "EVN:EVENTTYPECODE")
	Do message.SetValueAt("323", "PID:SETIDPATIENTID")
	Do message.SetValueAt("Justin", "PID:PATIENTNAME")
	Do message.SetValueAt("Adult", "PV1:PATIENTCLASS")
	Do message.SetValueAt("MMPC", "PV1:ASSIGNEDPATIENTLOCATION")
	Do message.SetValueAt("GRAND RAPIDS", "PV1:ADMITSOURCE")

	Do $$$AssertStatusOK(..Sender.SendTCP("127.0.0.1","435",message))

	Do ..WaitForCallInterval("HL7TCPService")

	Do ..AssertMessageTrace("HL7TCPService","MsgRouter","HL7FTPOperation")
```

- method **TestTCPMessageFromHolland()**
  - Creates a HL7 message that has "HOLLAND" as the Admit Source which means it should be sent to the "HL7TCPOperation" configuration item.
  - Using the UnitTest.HL7Sender we send the message over TCP to the "HL7TCPService".
  - We wait for the call interval for that configuration item.
  - Lastly, we assert the path the message took through the production; HL7FileService -> MsgRouter -> HL7TCPOperation

```
	Set message = ..CreateMessage("2.1:ADT_A12")
	Do message.SetValueAt("2.1", "MSH:VERSIONID")
	Do message.SetValueAt("ADT^A12", "MSH:MESSAGETYPE")
	Do message.SetValueAt("A12", "EVN:EVENTTYPECODE")
	Do message.SetValueAt("323", "PID:SETIDPATIENTID")
	Do message.SetValueAt("Justin", "PID:PATIENTNAME")
	Do message.SetValueAt("Adult", "PV1:PATIENTCLASS")
	Do message.SetValueAt("MMPC", "PV1:ASSIGNEDPATIENTLOCATION")
	Do message.SetValueAt("HOLLAND", "PV1:ADMITSOURCE")

	Do $$$AssertStatusOK(..Sender.SendTCP("127.0.0.1","435",message))

	Do ..WaitForCallInterval("HL7TCPService")

	Do ..AssertMessageTrace("HL7TCPService","MsgRouter","HL7TCPOperation")
```
