**GIFT Understandings**

GIFT is a service-oriented framework of tools, methods & standards.

External applications are programs that GIFT has integrated with but did not create, nor manage the application. Examples include but are not limited to **Virtual Battlespace (VBS)**.

But so far, it seems like the code I downloaded has GIFT already integrated with VBS.

If GIFT is meant to be used separately, then why provide VBS integration along with it in the code. OK Maybe VBS isn’t actively integrated with GIFT. They’ve just written measures to help **IF** we were to integrate with VBS.

**LMS(Learning Mgmt. System) Module**

*GIFT/config/lms/LMSConnections.xml*

This configures the LMS connections for the LMS module when it starts. The XML file can contain multiple LMS connections, thereby allowing the LMS module to connect with multiple LMS backends simultaneously. Any requests from GIFT (for say GET /learner’s records) are sent to all the configured LMS backends. The returned learner’s records are in the GIFT Tutor User Interface (TUI) on the LMS course history webpage.

**Gateway Module**

So, there’s a Gateway Module that is used for communication between GIFT and any external training application (like VBS).

*GIFT/config/gateway/configurations/default.interopConfig.xml*

The interop config file is used to instantiate interop interface plugins (i.e., classes in Gateway package) which are used to facilitate communication between GIFT and external training applications. The interop configuration file is handled by the InteropConfigFileHandler.java and validated against interopConfig.xsd (GIFT\config\gateway\).

Out of the many interop interfaces created for GIFT, the following can be used for Dreamscape(etc. connection):

* Augmented Reality Sandtable (ARES)
* VR Engage

**Sensor Configuration**

Sensor Configuration files define what and how sensors are connected and configured to a GIFT sensor module.

There are 3 main sections of a sensor configuration file: sensors, filters, and writers:

* Sensors: Specifies the types of sensors, the sensor module will be collecting sensor data from.
* Filters: Specifies the available filters in the sensor module that can be used to filter raw sensor data
* Writers: Specifies the available writers in the sensor module that can be used to write sensor data or filtered data to disk.

**Learner Configuration**

*GIFT/config/learner/configurations/Default.learnerconfig.xml*

This learner config file defines the translators, classifiers and predictors for sensor data of various types. As the sensor data, filtered and/or unfiltered, is received by the Learner module it first passes through a translator followed by a classifier and then finally a predictor. The result is a learner state for the learner state attribute(s) (e.g., arousal, engagement) associated with the various sensor data attributes (e.g., humidity, self-assessment).

**Database Connections Config - Refer Config settings documentation**

* UMS Database Connection Configuration
* LMS Database Connection Configuration

**Custom Training Application Setup**

The requirements might ask us to control/integrate with a training app(e.g. a Java application) which isn’t currently configured by GIFT. We may be required to launch this app, monitor the state of this app & close the app. To do this, we create a Gateway Module Interop Connection (**GWIC** - my custom term). For e.g. *DIS (gateway.interop.dis.DISInterface) and VBS plugin(gateway.interop.vbsplugin.VBSPluginInterface)*

Similar to the above files, we need to create a **GWIC**. This GWIC class should contain the logic to register for SIMAN message types, just like in VBSPluginInterface. It is important to note that the assessment logic might need to know, at a minimum, when the training application started and stopped.

After you create the **GWIC** class (which handles communication between GIFT and the training application) you will need to author the interop configuration in the interopConfig.xml (GIFT/config/gateway/interopConfig.xml). This may involve augmenting the interopConfig.xsd depending on the types of configurations needed. (Look at the DIS interop configuration XML elements as an example.)

Now that the interopConfig.xml file is authored appropriately, the Gateway module will initialize the interop connections using the files content. The final step is to author the domain’s DKF (specifically the resources/transitions section) to specify the interop connections to use when the lesson is being started via the Training Application transition

***To create your own interop plugin, you have to do that in the gift/gateway/interop folder. You need to create a new folder in this. In this folder you have to create a new interface.***

Now, not all training apps require proper interoperability development. There is no standard communication protocol between the Gateway module and an external training application. This is due to the fact that some training applications are already interoperable and therefore already have a protocol in place which should be leveraged by GIFT to reduce additional development.

**Concept Assessment Logic**

Concepts are assessed by the conditions in the Domain Module. Assessment engines available in GIFT:

1. Default(available in GIFT, you can extend this by adding more )
2. SIMILE(external assessment engine that is configured using the output of the SIMILE workbench application)

3)You can create a new one if you want to.

A diagram of a computer program

Description automatically generated

A diagram of a service-oriented block diagram

Description automatically generated

So based on all this, I now have a decent grasp of the GIFT functionalities and am ready to get working on the code, in the following steps:

1. Get the initial setup done on a Windows PC.
2. Modify the code to integrate with our external training application(which I presume is what Dreamscape is going to be).