Contents

[1. Preliminary software design 1](#_Toc502225010)

[1.1. The Exception Reporter 1](#_Toc502225011)

[1.2. The Controller 1](#_Toc502225012)

[1.3. The Taximeter Client 2](#_Toc502225013)

[1.4. The Engine 2](#_Toc502225014)

[2. The preliminary production infrastructure 3](#_Toc502225015)

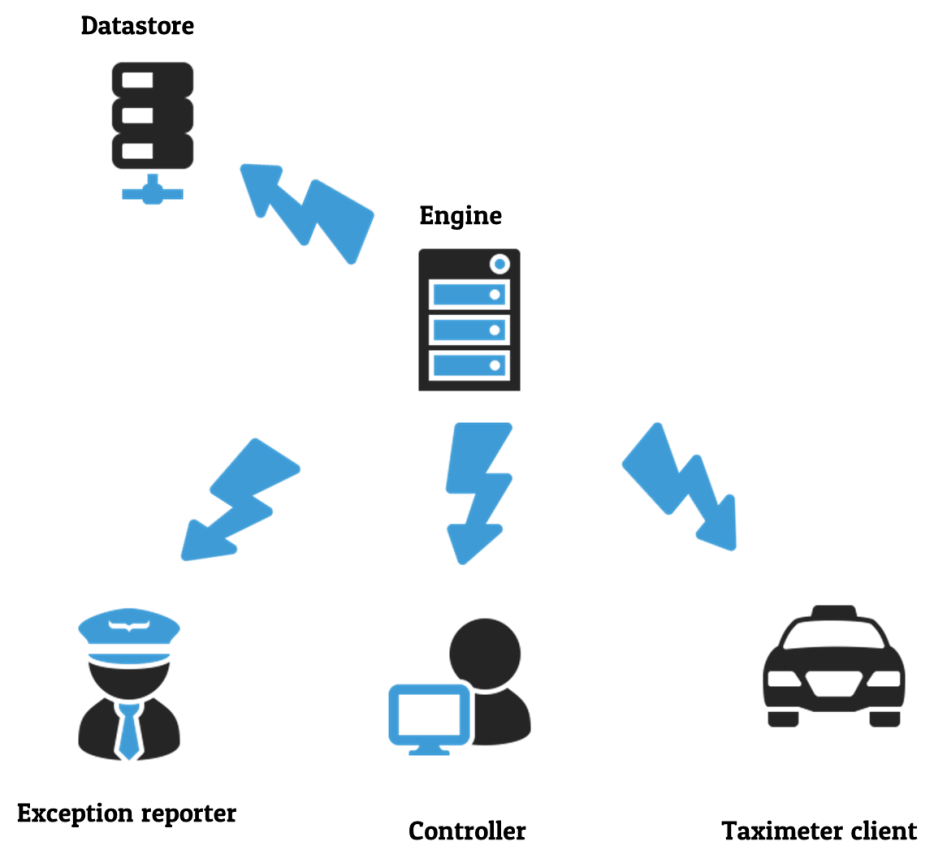
[2.1. Platform for the engine 4](#_Toc502225016)

[2.2. Platform for the datastore 4](#_Toc502225017)

# Preliminary software design

The purpose of the preliminary design is to get an overview of the project. This overview is necessary to break up the system into manageable pieces and to begin to prioritize the design/development work

In the figure below the project has been broken down into its major constituents.



## The Exception Reporter

The exception reporter is a piece of software that will be used by taxi companies to:

* Report exceptions
* View alarms pertaining to their company
* Download data about their company that has been reported to 3 parties

## The Controller

The controller is used by the employees of the reporting center running the Disclosure System and will be used to:

* Set up users and privileges
* Verify users and ensure that they are only given access to data and functionality that they have privileges to.
* Register taxi companies and their vehicles.
* Review all data in the data store
* Create logs of all user actions
* Present alarms in case of failed or omitted transitions
* Search for data selections of taximeter data and exception reports
* Initiate the creation of reports to third parties
* Register data requests from third parties

## The Taximeter Client

The taximeter client is the software that will be running in taxis that are reporting to the disclosure system. The Taximeter client will need to:

* Store vehicle parameters and protect these parameters from being tampered
* Covert raw taximeter data to XML and send it asynchronously and encrypted to the Engine.
* Buffer and encrypt data for later transmission in the case communications breakdown with the engine and ensure that this data is not tampered with or deleted while it is stored.

## The Engine

The engine is the sever software that will contain the bulk of the business logic of the system. The server must maintain secure communications and stay available to the reporters, controllers and clients in the system.

The Engine is the only piece of software in the system that will be able to directly read or white data to or from the data store. As such the server is the guardian of the data and must prevent malicious transactions from harming the data.

The server should also be able to see the central components of the system and monitor them and itself for failures such as network down & database down.

The server must monitor the incoming data from the Taximeter Clients and generate appropriate alarms in case of deficient or omitted transmissions.

Create correctly formatted files for third parties who request information.

The server’s role is so complex that we will break down the server design to some major libraries or function sets that can be developed independently to simplify the development of the server.

### XML Library

The final output of the data when being reported to third parties is XML so it makes sense to use XML as the overall system format for transporting data between the software components over the network. We will develop an XML library that facilities the simple conversion of relational data to XML and back again.

### Communications library

Communication is vital in a system that will be very distributed. The communication is not only vital it also poses a security risk. The communication which will be enabled via TCP sockets will have to take place in an asynchronous non-blocking manner as it allows all software components to user the same socket meaning the software will be able to communicate with the engine without any configuration for the network (a synchronous configuration would require a unique port number for each client to ensure stability.) For security purposes all TCP communication will be encrypted.

The communications functionality will be a relatively complex piece of code and will therefore be developed as a library covering both server and client functionality.

### The data store

This is the database that will store all data. The data store has no business functionality or any kind of intelligence whatsoever it is just a data store. As mentioned before the only user of the database will be the engine. This means that no database connection information is stored in the controller, exception reporter or the taximeter client which is good for security. Neither will the database need a public IP, as in the case of the server, protecting the database further.

From TFS 2016:47 we can deduce that a minimum of the following tables will be needed

* Users
* Taxi companies
* Vehicles
* Exception Reports (from customer)
* Taximeter data
  + Batch header
  + Taximeter report header
  + Trip data
* Data requests from third parties
* Data outputs sent to third parties
* Transmission deficiency reports
* Transmission omission reports
* Alarms
* User change logs

# The preliminary production infrastructure

We wish to implement a 3 tier client server architecture this will allow us to scale performance where it is needed, meaning that data, business logic and client hardware can be upgraded independently of each other.

We have already pointed out that the database should not have a public IP and should therefore be housed on its own machine for security reasons.

Given the background of the development team primarily in the dotNet development environment we will be using windows operating systems where ever appropriate and there is no reason not to use the latest widely adopted version Windows10 in its 64bit version.

## Platform for the engine

As regard to which Windows version we feel that there is no need to implement the server version. The security of the Disclosure System is handled by the Disclosure System as well as the communication being encrypted even on LAN communication negating the need for a domain controller. Using a Win10 Pro operating system saves cost due to less hardware and electricity for the same reason with both environmental benefits and will increase the uptime of the system if it needs to run off UPS.

Regarding performance this system is not viewed as a real time system. Whether an operation takes 1-2 seconds or 7-8 seconds is not critical for the correct function of the system. The only important factor speed wise is that the interface for the Exception Reporter and Controller needs to be sufficiently responsive such that the user does not become frustrated. Therefore the server need not need the scalability offered by the Windows Server Environment.

The physical server should have the following minimum specifications which should be sufficient for at least 5000 connected taximeters.

* Intel i5 processer
* 12 GB RAM
* 500 GB SSD

Add one core, 2GB RAM and 50GB HDD to your configuration per 1000 taxi clients over 5000 clients.

## Platform for the data store

We can further save resources by not using Windows Server / SQL Server for the database environment. The development team has most experience with development in Microsoft environments however as the Data Store is not going to have any intelligence and is basically a software package that is being used out of the box free open source software can be considered. Therefore we propose PostgreSQL 9.6 on Debian 9 as a perfectly capable database on a very lightweight operating system. Again this design decision will cut licensing and hardware costs as well as power consumption.

The physical server should have the following minimum specifications

* Intel i5 processer
* 16 GB RAM
* 2TB SSD

## Platform for the Exception reporter

The users of the Exception Reporter are not employees of the system owner and assumption cannot be made as to what platforms they will be working on. This makes it difficult to develop a specific piece of software for this purpose. Developing the same software for multiple platforms is too costly and time consuming. The only sensible option remaining is that the Exception Reporter be hosted on a webserver and run in a browser from almost any platform. In keeping with the 3 tier client server philosophy this server will be housed on a separate server with Win10 Pro 64bit operating system running IIS as a webserver.

The physical server should have the following minimum specifications

* Intel i5 processer
* 12 GB RAM
* 500 GB SSD

## Platform for the controller

The platform for the controller is Win10 Pro 64bit.

The physical PC should have the following minimum specifications

* Intel i5 processer
* 6 GB RAM
* 250 MB HDD

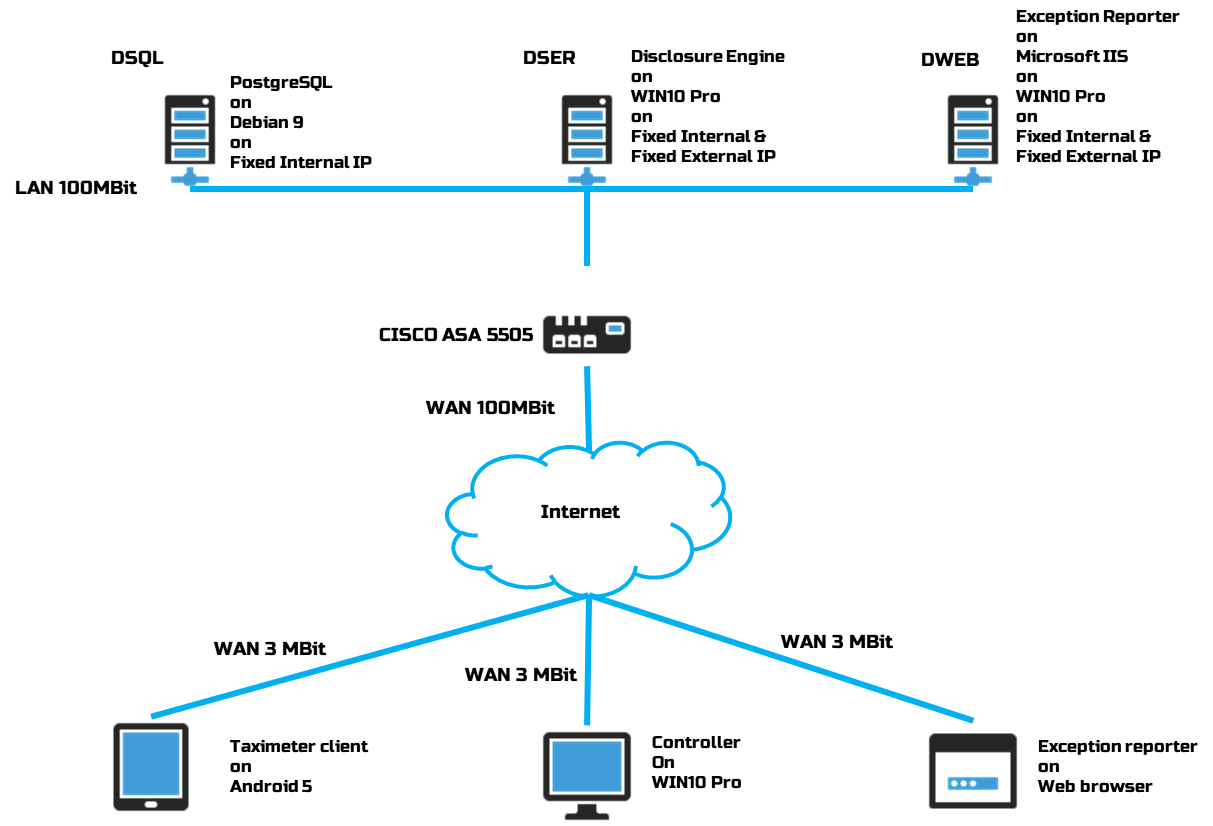
## Platform for the Taximeter Client

For the taximeter client we will select the Android platform version 5 and upwards. This application will take very little resources from the tablet meaning any tablet or smartphone manufactured after 2016 should be fine but we will set some minimum specs to be sure.

* Intel Atom x3 C3200
* 1GB RAM
* 16GB storage

## The Production network Environment and summary

Again no component of this system is real time and in the production design serves to not exaggerate the need for overly powerful hardware rather efficient code and lightweight operating systes. The same goes for the LAN where a 100MBit connection will suffice as will a 100MBit internet connection. For the firewall / router we select the CISCO ASA 5505.



## Test environment

The test environment will consist of a DataBase Server with an Intel Atom Processor and 2 GB RAM with Debian 9 and PostgreSQL 9.6