# Глава 2. Архитектура на облачна система за управление на поръчки от клиенти

Тази глава разглежда решение от високо ниво, което да съсредоточава върху всички основни потребителски, бизнес и ИТ изисквания. Важна част от глава 2 са градивните елементи и интерфейси, изграждащи системата, както и комуникационните модели, които да ръководят композицията. Сложността на операциите стреми да бъде сведена до минимум. Представени са всички случаи на употреба и бизнес сценарии, съвместно с които се моделират приложенията за обслужване на клиенти. Освен това дизайнът обхваща функционалност, използваемост, устойчивост, производителност, икономически, технологични ограничения, компромиси и естетически проблеми на бекенд частта.

## 2.1. Същност, цел и обхват на софтуерната архитектура

Думата „**архитектура**“ често се използва в контекста от високо ниво, което е отделено от детайлите на по-ниско ниво (Martin et al., 2017). Софтуерният продукт, разглеждан в настоящия труд, се състои от 2 клиентски приложения, които се свързват към разпределена бекенд система, базиранa на микроуслуги, работеща върху множество процеси и сървъри (хостове). Всяка услуга се изпълнява в отделен процес като контейнер, разположен в клъстер от виртуални машини. Това разделение на подсистеми и отделни нива и компоненти цели да постигне разбираемост и лесна поддръжка. Работните рамки са съвместими на всяко ниво, без да се дублират функционалности.

На фигура 2.1 са показани приложенията, които изграждат системата за управление на поръчките от клиенти.



Фиг. 2.1. Диаграма от високо ниво на главните приложения (разработка на автора)

### 2.1.1. **Ключови бизнес процеси и дейности свързани със системата за** управление на поръчките

Тази подточка представя важни случаи на употреба, които са критични за бизнеса и са част от основния домейн. Използвани са UML диаграми на бизнес сценариите. Те идентифицират действия, които очакваме потребителите да направят.

Най-подходящ за взаимодействие с крайните потребители са мобилните приложения. Важни техни характеристики са, че поддържат функции като местоположение, камера и работят с уеб API. Клиентите на фирмата, които се явяват крайните потребители, управляват и проследяват поръчките и доставките в реално време с мобилно приложение. Целта му е да помага с планирането и логистиката, да въздейства върху крайния резултат с информация и данни. Тази информация, на смартфона, трябва винаги да е актуална, тъй като текущото състояние на поръчка и местоположение на доставките се проследява на живо. Други възможности са преглед на история, създаване на нова, промяна или отказване на **не активна** съществуваща поръчка. Приложението може да се разпостранява безплатно чрез Google Play Store и Apple App Store.

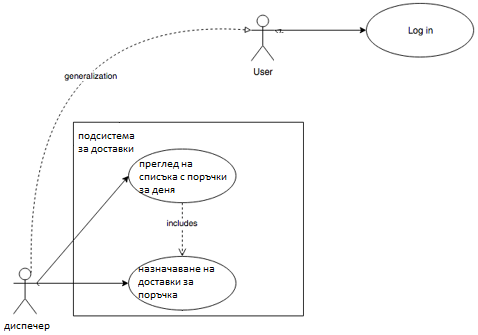
Обхватът на мобилното приложение, насочено към крайните клиенти, включва екран за вход, интерфейс за текущите поръчки и доставки към тях. Също така панел за създаване или промяна на поръчка. Фигура 2.2 представя процесите под формата на диаграма.



Фиг. 2.2. Диаграма на главен бизнес сценарий (разработка на автора)

Уеб порталът е софтуер насочен към диспечерите, е част от цялостната система за управление на транспорта (TMS). Чрез него могат да се създават поръчки и доставки, като същевременно се сравняват, за да се гарантира, че поръчките се доставят от най-подходящото превозно средство. Уеб порталът служи като инструмент за вземане на решения, с предварително зададени предложения, които могат да бъдат одобрени, или отхвърлени и променени, според гледната точка на диспечера на смяна. Вземайки под внимание текущите събития, подсистемите зад уеб портала насрочват за доставка това, което и когато клиентът е поръчал. Те разчитат на правилна информация за поръчка и актуализация на събития. Целта е да се минимизират разходите.

Обхват на уеб портала включва балансиране на работното натоварване на превозните средства, позволява проследяване и коригиране, както на поръчките, така и на доставките, осигурява предварително зададени решения, на база на които диспечерите могат да коригират и контролират броя на доставките. Диспечерите имат възможност да поправят грешни данни, като говорят с клиентите или шофьорите. Същевременно всички промени се отразяват в мобилното приложение.



Фиг. 2.2. Диаграма на главен бизнес сценарий (разработка на автора)

### 2.1.2. Функционални и нефункционални изисквания към бекенд системата

Ясно дефинираните изисквания са основата на успешен проект, тъй като включват набор от процеси като анализ, спецификация и валидиране. Функционалните изисквания са продуктови характеристики, които разработчиците трябва да внедрят, за да позволят на потребителите да изпълнят своите задачи. Като цяло функционалните изисквания описват поведението на системата при определени условия. Някои от основните изисквания са:

• Регистриране на акаунт;

• Вписване на потребител;

• Отписване на потребител;

• Преглеждане на текущите поръчки;

• Разглеждане на детайлите за определена поръчка и доставки;

• Филтриране на елементите по дата;

• Добавяне на нова поръчка;

• Промяна или премахване на съществуваща поръчка;

• Регистриране на нова доставка към поръчка;

• Промяна или премахване на съществуваща доставка;

Нефункционалните изисквания често се наричат ​​„атрибути за качество“ на системата. Те са критериите за оценка на това как една софтуерна система трябва да работи.

Следващите точки отбелязват някои от основните изисквания:

• Системата трябва да е високо-достъпна и да може автоматично да разширява мащаба, за да отговори на увеличаващия се трафик (също така да намалява мащаба, след като трафикът спадне);

• Трябва да осигурява лесен диагностични дневници, за да помогне при отстраняване на неизправности или други проблеми, които би могли да възникнат по време на работа;

• Трябва да поддържа гъвкав процес на развитие, включително подкрепа за непрекъсната интеграция и внедряване (Continuous integration / deployment);

• Трябва да поддържа междуплатформен хостинг и развитие;

TODO: ИЗИСКВАНИЯ ЗА бързодействие и изпълнение / натоварване / ИЗИСКВАНИЯ ЗА ОБЕМ ДАННИ / брой потребители /

## 2.2. Концептуален модел на системата

Концептуалните модели са абстрактни представяния за това как трябва да протича изпълнението на задачите. Те представят визуално концепция или операция. За визуализиране и конструиране на елементите е използван унифицираният език за моделиране (Unified Modeling Language).

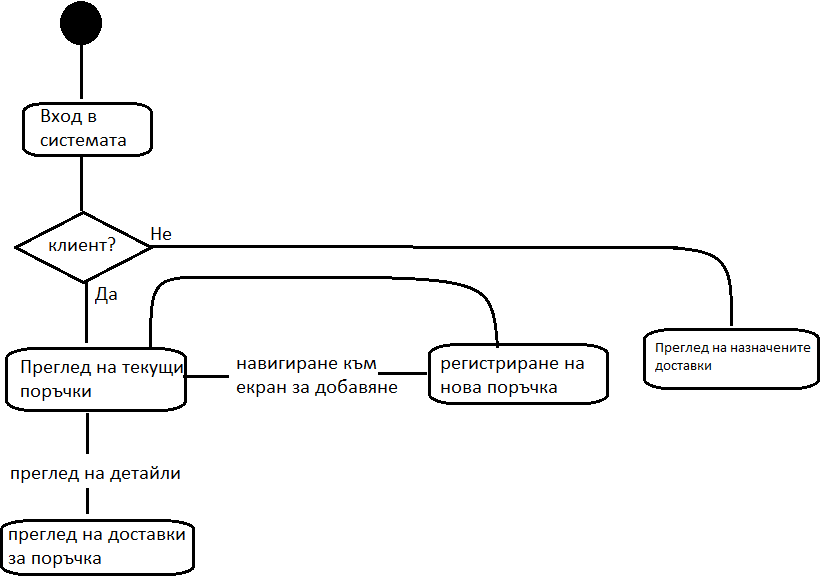
### 2.2.1. Поведенчески диаграми

Поведенческите диаграми идентифицират как различните елементи взаимодействат помежду си.

#### 2.2.1.1 Диаграми за активност UML

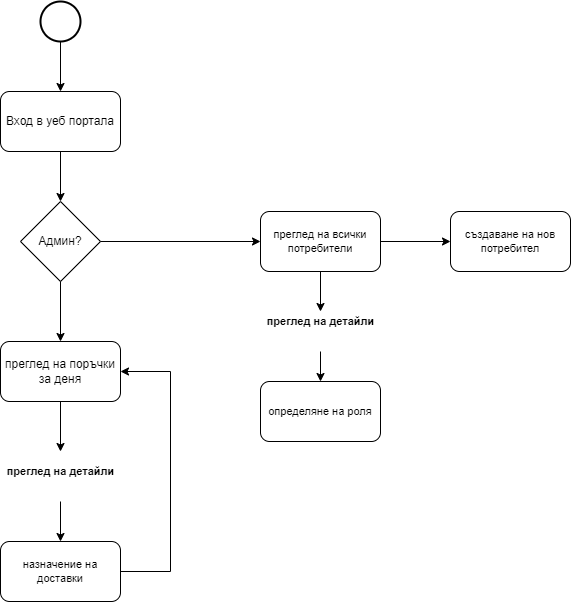
Диаграмите за активност изглеждат много подобни на блок-схемите. Наличието на тези прилики улеснява комуникацията между технически и не-технически лица (stakeholders).

Следната диаграма представя работни потоци и общи операции за мобилното приложение:



Фиг. 2.3. Диаграма на активноста за мобилно приложение. (разработка на автора)

Следната диаграма изобразява потока от операции в уеб портала:

 Фиг. 2.4. Диаграма на активност за уеб портал. (разработка на автора)

#### 2.2.1.2. Диаграма на последователностите UML

Диаграмите на последователностите също са често използвани поведенчески диаграми в UML. Те идентифицират как обектите в система взаимодействат помежду си, за да реализират определена функционалност, като визуализират времевата линия и редът, в който се извършват операциите.

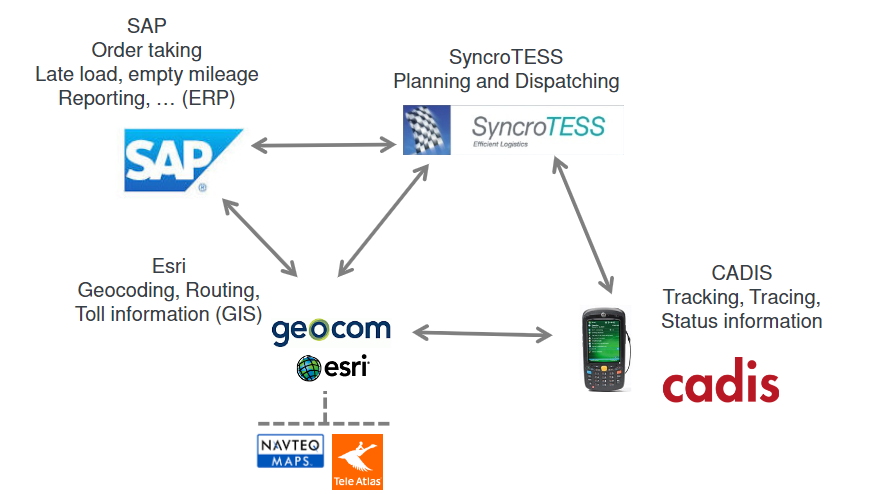
Пример може да разгледаме в контекста на подсистемата за поръчки:



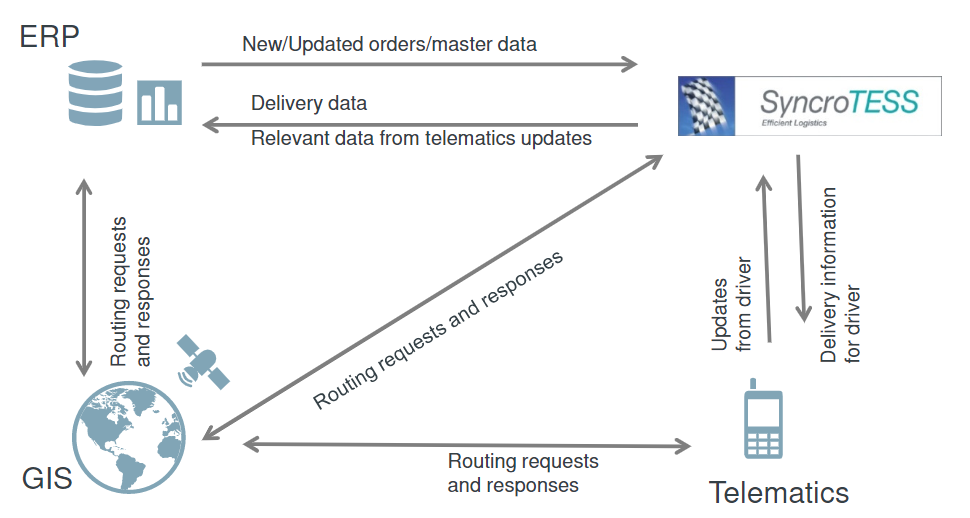
Фиг. 2.7. Диаграма на последователностите. (разработка на автора)

<https://blog.openreplay.com/jwt-authentication-best-practices>

#### 2.2.1.3. Общ преглед на системата и интерфейси



……………………..



…………………….

### 2.2.2. Структурни диаграми

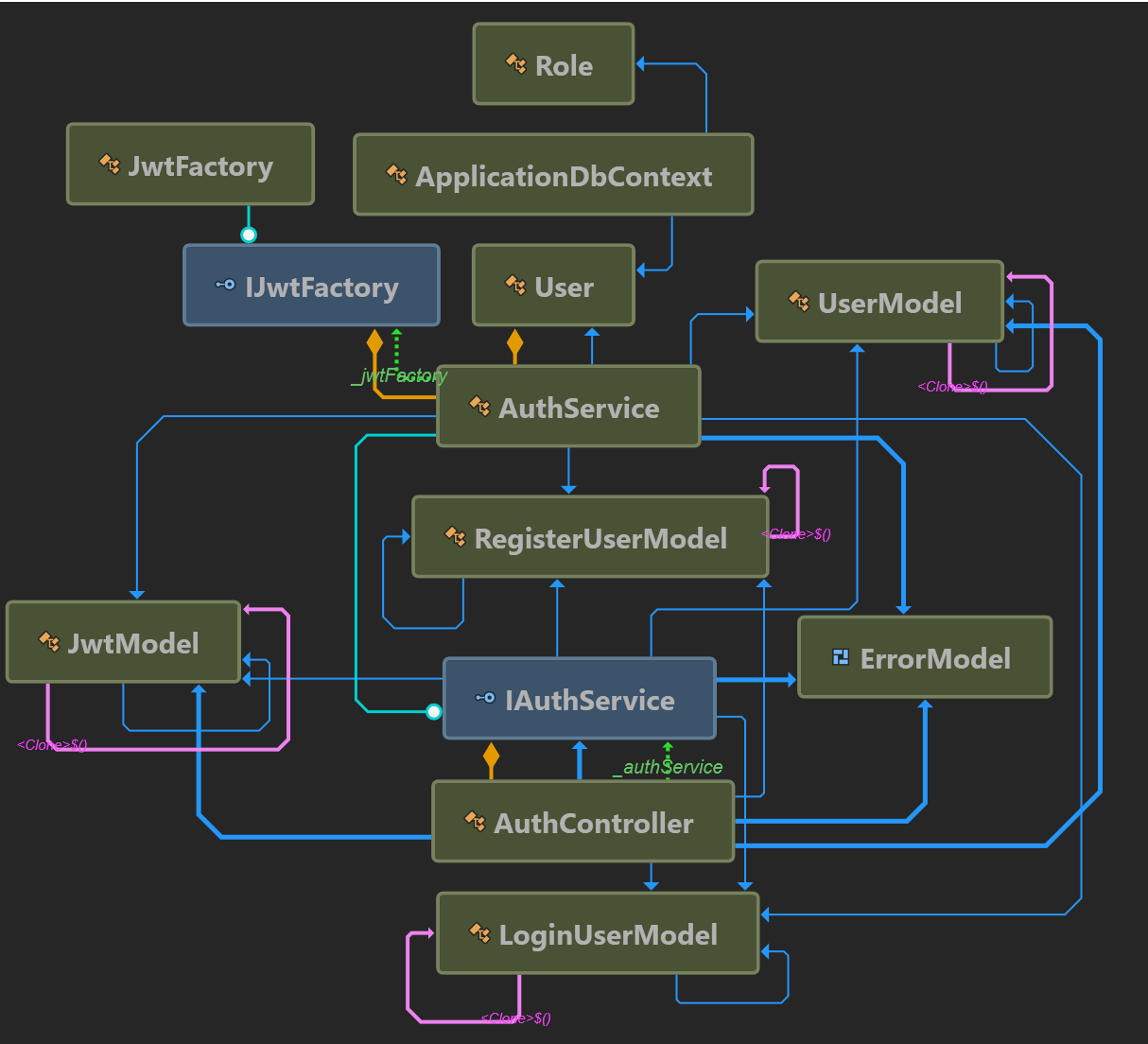
Структурните диаграми помагат за дефиниране цялостната структура системата, подобно на плана, който определя как изглежда една къща. Структурните диаграми моделират как изглежда системата в архитектурно отношение. Те ни помагат да дефинираме „речника“ на системата, гарантират съгласуваност от заинтересовани страни в проекта. Идентифицират различни връзки между различните частти.

Структурните UML диаграми изобразяват елементите на система, които са независими от времето и които предават концепциите и как те се свързват помежду си. Елементите в тези диаграми приличат на съществителните в естествения език.

#### 2.2.2.1. Диаграма на класовете UML

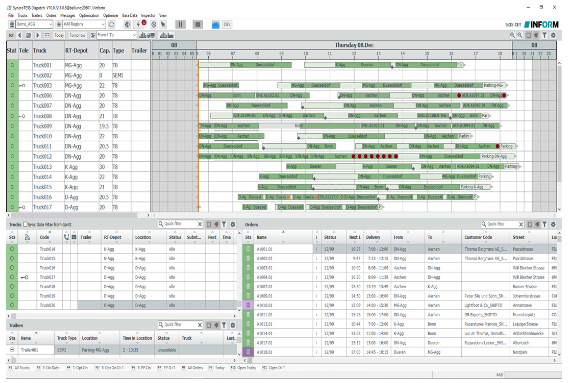
Диаграмите на класове са едни от най-често срещаните, когато става на въпрос за разработката на софтуер. Едно от основните неща, които тези диаграми правят е да идентифицира речника на системата. Например, те определят връзките между обектите, които съответстват на основните съществителни.

Следващата част представя диаграма на класовете, свързани с удостоверяване. Това е процесът на определяне кой има достъп до системата. Елементите от приложението и зависимости, които обслужват тази част са визуализирани на фиг. 2.2. **DbContext** и **ApplicationUser** представляват комбинация от класове, които оперират с базата от данни.  **AccountController** използва тези свойствa чрез **UsersService**, който капсулира логиката по безопасен за използване начин.



Фиг. 2.8. Интерактивна диаграма за БСВС „Разработка на продукт / Планиране на проект”. (разработка на автора)

## 2.3. Функционалност и потребителски интерфейс



Графичен потребителски интерфейс (GUI) Използван от разпределители и планиращи Информация за поръчките, които трябва да бъдат доставени Пълен контрол върху всички камиони

Изглед на приемащия поръчки (OTV) Дава на обслужването на клиенти представа как ще изглежда денят Важна цел: Постижим план до края на деня Определете допустимото ниво на свръхрезервиране (и кажете на поемащите поръчки)

Depot viewer (DV) – Уеб инструмент, използван в рамките на депата, за да видите какво се приближава Дайте актуализации на състоянието на пристигане/напускане на завода и натоварването за камиони без телематика и неинтегрирани инсталации Включвайте/изключвайте камиони без телематика

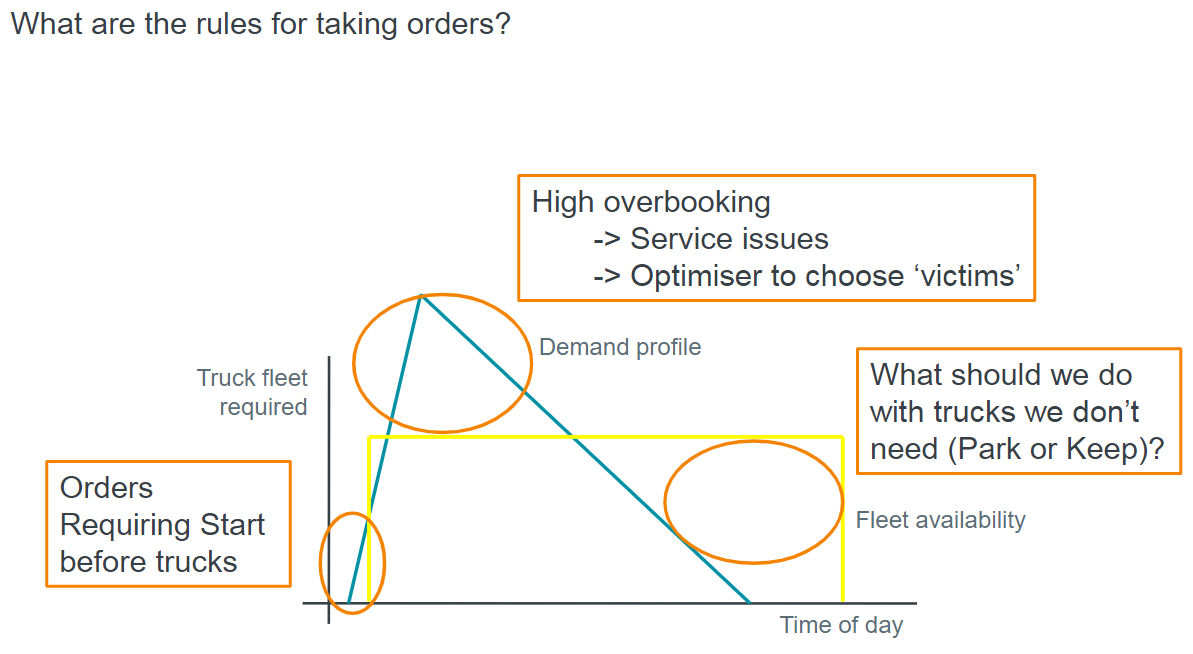
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

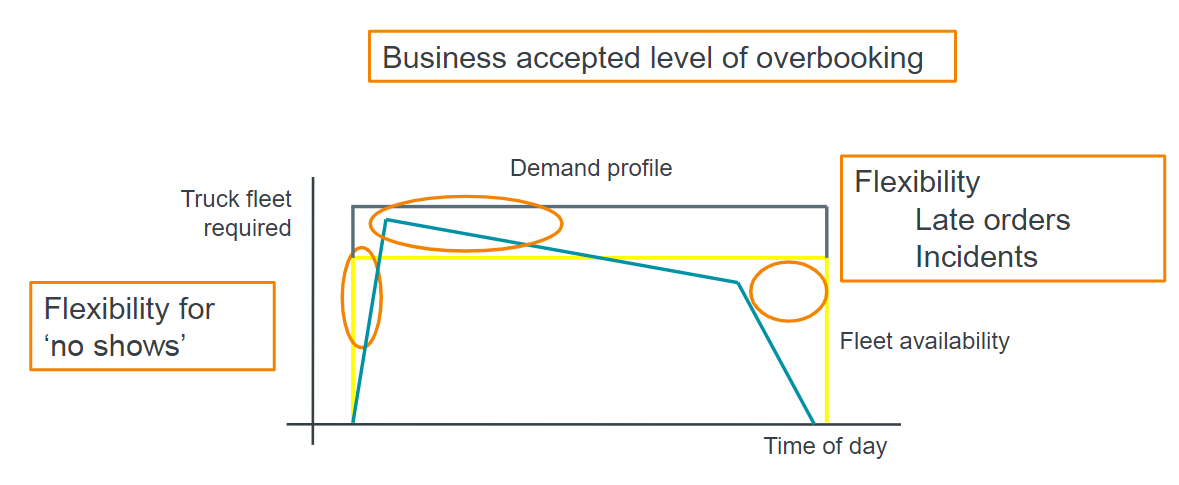
Планиране на камиони●Планиране от началото на деня до края на всяка смяна●Включва пътуване от паркинга до вкъщи през нощта●Може да премести камиони от едно депо в друго.Изпълнение на поръчката●Изцяло навреме, въз основа на информацията в поръчката●Ще се стреми към редовно доставка от клиента●Няма да търси свръхпредлагане на сайта●Ще трябва да се увери, че информацията за поръчката е правилна за времето и разпределението на товара

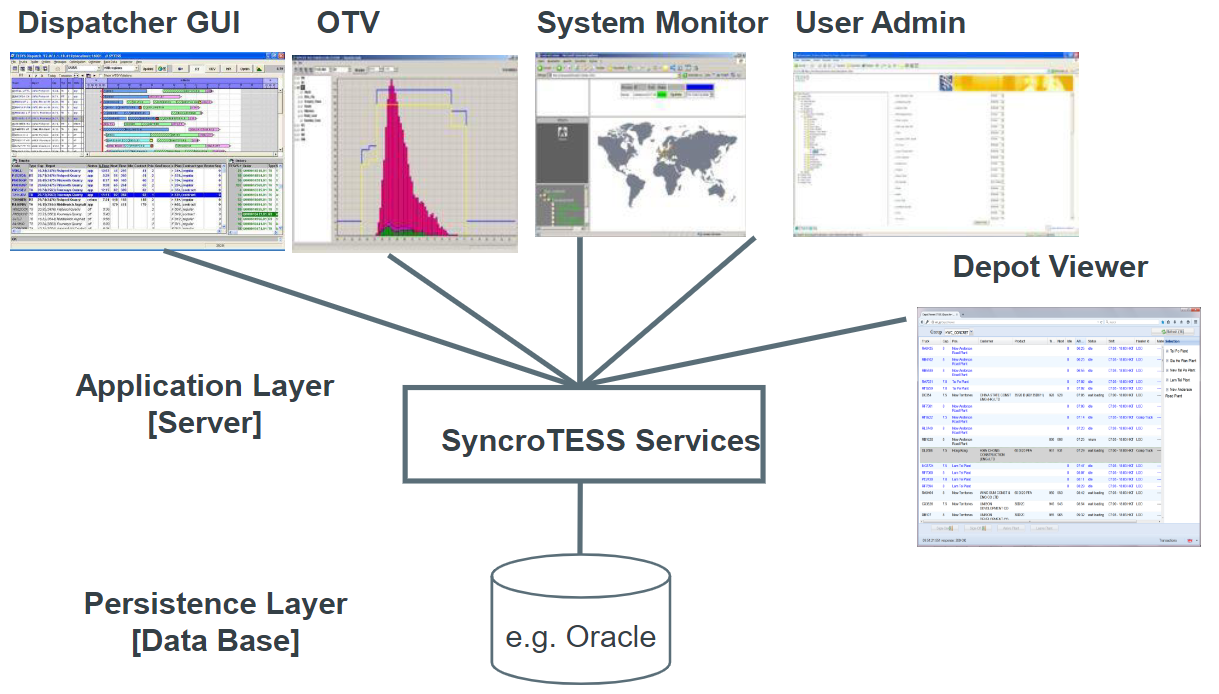
\_\_\_\_\_\_\_\_\_\_\_\_

Отговорности●Управление на профила на поръчката●Отговаряне на въпроси на клиенти относно доставките●Връзка с разпределителите за подпомагане на заявки за промяна на поръчката●Връзка с разпределителите●Промени на входящи поръчки●Закъснели заявки за поръчки●Поемане на „нормални“ поръчки●Известие за късно зареждане

правила за приемане на поръчки







Дава изглед на високо ниво какви са изискванията на камиона●Трябва да се използва като основа за Demand SmoothingView върху същите данни, които виждат всички останали●OTV изгледът се променя с промяната на поръчките и извършването на доставкиЕ неоптимизиран изглед на поръчките●Отразява времената от поръчките●По-малко изразителни за големи времеви прозорци за доставка и минимално разпределение на товара Също така позволява лесен достъп, за да видите кога са доставени поръчките●Възможно е да се отговори на въпроси на клиенти като●„Кога ще бъдете тук?“●„Колко вече сте доставено?'