Integration of Models with IDEAS

Model requirements

Typical models will follow the standard formula of inputs plus operations equal outputs. The IDEAS Model as a Service (MaaS) design allows for a wide variety of operations with a tightly controlled set of inputs and outputs. For this challenge, if your solution requires a model to run and generate results, it can be implemented server-side with IDEAS MaaS.

Inputs

Models are initiated with a PHP script which carries a JSON payload which comes from the front end web-app (or mobile app). The JSON payload contains all the user selected parameters including the bounding box for the model run. In addition to this JSON, most models will need other data which we will refer to as an external data source

External Data Sources

Each model will likely rely upon a specific external data source (e.g., NASA Landsat satellite data, or NWS weather forecasts). Some models may make use of generic input data of a specific type and format (e.g., a Digital Elevation Model (DEM) or any source of overhead Orthorectified imagery).

Data inputs must be either self-contained or accessed using a standardized web-service or a documented web API or accessible over the internet in another fashion.

Some examples of existing data source:

1. Satellite based radiometric data products from NASA MODIS. These products are available in a flat file structure on a web server. A shell scripts executes periodically via a cron job to download updated data. Data are stored in a flat folder structure on the server. Space is restricted, so old data are purged once new data are downloaded.
2. Web-based weather forecast data via NWS API. National Weather Service forecasts are obtained at model run time for a certain model. The model script contains calls to the service and handles the parsing of the messages. Data is stored with the model output, and purged if the model run is erased in the system. The model software does not store the data beyond a short period.
3. A locally cached DEM. The USGS NED is stored locally to improve performance dramatically, vs. pulling in gigabytes of data for each run. The USGS NED DEM is updated regularly on the USGS side, but only quarterly (for example) on the cached IDEAS copy. Depending on the model, the local storage of data can be very important.

Operations

Models must be executed via command line; beyond that, the software that runs is highly flexible. The current design allows for any size server or even a cluster of servers to be utilized. Current work is ongoing to allow dynamic scaling of model servers based on demand.

Compatible software

Current model environments can be Windows or Linux. Open source languages and libraries are preferred (Python, numpy, R, Java, C++, GDAL, etc.).

Reliance on proprietary software

Utilizing proprietary software, while less simple, is still acceptable. Currently available commercial licenses include Esri ArcGIS Server/Desktop with Image Analyst and other Extensions. Others explored for server-side implementations but not currently licensed for use include MATLAB and Exelis ENVI and ENVI Services Engine. If you model relies on proprietary software other than Esri ArcGIS, licensing will need to be worked out prior to model integration.

Server software vs. desktop software on a server

Some software packages for modelling are used in desktop mode simply running on a server, others are truly server software, creating end points for processing data. Either mode of operation is acceptable, and each has its own set of positive and negative attributes.

Outputs

Model outputs are designed to be ingested into a Geospatial Visualization tool. Required outputs are either kml, json, shape files, or georeferenced images. Outputs for a specific app may be different, and other formats can be included. The geospatial formats are purely for display on the web-based GeoViewer. If a model is to be utilized by a mobile app, the app likely needs the outputs in a specific format. In this case, the IDEAS system simply exposes the outputs on the web. Currently model outputs are stored with a model access key, a randomly generated ID, but are not protected with any other authentication. Knowing the token is all that is needed to access the results. Sensitive data should not be used in the current implementation during this challenge.

Ingesting Models Examples

Scenario A

Best case: Model already connects to live data or uses local cache of data, has operational or engineering level code which can be executed via command line, and produces outputs in kml, shape, GeoJSON or other standard GIS format. The integration process will likely execute in about two weeks.

Tasks for this process include creating a new model execution wizard from existing templates, ensuring the model execution scripts are correctly processing the parameters and properly initializing the model, formatting the outputs appropriately for use in the GeoViewer, and validating the results.

Scenario B

Worst case: Model has no operational and no or only basic engineering level code. Inputs, parameters, and outputs are non-standard. Model complexity is moderate, perhaps several intermediate outputs, but algorithm description is readily available and understood by technical development leads. The expected length of time to update such a model and integrate it with IDEAS is at least 8 weeks.

The tasks for such integration include re-coding the model, determining appropriate input data sources, defining parameterization, developing appropriate outputs, building an execution wizard, and possibly updating the IDEAS Model API to accommodate new requirements.

This type of scenario will not be undertaken during the Challenge.

Everything else

For the in-between scenarios, when we need to do some of the above, the timeline to integrated the model will be evaluated on a case-by-case basis. The further a given model is from the base requirements of a command line interface and an available data source the more time it will take to integrate.