# (Name of your distributed application)

Requirements

**1. Introduction and Context**

WMS is a distributed system for collecting weather data and forecasting short weather patterns. The system will include thousands of weather stations, at least 10 weather-data servers, and multiple foresting servers. End users will a local program to request/view slices of recent weather data and request/view forecasts.

Every weather station must be able to discover the addresses of operational severs. Weather servers also must be able to discover each other. The weather stations gather readings, like temperature, wind direction and speed, barometric pressure, perception, and visibility every couple of minutes. Each station tries to send that data to at least two different weather-data server every 30 minutes.

The weather-data servers share their weather data with other weather servers, so most of the servers have most of the data most of the time and all the data is replicated in at least two places.

Authorized weather forecasting servers can download weather data, in bulk, from weather-data servers at any time. They can grab slices of data based on geographically regions and date ranges.

Make sure your protocols are efficiency. Are make sure the protocols can handle stations and servers going offline and coming back online.

**2. Actors and their Goals**

There are two primary actors for WMS: weather measurement hardware and weather professionals. The weather measurement hardware has the goal of providing regular measurements to the system so they can be aggregated, viewed by weather professionals, and used in forecasts. Weather professionals have the goals of a) viewing any slice of weather data by date, geographic area, or measurement type and b) viewing forecasts for a geographic area.

**3. Functional Requirements**

1. Weather Data and Weather Stations
   1. The system will track temperature, wind direction, wind speed, barometric pressure, precipitation, humidity, and visibility from any location with a Weather Station
      1. Weather Stations will record measurements from physical sensor every few minutes.
         1. However, for the initial version, the Weather Stations will simply generate semi-realistic measure for the systems communications and infrastructure can be fully tested before integrating with physical sensors
         2. The frequency of measurement should be a configuration parameter for a Weather Station
      2. Weather Stations will transmit the measurements, e.g., weather data, to Data Servers
         1. To provide some redundancy, Weather Stations will send every piece weather data to at least two Data Servers.
         2. To save battery power, weather Stations may transfer the recorded measures less frequently than the measure frequency.
         3. The frequency of transmission should be a configuration parameter for the Weather Station
   2. Weather stations should be able to join or leave the system at any time, without interfering with the normal operation of any other process
2. Data Servers
   1. A Data Server can manage all or any subset of the known weather data.
   2. Data Servers should be able to join or leave the system without disruption of the system as a whole
   3. All the system’s processes should be discovery what Data Servers are available and detect when they are no longer available
   4. The system should make a “best effort” to ensure that every piece of weather is available on at least two different Data Servers.
      1. Data Servers should automatically replicate pieces of weather data among themselves, according to a replication strategy that optimizes query times from the Weather Clients and Forecast Servers, as well as network utilization among the Data Servers.
         1. Frequently accessed pieces of weather data could be replicated to all the servers.
      2. If a Data Server fails or goes offline for any reason, the system should allow it to sync with other Data Servers when it comes online so it holds an equal share of weather data, according to the replication strategy.
   5. A Data Server should be able to handle requests for weather data from Forecast Servers and Weather Clients
      1. A request could be limited to specific geographic areas, specified by a geospatial point, circle, or polygon.
      2. A request could be limited by a time range.
      3. A Data Server should only respond with weather data that it holds. If it doesn’t have any weather that satisfy the request, it respond in that indicates this condition
3. Forecast Servers
   1. A Forecast Server must be able to compute a 10-day forecast for a specified geographic area specified by a geospatial polygon and time of day specified by an hour
      1. Before computing a 10-day forecast, it will automatically collect the necessary weather data from Weather Servers.
      2. The forecast will consist of predicated weather data for the specified hour for each of the 10 days.
      3. The initial version of the Forecast Server should take a very simple approach to the forecast, based on rolling averages. This will allow developers to test the system communications and infrastructure before developing sophisticated forecast algorithms.
   2. Forecast Servers should be able to join or leave the system without disruption of the system as a whole
   3. Weather Clients should be able discovery available Forecast Servers
4. Weather Clients
   1. Any end user should be able to run a Weather Client at any time
   2. The weather client should allow the end user to query weather data from available Data Servers by area and time range, and view that data in a meaningful way
   3. The weather client should allow the end users to query any available Forecast Server for a 10-data forecast, and view that forecast in a meaningful way.

**4. Non-functional Requirements**

1. The system will be developed with C#, .Net, and the full suite of Visual Studio tools, including the Microsoft Test Framework.
2. The team will follow an Agile development method
3. The team will a Scrum broad to plan and coordinate tasks to be coordinated
4. The team will use Git for managing artifacts and version control
5. **Future Features**
6. In the Weather Station, replace the generation of simulated data with reading of real weather data from a variety of physical sensors
7. In the Forecast Server, replace the light-weight forecaster with a sophisticate forecast based on the science of meteorology.
8. **Glossary**

TBW