Weather History

Version 1.0

February 1, 2010

Document Control

Approval

The Guidance Team and the customer shall approve this document.

Document Change Control

|  |  |
| --- | --- |
| Initial Release: | Dec 15, 2009 |
| Current Release: | Feb 1, 2010 |
| Indicator of Last Page in Document: | ♦ |
| Date of Last Review: | Feb 1, 2010 |
| Date of Next Review: | TBD |
| Target Date for Next Update: |  |

Distribution List

This following list of people shall receive a copy of this document every time a new version of this document becomes available:

Guidance Team Members:

Dr. Steve Roach

Dr. Tanja Magoc

Ms. Evelyn Torres

Mr. Chris Cuellar

Customer: Dr. Craig Tweedie

Systems Engineering Laboratory

Department of Biology

The University of Texas at El Paso

Software Teams:

Tech Nebula

MK Ultra

Change Summary

The following table details changes made between versions of this document

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Modifier | Description |
| 0.1 | 12/15/09 | Roach | Use cases |
| 0.1 | 1/15/10 | Magoc | Sections 3, 4, and 5 |
| 0.2 | 1/24/10 | Roach | Revised 3, 4, and 5 |
| 0.3 | 1/26/09 | Cuellar | Diagrams |
| 0.9 | 2/1/10 | Roach | Sections 3, 4, 5 |
| 1.0 | 2/1/20 | Torres, Magoc | Sections 1, 2 |

Table of Contents

[Document Control ii](#_Toc252828003)

[Approval ii](#_Toc252828004)

[Document Change Control ii](#_Toc252828005)

[Distribution List ii](#_Toc252828006)

[Change Summary ii](#_Toc252828007)

[1. Introduction 1](#_Toc252828008)

[1.1. Purpose and Scope of Product 1](#_Toc252828009)

[1.2. Intended Audience 1](#_Toc252828010)

[1.3. Overview 1](#_Toc252828011)

[1.4. Definitions, Acronyms, and Abbreviations 2](#_Toc252828012)

[1.4.1. Definitions 2](#_Toc252828013)

[1.4.2. Acronyms and Abbreviations 2](#_Toc252828014)

[1.5. References 3](#_Toc252828015)

[2. General Description 4](#_Toc252828016)

[2.1. Product Perspective 4](#_Toc252828017)

[2.2. User Characteristics and Actor Descriptions 4](#_Toc252828018)

[2.3. Product Features 4](#_Toc252828019)

[2.3.1. Use Case Description 5](#_Toc252828020)

[2.3.1.1. Use Case #1: Acquire Historical Weather Data 5](#_Toc252828021)

[2.3.1.2. Use Case #2: Acquire List of Weather Stations 6](#_Toc252828022)

[2.3.1.3. Use Case #3: View Activity Log 7](#_Toc252828023)

[2.4. Operating Environment 8](#_Toc252828024)

[2.5. General Constraints 8](#_Toc252828025)

[2.6. Assumptions and Dependencies 8](#_Toc252828026)

[3. External Interface Requirements 9](#_Toc252828027)

[3.1. User Interfaces 9](#_Toc252828028)

[3.2. Hardware Interfaces 9](#_Toc252828029)

[3.3. Software Interfaces 9](#_Toc252828030)

[3.4. Communications Interfaces 9](#_Toc252828031)

[4. Behavioral Requirements 10](#_Toc252828032)

[4.1. Same Class of User 10](#_Toc252828033)

[4.2. Related Real-world Objects 10](#_Toc252828034)

[4.2.1. Weather Station 10](#_Toc252828035)

[4.2.2. Weather Data 10](#_Toc252828036)

[4.2.3. Activity Log 10](#_Toc252828037)

[4.3. Related Features 10](#_Toc252828038)

[4.3.1. Create List of Weather Stations 10](#_Toc252828039)

[4.3.2. Weather Data Request 11](#_Toc252828040)

[4.4. Stimulus 11](#_Toc252828041)

[4.4.1. Acquiring a List of Weather Stations 11](#_Toc252828042)

[4.4.2. Acquiring Weather Data 11](#_Toc252828043)

[4.5. Functional 12](#_Toc252828044)

[5. Non-behavioral Requirements 13](#_Toc252828045)

[5.1. Performance Requirements 13](#_Toc252828046)

[5.2. Security 13](#_Toc252828047)

[5.3. Qualitative Requirements 13](#_Toc252828048)

[5.3.1. Availability 13](#_Toc252828049)

[5.3.2. Maintainability 13](#_Toc252828050)

[5.3.3. Portability 13](#_Toc252828051)

[5.3.4. Design and Implementation Constraints 13](#_Toc252828052)

[6. Other Requirements 14](#_Toc252828053)

[6.1. Database 14](#_Toc252828054)

[6.2. Operations 14](#_Toc252828055)

[6.3. Site Adaptation 14](#_Toc252828056)

[7. Appendix A: Static Model 15](#_Toc252828057)

[8. Appendix B: Dynamic Model 16](#_Toc252828058)

[9. Appendix C: Functional Model 17](#_Toc252828059)

[10. Appendix D: Database Schema 18](#_Toc252828060)

# Introduction

This section describes the purpose, intended audience, and overview of the document as well as the scope and intended use of the systems being developed.

## Purpose and Scope of Product

The purpose of the Software Requirements Specification (SRS) document is to provide a clear and precise description of the functionality of the Weather History (WH) system. The SRS will serve as a reference for the development teams during the design, implementation and verification phases; the SRS is also an agreement between the client and the development teams regarding the functionality the finished product will perform.

In recent years, the Earth has experienced drastic climate changes. It has become of great importance to understand and study these changes and their impact on the human race. Scientists all over the globe including at the Systems Ecology Lab at University of Texas at El Paso (UTEP) have put an enormous amount of effort into gathering and analyzing weather data. Currently, the UTEP research team utilizes the Circumarctic Environmental Observatories Network (CEON) web-based mapping and information system. CEON allows access to near real-time reports of earthquakes, climate data, and webcam images. The success of this powerful application has inspired interest in extending both the functionality of the system and the geographical scope to which it applies. Weather History system will serve as an extension to CEON which will allow users to access historical climate data from weather stations from across the North American continent.

To provide scientists with enough information to understand the changes in the climate, the WH system will provide means to access historical data from historical weather data sources such as National Oceanic and Atmospheric Administration (NOAA). The tool will search for historical weather data specified by a list of weather stations, types of weather data to be collected, and a time range supplied by an end-user. This tool will provide a means for environmental scientists, researchers, university professors, students, and the general public to have easy access to historical weather data for further analysis and therefore will improve the research community’s ability to understand and make inferences about certain phenomena regarding our climate system.

## Intended Audience

The intended audience of this document is the client, the Guidance Team, and the software development teams.

## Overview

The SRS is divided into six major sections: Introduction (Section 1), General Description (Section 2), External Interface Requirements (Section 3), Behavioral Requirements (Section 4), Non-behavioral Requirements (Section 5), and Other Requirements (Section 6). This overview describes Section 2 through Section 6 of the SRS.

Section 2 provides a general description of the system including its overall structure and functionality, users and actors of the systems, the operating environment in which the system will run, existing constraints on the system, and assumptions and dependencies.

Section 3 describes the specification of requirements for interfaces between the system and external components, both human and other systems. It contains specifications with respect to user, software, hardware, and communication interfaces.

Section 4 includes five subsections. It describes the behavioral requirements of the system. The requirements are organized in the following categories: same class of user, related real-world objects, stimulus, related features, and functional requirements.

Section 5 includes three subsections. It outlines the non-behavioral requirements of the system which consists of performance, security and qualitative requirements with respect to availability, maintainability, portability, and design and implementation constraints.

## Definitions, Acronyms, and Abbreviations

This section describes the definitions, acronyms and abbreviations that are useful for understanding the contents of this document.

### Definitions

|  |  |
| --- | --- |
| **TERM** | **DEFINITION** |
| Actor | An actor is any outside entity that interacts with WH system. |
| Adobe Flex | An open source framework for building and maintaining web applications. |
| Client Program | A web service client that requests data. Initially, we anticipate the client to be the descriptive statistics system for CEON. This actor provides the necessary input to initiate a request for historical weather data or a list of weather stations. |
| Historical Weather Data Source | An organization that provides historical weather data. An example of historical weather data source is NOAA. |
| Local Database | The database where WH system will be storing historical data. |
| PHP | A scripting language designed for producing dynamic web pages. |
| PostgreSQL | An object-relational database management system. |
| R | A language and environment for statistical computing and graphics. |
| Web service | A software system designed to support interoperable machine-to-machine interaction over a network. |

Table 1.1: Definitions

### Acronyms and Abbreviations

|  |  |
| --- | --- |
| **ACRONYM/ABBREVIATION** | **MEANING** |
| CEON | Circum-arctic Environmental Observatories Network |
| DFD | Data Flow Diagram |
| e.g. | For example |
| ER | Entity Relationship Diagram |
| i.e. | Such as |
| ID | Identification |
| NOAA | National Oceanic and Atmospheric Administration |
| OS | Operating System |
| SRS | Software Requirements Specification |
| STD | State Transition Diagram |
| TBD | To Be Determined |
| UTEP | The University of Texas at El Paso |
| XML | Extensible Markup Language |

Table 1.2 Acronyms

## References

1. Tweedie, Craig. First interview. 8 September 2009.
2. Tweedie, Craig. MK Ultra and Tech Nebula interview. 21 September 2009.
3. Tweedie, Craig. Guidance team interview. 15 January 2010.
4. Team MK Ultra, Team Tech Nebula, Team Secui Prorsus. SRS, December 2009.

# General Description

This section describes the system being developed with respect to the main features of its functionality as well as the intended user characteristics.

## Product Perspective

The Weather History tool is an extension to the CEON system currently being used in the UTEP Systems Ecology Lab. The CEON application provides near-real time access to environmental monitoring data streams in Arctic and has become an important tool for the study of climate change. Its success has created the desire to expand the application since CEON focuses only on the Arctic region of the globe. WH will enable this extension to allow scientists to monitor and analyze weather data from the North America region by providing access to historical weather data streams needed for statistical analysis and predictions of climate changes.

## User Characteristics and Actor Descriptions

The system will be available via the World Wide Web, and therefore, there will be a wide variety of users who have access to the system. The system is intended for scientists who are knowledgeable about statistics and climate data. The intended audience will have post-secondary education. We assume that these users are familiar with web interfaces. The audience may include occasional users, and thus, the system should guide users to the completion of given tasks.

An actor is an external entity that interacts with the system to achieve a valuable goal. An actor may represent a human, a device, or another software system. In this section, a description of the actors shown in the use case diagram in Figure 1 is provided.

**Client Program:** Client Program represents a web service client that requests data. Initially, we anticipate the client to be the descriptive statistics system for CEON. This actor provides the necessary input to initiate a request for historical weather data or a list of weather stations.

**Local Database:** Local Database stores historical weather data that was retrieved recently from the historical weather data source. The intent is to reduce the workload on the historical weather source by keeping data that we have acquired.

**Historical Weather Data Source:** This actor provides historical weather data to the system if the requested data are not stored in the local database. An example of historical weather data source is NOAA.

**Administrator:** The administrator will receive error logs in order to monitor the system.

## Product Features

The weather history tool will be vital to the analysis of climate data; it will provide access historical weather data necessary to perform statistical and trend analysis. In order for the tool to access these data, it must also keep knowledge of what weather stations are available for querying. The tool will also maintain a local database which is intended to reduce the number of requests to the historical weather data sources. When data is returned from sources such as NOAA it will be stored in the local database; in the event that another user requests the same data the tool can obtain the data from the local database. The system administrator will also have access to an activity log to monitor system activity and the completion/incompletion of tasks.



Figure 1. Use case diagram

### Use Case Description

A use case describes the interactions between the actors and the system. This subsection describes the use cases for this system.

#### Use Case #1: Acquire Historical Weather Data

**Use Case Description:** The service client requests weather data specified by parameters that include weather stations, instrument data types, and time range. The use case assumes that the parameters are provided in the correct format (e.g., the start time point is earlier than the end time point in the time range).

**Actors:** Client Program, Local Database, Historical Weather Data Source.

**Precondition:** The system is idle and waiting for a request from the Client Program.

**Post-condition:** On successful completion, the requested weather dataset is returned to the Client Program. If the system is unable to complete the request, the system logs an error message into the activity log.

**Trigger condition:** The Client Program makes a historical weather data request specifying weather station(s), data type(s), and time range.

**Steps:**

1. The system verifies that the request contains all necessary parameters, which include a non-empty weather stations list, a non-empty set of instrument data types for each weather station, and time range (Alt 1).
2. The system creates a database search query using parameters provided in the request and queries the local database for the requested data.
3. The local database returns the requested data (Alt 2, Alt 3).
4. The system processes data into a formatted file.
5. The system returns the formatted file to the Client Program.
6. End of use case.

**Alternate Flows**

**Alt. 1: The request does not contain all necessary parameters.**

* 1. The system returns an error message to the Client Program that some parameters are missing.
  2. End of use case.

**Alt 2: The system is not able to access the local database.**

2-1. The system logs an error in the activity log.

2-2. The use case continues at Alt 3.

**Alt. 3: The local database does not contain the requested data.**

* 1. Local database returns an empty search result.
  2. The system verifies that historical weather data source was not queried for the same weather data in the last hour (Alt 3.1).
  3. The system queries the historical weather data source for the requested data.
  4. The historical weather data source returns the requested data (Alt 3.2, Alt 3.3).
  5. The system parses the received data.
  6. The system stores parsed data into the local database (Alt 3.4).
  7. The use case continues at step 4.

**Alt. 3.1: The historical weather data source has been queried for the same weather data in the last hour.**

3.1-1. The system returns an error message to the Client Program stating that the requested data are not available.

**Alt. 3.2: The system is unable to establish a connection to the historical weather data source.**

3.2-1. The system logs an error in the activity log.

3.2-2. The system returns an error message to the Client Program.

3.2-3. End of use case.

**Alt. 3.3: The historical weather data source does not contain the requested data.**

3.3-1. The historical weather data source returns an empty search result.

3.3-2. The system returns a message to the Client Program describing that the requested data were not found.

3.3-3. End of use case.

**Alt. 3.4: The system is not able to access the local database.**

3.4-1. The system logs an error in the activity log.

3.4-2. The use case continues at step 4.

#### Use Case #2: Acquire List of Weather Stations

**Use Case Description:** The Client Program requests a list of weather stations that satisfy a set of provided parameters. The parameters could be a station’s ID, state, location, longitude, latitude, or a priority list.

**Actors:** Client Program, Historical Weather Data Source.

**Precondition:** The system is idle and waiting for a request from a Client Program.

**Post-condition:** On successful completion, the requested list of weather stations is returned to the Client Program. If the system is unable to complete the request, the system logs an error message into the activity log.

**Trigger condition:** The system receives a request for a list of weather stations that satisfy a set of provided parameters.

**Steps:**

1. The system verifies that the parameters for longitude, latitude, location, state, or ID are included in the request (Alt 3, Alt 4).
2. The system creates a database search query using parameters provided in the request and queries the historical weather data source for the requested data.
3. The historical weather data source returns the list of weather stations available that satisfy the requested parameters (Alt 1, Alt 2).
4. The system formats the received results into a list of weather stations.
5. The system returns to the Client Program the list of requested weather stations.
6. End of use case.

**Alternate Flows**

**Alt. 1: The system is not able to establish the connection to the historical weather data source.**

* 1. The system logs an error message in the activity log.
  2. The system returns an error message to the Client Program.
  3. End of use case.

**Alt. 2: The historical weather data source does not find any weather stations satisfying provided parameters.**

2-1. The historical weather data source returns an empty search result.

2-2. The system returns a message to the Client Program that there are no weather stations satisfying the provided parameters.

2-3. End of use case.

**Alt. 3: The request prompts for the priority list.**

3-1. The system request the priority list from the local database.

3-2. The local database returns the priority list.

3-3. The system returns the priority list to the Client Program.

3-4. End of use case.

**Alt. 4: The local database does not contain the priority list.**

4-1. The system returns an error message to the Client Program.

4-2. End of use case.

#### Use Case #3: View Activity Log

**Use Case Description:** The Activity Log is stored in the local database. It contains enough information for the Administrator to track the system activity, namely any error message generated by the system, times that processes are initiated, and whether processes complete their assigned tasks.

**Actors:** Administrator, Local Database.

**Precondition:** The local database is online.

**Post-condition:** On successful completion, the system will have displayed the Activity Log.

**Trigger:** The Administrator activates the Activity Log process.

**Steps:**

1. The system prompts the Administrator for a userid and password.
2. The Administrator enters the userid and password. The password is overwritten with “\*” characters as it is entered.
3. The system verifies the userid and password (Alt 1).
4. The system queries the Activity Log in the local database for all of the messages.
5. The Local Database returns all of the messages.
6. The system displays the messages and time stamps for the messages, ordered by time from most recent to oldest. The system displays 20 messages at a time as well as left and right arrow icons for scrolling through messages.
7. The Administrator reads the messages and selects the left arrow icon (Alt 2).
8. The system displays the 20 messages previous to the oldest message currently displayed.
9. The Administrator selects the Quit option.
10. End of use case.

**Alternate Flows**

**Alt. 1: The userid and password do not match the Administrator’s userid and password.**

* 1. The system displays an error message.
  2. Use case returns to step 1.

**Alt. 2: The user selects the right arrow icon.**

2-1. The system displays the 20 messages after to the most recent message currently displayed.

2-2. Use case continues at step 9.

## Operating Environment

* The user interface requires Macromedia Flash Player 10. It is assumed to be installed on the user’s machine.
* The server-side software for CEON runs on Microsoft Windows Server 2003. It is assumed that the new system will also run on this machine.
* The server-side software requires PHP, Adobe Flex, and R. It is assumed that all of these are installed.

## General Constraints

The development team will consist of an analyst, a designer, two or three programmers, and a V&V specialist.

## Assumptions and Dependencies

* It is assumed that users have web browser software and are familiar with basic use of web browsers.
* It is assumed that the response time will be fast enough. No response time or performance metrics have been given.
* Weather data is assumed to be stored in the weather database, which is accessible from the server.
* The user interface will be built using the shell provided by the client.

# External Interface Requirements

This section contains the specification of requirements for interfaces among different components and their external capabilities, including all its users.

## User Interfaces

1. The system shall use standard user interface controls such as buttons, text boxes, radio buttons, check boxes, labels, list boxes, spin boxes, combo boxes, sliders, scroll bars, tabs, tool tips, progress bars, and file selection dialogs [Galitz pp 443-552 et al].
2. The user interfaces shall be presented as web pages and shall be displayed by web browsers.
3. Each page of the system shall have full screen, minimize, and exit options.
4. Each page of the system shall allow a user to return to the previous page by selecting a “back” option.
5. The system shall provide a visual display such as a progress bar or hour-glass icon while any function is being performed.
6. The system shall provide an interface, hereafter called the *Administrator Login Page*, which displays the prompt for the Administrator to enter the userid and password.
7. The system shall provide an interface for viewing the Activity log, hereafter called the *Activity Log Interface*.
8. The *Activity Log Interface* shall display messages and time stamps for the messages, ordered from the most recent to oldest.
9. The *Activity Log Interface* shall display 20 messages at a time as well as left and right arrow icons for scrolling through messages.

## Hardware Interfaces

There are no hardware-specific requirements.

## Software Interfaces

1. The system shall use a PostgreSQL 8.0 or later database management system.

## Communications Interfaces

1. The system shall use web services to obtain data from historical weather data source.
2. The system shall provide an xml web service interface to acquire weather history.

# Behavioral Requirements

This section contains specific behavioral requirements for the system.

## Same Class of User

1. The server program shall have only one class of user, the Administrator, who has access to the Activity Log.
2. The system shall require the Administrator to enter a valid combination of userid and password in order to use the system.

## Related Real-world Objects

Real-world objects are entities with either physical or conceptual counterparts in the real world. The entity-relation diagram that motivates the real-world objects described in this section can be found in Appendix A.

### Weather Station

1. Each weather station shall have a unique ID and a unique location, which is specified by longitude and latitude.
2. Each weather station shall have a set of instruments that collect weather data.

### Weather Data

Weather data is recorded for a weather station at a given time.

1. The types of instruments possible for a weather station shall include those that measure temperature in C, relative humidity, wind speed in mph, wind direction, precipitation, weather condition (e.g., clear, thunderstorm), wind degree, barometric pressure in mb, dew point in C, heat index in C, windchill in C, visibility in km.
2. A weather data element shall be identified by a weather station, a date and time, and the data value for an instrument on the weather station at that time.

### Activity Log

1. The system shall keep an activity log that will list all of the following types of errors that occur within the system:

* The historical weather data source does not contain requested weather data.
* The system cannot establish a connection to the historical weather data source.

1. The activity log shall include the following information for each entry:

* The type of the error.
* The date and time when the error occurred.

## Related Features

A related feature is an externally desired service provided by the system that may require a sequence of inputs to affect the desirable results. Use cases that outline this section can be found in the section 2 of this SRS.

### Create List of Weather Stations

1. The system shall accept requests for a list of weather stations based on any of the following attributes:

* Station’s ID
* Station’s longitude
* Station’s latitude
* Station’s location defined by longitude and latitude
* State where the station is located
* A rectangular region defined by latitudes and longitudes of the corner points of the rectangle.

### Weather Data Request

1. The system shall allow a request for data from multiple weather stations, multiple instrument data types for each weather station, and a single time range to be made in a single request.

## Stimulus

Transitions between the states of the system are often initiated by a stimulus. State transition diagram that that motivates the requirements described in this section can be found in Appendix B.

1. When the system receives a request from the Client Program, the system shall verify that the request is either for a list of weather stations or for a table of historic weather data.

### Acquiring a List of Weather Stations

1. When the system receives a request for a list of weather stations, the system shall query the historical weather data service and return the list to the Client Program.
2. When the system receives a request for a list of weather stations by any of the attributes listed in REQ 21, the system shall query historic weather data source for the available weather stations satisfying the requested parameters.
3. When the system receives from the historical weather data source the list of weather stations satisfying the requested parameters, the system shall return it to the Client Program. If the system cannot establish connection to the historical weather data source, the system shall log an error in the activity log along with the time stamp when the failure occurred, and return an error message to the statistical module.
4. The system shall query the historical weather data source for all weather stations within one degree from the requested longitude and/or latitude.
5. The system shall query historical weather data source for all weather stations within the requested state.
6. The system shall query historical weather data source for all weather stations within the rectangular area defined by the latitude and longitude of the corner points.

### Acquiring Weather Data

NOAA only allows a user to request the same data once per hour. The system should store any data it receives from the Historical Weather Data Source so that repeated requests can be serviced.

1. When the system receives a request for weather data, the system shall verify that the request contains all of the following parameters: a non-empty set of weather stations, a non-empty set of instrument data types, and a time range. If any of the parameters is missing, the system shall return an error message to the Client Program.
2. When the system receives a request for weather data and the system has verified that the request contains all required parameters (described in REQ 30), the system shall query the local database for the requested data.
3. When the system receives an empty file from the local database while querying the database for weather data, the system shall query the historical weather data source for the requested data.
4. When the system receives requested weather data from the local database, the system shall verify that all requested data are received and return data to the statistical module.
5. When the system receives requested weather data from the local database and the system verifies that only a subset of requested data is received, the system shall verify that the historical weather data source has not been queried in the last hour for the data that are missing. If the historical weather data source has been queried in the last hour for the data that are missing, the system shall return to the Client Program the subset of the requested data and a warning that not all requested data exist.
6. When the system receives requested weather data from the local database, the system verifies that only a subset of requested weather data is received and the historical weather data source has not been queried in the last hour for the data that are missing, the system shall query historical weather data source for the data that are missing.
7. When the system receives weather data from the historical weather data source, the system shall parse data, verify data all requested data are received, store data in the local database along with the time stamp when data was received from the historical weather data source, and return data to the statistical module. If not all requested data are received, the system shall record “N/A” in the local database for missing data and return a warning to the statistical module that not all requested data are available.
8. When the system receives an empty file from the historical weather data source, the system shall log in an error in the activity log, record “N/A” in the local database for the requested data, and return an error message to the statistical module.

## Functional

No further functional requirements have been identified.

# Non-behavioral Requirements

## Performance Requirements

No performance requirements have been identified.

## Security

1. The server system shall require the Administrator to login.
2. The system shall be delivered with a default sys admin.
3. The system shall require the sysadmin to change password on first login.

## Qualitative Requirements

### Availability

No availability requirements have been identified.

### Maintainability

1. The parts of the system coded in Flex shall be coded using Adobe Flex naming convention specified in http://opensource.adobe.com/wiki/display/flexsdk/Coding+Conventions.

### Portability

1. The user interface shall run on Microsoft Internet Explorer 8.0, Mozilla Firefox 3.5, Google Chrome 3.0, and Apple Safari 4.0.

### Design and Implementation Constraints

1. The system shall make use of a PostgreSQL 8.3 Database.
2. The system shall be developed as a client-server application with the server providing data access services.

# Other Requirements

## Database

1. The system shall store the weather data in the CEON Weather Database managed by PostgreSQL.
2. The system shall not store duplicate set of data into the database.
3. The database schema shall be based on the existing database schema, presented in Appendix D.

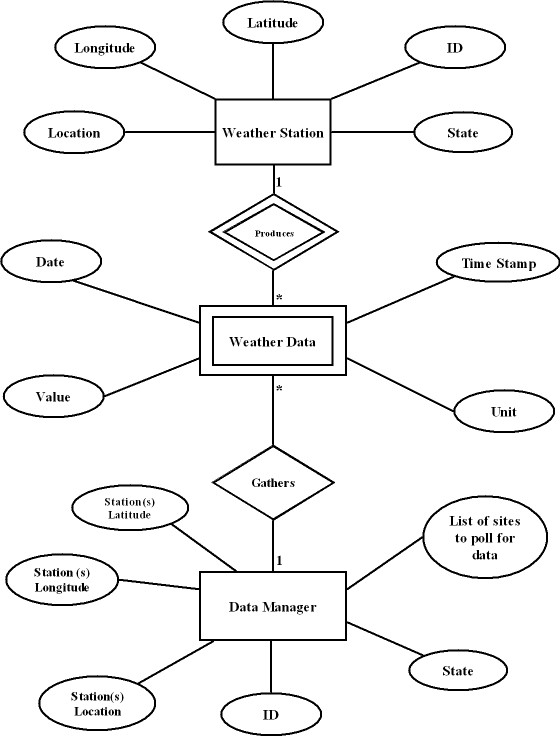
## Operations

No operations requirements have been identified.

## Site Adaptation

No site adaptation requirements have been identified.

# Appendix A: Static Model



# Appendix B: Dynamic Model



# Appendix C: Functional Model



# Appendix D: Database Schema

TBD.

♦