WaterSMART / CADSWES Study Manager / Hydrologic Simulator Plugin

Phil Weinstein / March 2012 / **Development Plan**

Document Status:

- 3-16-2012: Ready for Initial Review [PDF]

Contents:

(1) Project Objective:

Implement a "Hydrologic Simulator Plugin" for the CADSWES Study Manager to integrate an "R"-implemented Hydrologic Simulator being developed by Andrew Verdin to analyze stream flows at Lees Ferry.

The plugin to be developed is a Windows DLL to be used with the Study Manager, similar to the recently developed "Rdf Annualizer" and the "Rdf to Excel" plugins -- SEE IMAGES.

(2) Overview, Assumptions and Decisions -- *needs review:*

1. The Hydrologic Simulator will be implemented in "R" and will provide a specific set of high-level public functions (probably six) that will be known within the Hydrologic Simulator Plugin's implementation.

2. In the context of the Study Manager, the Hydrologic Simulator will be represented as an "Event" node which only generates output files. All inputs to the simulator functions will be characterized as parameters to those functions.

3. The Hydrologic Simulator Plugin will interact with the Hydrologic Simulator by dynamically generating an "R" script file and invoking "R" with that script. The plugin may be able to receive status from the simulator in three possible ways:
   - process exit status (success or failure)
   - "standard output" text
   - "standard error" text (*if this can be generated from "R"*).

4. The series data files generated from the Hydrologic Simulator will need to be documented and supported as a file *type* known to the Study Manager. These files will be of a format interpretable as (old-style) DMI input files. This is technically not a function of the Hydrologic Simulator *Plugin*, except for a minor "provided" (output) file type attribute associated with the plugin.

(3) Design and Preliminary Tasks:

By Andrew:

1. Characterization of the several (probably, six) functions implemented by the Hydrologic Simulator -- AND their parameters, including text labels and descriptive text (appropriate for a "help" presentation) and basic data types (boolean flag, character flag, integer, decimal fraction or other "float", character string, etc). [This is partially done].

2. Development of sample "R" script files with hard-coded test parameters for each of the supported functions.

By Phil:

1. C++ POD ("plain old data") classes to represent those parameter sets.
2. XML schema (informal) for those parameter set classes.

(4) Development Tasks:

A Study Manager plugin is developed by implementing a subclass of the Study Manager's EventPlugin base class with virtual method implementations to provide the following capabilities. These are discussed further in subsequent sections.

1. Descriptive text information ("type" and "description")
2. Characterization of input ("accepted") files and output ("provided") files.
3. Plugin GUI creation
4. XML-based serialization of configuration parameters maintained by the plugin.
5. Plugin configuration parameter validation
6. Plugin execution (generation of "provided" files)

(4.1) Descriptive text information

A text string for (a) an "event type" and (b) an "event description" are provided. The table below shows these strings for two previously implemented plugins:

Plugin	Event Type	Event Description
Rdf Annualizer:	"RdfAnnualizer"	"Generate an annualized RDF file from an input RDF file"
Rdf to Excel:	"RdfToExcel"	"Generate an Excel workbook from an RDF file"

(4.2) Characterization of input ("accepted") files and output ("provided") files.

As currently envisioned, inputs to the Hydrologic Simulator Plugin will be configured solely through parameter settings edited in the plugin's GUI. That is, no input ("accepted") file will be configured for this plugin within the Study Manager's workspace.

The output ("provided") file generated from the Hydrologic Simulator Plugin will be characterized by a new FileBase subclass. This includes virtual method implementations for:

- Providing a text description of the file type.
- Creating a cloned copy of an instance.
- Indicating whether or not the file can be opened by the Study Manager. (This is supported for all currently defined file types, e.g. the various RDF file types, except for "ExcelFile" -- i.e. not supported).

The FileBase class itself (provided by the Study Manager framework) implements a persistent file reference. (No

additional development is need for that functionality).

The output ("provided") files generated from the Hydrologic Simulator Plugin will contain numeric series data of a form **yet to be fully defined,** but will be readable as (old-style) DMI Input files. This will be a function of the "R"-implemented Hydrologic Simulator itself, and not of the Study Manager *plugin.*

(4.3) Plugin GUI creation

The plugin is provided with an empty parent widget in which the plugin's widget's are placed. The purpose of these widgets is to present and edit plugin parameters, plus a "Help" button to present HTML-formatted help information.

All instance data and handles to the plugin's widgets are encapsulated in a "delegate" (DelegateBase) subclass instance created within the plugin's EventPlugin implementation. The delegate emits a "configChanged" Qt signal each time an edit operation is performed by the user. Preexisting parameter values are delivered to the plugin -- and are asynchronously saved from the plugin -- via an XML DOM element -- see the next section.

Note: Although it is not technically necessary for the plugin's delegate to implement an internal representation of parameter data (as a POD C++ class -- "plain old data") apart from the XML DOM element and the state of the GUI, I recommend that we do. Having this C++ representation of parameter data greatly simplifies the interfaces between various components of the plugin's implementation.

Here are images of the GUIs for two previously implemented Study Manager plugins.

(4.4) XML-based serialization of configuration parameters maintained by the plugin.

The values of all Hydrologic Simulator Plugin parameters presented in the plugin's GUI, and edited by the user, will be serialized in XML. This is accomplished by inserting child elements into, and reading child elements from an XML DOM element provided as a parameter to virtual methods implemented by the plugin.

(4.5) Plugin configuration parameter validation

The level of parameter validation implemented by the plugin is flexible. At a minimum the following conditions should be confirmed:

- the presence of values for all required (non-optional) parameters and environment variables.
- the existence of input files (represented as parameters).

(4.6) Plugin execution (generation of "provided" files)

Upon request from the Study Manager, the plugin will create an "R" script file for the selected function, and will invoke "R" to execute the script, and report general results of that execution (success or failure, and any available failure information). The generated "R" script will internally represent the values of all relevant parameters supported by the plugin GUI for the selected Hydrologic Simulator function.

The current implementation of the Study Manager directly supports only synchronous plugin execution. That is, the result of the execution needs to be returned as the value of the plugin's "execute" method. Since we don't expect the Hydrologic Simulator execution to take very long (e.g. probably about 6 seconds) there shouldn't be a problem with this. (Deploying a separate thread to execute the "R" process will not be necessary, and wouldn't even have very much benefit, given the current architecture -- i.e. the synchronous nature of the plugin's *execute* method).

(4.7) Help file / function

It is the responsibility of the plugin to provide and implement a control (a "Help" button) to show an HTML file containing helpful information about the various functions provided by the Hydrologic Simulator, including function parameter descriptions.

For reference, here are the help files for two previously implemented Study Manager plugins (current as of 3-2012):

- RDF Annualizer
- RDF to Excel

[Suggestion: perhaps these help files should have revision date information at the bottom].

--- (end) ---