GEO CONTROL SYSTEM

Requirements document

Abstract

This document describes the requirements for the GeoControl system, project developed in Software Engineering course, ay 2024-2025

Contents

The rest of the document is based on the APIs delivered at the beginning of the project. In other words, this document results from a reverse documentation task (produce a requirement document starting from the APIs). Usually, the opposite happens (produce APIs, software design, and code from the requirement document). This choice was made to minimize ambiguities: asking the students to write a requirement document from just an informal description would produce a huge variety of results, quite difficult to evaluate by the teachers. On the other hand, starting from defined APIs should reduce the variability of the student requirement documents.

GeoControl can be developed in two versions: a simulated version (no actual connection with gateways and sensors) and a real one (with actual connection with gateways and sensors).

In the course, GeoControl will be implemented in a simulated version, since providing each group's hardware environment (sensors, gateways) would be too complicated. So, the document will describe the simulated version. For comparison, in the Appendix, the main differences with the real version are highlighted.

Informal Description

GeoControl is a monitoring system designed to manage different kinds of sensors able to measure geological, meteorological, environmental, and other physical variables such as position, temperature, pressure, humidity, gas concentration, etc.

Business model

Value proposition: provide a software platform to store and process measurements from various sensors (ex, humidity, temperature, pressure, position, etc.) in a physical environment.

Customer segments: Business-to-business. Geocontrol is aimed at private or public entities charged with installing and monitoring sensor networks. Examples include:

- Public agencies (ARPA, Unione Montana, and similar) monitoring meteorological status of specific areas
- Companies owning or managing large buildings, monitoring internal comfort (temperature, humidity, pollutant gases concentration)
- Public agencies monitoring seismic status of specific areas

Revenue streams: the product could be sold in several ways: yearly license fee, monthly fee, or fee per number of sensors. On top of that, other fees could be charged for support in training or overall configuration.

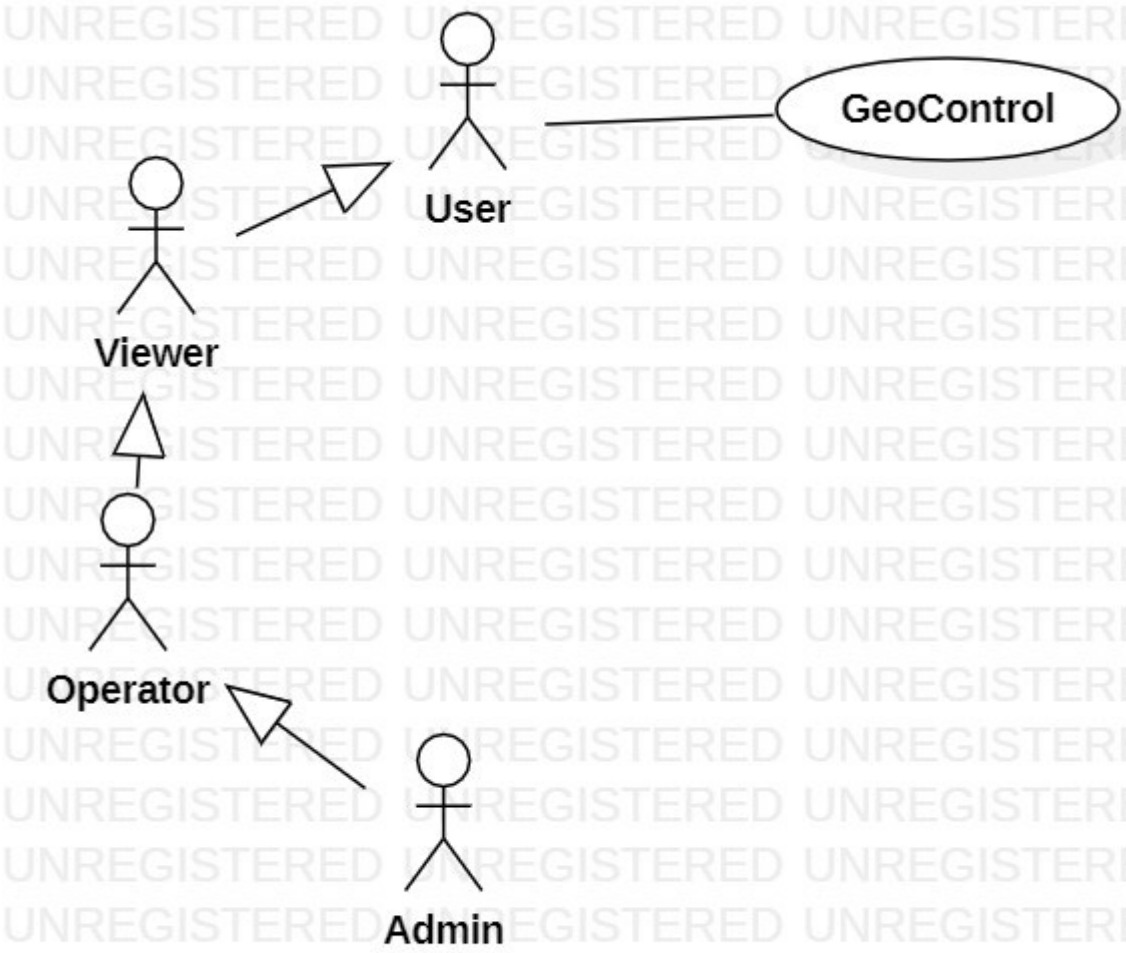
Comment

The best way to represent a business model is the Business Model Canvas (BMC) technique. Here we use a simplified version since the BMC was not part of the course program. The main point is that the product is B2B (no B2C) and sold in various ways, but not to simple individual customers (so there is no need to provide payment functions).

Stakeholders

Name	Description
Company X	Develops the Geocontrol systems, sells it and customizes it for different buyers (such as ARPA or Union Mountain Communities). Within Company X specific involved roles are owners of company, developer and technical roles, marketing roles.
Buyer	Private company or public agency (such as Union of Mountain Communities) which buys Geocontrol from Company X and operates it to collect and monitor measurements from the sensor network.
User	Employee of Buyer who actually uses the application. She can be Viewer, Operator or Admin.
Gateway, Sensors	Collects and redirects measurements from sensors. Changes in these hardware components (and especially their digital interface) could cause changes in Geocontrol

Context Diagram



User is the generic user, who can specialize in Viewer (read-only access to measurements), Operator (setup of network gateways and sensors, force-write measurements), and Admin (user management). See the table of access rights below for details.

Comments:

There is no connection with the environment (no actor ‘Sensor’ or ‘Gateway’ in the context diagram). The GeoControl system (as defined by the APIs) is a purely software product (no hardware parts like gateways and sensors) that simulates the real system (in the APIs, the measurements from sensors are simulated via the endpoint Post Measurement).

Components of the Geocontrol system (front end, back end) appear in the system design and in the deployment diagram, not here.

Also, note that the context diagram shows the interaction between the system and actors during ‘operation’, not during ‘development’. Software developers, sensor vendors, and so on may be stakeholders but are not actors.

Interfaces

Actor	Physical interface	Logical interface
User	PC	REST API
Viewer	PC	REST API
Operator	PC	REST API
Administrator (Admin)	PC	REST API

Stories and personas

TBD

Could be: public agency – private company

Anna Rossi is a field geologist at a regional environmental agency. Her job starts every dawn by validating the previous day's humidity, temperature, and rainfall figures. She wrangles three vendor dashboards, exports raw CSV files, harmonises column names in Excel, and scans thousands of rows for anomalies. Crashes and mismatched date formats turn what should be a quick check into a half-hour ritual that still leaves her wondering whether she missed something. She wants one place where clean data appears immediately, and abnormal values are impossible to overlook.

Marco Bianchi belongs to the same agency's ICT unit. He installs LoRa gateways in remote mountain shelters, maps every sensor to its physical position, and runs traceability logs for audits. Today, he relies on a patchwork of manufacturer utilities and homemade scripts, none of which share authentication. Because he cannot rehearse deployments from the office, he often drives to a site twice—first to install, then again to fix configuration errors discovered only when real traffic starts flowing. Any downtime hurts the agency's reputation, so he prizes anything that lets him configure and test the full chain before leaving civilisation.

Carla De Santis administers the platform. She must open separate spreadsheets and directories whenever staff join, change teams, or leave to create or disable accounts. Password resets arrive by email, and she has no quick way to see which accounts are still active. Keeping this manual inventory accurate steals hours weekly; she needs a single console that lets her add, list, and deactivate users in minutes.

Current ("as-is") stories

Anna starts at seven, pulls readings from three vendor sites, merges them in Excel and spends twenty minutes chasing a suspected humidity spike because date formats clash. By the time the morning bulletin is ready, half an hour is gone, and confidence is low.

A summer storm hits later; civil-protection officers phone for confirmation because no tool shows live rainfall across every station. Anna repeats the export-merge ritual under pressure.

Marco receives new gateways. IDs can be assigned only on-site, so he drives two hours, edits JSON over SSH, and discovers a typo back at base. A second journey is inevitable.

During another installation, an inclinometer reports null values. The only diagnostic option is a serial cable on a windy ridge; Marco sends raw strings, unsure whether the fault sits in the sensor, radio link, or database until he returns.

Carla compiles user lists from three sources every week to keep the access roster clean. She cross-checks names line by line and repeats the exercise after each resignation, losing half a day every time.

Future (“to-be”) stories with GeoControl in place

Anna signs in as a viewer, and the dashboard immediately shows the last twenty-four hours for her network. One click isolates outliers; the other displays mean and standard deviation. She copies the JSON payload into her report and finishes in five minutes. When heavy rain arrives, she repeats the query and forwards the live chart without opening Excel.

Marco authenticates as an operator, defines a new network, adds gateways and sensors in the simulator, and injects a synthetic temperature reading. A quick statistics call confirms the data path works, so he leaves for the mountains confident that no configuration typo will haunt him. When one inclinometer comes up blank, he sends another test reading from his tablet; the value instantly confirms a hardware fault, which he replaces on-site.

Carla logs in as admin, lists all users, filters by “last login more than ninety days ago,” and deactivates four dormant accounts in under two minutes. She then creates an account for a new hire with a single form and no spreadsheet juggling. Her weekly roster check now takes minutes instead of hours.

Comment:

Since this project is a B2B case (much more limited number of users versus a B2C case), there are not many personas; in fact, personas match up with actors.

Functional Requirements

ID	FR
FR1	**Authentication**
FR1.1	Authenticate user
FR2	**Manage users**
FR2.1	Retrieve all users
FR2.2	Create a new user
FR2.3	Retrieve a specific user
FR2.4	Delete a specific user
FR3	**Manage networks**
FR3.1	Retrieve all networks
FR3.2	Create a new network
FR3.3	Retrieve a specific network
FR3.4	Update a network
FR3.5	Delete a specific network
FR4	**Manage gateways**
FR4.1	Retrieve all gateways of a network

FR4.2	Create a new gateway for a network
FR4.3	Retrieve a specific gateway
FR4.4	Update a gateway
FR4.5	Delete a specific gateway
FR5	**Manage sensors**
FR5.1	Retrieve all sensors of a gateway
FR5.2	Create a new sensor for a gateway
FR5.3	Retrieve a specific sensor
FR5.4	Update a sensor
FR5.5	Delete a specific sensor
FR6	**Manage measurements**
FR6.1	Retrieve measurements for a set of sensors of a specific network
FR6.2	Retrieve statistics for a set of sensors of a specific network
FR6.3	Retrieve outliers for a set of sensors of a specific network
FR6.4	Store measurements for a specific sensor
FR6.5	Retrieve measurements for a specific sensor
FR6.6	Retrieve statistics for a specific sensor
FR6.7	Retrieve outliers for a specific sensor

Comment: FRs are 1:1 with the APIs. This is mainly because APIs were given. In a real-life case, the APIs would follow the definition of requirements, so a 1:1 match would be less likely.

The access rights (which actor should be authorized to use which functions) are better represented in a separate table (and not in the FR) for two reasons. They can change over time and are orthogonal to the list of functions.

Non-Functional Requirements

Id	Type	Description	Refers to FR
NFR1	Security	Only authorized users can access functions	All
NFR2	Reliability	Max 6 measurements per sensor per year lost	FR6.4
NFR3	Localization	Dates and times must be expressed in UTC in milliseconds	FR6
NFR4	Efficiency	All functions should respond in < 0.5sec	All

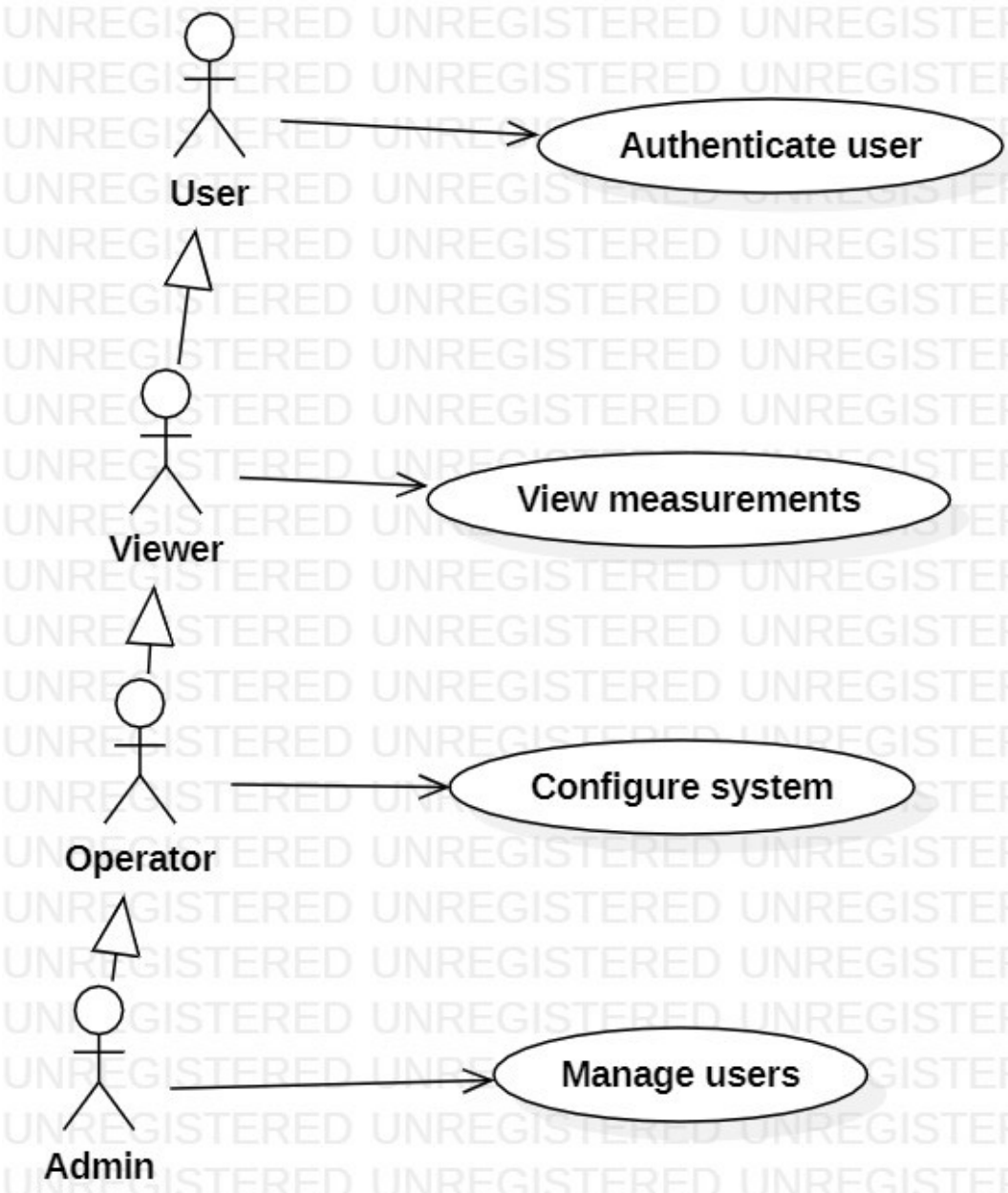
Access Rights

ID	FR	Admin	Operator	Viewer	Unlogged user
FR1	**Authentication**				
FR1.1	Authenticate user	✅	✅	✅	✅
FR2	**Manage users**				
FR2.1	Retrieve all users	✅	❌	❌	❌
FR2.2	Create a new user	✅	❌	❌	❌
FR2.3	Retrieve a specific user	✅	❌	❌	❌
FR2.4	Delete a specific user	✅	❌	❌	❌
FR3	**Manage networks**				
FR3.1	Retrieve all networks	✅	✅	✅	❌
FR3.2	Create a new network	✅	✅	❌	❌
FR3.3	Retrieve a specific network	✅	✅	✅	❌
FR3.4	Update a network	✅	✅	❌	❌
FR3.5	Delete a specific network	✅	✅	❌	❌
FR4	**Manage gateways**				
FR4.1	Retrieve all gateways of a network	✅	✅	✅	❌
FR4.2	Create a new gateway for a network	✅	✅	❌	❌
FR4.3	Retrieve a specific gateway	✅	✅	✅	❌
FR4.4	Update a gateway	✅	✅	❌	❌
FR4.5	Delete a specific gateway	✅	✅	❌	❌
FR5	**Manage sensors**				
FR5.1	Retrieve all sensors of a gateway	✅	✅	✅	❌
FR5.2	Create a new sensor for a gateway	✅	✅	❌	❌
FR5.3	Retrieve a specific sensor	✅	✅	✅	❌
FR5.4	Update a sensor	✅	✅	❌	❌
FR5.5	Delete a specific sensor	✅	✅	❌	❌
FR6	**Manage measurements**				
FR6.1	Retrieve measurements for a set of sensors of a specific network	✅	✅	✅	❌

FR6.2	Retrieve statistics for a set of sensors of a specific network	✅	✅	✅	❌
FR6.3	Retrieve outliers for a set of sensors of a specific network	✅	✅	✅	❌
FR6.4	Store measurements for a specific sensor	✅	✅	❌	❌
FR6.5	Retrieve measurements for a specific sensor	✅	✅	✅	❌
FR6.6	Retrieve statistics for a specific sensor	✅	✅	✅	❌
FR6.7	Retrieve outliers for a specific sensor	✅	✅	✅	❌

Comment: This table was missing from the template document due to a mistake. However, the table is essential. Adding access rights to the list of FR (ex, FR2.1 would be written as 'Administrator retrieves all users') is a much less clear option; it clutters the list of FRs and complicates maintenance of the document (access rights typically change over time, faster than FRs)

Use Case Diagram



Comment: UC should be high-level, nearly all of the solutions proposed by the teams equate UC with functions. Since most functions are low-level (e.g., create user), the result is a large number of low-level UCs. This cannot scale up in larger projects (in fact, the UCD proposed by the teams contains dozens and dozens of UCs; in a larger project, it would contain hundreds of UCs). We have accepted this approach, but please note that UCs should be higher-level, focusing on the value added for actors (as in the proposed UCD). Notice also that the UCs we propose are closer to stories.

Remark also, in the following use cases, the mapping UC – FR: in most cases, a UC maps to many FRs

Use Cases

UC Authenticate user

Actors involved	User
Pre condition	
Post condition	
Nominal scenarios	-sign in (user has account)
Variants	-sign up (user has no account and defines it)
Exceptions	-sign in, password wrong -sign in, username wrong or not existing -sign up,

UC Manage users

Actors involved	Admin
Pre condition	User successfully logged in as Admin
Post condition	
Nominal scenarios	-Create a new user (FR2.2) -Delete a specific user (FR 2.4)
Variants	-Modify characteristics of a specific user (FR2.3 FR2.4 FR2.2)
Exceptions	-Create a new user, user already exists (FR2.2)

UC Configure system

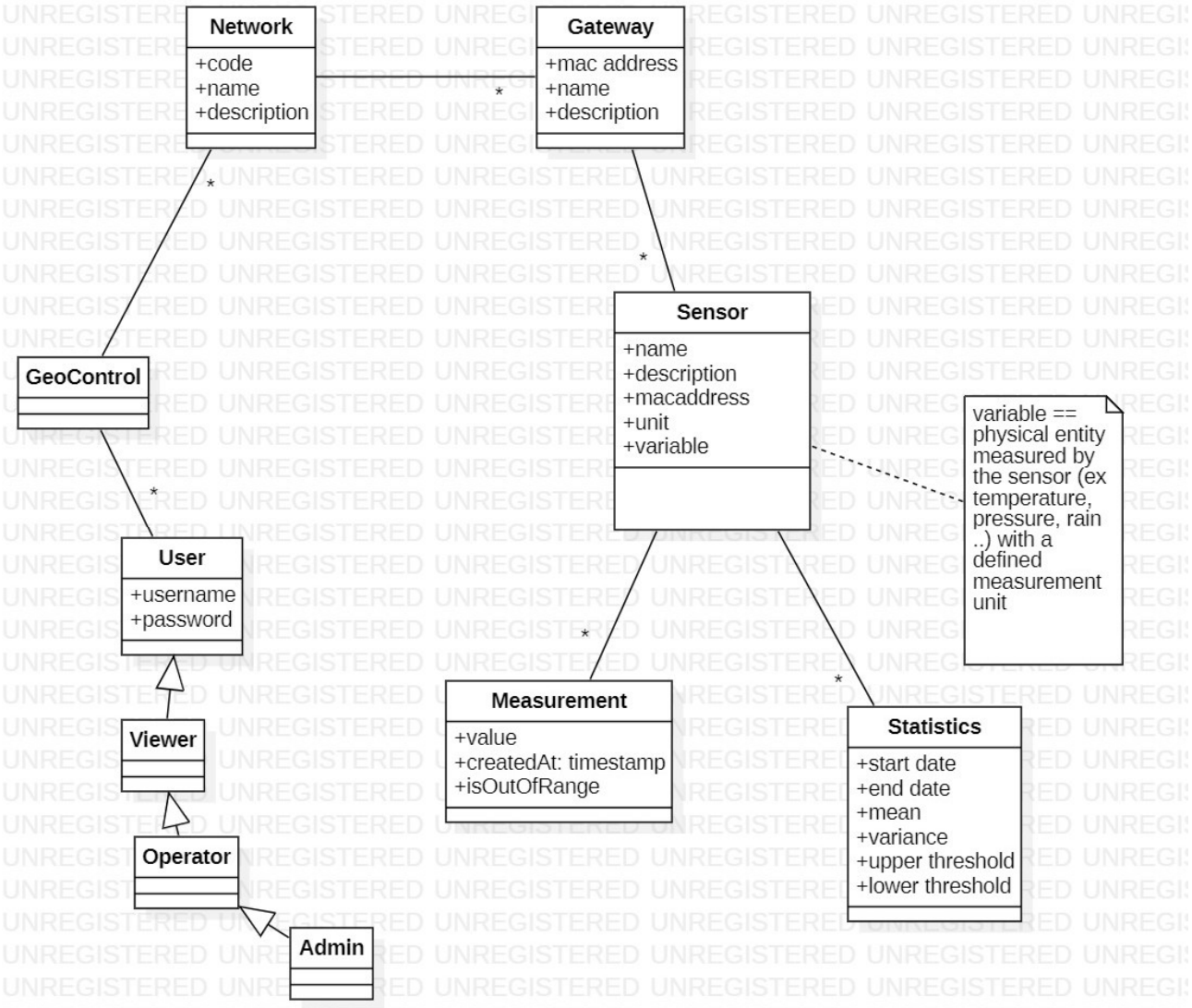
Actors involved	Operator
Pre condition	-User successfully logged in as operator
Post condition	
Nominal scenarios	-create a network, create a gateway, create a sensor -create a sensor, attach to existing network and gateway
Variants	
Exceptions	-network not found -gateway not found

UC View measurements

Actors involved	Viewer
Pre condition	-User successfully logged in as viewer -System configured (at least a sensor attached to a gateway, attached to a network has been defined)
Post condition	-

Nominal scenarios	-Viewer retrieves measurements for a certain sensor (FR 6.5) -Viewer retrieves outliers for a certain sensor (FR6.7) -Viewer retrieves statistics for a certain sensor (FR6.6)
Variants	-Viewer retrieves measurements for all sensors of a certain network (FR6.1) -Viewer retrieves outliers for all sensors of a certain network (FR6.3) -Viewer retrieves statistics for all sensors of a certain network (FR6.2)
Exceptions	-sensor not found -network not found -gateway not found

Glossary



Comments:

A separate class should not represent outliers; an outlier is a measurement, so it is represented by a Boolean field ‘isOutOfRange’.

Viewer, Operator, and Admin could also be represented by a ‘type’ attribute in class User.

The relationship between Network and Gateway is 1 to many (and so gateway to Sensor). The latter represents a physical topology and is clearly 1 to many. The former represents a logical organization (Network is a logical concept, not physical), so it could be possible to consider a gateway (and its sensors) belonging to more than one Network. This should be discussed with the stakeholders.

System Design

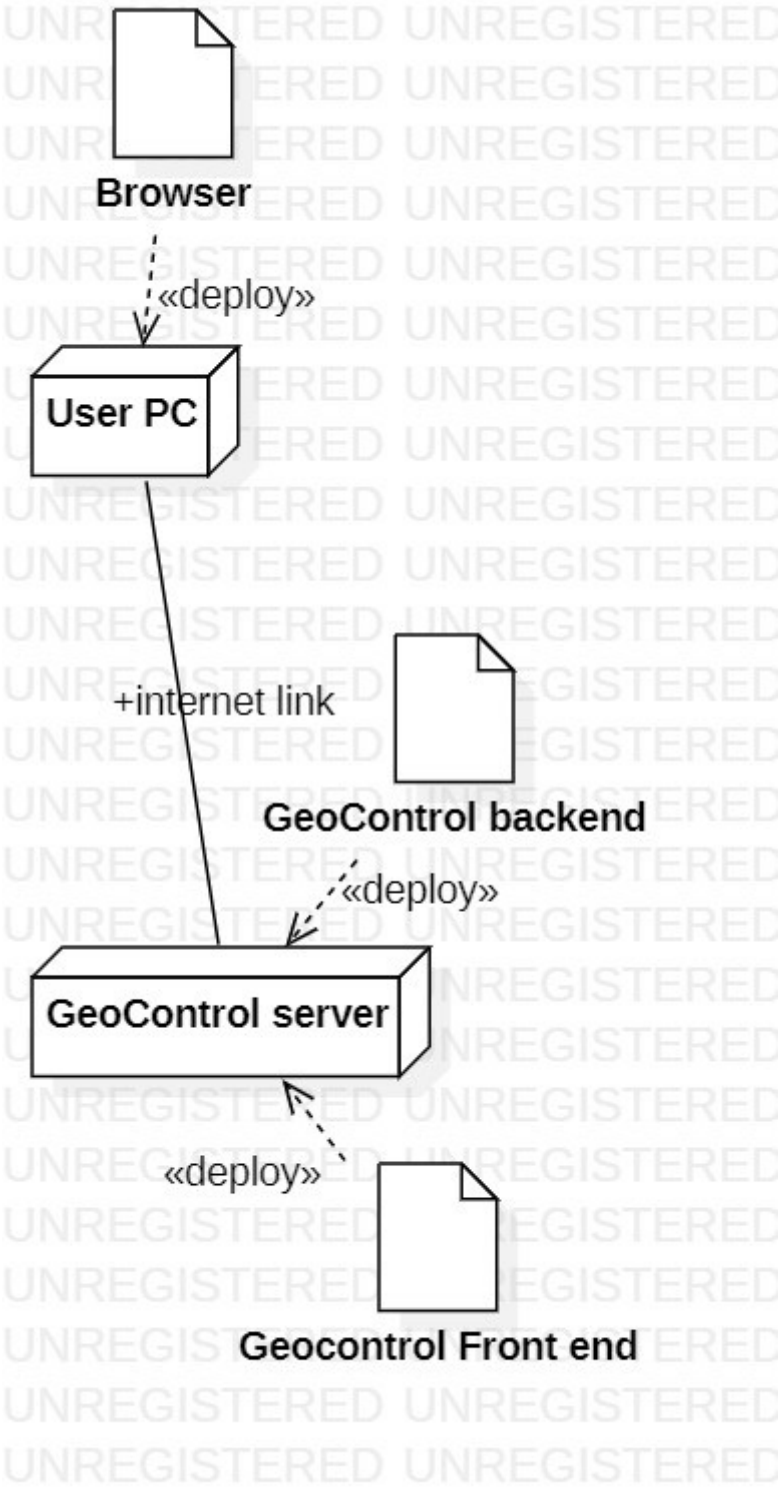
The system is composed of

- Geo Control back end
    - Geo Control Database

Comment: The system design shows the components of the Geocontrol system and should be read in parallel with the context diagram. The main information shown by both is that the hardware part (sensors, gateways, servers running front end, back end, and database) is ‘not’ part of the system. This is important for contractual reasons (the customer needs to know what he pays for, what is included in the contract, referring to the product, and what is not included). Normally, the requirement document is a technical annex to the contract signed between the vendor and the customer. The contract contains the legal wording, commercial information (pricing, delivery dates, penalties), and the technical annex, which is the more technical part of the contract.

A GUI is needed for a simpler use of GeoControl, which could be implemented via a front-end component. The front end is not part of the product either, but it appears in the Deployment diagram.

Deployment Diagram



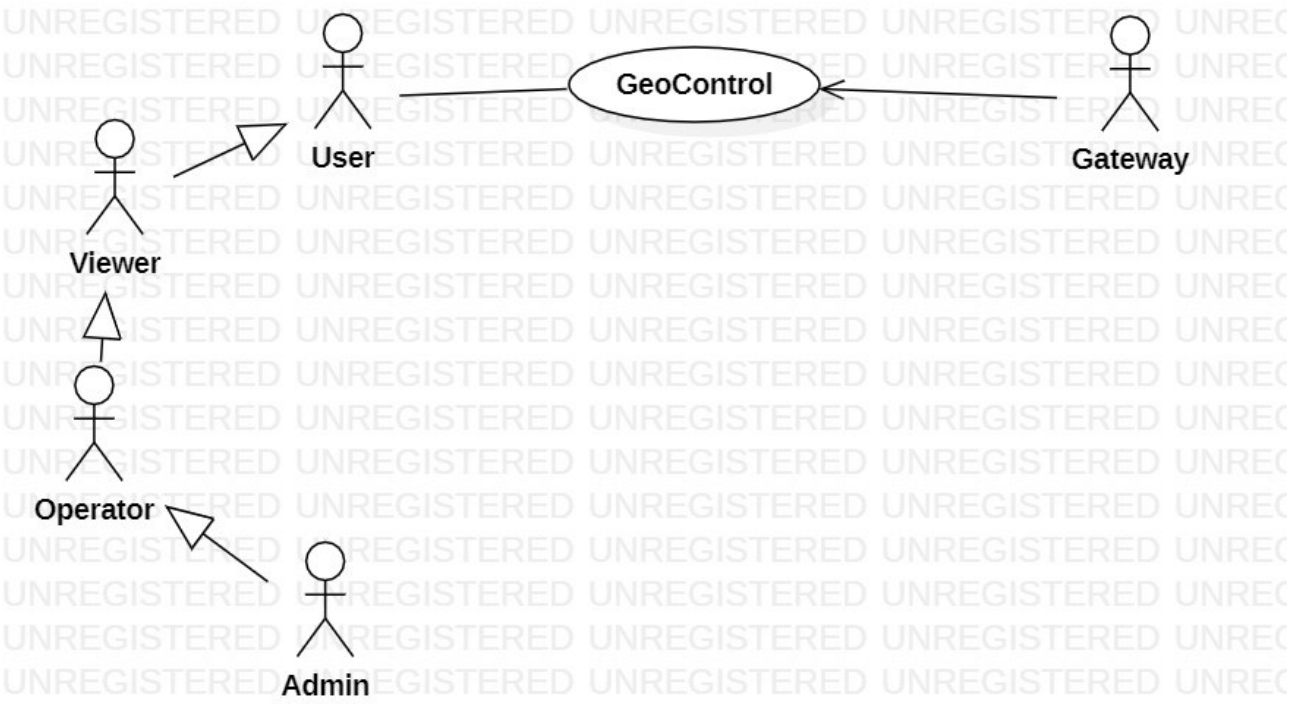
Comment

The DD shows the hardware/software architecture of the simulated Geocontrol. In particular, the GeoControl server (not part of the system) must appear here.

APPENDIX

Here we present a possible real (not simulated) version of GeoControl.

Context diagram



Comment

This Context diagram assumes that the product is purely a software product (APIs and their implementation), so the hardware part (sensors and gateways) is outside. Gateways are connected to sensors, but this cannot be represented here (but appears in the Deployment diagram).

Other options are possible (ex, gateways and sensors could be part of the system). The important point is that the Context diagram and system design must be consistent.

FR

The FRs should be enriched with functions to read and possibly process measurements from the gateways.

NFR

No changes here.

UCD and UCs

At least a UC should be added to manage the reading, processing, and storage of measurements.

Glossary

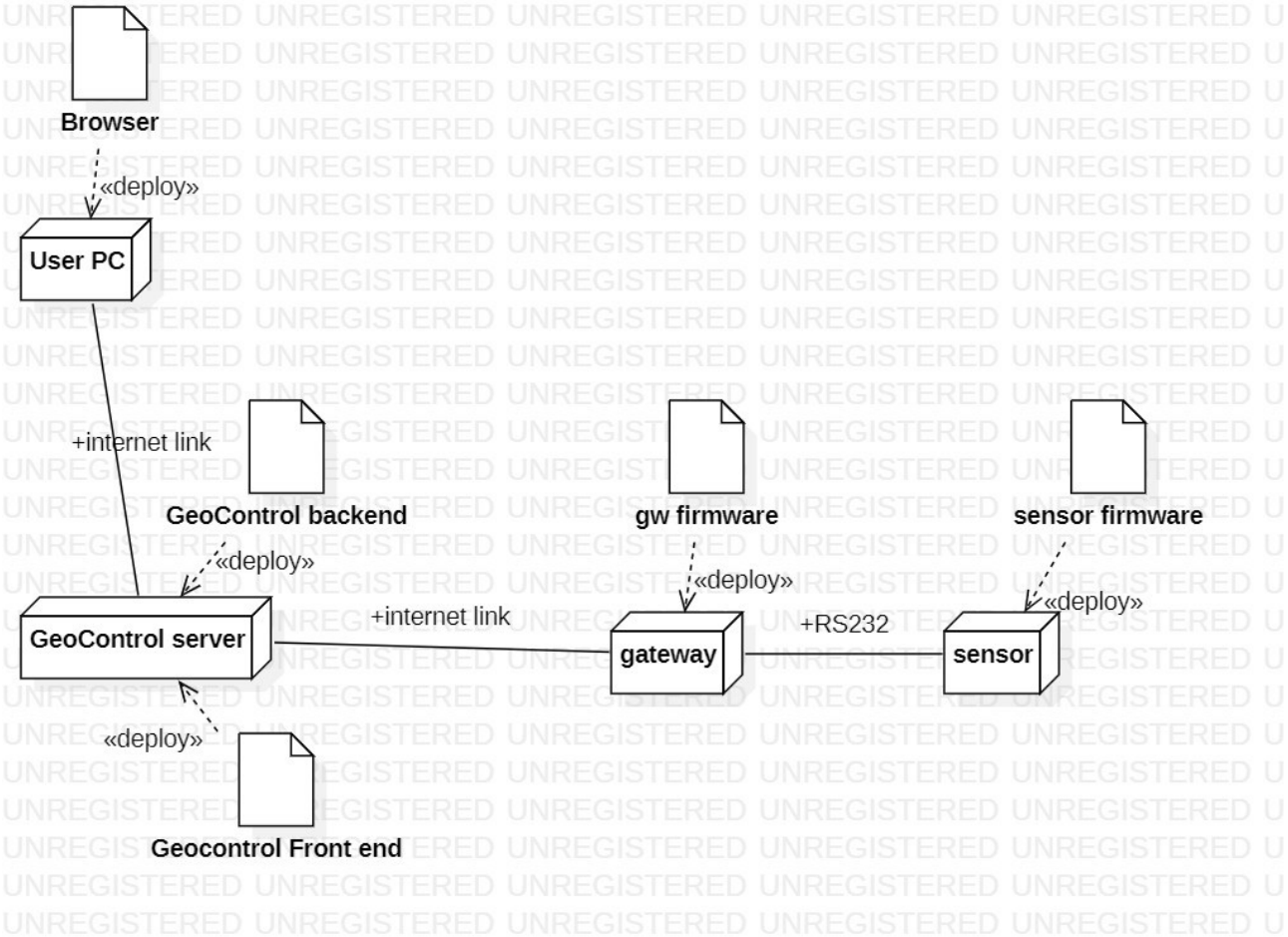
No changes here

System design

No changes here, considering GeoControl as a purely software product.

As said above in the comments about the context diagram, another option could include in the system both gateways and sensors. In this case the system design should list them too.

Deployment diagram



The DD shows the complete picture of all hardware and software components needed to run the system, without considering the boundary in/out (as in context diagram and system design). So here, the sensor, gateway, servers, and browser appear, even if they are not part of the system.