climagraph Documentation

Release 0.1.2

aert

January 21, 2014

Contents

Climatic data plotting.

Notable Features:

- Import data from CSV
- Data consolidation
- Charting

CHAPTER 1

Quick Start

1.1 Requirements

1.1.1 Dependencies

- Debian 7 (or Ubuntu >= 12.04)
- Python 2.7
- Vagrant *(Optional, to test staging env)*

1.2 Installation

1.2.1 Development

1. Install requirements:

   ```
   $ make develop
   ```

2. Run development web server:

   ```
   $ make runserver
   ```

3. Test in web browser:

   ```
   http://localhost:9000
   ```

1.2.2 Vagrant

1. Start vagrant and provision:

   ```
   $ make vagrant
   ```

2. Test in web browser:

   ```
   http://<vagrant_ip>
   ```

1.2.3 Release

1. Install requirements:

   ```
   $ pip install climagraph
   ```

2. Generate default conf:

   ```
   $ climagraph init
   ```

3. Tweak conf file in `~/.climagraph/config.conf`

4. Init database, assets and run gunicorn server:

   ```
   $ climagraph syncdb
   $ climagraph collectstatic
   $ climagraph start
   ```

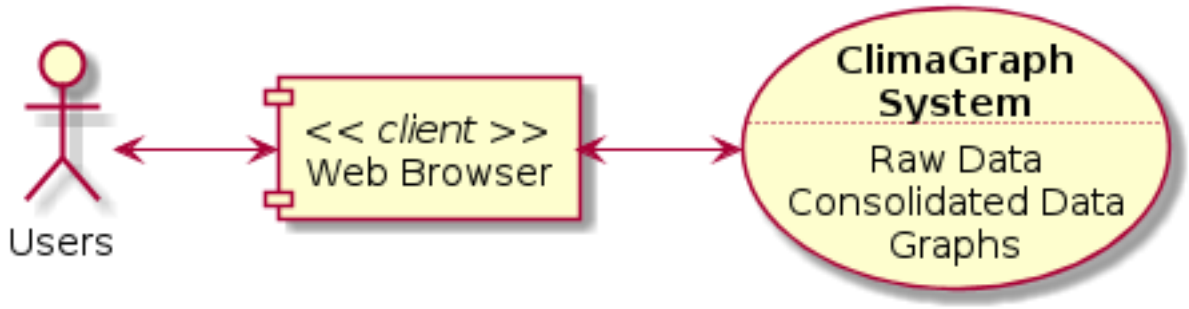
CHAPTER 2

Features

2.1 Overview

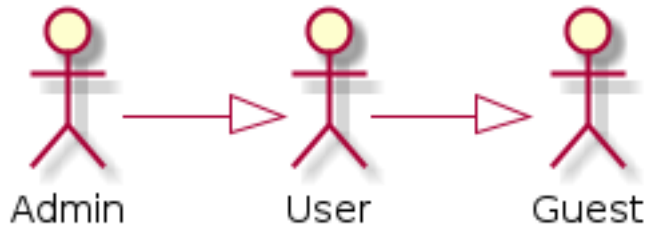
`climagraph` is a toolbox for climatic data plotting.

2.1.1 Context



2.1.2 Global Use Cases

Stakeholders

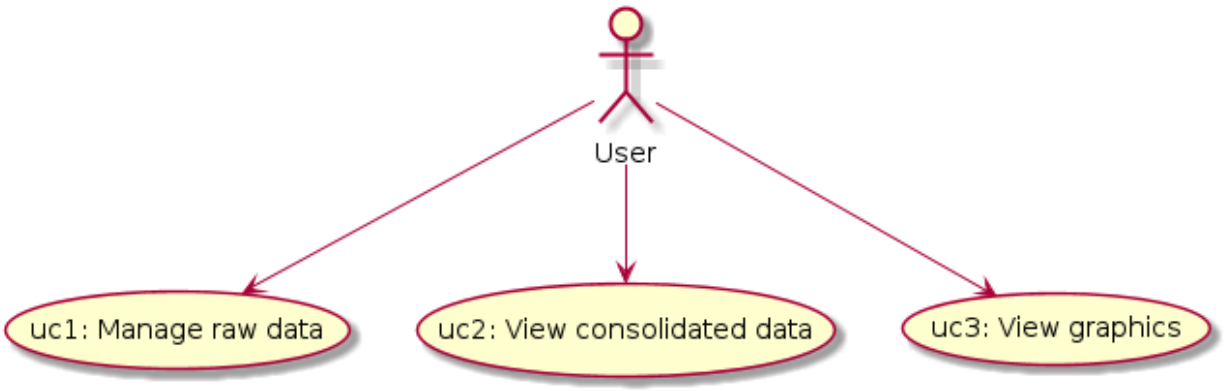


User Authenticated users.

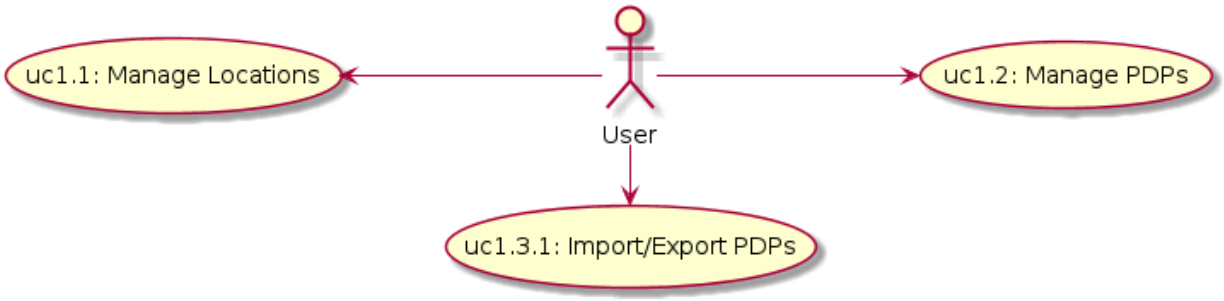
Guest Unauthenticated users.

Admin User with administrative rights.

Global Use Cases



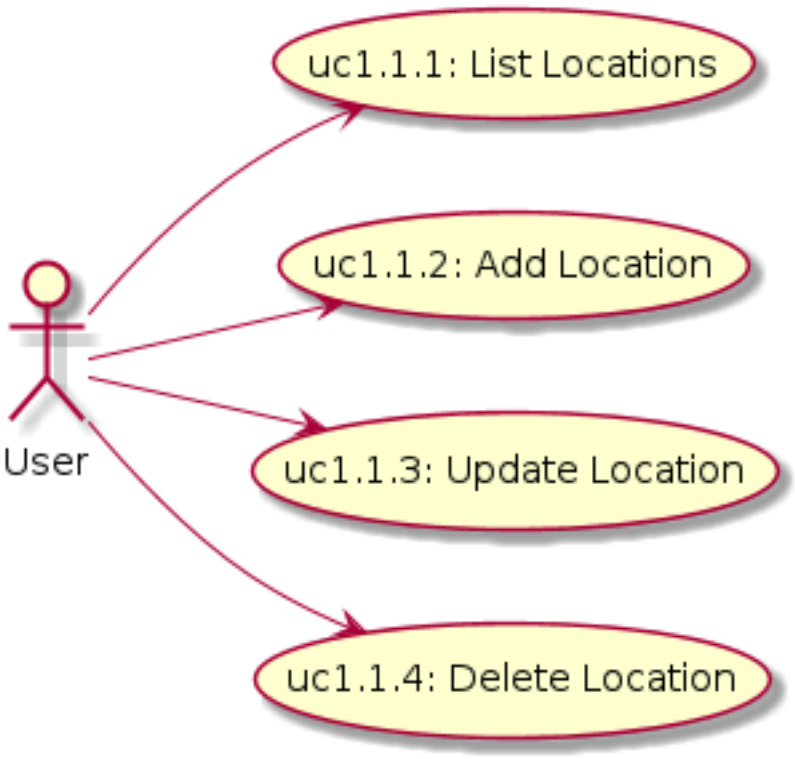
uc1: Manage raw data



Raw data points are called `Primary DataPoints` (**PDP**).

`Consolidated DataPoints` (**CDP**) are purely computed values resulting from the aggregation of `PDPs`.

Converting `PDPs` to `CDPs` is a core feature of the system. These `CDPs` are used in the plotting system.



uc1.1: Manage Locations

uc1.1.1: List Locations

Actors

1. User

Preconditions

1. User is authenticated

Post-conditions

1. Locations are listed

Normal flow

1. The user launches this use case.
2. The system displays the locations list. Each row contains :

   Data:

   Name Location name, `rule 1 - char(100)`

   Description Description - `char(500)`

   Latitude Geospacial latitude - `double`

   See Google API

   Longitude Geospacial longitude - `double`

Geometry List of geospacial points delimiting an area - `text`

Parents List of parent locations, `rule 2` - `foreign keys`

Business Rules

rule 1 `Name` must be unique.

rule 2 A location can be included in many other locations.

rule 3 Each modification on the `Parents` field invalidates all `CDP`.

These consolidates datapoints must then be recomputed before display.

uc1.1.2: Add Location

Actors

1. User

Preconditions

1. User is authenticated.

Post-conditions

1. New location is added with correct informations.
2. All `CDP` are invalidated.

Normal flow

1. The user launches this use case.
2. The system displays the following input form:

   Data:

   Name Location name, `rule 1` - `char(100)`.

   Description Description - `char(500)`.

   Latitude Geospacial latitude - `double`.

   Longitude Geospacial longitude - `double`.

   Geometry List of geospacial points delimiting an area - `text`.

   Parents List of parent locations, `rule 2` - `foreign keys`.

   Actions:

   Submit Creates the new location.
3. The user launches `Submit` action.
4. The system creates the new location.
5. The system invalidates all `CDP`.

Alternate flows 4a. Input data is invalid :

4.a.1. The system goes back to `step 2` adding error informations.

Business Rules See *Business Rules*.

uc1.1.3: Update Location

Actors

1. User

Preconditions

1. User is authenticated.
2. An existing location has been selected.

Post-conditions

1. Selected location's data is updated accordingly to modifications done.
2. All `CDP` are invalidated if necessary.

Normal flow

1. The user launches this use case.
2. The system displays the following **prepopulated** input form:

   Data:

   Name Location name, `rule 1` - `char(100)`.

   Description Description - `char(500)`.

   Latitude Geospacial latitude - `double`.

   Longitude Geospacial longitude - `double`.

   Geometry List of geospacial points delimiting an area - `text`.

   Parents List of parent locations, `rule 2` - `foreign keys`.

   Actions:

   Update Updates selected location.
3. The user launches `Update` action.
4. The system updates selected location.
5. The system invalidates all `CDP` if `Parents` fields has been modified.

Alternate flows 4a. Input data is invalid :

4.a.1. The system goes back to `step 2` adding error informations.

Business Rules See *Business Rules*.

uc1.1.4: Delete Location

Actors

1. User

Preconditions

1. User is authenticated.
2. An existing location has been selected.
3. Selected location must have no child location.
4. Selected location must have no `PDP` attached.

Post-conditions

1. Selected location is removed from database.
2. All `CDP` are invalidated.

Normal flow

1. The user launches this use case.
2. The system displays the following confirmation message:

   Do you really want to remove this location : **<Name>** ?

   Data:

   Name Location name.

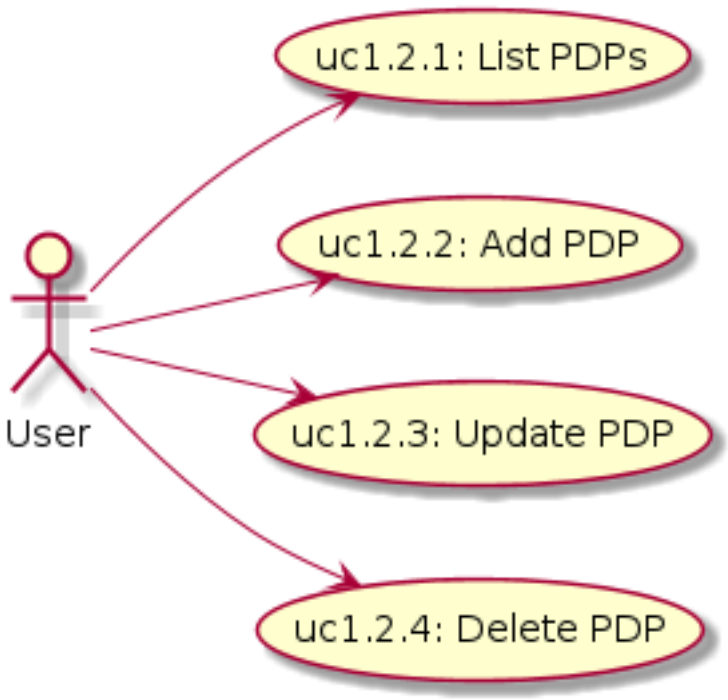
   Actions:

   Delete Deletes selected location.
3. The user launches `Delete` action.
4. The system deletes all `PDP` linked to current location.
5. The system deletes selected location.
6. The system invalidates all `CDP`.

Alternate flows 4a. Input data is invalid :

4.a.1. The system goes back to `step 2` adding error informations.

Business Rules See *Business Rules*.



uc1.2: Manage primary data points

uc1.2.1: List PDPs

Actors

1. User

Preconditions

1. User is authenticated
2. There is at least one `Location` in database.

Post-conditions

1. PDPs are listed
2. Search criterias are applied to the results

Normal flow

1. The user sets his search criterias and submits the form:

   Search Criterias:

   Location Geographical area

   DataPoint Type Kind of data, *ex.: Wind, Rain, Rice Production, etc.*

   Date Span Start date & end date

   Actions:

   Search Submits the input form

2. The system displays the Primary DataPoints list according to previous search criterias. Each row contains :

   Data:

   Date Sampled Date of the sample - `date`

   Timeslot Timeslot for the sampling in seconds - `long`

   Location *Exact* location of the sample, see `rule 1` - `label`

   Value measurement reported - `double, nullable`

Business Rules

rule 1 Locations displayed in result list are exact locations of the sampled data.

They can be sub locations included in the area of the search criteria.

rule 2 When one of the `PDPs` field is changed, all `CDPs` are invalided.

uc1.2.2: Add PDP

Actors

1. User

Preconditions

1. User is authenticated.

Post-conditions

1. New `PDP` is added with correct informations.
2. All `CDP` are invalidated.

Normal flow

1. The user launches this use case.
2. The system displays the following input form:

   Data:

   DataPoint Type Kind of data, *ex.: Wind, Rain, Rice Production, etc.* - `foreign key`

   Location Location of the sample, `rule 1` - `label`

   Date Sampled Date of the sample - `date`

   Timeslot Timeslot for the sampling in seconds - `long`

   Value measurement reported - `double, nullable`

   Actions:

   Add Creates the new `PDP`.
3. The user launches `Add` action.
4. The system creates the new `PDP`.
5. The system invalidates all `CDP`.

Alternate flows 4a. Input data is invalid :

4.a.1. The system goes back to `step 2` adding error informations.

Business Rules See *Business Rules*.

rule 1 The `Location` field should be the most precise available one.

uc1.2.3: Update PDP

Actors

1. User

Preconditions

1. User is authenticated.
2. An existing `PDP` has been selected.

Post-conditions

1. Selected `PDP`'s data is updated accordingly to modifications done.
2. All `CDPs` are invalidated if necessary.

Normal flow

1. The user launches this use case.
2. The system displays the following prepopulated non modifiable data:

   DataPoint Type Kind of data - `foreign key`

   Location Location of the sample - `label`

   Date Sampled Date of the sample - `date`

   Timeslot Timeslot for the sampling in seconds - `long`
3. The system displays the following **prepopulated** input form:

   Data:

   Value measurement reported - `double, nullable`

   Actions:

   Update Updates selected location.
4. The user launches `Update` action.
5. The system updates selected `PDP`.
6. The system invalidates all `CDPs`.

Alternate flows 4a. Input data is invalid :

4.a.1. The system goes back to `step 2` adding error informations.

Business Rules See *Business Rules*.

uc1.2.4: Delete PDP

Actors

1. User

Preconditions

1. User is authenticated.
2. An existing `PDP` has been selected.

Post-conditions

1. Selected `PDP` is removed from database.
2. All `CDP` are invalidated.

Normal flow

1. The user launches this use case.
2. The system displays the following confirmation message:

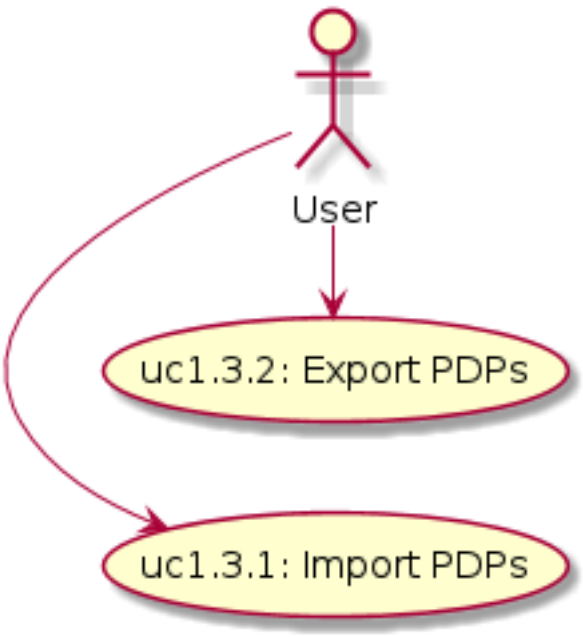
   Do you really want to remove this location Data Point ?

   Delete Deletes selected location.
3. The user launches `Delete` action.
4. The system deletes selected `PDP`.
5. The system invalidates all `CDPs`.

Alternate flows 4a. Input data is invalid :

4.a.1. The system goes back to `step 2` adding error informations.

Business Rules See *Business Rules*.



uc1.3: Import / Export

uc1.3.1: Import PDPs

DRAFT.

This use case allows primary data points to be batch added to the system.

Actors

1. User

Preconditions

1. User is authenticated.

Post-conditions

1. All `PDPs` are *added* with correct informations *if all data are valid*.
2. All `PDPs` are *rejected* if `1.` is not satisfied.
3. All `CDP` are invalidated.
4. A summary of success / failures is displayed.

Normal flow

DRAFT.

1. The user launches this use case.
2. The system displays the following input form:

   Data:

   File The **CSV** file to be uploaded - `file`

Actions:

Step 1/2 - Upload File Starts import operations.

3. The user launches `Import` action.
4. The system saves CSV data to the server.
5. The system validates `PDPs` data contained in the file and shows a summary of the data to be integrated.
6. The system asks for action :

   Actions:

   Step 2/2 - Add Data Points Integrates uploaded data to the system's database.
7. The system saves the `PDPs` in temporary databases.
8. The system checks data integrity according to existing `PDPs`, see `rule 2`.
9. The system creates final `PDPs` and removes temporary ones.
10. The system invalidates all `CDPs`.

Alternate flows

DRAFT.

5a. Error validating CSV data:

5a.1. The system removes the upload file. 5a.2. The system displays a summary of the error 5a.3. *End of the use case*

9a. Error creating Final `PDPs`:

9a.1. The system rolls back changed data: all succeeded `PDPs` are deleted. 9a.2. The system displays a summary of the error 9a.3. *End of the use case*

Business Rules

rule 1 Only **CSV** files are allowed.

rule 2 Imported `PDPs` must not be already existing in database, i.e. primary keys must be different.

uc1.3.2: Export PDPs

DRAFT.

This use case allows the user to backup `PDPs` and `CDPs`.

Actors

1. User

Preconditions

1. User is authenticated.

Post-conditions

DRAFT.

Normal flow

DRAFT.

Alternate flows

DRAFT.

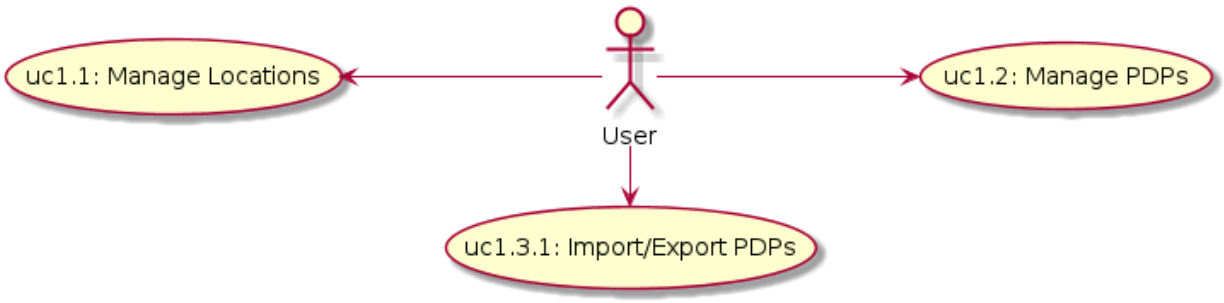
Business Rules

DRAFT.

uc2: View consolidated data

uc3: View graphics

2.2 uc1: Manage raw data

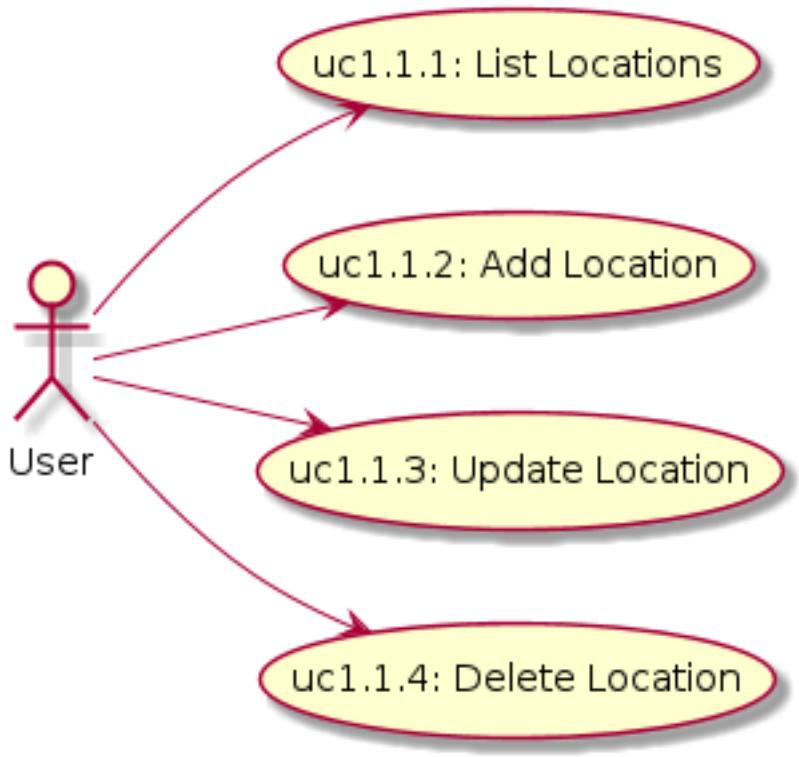


Raw data points are called `Primary DataPoints` (**PDP**).

`Consolidated DataPoints` (**CDP**) are purely computed values resulting from the aggregation of `PDPs`.

Converting `PDPs` to `CDPs` is a core feature of the system. These `CDPs` are used in the plotting system.

2.2.1 uc1.1: Manage Locations



uc1.1.1: List Locations

Actors

1. User

Preconditions

1. User is authenticated

Post-conditions

1. Locations are listed

Normal flow

1. The user launches this use case.
2. The system displays the locations list. Each row contains :

   Data:

   Name Location name, `rule 1 - char(100)`

Description Description - `char(500)`

Latitude Geospacial latitude - `double`

See Google API

Longitude Geospacial longitude - `double`

Geometry List of geospacial points delimiting an area - `text`

Parents List of parent locations, `rule 2` - `foreign keys`

Business Rules

rule 1 `Name` must be unique.

rule 2 A location can be included in many other locations.

rule 3 Each modification on the `Parents` field invalidates all `CDP`.

These consolidates datapoints must then be recomputed before display.

uc1.1.2: Add Location

Actors

1. User

Preconditions

1. User is authenticated.

Post-conditions

1. New location is added with correct informations.
2. All `CDP` are invalidated.

Normal flow

1. The user launches this use case.
2. The system displays the following input form:

   Data:

   Name Location name, `rule 1` - `char(100)`.

   Description Description - `char(500)`.

   Latitude Geospacial latitude - `double`.

   Longitude Geospacial longitude - `double`.

   Geometry List of geospacial points delimiting an area - `text`.

   Parents List of parent locations, `rule 2` - `foreign keys`.

Actions:

Submit Creates the new location.

3. The user launches `Submit` action.
4. The system creates the new location.
5. The system invalidates all `CDP`.

Alternate flows

4a. Input data is invalid :

4.a.1. The system goes back to `step 2` adding error informations.

Business Rules

See *Business Rules*.

uc1.1.3: Update Location

Actors

1. User

Preconditions

1. User is authenticated.
2. An existing location has been selected.

Post-conditions

1. Selected location's data is updated accordingly to modifications done.
2. All `CDP` are invalidated if necessary.

Normal flow

1. The user launches this use case.
2. The system displays the following **prepopulated** input form:

Data:

Name Location name, `rule 1` - `char(100)`.

Description Description - `char(500)`.

Latitude Geospacial latitude - `double`.

Longitude Geospacial longitude - `double`.

Geometry List of geospacial points delimiting an area - `text`.

Parents List of parent locations, `rule 2 - foreign keys`.

Actions:

Update Updates selected location.

3. The user launches `Update` action.
4. The system updates selected location.
5. The system invalidates all `CDP` if `Parents` fields has been modified.

Alternate flows

4a. Input data is invalid :

4.a.1. The system goes back to `step 2` adding error informations.

Business Rules

See *Business Rules*.

uc1.1.4: Delete Location

Actors

1. User

Preconditions

1. User is authenticated.
2. An existing location has been selected.
3. Selected location must have no child location.
4. Selected location must have no `PDP` attached.

Post-conditions

1. Selected location is removed from database.
2. All `CDP` are invalidated.

Normal flow

1. The user launches this use case.
2. The system displays the following confirmation message:

Do you really want to remove this location : **<Name>** ?

Data:

Name Location name.

Actions:

Delete Deletes selected location.

3. The user launches `Delete` action.
4. The system deletes all `PDP` linked to current location.
5. The system deletes selected location.
6. The system invalidates all `CDP`.

Alternate flows

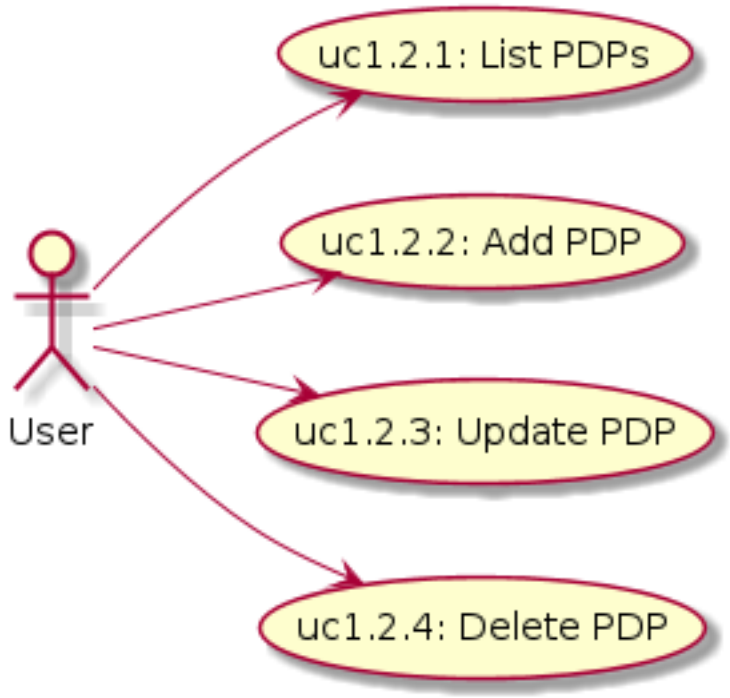
4a. Input data is invalid :

4.a.1. The system goes back to `step 2` adding error informations.

Business Rules

See *Business Rules*.

2.2.2 uc1.2: Manage primary data points



uc1.2.1: List PDPs

Actors

1. User

Preconditions

1. User is authenticated
2. There is at least one `Location` in database.

Post-conditions

1. PDPs are listed
2. Search criterias are applied to the results

Normal flow

1. The user sets his search criterias and submits the form:

   Search Criterias:

   Location Geographical area

   DataPoint Type Kind of data, *ex.: Wind, Rain, Rice Production, etc.*

   Date Span Start date & end date

   Actions:

   Search Submits the input form

2. The system displays the Primary DataPoints list according to previous search criterias. Each row contains :

   Data:

   Date Sampled Date of the sample - `date`

   Timeslot Timeslot for the sampling in seconds - `long`

   Location *Exact* location of the sample, see `rule 1` - `label`

   Value measurement reported - `double, nullable`

Business Rules

rule 1 Locations displayed in result list are exact locations of the sampled data.

They can be sub locations included in the area of the search criteria.

rule 2 When one of the `PDPs` field is changed, all `CDPs` are invalided.

uc1.2.2: Add PDP

Actors

1. User

Preconditions

1. User is authenticated.

Post-conditions

1. New `PDP` is added with correct informations.
2. All `CDP` are invalidated.

Normal flow

1. The user launches this use case.
2. The system displays the following input form:

   Data:

   DataPoint Type Kind of data, *ex.: Wind, Rain, Rice Production, etc.* - `foreign key`

   Location Location of the sample, `rule 1` - `label`

   Date Sampled Date of the sample - `date`

   Timeslot Timeslot for the sampling in seconds - `long`

   Value measurement reported - `double, nullable`

   Actions:

   Add Creates the new `PDP`.
3. The user launches `Add` action.
4. The system creates the new `PDP`.
5. The system invalidates all `CDP`.

Alternate flows

4a. Input data is invalid :

4.a.1. The system goes back to `step 2` adding error informations.

Business Rules

See *Business Rules*.

rule 1 The `Location` field should be the most precise available one.

uc1.2.3: Update PDP

Actors

1. User

Preconditions

1. User is authenticated.
2. An existing `PDP` has been selected.

Post-conditions

1. Selected `PDP`'s data is updated accordingly to modifications done.
2. All `CDPs` are invalidated if necessary.

Normal flow

1. The user launches this use case.
2. The system displays the following prepopulated non modifiable data:

   DataPoint Type Kind of data - `foreign key`

   Location Location of the sample - `label`

   Date Sampled Date of the sample - `date`

   Timeslot Timeslot for the sampling in seconds - `long`
3. The system displays the following **prepopulated** input form:

   Data:

   Value measurement reported - `double, nullable`

   Actions:

   Update Updates selected location.
4. The user launches `Update` action.
5. The system updates selected `PDP`.
6. The system invalidates all `CDPs`.

Alternate flows

4a. Input data is invalid :

4.a.1. The system goes back to `step 2` adding error informations.

Business Rules

See *Business Rules*.

uc1.2.4: Delete PDP

Actors

1. User

Preconditions

1. User is authenticated.
2. An existing `PDP` has been selected.

Post-conditions

1. Selected `PDP` is removed from database.
2. All `CDP` are invalidated.

Normal flow

1. The user launches this use case.
2. The system displays the following confirmation message:

   Do you really want to remove this location Data Point ?

   Delete Deletes selected location.
3. The user launches `Delete` action.
4. The system deletes selected `PDP`.
5. The system invalidates all `CDPs`.

Alternate flows

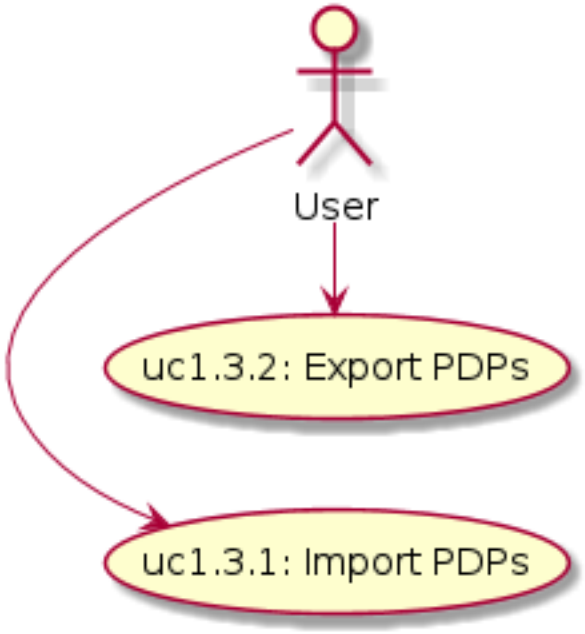
4a. Input data is invalid :

4.a.1. The system goes back to `step 2` adding error informations.

Business Rules

See *Business Rules*.

2.2.3 uc1.3: Import / Export



uc1.3.1: Import PDPs

DRAFT.

This use case allows primary data points to be batch added to the system.

Actors

1. User

Preconditions

1. User is authenticated.

Post-conditions

1. All `PDPs` are *added* with correct informations *if all data are valid*.
2. All `PDPs` are *rejected* if `1.` is not satisfied.
3. All `CDP` are invalidated.
4. A summary of success / failures is displayed.

Normal flow

DRAFT.

1. The user launches this use case.
2. The system displays the following input form:

   Data:

   File The **CSV** file to be uploaded - `file`

   Actions:

   Step 1/2 - Upload File Starts import operations.
3. The user launches `Import` action.
4. The system saves CSV data to the server.
5. The system validates `PDPs` data contained in the file and shows a summary of the data to be integrated.
6. The system asks for action :

   Actions:

   Step 2/2 - Add Data Points Integrates uploaded data to the system's database.
7. The system saves the `PDPs` in temporary databases.
8. The system checks data integrity according to existing `PDPs`, see `rule 2`.
9. The system creates final `PDPs` and removes temporary ones.
10. The system invalidates all `CDPs`.

Alternate flows

DRAFT.

5a. Error validating CSV data:

5a.1. The system removes the upload file. 5a.2. The system displays a summary of the error 5a.3. *End of the use case*

9a. Error creating Final `PDPs`:

9a.1. The system rolls back changed data: all succeeded `PDPs` are deleted. 9a.2. The system displays a summary of the error 9a.3. *End of the use case*

Business Rules

rule 1 Only **CSV** files are allowed.

rule 2 Imported `PDPs` must not be already existing in database, i.e. primary keys must be different.

uc1.3.2: Export PDPs

DRAFT.

This use case allows the user to backup `PDPs` and `CDPs`.

Actors

1. User

Preconditions

1. User is authenticated.

Post-conditions

DRAFT.

Normal flow

DRAFT.

Alternate flows

DRAFT.

Business Rules

DRAFT.

2.3 uc2: View consolidated data

2.4 uc3: View graphics

CHAPTER 3

Developers

3.1 Contributor Guide

Those who wish to contribute directly to the project can contact me at dev.aert@gmail.com to talk about getting repository access granted.

3.1.1 More Information

- GitHub : http://github.com/aert/aert-climagraph
- Documentation : http://aert-climagraph.readthedocs.org

3.1.2 License

This project is licensed under the MIT license.

3.1.3 Support

- Issue Tracking : https://github.com/aert/aert-climagraph/issues
- Pull Request : https://github.com/aert/aert-climagraph/pulls

3.2 Testing

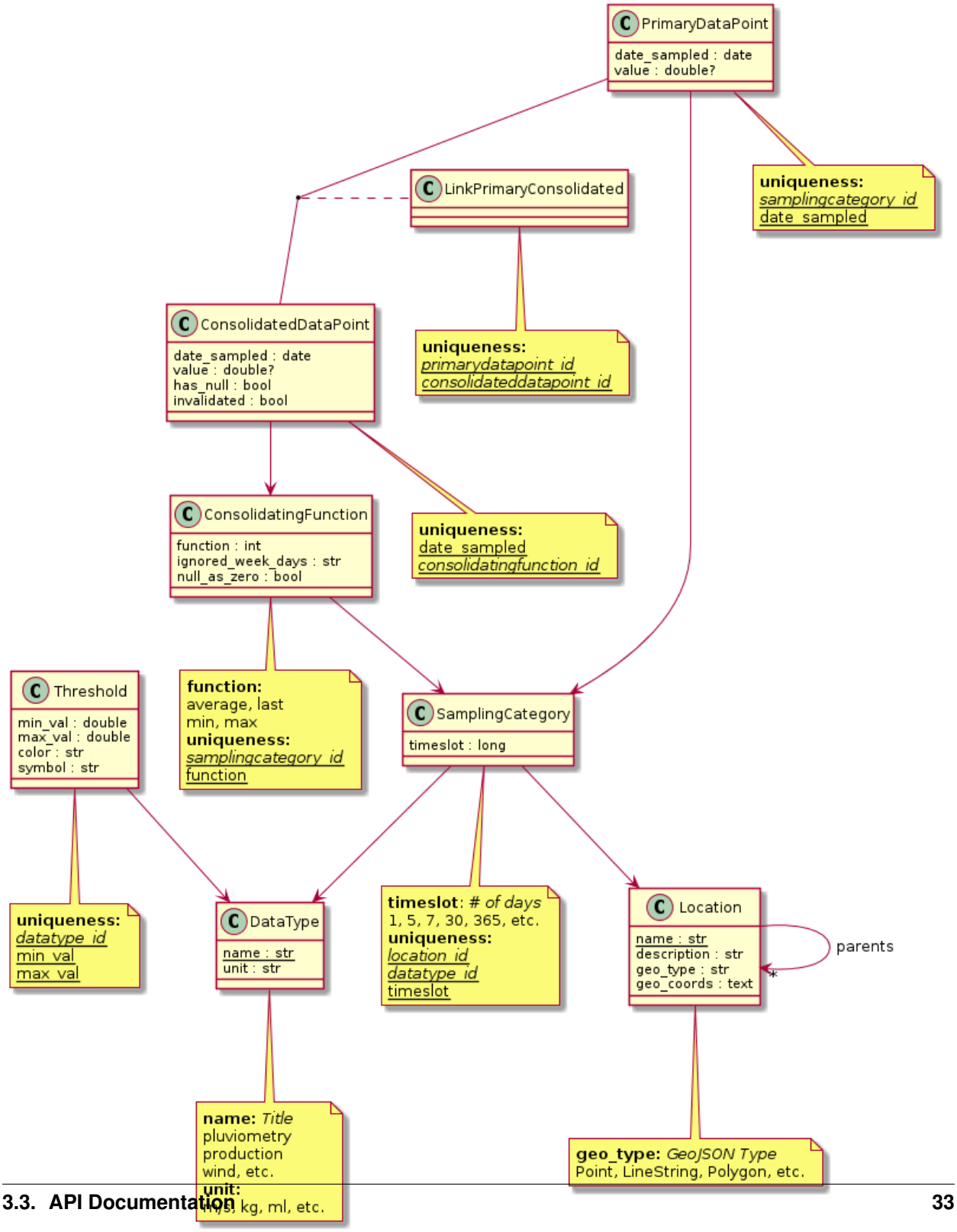
- Using Vagrant:

```
$ make vagrant_runtest
```

3.3 API Documentation

3.3.1 `climagraph.model`

Overview



CHAPTER 4

Reference

4.1 Changelog

4.1.1 Version 0.1.2

- Docs initialization `+docs @20140106`

CHAPTER 5

Screenshots