



2016 - 2017

Noreillie Sven

Vervoort Peter

## ***G4S OLDMan***

(Order License Device MANagement)

Ondernemingsproject voorgedragen tot het behalen van het diploma van Gegradueerde in de Informatica

Promotor: De heer Mathy Paesen



## Preface

First of all, we would like to thank G4S Belgium and particularly Manuel Mestré to give us the support and the opportunity to work on this assignment. His professional and constructive feedback, together with the time he reserved for our project, assisted us to accomplish this thesis.

The knowledge, insight and monitoring of our promotor Mathy Paesen gave us the drive and determination to progress in our project. Throughout the assignment, he monitored us closely, gave advice and probed our choices with critical questions.

And finally, we can't be grateful enough to thank our family to give us the time, support and assistance over these last three years, we would not have accomplished this without them.



# Table of Contents

Preface.....	2
Table of Contents .....	4
Table of Figures .....	7
Table of Tables.....	8
Abbreviations .....	9
Glossary .....	10
Situation of the project.....	12
Description stakeholder .....	12
Situating the project within G4S ICT landscape .....	13
Description of the current situation .....	14
Opportunity .....	14
Planning .....	16
Waterfall Planning.....	16
Agile Planning.....	17
Sprints overview and evaluation.....	18
Analysis stage .....	19
Product Vision.....	19
MoSCoW analysis .....	20
Problem statement matrix .....	21
PIECES analysis.....	22
Performance.....	22
Information Output.....	22
Information Input .....	22
Information of Stored data .....	23
Economics.....	23
Control .....	23
Efficiency.....	24
Service .....	24
Risk management.....	25
Project scope .....	27
Tracking orders regarding to G4SLogin.....	27
Tracking repairs of mobile devices used to report in G4SLogin .....	28
Check the accuracy of data .....	29

Analysis of all collected data.....	29
Configuration of users and accesses.....	29
Information Flow Diagram .....	30
Event response list .....	31
Non-functional requirements .....	33
Configuration management .....	33
Dependency on other parties.....	33
Deployment .....	33
Maintainability .....	33
Network topology.....	34
Performance / response time.....	34
Platform compatibility .....	34
Reporting.....	34
Scalability (horizontal, vertical) .....	34
Security .....	34
Loosely coupled data synchronization flow .....	35
Analyze Database .....	36
Entity Framework code setup.....	36
ERD Diagram.....	37
User management.....	38
OrderItems .....	40
LoginLicenses .....	41
MobileDevices .....	42
ToBeTreated Devices.....	43
States.....	44
Used and implemented technologies.....	45
Version control: Git + visualstudio.com .....	45
Repository and Unit of Work pattern .....	45
Entity Framework.....	46
Unity .....	46
Auto mapper .....	46
Mocking framework.....	47
AngularJS .....	47
Bootstrap .....	49
Purchased front end template .....	49
Business rules implementation .....	50
Building for change .....	51

Technical implementation details.....	53
Validation process .....	53
Tags for dashboarding .....	59
Example .....	59
Synonym possible for devices.....	59
Users – UserRoleGroups – UserRoles.....	60
Translations .....	60
Epilogue .....	61
Appendix .....	62
References.....	63

## Table of Figures

Figure I: G4S BELUX - DIRCOM.....	13
Figure II: G4S Belgium - ICT .....	13
Figure III: Information Flow Diagram.....	30
Figure IV: ERD Diagram.....	37
Figure V: ERD Diagram – ASP.NET user management.....	38
Figure VI: ERD Diagram – User management.....	39
Figure VII: ERD Diagram – PurchaseOrder - OrderItem .....	40
Figure VIII: ERD Diagram - Login Licenses .....	41
Figure IX: ERD Diagram - Mobile Devices.....	42
Figure X: ERD Diagram - ToBeTreated Devices.....	43
Figure XI: ERD Diagram - States.....	44
Figure XII: Repository and Unit of Work pattern.....	45
Figure XIII: Life Cycle of a Purchase Order .....	50
Figure XIV: Device State Diagram.....	51
Figure XV: Object Create/Patch/Update Flow.....	53
Figure XVI: Import Device Data from CSV.....	55
Figure XVII: Device Create Validation .....	56
Figure XVIII: Device Update or Patch Validation.....	57
Figure XIX: LoneWorkerProtect settings Validation .....	58
Figure XX: Faulty devices.....	59



## Table of Tables

Table I: Product Vision - Vision Statement.....	19
Table II: Problem statement matrix.....	21
Table III: Risk management.....	26
Table IV Risk Management - Legend 1 .....	26
Table V: Risk management - Legend 2 .....	26
Table VI: Event Response List .....	32

## Abbreviations

AJAX: Asynchronous JavaScript and XML

API: Application program interface

CLR Object: Common Language Runtime Object

CSS: Cascading Style Sheets

DHTML: Dynamic HTML

DOM: Document Object Model

IMEI: International Mobile Equipment Identity

JSON: JavaScript Object Notation

JWT: JSON Web Token

KPI: Key Performance Indicator

LoginCC: Login Contact Center

OC: Operational Coordinator

OS: Operating System

PO: Purchase Order

POCO: Plain Old CLR object

REST guidelines: Representational State Transfer

SAP: Systems, Applications and Products

SOAP: Simple Object Access Protocol

SPOC: Single Point of Contact

VPN: Virtual Private Network

XML: Extensible Markup Language

# Glossary

**Ajax:** Ajax (Asynchronous JavaScript and XML) is a method of building interactive applications for the Web that process user requests immediately. Ajax combines several programming tools including JavaScript, dynamic HTML (DHTML), Extensible Markup Language (XML), cascading style sheets (CSS), the Document Object Model (DOM), and the Microsoft object, XMLHttpRequest. (TechTarget, 2010)

**API:** An application program interface (API) is code that allows two software programs to communicate with each other. The API defines the correct way for a developer to write a program that requests services from an operating system (OS) or other application. APIs are implemented by function calls composed of verbs and nouns. The required syntax is described in the documentation of the application being called. (Rouse, Margaret, 2017)

**CLR:** The .NET Framework provides a run-time environment called the common language runtime, which runs the code and provides services that make the development process easier. Compilers and tools expose the common language runtime's functionality and enable you to write code that benefits from this managed execution environment. (Microsoft, 2012)

**DOM:** The Document Object Model (DOM) is an application programming interface (API) for valid HTML and well-formed XML documents. It defines the logical structure of documents and the way a document is accessed and manipulated. In the DOM specification, the term "document" is used in the broad sense - increasingly, XML is being used as a way of representing many different kinds of information that may be stored in diverse systems, and much of this would traditionally be seen as data rather than as documents. Nevertheless, XML presents this data as documents, and the DOM may be used to manage this data. (Le Hégaré, Wood, & Robie, 2000)

**IIS server:** Stands for "Internet Information Services." IIS is a web server software package designed for Windows Server. It is used for hosting websites and other content on the Web. (Christensson, 2013)

**IMEI:** International Mobile Equipment Identity: IMEI (International Mobile Equipment Identity) is a 15- or 17-digit code that uniquely identifies mobile phone sets.

**JSON:** JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language. (Introducing JSON, 2002)

**JWT:** JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed. Authentication: This is the most common scenario for using JWT. Once the user is logged in, each subsequent request will include the JWT, allowing the user to access routes, services, and resources that are permitted with that token. Single Sign On is a feature that widely uses JWT nowadays, because of its small overhead and its ability to be easily used across different domains. (Auth0, 2015)

PO: Purchase Order, a template used by SAP to order items and or services.

Poweruser: Individual that operates a computer or device with advanced skills knowledge, experience and capabilities

REST guidelines: REST (REpresentational State Transfer) is an architectural style, and an approach to communications that is often used in the development of Web services. The use of REST is often preferred over the more heavyweight Simple Object Access Protocol (SOAP) style because REST does not leverage as much bandwidth, which makes it a better fit for use over the Internet. The SOAP approach requires writing or using a provided server program (to serve data) and a client program (to request data). (Rouse, 2014)

SAP: A German software company whose products allow businesses to track customer and business interactions. SAP is especially known for its Enterprise Resource Planning (ERP) and data management programs. SAP is an acronym for Systems, Applications and Products. (Webfinance Inc., 2008)

User Stories: User stories are part of an agile approach that helps shift the focus from writing about requirements to talking about them. User stories are short, simple descriptions of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system. (Cohn, 2013)

FlocId: A number used to define and link a client to the address of one of his sites. This number is used by the planning software of G4S.

# Situation of the project

## Description stakeholder

G4S is the world's leading, global, integrated security company specializing in the provision of security and related services across six continents. This in the following sectors:

- Large cap companies
- Government and public companies
- Small caps
- Financial companies
- Energy and utility companies
- Chemical and petrochemical companies
- Transport and logistic organizations
- Health care organizations
- Retail shops and companies
- Private persons

Their strategy addresses the positive, long-term demand for security and related services. The enduring strategic aim is to demonstrate the values and performance that make G4S the company of choice for customers, employees and shareholders.

G4S' people and values underpin everything they do. Their 'One G4S' model brings all areas of the business together and is designed to ensure that the way they go about their business is consistent across global operations.

With more than 610 000 employees they belong to of one the biggest private employers worldwide.

(G4S, 2010)

## Situating the project within G4S ICT landscape

### G4S BELUX - DIRCOM

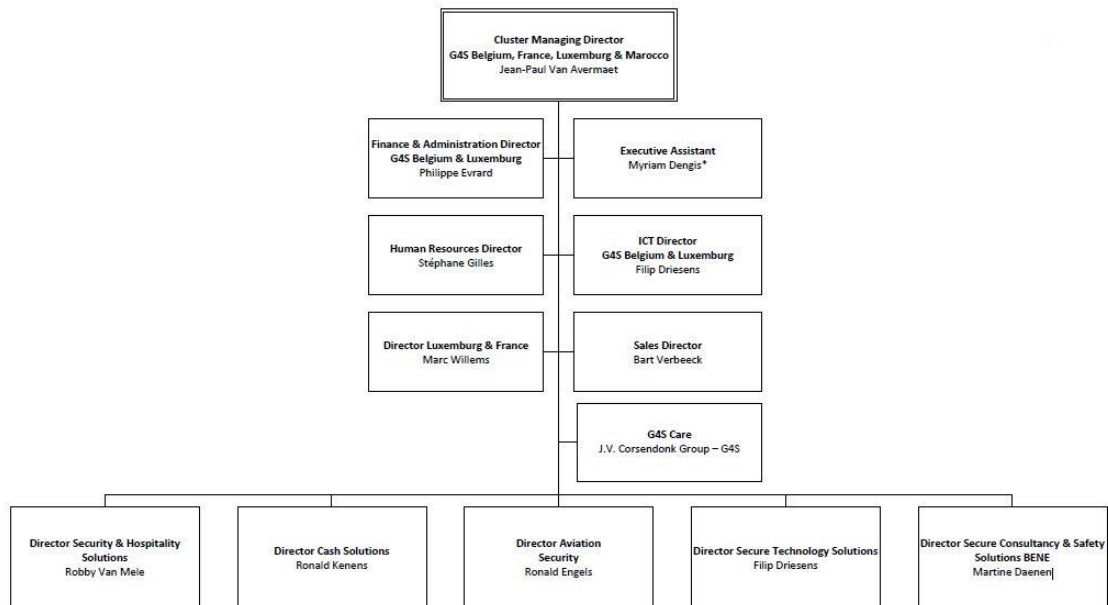


Figure I: G4S BELUX - DIRCOM

### G4S Belgium - ICT

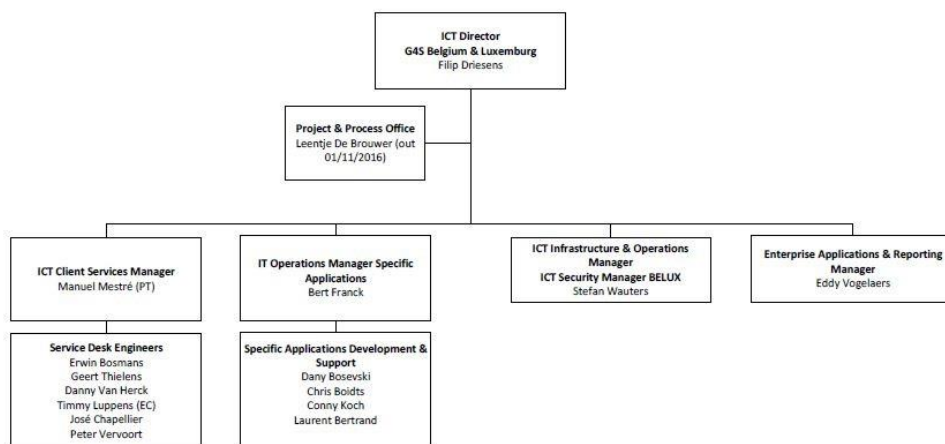


Figure II: G4S Belgium - ICT

## Description of the current situation

The G4S Login application uses different mobile devices for consultation and registration purposes. The configuration, maintenance, service and financial management of these devices is managed by the Business Support department of G4S. To execute these services, they need quite some facts about the devices.

Currently this information is stored in different files. Each file contains information necessary to manage a specific area (e.g.: connote of invoices, registration of pin codes, overview of current rented devices with their rental conditions and prices). These files, often spreadsheets, are stored on a file server or shared by Google Drive. Recurrent or on demand, the available data in the different files is analyzed manually.

Recently a communication protocol has been initiated to report broken devices and repairs.

To streamline the security and safety operations, G4S uses an application - developed by Trackforce - called G4S Login.

This web-based all-in-one system, provides a real-time reporting tool.

Some key points of this application are:

- Event and incident reporting
- Post order management
- Time & attendance tracking
- Visitor registration
- Guard tours & patrols
- Geolocation tracking by dispatch for operational and safety reasons

Through the years, this application product evolved to the needs of the security business of G4S. Static guards use their desktop to report, but the growing need for mobile reporting and dynamic services of the guards is the cause for expanding this application with new features.

The management, deployment and high level configuration of this application - including the mobile devices - is done by G4S ICT. For this management, some Service Desk Engineers were trained to support operations with G4S Login. They even support other European G4S entities by knowledge sharing, providing demonstrations, supporting roll-outs at big clients and assisting in solving high level technical issues.

## Opportunity

The previously described administration can be simplified and partially automated by an application. The current administrative productivity is not measured, hence productivity enhancements cannot be measured either.

One can define different user profiles with different rights. This allows to share the relevant information with the corresponding responsible people. The application can also manage the maintenance and consulting rights of information: who is allowed to put in and change specific data - who is allowed to consult specific data.

The implementation of user profiles allows to increase the number of users significantly. Currently only 3 people have access to the existing data. The application can make this information available in a user-friendly way for the operations management, chiefs and others. In our division, this would be about 110 people.

During the repair of a device, every user with the correct rights can follow up the status of the device in real-time.

The application can also act as a logbook of all modifications implemented on a device. This allows to correct errors and improve processes.

Analysis can be done within the application itself. This improves visualization and shows trends of some activities for the users. The analysis can be sent automatically to the users periodically. The export of the available and eventually filtered data to a spreadsheet, allow for sporadic necessary analysis.

This project will provide an application to centralize and govern all information of orders, application licenses and devices. Apart from the registration of this information, the project will provide the possibility to monitor the delivery time of orders and repairs. It will also be responsible for a data constancy check of the configuration of the mobile devices used in the G4S Login application.



# Planning

When we started this thesis project early September, the waterfall approach was the mandatory approach implemented by the school. The second half of the first trimester our promotor gave us the choice to step away from the cascading style approach and continue with an agile way to deliver this application. As a team, we decided to finish the started analyses and progress afterwards with an agile mindset.

## Waterfall Planning

Delimitation Project / Defining the problem / System requirements

Verbal description of the company

Definition scope September 2016

MoSCoW September 2016

Overview business functions

Function decomposition diagram September 2016

Situation of the project within business September 2016

Define the information flow

With the outside world September 2016

With other business features October 2016

Problem analysis of the current system

Problem list October 2016

Urgency October 2016

Possible solutions

October 2016

Target list for the new system

List of quality aspects October 2016

Event respond list October 2016

Context diagram October 2016

## Agile Planning

User Management	8/11/2016 - 6/12/2016
Manual Input Devices/Licenses/Tags	6/12/2016 - 10/01/2017
Import Process -> Csv from Login	6/12/2016 - 10/01/2017
Data Accuracy Check	10/01/2017 - 24/01/2017
Repair Process	24/01/2017-21/02/2017
Tracking Orders Process	24/01/2017-21/02/2017
Tracking Devices Process	21/02/2017 - 21/03/2017
Tracking Repairs Process	21/02/2017 - 21/03/2017
Export Data	21/03/2017 - 21/04/2017
Analysis of All Collected Data (Nice To Have)	21/04/2017 - 21/05/2017
Debugging & Deployment	21/04/2017 - 21/05/2017

## Sprints overview and evaluation

Developing the CSV import took longer than initially planned. Receiving the example CVS file was delayed and not in a valid format the first time. Also, converting the individual lines into Objects to pass through validation processes took longer than expected.

The sprints at the start of 2017 were delayed because of unforeseen health issues in the near family of one of the project members.

Overall the planning was designed to deal with inconvenient delays and included some overflow days for these situations. Even though planning was respected, the last weeks of the project were still quite busy.

## Analysis stage

### Product Vision

<b>Vision Statement:</b> Management and reporting of devices, orders and repairs			
<b>Target group</b>	<b>Needs</b>	<b>Product</b>	<b>Value</b>
Login Contact Center (LoginCC) Employees: Responsible for high level maintenance and development  Operational Coordinator (OC) Single Point Of Contact (SPOC) for clients, implement G4S Login on site to automate real time reporting  IT Management	Being able to follow up orders and repairs of G4S Login materials in real-time	Measuring of time spent in different processes  Follow up tool for state of repairs  Centralized tool for managing orders and materials  Provides data consistency of configuration settings for mobile devices	Reduces time spent on collecting information  Providing faster and correct information  Reduces data errors by centralizing and controlling data flows

Table I: Product Vision - Vision Statement

## MoSCoW analysis

### Must have

- Tracking orders regarding to G4S Login
- Tracking repairs of mobile devices used to report in G4S Login
- Check the accuracy of data that has been exported from G4SLogin
- Timestamps of all transactions should be recorded automatically
- Exportable data (csv, excel)
- Each user can only consult data that belongs to his operational section

### Should have

- Consulting the history of state changes
- Consulting specific orders
- Push notifications when new orders are registered or when extra parts need to be ordered
- A dashboard to view all orders & repairs in each state
- User management (defining different roles for a user)
- Multilanguage interface for end users
- When data is imported, anomalies should be recognized automatically

### Could have

- Integrated analysis tools of the collected data
- Automatic notification if a rented device is cancelled before the end of the prepaid period

### Won't have

- Integrated invoice calculation
- Review of pricing and management of actual prices

## Problem statement matrix

Problem or opportunity	Urgency	Visibility	Priority	Proposed solution
Several systems must be queried to get all information about a certain device which can be time consuming	High	High	1	New development of centralized system
Adding and updating data is complex and takes a lot of time because it must be entered in various systems	ASAP	Medium	1	New development
OC's can't view the state of their order or repair directly and are required to email LoginCC for more information on the subject	Medium	High	3	Manageable users and roles
There is not enough data to extract Key Performance Indicators (KPI) on processing time of a repair	Low	High	4	Development of Analysis module
Analysis is difficult because data is spread out and can be inconsistent or incomplete	High	Medium	2	Development of exports
Efficiency of LoginCC is hard to measure because time is not taken into account in current systems	High	Medium	1	Include timestamps on every transaction
Cancelled prepaid rent licenses are hard to find while they can still be used for other short term projects	Low	Low	3	Development of notifications module
Notification of decreasing stock can help to order new parts in time in order to speed up replacement	Low	Low	3	Development of notifications module
Wrong decisions can be made because of inconsistent data	ASAP	Medium	2	New development
History of data could help in tracing back issues with a device or license	High	Low	4	Develop auditing module
There is a lack of access to management and decision making information	High	High	4	Development of Analysis module
Data inconsistency can exist now because of the use of several parallel systems. These inconsistencies must be detectable in the future to avoid wrong decisions	ASAP	Medium	2	Development of data imports and notifications
Access to information must be secured	High	High	2	Use secure connections and authenticating

Table II: Problem statement matrix

## PIECES analysis

PIECES framework for problem, opportunity and directive identification.

### *Performance*

- Problems and analysis:
  - Too much time is spent on administration of orders and repairs.
  - Collecting information in several systems is time consuming.
  - Adding and updating data is complex and takes a lot of time.
- Analysis:
  - Collecting information in several systems is time consuming.
  - Adding and updating data is complex and takes a lot of time.
- Causes:
  - System is not centralized.
  - Cluttering of the system grows due to extra needs.
- Possible solutions:
  - Hire more employees which will improve throughput.
  - By centralizing information and by automating in- and output, performance can be increased dramatically.

### *Information Output*

- Problems and analysis:
  - Information must be created by consulting several data sources.
  - Data is not directly available for everyone who needs it. At the moment the information request is sent by mail to LoginCC.
  - Analysis is not possible because data is spread out and incomplete.
- Causes:
  - Information is written in different locations.
  - Data about delivery dates are not recorded.
- Solutions:
  - Centralize all data in one system.
  - Record timestamps of each action.

### *Information Input*

- Problem and analysis:
  - Input of data in several files.
  - Data is inconsistent due to several parallel systems.
- Cause:
  - There is no validation on data fields.
  - Through time several systems were started up independently.
- Solution:
  - Centralize data with validation conditions.

### *Information of Stored data*

- Problems and analysis:
  - Stored data is not accurate.
- Cause:
  - Data is stored redundantly in multiple files or databases.
  - Through time, several systems were started up running independently from each other.
- Solution:
  - Create one centralized data system.

### *Economics*

- Problems and analysis:
  - Labor costs are unknown and not measurable.
  - Cancelled orders within the first year of prepaid rent are wasted.
  - No possibility to easily detect unused prepaid time.
- Cause:
  - No timestamps are registered for actions in the current system.
  - Calculations of prepaid period are complex and must be done manually and are therefore often skipped.
- Solution:
  - Register timestamps.
  - Provide automatic warnings for not using prepaid time.
  - Hire more employees.

### *Control*

- Problems and analysis:
  - Low stock of spare devices.
  - Data is not consistent over several systems.
  - Therefore, wrong decisions can be made.
  - Inputted data can be wrong.
  - No possibility to trace back data.
  - Quantity of spare devices isn't monitored.
- Cause:
  - No backup of data.
  - No validation of input.
  - No notifications.
- Solutions:
  - Provide data validation.
  - Provide automatic warnings.
  - Through auditing tables, data can be restored.



## Efficiency

- Problems and analysis:
  - Time spent on administration consumes available time for added value tasks.
  - OC's can't access data directly and lose time waiting on the current process.
  - Through spread out data, information isn't easily collected and isn't available for those who need it.
- Cause:
  - No centralized system available to query data.
- Solutions:
  - Reduce administration time by using a central system.
  - Give OC's and [powerusers](#) direct access to information they need.

## Service

- Problems and analysis:
  - State of repair and order is unknown.
  - Current system is cluttered by several files.
  - OC's have to wait for feedback on information of LoginCC they can't consult this themselves in real-time.
  - The possibility exists that inaccurate information is provided.
- Cause:
  - Current data isn't accessible for everyone.
  - Current data isn't consistent due to several parallel systems.
- Solution:
  - Create limited access of data for OC's through a centralized system.

## Risk management

Risk	Chance	Cause	Consequence	Class	Solution
Serious health issues of a team member	1	Health	Project slows down	2	<a href="#">User stories</a> will get reprioritised, other members take over
Lost files	2	Hardware failure or theft	Possible rework	1	Put in place frequent backups + cloud storage
Co-worker timing issues	3	Scheduling	Possible time delays	1	Use of online communication tools
Scope increases during project	2	Underestimation of initial scope	Possible unfinished items	2	Have initial scope signed and agree upon scope expansion
Project cancellation	1	Peter losing his job or IT restructuring	Fatal	4	-
Stakeholder switches position within G4S	2	New lucrative vacancy for stakeholder	Interest might be lost in project	3	Have backup stakeholder ready
G4S loses interest in project	1	-	Project might be unfinished	3	-
Deployment issues	3	Licencing	Project slows down	2	Provide enough time in planning for deployment. Use backup deployment plan external of G4S
Deployment environment changes	1	-	Project slows down, possible rewriting of existing code	3	Use backup deployment plan external of G4S
TrackForce goes bankrupt	1	-	Project stopped	4	-
G4S ends contract with TrackForce (Login) supplier	1	Enforced by global G4S group	Project stopped	4	-

Family member requires medical attention	3	-	Less time available for group work	2	Provide dedicated time slots in project planning to allow overflow. Possible nice-to-have features would be dropped
All types of devices, used by G4S Login, change	1	New profitable devices on the market which comply to the needs of Trackforce and G4S	Project slows because of new configuration	2	Provide flexible device structure within application

Table III: Risk management

Chance class	Percentage	Description
1	0-5%	Unlikely
2	5-20%	Possible
3	25-50%	Probable
4	50-100%	Nearly certain

Table IV Risk Management - Legend 1

Consequence class	Time	Quality
1	< 2 week delay	Not noticeable decrease
2	< month delay	Noticeable decrease
3	< 2 months delay	Decreased, possible bugs
4	> 2 months delay	Major quality loss, unfinished project

Table V: Risk management - Legend 2

## Project scope

The application will exist out of 5 parts:

1. Tracking orders regarding to G4SLogin
2. Tracking repairs of Mobile devices used to report in G4SLogin (spare devices included)
3. Check the accuracy of data that has been exported from G4SLogin. This data includes the configuration and properties of the G4SLogin Mobile devices.
4. Analysis of all collected data. (this point is nice to have and will only be created if there is room left in the project)
5. Configuration of users and accesses

### *Tracking orders regarding to G4SLogin*

- It should be easy to manually insert new [purchase orders \(PO\)](#).
- For each order, it needs to be possible to track if this order is ... by timestamp:
  - Not yet delivered
  - Delivered to LoginCC but the configuration isn't finished
  - All extra parts (like sim cards) are/ are not yet ordered.
  - The order has been configured and shipped by internal G4S delivery
  - The order has arrived at the OC and is ready to be used in operations
- These timestamps should be recorded automatically if the state is changed.
- It should be possible to consult these timestamps
- It should be possible to look up a specific order.
- For each order the following details should be possible to consult:
  - Which device(s) is/are active for this order?
  - Who created this order?
  - For which site/client this order is made. (beware, one PO can contain multiple clients, so this information should be collected for each ordered item on this PO)
  - Who the Operational Coordinator for each item is.
  - For each device, the International Mobile Equipment Identity ([IMEI](#)) N°, call N° and sim N° should be registered.
- The application should notify if a new order has been registered or if extra parts need to be ordered to complete it (e.g. sim cards). These notifications are triggered by conditions that are coupled on device types.
- There should be a dashboard available with an overview of all orders in each state.
- The created data of orders should be exportable (e.g. csv, excel) so it can be analyzed.

### *Tracking repairs of mobile devices used to report in G4SLogin*

- The application should provide a dashboard with an overview of the number of items for each repair state:
  - Number of malfunctioning devices on their way to LoginCC
  - Number of malfunctioning devices arrived at LoginCC waiting to be analyzed
  - Number of malfunctioning devices analyzed but not yet repaired or replaced
  - Number of repaired or replaced devices waiting to be shipped to the OC
  - Number of devices on route to the OC
  - Number of malfunctioning and non-repairable devices LoginCC has at that time
  - Number of devices send to the supplier, and waiting to be replaced
  - Number of spare devices
- This dashboard should have a filter, to filter out by type of device and/or state.
- It should be possible for an OC to look up the devices and their state used on the sites he is responsible for.
- It should be possible to register one or several of these devices as malfunctioning and on route to LoginCC. This registration should capture at least a description of the malfunction.
- LoginCC should be able to import or change the state of one or several of these devices. If the state is changed, a comment field should be available for each action.
- LoginCC should be able to manage which OC can see which device, and when a device is replaced this should be adapted automatically.
- When a device is replaced the PO of the malfunctioning device should be updated with the info of the new device automatically.
- When a device is replaced, the data collected from the malfunctioning device (e.g. sim N°, call N°) should be transferred automatically to the new device.

### *Check the accuracy of data*

- The stakeholder will provide a way to export the configuration data of the mobile devices, which is available in G4SLogin. This data can be exported daily or on demand. The export will arrive in a csv format or into a web service. The export will contain all needed data to perform the consistency check which is needed for this project.
- Devices that aren't in the export but are logged in to the application will be listed in a list "to be treated devices".
  - Each item on this list needs to be treated manually by a LoginCC employee, the state of the device needs to be updated.
  - This device needs to be automatically removed from the PO it was affected to. (proposition: on the overview of the PO the removed device is still listed but greyed out and uneditable until replaced by another device)
- Because PO's can become incomplete after this data import there should be an overview available for each PO that isn't complete anymore.
- Devices that aren't logged in the application but do appear in the data import should be added automatically. If the data of the import is incomplete this device should afterwards be dropped in the list "to be treated devices". So, after manual intervention all necessary data is completed.
- In the list "to be treated devices" it should be possible to filter between all devices, old devices or new devices.
- If the data import shows incomplete data of known devices these should be listed in a list to update device in G4SLogin. The missing data should be highlighted. So, a LoginCC employee can easily update the configuration of these devices directly in G4SLogin.

### *Analysis of all collected data*

- If the project has room for it integrated analyses will be provided.
- It is clear that all data should be able to be exported in csv or excel format, within the scope.
- The exports that should be in place are exports of the same structure of the DB tables but filtered by a certain period.

### *Configuration of users and accesses*

- Adding, removing and editing a user profile should be available for admin users
- Copying an existing user to create a new user should be possible
- Managing accesses to different menus for a user should be available for admin users
- Managing accesses to different devices and/PO's should be available for admin users
- Copying the full or partial device list from a user to give access to the selected devices to another user should be available for admin users.

## Information Flow Diagram

Since the stakeholder doesn't allow us to create interfaces with other existing software, we import data through a csv file. Only information about mobile devices is imported. Before this data is stored in the data base, it will be checked by a validation process. This loose coupling from G4S Login information to G4SOLMan is described in detail in 'Loosely coupled data synchronization flow'.

All information imported and created in G4SOLDMan can be exported in to a CSV file. In the G4STableDirective a button linked to the export function is added. This way every table we present has this option.

Users with the correct rights have access to add, edit or even delete data and so changing the information gathered in the database. We do speak about deleting data, but we should rather say soft deleting. Every object in the database has a property soft delete. When this is active the data won't be presented to the users, but still exists. An administrator has the right to undo this soft delete property. By doing this, he restores the soft deleted data.

By using tags that are coupled on the states of objects we can present the user with dashboards.

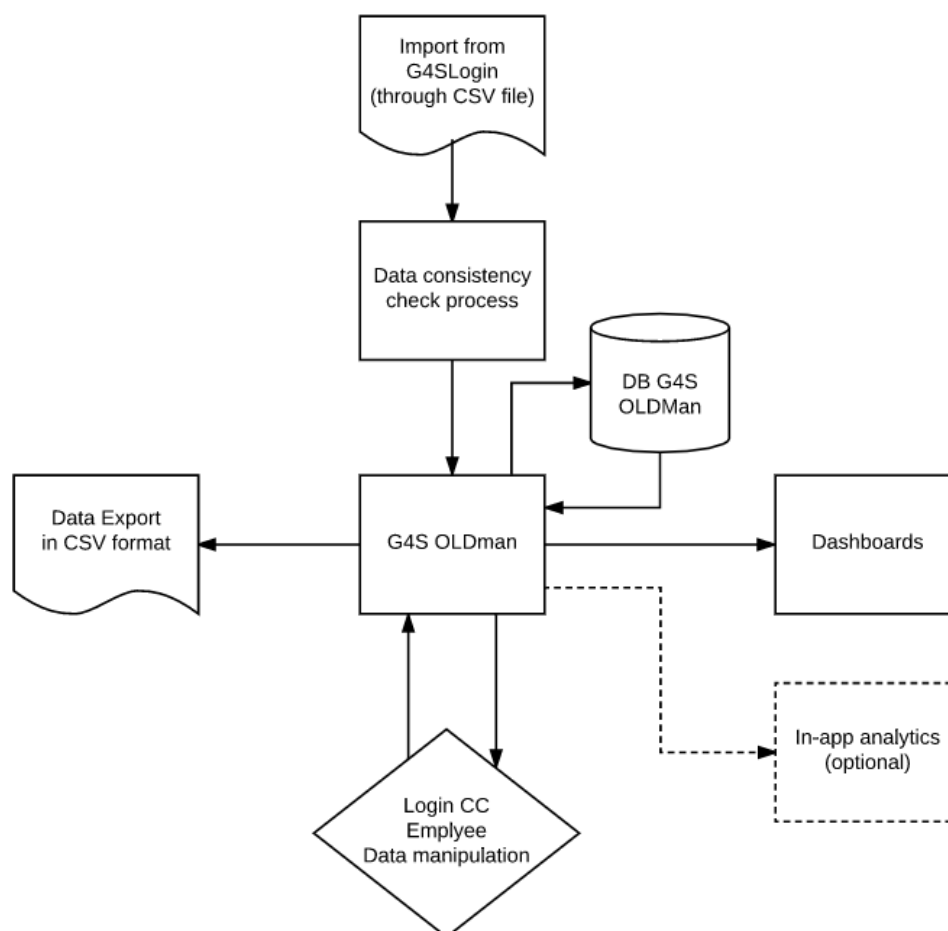


Figure III: Information Flow Diagram

## Event response list

Event	Trigger	Source	Use case	Response	Destination
LoginCC employee creates new Purchase Order	New PO	LoginCC employee	New Order	Input form for new PO	G4S OLDMan & OC
LoginCC employee adds items to a PO	New PO	LoginCC employee	New Order	Input form for new PO item	G4S OLDMan & OC
LoginCC employee updates an item PO	Delivery of materials	LoginCC employee	Delivery of items	Change status of PO items & updates device object	G4S OLDMan & OC
LoginCC employee closes an item on a PO	Termination of leasing	LoginCC employee	Close Order Line	Closes the PO item line	G4S OLDMan & OC
LoginCC employee closes a PO	All leasings are terminated	LoginCC employee	Close Order Line	Closes the PO	G4S OLDMan & OC
LoginCC employee creates a sim card	Delivery of devices	LoginCC employee	Delivery of items	Input sim details	G4S OLDMan & OC
LoginCC employee updates a device	Configuration of new device	LoginCC employee	Delivery of items	Input device details	G4S OLDMan & OC
LoginCC employee updates a device	Shipment of device to operations	LoginCC employee	Delivery of items	update device status	G4S OLDMan & OC
OC creates a repair note	Defect device	OC	Repair Device	Creation of repair note	G4S OLDMan & OC
Defect device arrives @ LoginCC	Arrival of defect device	LoginCC employee	Repair Device	update device status	G4S OLDMan & OC
Analysis of defect device	Arrival of defect device	LoginCC employee	Repair Device	update device status	G4S OLDMan & OC
Repair of defect device	Defect device analysed	LoginCC employee	Repair Device	update device status	G4S OLDMan & OC



Replacement of devect device	Defect device analyzed as broken	LoginCC employee	Repair Device	update device satus	G4S OLDMan & OC
Shipment of defect device to supplier	Defect device analyzed as broken	LoginCC employee	Repair Device	update device satus	G4S OLDMan & OC
Arrival of spare device @ LoginCC	Delivery of spare materials	LoginCC employee	Delivery of items	Create new spare device	G4S OLDMan & OC
Consultation of the status of all PO's	Analyse request	G4S OLDMan	Overview PO's	Show dashboard PO's	LoginCC employee
Consultation of the status of all Devices	Analyse request	G4S OLDMan	Overview PO's	Show dashboard Devices	LoginCC employee
New device imported	Upload data Web Service/CSV	G4S LoginCC	Import actual data from G4S Login	Input new device and list it up in "to be treated devices" list	G4S OLDMan
Device data changed	Upload data Web Service/CSV	G4S LoginCC	Import actual data from G4S Login	Update new device and list it up in "to be treated devices" list	G4S OLDMan
Existing device not available in import list	Upload data Web Service/CSV	G4S LoginCC	Import actual data from G4S Login	list it up in "to be treated devices" list	G4S OLDMan
Exporting data	demand for data export	G4S OLDMan	export data	Export to csv the requested data	LoginCC employee

Table VI: Event Response List

## Non-functional requirements

### *Configuration management*

The configuration file of the application contains the database connection string. This way the application can be ported in to multiple deployments by just adapting this connection string. Also, our code first approach of the entity framework avoids the need of a complete database configuration. Once the application is deployed and it connects to the database for the first time, it will create the tables, headers and constraints for you.

The functional configuration is made this way so that almost everything is dynamically configurable. One of the few exceptions here are the user rights to access certain views. These are hard coded for each view. To configure certain levels of access, the application provides the possibility to create user rights groups. An account has access to all views that are assembled in the group it has access to.

Even the translation module is made to avoid maintenance and configuration work. By adding a new language through the functional configuration, the application will automatically populate the translation table with the missing fields it encounters when visiting a view for the first time. This way only the visited views in that language need to be translated. Besides this approach, a translation toggle button is set in place. When this “not-translated-mode” is toggled, the key words are visible. This way if the same view has multiple translations with the value “detail”, and you want to change one of these to “details” you can find the exact keyword for this word.

### *Dependency on other parties*

The stakeholder didn't want us to interface this application directly with other applications. If support or development costs are raising too high, they should have the possibility to stop this application without it having a big impact on operations.

Because of this condition the dependency of other parties is minimal. From the G4SLogin application, data is exported through a csv or other Excel format file into this application. This is the only dependency we have. The format and type of this file are the only agreements we have with Trackforce.

The database server and application server of G4S are over dimensioned for this scope and can provide the needed platform to deploy on. The access to these servers is no issue since one of the team members of this thesis is a member of the G4S ICT-service desk team.

### *Deployment*

The stakeholder provided us with full access to a set up SQL database. Once we are connected to the internal network of G4S we can configure the DB ourselves.

On an [IIS](#) server, the stakeholder created a website named G4SOLDMAN. We didn't get full access to the server, but IT provided us with a console tool to configure this website. If we need to deploy software or save libraries to the server directly we are assisted by the network engineer who is responsible for all G4S networks in Belgium.

### *Maintainability*

This application doesn't use self-created frameworks. All frameworks that are used are obtained by the “nuget package manager” of visual studio. (for example: Entity Framework, Angular-JS, Unity ...). This way, new development or maintenance of the application isn't limited to specific developer skills.

The application is constructed out of several different layers:

- Database layer
- Business layer
- Application layer

Thanks to this layer approach and the use of several different services, it allows to minimize the development and maintenance of the application.

### *Network topology*

The application is only available for users that are connected on the G4S virtual private network (VPN) connection. This way we could avoid a load of network issues and validations.

### *Performance / response time*

This website is a single page application. Each page will make his own Ajax (Asynchronous JavaScript and Extensible Markup Language (XML) calls to the back-end. The number of these calls can rise in volume depending on the contents of the page, but every call demands low volume packets. This way a fast response time is ensured.

### *Platform compatibility*

The website is built on Angular-JS version 1.3. This version has dropped support for Internet Explore version 10 and higher, Firefox, Safari but it is created and tested for the Chrome browser. This browser is used as the main browser for G4S worldwide.

### *Reporting*

The web site provides some dashboard views to get a fast display of pre-defined volumes of data in different states. All the raw data is accessible directly on the SQL database. In the application, all tables presented to the user, are exportable in to a csv format.

### *Scalability (horizontal, vertical)*

Functional scalability is provided. For example, device types and repair reasons are configurable by admin user. If the device landscape in G4S Login changes, the device types can be added.

Non-functional scalability is assured by only using .NET frameworks and compliance with representational state transfer ([REST](#)) guidelines.

### *Security*

The security is limited to username and password for each account. When logging in, a JavaScript object notation web token ([JWT](#)-token) is generated to confirm the communication between the front- and back-end.

The security is limited because the application is only reachable trough the VPN of G4S. This VPN connection handles all other security issues.

## Loosely coupled data synchronization flow

Since interfacing between G4SOLDman and other software is not allowed by the stakeholder, we implemented data exchange through a csv file.

The G4S Login application will export the data needed from the mobile devices into a csv format with a predefined structure. This file is imported in G4SOLDMan. Once a csv is imported, it will turn a validation process for each line of the csv. If the data is valid, it will be stored or updated. When the data is incomplete or indicated as corrupt by the validation, this data will be stored into a ToBoTreated list in the application. By manual intervention of this list, the data can arrive in the production site. If new imports complete the incomplete data or correct the corrupt data, the remaining of that object in “to be treated data” will be removed automatically.

By consulting the operational data in G4SOLDMan a manual intervention in G4S Login can occur. This way both applications are loose coupled.

## Analyze Database

Because we started with a completely new SQL database we have chosen to work with Entity Framework using a Code First approach. This way of working allowed us to adapt the database tables during the development phase if needed. The build in migrations system of Entity Framework assisted us to adapt the database along with the code by using predefined conventions and attributes throughout the codebase.

### Entity Framework code setup

Entity Framework works by encapsulating plain old [CLR](#) objects (POCOs) in a DbSet property in a predefined configuration class. It is possible to make a new unique class for every entity needed in an application.

However, we chose to use a shared base class for all our entities called EntityBase.

Every entity inherits from this class giving it a private key, auditing properties, and a property indicating if the entity has been logically removed from the database or not. An example of this class can be found below

```
public class EntityBase
{
    [Key]
    [Column(Order = 0)]
    public int Id { get; set; }

    public bool SoftDelete { get; set; }

    public System.DateTimeOffset? DeletedAtUtc { get; set; }

    public System.DateTimeOffset CreatedAtUtc { get; set; }
}
```

Having every entity inheriting EntityBase allows us to use a generic approach for all services that use entities in our application such as our readers, writers and validators

```
Public interface IValidator<Tentity> where Tentity : EntityBase
```

A base class of all these services exists, containing a generic implementation, which is valid for every type of entity, for instance a ValidatorBase class would only check if the Id exists when updating a record or if the SoftDelete property is toggled correctly in case of an insert or update. If specific business rules are needed for a particular entity the ValidatorBase class would be extended and implemented specifically for that entity, only the specific methods that require changes would be overridden with custom entity-specific code. The Unity container implemented in our application ties the custom implementation to the generic interface in a centralized container configuration class, this ensures dependent classes not having to know if a certain validator was implemented specifically or not.

Since the entities MobileDevice, License and ToBeTreatedDevice have common code, we made a parent class ItemBase. The parent class of this ItemBase class inherits the EntityBase class and extends it with properties all item entities share. This way these entities still comply for our generic approach

## ERD Diagram

Below is a full diagram of the G4SOLDMan database. Because the database contains 35 tables we have split the diagram in to smaller sections which are discussed in detail afterwards.

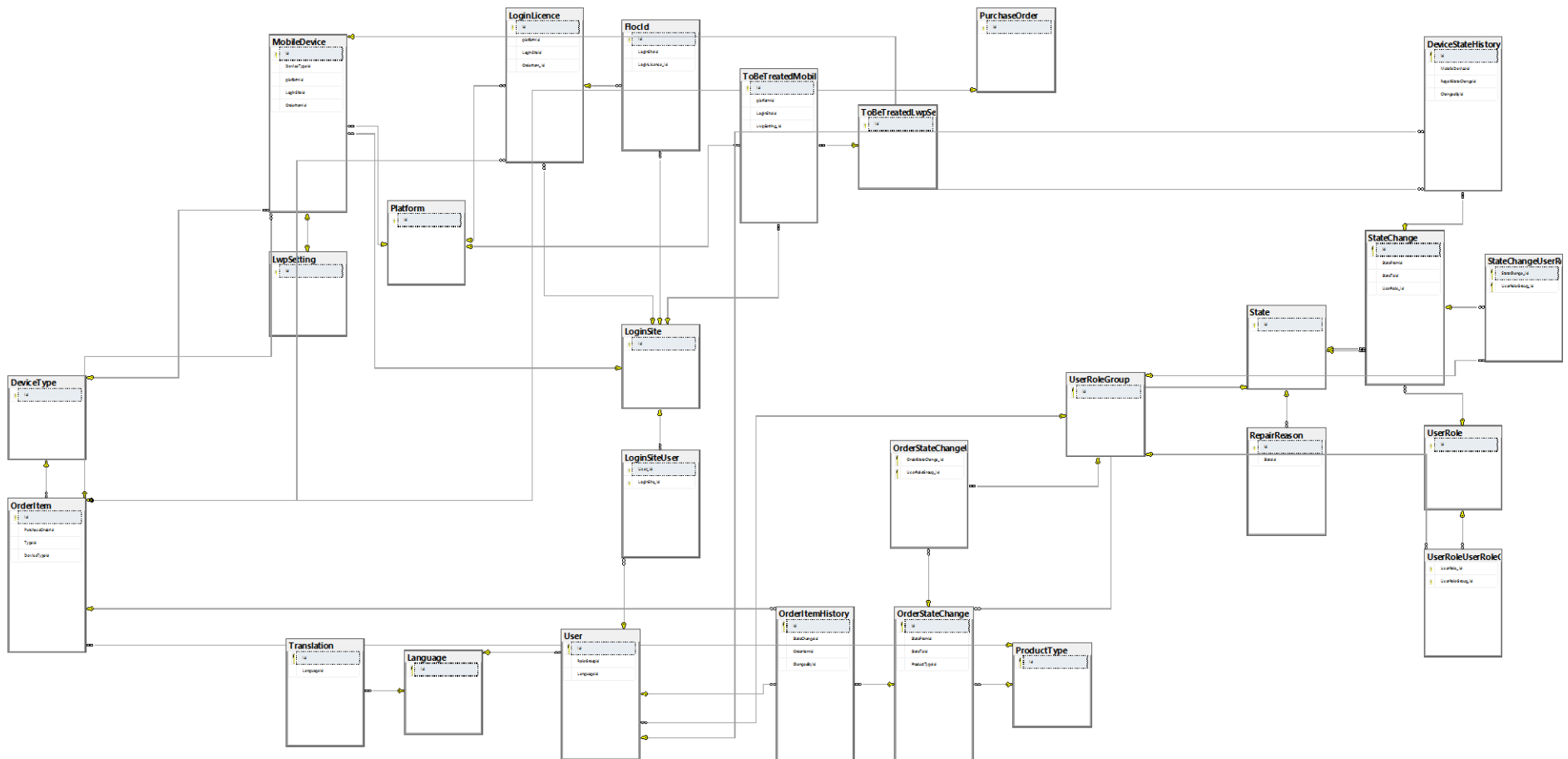


Figure IV: ERD Diagram

## User management

Users are a central element in the G4SOLDMan application. They are stored in the AspNetUsers table as well as the User table. The first will save information needed for signing in and role validation and is maintained by a separate connection managed by ASP.NET Identity Framework, this is the only place a password is handled. There are no references to the other entities we created to drive the application itself. A user is authenticated and authorized using the AspNet tables and is later matched to a User in our own table.

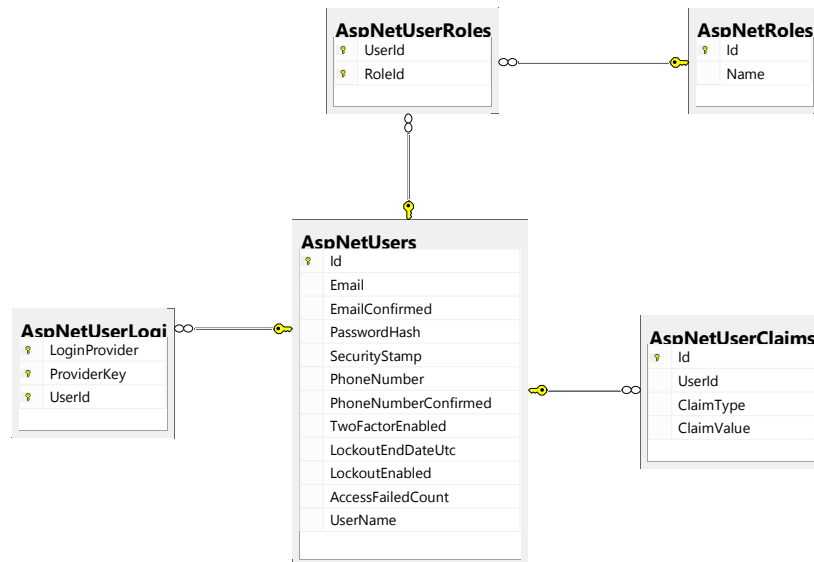


Figure V: ERD Diagram – AspNet user management

The User table contains all properties seen in the user detail screens of the application except the password. In the diagram below we see the User table and its references to others. The Translation and Language table are required to make our application multilingual and future proof as more languages can be added and translations are managed by a user with elevated rights.

Users link to a set of predefined UserRoles through an optional UserRoleGroup entity. A user can only be a member of one UserRoleGroup which contains one or more UserRoles. This in combination with the zero or more reference to LoginSite creates a database structure suited to make sure the right person gets to see only the relevant or allowed data, while ensuring a controlled set of rights to manipulate the data.

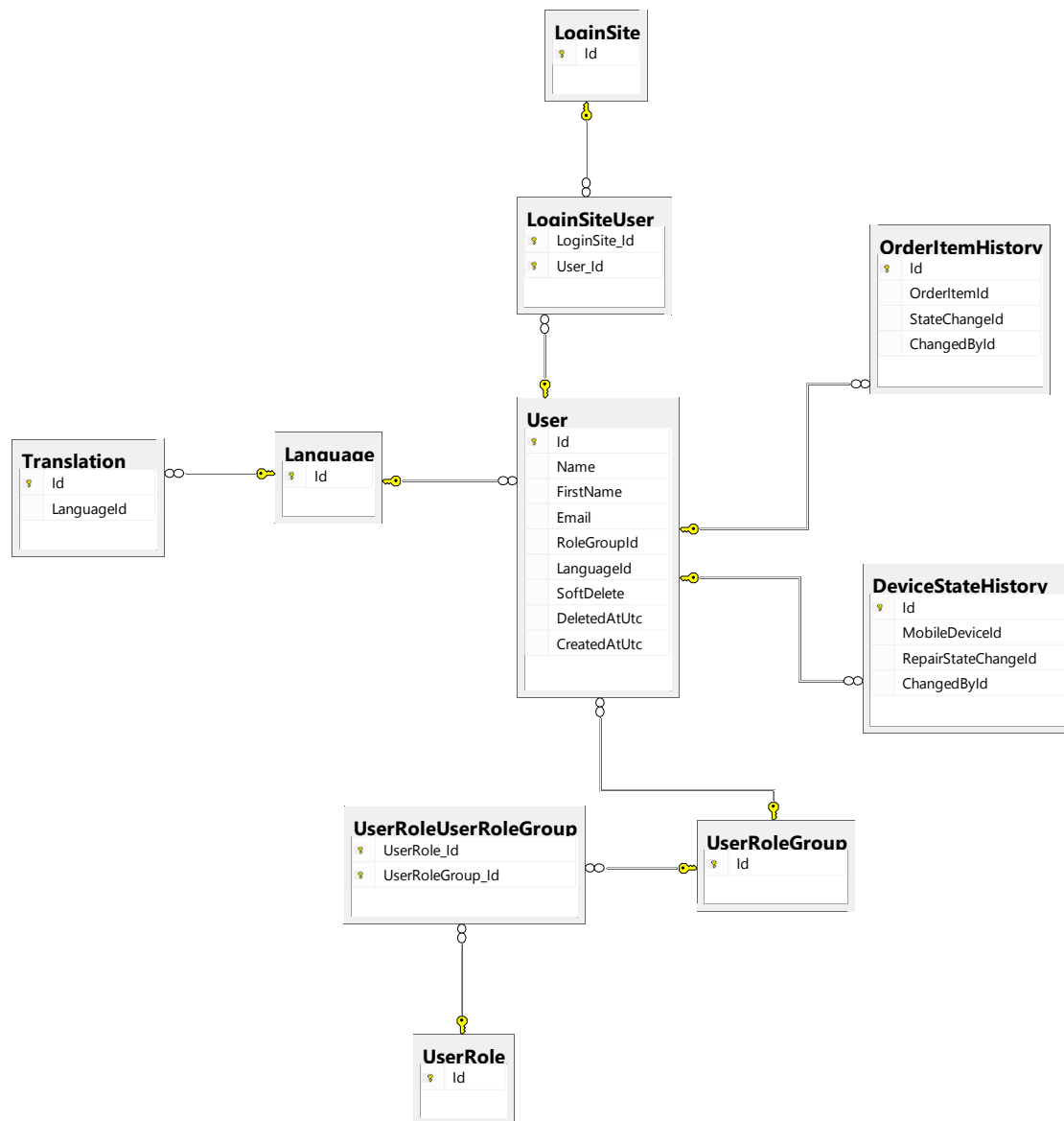


Figure VI: ERD Diagram – User management

In the diagram shown above we see a reference to the OrderItemHistory table and DeviceStateHistory table. These tables are described in detail in the State section. The reference exists to track the user who initiated a change in an order or device at a given time.



## OrderItems

A PurchaseOrder table exists to hold data matching purchase order information from the [SAP](#) system implemented at G4S. It contains a collection of OrderItems with details about the order that was made. The OrderItem table references a user manageable DeviceType and ProductType table to define what kind of OrderItem it is. Validation in software will determine if the OrderItem should be linked to a collection of MobileDevices, a collection of LoginLicenses, or neither. The OrderItem contains an optional list of OrderItemHistory records to track the order and delivery process based on the settings of the ProductType.

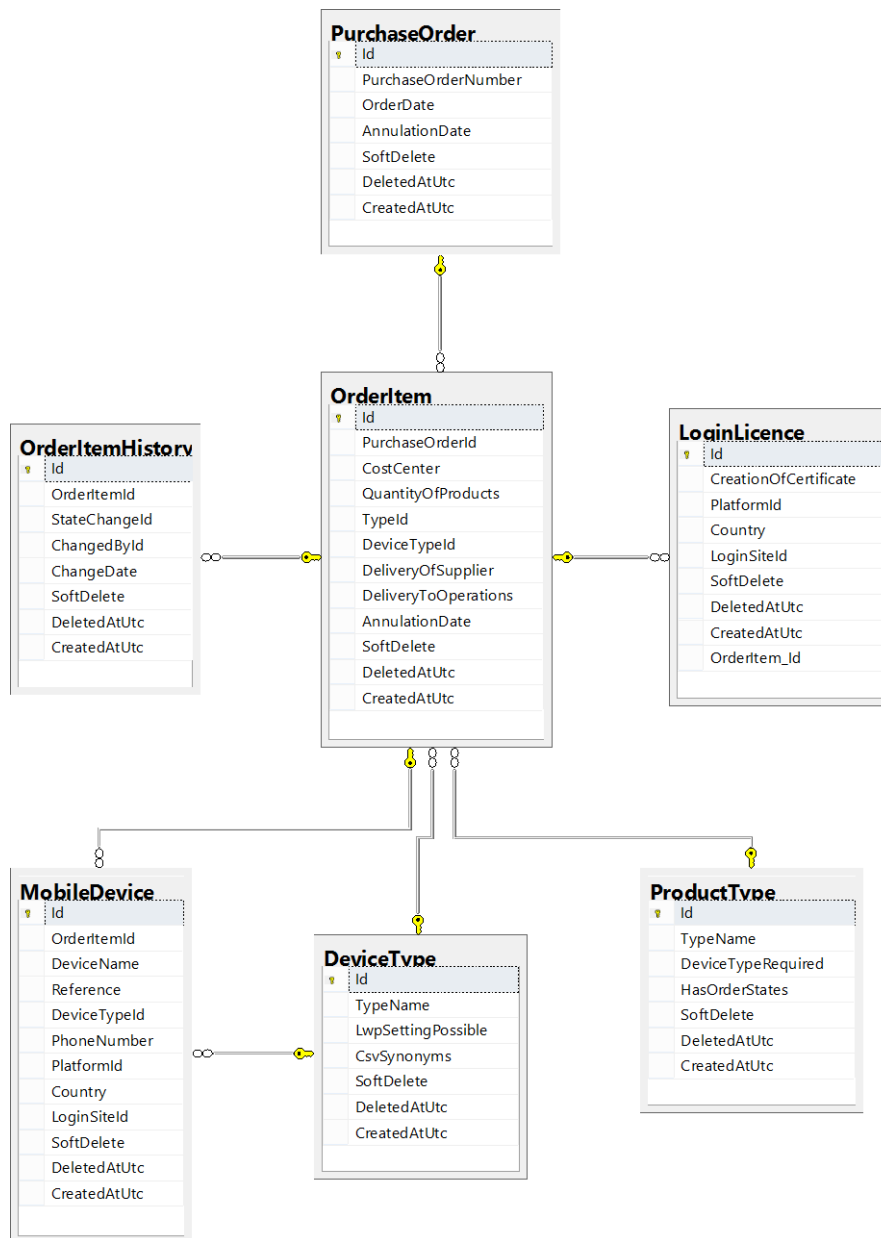


Figure VII: ERD Diagram – PurchaseOrder - OrderItem

## LoginLicenses

A LoginLicense is an entity derived from ItemBase containing a necessary Platform and LoginSite. It can contain an optional collection of [Floclds](#), but will mostly contain maximum 1 Flocld. The reference To LoginSite is required to filter the LoginLicense for a certain user, as not every user is allowed to query data related to any LoginSite. LoginLicense also has an optional relation to OrderItem, which creates a reference to PurchaseOrder as well.

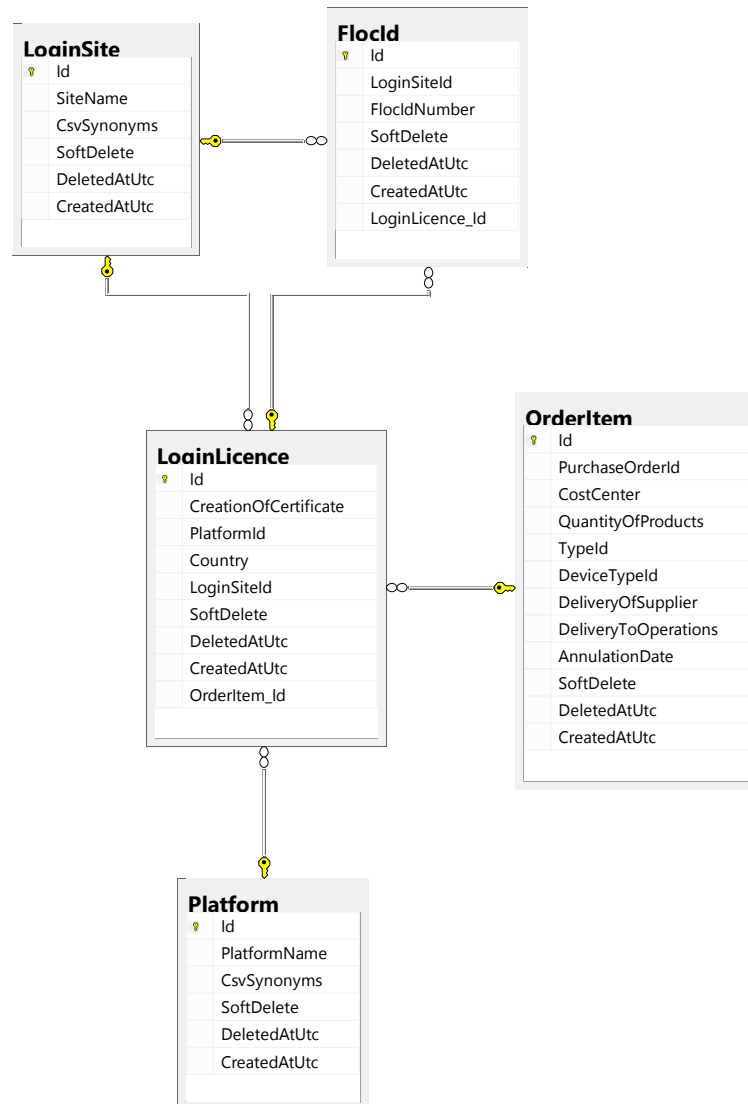


Figure VIII: ERD Diagram - Login Licenses

## MobileDevices

Like LoginLicense, A MobileDevice is an entity derived from ItemBase containing a necessary Platform and LoginSite. These are used for filtering data depending on the user querying the data. Also like LoginLicense, a MobileDevice links to an optional OrderItem to create a Reference to PurchaseOrder. A DeviceType is a required reference which is used in the application to determine user definable properties such as the need for LwpSetting for a certain type of device. Allowing new DeviceTypes to be created results in a flexible application which will be futureproof.

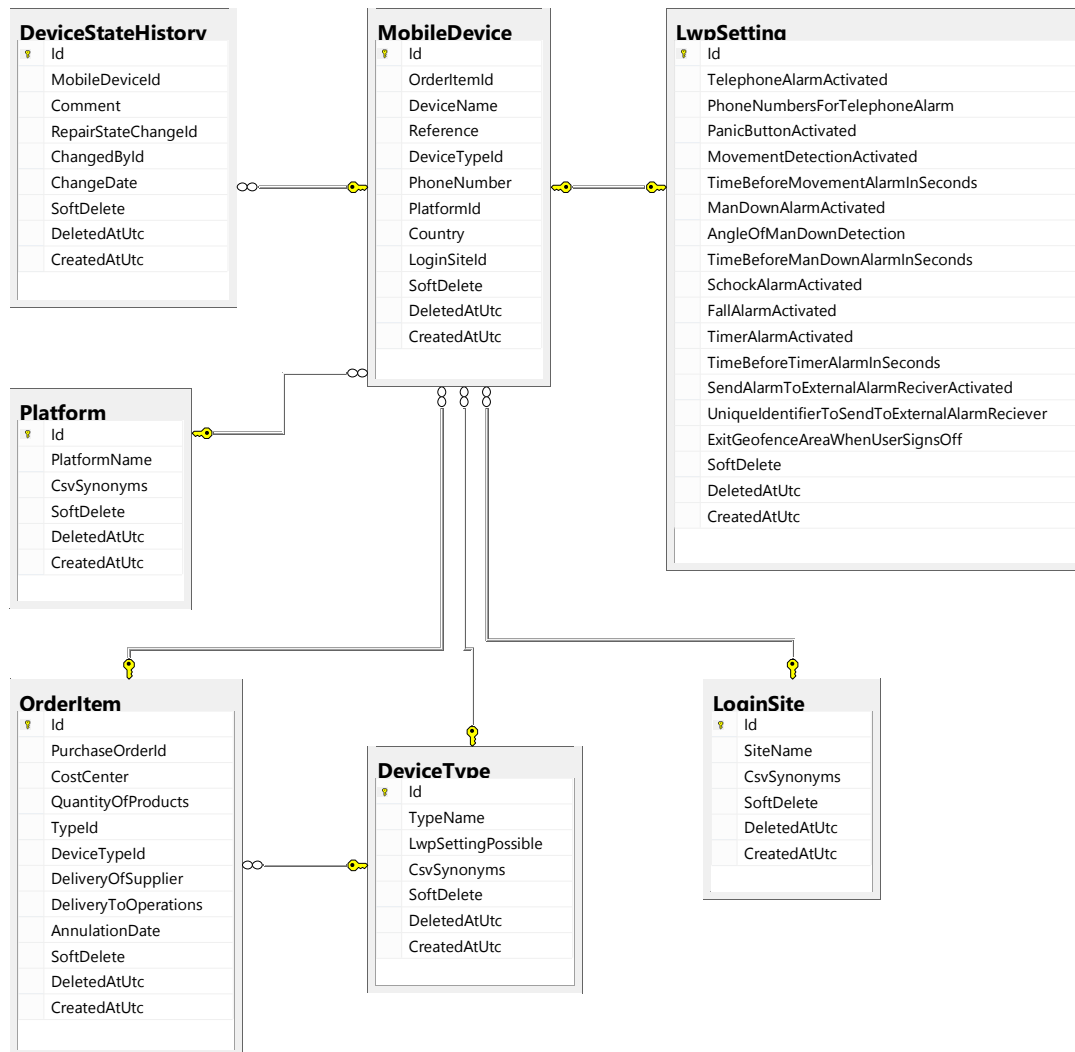


Figure IX: ERD Diagram - Mobile Devices

MobileDevice links with a one-to-one reference to LwpSetting, an entity which contains settings to protect users of the devices itself. The main goal of storing LwpSettings in the database is to create warnings and automatic state changes when an import from G4S Login changes values known in our application. These warnings would indicate a change made in the field to some devices and requires manual intervention.

## ToBeTreated Devices

ToBeTreated is a special kind of MobileDevice for imported items whom are failing validation tests because of changes in the data which are not allowed by rules, set in the G4SOLDMan application.

These tables are a copy of the MobileDevice structure adding only a ValidationWarning collection to both the ToBeTreatedMobileDevice and ToBeTreatedLwpSetting.

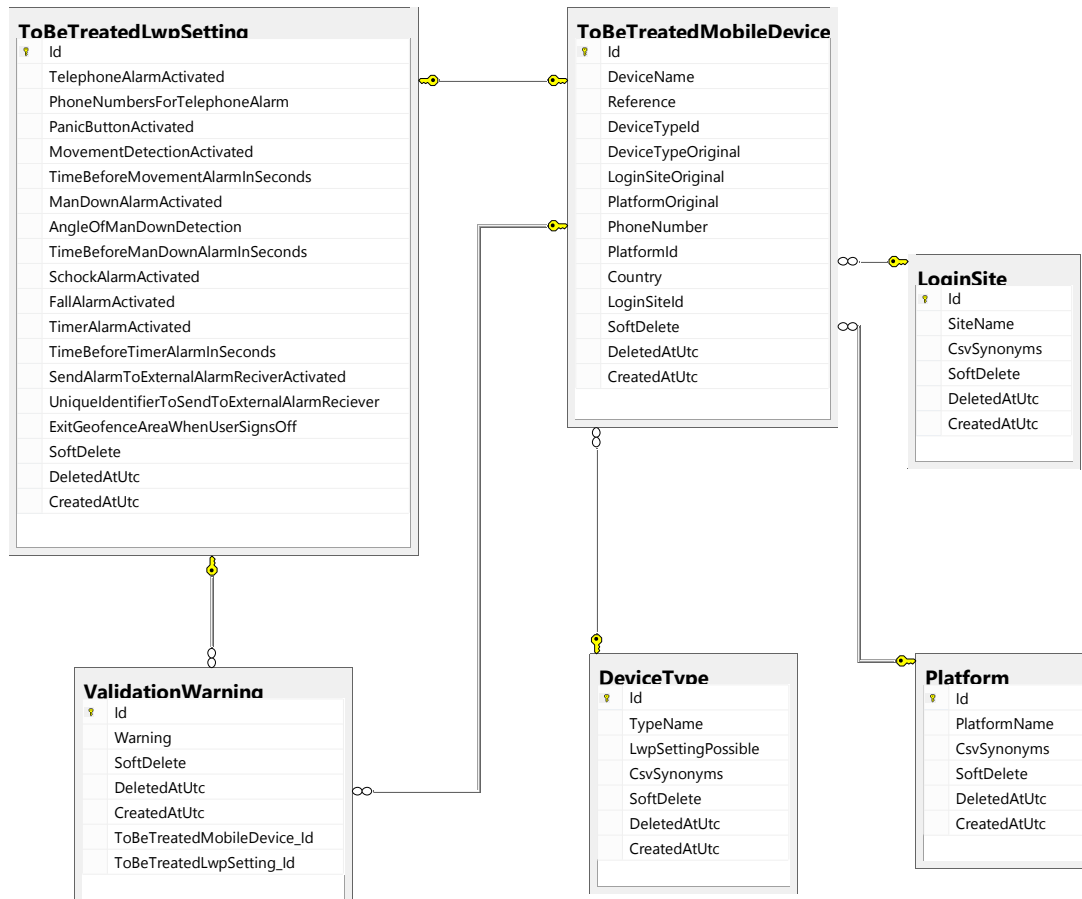


Figure X: ERD Diagram - ToBeTreated Devices

## States

Last but not least, the State mechanism takes up a large number of tables to allow for a state changing system which is user configurable and flexible at the same time.

The central entity in this is the State entity containing a name and description. It links to a StateKind table to determine if the state is for an OrderItem or a MobileDevice. A RepairReason collection is referenced from State to allow a predefined set of reasons to be given if needed when the state of an object is changed.

States have a double reference to StateChange to allow for a start and end state for a given change. This is user definable and allows for a device to travel through different states depending on the configuration made by our user. The same goes for the OrderStateChange table with as only difference it being used only for OrderItems.

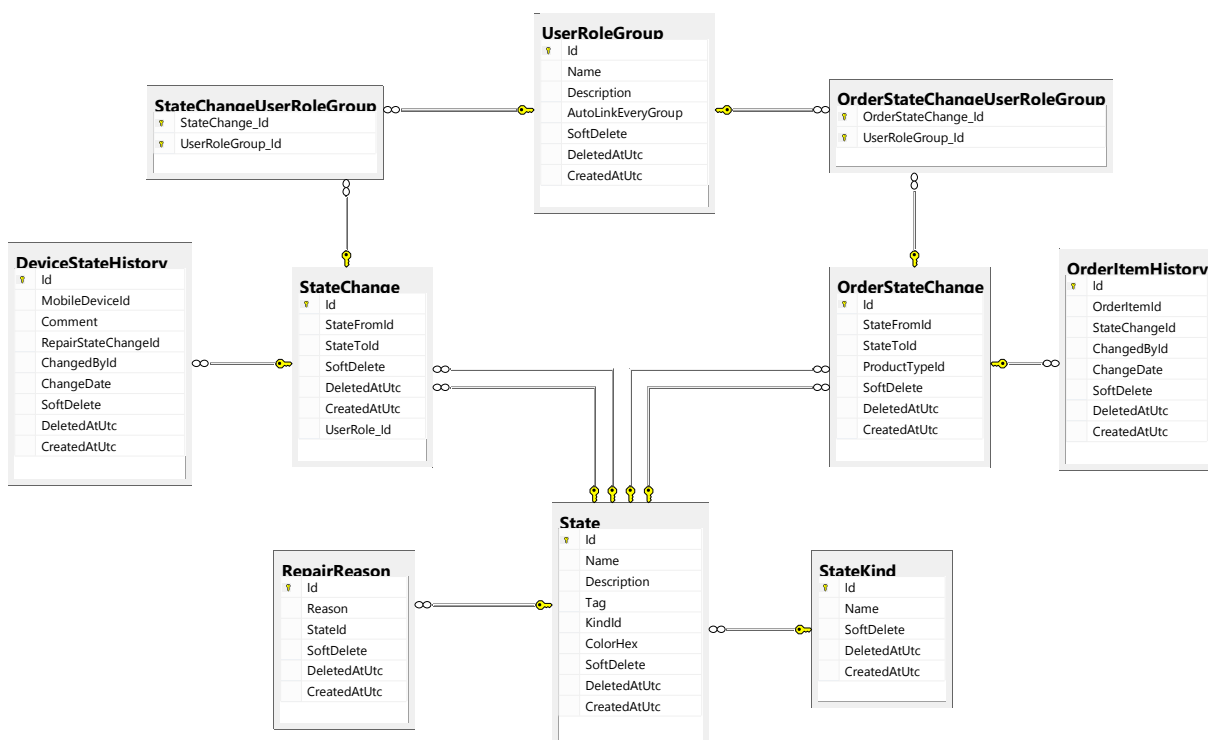


Figure XI: ERD Diagram - States

Using this structure allows us to dynamically build a workflow for MobileDevices and OrderItems and to add business rules later when the flow might change for a given object.

To track the states a MobileDevice or OrderItem has gone through respectively the DeviceStateHistory and OrderItemHistory tables exist. These connect a MobileDevice or OrderItem to a StateChange or OrderStateChange and combine it with the Current date and time, User responsible for the change and an optional Comment depending on the existence of RepairReasons on the State, this last part is enforced through business validation.

StateChanges and OrderStateChanges reference a collection of UserRoleGroups through 2 intersection tables shown in the diagram above. This last part further extends the dynamic state mechanism to a system in which only certain users can trigger a dynamically defined state or action.

## Used and implemented technologies

### Version control: Git + visualstudio.com

For code collaboration, we decided to go with a provider that would support Git as a version control system (VCS). We needed a repository which was private and cheap, possibly with added features such as feature tracking and sprint management.

As we are working in a Microsoft .NET environment, we chose visualstudio.com as a solution to our problem, it provided us with a free private repository, sprint management, bug and feature tracking as well as a proper integration with Microsoft Azure for automated deployments.

### Repository and Unit of Work pattern

Most applications need to consult a database or other data storage resource at one time or another. Directly accessing the data can result in disadvantages such as duplication of code and a higher risk in programming errors. Working with data directly makes it more difficult to implement general strategies regarding data retrieval like caching.

The repository pattern aims to hide data persistence code from the business logic. The goal is to provide a service to the business logic for persisting and query business entities, while being unaware of the tools required for this or the kind of storage being used.

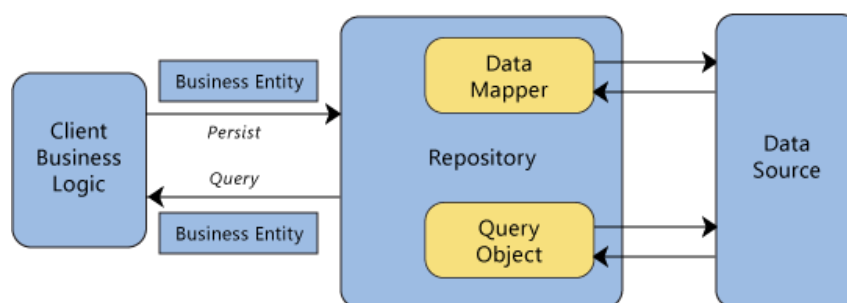


Figure XII: Repository and Unit of Work pattern

In the G4SOLDMan application a generic approach to the repository pattern has been taken. We have implemented a general implementation, valid for every object that inherits from our EntityBase class. Our repository uses the Entity Framework as an Object Relational Mapper to persist and query the database. The advantage of this approach is highly reusable code, easily maintainable and adjustable and most importantly, easy to unit test and mock.

In addition to the repository pattern a Unit of work pattern was implemented to allow for multi-entity transaction in the business logic. This pattern allows business logic to create a scope in which entity manipulations are tracked within one database connection. Upon failure of a certain call the entire transaction can be reversed, something which would not be possible in the repository pattern.

A combination of both patterns result in a business layer which can be built without having to worry about data accessing. Because of the generic repositories, our business layer, more specifically our readers and writers, was able to be implemented in a generic way as well.

## Entity Framework

Entity framework, also called EF, is an Object Relational Mapper that creates connections, commands and executes commands, to create and execute queries and processes these results. It even materializes objects from these results.

EF is created by Microsoft for the .NET development stack. It simplifies the effort that a developer must put in when he is going from objects in his software to a relational data that is persisted in the database.

EF uses mapping by conventions between the classes and the database structure to optimize both the database setup as the POCO classes each in their way to serve their purpose.

In our application, we started from an empty database. Therefore, we used the code-first approach of EF. Code-first references to updating the database scheme to align with changes you've made to the model. The changes are controlled by migrations that store the adaptations between the data model and their relationship. A migration can be implemented and reversed. The first migration will create the database, the following migrations will change this database.

Pre-defined keywords in EF will allow you to put in constraints, create specific relations between objects and even define foreign, primary and surrogate keys by using data annotations.

We have chosen for Lazy Loading to speed up the development process. We hide these virtual objects behind a repository pattern.

## Unity

G4SOLDMan is a large application with multiple layers, in order to keep this maintainable a loosely coupled solution is preferred. We chose to utilize Unity to accomplish inversion of control. Unity is a dependency injection container that is extensible and allows easy configurable and generic object creation.

When using an inversion of control framework, we don't have to instantiate underlying services we use, therefore we don't have to control their lifetime within the application either. The Unity container takes care of this by injecting constructors and properties with the necessary implementations of the interfaces we require for that specific class. This way we don't take on direct dependencies on certain implementation but code against the interface of a class. This coding style automatically results in better separation of layers throughout the testable code because of mocking out interfaces.

## Auto mapper

The backend of the G4SOLDMan application is heavily layered to achieve a strong separation of concerns. Each layer works with well described models tailored for the task it was given. This can be better explained using an example of a typical backend operation: Creating an object in the application, for instance the creation of Mobile device.

An HTTP POST request enters the application containing a MobileDevicePostModel JavaScript Object notation ([Json](#)) object. First the application will check the URL to determine the correct controller to instantiate, it then calls the Create method on a generic base controller with a deserialized version of the Json object.

The controller will do a basic validation and integrity test before converting the model into an entity of Mobile Device for the business logic to work with. It then resolves and calls a generic writer of Mobile Device to further validate and create the object in a data store.

The business logic can further convert the entity into something the repository classes need for persisting the given data. It even converts it into a flatter object to save historic information.

As explained above, data may need to be converted three times from API endpoint to database for each layer to properly do its task. Doing manual conversions would be labor intensive, prone to errors and generally considered bad coding. Therefore, we implemented a convention based object to object mapper.

We chose Auto mapper as our tool for this task because it allows for centralized configuration of mapping conventions by using a fluent [API](#). It is also capable of converting generic types without having to specify each one separately and works 'by convention' meaning it, by default, assumes properties on both source and target with the same name will be mapped to one another, again lowering the labor cost mapping both objects.

## Mocking framework

Before we use save, update or patch data in our database we use validator services to be sure that the action complies with the business rules. For these validator services, we wrote unit test.

To avoid hitting the database directly during these tests we used the Mocking Framework.

This framework, written for Entity Framework 6 onwards, enabled us to create testing doubles. Not only can we create testing doubles, it gives us a lot of control on how they will behave.

These in-memory implementations of test doubles and adaptable behavior of functions allow us to test these validators without directly hitting the database.

## AngularJS

AngularJS is a JavaScript client-side framework which extends the HTML standard used all over the web. The framework processes static HTML for the existence of an ng-app attribute and starts an Angular application within the scope of that element. Once this app scope is known, the framework can replace HTML directives with the implementations given in matching JavaScript files by using predefined structures. At this point, AngularJS has a binding between the HTML view and the Angular controller in JavaScript and watches changes coming from both the view as from the controller to keep updates in sync. Below the main structures of AngularJS are listed.



- Controllers
  - A controller is a JavaScript function defined by name and registered to the AngularJS application, it defines the behavior of a certain view without having to update the Document Object Model ([DOM](#)) manually or by registering callbacks for model changes on the view. Multiple controllers can be assigned to the same view (based on URL routing) to achieve a level of consistency throughout the app. It also gives developers less dependencies to worry about and an easily maintainable place to connect views to logic.
- Directives
  - At a high level, a directive is a marker on a DOM element that gets parsed by AngularJS's HTML compiler to attach a specified behavior or a transformation of the element. Angular comes with a large set of built-in directives like ngModel or ngHide to provide dynamic behavior to a static HTML file.
  - Directives can also be custom designed in an application to create custom components or templates for the application. In G4SOLDMan, more than 10 different directives were added to the default set provided by AngularJS to create reusable views like our table, detail pages, panels, dashboard items, breadcrumbs, ... but also for behavioral items such as role validation, routing history, collapsible and closable panels, table sorting and form validation.
  - The use of directives allowed us to make massive changes to views or small behavioral logic in the entire application in one well-defined place which creates an easy maintainable application.
- Services
  - Services are JavaScript functions defined by name and registered to the AngularJS application. They can be injected into all other structures and are therefore highly reusable. The benefit of this is that all controllers can use the same logic defined in a service and use it without knowing how the inner implementation works. This achieves the same level of separation of concerns we know from multi-tier backend applications.
- Filters
  - Filters in AngularJS are small JavaScript functions that can be reused throughout an application and can manipulate or format data of an expression to be more suitable to the user. Examples of this are typically date and time formatting or the way a decimal value should be shown.
- We provided some filters of our own to have a consistent format for a date, or for translating static elements on a page.

The use of AngularJS in our application allowed us to create a frontend which is highly maintainable, with a good separation of logic and views. This helped us to build new views containing multiple functions and a consistent look-and-feel in less time. Angular made it easier for us to make a large administrative application as a single page application which is more performant due to the asynchronous nature of the framework as well as less calls having to be made after the initial page load.

(AngularJS, 2010)

## Bootstrap

The Bootstrap framework is one of the more popular HTML and Cascading Style Sheets (CSS) frameworks currently available for developing responsive and mobile-first websites. This means the layout of web pages adjust dynamically to the size of the window given by a browser. This is mainly handled by using CSS classes on div elements in HTML, describing the way the page needs to be shown on a specific device using the grid system.

It also features styling for some reusable components such as buttons, modals, dropdowns,...

The G4SOLDMan application uses the Bootstrap 3 framework underneath a specific template which builds upon the framework with a dedicated style.

## Purchased front end template

We purchased a CSS template built upon Bootstrap 3 to create a consistent look-and-feel throughout the application. We chose a template containing administrative features like tables, forms, detail pages and panels, as well as a style for modal views and navigation bars. A dashboard layout and theme was also provided in the template with some styled implementations for various charting JavaScript frameworks.

## Business rules implementation

Several business rules result in a complex lifecycle for Purchase orders and mobile devices. A purchase order would find itself in a certain state at a certain point in time, predefined business actions would result in a transition of one state to another, possibly with some added information.

Rather than going through all possible states with actual use cases, we made a flow chart of these possible transitions between states. This diagram below explains the full process for a purchase order.

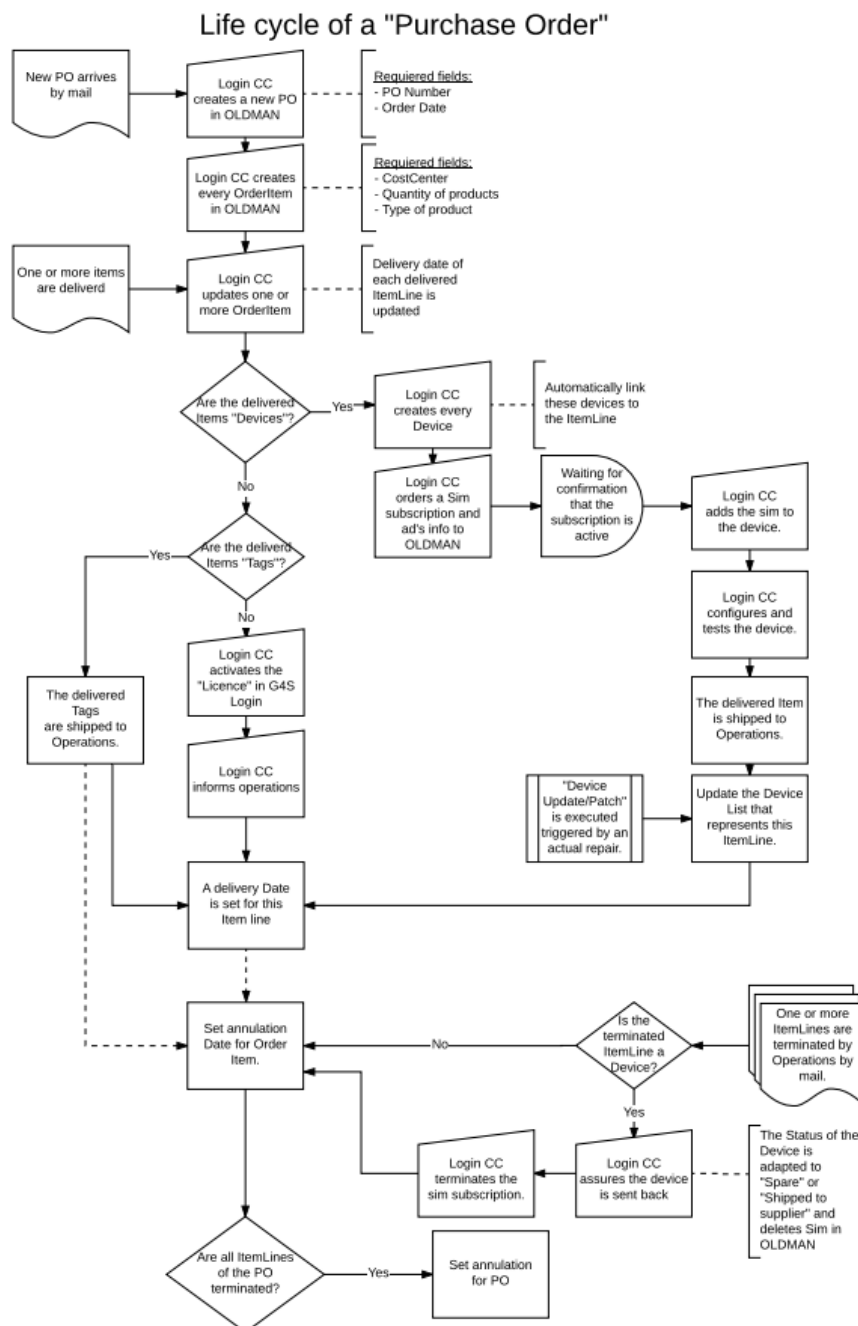


Figure XIII: Life Cycle of a Purchase Order

## Building for change

The mobile devices that are monitored by the application also have several different states. Because the landscape of the mobile devices used by G4S follows the need and availability on the market, the application must allow be built for change. Therefore, we have implemented the states in our application in a dynamic way. Users with elevated rights can alter the flow of the application by adding or removing states, changing the allowed transitions between the states and defining the groups of users or device types (which are also user configurable) that are allowed to use these states.

We require a state system which is built for change while also being easy to use for the current set of business rules. Therefore, we analyzed this and created a diagram of state transitions to be seeded in our dynamic system, a user with elevated rights can later adjust the system to adhere to the future business rules.

### DEVICE STATE DIAGRAM

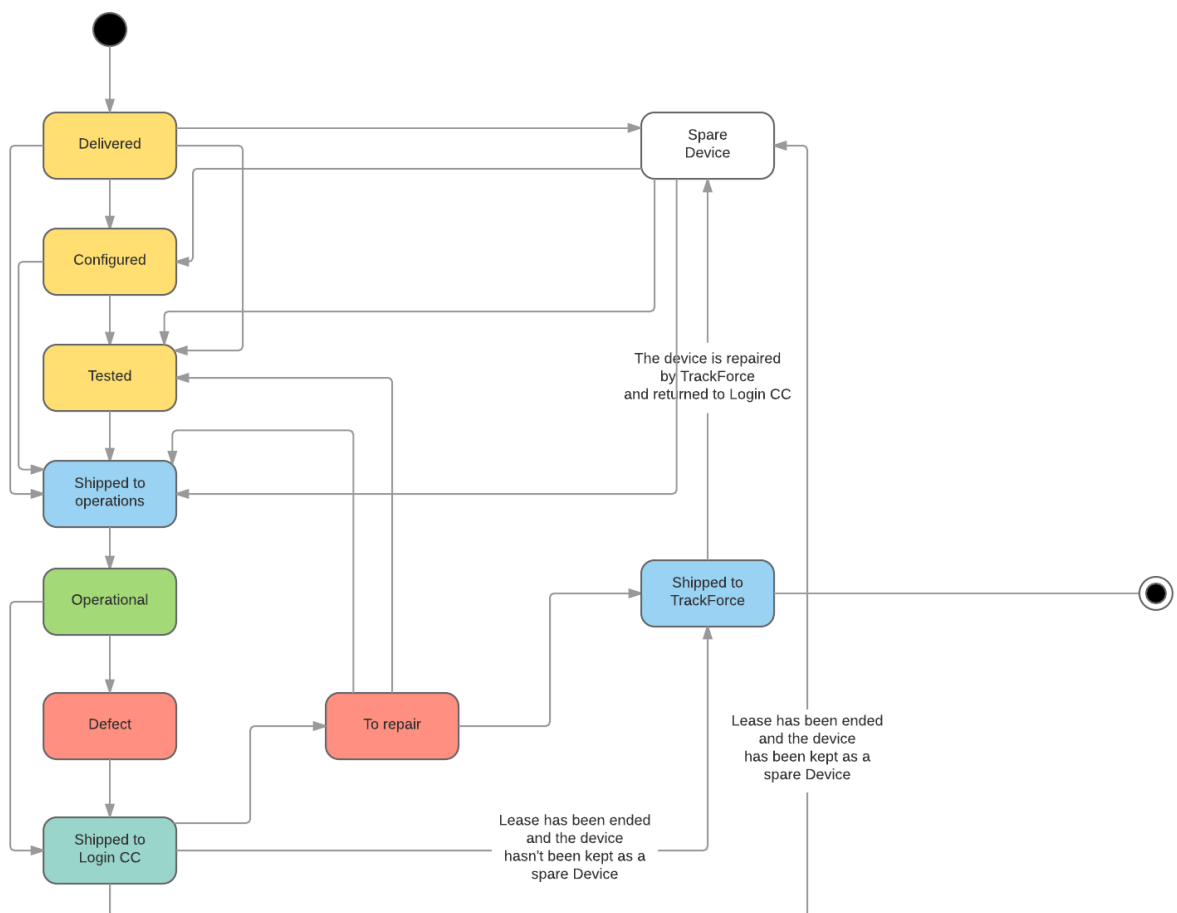


Figure XIV: Device State Diagram

Because of the way our state system is build, a user consulting the mobile devices screen is dynamically limited in his actions, the following restrictions are implemented:

- The user will only see data that belongs to a login site which he is entitled to see
  - Login site and the relation to users is configurable in the application
- He can edit a device if he belongs to a UserRoleGroup that is allowed edits
  - UserRoleGroups are dynamic for a user and contain a configurable set of UserRoles
- He can change the state of a device based on the flow of the states
  - State transitions are dynamic and are only allowed for certain UserRoleGroups
  - The type of the device (which is configurable) is checked for that transition

We tried to build our application in a generic way and as a result the state system described above for mobile devices was automatically implemented for purchase orders as well.

When state changes are made, the application track these in a StateHistory table for reporting reasons. A dashboard was made to show various statistics about the states devices are currently in. More information about dashboarding in relation to states can be found in the following chapters.

## Technical implementation details

### Validation process

Because data is imported from a csv file or manually added by a LoginCC employee, we needed a validation process before this data is saved in our database. This validation of data will not only ensure that we import or update correct, complete and non-redundant data, it will give the users a form of insight on the data coming from G4S Login.

Rather than creating one big validation process to verify and validate this data, we separated this process in different pieces. The validators needed for a specific part of data can then be linked together. We benefit in many ways of the splitting of the validation process. It will be more dynamic, efficient, easier to maintain and it will have a positive impact on performance.

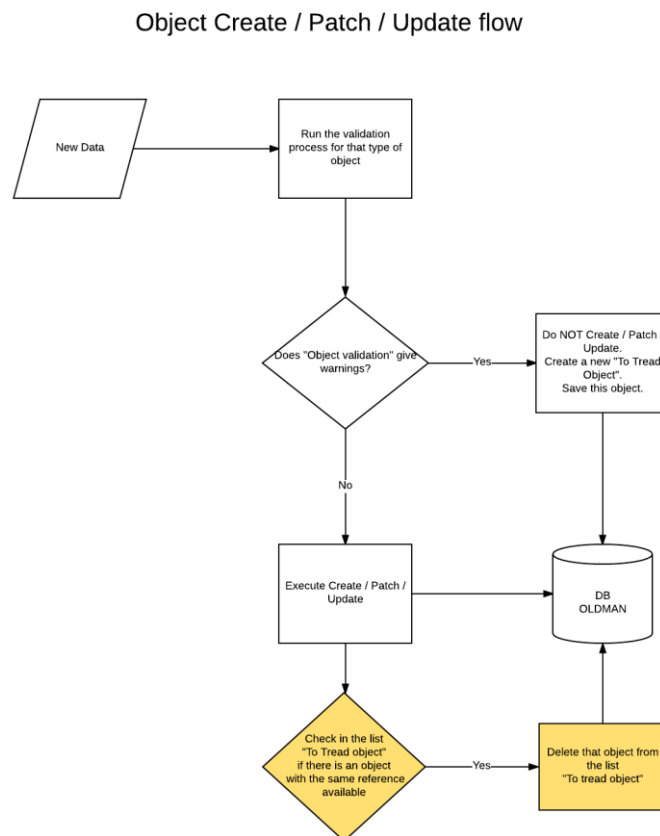


Figure XV: Object Create/Patch/Update Flow

Every validator will return a success code and if this result is invalid it will be accompanied by a collection of warning messages. These messages will give more insight about why the result is invalid. To maintain a uniform validation process we made a generic validator for all our entities. For entities where special business rules apply, we created a specific validator which will be automatically selected when requested through our Unity container.

```

public interface IValidator<Tentity> where Tentity : EntityBase
{
    Task<ValidationResult> ValidateAsync(Tentity entity);
    Task<ValidationResult> ValidateInsertAsync(Tentity entity);
    Task<ValidationResult> ValidateDeleteAsync(Tentity entity);
    Task<ValidationResult> ValidateUpdateAsync(Tentity entity);
    Task<ValidationResult> ValidateRestoreAsync(Tentity entity);
}

```

As with all code in our application, validators are also asynchronous. We do this to enhance performance when the application receives a large number of requests. Asynchronous coding in ASP.Net results in the request thread pool being freed up when database or file access is needed. In a synchronous application consulting a database to check if a certain value already exists would block that request thread until the call is returned. As this might will usually take less than a few milliseconds this does not seem to be an issue, however when an application receives a heavy load with a lot of users requesting data at the same time, the available threads left are disappearing quickly.

Asynchronous calls to a database will return the request to the pool and pick up another one when the call returns, resulting in a higher number of users that can simultaneously use our application.

This also benefits our CSV import because of the larger size of data that needs to be read and processed at the same time. We can only read the file using one thread, but process it using multiple threads at once.

In case of the CSV import, when the result of validation is not valid, the CRUD action will not be performed in the database, but the imported data is saved in a ToBeTreated table. This table can provide the users a list of objects that aren't created, updated or patched correctly. It also allows the user to correct the data or intervene in G4SLogin if the data originates from there.

If the result is valid the CRUD action will be performed in the database and the ToBeTreated entity with a matching reference is removed.

### Import Device Data from Csv

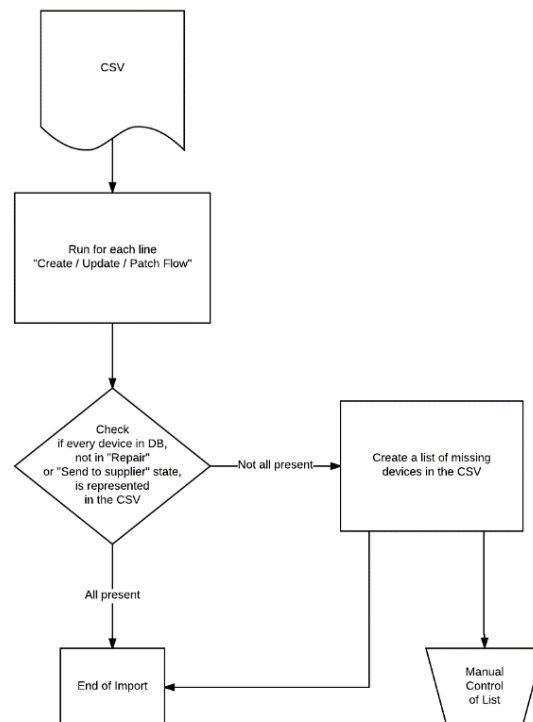


Figure XVI: Import Device Data from CSV

The validation process for the mobile devices and the possible related Lone Worker Protection (LWP), is a complex process with many specific conditions. Before we created these validators, we analyzed it in-depth. This resulted in several flow diagrams, which can be found on the following pages. By studying these flow diagrams, you also get an insight in the chaining process of different validators.

Not only getting insight in the conditions that needed to be implemented, these diagrams provided us with structure before we created our validators.



## Device Create Validation

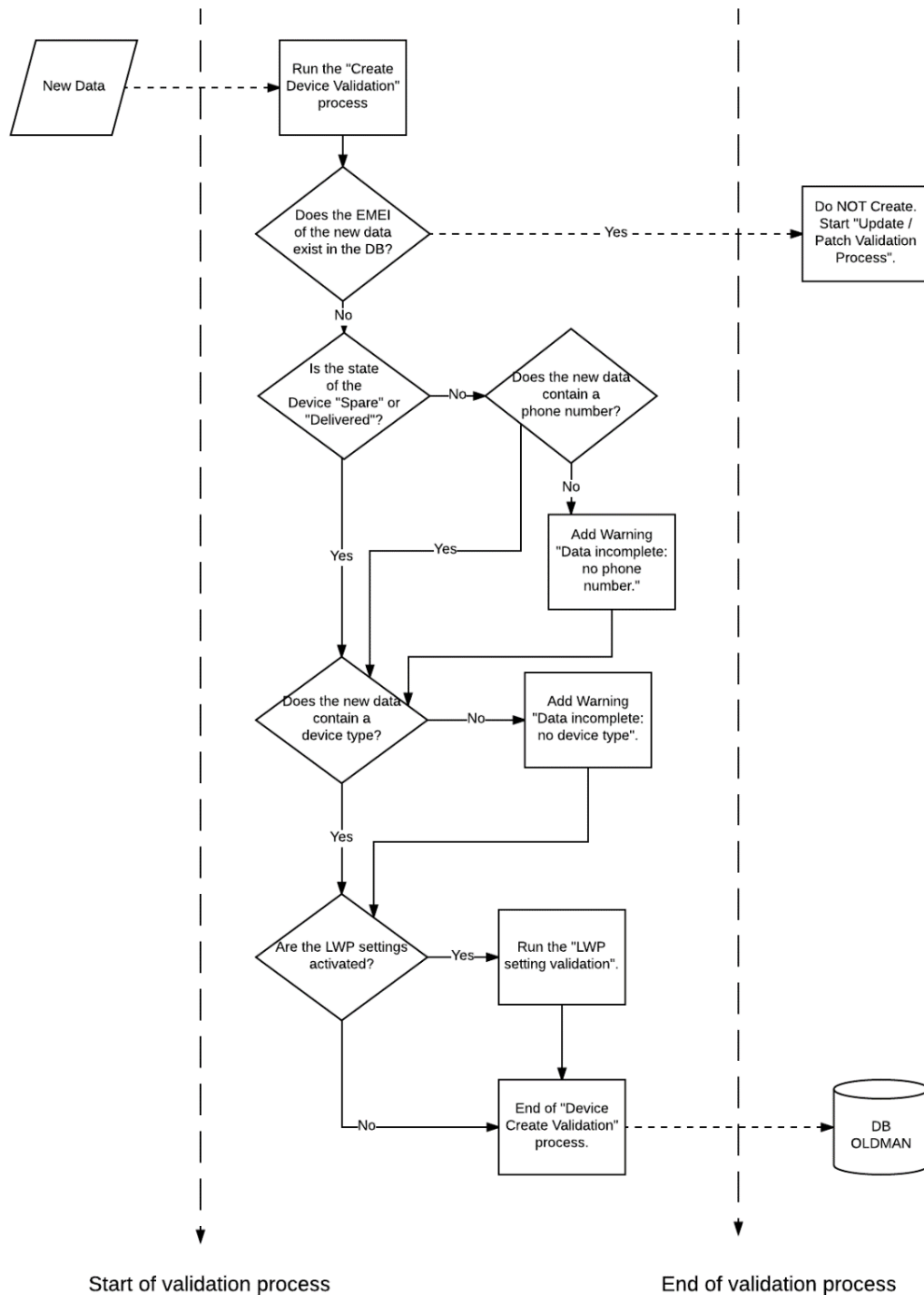


Figure XVII: Device Create Validation

## Device Update or Patch Validation

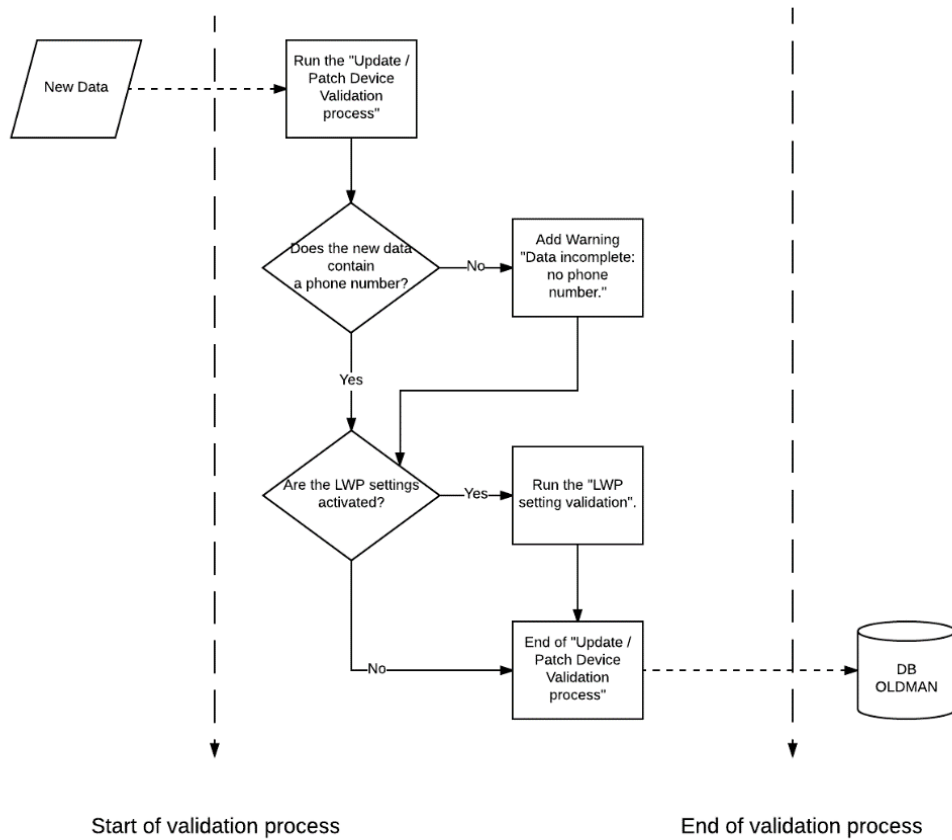


Figure XVIII: Device Update or Patch Validation

## LoneWorkerProtect settings Validation

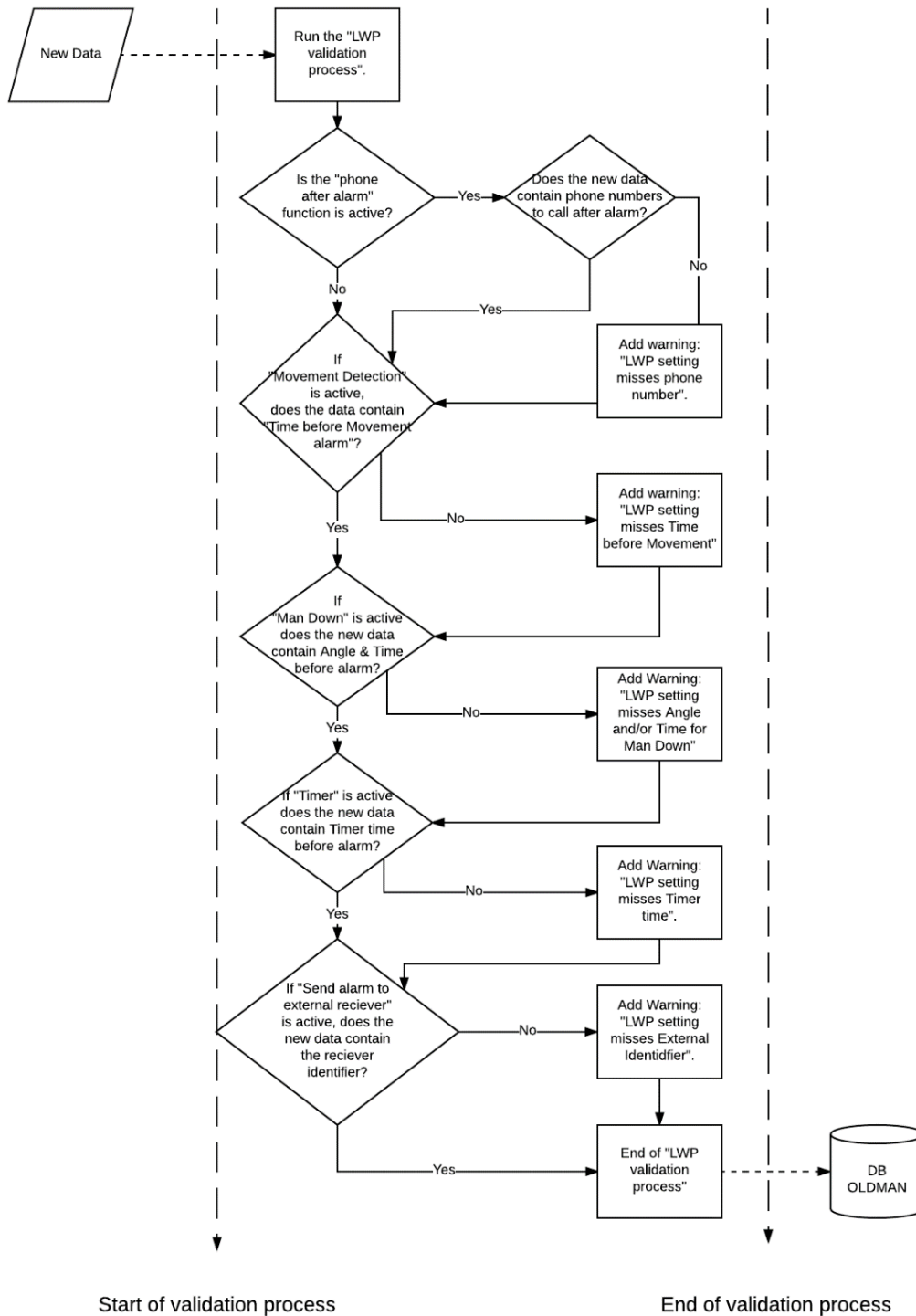


Figure XIX: LoneWorkerProtect settings Validation

## Tags for dashboarding

A dashboard was made to show various stats about the states devices are in. The implementation of this dashboard resulted in added properties to the dynamic state object to allow a dynamic dashboard to be created.

A user with elevated rights can decide which states are shown on the dashboard with an icon to match its dynamic description. Multiple states can even be combined

### Example

A state named “reported broken” and a state named “dead on arrival” can both have a tag named “faulty devices”, resulting in a dashboard item being added with the name faulty devices and a value that matches all the devices currently in a state reported broken and dead on arrival. In the example below we see that currently 3 devices are in these states, accounting for 2% of all devices currently in use, we created the tag being, an icon was added to the tag to further build the custom dashboard. The name is dynamic, but will also run through the dynamic translation flow, which means a tag can be chosen which can be translated to all languages which are known to the application, see Translations in this chapter for more details.

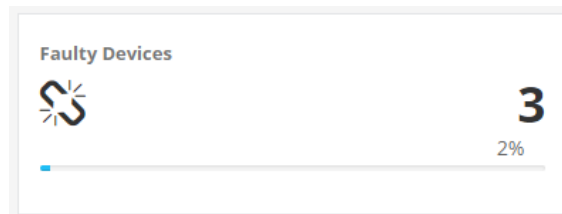


Figure XX: Faulty devices

## Synonym possible for devices

We have user configurable lists in our application to allow an administrator to define what values are available in dropdowns or selection menus throughout the application. Some of these define properties for a mobile device, such as Platform, DeviceType and LoginSite. DeviceType even contains properties indicating how a certain mobile device should behave when being created or used with regards to states.

A problem occurred when we started developing our CSV import functionality. The given CSV file contained different names for the same platforms and device types. Sometimes because they were in another language, sometimes because the name used by the staff of G4S is different from the actual type of the device. As a result we could not automatically match the name found in the device to the correct device type or platform and all records were being save to our ToBeTreated tables.

We came up with a solution of synonyms. We decided to implement an extra field containing all possible synonyms for a given object. That way, administrators working through the list of ToBeTreated devices can match an “ActiveWand” record to an “ActiveGuard” device type simply by adding “ActiveWand” to the list of synonyms for that DeviceType. This resulted in an import mechanism that was built for the unknown.

## Users – UserRoleGroups – UserRoles

To be able to control access to different views and functionality for every user, we implemented a fixed UserRole collection, this collection is one of the only things not configurable in the application as it needs to be hard coded. If the user doesn't have the correct role he won't be able to view that part of the application or won't be able to use the function. He won't even notice a function is available.

We could have linked a collection of UserRole to each user and allowed for this collection to be configurable by an administrator. However, this would be a painstaking task when you need to add a lot of users at a given moment.

To fix this we implement the UserRoleGroup entity to link a user to a collection of roles. A user must be a member of a UserRoleGroup and the collection of roles is managed within that group. This way an administrator only needs to define access rights once for all kinds of users. He would later set a given group on a new user or edit a user with a different group.

## Translations

Since we want to provide this software in the 2 most used languages in Belgium we created a translation service. Using such a translation technique allows G4S to potentially deploy this software in other countries.

Like most other processes in G4SOLDman we approached this one as dynamic as possible. An administrator can create new available languages. Once a user navigates to different windows in the software, keys will be translated by looking for the value in preloaded translation table for that specific language. The software also automatically populates these translation tables once it encounters a new key.

To configure and to correct the translations of these keys an extra view is provided for the administrator. Since finding the exact key word for a certain translation can be difficult we made different groups for the keywords. The translation configuration view allows the admin to sort, search and order the translation table that is provided.

Even with all these options finding the correct keyword can be difficult from time to time. If two keywords are translated the same way but we want to correct one of them (perhaps for use in a different context) an administrator should figure out the correct keyword by trial and error. To avoid this discouraging process an extra function was added to switch off the translation entirely for that browser session. The result is an application in which nothing is translated and the core keys are shown 'Group.Key'. This simplifies the searching for the exact keywords dramatically.

## Epilogue

We can note that we have properly completed this assignment. The results include the expressed wishes by the stakeholder. With our experience, we have accomplished the somewhat extremely complex technical puzzle.

In September when we started this project we had the chance to get a clear description of the wishes of the stakeholder. Nevertheless, during the completion of it we encountered that a well described wish can be more work than expected.

Although creating and following a clean and well thought planning is an art on itself, we started full of excitement with our basic planning. During the course, we soon realized that fine tuning this planning was a must to arrive in time with a finished product. This did not only challenge us but it made us work better as a team. We learned not to panic but work constructively with the eye pointed to a solution.

We can only rely on the feedback of the stakeholder, since we were not allowed to involve all end users during the project. Even we can say that the delivered result can be a great quality upgrade if we compare the sharing of correct information to the end users in the current state with the available features that we have provided in this software.

## Appendix

**Appendix 1:** Aanmelden van project

**Appendix 2:** InvulbladEindwerk\_Vervoort\_Noreillie.doc

**Appendix 3:** Meeting report 19/10/2016

**Appendix 4:** Meeting with promoter

**Appendix 5:** Meeting report 10/02/2017

**Appendix 6:** Material export incomplete

**Appendix 7:** Deploying the website and DB login

**Appendix 8:** Meeting report 02/05/2017

## References

- AngularJS. (2010, February 15). *AngularJS*. Retrieved May 08, 2017, from AngularJS - Superheroic Javascript MVW Framework: <https://angularjs.org>
- Auth0. (2015, October 08). *JSON Web Token Introduction - jwt.io*. Retrieved May 08, 2017, from JWT: <https://jwt.io/introduction/>
- Christensson, P. (2013, December 11). *IIS (Internet information Services) Definition*. Retrieved May 15, 2017, from TechTerms: <https://techterms.com/definition/iis>
- Cohn, M. (2013, Octobre 28). *User Stories and User Story Examples by Mike Cohn*. Retrieved 05 08, 2017, from Mountain Goat Software: [www.mountaingoatsoftware.com/agile/user-stories](http://www.mountaingoatsoftware.com/agile/user-stories)
- G4S. (2010, March 24). *G4S Sectors*. Retrieved May 08, 2017, from G4S België: <http://www.g4s.be/nl-BE>
- G4S. (2017, 5 14). *G4S Corporate Financial Annual Report*. Retrieved from G4S Corporate: [http://www.g4s.com/-/media/G4S/Global/Files/Annual-Reports/AR-2016-Extracts/G4S\\_2016IR\\_Final\\_PDF.ashx](http://www.g4s.com/-/media/G4S/Global/Files/Annual-Reports/AR-2016-Extracts/G4S_2016IR_Final_PDF.ashx)
- Introducing JSON*. (2002, December 11). Retrieved May 08, 2017, from JSON: [www.json.org](http://www.json.org)
- Le Hégarret, P., Wood, L., & Robie, J. (2000, November 13). *What is Document Object Model?* Retrieved May 15, 2017, from W3C Recommendation: <https://www.w3.org/TR/DOM-Level-2-Core/introduction.html>
- Microsoft. (2012, February 2). *Common Language Runtime (CLR)*. Retrieved May 08, 2017, from Microsoft Developer Network: [https://msdn.microsoft.com/en-us/library/8bs2ecf4\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/8bs2ecf4(v=vs.110).aspx)
- Rouse, M. (2014, December 02). *What is REST (Representational state transfer)? Definition from whatis.com*. Retrieved May 15, 2017, from TechTarget: <http://searchmicroservices.techtarget.com/definition/REST-representational-state-transfer>
- Rouse, Margaret. (2017, April 28). *What is application program interface (API)? - Definition from WhatIs.com*. Retrieved May 15, 2017, from TechTarget: <http://searchmicroservices.techtarget.com/definition/application-program-interface-API>
- TechTarget. (2010, September 28). *What is Ajax? (Asynchronous JavaScript and XML?) Definition from Whatis.com*. Retrieved May 08, 2017, from TechTarget: <http://searchwindevelopment.techtarget.com/definition/Ajax>
- Webfinance Inc. (2008, November 20). *What is SAP? defenition and meaning*. Retrieved May 08, 2017, from Business Dictionary: <http://www.businessdictionary.com/definition/SAP.html>



## Appendix 1: Aanmelden van project

2017-5-17

Gmail - Aanmelden van project Noreillie en Vervoort 3e jaar IT



peter vervoort <peter.vervoort@gmail.com>

---

### Aanmelden van project Noreillie en Vervoort 3e jaar IT

2 berichten

**peter.vervoort** <peter.vervoort@gmail.com>

12 september 2016 om 23:25

Aan: patrick.fox@ucll.be


Cc: Sven Noreillie <sven.noreillie@gmail.com>, Manuel Mestré <manuel.mestre@be.g4s.com>

Dag Patrick,

Via deze weg dien ik het project van het 3e jaar IT.  
Dit is voor het project van student Noreillie Sven en Vervoort Peter.  
U vindt het document in bijlage.

Mvg,  
Peter Vervoort

---

 InvulbladEindwerk\_Vervoort\_Noreillie.doc  
35K

---

**Patrick Fox** <patrick.fox@ucll.be>

19 september 2016 om 20:31

Aan: Peter Vervoort <peter.vervoort@gmail.com>

Cc: Sven Noreillie <sven.noreillie@gmail.com>

?Bedankt Peter en Sven

Mvg,  
Patrick Fox

## Appendix 2: InvulbladEindwerk\_Vervoort\_Noreillie.doc

Title:

G4S OLDMan (OrderLicenseDeviceMANagement) (TBD)

Subject:

Multi-user, multi-language web application that provides central management of orders and repairs for G4S Login mobile devices or licenses. The goal is to relieve the current LoginCC desk with questions related to the device repair status or delivery flow. It does so by automatically managing data now, existing in several Excel documents.

Users will be limited to only their own devices and have the ability to start a repair process. Users with a management role will have the possibility to view KPI's. Examples include: duration of repairs and deliveries, volumes of devices by status (operational, spare and defected). Admin members will be able to adjust the flow of repair for each type of product, manage the different user levels, ...

The application will periodically import data from G4S Login and execute checks for consistency based on several predefined conditions in order to push out notifications. Auditing and several levels of exports will be mandatory.

Company & sector:

G4S Belgium – Safety and security sector

Members:

Name	Group	Phone	Email
Vervoort Peter	A	0493246311	Peter.vervoort@gmail.com
Noreillie Sven	A	0498472528	Sven.noreillie@gmail.com

Platform:

- C# ASP.Net Web application
- Angular 1.4
- Entity Framework
- Database to be decided (MS Sql, MySQL, Postgres) Depending on pricing and licensing
- Sandboxed virtual server

Preferred Promotor:

Marchelbach Guy, Verelst Henk

### Appendix 3: Meeting report 19/10/2016

Meeting 19/10/2016

Attended Stakeholder – Manuel Mestré and Stefan Wauters by:  
Project Members – Sven Noreillie and Peter Vervoort

Discussed points:

- What kind of infrastructure is available for the project?
  - G4S can provide access to a MS-SQL
  - For Web Server G4S can provide IIS Server 2000 till 2012 version
- The setup and accesses to the infrastructure can be made/given on demand, by mail to Stefan Wauters.

This process isn't labor intensive and can be realized quick depending on the workload of operations. Stefan Wauters also confirmed that a development area is available before the project will be deployed in production environment.
- The stakeholder confirmed us that the data export (information sharing from G4SLogin to G4SOLDMan) will be executed through csv files.
  - The stakeholder requests from the project team which data needs to be exported from G4SLogin. Project members will give an overview of the wanted data by mail.
  - The stakeholder confirmed that he will provide an initial export for development faze and testing purposes.
  - In a later faze, when all export parameters are clearly defined, the supplier of G4SLogin will provide the possibility to export this csv file out of G4SLogin true an extra export button.
- Project team members shared with the stakeholder the following documents in first draft version to review:
  - Planning/deliverables
  - Product Vision
  - MoSCow
  - Event response list

#### Appendix 4: Meeting with promoter

Meeting with promoter Mathy Paesen  
Attended by project members Sven Noreillie and Peter Vervoort.

Promoter had a meeting with all other IT docents. They decided to ask all projects to work in an agile way on their projects. Team members confirmed that they will approach the project in an agile way after they done the primary analyses. (Definition scope, MoSCoW, function composition diagram, ..., Global DFD, Definition of subsystems)

##### Notes not to forget:

- Add FloclD to lexicon
- Mobile Devices should be linked to sites
- Promoter suggests to solve the 1 to 1 relations by using inclusive subtyping
- Henk Verelst suggests to look at <https://www.nuget.org/packages/TrackerEnabledDbContext/>  
For an example with DB of Entity Framework

Promoter states that the team should challenge the stakeholder.

1. Should states be implemented in a dynamic way?  
Yes: Extra code work and extra overhead / admin once OLDMan is deployed to maintain the software.  
No: Fixed states for each object (Device, License, Tag)
2. Should the logging occur by recording who, when changed what object property, old and new value (change records: head table id type user date, sub table whit wat changed old value new value and version number). Or should the logging record the full object in a history table and add who changed when and a version number (full log).  
Only change records will cause a slower export but saves coding time and DB volume.  
Full log, will export faster but is more coding time and huge DB volume.  
Is for the export a delay tolerated or should it be near real time?

##### To Do before next meeting with promoter:

- Remake planning of project -> agile approach
- Create the following analyze or diagram:
  - Define major processes
  - DFD for each of the major processes
  - Global DFD
- Create Product backlog for first epic (1<sup>st</sup> epic can be User management)

## Appendix 5: Meeting report 10/02/2017

2017-5-17

Gmail - Meeting report OLDMAN 10/02/2017



peter.vervoort <peter.vervoort@gmail.com>

### Meeting report OLDMAN 10/02/2017

2 berichten

peter.vervoort <peter.vervoort@gmail.com>

10 februari 2017 om 21:55

Aan: Manuel Mestré <manuel.mestre@be.g4s.com>, Sven Noreillie <sven.noreillie@gmail.com>

Cc: Mathy Paesen <mpaesen@gmail.com>

#### Present :

- Stakeholder Manuel Mestré
- Project member Sven Noreillie
- Project member Peter Vervoort

#### First topic flow charts of device data validation on create/update/patch:

- The charts are understandable and approved by stakeholder
- Only comment/remark from stakeholder, is the simcard order date stored?
- Project members explained the dynamic state pattern they are implementing for all objects.
- Dynamic: an admin user can create states for each object type and afterwards allow or refuse changes between two particular states.
- Each state records a creation date.
- Project members will implement several basic states in a seed method. Before the next meeting they will create a temporary overview of these basic states.

#### Second topic flow chart of life cycle of a purchase order in OLDMAN:

- The charts are understandable and approved by stakeholder.
- In some processes in this flow chart, states of several objects will change automatically.
- Project members will try to create a detailed flow chart that captures these automatic state changes.

#### Third topic access rights to views and/or processes in OLDMAN:

- Project members explain the approach on how they will tackle user access to several menus and/or processes.
- For each menu, view and process a user wants to access, his account will need the specific access right defined by the menu, view and process.
- An account is linked to one or more right groups.
- Right groups is a collection of one or more access rights, defined by a name.
- The access rights are hard coded in the application.
- An admin can create right groups and add the access rights he wants to this group.
- An admin can also edit the user groups linked to an account.

#### Fourth topic dashboard principal parameters / KPI's that needs to be tracked:

- Project members ask the stakeholder what he finds the most important parameters to track in the dashboards?
- Stakeholder makes the following list in a non-specific order of the most important parameters to track:

<https://mail.google.com/mail/u/0/?ui=2&ik=7424adbad0&view=pt&q=manuel.mestre%40be.g4s.com&qs=true&search=query&th=15a29d2c3aabe9be&siml=15a29cf1b93ef663&siml=15a29d2c3aabe9be>

1/2

2017-5-17

Gmail - Meeting report OLDMAN 10/02/2017

1. Quantity of spare devices ordered by type
2. Quantity of devices in repair state ordered by type
3. Average repair time
4. Quantity of devices that still need to be delivered by type
5. Average delivery time

- These are the most important parameters to track in a dashboard. Other parameters can be interesting but not that important.

#### Last topic example CSV export from Login:

- The stakeholder should receive an example csv file, that is the result of an export of all available data in login about the devices.
- The stakeholder will forward this example by mail to the team members if he has received it.
- TrackForce promised to deliver this csv file next Monday.

Next meeting is planned 17/03/2017 4pm till 5pm.

## Appendix 6: Material export incomplete

2017-5-17

Gmail - G4S OLDMan : material export incomplete



peter vervoort <peter.vervoort@gmail.com>

---

### G4S OLDMan : material export incomplete

1 bericht

---

**peter.vervoort** <peter.vervoort@gmail.com>  
Aan: Manuel Mestré <manuel.mestre@be.g4s.com>  
Cc: Sven Noreillie <sven.noreillie@gmail.com>

16 maart 2017 om 16:26

Manuel,

In the CSV export we got from TrackForce we mis 4 columns:

- Temps avant alerte (en secondes) (de la détection d'absence de mouvements)-
- Envoyer les alertes à une centrale d'alarme externe
- Identifiant unique à envoyer à la centrale d'alarme externe
- Sortir de la zone de geofence lorsque l'utilisateur se déconnecte
- Temps avant alerte (en secondes) (de la protection par timer)

Can you ask Sylvain of TrackForce to add these columns to the export?  
Or should i take directly contact with him, keeping you in cc of the mails?

Kind regards,  
Peter Vervoort

## Appendix 7: Deploying the website and DB login

2017-5-17

Gmail - Scriptie G4SOLDMan : deploying the website and DB login



peter.vervoort <peter.vervoort@gmail.com>

---

### Scriptie G4SOLDMan : deploying the website and DB login

1 bericht

---

**peter.vervoort** <peter.vervoort@gmail.com>

19 april 2017 om 09:44

Aan: stefan.wauters@be.g4s.com

Cc: Manuel Mestré <manuel.mestre@be.g4s.com>, Sven Noreillie <sven.noreillie@gmail.com>

Hi Stefan,

Our thesis is near completion and we want to test deployment and database setup.

Can you configure a connection string to a SQL db for us?

To test our deployment is there a sandbox environment we could use?

Or do you have any other suggestion where we could start testing our deployment?

If our progress doesn't encounter any major delays we expect to deploy for the first time at the end of next week.

With kind regards,

Peter Vervoort

## Appendix 8: Meeting report 02/05/2017

2017-5-17

Gmail - Meeting report G4SOLDMan 02/05/2017



peter.vervoort <peter.vervoort@gmail.com>

### Meeting report G4SOLDMan 02/05/2017

1 bericht

**peter.vervoort** <peter.vervoort@gmail.com>  
Aan: Manuel Mestré <manuel.mestre@be.g4s.com>  
Cc: Sven Noreillie <svennoreillie@gmail.com>

2 mei 2017 om 17:09

#### Present 02/05/2017:

- Stakeholder Manuel Mestré
- Co-worker Login CC G4S José Chapellier
- Project member Peter Vervoort
- Project member Sven Noreillie Excused

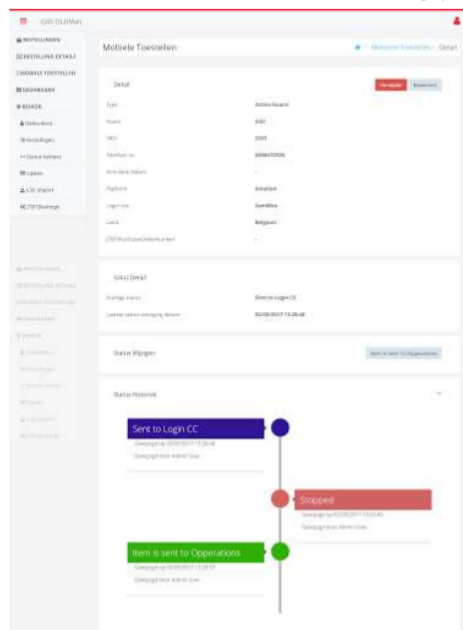
#### The only point on the agenda for this meeting was a demo of the actual code.

- The software is for test reasons deployed on an [azurewebsites.net](https://azurewebsites.net) web server.
- The actual deployment of the software hasn't the import CSV feature. This function will be added in a later stage.
- Manuel Mestré and José Chapellier had some questions after the demonstration:  
**Every point represents a question by the stakeholder.**  
**Some points are answered inline in the blue color.**

- States should be color coded, for easier recognition in history overview  
Fixed today, the states can be linked to a color (chosen in a color picker). Like you can see in next print screen.

2017-5-17

Gmail - Meeting report G4SOLDMan 02/05/2017



- The axes on the charts in the dashboards don't have values or a title.
- What kind of inscription is used for the password hashing?  
*We use the default implementation by asp.net. It uses a Key Derivation Function with random salt to produce the hash. The salt is included as part of the output of the KDF. Thus, each time you "hash" the same password you will get different hashes. To verify the hash the output is split back to the salt and the rest, and the KDF is run again on the password with the specified salt. If the result matches to the rest of the initial output the hash is verified.*  
Hashing:  

```
public static string HashPassword(string password) { byte[] salt; byte[] buffer2; if (password == null) { throw new ArgumentNullException("password"); } using (Rfc2898DeriveBytes bytes = new Rfc2898DeriveBytes(password, 0x10, 0x3e8)) { salt = bytes.Salt; buffer2 = bytes.GetBytes(0x20); } byte[] dst = new byte[0x31]; Buffer.BlockCopy(salt, 0, dst, 1, 0x10); Buffer.BlockCopy(buffer2, 0, dst, 0x11, 0x20); return Convert.ToBase64String(dst); }
```

  
Verifying:  

```
public static bool VerifyHashedPassword(string hashedPassword, string password) { byte[] buffer4; if (hashedPassword == null) { return false;
```



```

} if (password == null) { throw new ArgumentNullException("password"); } byte[] src = Convert.FromBase64String(hashPassword); if
((src.Length != 0x31) || (src[0] != 0)) { return false; } byte[] dst = new byte[0x10]; Buffer.BlockCopy(src, 1, dst, 0, 0x10); byte[] buffer3 = new
byte[0x20]; Buffer.BlockCopy(src, 0x11, buffer3, 0, 0x20); using (Rfc2898DeriveBytes bytes = new Rfc2898DeriveBytes(password, dst,
0x3e8)) { buffer4 = bytes.GetBytes(0x20); } return ByteArraysEqual(buffer3, buffer4); }

```

- Translations should be seeded on deployment for NL and ENG.
- Why is the spacing in the history on the left side bigger than the right side?  
The difference in spacing isn't relevant to the time between two states.  
This difference is originated from the presentation component itself.  
If the project has some time left we will try to solve this.
- When the last functions are finished, Manuel would like that we run some simulations of actual cases.
- Manuel also wants us to run a performance test on a large volume of data
- The list separator of the CSV export is desirably a semi column -> ;

With Kind Regards,

Sven & Peter

