ModelServices

REST Specification and Service Signatures

Olaf David

James P Lyon

Wesley Lloyd

Kenneth W Rojas

ModelServices: REST Specification and Service Signatures

Olaf David
James P Lyon
Wesley Lloyd
Kenneth W Rojas

Disclaimer: This document is a draft. Service signature details and URL information may change during development because of requirement refinements.

Publication date May 15th 2012 (rev 2)

Abstract

This document specifies the structure, programming interface, and usage of model and data services as provided by the Cloud services integration platform CSIP. A client can use those services to obtain data, such as soils, management, and climate, or to execute simulation models by providing model parameter and retrieve the results. This service specification is fully based on RESTful principles using JSON as payload for transferred data. The services also contain support for service registration, lookup, and payload template specification.

Table of Contents

List of Figures

List of Tables

List of Examples

Chapter 1. Introduction

1. Cloud Services Integration Platform

The Cloud Services Innovation Platform (CSIP) provides a simple, open Web interface to its services. The CSIP APIs conform to the design principles of Representational State Transfer (REST) web services. Data is passed to and from the Web service as JSON data objects. Such an approach ensures a maximum level of interoperability and flexibility between heterogeneous systems and applications.

The ModelServices REST web service resembles the schema of Web Processing Services (WPS) developed by the OpenGIS Consortium. However, CSIP simplifies data definitions, data descriptions, and meta-data in order to allow an easy use of the offered web services. The CSIP webservice interface defines three operations that can be requested by a client and performed by a CSIP implementation.

1. *Provide a list of available model and data services.* This operation allows a client to request and receive back service meta-data documents that describe the abilities of the specific implementation. This operation provides the names and general descriptions of each of the processes offered by a CSIP instance. This operation also supports negotiation of the specification version being used for client-server interactions.

2. *Describe a specific operation in detail.* This operation allows a client to request and receive back detailed information about the model/data service that can be run on the CSIP instance, such as inputs required, their allowable formats, and the outputs that can be produced.

3. *Execute the model or data service.* This operation allows a client to run a specified process implemented by CSIP, using provided input parameter values and returning the outputs produced. Execution meta-data is returned to provide for delayed retrieval of model results, if the model executes over a longer period.

2. RESTful Webservices with JSON

This specification uses only RESTful web services. REST stands for Representational State Transfer. It is an architectural approach where software system can be build in which clients can make requests to the services and end points. REST is a widely used method to implement client/server architecture. REST allows building software applications in which clients can make requests of services that are simple to use, easy to scale, and highly inter operable. REST only requires an HTTP library to be available for most of the operations. Simplicity REST relates resources to URI's and the uniform interface. You can type different URIs for accessing different resources which much similar to typing URLs in browser.

JSON (JavaScript Object Notation) is a web focused data format. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

JSON is built on two structures:

- A collection of name/value pairs. In various languages, this is realized as an object, record, structure, dictionary, hash table, keyed list, or associative array.

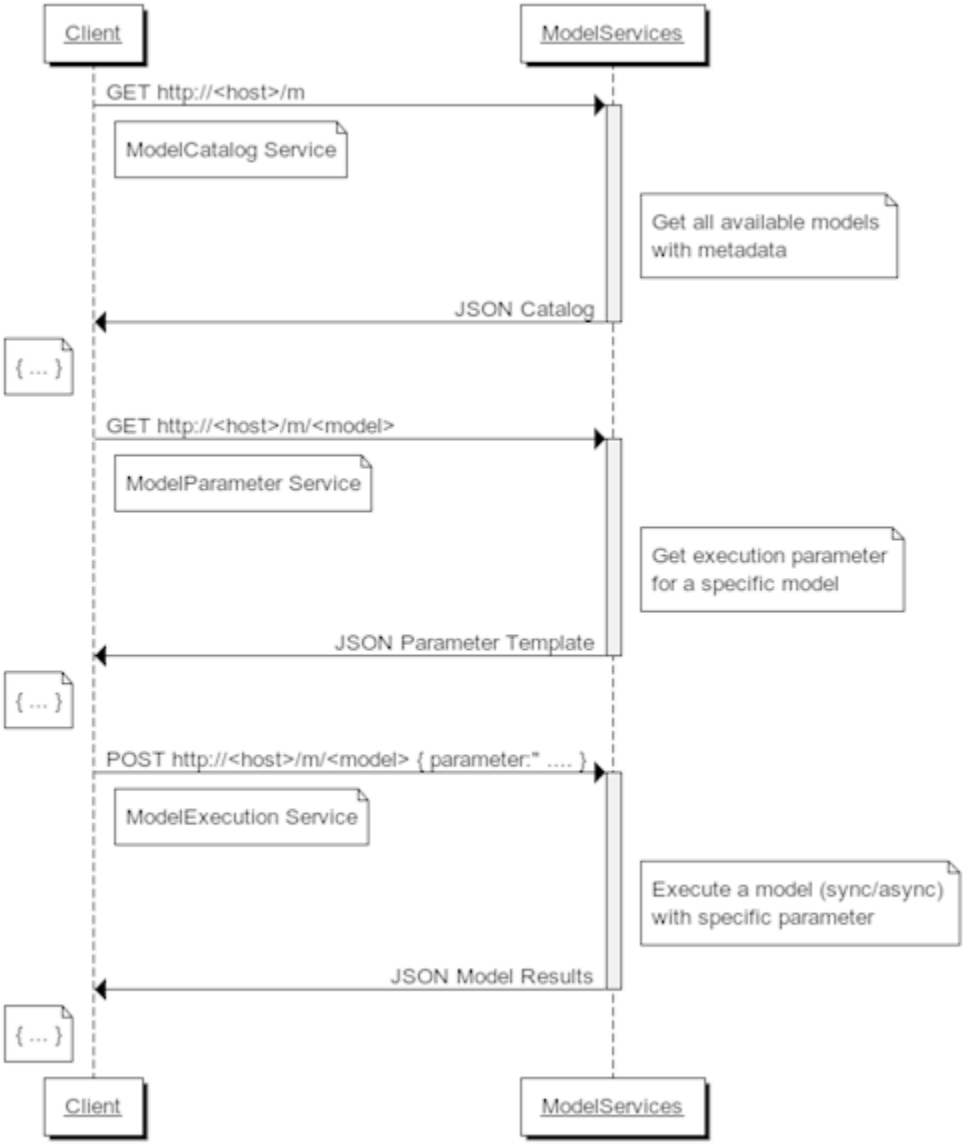
- An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

JSON is used by the ModelSevices architecture for data exchange. Model parameter, execution results, meta-information, are all being passed as JSON Objects from the client to the web-service. It provides support for catalog listing of models, exploration of model capabilities, and execution control.

Chapter 2. ModelServices

This section specifies the structure of the ModelServices, the (i) ModelCatalog service, the (ii) ModelParameter service, and (iii) ModelExecution service. The Figure below shows the purpose of the service calls in sequence.

Figure 2.1. ModelServices Exploration and Invocation Sequence



Essential for application development is only the knowledge of the ModelExecution service. A client application usually binds its code directly to the ModelExecution service. The ModelCalalog and ModelParameter Service are being

used for verifying service availability and Input Parameter signatures. A client may also explore services and service signatures "on-the-fly", by executing the first two steps (GET/GET). Which this information a execution request can be crafted and submitted for execution.

1. ModelCatalog Service

A ModelCatalog service allows the entry level exploration of available model services. Such services are listed as an array of JSON objects with sufficient meta data to call an individual service.

Request and Response are shown below. The catalog of services is requested by calling a http GET operation against a base URL.

Example 2.1. ModelCatalog Request

```
GET /<codebase>/ HTTP/1.1
Host: <host>
Accept: application/json
```

Example 2.2. ModelCatalog Response

```
{
  "service-version":
          <model services implementation version>,
  "services": [
    {
        "name": <model1 name>,
        "version": <model1 version>,
        "description": <model1 description>,
        "doc-ref": <model1 doc url>,
        "path": <model1 path>
    },
    {
        "name": <model2 name>,
        "version": <model2 version>,
        "description": <model2 description>,
        "doc-ref": <model2 doc url>,
        "path": <model2 path>
    },
    {
       ...
    }
  ]
}
```

Response elements:

`services-version` (optional)
: ModelServices specification version.

`services`
: Array of all model services available at this server. At least one service should be listed.

`name`
: The name of a model service, usually a single word.

`version` (optional)
: The version of this model service.

`description` (optional)
: A brief, one-line description of the model service.

`doc-ref` (optional)
: A reference to further documentation of the web service

`path`
: The path for ModelParameter/ModelExecution Services. The path can be relative to the base URL or a full URL

A request for a example catalog and the response for model services is shown in the examples below.

Example 2.3. Example ModelCatalog Request

```
GET /csip/m HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
```

Example 2.4. Example ModelCatalog Response

```
"services" : [
    {
        "name": "Rusle2",
        "version": "1.1",
        "description": "Revised Universal Soil Loss Equation
                        - Rill and Sheet Erosion",
        "path": "/rusle2/1.1"
    },
    {
        "name": "EFH2",
        "version": "1.1",
        "description": "Storm runoff model based on conventions
                    in Engineering Field Handbook.",
        "path": "m/efh2/1.1"
    }
   ...
]
```

The paths in this example are relative to the base URL used to obtain that catalog.

2. ModelParameter Service

The ModelParameter service provides detailed execution information about a single ModelService. Input Parameter are specified with at least name, default value, and physical unit. The generated '`parameter`' JSON object can be used as a template for an ExecutionService call. The default parameter values are expected to produce valid ModelExecution results.

The ModelParameter Request is performed as a HTTP GET with the <path> and <version> obtained from the ModelCatalog result.

Example 2.5. ModelParameter Request

```
GET /<codebase>/<path>/<version> HTTP/1.1
Host: <host>
Accept: application/json
```

Example 2.6. ModelParameter Response

```
{
    "parameter": [
        {
            "name": "<parameter name>",
            "value": "<default value>",
            "unit": "<physical unit>",
            "min": "<minimum value>",
            "max": "<maximum value>",
            "description": "<parameter description>"
        },
        {
            "name": "<parameter name>",
            "value": "<default value>",
            "unit": "<physical unit>",
        },
        ..
    ],
    "metainfo": { }
}
```

Response elements:

`name`
: parameter name.

description (optional)
: brief parameter description.

value
: default value

unit
: the physical unit of the parameter value.

min (optional)
: an minimum value.

max (optional)
: an maximum value.

Example 2.7. Example ModelParameter Request

```
GET /csip/m/efh2/1.1 HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
```

Calling a ModelParameter request requires a URL that is being constructed from the ModelCatalog information. The host name of that URL is the base URL of the ModelCatalog call if the pat catalog entry is relative. Note that the specific service version as provided in the catalog must be appended to the path, hence: `GET /csip/m/efh2/1.1`

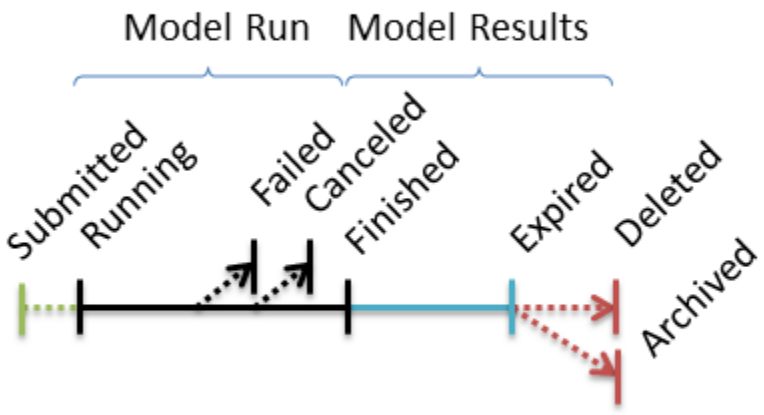
Example 2.8. Example ModelParameter response for EFH2


```
{
    "parameter": [
        {
            "name": "precip",
            "value": 14,
            "unit": "inch"
        },
        {
            "name": "runoffcurvenumber",
            "value": 90
        },
        {
            "name": "stormtype",
            "value": "I"
        },
        {
            "name": "watershedlength",
            "value": 1500,
            "unit": "ft"
        },
        {
            "name": "watershedslope",
            "value": 0.5,
            "unit": "%"
        }
    ],
    "metainfo": {}
}
```


3. ModelExecution Service

A ModelExecution Service runs the simulation model. It expects model parameter as input in a JSON envelope and optional metadata controlling the model execution. This service call provides the model output as a result JSON object.

Figure 2.2. Execution phases



The Figure above shows the execution phases that ate common for all model services. There are two main groups with respect to the actual model run and the management of model results. Since a model service can be submitted in two modes "`sync`" and "`async`" there are two variant of managing those phases (Further details below). A ModelExecution request submits a model run, that eventually will cause it to get into a "`Running`" state. The model can then successfully terminate ("`Finished`"), it can for some reason fail ("`Failed`"), or canceled by user request ("`Cancelled`"). This concludes the model run phases. All the following phases relate to output data management (if applicable). Model output data will be kept available for retrieval, until it expires ("`Expired`"). The time to keep output data available can be controlled via the request meta info entry "`keep_results`". In addition there can also be a meta info entry to decide what will happen next, deleting the output data "`Delete`" or archive it for long term storage "`Archive`"

Each ModelExecution request and response payload contains three sections, as shown below.

Example 2.9. ModelExecution Response

```
{
  "metainfo" : {

   },
  "parameter" : [

   ],
  "result" : [

   ]
}
```

metainfo (optional)
: providing meta data to the service to control execution, and returned by the service to indicate execution status and statistics and other information

parameter
: providing input data for the model as an array of singe parameter JSON objects.

result
: result data after model execution as an array of JSON data objects

Example 2.10. ModelExecution request example

```
{
    "parameter": [
        {
            "name": "precip",
            "value": 14,
            "unit": "in"
        },
        {
            "name": "runoffcurvenumber",
            "value": 90
        },
        {
            "name": "stormtype",
            "value": "I"
        },
        {
            "name": "watershedlength",
            "value": 1500,
            "unit": "ft"
        },
        {
            "name": "watershedslope",
            "value": 0.5,
            "unit": "%"
        }
    ],
    "metainfo": {
          "mode":"sync",
          "keep_results: "200000"
    }
}
```

The example above shows a request JSON for calling the a model service (EFH2).

Example 2.11. ModelExecution invocation

```
POST /csip/m/efh2/1.1 HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
```

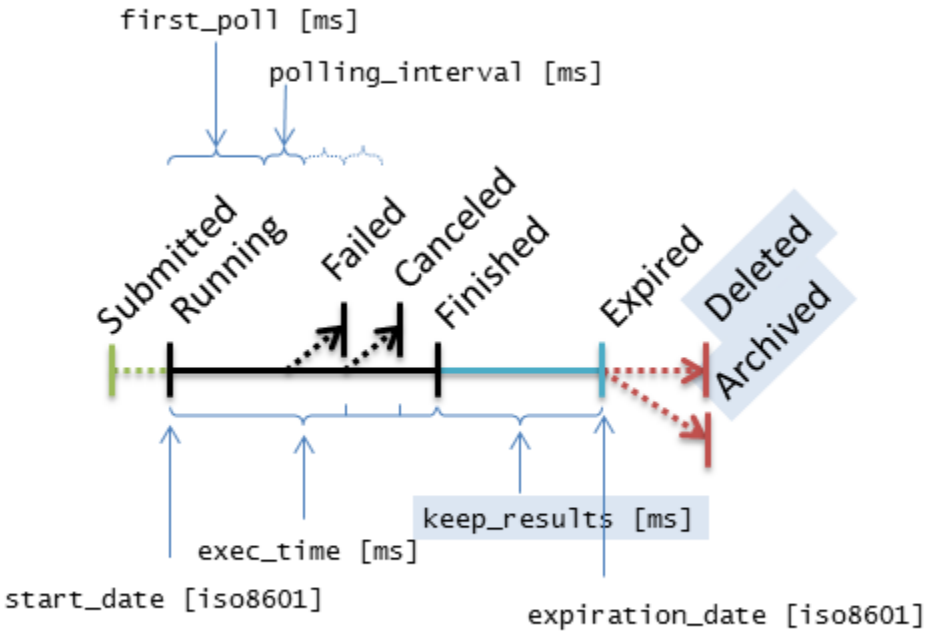
Example 2.12. ModelExecution response example

```
{
    "parameter": [
        {
            "name": "precip",
            "value": 14,
            "unit": "inch"
        },
        {
            "name": "runoffcurvenumber",
            "value": 90
        },
        {
            "name": "stormtype",
            "value": "I"
        },
        {
            "name": "watershedlength",
            "value": 1500,
            "unit": "ft"
        },
        {
            "name": "watershedslope",
            "value": 0.5,
            "unit": "%"
        }
    ],
    "metainfo": {}
}
```

3.1. Metainfo variables

The metainfo JSON object contains variables that control the model execution or provide feedback for execution.

Figure 2.3. Variables



The result data (output JSON and output files) will be kept for a period of time after the model run successfully finished. If the model run failed or was cancelled, and no output will be stored. The time to keep results can be specified in the request using the `"keep-results"` entry within the `"metadata"` element.

In asynchronous execution mode, the returned `metadata` property `"polling_intervall"` provides a reasonable hint for the client when to poll again for status update. This value is model specific. Expect higher values for long running models and smaller values for models that run only for a short period. A client should perform subsequent requests after periods that are not smaller than this value. Example: If a model takes on average 1 minute to finish, there is no need for the client to poll every 100 ms for a status update, the `"polling_interval"` in such a case would be 5000 (5 seconds). There is no advantage for a client to use a shorter polling interval. The polling interval is provided by CSIP based on the average model runtime.

Table 2.1. Metainfo tags

Key	Description	Required	Provided By	Example
`mode`	The execution mode of the service, `"sync"` or `"async"`	NO, defaults to `sync`	request	`"mode" : "async"`
`expiration_action`	Action to take after expiration date: `"Delete"` | `"Archive"`	NO, defaults to `"Delete"`	request	"expiration_action" : "archive"
`keep_results`	Number of milliseconds to keep the results **after** the model finished execution	NO, defaults to "300000" (5 minutes)	request	`"keep_results": 600000`

Key	Description	Required	Provided By	Example
`timezone`	timezone	NO, all time information defaults to UTC	request	"timezone": "America/Denver"
`status`	the status of the model execution: `"Submitted"` | `"Finished"` | `"Canceled"` | `"Failed"` | `"Running"` | `"Unknown"`	-	response	`"status" :"Finished"`
`cpu_time`	the time in ms for service execution	-	response	"cpu_time": "12324"
`start_date`	the time stamp for service execution	-	response	"start_date" : "2012-03-21T15:23:42-0700"
`expiration_date`	thew date when the output date will expire.	-	response	"expiration_date" : "2012-03-21T15:29:42-0700"
`polling_interval`	Recommended number of milliseconds between two client polls.	-	response	`"polling_interval": 2000`
`first_poll`	Recommended time in ms to wait for the first poll	-	response	`"polling_interval": 20000`
`suid`	simulation unique identifier	-	response	"suid" : "8553567a-ee1f-4270-81fb-ec9e9a5676a6"

Notes:

- All Dates are provided in ISO-8601 date format with timezone field.(e.g. '`2012-03-21T15:23:42-0700`'). If a request provides timezone information all dates will be mapped to this timezone. If not, dates are UTC.
- The expiration action `"Archive"` is not implemented yet.
- Timespans units are always milliseconds, for both, request and response `metainfo` values.
- suid's are Version 1 UUID's (see RFC 4122: A Universally Unique IDentifier (UUID) URN Namespace). Thus, the suid creation time can be retrieved from such a `suid`. Suid stands for "simulation unique identifier".

3.2. Service Execution Modes

A synchronous ModelExecution service returns when the model run is finished, thus the calling client is blocked during this operation. This is the default way to execute a service. A asynchronous ModelExecution service starts a model and immediately returns to the client. Subsequent model service calls must be made to gain information about the model status and find out if the model finished. Each model service can be executed both ways, synchronously and asynchronously. The metainfo key "mode" controls the execution mode when submitting the initial request.

The example below starts a model in asynchronous mode.

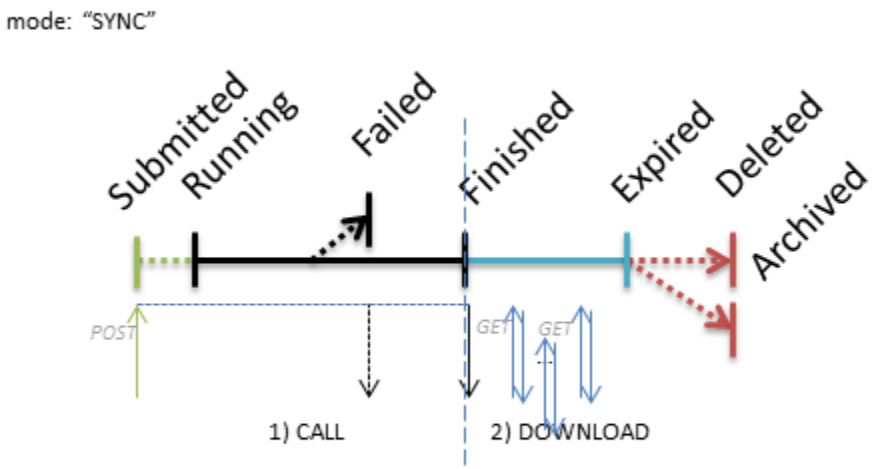
Example 2.13. ModelExecution response example

```
{
    "metainfo": {
            "mode": "async",
             ...
     }
 …
```

3.2.1. Synchronous Execution

The figure below shows the caller sequence for synchronous execution. The POST request with the model parameter returns if either the model finished successfully, or it failed. During that time the client caller is blocked. In such a case the client must account for issues like "call threading" to keep for example a user interface responsive.

Figure 2.4. Synchronous Service Phases

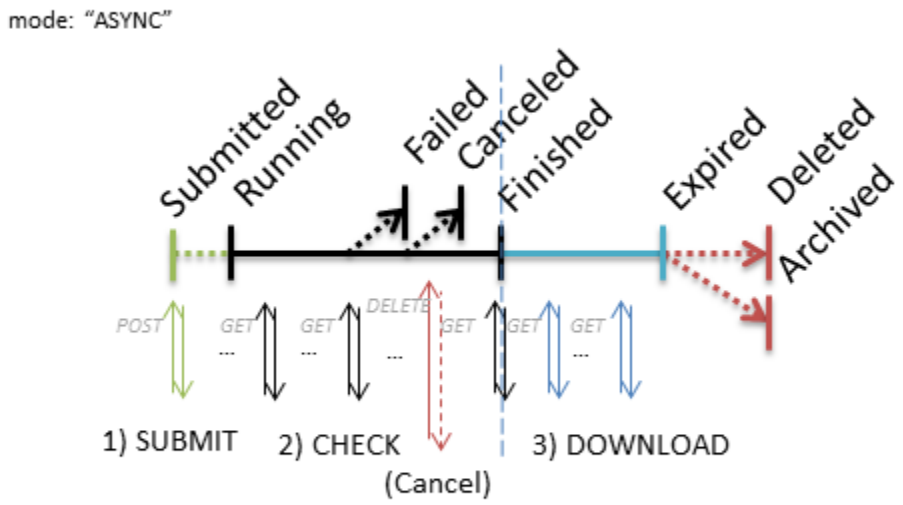


After the model is done executing, optional output data can be fetched until it expires.

3.2.2. Asynchronous Execution

While synchronous execution might be acceptable for short running models (execution time < 2 sec) it is unpractical for long running models. Such models might execute for minutes, hours, or even days. The figure below shows the principal flow of calls for such asynchronous execution. If the mode is set to "async", the initial service call (POST) submits the model for execution and returns immediately.

Figure 2.5. Asynchronous Service Phases



The response metainfo element contains the `suid`, a unique key, that uniquely defines the model run. The model status indicates its submission. In addition the client finds two entries, "`first_poll`" and "`polling_interval`". Now the client should wait "`first_poll`" milliseconds, for the next GET in order to query the status.

Example 2.14. Result for submitted model execution

```
{
  "metainfo": {
    "status": "Submitted",
    "suid": "182b8505-7534-11e1-b93f-1d1c56c85325",
    "tstamp": "2012-03-23T22:04:00+0000",
    "polling_interval": "2000",
    "first_poll": "25000",
  },
  "parameter": [
..
  ]
}
```

A Rest query is being executed using the `suid`. The call as shown below is constructed with the '`/csip/q`' path and the appended `suid`.

Example 2.15. Async Query

```
GET /csip/q/182b8505-7534-11e1-b93f-1d1c56c85325 HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
```

The query result shows that the model is still running, and has not finished yet. Any subsequent check on the model status like this should be done after "`polling_interval`" milliseconds.

Example 2.16. Query Result

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "metainfo": {
    "status": "Running",
    "suid": "182b8505-7534-11e1-b93f-1d1c56c85325",
    "tstamp": "2012-03-23T22:04:00+0000",
    "polling_interval": "2000",
    "first_poll": "25000",
  },
  "parameter": [
..
  ]
}
```

Eventually the status will switch to `Finished` and the model output it contained in the `result` JSON section, and can be fetched from CSIP. It is important to keep the `suid` during server side model execution, it serves as token to identify the active model run and its data.

3.3. Optional Data Download after Execution

Simulation models produce data output usually into files. The CSIP web services, however, present selected model output as `result` entries to the client. Such output might be obtained from the 'in-memory' model directly, or is is extracted from the model output files. In some cases is is required to allow the client to access the native model output files. For such a use case, CSIP provides a query service. An example is shown below.

A model finishes and provides in its `result` JSON a list of file files for download. A client might fetch the files as referenced by the provided URL. The file size is provided too.

Example 2.17. Model response with data download


```
{
  "metainfo": {
    "status": "Finished",
    "suid": "182b8505-7534-11e1-b93f-1d1c56c85325",
    "tstamp": "2012-03-23T22:04:00+0000",
    "polling_interval": "2000",
    "first_poll": "25000",
    "cputime": "28155",
    "expiration_date": "2012-03-23T22:09:28+0000"
  },
  "parameter": [
    {
      "name": "wepsrun",
      "value": "weps.run"
    },
    {
      "name": "climate",
      "value": "cli_gen.cli"
    },
    {
      "name": "wind",
      "value": "win_gen.win"
    },
    {
      "name": "management",
      "value": "S Wheat-Beet-SB CMZ 1.man"
    },
    {
      "name": "soils",
      "value": "Heldt_48_90_CL.ifc"
    }
  ],
  "result": [
    {
      "name": "sci",
      "value": "http:\/\/csip.engr.colostate.edu:8083\/csip\/q\/182b8505-7534-11e1-b93f-1d1c56c85325\/sci_ene
      "size": "21729"
    },
    {
      "name": "stir",
      "value": "http:\/\/csip.engr.colostate.edu:8083\/csip\/q\/182b8505-7534-11e1-b93f-1d1c56c85325\/strir_e
      "size": "91638"
    },
  ]
```


Note that a client must fetch any temporary stored server-side data before it expires. The expiration date is given in `metainfo`. In addition, the `suid` is being used to assign the files to the specific model run, `suid` `182b8505-7534-11e1-b93f-1d1c56c85325` in the example above.

3.4. Error Messages

When a data service fails internally, or the model execution fails because of invalid input, the service provides an error message. This message is embedded into the metainfo section of the JSON response.

The figure below shows a response containing an error message. It can be assumed, that the presence of an error key in `metainfo` indicates an execution problem.

Example 2.18. Error Handling

```
{
"metainfo": {
  "status":"Failed",
  "error": "Insufficient input data"
}
"result": [
]
"parameter":[
 ...
]
}
```

The response containing the error will still contain original parameter payload and other original request JSON content.

3.5. Ensemble Execution

The method of submitting multiple sets of parameters for model execution is called an ensemble run. Each individual parameter set represents an individual run. CSIP will split up all parameter sets and will execute a single run with each parameter set. All runs will execute in parallel.

An example run is finished if all single runs have finished. The `result` JSON will the contain the results of the model run. Like the array of parameters, the results will contain the array of result sets in correct order. The first result set relates to the first parameter set.

Example 2.19. Ensemble Execution

```
{
"metainfo": {
  "parametersets":25
}
"parameter":[
   [
    "name:"p1",
    "value":"23",
    ...
   ],
   [
    "name":"p1",
    "value:"24",
   ...
   ]
   ...
]
}
```

Ensemble runs can be, like single runs, executed in asynchronous and synchronous mode.

Chapter 3. CSIP Service Signatures

This chapter introduces the various service signatures, as provided by CSIP. The parameter and result elements may change during development because of requirement adjustments and service improvements.

All ModelService URLs served up by CSIP do have a common structure. The service context path always starts with '`csip`', followed with an '`m`' indicating the modeling services category. There are other service categories in CSIP that are not part of this document, such as database services ('d'), or the query service 'q' as described earlier.

Example 3.1. CSIP Service URL

```
http://<host>:<port>/csip/m/<model>/<version>"
```

Description:

<host>
: The service host name.

<port>
: The service port, if not specified the port is 80.

<model>
: The name of the model service. It usually corresponds to a known simulation model.

<version>
: The version of the model service. This can be an arbitrary string, it does not have to be a numerical value.

An example CSIP service URL is shown below for the Rusle2 model.

Example 3.2. CSIP URL example for Rusle2

```
http://csip.engr.colostate.edu:8083/csip/m/rusle2/1.1"
```

This URL represents the version 1.1 of the Rusle2 model service at port `8083` on `csip.engr.colostate.edu`.

1. RUSLE2

RUSLE2 is a computer model containing both empirical and process-based science that predicts rill and interrill soil erosion by rainfall and runoff. The computational engine of RUSLE2 is available as a web service in CSIP.

1.1. ModelCatalog

Request

```
GET /csip/m HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

"services": [
    {
```

```
        "name": "Rusle2",
        "version": "1.1",
        "description": "Revised Universal Soil
                Loss Equation - Rill and Sheet Erosion",
        "path": "rusle2/1.1"
    },
    {
        <other services>
    }
]
```

1.2. ModelParameter

Request

```
GET /csip/m/rusle2/1.1 HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "metainfo": {
    },

    "parameter": [
        {
            "name": "latitude",
            "description": "Location latitude, continental US range. For climate lookup",
            "unit": "deg",
            "min": "25",
            "max": "50",
            "value": 36.250515
        },
        {
            "name": "longitude",
            "description": "Location longitude, continental US range. For climate lookup",
            "unit": "deg",
            "min": "-124",
            "max": "-66",
            "value": -84.026264
        },
        {
            "name": "steepness",
            "description": "Average slope steepness. R2:SLOPE_STEEP",
            "unit": "%",
            "min": "0.001",
            "max": "100",
            "value": 3
        },
        {
            "name": "length",
            "description": "Average horizontal slope length. R2:SLOPE_HORIZ",
            "unit": "ft",
            "min": "0.001",
            "max": "1000",
            "value": 150
        },
        {
            "name": "topo_steepness",
```

```
            "description": "Slope steepness for each segment. steepness values as array. R2:TOPO_STEEP",
            "unit": "%",
            "min": "0.001",
            "max": "100",
            "value": [
                4,
                2
            ]
        },
        {
            "name": "topo_length",
            "description": "Horizontal slope length for each topo segment. length values as array. R2:TOPO_HO
            "unit": "ft",
            "min": "0.001",
            "max": "1000",
            "value": [
                30,
                120
            ]
        },
        {
            "name": "fuel",
            "description": "fuel type. R2:SLOPE_FUEL_PTR",
            "value": {
                "path": "fuels\\Local",
                "name": "Diesel",
                "type": "FUEL",
                "key": "8708124062829849523",
                "version": "0.8",
                "metadata": {
                   "owner": "dave.lightle",
                   "group": "NRCS_Fld_Office",
                   "date": "2008-08-12 13:33:31"
                }
            }
        },
        {
            "name": "soil",
            "description": "soil (arrayed). String keys returned from CSIP soil service. Corresponds to R2:SO
            "value": [
                123456,
                234567
            ]
        },
        {
            "name": "soil_length",
            "description": "soil segment lengths (arrayed). R2:SOIL_HORIZ.",
            "unit": "ft",
            "value": [
                100,
                50
            ]
        },
        {
            "name": "management",
            "description": "management description, from LMOD query or constructed in URB editor.",
            "value": [
                {
                    "lmod_file": {
                        "path": "managements\\CMZ 01\\b.Mullti-year Rotation Templates\\Alfalfa rotations",
                        "name": "soybeans; nr, Sfcult 2x; corn FC twist, anhyd, sfcult 2x; Alf 3yrs FC twist
                        "key": "9177684190369916811",
                        "type": "MANAGEMENT",
                        "version": "0.8",
                        "metadata": {
```

```
                        "meta": [
                            {
                                "name": "generator",
                                "content": "LMOD / CakePHP"
                            },
                            {
                                "name": "owner",
                                "content": "robin.martinek"
                            },
                            {
                                "name": "group",
                                "content": "NRCS_Sta_Agron"
                            },
                            {
                                "name": "date",
                                "content": "2007-09-14 09:09:02"
                            }
                        ]
                    },
                    "params": {
                        "param": [
                            {
                                "name": "DURATION_IN_MAN",
                                "type": "Int",
                                "unit": "year",
                                "data": {
                                    "datum": [
                                        5
                                    ]
                                }
                            },
                            {
                                "name": "EXT_RES_PTR",
                                "type": "ObR",
                                "dim": {
                                    "name": "OP_DATE"
                                },
                                "file_type": "RESIDUE",
                                "data": {
                                    "datum": [
                                        {
                                            "file_key": null
                                        },
                                        {
                                            "file_key": null
                                        },
                                        {
                                            "file_key": null
                                        },
                                        {
                                            "file_key": null
                                        },
                                        {
                                            "file_key": null
                                        },
                                        {
                                            "file_key": null
                                        },
                                        {
                                            "file_key": null
                                        },
                                        {
                                            "file_key": null
                                        },
                                        {
```

```
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                },
                                {
                                    "file_key": null
                                }
                            ]
                        }
                    },
                    {
                        "name": "MAN_FUEL_USE_PTR",
                        "type": "ObR",
                        "file_type": "FUEL",
                        "data": {
                            "datum": [
                                {
                                    "file_key": "8708124062829849524",
                                    "value": "fuels\\default"
                                }
                            ]
                        }
                    },
                    {
                        "name": "MAN_IRRIG_SUB_OBJ_PTR",
                        "type": "SbR",
                        "data": {
                            "datum": [
                                "MAN_IRRIG_NONE_SUB_OBJ"
                            ]
                        }
                    },
                    {
                        "name": "MAN_OP_DEPTH",
```

```
                "type": "Flt",
                "unit": "inch",
                "dim": {
                    "name": "OP_DATE"
                },
                "data": {
                    "datum": [
                        4,
                        4,
                        1.5,
                        null,
                        7,
                        4,
                        4,
                        4,
                        2.5,
                        null,
                        7,
                        4,
                        1.5,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null
                    ]
                }
            },
            {
                "name": "MAN_OP_FUEL_USE_PTR",
                "type": "ObR",
                "dim": {
                    "name": "OP_DATE"
                },
                "file_type": "FUEL",
                "data": {
                    "datum": [
                        {
                            "file_key": "8708124062829849524",
                            "value": "fuels\\default"
                        },
                        {
                            "file_key": "8708124062829849524",
                            "value": "fuels\\default"
                        },
                        {
                            "file_key": "8708124062829849524",
                            "value": "fuels\\default"
                        },
                        {
                            "file_key": "8708124062829849524",
                            "value": "fuels\\default"
                        },
                        {
                            "file_key": "8708124062829849524",
                            "value": "fuels\\default"
                        },
                        {
                            "file_key": "8708124062829849524",
                            "value": "fuels\\default"
                        },
                        {
```

```
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            },
                            {
                                "file_key": "8708124062829849524",
                                "value": "fuels\\default"
                            }
                        ]
                    }
                },
                {
                    "name": "MAN_OP_SPEED",
                    "type": "Flt",
```

```
                "unit": "mile/hour",
                "dim": {
                    "name": "OP_DATE"
                },
                "data": {
                    "datum": [
                        5,
                        5,
                        5,
                        null,
                        5,
                        5,
                        5,
                        5,
                        5,
                        null,
                        5,
                        5,
                        5,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null
                    ]
                }
            },
            {
                "name": "MAN_OP_VEG_NUM_HARV_UNITS",
                "type": "Flt",
                "unit": "acre^-1",
                "dim": {
                    "name": "OP_DATE"
                },
                "data": {
                    "datum": [
                        null,
                        null,
                        20,
                        null,
                        null,
                        null,
                        null,
                        null,
                        112,
                        null,
                        null,
                        null,
                        1.5,
                        1.5,
                        1.6,
                        1.5,
                        1.5,
                        2.5,
                        1.65,
                        1.65,
                        2.5
                    ]
                }
            },
            {
                "name": "MAN_OP_VEG_RETARD_CLASS",
```

```
                "type": "Lst",
                "dim": {
                    "name": "OP_DATE"
                },
                "data": {
                    "datum": [
                        null,
                        null,
                        "MOD_LOW_RETARD",
                        null,
                        null,
                        null,
                        null,
                        null,
                        "NO_RETARD",
                        null,
                        null,
                        null,
                        "MOD_HIGH_RETARD",
                        "MOD_HIGH_RETARD",
                        "MOD_HIGH_RETARD",
                        "MOD_HIGH_RETARD",
                        "MOD_HIGH_RETARD",
                        "EXTREME_RETARD",
                        "MOD_HIGH_RETARD",
                        "MOD_HIGH_RETARD",
                        "EXTREME_RETARD"
                    ]
                }
            },
            {
                "name": "NUM_ROTATION_BUILDER_MANS",
                "type": "Int",
                "dim": {
                    "name": "NUM_ROTATION_BUILDER_MANS"
                },
                "data": {
                    "datum": [
                        1
                    ]
                }
            },
            {
                "name": "OP_DATE",
                "type": "Dte",
                "unit": "date",
                "dim": {
                    "name": "OP_DATE"
                },
                "data": {
                    "datum": [
                        "5/15/2",
                        "5/20/2",
                        "5/25/2",
                        "10/10/2",
                        "10/15/2",
                        "10/20/2",
                        "5/5/3",
                        "5/10/3",
                        "5/10/3",
                        "10/10/3",
                        "10/15/3",
                        "4/20/4",
                        "4/20/4",
                        "7/15/4",
```

```
                                "9/1/4",
                                "6/1/5",
                                "7/15/5",
                                "9/1/5",
                                "6/1/6",
                                "7/15/6",
                                "9/1/6"
                            ]
                        }
                    },
                    {
                        "name": "OP_PTR",
                        "type": "ObR",
                        "dim": {
                            "name": "OP_DATE"
                        },
                        "file_type": "OPERATION",
                        "data": {
                            "datum": [
                                {
                                    "file_key": "8708124062829849102",
                                    "value": "operations\\Cultivator, field 6-12 in sweeps"
                                },
                                {
                                    "file_key": "8708124062829849102",
                                    "value": "operations\\Cultivator, field 6-12 in sweeps"
                                },
                                {
                                    "file_key": "8708124062829849131",
                                    "value": "operations\\Drill or airseeder, double disk  "
                                },
                                {
                                    "file_key": "8708124062829849228",
                                    "value": "operations\\Harvest, killing crop 20pct standing st
                                },
                                {
                                    "file_key": "8708124062829849118",
                                    "value": "operations\\Chisel, twisted shovel "
                                },
                                {
                                    "file_key": "8708124062829849287",
                                    "value": "operations\\Fert applic. anhyd knife 12 in"
                                },
                                {
                                    "file_key": "8708124062829849102",
                                    "value": "operations\\Cultivator, field 6-12 in sweeps"
                                },
                                {
                                    "file_key": "8708124062829849102",
                                    "value": "operations\\Cultivator, field 6-12 in sweeps"
                                },
                                {
                                    "file_key": null,
                                    "value": "operations\\planter, double disk opnr "
                                },
                                {
                                    "file_key": "8708124062829849227",
                                    "value": "operations\\Harvest, killing crop 50pct standing st
                                },
                                {
                                    "file_key": "8708124062829849118",
                                    "value": "operations\\Chisel, twisted shovel "
                                },
                                {
                                    "file_key": "8708124062829849102",
```

```
                    "value": "operations\\Cultivator, field 6-12 in sweeps"
                },
                {
                    "file_key": "8708124062829849131",
                    "value": "operations\\Drill or airseeder, double disk  "
                },
                {
                    "file_key": "8708124062829849219",
                    "value": "operations\\Harvest, hay, legume "
                },
                {
                    "file_key": "8708124062829849219",
                    "value": "operations\\Harvest, hay, legume "
                },
                {
                    "file_key": "8708124062829849219",
                    "value": "operations\\Harvest, hay, legume "
                },
                {
                    "file_key": "8708124062829849219",
                    "value": "operations\\Harvest, hay, legume "
                },
                {
                    "file_key": "8708124062829849219",
                    "value": "operations\\Harvest, hay, legume "
                },
                {
                    "file_key": "8708124062829849219",
                    "value": "operations\\Harvest, hay, legume "
                },
                {
                    "file_key": "8708124062829849219",
                    "value": "operations\\Harvest, hay, legume "
                },
                {
                    "file_key": "8708124062829849219",
                    "value": "operations\\Harvest, hay, legume "
                }
            ]
        }
    },
    {
        "name": "RES_ADDED",
        "type": "Flt",
        "unit": "pound/acre",
        "dim": {
            "name": "OP_DATE"
        },
        "data": {
            "datum": [
                3245.2988109423,
                null,
                null,
                399.2648907264,
                null,
                null,
                null,
                null,
                null,
                3136,
                null,
                null,
                null,
                405,
                405,
```

```
                    432,
                    398.62212105315,
                    405,
                    675,
                    438.48433315847,
                    445.5
                ]
            }
        },
        {
            "name": "ROT_BUILD_CORRECTION_TYPE",
            "type": "Lst",
            "dim": {
                "name": "NUM_ROTATION_BUILDER_MANS"
            },
            "data": {
                "datum": [
                    "ROT_BUILD_CORRECT_TYPE_NONE"
                ]
            }
        },
        {
            "name": "ROT_BUILD_END_DATE",
            "type": "Dte",
            "unit": "date",
            "dim": {
                "name": "NUM_ROTATION_BUILDER_MANS"
            },
            "data": {
                "datum": [
                    "1/1/0"
                ]
            }
        },
        {
            "name": "ROT_BUILD_MAN_IRRIG_SUB_OBJ_PTR",
            "type": "SbR",
            "data": {
                "datum": [
                    "MAN_IRRIG_NONE_SUB_OBJ"
                ]
            }
        },
        {
            "name": "ROT_BUILD_START_DATE",
            "type": "Dte",
            "unit": "date",
            "dim": {
                "name": "NUM_ROTATION_BUILDER_MANS"
            },
            "data": {
                "datum": [
                    "1/1/0"
                ]
            }
        },
        {
            "name": "ROTATION_BUILDER_MAN_PTRS",
            "type": "ObR",
            "dim": {
                "name": "NUM_ROTATION_BUILDER_MANS"
            },
            "file_type": "MANAGEMENT",
            "data": {
                "datum": [
```

```
                    {
                        "file_key": null,
                        "value": "#ENTRY_NONE"
                    }
                ]
            }
        },
        {
            "name": "ROTATION_BUILDER_PTR",
            "type": "SbR",
            "data": {
                "datum": [
                    "ROTATION_BUILDER"
                ]
            }
        },
        {
            "name": "RUSLE2:SCIENCEVERSION",
            "type": "Str",
            "data": {
                "datum": [
                    "20061020"
                ]
            }
        },
        {
            "name": "VEG_PTR",
            "type": "ObR",
            "dim": {
                "name": "OP_DATE"
            },
            "file_type": "VEGETATION",
            "data": {
                "datum": [
                    {
                        "file_key": null
                    },
                    {
                        "file_key": null
                    },
                    {
                        "file_key": "8708124062829849611",
                        "value": "vegetations\\Soybean, group 0 and I,  7in rows"
                    },
                    {
                        "file_key": null
                    },
                    {
                        "file_key": null
                    },
                    {
                        "file_key": null
                    },
                    {
                        "file_key": null
                    },
                    {
                        "file_key": null
                    },
                    {
                        "file_key": "8708124062829849962",
                        "value": "vegetations\\Corn, grain"
                    },
                    {
                        "file_key": null
```

```
                                    },
                                    {
                                        "file_key": null
                                    },
                                    {
                                        "file_key": null
                                    },
                                    {
                                        "file_key": "8708124062829850359",
                                        "value": "vegetations\\Alfalfa, spring seed "
                                    },
                                    {
                                        "file_key": "8708124062829850356",
                                        "value": "vegetations\\Alfalfa, spring seed regrowth after cu
                                    },
                                    {
                                        "file_key": "8708124062829850357",
                                        "value": "vegetations\\Alfalfa, spring seed senes to y2 regro
                                    },
                                    {
                                        "file_key": "8708124062829850355",
                                        "value": "vegetations\\Alfalfa, yr2 regrowth after cutting "
                                    },
                                    {
                                        "file_key": "8708124062829850355",
                                        "value": "vegetations\\Alfalfa, yr2 regrowth after cutting "
                                    },
                                    {
                                        "file_key": "8708124062829850352",
                                        "value": "vegetations\\Alfalfa, yr2 senes to yr3 regrowth"
                                    },
                                    {
                                        "file_key": "8708124062829850366",
                                        "value": "vegetations\\Alfalfa, yr3 regrowth after cutting "
                                    },
                                    {
                                        "file_key": "8708124062829850366",
                                        "value": "vegetations\\Alfalfa, yr3 regrowth after cutting "
                                    },
                                    {
                                        "file_key": "8708124062829850364",
                                        "value": "vegetations\\Alfalfa, yr3 senes to yr4 regrowth"
                                    }
                                ]
                            }
                        }
                    ]
                }
            }
        },
        {
            "lmod_file": {
                "path": "managements\\CMZ 01\\a.Single Year/Single Crop Templates\\Rye",
                "name": "rye; FC, st pt, disk, fcult, z1",
                "key": "9177684190369918363",
                "type": "MANAGEMENT",
                "version": "0.8",
                "metadata": {
                    "generator": "LMOD / CakePHP",
                    "url": "http://<lmod>/rest_files/view/9177684190369918363?format=json"
                    "owner": "dave.lightle",
                    "group": "NRCS_Sta_Agron",
                    "date" : "2002-09-30 13:47:25"
                },
                "params": {
```

```
                        "comment": "(content omitted)"
                    }
                }
            },
            {
                "name": "mgmt_length",
                "description": "soil segment lengths (arrayed). R2:SOIL_HORIZ.",
                "unit": "ft",
                "value": [
                    100,
                    50
                ]
            },
            {
                "name": "rock_cover",
                "description": "rock cover for each (mgmt, soil) pair moving down the slope. R2:SLOPE_ROC
                "unit": "%",
                "min": "0",
                "max": "100",
                "value": [
                    0,
                    0,
                    0
                ]
            }
        ]
    }
]
}
```

1.3. ModelExecution

Request

```
POST /csip/m/rusle2/1.1 HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
Content-Type:application/json
Content-Length: <length>

{
    "metainfo": {
    },

    "parameter": [
        //.. see ModelParameter response
    ],
}
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "metainfo": {
        "status": "Finished",
        "suid": "182b8505-7534-11e1-b93f-1d1c56c85325",
        "tstamp": "2012-03-23T22:04:00+0000",
        "polling_interval": "2000",
        "first_poll": "2000",
        "cputime": "1055",
```

```
        "expiration_date": "2012-03-23T22:09:28+0000"
    },

    "parameter": [
        //.. see request
    ],
    "result": [
        {
            "name": "slope_delivery",
            "value": 3,
            "unit": "t/ac/y"
        },
        {
            "name": "t_value",
            "value": 5,
            "unit": "t/ac/y"
        },
        {
            "name": "degrade",
            "description": "Soil degradation",
            "value": 2.60436106203552,
            "unit": "t/ac/y"
        },
        {
            "name": "climate",
            "description": "the climate record used for this run (based on provided lat/lon)",
            "value": "climates\\USA\\Tennessee\\Anderson County"
        }
    ]
}
```

2. WEPS

The Wind Erosion Prediction System (WEPS) is a process-based, daily time-step, computer model that predicts soil erosion via simulation of the fundamental processes controlling wind erosion. WEPS can calculate soil movement, estimate plant damage, and predict PM-10 emissions when wind speeds exceed the erosion threshold.

2.1. ModelCatalog

Request

```
GET /csip/m HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

"services": [
    {
        "name": "WEPS",
        "version": "1.1",
        "description": "Wind Erosion Prediction System",
        "path": "weps/1.1"
    },
    {
        <other services>
    }
```

```
]
```

2.2. ModelParameter

Request

```
GET /csip/m/weps/1.1 HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "metainfo": {},
  "parameter": [
      {
        "name": "management",
        "value": "<management description in LMOD JSON format>"
      },
      {
        "name": "soil",
        "value": "<mukey>:<cokey>",
        "description": "composite soil key returned from CSIP soil service"
      },
      {
        "name": "database",
        "value": "r2/ecat/ecat-2010-06-25"
      },
      {
        "name": "field_shape",
        "value": "rectangle"
      },
      {
        "name": "field_length",
        "value": 2640,
        "unit": "foot"
      },
      {
        "name": "field_width",
        "value": 2000,
        "unit": "foot"
      },
      {
        "name": "field_orientation",
        "value": 12,
        "unit": "degrees",
        "description": "orientation (rotation) of field from true north, 0-360"
      },

      {
        "name": "resolve_location",
        "value": true
      },
      {
        "name": "latitude",
        "value": 40.55924
      },
      {
        "name": "longitude",
```

```
        "value": -105.08275
      },
      {
        "name": "elevation",
        "value": 5028,
         "unit": "ft"
      },
      {
        "name": "barrier_north",
        "value": "cross wind trap strip",
        "description": "wind barrier type for specific edge of field"
      },
      {
        "name": "barrier_south",
        "value": null,
        "description": "wind barrier type for specific edge of field"
      },
      {
        "name": "barrier_east",
        "value": null,
        "description": "wind barrier type for specific edge of field"
      },
      {
        "name": "barrier_west",
        "value": null,
        "description": "wind barrier type for specific edge of field"
      },
      {
        "name": "water_erosion_loss",
        "value": 5.0,
        "unit": "ton/acre/year",
        "description": "water erosion computed externally, entered as an input for use by SCI and other calcu
      },
      {
        "name": "soil_rock_fragments",
        "value": 0,
        "unit": "percent",
        "description": "percent cover of surface due to rock fragments"
      }
    ]
  }
}
```

2.3. ModelExecution

Request

```
POST /csip/m/weps/1.1 HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
Content-Type:application/json
Content-Length: <length>

{
    "metainfo": {
    },

    "parameter": [
        //.. see ModelParameter response
    ],

}
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "metainfo": {
    },

    "parameter": [
        //.. see ModelParameter response
    ],
    "result": [
        {
            "name": "soil_conditioning_index",
            "value": 0.171,
            "description": "total SCI (soil conditioning index)"
        },
        {
            "name": "sci_om_factor",
            "value": -0.3817,
            "description": "SCI Organic Matter subfactor"
        },
        {
            "name": "sci_er_factor",
            "value": 0.9782,
            "description": "SCI Erosion Rate subfactor"
        },
        {
            "name": "sci_fo_factor",
            "value": 0.3195,
            "description": "SCI Field Operations subfactor"
        },

        {
            "name": "diesel_energy",
            "value": 35.358,
            "unit": "liter(Diesel)/hectare",
            "description": "energy used by management operations, expressed as fuel consumption equivalent"
        },

        {
            "name": "average_biomass",
            "value": 0.2391,
            "unit": "kilogram/meter^2",
            "description": "Average biomass"
        },
        {
            "name": "wind_eros",
            "value": 0.0124,
            "unit": "kilogram/meter^2/year",
            "description": "Wind erosion (computed by WEPS)"
        },
        {
            "name": "water_eros",
            "value": 0.0,
            "unit": "kilogram/meter^2/year",
            "description": "Water erosion (external input)"
        },
        {
            "name": "avg_all_stir",
            "value": 68.7258,
```

```
            "description": "Average STIR"
        }

    ]

}
```

3. STIR

The NRCS STIR (Soil Tillage Intensity Rating) value was developed by the USDA-NRCS to measure soil disturbance by management operations. This measure replaces the older SDR (Soil Disturbance Rating), which was subjective in nature. The STIR value is an input to the SCI (Soil Conditioning Index), in addition to being important on its own. Although this calculation is done in RUSLE2, these calculations depend only on the management operations and don't require erosion calculation inputs or outputs.

3.1. ModelCatalog

Request

```
GET /csip/m HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

"services": [
    {
        "name": "STIR",
        "version": "1.1",
        "description": "Soil Tillage Intensity Rating",
        "path": "stir/1.1"
    },
    {
        ...
```

3.2. ModelParameter

Request

```
GET /csip/m/stir/1.1 HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "metainfo" : { },
  "parameter" : [
      {
        "name" : "management",
```

```
        "value" : "<management description in LMOD JSON format>"
      },
      {
        "name" : "end_start_set_choice",
        "description" : "choices: 'user' (user-selected crop intervals), 'kill' (kill operation ends interval
        "value" : "user"
      }
    ]
}
```

3.3. ModelExecution

Request

```
POST /csip/m/stir/1.1 HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
Content-Type:application/json
Content-Length: <length>

{
    "metainfo": {
    },

    "parameter": [
        //.. see ModelParameter response
    ],
}
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "metainfo": {
        "status": "Finished"
    },
    "parameter": [
        {
            "name": "management",
            "value": "<management description in LMOD JSON format>"
        }
    ],
    "result": [
        {
            "name": "man_duration",
            "value": 1,
            "unit": "year"
        },
        {
            "name": "annual_stir",
            "value": 7.3
        },
        {
            "name": "rotation_stir",
            "value": 7.3
        },
        {
            "name": "op_stir_table",
```

```
            "value": [
                {
                    "op-name": "Kill crop",
                    "op-stir": 0,
                    "op-url": "http://oms-db.engr.colostate.edu/r2/operations/Kill%20crop.xml"
                },
                {
                    "op-name": "Drill or airseeder, double disk, w/ fluted coulters",
                    "op-stir": 7.15,
                    "op-url": "http://oms-db.engr.colostate.edu/r2/operations/Drill%20or%20airseeder,
%20double%20disk,%20w_%20fluted%20coulters.xml"
                },
                {
                    "op-name": "Harvest, killing crop 50pct standing stubble",
                    "op-stir": 0.15,
                    "op-url": "http://oms-db.engr.colostate.edu/r2/operations/Harvest,
%20killing%20crop%2050pct%20standing%20stubble.xml"
                }
            ]
        }
    ]
}
```

4. SCI

The NRCS SCI value (Soil Conditioning Index) was developed by the NRCS to measure qualitative changes to the soil health by management operations. The SCI calculations were added to RUSLE2 version 1.9.15 (2003-07-24). In general, a positive SCI value indicates increasing soil organic matter, and a negative value indicates decreasing organic matter. The SCI value is required to be calculated by RUSLE2 by NRCS policy since around 2005, and it depends on the STIR (Soil Tillage Intensity Rating) value.

The SCI and its background are described in detail in the USDA National Agronomy Manual, Section 508.

4.1. ModelCatalog

Request

```
GET /csip/m HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

"services": [
    {
        "name": "SCI",
        "version": "1.1",
        "description": "Soil Conditioning Index",
        "path": "sci/1.1"
    },
    {
        <other services>
    }
]
```

4.2. ModelParameter

Request

```
GET /csip/m/sci/1.1 HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "metainfo" : {},
  "parameter" : [
      {
        //.. Union of parameters from RUSLE2 and WEPS services.
      },
    ],
}
```

4.3. ModelExecution

Request

```
POST /csip/m/sci/1.1 HTTP/1.1
Host: csip.engr.colostate.edu:8083
Accept: application/json
Content-Type:application/json
Content-Length: <length>

{
    "metainfo": {
    },

    "parameter": [
        //.. see ModelParameter response
    ],
}
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "metainfo" : {
        "status" : "Finished",
    },
  "parameter" : [
        //.. Parameters from original model execution request
    ],
  "result" : [
      {
         "name" : "sci_value",
         "description" :"The total SCI value.",
         "value": -0.171
      },{
```

```
         "name": "sci_om",
         "description" :"The SCI OM (Organic Matter) subfactor.
                         This is 40% of the total SCI value.",
         "value": -0.3817
      },{
         "name": "sci_fo",
         "description" :"The SCI FO (Field Operations) subfactor.
                         This is 40% of the total SCI value.",
         "value": 0.9782
      },{
         "name": "sci_er",
         "description" :"The SCI ER (Erosion Rate) subfactor.
                         This is 20% of the total SCI value.",
         "value": 0.3195
      }
    ]
}
```

Chapter 4. Examples

The ModelServices API is callable from various clients such as CURL, Java, or RestClient browser plugin. CURL is a command line tool for HTTP requests, Java can be used with several libraries to perform Restful HTTP service calls, and RestClient exists for several browsers as a UI plugin.

CURL

For simple service testing, CURL is a useful tool to submit `http` request and receive the response. CURL is a simple command line tool for Linux and Windows. A typical usage of CURL for CSIP is shown below.

The example below fetches the model parameter of the Rusle2 model service as JSON string using an HTTP GET (default curl operation)

```
curl -H "Accept: application/json"
    "http://csip.engr.colostate.edu:8083/csip/m/rusle2/1.1"
```

The following example executes a the SCI model version 1.1 by sending a HTTP POST request and submitting the request data. The header specifies the content mime type as "`application/json`" and the JSON request data is obtained from the file "`sci.json`".

```
curl -X POST -H "Content-Type: application/json"
     "http://csip.engr.colostate.edu:8083/csip/m/sci/1.1" -d @sci.json
```

Java

```
import com.sun.jersey.api.client.*;
...
Client client = Client.create(new DefaultClientConfig());
WebResource service = client.resource(UriBuilder.fromUri().build(
          "http://csip.engr.colostate.edu/csip/m/sci/1.1"));

String response = service.accept(MediaType.WILDCARD).get(String.class);
```

1. Best Practices

It is recommended for the client to follow some rules:

- In asynchronous execution mode, especially if the model execution is expected to take a significant time (>2 sec), the returned `metadata` property "`polling_intervall`" and "`first_poll`" should be respected by the client. A client should start polling the first time after "`first_poll`" milliseconds, and then in subsequent "`polling_interval`" milliseconds. There is no advantage for a client to use a shorter polling interval.
- All timestamps are given in ISO-8601 format with timezone. In Java the pattern "`yyyy-MM-dd'T'HH:mm:ssZ`" can be used to parse such information into a '`java.util.Date`' object and further process it into `Calendar` or other representations.
- For all ModelExecution services, the JSON payload may contain other JSON objects next to `metainfo` and `parameter`. The service will retain those entries and will send them back untouched as a part of the response. Such behavior does not apply for ModelCatalog services and ModelParameter services. Therefore, a client may use the service payload to carry additional (state) information with the service, such as user interface controls or database data.

Bibliography

[Argent2004] Argent, R.M. An overview of model integration for environmental applications—components, frameworks and semantics Environmental Modelling & Software, Volume 19, Issue 3, Pages 219-234, Mar 2004

[David2002] David O., S.L. Markstrom, K.W. Rojas, L.R. Ahuja and I.W. Schneider, The Object Modeling System. In: L.R. Ahuja, L. Ma and T.A. Howell, Editors, Agricultural System Models in Field Research and Technology Transfer, Lewis Publishers, Boca Raton (2002), pp. 317–330.

[RFC4122] RFC 4122: A Universally Unique IDentifier (UUID) URN Namespace, http://www.ietf.org/rfc/rfc4122.txt

[ISO8601] ISO 8601:2004, Data elements and interchange formats -- Information interchange -- Representation of dates and times. http://www.iso.org/iso/catalogue_detail?csnumber=40874

GeoServices REST Specification Version 1.0, Esri White Paper,380 New York Street Redlands, California 92373-8100, September 2010

Open Geospatial Consortium, Inc. OpenGIS® Web Processing Service (WPS) Specification. WWW document, http://portal.opengeospatial.org/files/?artifact_id=24151, 2007. s