



ИКОНОМИЧЕСКИ УНИВЕРСИТЕТ –
ВАРНА
ФАКУЛТЕТ ИНФОРМАТИКА
КАТЕДРА ИНФОРМАТИКА

Йордан Иванов Йорданов

**Облачна информационна система за управление на
поръчките от клиенти в производствено
предприятие**

ДИСЕРТАЦИЯ

за присъждане на образователна и научна степен „доктор“
по докторска програма
„Информатика“, професионално направление
"Информатика и компютърни науки“

Научен ръководител: доц. д.н. Павел Петров

ВАРНА

СЪДЪРЖАНИЕ

Списък на използваните съкращения	4
Въведение.....	5
Глава 1. Проблеми на информационното осигуряване при управление на поръчките от клиенти	9
1.1. Управление на веригите от поръчки и доставки и тяхното приложение в системите за планиране на ресурси.....	9
1.1.1. Специфики при управлението на веригите от доставки в производствено предприятие.....	9
1.1.2. Същност и принципи на системите за планиране на ресурси в производствено предприятие.....	32
1.2. Възможности за дигитализация на процесите по управление чрез прилагане на облачни технологии	54
1.2.1. Определение и качества на облачните системи	54
1.2.2. Управление на бизнес процесите чрез ориентиран към домейн дизайн	68
1.3. Специфики при управление на поръчките от клиенти в производствено предприятие	69
1.3.1. Киберсигурност и защита на данните предоставени публично на клиентите	69
1.3.2. Предизвикателства при управлението на клиентските поръчки в производствените организации	78
Customer satisfaction	79
Глава 2. Архитектура на облачна система за управление на поръчки от клиенти.....	83
2.1. Ключови бизнес процеси и дейности свързани със системата за управление на поръчките	83
2.2. Концептуален модел на системата.....	90
2.2.1. Поведенчески диаграми	90
2.2.2. Структурни диаграми.....	94

2.2.3. Архитектурни диаграми	96
2.3. Функционалност и потребителски интерфейс.....	98
2.4. Комуникационни модели между програмните интерфейси	107
2.4.1. Синхронна комуникация	108
2.4.2. Асинхронна комуникация	116
2.4.3.3. Комуникационни модели за достъп до бекенда	117
Глава 3. Изграждане на облачна система за производствено предприятие Holcim Group	121
3.1. Обща характеристика на дейността на компанията Holcim Group.....	121
3.1.1. Основни бизнес процеси в компанията.....	121
3.1.2. Стимулиране на продажбите чрез цифрови технологии.....	123
3.2. Избор на технологични средства за разработка и операции	124
3.4. Приложение на избраните технологии за изграждане на инфраструктурата в облачно базирана среда	132
3.4.1 Хранилищата за данни в подсистемите	132
3.4.2. Софтуерното внедряване и поддръжка в облачна среда	134

Списък на използваните съкращения

Акроним	Пълно Наименование
API	Application Programming Interface
ERP	Enterprise Resource Planning,
SCM	Supply Chain Management

Въведение

Успехът в производствения бизнес до голяма част зависи от ефективното администриране на поръчки и продажби, управление на веригите за доставки (SCM), случващи се в постоянно променяща се бизнес среда. Въпреки нарастващото значение на електронната логистика и системите за планиране на ресурси на предприятие (ERP) в производствения сектор, много организации се борят с проблеми като неефективност, липса на видимост в реално време, ограничена мащабируемост и проблеми с интеграция с други бизнес процеси. Тези проблеми често водят до намалена производителност, високи разходи, проблеми при обслужването на клиентите и други. Базираните в облак системи се отличават с потенциал за подобряване възможностите на ERP и SCM, като предлагат мащабируеми, гъвкави решения в реално време. Тези системи трансформират начина, по който се управляват поръчките, от момента, в който клиентът подаде поръчка, през производствения процес до доставката на крайния продукт. Интеграцията с ERP системи осигуряват ефективно, всеобхватно решение за управление на ключови бизнес функции. Целта на дисертацията е да докаже как облачните системи могат да бъдат успешно внедрени и интегрирани със съществуващи ERP системи и как те могат да допринесат за цялостно подобряване представянето на бизнеса.

Актуалността на изследваната тема се обуславя от тенденцията облачните технологии да се превръщат в инструмент за стратегическа трансформация и дигитализация на бизнеса. Облачните платформи позволяват бърза реализация на иновативни идеи. Това предимство поставя компаниите една стъпка пред конкурентите. Същевременно, дигитализацията предоставя възможности за интегриране и оптимизиране на веригите за поръчки и доставки. Също така, изследването предлага базова представа за ползите и проблемите от внедряването на подобни системи, обхващайки икономически и технологични въпроси, които са от първостепенно значение.

Основната теза на изследването е, че логистичният процес, ориентиран към крайният потребител, се характеризира със значителна комплексност, включвайки множество информационни системи и ресурси като хора, превозни средства и други. Тези ресурси са изцяло насочени към доставянето на продукт до крайните клиенти, използвайки процеси за комуникация в реално време.

Целта на дисертационния труд е да оцени въздействието на базираните в облак системи за управление на поръчки от клиенти върху ефективността на планиране на ресурси и управление на веригата за доставки в производствено предприятие. Същевременно, цели да изследва и опише проблеми и решения, представени от дигитализирането на процесите, като управление на сложността на бизнес логиката, анализиране на проблемите с киберсигурността, защитата на данните, логистичната координация и други. Научният труд анализира решенията, целящи да помогнат на производствените предприятия да подобрят конкурентоспособността си на пазара.

За изпълнение на целта са поставени следните **задачи**:

- 1) да опише взаимосързаността между управлението на веригите за доставки, електронната логистика и системите за планиране на ресурсите в производствена организация, ориентирани към обслужването на клиенти;
- 2) да разгледа възможностите за дигитализиране управлението на процедурите по приемане и обработка на заявки от клиенти, експедиция на готовата продукция, избор на канали за разпределение, поддържане на връзки и логистично обслужване;
- 3) да изследва процесите по проектиране, внедряване и въздействие на система, имаща за цел да бъде доставен продукт, в точното количество и качество, на определени предварително място, време и потребител, с оптимални разходи;
- 4) да проучи проблемите, пред които са изправени производствените организации, като осигуряването на мониторинг на

информацията за материални потоци в реално време, подобряване на оперативната ефективност, осигуряване на мащабируемост в контекста на управлението на веригата за доставки, оптимизира комуникацията между служители и клиенти;

Обект на дисертационното изследване са методите за събиране, съхраняване, обработка и разпространение на информация, свързана с поръчките от клиенти в производствено предприятие, въз основа на които се извършват логистични видове дейности като товарене, превозване и разтоварване на стоково материални ценности.. Това обхваща всички системи, дейности, включени в управлението на потока от стоки и информация от точката на произход до точката на потребление..

Предмет на изследването са внедряването и въздействието на разпределена, базирана в облак, информационна система, ориентирана към оптимизацията на ресурсите в производствено предприятие насочена към вземане на обосновани управленски решения и координация. В частност дизайна, внедряването, работата и резултатите от система, целяща ефективно придвижване през фазите на производствено-логистична дейност. Базирана на уеб услуги, системата работи върху множество процеси, сървъри (хостове), мобилни и клиентски приложения. Всяка услуга се изпълнява в отделен процес като контейнер, разположен в кълстер от виртуални машини. Отделните приложения взаимодействват помежду си с помощта на протоколи за комуникация. Разгърнатата в облачната платформа, инфраструктурата се управлява от инструменти за оркестрация, непрекъсната интеграция и внедряване от високо ниво.

За да изпълни поставените цели и задачи и да предостави цялостен анализ на темата, изследването **прилага смесен подход**, съчетавайки количествени и качествени данни с методите на логически и статистически анализ, моделиране и проектиране на информационни системи и др. Проучването започва с преглед на литературата, за да определи теоретичната си основа. Втора част включва концептуално решение от

високо ниво, което да съсредоточава върху всички основни потребителски, бизнес и ИТ изисквания. Трета част представя качествено изследване, включващо приложимо предложение за система интегрирана в компания, обмисляща приемането на базирана в облак система за управление на поръчките от клиенти.

Структурата разделя научния труд на 3 части. След въведението, глава първа полага теоретични основи, които да дадат бизнес контекст на изследването. Създава се базово разбиране, което дава възможност за развитие и реализация. Главата включва въведение в управлението и анализа на веригите за доставки, управление на риска, логистиката, доставянето, както и на обслужването на клиенти, стратегическото планиране, прилагането на облачни технологии. Следвайки целите и задачите на дисертацията, глава втора разглежда концептуално и архитектурно софтуерно решение. Вследствие, глава трета технологични насоки за изграждане и интегриране на системата в произведено предприятие от глобален мащаб.

Глава 1. Проблеми на информационното осигуряване при управление на поръчките от клиенти

Първа глава представя теоретични основи в областта на стопанската и информационна логистика, управлението на веригите от поръчки и доставки, системите за планиране на ресурси, както и проблемите и решенията по дигитализация на процесите чрез прилагане на облачни технологии. Тази глава полага основи за следващите фази по проектиране, изграждане и внедряване на облачна информационна система.

1.1. Управление на веригите от поръчки и доставки и тяхното приложение в системите за планиране на ресурси

Взаимосвързаността на глобалната икономика налага ефективна координация на различни бизнес процеси. Ключова област е пресечната точка на управлението на веригата за доставки, информационната логистика, продажбите и дистрибуцията в планирането на ресурсите на предприятието. Разбирането на тези връзки е от първостепенно значение за рационализирането на бизнес процесите, подобряването на удовлетвореността на клиентите и стимулирането на цялостния бизнес растеж.

1.1.1. Специфики при управлението на веригите от доставки в производствено предприятие

Специфики при управление на поръчките от клиенти се свързват с управлението на веригата от доставки. В литературата съществуват множество различни дефиниции за термина „верига на доставките“. Според Chopra and Meindl „веригата за доставки се състои от всички етапи, които пряко или непряко участват в изпълнението на заявките на клиента. Веригата на доставки включва не само производителя и доставчиците, но и превозвачите, складовете, търговците на дребно и самите клиенти“.

Ganeshan and Harrison, пък дефинират веригата за доставки, като: „мрежа от съоръжения и възможности за дистрибуция, която изпълнява функциите на доставка на материали, превръщането на тези материали в междинни и готови продукти и разпространението на тези готови продукти на клиентите.“. Друга дефиниция, която откриваме „веригата за доставки е съвкупност от процеси и ресурси, необходими за извършване и доставка на продукт на крайния потребител.“

Настоящия труд се спира на определението на Бл. Благоев, който дефинира понятието, като „ясно очертана верига от свързани двойки логистични звена „доставчик – получател“ (структурирани подразделения на фирмата и/или логистичните й партньори), по която конкретната стока и/или услуга се доставя на крайния потребител в съответствие с неговата заявка и изисквания“.

Веригата за доставки е канал за ефективно движение на материали, продукти, услуги или информация от доставчици към клиенти. Основната му задача е да достави правилния продукт или услуга на клиентите в точното време, място, количество, състояние и цена. Дългосрочният успех на една компания зависи от нейната ефективна верига за доставки. Фигура 1.10 илюстрира права веригата за доставки, започва със суровини, движеща се от фирмата производител към крайния потребител.



Фиг 1.10. Права верига за доставки.

Една по-сложна верига за доставки включва доставчици, производители, дистрибутори, търговци на дребно и крайни потребители.



Сложността се увеличава с разстоянията, езиковите бариери и културните различия, което го прави от съществено значение за дългосрочен успех.

Обратната логистика като част от веригата за доставки

Ако веригата за доставки се движи от фирмата производител към крайния потребител, то противоположното движение – от крайния потребител към производителя – е всъщност обратната логистика. Обратната логистика се характеризира с някои основни дейности, като връщане на продукти от потребителите, които продукти по една или друга причина не са успели да задоволят потребността на дадения потребител, връщане на продукти с цел ремонт, рециклиране, замяна и др. Това дали продуктите ще бъдат ремонтирани и препродадени или ще бъдат рециклирани зависи както от спецификата на самия продукт, така и от причините за неговото връщане. В книгата си „Обратни вериги на доставка“, Surendra Gupta прави сравнение между правата и обратна верига за доставки, като извежда основните им прилики и различия, някои от които

са описани в следната таблица.

Сравнение между права и обратна верига за доставки

Права верига за доставки	Обратна верига за доставки
Базирана на оптимизиране на печалбата и разходите	Базирана на екологичните принципи и закони, както и на оптимизирането на печалбите и разходите
Сравнително по-лесно и ясно прогнозиране на търсенето на продукти	По-трудно прогнозиране за връщане на продукти
По-малко вариации в качеството на продукта	Високи различия
Времето и стъпките за обработка са добре дефинирани	Времето и стъпките за обработка зависят от състоянието от върнатия продукт
Стоките се транспортират от едно място до много други места	Върнатите продукти се събират от много места и пристигат в едно преработвателно предприятие
Оценката на разходите е по-лесна заради счетоводните системи	Определянето и представянето на разходите е сложно
Скоростта е конкурентно предимство	Скоростта не е критичен фактор
Опаковката на продуктите е стандарта	Опаковката е високо променлива
Стандартна структура на самият продукт	Структурата на продукта е модифицирана
Прозрачност на процесите, дължащи се на проследяването на продуктите в реално време	По-малко видими процеси, поради липсата на възможности за информационна

[Supply Chain Fundamentals: Understanding the Basics \(udemy.com\)](#)

Demand

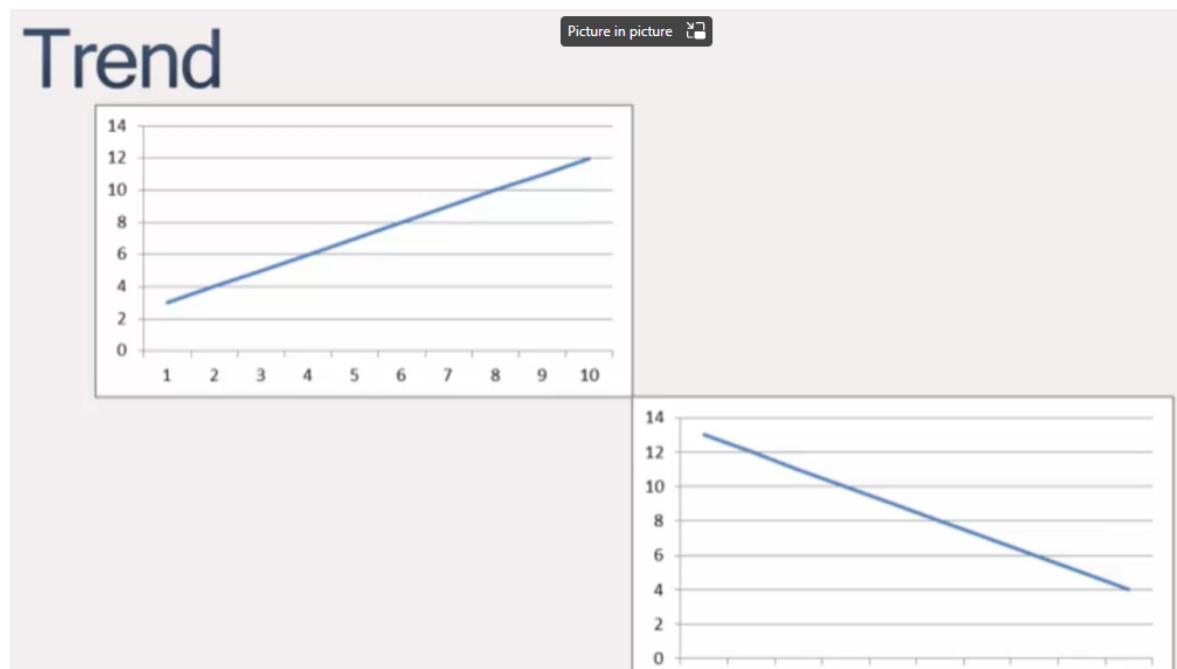
Търсенето се отнася до необходимостта от артикул, продуктов компонент или услуга и се дели на независимо и зависимо. Независимото търсене се осигурява от клиента, докато зависимото търсене е свързано с търсенето на друг продукт.

Разбирането на поведението на търсенето помага да се предвиди, тъй като независимото търсене обикновено се осигурява от клиентите. Веригите за доставки имат за цел да имат правилния продукт наличен в точното време и място. Например, универсален магазин трябва да предвиди броя велосипеди, които

очаква да продаде за даден период от време, за да ги поръча от производителя и да ги направи налични за покупка.

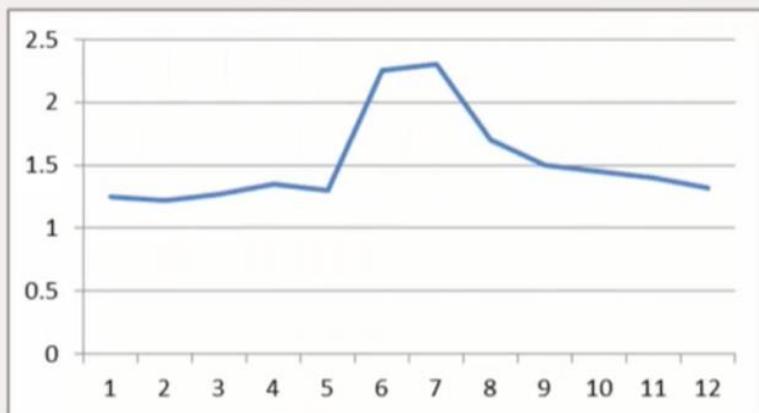
Експертите в областта трябва да предвидят нуждите на клиентите и да действат предварително, като прогнозират търсенето, като използват исторически данни, колебания и разбираят четири основни модела на търсене: тенденция, случайни колебания, сезонни колебания и цикличност.

Тенденциите са постоянни.



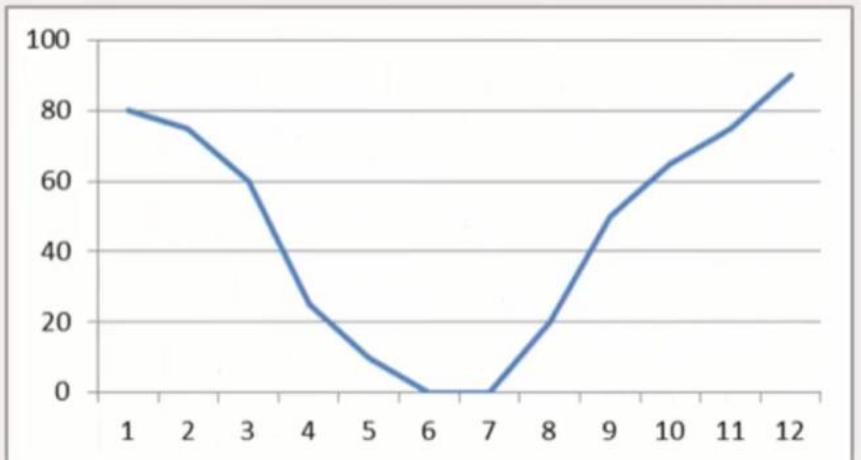
, случайните колебания са причинени от случаен събития

Random Fluctuation



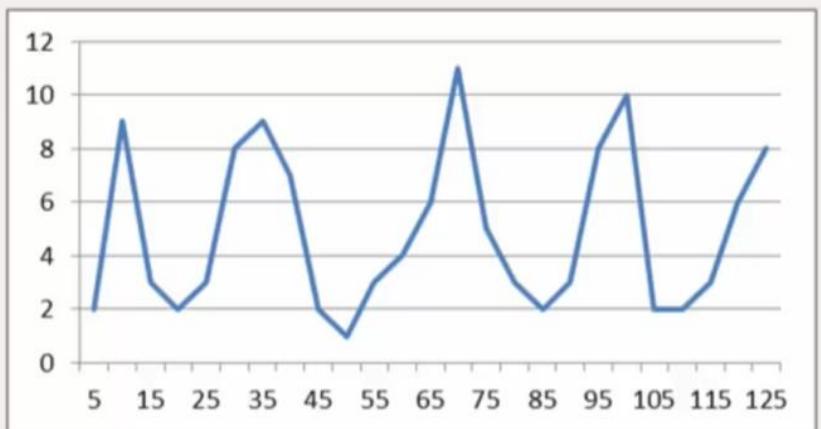
, сезонните колебания се появяват за кратки периоди от време

Seasonal Fluctuation



а цикличните модели се появяват за периоди от повече от една година.

Cyclical



Фокусът на научния труд е върху елемента на обществените поръчки. Това е покупката и продажбата на продукти. Много пъти, веригата за доставки се асоциира с процеса на закупуване. Но всъщност, това е само един от елементите в управлението на веригата за доставки. След това е управлението на жизнения цикъл на продукта, с етапи по внедряване, растеж, зрялост и спад. Процесите на планиране и доставка трябва да вземат предвид етапите на развитие. На пример, в автомобилната индустрия всяка година излизат нови модели, като същевременно бъдещите са в етап на планиране.

Управлението на веригата за доставки е и контрол на материали, информация и финанси, докато те се придвижват от производител до краен клиент. То включва координиране и интегриране на потоци както вътре, така и между компаниите. Практиката за управление на веригата за доставки взема пример до голяма степен от областите на индустриталното инженерство, системното инженерство, управлението на операциите, логистиката, информационните технологии. Търси се фина настройка на тези елементи, които подпомагат на продукт да бъде реализиран възможно най-бързо.

Управлението на веригата за доставки не е просто купуване и продажба на продуктите. Изброените по-горе елементи, помагат за увеличаване на производителността, ефективността, навременността, намаляване на разходите. Управлението на веригата за доставки се състои от взаимосвързани мрежи и канали, комбинирани в предоставянето на продукти и услуги, изисквани от крайния клиент във веригата. Корпоративните информационни системи дават възможност за комуникация с доставчици, клиенти по начин, който помага за подобряване на преминаването на материали, консумативи и услуги. Те дават възможността да се отговори на нуждите на клиента по най-бърз начин.

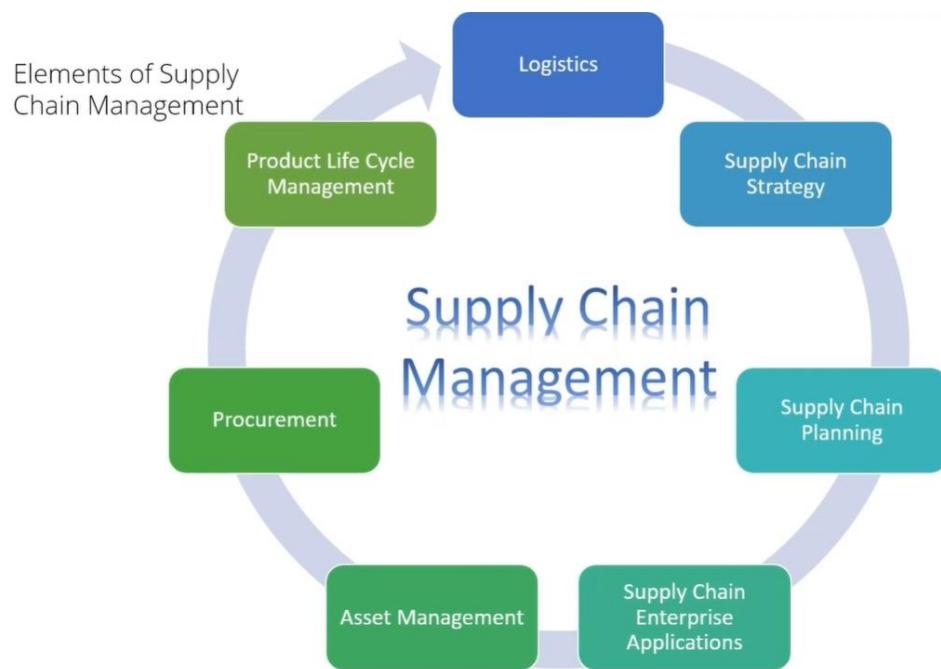
Управлението на веригата за доставки се дефинира като проектиране, планиране, изпълнение, контрол, наблюдение на дейностите с

цел създаване на нетна стойност, изграждане на конкурентна инфраструктура, използване на световната логистика, синхронизиране на предлагането с търсенето и измерване на ефективността глобално.

Бизнесът изисква справяне с предизвикателствата, които могат да възникват в най-различни ситуации. Поток от информация, помага на решението, в кой момент трябва да има повече или по-малко запаси, какви видове продукти и други. Интегрираното планиране и изпълнение на процеса, необходим за оптимизиране на потока от материали, информация и финансов капитал, включва планиране на търсенето, снабдяване, производство, управление на запасите и съхранение, транспортиране или логистика, връщане на излишни или дефектни продукти. В тези усилия се използват както бизнес стратегия, така и специализиран софтуер за създаване на конкурентно предимство.

елементи

Управлението на веригата за доставки е координирането на различни елементи, включвайки логистика, стратегия, планиране, управление на активи, доставки и жизнен цикъл на продукта. На фигура 1.10 е илюстриран модел на елементите.



Фиг 1.10. Модел на елементите, съставящи управлението на веригите за доставки.

Логистика

Първият елемент е логистика, включваща много хора, съоръжения и доставчици. Логистиката е процес на планиране, реализиране и контрол на движението и съхранението на потока сировини и свързаната с това информация, от мястото на доставяне до мястото на приемане. European Logistics Association (ELA) формулира определение: „Организация, планиране, контрол и реализация на придвижването на стоковия поток от проектирането и закупуването, през производството и разпределението до крайния потребител с цел удовлетворяване изискванията на пазара с минимални операционни и капиталови разходи“. Логистиката е сложна съвкупност от разнообразни, но обвързани в единна система дейности и

операции по придвижването на материалните и съпътстващите ги потоци. Те се идентифицират с размерност, начална и крайна точка на движение, дължина на изминавания път, скорост и време на движението, време на престой, вид и особености на използваните транспортни средства, условия на транспортиране, номенклатура, условия на договорите за покупко-продажба.

Логистична система може да бъде дефинирана като „относително устойчива съвкупност от звена, взаимно свързани и обединени от единно управление на логистичния процес за реализация на корпоративната стратегия на организация на бизнеса (В.И. Сергеева, 2004, Корпоративая Логистика). Основна цел на тази система е доставката на стоки и изделия на зададено място, в необходимото количество и асортимент, максимално подгответи за производствено потребление при зададено равнище на разходи. Логистичната система е съвкупност от логистична мрежа и съответна система за администриране, изградени в конкретно предприятие. Логистична мрежа на фирмата включва пълната съвкупност от обвързаните с фирмата и помежду си в логистични вериги и звена, осигуряващи изпълнението на всички логистични функции и операции по придвижването на материалите и съпътстващите ги потоци за изпълнение заявките на потребителите.

Информационният поток е съвкупност от устно, документално или предавани по друг начин данни за конкретния материален поток (Юлиан Василев, 2016). Един от най-често срещаните информационни потоци в логистиката е потокът „поръчки от клиенти“. Всяко предприятие прилага индивидуален подход при неговото организиране.

Изпълнението на логистиката и планирането на производството са важни компоненти на веригата за доставки. Планирането на производството съгласува търсенето с производствения капацитет и създава график за производство и доставки файлове за готов продукт и компонентни

материали. Основните данни предоставят основен обект за транзакции и могат да се използват за определяне на потока от материали, доставчици и клиенти във веригата за доставки.

стратегия и планиране на веригата за доставки

След като те бъдат определени, се изгражда стратегия и планиране на веригата за доставки, с които предприятието да се конкурира на пазара.

Информация от анализа на предлагане и търсене, определя, в кой момент трябват повече запаси или например кои продукти трябва да биват предлагани в определен момент. Интегрираното планиране и изпълнение на процеса, необходим за оптимизиране на потока от материали, информация и финансов капитал, включва планиране на търсенето, снабдяване, производство, управление на запасите и съхранение, транспортиране или логистика, връщане на излишни или дефектни продукти.

Логистичният мениджмънт е тази част от веригите на доставки, която включва планиране, изпълнение и ефективен контрол на пласмент и съответните потоци от точката на зареждане до точката на потребление за удовлетворяване на изискванията по потребителите (CSCMP, Washington, 2006).

Логистичният мениджмънт е процесът на планиране, организиране, координиране и регулиране на развитието на логистичната дейност във фирмата. Той е система от постоянно вземани стратегически, тактически и оперативни управленски решения свързани с изпълнението на логистичните операции, въздействия върху структурни звена, доставчици и др, изграждащи логистичната мрежа, за осигуряване на установените и потенциални конкурентни предимства чрез ефективно придвижване на материалите и съпътстващите ги потоци.

В съвременната икономика логистичното планиране е задължителен елемент на произведено стопанска дейност на фирмено планиране. Основна задача на стратегическия логистичен план е реализация ла

логистичната стратегия.

Оптимизационните задачи в логистичното планиране за многобройни и разнообразни, но като основни могат да се разграничават отделни функционални области. Управление на поръчките – регламентиране и съчетаване на компонентите на цикъла на изпълнение на поръчките (приемане, обработка, доставка), избор на технически средства и технологии на приема, обработката и комплектоване на поръчките, внедряване на електронен обмен на данни, параметри на качеството на обслужване.

Логистичният подход и ефективното управление на материалните и съществуващите ги потоци изискват координирана реализация на разнообразните функции и операции изпълнявани в логистичната система. Логистичната координация се осъществява не само на стратегическо равнище. Тя е и ежедневна оперативна дейност, тъй като е с огромно въздействие както върху ритмичността на стопанската дейност, така и върху ефективността на самата логистика и успешната реализация на стратегията.

Производството е процес на трансформиране на основни материали в готови стоки, с основна цел създаване на стойност за производителите и потребителите. Тази процедура изискава материални, трудови и административни разходи, като носи печалба. Производството е основен компонент на икономиката на всяко общество, а ефективното производство на продукти и услуги повишава жизнения стандарт. Има множество производствени среди, включително ИТО (проектирано по поръчка), МТО (производство по поръчка), ATL (сглобяване по поръчка) и MTS (производство на склад). Всяка стратегия има своите предимства и недостатъци, но всички те се стремят да създадат стойност както за клиента, така и за организацията. Чрез прилагането на тези стратегии предприятията могат да подобрят качеството и доставката на своите продукти, което в крайна сметка е от полза за потребителя и бизнеса.

корпоративните системи

Друг важен елемент са корпоративните системи, като тези за планиране на ресурсите, управление и следене на превозните средства. Те дават необходимата информация, която да подпомогне вземане на решения. ERP системите дават възможност за комуникация с доставчици и клиенти, по начин, чрез който те обменят информация, подобряваща преминаването на материали, консумативи и услуги. Тази взаимосвързаност бива разгледана от гледна точка на клиентското обслужване. Управлението на веригата за доставки се дефинира като проектиране, планиране, изпълнение, контрол, наблюдение на дейностите по веригата за доставки с цел създаване на нетна стойност, изграждане на конкурентна инфраструктура, използване на световната логистика, синхронизиране на предлагането с търсенето и измерване на производителността в световен мащаб. Глобалните компании се стремят към пазари, където имат предимството както с доставките, така и с клиентите.

Управление на активи

Управление на активи, е следващия елемент, като той включва инфраструктурата, която е необходима за поддържане на верига за доставки, било то сгради, складове, оборудване, хора, а също и инвентар.

SAP оценява ефективността на веригата за доставки чрез различни показатели, като надеждност на доставката, отзивчивост, гъвкавост, разходи и управление на активи. Надеждността на доставката измерва качеството на продукта, реакцията измерва скоростта на доставка на продукта, Six Sigma измерва гъвкавостта, гъвкавостта измерва времето за реакция, оценката на разходите разглежда общата верига за доставки и логистичните разходи, разходите за транспорт и дистрибуция, а управлението на активи оценява управлението на активи.

Снабдяване

Логистиката на снабдяването е една от основните логистични

подсистеми. В случая, снабдяването е свързано с управлението на потока от стоки и услуги, от точката на произход до точката на потребление. Изключително важно е точният продукт да се достави на точното място в точното време. Това гарантира конкурентно предимство пред други, които развиват подобни бизнеси, както и максимизиране на клиентската стойност. Управлението на веригата за доставки е един от елементите на добре управлявана организация.

Управление на веригата за доставки се използва за осигуряване на непрекъснато снабдяване, управление на договорните задължения и предотвратяване на прекъсвания в доставките. Също така управление на риска, спазването на организационните разпоредби и споразумения. В зависимост от естеството на бизнеса, трябва да се спазват индустритални и правителствени изисквания, да се извличат анализи на поръчките.

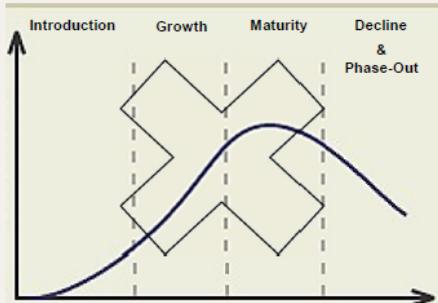
TODO:

[Strategic Cost Management: Procurement and Supply Chain 2022](#)
[\(udemy.com\)](#)

life cycle

Жизненият цикъл на продукта, илюстриран на фиг.1.11 се състои от пет етапа: въвеждане, растеж, зрялост, спад и елиминиране.

Product Life Cycle



- Introduction Stage
- Growth Stage
- Maturity Stage
- Decline Stage
- Phase-Out Stage

Фиг 1.10. Модел на елементите, съставящи управлението на веригите за доставки.

Фазата на въвеждане се характеризира с високи разходи за реклама и промоция, високи разходи за проектиране и производство и високи цени. Продуктът навлиза в етап на растеж, където производството се увеличава и производствените разходи намаляват в резултат на по-високи производствени обеми. Докато други копират техните версии, възниква конкуренция и се произвеждат конкурентни продукти. Когато един продукт достигне зрелост, той е утвърден и утвърден.

Фазата на спад настъпва, когато продажбите и печалбите намалеят и производителите представят нови продукти, за да стимулират търсенето и да намалят разходите. Етапът на поетапно спиране настъпва, когато производството спре, като в този момент поддръжката може да продължи или да бъде възложена на външни изпълнители. Продължителността на жизнения цикъл варира в зависимост от естеството на продукта, потребителското търсене и наличието на по-нови иновации. Като производители е от решаващо значение да разберат къде съществуват техните продукти на кривата и да вземат съответни решения. В идеалния случай продуктите на компанията трябва да бъдат равномерно разпределени във всички етапи, като въвеждането на нови продукти служи като

постоянен двигател на бъдещия растеж. Максимизиране на зрелите продукти за увеличаване на печалбите и разпознаване на намаляващи продукти, за да се определи кога да бъдат премахнати.

Управлението на веригата за доставки представлява прегледа над информацията за материалите и финансите, докато те се движат в процеса от производител до потребител. Състои се от взаимосвързани мрежи, канали и предприятия, комбинирани в предоставянето на продукти и услуги, изисквани от крайния клиент. Практиката за управление на веригата за доставки до голяма степен приема практика от областите на индустриалното, системното инженерство, управлението на операциите, логистиката, информационните технологии. Ролята на този модел е насочена към продажбата на продукти, увеличаване на производителността, ефективността, навременността, намаляване на разходите за преработване.

Управление на веригата за доставки служи за увеличаване на печалбата, увеличаване на паричния поток, подобряване обслужването на клиентите, намаляване на оперативните разходи, подобряване на финансовите позиции.

Елемент на логистичната мрежа е логистичната верига, синоним на понятието верига на доставките. Тя се дефинира като линейно подредена съвкупност от обвързани помежду си логистични звена за придвижване на конкретни материални и съпътстващи ги потоци, свързани с производството и доставката на заявения от потребителя краен продукт.

Проектирането на логистичната система е един от основните стратегически проблеми на логистиката и основна задача на логистичния мениджмънт. Системата трябва да осигурят ефективна реализация на логистичната функция на фирмата в съответствие с динамиката на вътрешната и външна среда.

Обслужването на потребителите и целият комплекс от логистични дейности следва да се осъществява на планова основа, съобразно финансовите възможности на фирмата, в рамките на възприета логистична технология и реализираща я информационно-управляваща система.

компании

Поглед над топ 20 на веригите за доставки на Gartner за 2022 г., показва някои много познати имена.

Таблица: The Gartner Supply Chain Top 25 for 2022

RankCompany

- | Rank | Company |
|------|--------------------|
| 1 | Cisco Systems |
| 2 | Schneider Electric |
| 3 | Colgate-Palmolive |
| 4 | Johnson & Johnson |
| 5 | PepsiCo |
| 6 | Pfizer |
| 7 | Intel |
| 8 | Nestlé |
| 9 | Lenovo |
| 10 | Microsoft |

Това, което е общо сред тях, е силното присъствие на пазара, силната способност да реагират на промените и силното позициониране в тяхната област. Всичко това е свързано с управлението на веригата за доставки и цялостната им бизнес стратегия. Крайният резултат е максимизиране на клиентската стойност и наличие и поддържане на устойчиво конкурентно предимство.

Управление на веригата за доставки е насочена към възвръщаемост на инвестиция и отговаряне на изискванията на клиентите. Компаниите използват управление на веригата за доставки, за да управляват договорните задължения, осигурят непрекъснато снабдяване и да избегнат прекъсвания на доставките. Цели поддържане на добри отношенията с доставчици и клиенти, така че ако някакъв вид проблем възникне, да може да се разреши бързо. Също така помага в управляване на риска, в контекста на спазване организационните, както и специфичните за индустрията разпоредби и споразумения. В зависимост от естеството на бизнеса, има индустритални и

правителствени изисквания, които трябва да се спазват.

Друг елемент е създаването на единен изчерпателен изглед (и анализ) към доставчиците на обществените поръчки. Големите компании разглеждат базата за доставки и прилагат определен тип критерии или контролен списък, на които доставчиците трябва да отговарят. Всичко това е с цел доставчиците да са в добро състояние.

Управление на веригата за доставки, по същество е насочена към увеличаване на печалбата и паричния поток както и качеството на обслужване на клиентите, намаляване на оперативните разходи, подобряване на финансовото състояние.

- 1) [Supply Chain Fundamentals: Understanding the Basics \(udemy.com\)](#)
- 2) [SAP : Supply Chain Logistics in R/3 \(udemy.com\)](#) => pecycpc
- 3) Nice to have [Supply Chain Management for Beginners \(udemy.com\)](#)
- 4) Close topic: [Fundamentals in Oracle Transportation Management \(OTM\) Cloud \(udemy.com\)](#)
- 5) [Supply Chain: Planning of Resources & Detailed Scheduling \(udemy.com\)](#) => бест пра

Добри практики

Управление на риска

Всяка верига за доставки е изправена пред рискове от различни източници, като доставчици, логистика и пазарно търсене. Рисковете могат да бъдат под различни форми, като например закъснели дати на доставка, проблеми с качеството и много други. За предотвратяване и подготовка за неизвестни рискове, всички известни и вътрешни рискове трябва да бъдат идентифицирани, създавайки регистър на рисковете, оценете всеки риск, задайте вероятност за неговото възникване и въздействие и определете приоритета на всеки риск. Добавете вероятността, въздействието и рейтинга на риска към регистра на риска срещу всеки риск. Разработете план за реагиране на риска, като определите собственик и осигурите одобрение и финансиране за отговора. Наблюдавайте плана за реагиране и го актуализирайте, ако е необходимо.

Сертификати

Поддържането на сертификат и спазването на признатите в индустрията стандарти, като ISO-9001, ISO-14001, ISO-26000, ISO-31000, създава доверие и отличава една компания от конкурентите. Тези стандарти обхващат различни показатели, включително качество, околната среда, социална отговорност и риск.

Показатели на веригата

Можем да разделим показателите за измерване на ефективността в три категории: ориентирани към клиента, финансови и оперативни. Ориентираните към клиента показатели се фокусират върху предоставянето на точна поръчка, скорост на изпълнение и гъвкавост. Финансовите показатели, като разходи за печалба и инвентар, помагат за контролиране и стимулират намаляване на разходите. Оперативните показатели измерват производителността на ежедневните операции, както и на по специални като управлението на активи и ресурси.

Качествените показатели проследяват производствени дефекти, връщания, гаранционни проблеми и точност на документите. Показателите за производителност измерват продукцията спрямо входа, а показателите за управление на активите измерват използването на инвентара. Остарелите наличности могат да се управляват чрез преглед на правилата за поръчки или закупуване и извършване на необходимите промени.

Показатели на веригата за доставки

Показателите са от решаващо значение за измерване на ефективността и извършване на подобрения. Има три категории: ориентирани към клиента, финансови и оперативни. Фокусираните върху клиента показатели се фокусират върху предоставянето на перфектна поръчка, скорост на изпълнение и гъвкавост. Финансовите показатели, като разходи за печалба и инвентар, помагат за контролиране на разходите и стимулират намаляване на разходите. Оперативните показатели измерват ежедневните операции, включително качество, производителност и управление на активи.

Качествените показатели проследяват производствени дефекти, връщания, гаранционни проблеми и точност на документите. Показателите за производителност измерват продукцията спрямо входа, а показателите за управление на активите измерват използването на инвентара. Остарелите наличности могат да се управляват чрез преглед на правилата за поръчки или закупуване и извършване на необходимите промени.

Непрекъснато усъвършенстване

Непрекъснатото подобреие включва непрекъснати усилия за подобряване на процесите и продуктите, което води до подобрена удовлетвореност на клиентите и дългосрочен успех. Компаниите инвестираят в текущи подобрения, като потенциалните спестявания надвишават разходите. Бенчмаркингът, конкурентен или най-добър в класа, помага да се поставят цели и задачи.

Стратегия победител

Победителите в поръчките са характеристики, които привличат клиенти, предлагайки достъпност, качество и бързина. За да се отличат, фирмите могат да използват стройни или гъвкави стратегии, като се фокусират върху ефективността и гъвкавостта. Балансирането на тези очаквания е от решаващо значение, тъй като конкуренцията също може да се фокусира върху подобни аспекти.

Извод

В обобщение, управлението на веригата за доставки е от важно значение за предприятията, за да осигурят непрекъснати доставки, да управляват договорните задължения, да заздравят връзките с доставчиците и да поддържат конкурентно предимство. Чрез внедряване на управление на веригата за доставки компаниите могат да увеличат печалбата, да подобрят обслужването на клиентите, да намалят оперативните разходи и да подобрят финансовото си състояние. Следващата подточка включва ERP модулите, свързани с заявки за покупка, управлението на материалите и други.

1.1.2. Същност и принципи на системите за планиране на ресурси в производствено предприятие

В бизнеса се използват корпоративни информационни системи, за да се обхванат всички логистични процеси. Един от аспектите на планирането на материалните потоци означава да се използва ERP система за изчисляване на количеството сировини и материали, които да се поръчат (Юлиан Василев, 2016). Корпоративните информационни системи (или „система за управление на бизнеса“) целят да осигурят потребителите с адекватна информация, необходима за ефективно организиране и реализация на логистичните функции и операции по придвижването на материалните и съпътстващи ги потоци. Също така предоставят нужната информация на логистичните мениджъри, изпълнители и конкретните потребители. Информационните и комуникационни технологии се характеризират със ускорен и ефективен пренос на информационни потоци в рамките на логистичната система, надеждно съхранение, обединяване и разделяне на данни в бази.

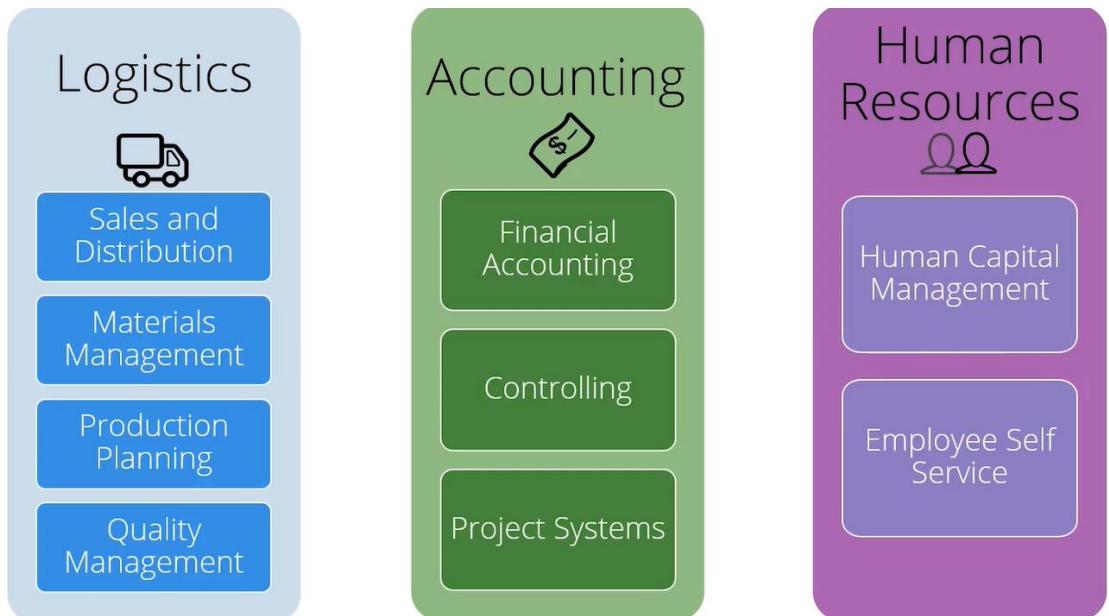
ERP информационните системи позволяват ефективното планиране на дейността на предприятие, в т.ч. разходи за обновяване на оборудването и инвестициите в производството на нови изделия (Банабакова, В. К, 2019). Те произхождат от RP технологията (Requirements/resource planning - планиране на потребностите/ресурсите). Основната цел на RP е съкращаване на количеството на запасите от материали, незавършено производство и готова продукция, съгласуване на графика на доставките с работата на отделните производствени звена и процеса на закупуване (Филипов, Ст. Г., 2019). С логистичните технологии и тяхното приложение се цели осигуряване на оптимални решения в логистичната система (Благоев, Бл., 2010). Логистичната технология се определя като стандартизирана последователност (алгоритъм) на изпълнение на отделни логистични функции, и/или процеси в логистичната система или в отделни нейни функционални области (Сергеева, В., 2004). Някои от тези алгоритми и поддържащите ги информационно управляващи системи са получили и нормативна регламентация. Такива са MRP I и MRP II, за които са разработени и утвърдени международни стандарти ISO (Стоянов, Ст. Хр., 2019).

Подобренията от въвеждането на ERP се изразяват в увеличаване броя на изпълнените поръчки, повишаване качеството на логистичното обслужване към клиентите, възможности за промени в обема на поръчките, съкращаване на времето от поръчката до доставката и други (Банабакова, В. К, 2019).

SAP бива пуснат за първи път преди 50 години в Германия. В рамките на тази система се управляват всички функционални области на даден бизнес: човешки ресурси, финанси и функции за закриване на отчетен период, продажби, управление на клиенти, фактуриране и задължения, управление на инвентара, логистика, и други (). Всяка отделна функция, от която едно предприятие може да има нужда, е напълно достъпна и интегрирана в SAP. Това е водещата ERP система на пазара. Използван е от около 87% от глобалните организации. SAP поддържа производствена (например Harley Davidson), нефтената и газовата индустрия (Shell), държавни органи и други.

В зависимост от компанията и много други фактори, внедряването на SAP може да отнеме дълго време и много ресурси. За сметка на това той предлага инструменти, които автоматично получават, съгласуват и извършват определени действия. Това позволява съредоточаване върху други задачи, които могат да помогнат за растежа на компанията. SAP разполага с огромни количества данни, които могат да бъдат използвани за вземане на ефективни и сигурни бизнес решения.

SAP е разделен на различни области, които работят заедно, наречени модули, като има два вида: технически и функционални. Програмисти или ИТ персонал работят над техническите модули. Функционалните модули, които се използват от потребителите на системата, съдържат вградени транзакции, съответстващи на бизнес процесите. SAP има много функционални модули, като на фигура 1.1. са представени 3 от най-използваните.



*Фиг. 1.1. Модули и подмодули на SAP. Източник: Pluralsight<
<https://www.pluralsight.com/sap>>, [18.10.2022]*

Първата група е от модул Логистика. Подмодула „Продажби и дистрибуция“ е насочен към обработка и доставка на поръчки за продажби. Модулът за управление на материалите включва закупуване и управление на инвентара. Производственото планиране съдържа основни данни като спецификация на материалите, маршрути и изпълнение на планирането на материалните изисквания. SAP Quality Management управлява контрола на качеството и несъответствията. Финансовото счетоводство на SAP включва главната книга, дължимите сметки и вземанията. SAP Controlling е мястото, където се извършва анализът на себестойността на продукта и рентабилността. Project Systems управлява счетоводните аспекти на планирането, мониторинга и функциите за изчисляване на разходите по проекта. Управлението на човешките ресурси обхваща целия цикъл на един служител в една компания, от наемане до прекратяване, включително ведомост. Функционалните модули са тясно интегрирани. Те ефективно изпращат информация помежду си, което е една от най-силните страни на SAP ERP системата.

SAP се състои от два типа данни: основни и транзакционни. Основните данни са градивните елементи за всички транзакции, като

клиенти, доставчици, активи, материали и други. Те са относително статични. Необходими са специални разрешения за да се манипулират или създават. Данните за транзакциите, като продажби, покупки и фактури, се променят непрекъснато. Всеки SAP модул има своя собствена независима организационна структура, която определя взаимоотношенията между различните работни групи и отдели. Следващата таблица подчертава някои, но не всички аспекти на организационните структури.

*Таблица 1.1.
Организационни структури в SAP
(адаптирано от автора по Croll, 2022)*

Finance	Sales and Distribution	Materials Management	Human Capital Management
Chart of Accounts	Sales Organization	Plant	Company Code
Company	Distribution Channel	Storage Location	Personnel Area
Company Code	Division	Purchasing Organization	Personnel Subarea
Business Area	Sales Area	Purchasing Group	

Модула за финанси съдържа **сметкоплан**, който изброява всички сметки и се основава на счетоводни правила, определени от държавата. Следва **компания**, на чието ниво могат да се създават индивидуални финансови отчети. По-долу са **фирмените кодове**. Една компания може да има множество кодове и всеки фирмени код може да има **множество бизнес области**. Пример за бизнес област в рамките на фирмени код би било **производство**.

При продажбите и дистрибуцията **търговската организация** е на най-високо ниво и цялото отчитане на продажбените дейности се извършва на ниво търговска организация. След това е **каналът за дистрибуция**, представляващ начин, по който се достига до клиентите. Следващият компонент е **дивизия**, свързана с обработката на конкретна продуктова линия. Една компания може да има едно подразделение, което продава потребителски продукти и отделно подразделение за консултантски услуги.

Комбинацията от търговска организация, дистрибуционен канал и подразделение се нарича **търговска зона**.

На първо ниво в модула за управление на материалите стои **заводът**. Той може да бъде производствено съоръжение, дистрибуторски център или дори офис. **Местата за съхранение** в заводите са физическите места, където се складират запасите. **Организациите за закупуване** водят преговори и дейности по доставки от доставчици. Организациите за закупуване могат да се справят с доставките за множество фирмени кодове или могат да бъдат ограничени и да извършват покупките за конкретен завод. **Организациите за покупки** често се разделят на групи за покупки, които се занимават със специфични аспекти, като специфични материали в рамките на процеса на закупуване.

В управлението на човешките ресурси отново се среща **фирмен код**, който е собствена независима счетоводна единица. **Персоналните области** стоят в рамките на фирмения код, като те самите се разделят на **подобласти**. Организационните структури определят взаимоотношенията между различните работни групи и отдели. Те биват конфигурирани първоначално и обикновено остават статични, освен ако дадена компания не придобие друга или се откаже от части от своя бизнес.

Manufacturing

Start by discussing the role of manufacturing in a business and how it fits into the supply chain.

[Supply Chain Fundamentals : Logistic & Transportation \(udemy.com\)](#)

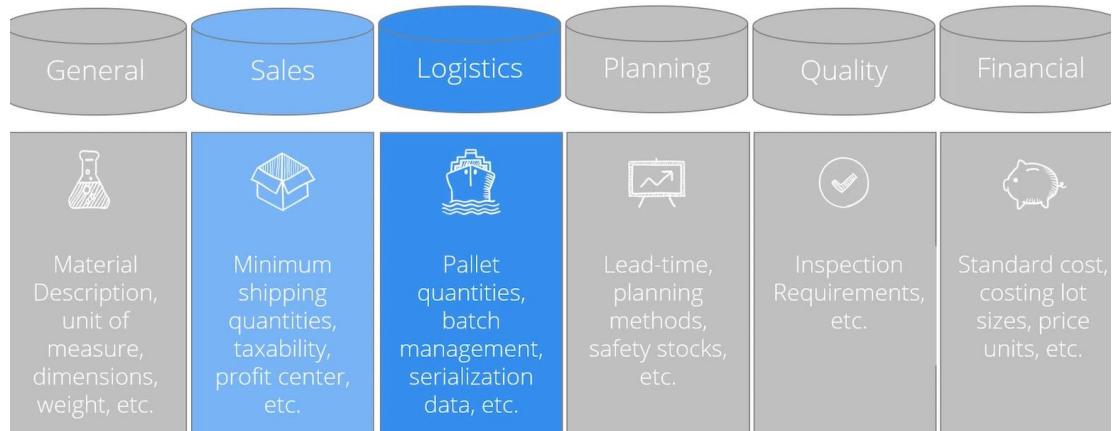
Sales/Distribution

Then, introduce how sales and distribution link to manufacturing, where the finished goods are distributed to the customers.

Discuss traditional systems for sales/distribution and their limitations.

[Supply Chain Fundamentals : Logistic & Transportation \(udemy.com\)](#)

Фокусът на дисертацията е върху модула за продажби и дистрибуция. Цялата функция на този модул е да „продава стоки и услуги на клиентите на предприятието“. Основен обект на модула за продажби и дистрибуция е „материал“. Той може да се закупи, произведе, продаде, върне и/или прехвърли. Фигура 1.2. представя данните, които са част от този запис: продажби и логистика, количества за материали и доставки и други.



*Фиг. 1.2. Данни, част от записа за материал в модул за продажби и дистрибуция. Източник: Pluralsight<
<https://www.pluralsight.com/sap>>, [18.10.2022]*

Вторият най-критичен обект от гледна точка на продажбите и дистрибуцията е главният запис на клиента. Той представлява субект, на който се продават стоки и/или услуги. Част от атрибутите на запис на клиент са: името и адреса на клиента, условия на плащане, специфични опции за ценообразуване, които може да се прилагат само за този клиент, както и различните партньорски функции. ERP системите поддържат няколко версии на клиент. Първата партньорска функция представлява субектът, на който продаваме стоки и услуги (sold-to party). Партньор за доставка (sold-to partner) представлява мястото, където изпращаме стоки или услуги. С други думи, това е адреса за доставка на клиента, който може да е различен от адреса на купувача. Партньорът за фактуриране представлява къде трябва да бъде изпратена фактура. Местоположението, на което се изпращат фактури, може отново да бъде напълно различно от това на адреса за доставка. Функцията партньор на платеца представлява субектът, който отговаря за

плащането на фактура. В много случаи и четирите функции може да имат едни и същи данни, но в някои случаи тези четири функции са не само различни физически адреси, но и напълно различни обекти. Фигура 1.3. илюстрира на кратко четирите вида:

Sold-To Party (SP)	Ship-to Party (SH)	Bill-To Party (BP)	Payer (PY)
The entity to which you are selling goods/services.	The entity to where you are shipping the goods or delivering services.	The entity to where you are sending the billing document/invoice.	The entity responsible for remitting payment of a billing document.

*Фиг. 1.3. Видове партньорски функции. Източник: Pluralsight<
<https://www.pluralsight.com/sap>>, [18.10.2022]*

Ценообразуването в SAP е свързано с различни видове условия. Всеки тип условие се свързва с различен тип цена. Например, цената на артикул, е цената, която е публикувана в продуктовия каталог. Възможно е обаче да има специфична за клиент цена. В SAP тези настройки се поддържат като отделен тип условие. Типа състояние позволява да се поддържа ценообразуване на различни нива. Например, каталожна цена за един материал, е различна в зависимост от това къде се доставя материала. Това се определя, когато се настройва ценовия запис в този тип условие. SAP разполага с отчети за цените. Те позволяват сравняване и анализиране на всякакви ценови условия въз основа на всяка комбинация от критерии. Това е инструмент, който позволява да се анализира всяко поддържано ценообразуване. Таксите са друг вид запис на условие, който трябва да бъде създаден. Ако ценообразуването, което влиза в договорената позиция, не е очакваното ценообразуване, тогава техника за отстраняване на неизправности е анализа чрез доклад на цените. Следната фигура показва пример за такъв доклад.

Material Price						
Sales Org. 0002 Distr. Channel 01		CHARLOTTE, NC FOOD				
Material						
CnTy S	Scale quantity	UoM	Amount	Unit per UoM	Valid From	Valid To
2891 K004 PMIN			2,300.00-	USD	07/03/2017	07/03/2099
			2,300.00	USD	08/17/2017	12/31/9999
Sales Org. 0002 Distr. Channel 01		CHARLOTTE, NC FOOD				
Material						
CnTy ReSt S	Scale quantity	UoM	Amount	Unit per UoM	Valid From	Valid To
2891 PRO0			2,300.00	USD	1 EA	08/13/2017 08/13/2017
Sales Org. 0005 Distr. Channel 12		Germany Frankfurt Sold for resale				
Material						
CnTy ReSt S	Scale quantity	UoM	Amount	Unit per UoM	Valid From	Valid To
T-EC0100 PRO0			59.99	EUR	1 PC	12/04/2006 12/31/9999
T-EC0101 PRO0			59.99	EUR	1 PC	03/27/2006 12/31/9999
T-EC0102 PRO0			59.99	EUR	1 PC	12/04/2006 12/31/9999

Фиг. 1.4: Примерен доклад за цените. Източник: Pluralsight<
<https://www.pluralsight.com/sap>>, [18.10.2022]

Фигура 1.5. представя примерен документ на поръчка за продажба.

Display Standard Order 15994: Overview

The screenshot shows the SAP S/4HANA Order Overview screen for Standard Order 15994. The top navigation bar includes icons for Home, List, Details, and Summary, followed by 'Orders' and a summary icon.

Order Details:

Standard Order	15994	Net value	799.90	USD
Sold-To Party	300711	Holden & Associates / 2300 LOOP 410 S.W. / SAN ANTONIO TX 78245		
Ship-To Party	300711	Holden & Associates / 2300 LOOP 410 S.W. / SAN ANTONIO TX 78245		
PO Number		PO date		

Navigation Tabs: Sales, Item overview, Item detail, Ordering party, Procurement, Shipping, Reason for rejection.

Delivery Information:

Req. deliv.date	D 05/27/2020	Deliver.Plant		
<input type="checkbox"/> Complete dlv.		Total Weight	30	KG
Delivery block		Volume	0.000	
Billing block		Pricing date	05/15/2020	
Payment card		Exp.date		
Card Verif.Code				
Payment terms	ZB01 14 Days 3%, 30/2%, 45 Incoterms	FOB	From the Plant	
Order reason				

All items:

Item	Material	Order Quantity	Un	Description	S	Customer Material Numb
101400-400	<input type="text"/>	10 PC		Motorcycle Helmet - Stand	<input checked="" type="checkbox"/>	

Фиг. 1.5. Примерен документ на поръчка за продажба.

Източник: Pluralsight <<https://www.pluralsight.com/sap>>, [18.10.2022]

Притежава три секции: заглавна част, секция за общ преглед на артикула и секция с подробности. Има стандартни типове поръчки, които са налични по подразбиране. Най-често срещаният е стандартният запис. Този тип поръчка се използва за продажба на действителни стоки на клиент. Следва тип поръчка за връщане. Това е точно обратното на стандартна поръчка. В този случай се връща инвентар от клиент обратно в склада, по някаква причина. Може да е дефект или грешен продукт, но в края на краищата причината е без значение, по важното е дали връщането е

разрешено или не. Те обикновено са обвързани с възстановяване на сумата обратно на клиента. Други типове документи, като дебитни и кредитни известия, са техническа част от модула Продажби и дистрибуция, но тези типове поръчки всъщност не влияят върху инвентара. Въздействието, което тези видове поръчки имат, е върху счетоводните книги. В този случай клиентът бива таксуван или кредитиран. SAP поддържа създаването на персонализирани типове поръчки. Те започват с буквата Z. Такива примери могат да бъдат групова поръчка или тип безплатна поръчка.

Модула на продажби и дистрибуция се интегрира към модула за управление на материалите, за да провери дали има наличен инвентар за количеството на поръчка. Ако е така, той бива разпределен автоматично. Тази функция се нарича проверка на наличността. Друга важна концепция на модула за продажби и дистрибуция са „изчерпаните поръчки“. Това означава, че има повече поръчки, отколкото инвентар. Разпределението на инвентара се случва на принципа: първи дошъл, първи обслужен. Това означава, че първата поръчка, създадена в системата, получи първа наличност в инвентара. Ако инвентара е напълно изчерпан, следващата поръчка, която влиза в системата, се счита за изчерпана поръчка. Проверката на наличността няма да доведе до потвърждение или резервация на инвентар, тъй като наличното количеството е 0. В SAP съществуват поръчки, които могат да бъдат блокирани по различни причини, като например блокирани за преглед, доставка или фактуриране. Също така съществуват и просрочени поръчки, които не са изпратени навреме.

Документа за доставка е транзакционен документ, използван за управление на логистичната страна по процеса на поръчка. Това е дистрибуторската страна на модула за продажби. Документа за доставка позволява да се управляват функциите за опаковане и изпращане. Създаването на доставка за поръчка може да стане чрез ръчен процес, изпълнявайки транзакцията VL01N. Потребителите могат също да създават колективни доставки, като използват транзакцията VL10A. Този метод обикновено се предпочита, тъй като VL10A може да се автоматизира чрез пакетно задание, което да се изпълнява периодично. След като документ за доставка бъде създаден в системата, той автоматично бива свързан с първоначалната поръчка за продажба. Чрез функцията за документооборот, показана на фигура 1.9, може лесно да се види кои доставки са създадени за дадена поръчка. Потокът на документи проследява състоянието, както и навигация за детайлизиране. Ако поръчка има няколко редови позиции, всяка с различна дата на доставка, това води до отделни документи за доставката. Индивидуалните доставки могат да се обработват с транзакцията VL02N, а колективните чрез VL060. Кой метод се използва зависи от това колко доставки се обработват за даден ден, както и с това как са настроени и управлявани физическите процеси. Има транзакции в SAP, които позволяват да се консолидират множество доставки до един и същ клиент, обособени като пратка (shipment). В рамките на пратка можете да се използват и единици за обработка, представляващи гигантски кутии, които опаковат по-малки в една единица, която да се транспортира на палет. Това дава допълнителен слой за проследяване при изпращане на голям микс от продукти. И накрая, възможност за сторниране (премахване на грешка).

Activities Due for Shipping "Sales orders, fast display"

Light	Goods Issue Date	DPri	Ship-to	Route	OriginDoc	Gros	W	Volu	VU	Document
05/18/2020		2	300711		<u>15992</u>	60	KG			
05/14/2020		2	300711	000001	<u>15993</u>	60	KG			
04/21/2020			300711		<u>15987</u>	9	KG			
04/08/2020			1 1000		<u>15991</u>	300	KG			
			2 300711		<u>15990</u>	40	KG			
04/02/2020			1 T-S50A	R00025	<u>15986</u>	40	KG			
04/01/2020			2 300711		<u>15988</u>	60	KG			
03/25/2020			2 300711		<u>15989</u>	40	KG			
03/11/2020		2	1033	R00120	<u>15976</u>	20	KG	0.150	M3	
		2	T-S62F	R00110	<u>15975</u>	14	KG			
03/09/2020			COL112	000001	<u>15953</u>	40	G	0	L	
02/27/2020		2	T-S62A	R00130	<u>60000123</u>	750	KG			
02/24/2020		2	T-S62A	R00130	<u>15961</u>	32	KG	0.200	M3	
02/21/2020		3	1175	R00110	<u>15954</u>	20.0	KG	0.150	M3	
02/20/2020		2	T-S62A	R00025	<u>15962</u>	205	KG			
02/18/2020		2	T-S62B	R00135	<u>15949</u>	205	KG			
		2	T-S62A	R00025	<u>15948</u>	328	KG			
02/17/2020		2	T-S62A	R00130	<u>15947</u>	266.	KG			
		2	T-S62A	R00130	<u>15946</u>	70	KG			
		2	T-S62B	R00135	<u>15946</u>	410	KG			
02/14/2020		2	T-S62A	R00130	<u>15940</u>	307.	KG			
		2	T-S62A	R00130	<u>15938</u>	102.	KG			
02/13/2020		2	T-S62F	R00110	<u>15945</u>	14	KG			
02/12/2020		2	T-S62A	R00130	<u>15939</u>	205	KG			
12/27/2019		2	41		<u>15932</u>	1,00	KG			
12/18/2019		2	T-L64A		<u>15928</u>	1.95	KG	11,25	CC	
12/17/2019		2	2004	R00125	<u>15930</u>	13	KG	90,00	CC	
		2	41		<u>15927</u>	1,00	KG			
12/13/2019		2	T-L64B	R00125	<u>15931</u>	60	KG	108,0	CC	
12/09/2019		2	41		<u>15915</u>	51	KG			
		2	41		<u>15916</u>	5,37	KG			
12/06/2019		2	41		<u>15883</u>	1	KG			

Фиг. 1.6. Екран за дейности на изпращане на поръчка за продажба. Източник: Gartner, Inc. <<https://medium.com/can>>, [09.10.2022]

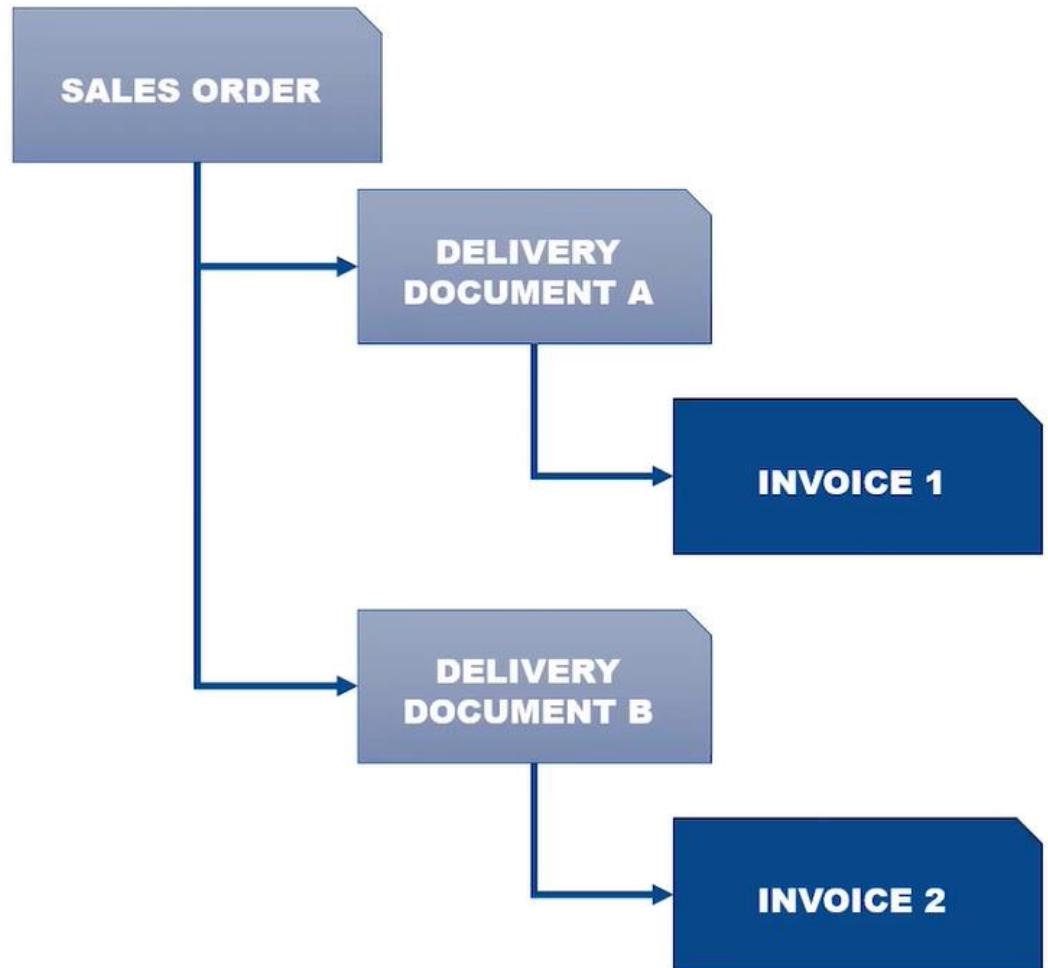
Финансовите осчетоводявания, свързани с продажбите и разпространението на стоки и услуги, използват документ за фактуриране. Документът за доставка предоставя функции, които позволяват поръчаните стоки или услуги да бъдат изпратени до клиент. Документът за фактуриране е реалното фактуриране на клиента за тези стоки или услуги. Обикновено тези два документа са свързани. Много рядко се генерира фактура на ниво поръчка, защото таксуването към клиента се извършва след предоставяне на стоките. Следната фигура представя примерен документ за фактуриране.

Invoice (F2) 90005177 (F2) Display: Overview of Billing Items							
Accounting		Billing documents					
F2 Invoice (F2)	90005177	Net Value	5,500.00	DEM			
Payer	1390	Technik und Systeme GmbH / Schwalbenweg 43 / D-52078 Aachen					
Billing Date	01/03/1997						
Item	Description	Billed Quantity	SU	Net value	Material	Tax amount	
10	Pumpe Stahlguss Etanorm 170-230	1	PC	5,500.00	P-109	825.00	

Фиг. 1.7. Примерен документ на фактура. Източник: Gartner, Inc. <<https://medium.com/can>>, [09.10.2022]

Точно както при създаването на доставка, за създаване на фактури също има два метода: индивидуално чрез транзакция VF01 или колективно чрез VF04. VF04 е предпочитан, защото може да се изпълнява автоматично, търсейки всички документи за доставка, които са били обработени през деня. Фактурите са свързани с документ за доставка и съответна поръчка. Тези връзки са видими през документния поток. Фигура 1.11 показва връзките между документите. Подобно на доставките в т.н. пратки, фактурите могат да бъдат консолидирани в т.н. месечно извлечение. SAP може да съхранява всички фактури до определен етап, например в края на месеца, и след това да ги комбинира в едно извлечение. Това извлечение е документът за изпращане към клиента. Заявката VF05 позволява търсене на фактури по платец, материал или по организационната структура. В счетоводния изглед се намират публикациите на главната книга, които показват връзката между приходите и себестойността на продадените стоки.

След изпращане на фактурата към клиент се очаква и плащане, което се прилага към фактуриране. Продажбите и дистрибуцията продължават до момента на създаване на фактура. Всичко след това попада във финансовия модул.



Фиг. 1.8. Връзки между документите на поръчка, доставка и фактура (Anwar, 2014).

Управлението на веригата за доставки е от съществено значение за бизнеса, за да изгради конкурентна инфраструктура, да използва световната логистика, да синхронизира предлагането с търсенето и да измерва производителността в световен мащаб. SAP предлага интегрирано планиране и изпълнение на процеси, включително планиране на търсенето, снабдяване, производство, управление на инвентара и съхранение, транспортиране или логистика и връщане на излишни или дефектни продукти. SAP е перспективен начин за координиране на активи за оптимизиране на доставката на стоки и услуги и информация. Той включва ключови доставчици и производители, които работят заедно за улесняване на веригата за доставки. Когато работите с продавачи и доставчици, е важно да разберете техните роли и да нямаете доставчик на ключова част или продукт. Програмите за доставчици трябва да имат добре обмислен план за бъдещи нужди, а складовете трябва да бъдат разположени близо до крайните клиенти. Клиентите също трябва да разберат своите нужди, за да планират бъдещи изисквания. SAP е един от елементите на добре управлявана организация, който позволява това. Топ 20 на веригата за доставки на Gartner за 2017 г. показва, че компании като Apple, Procter & Gamble и Amazon имат силно присъствие на пазара, способност да реагират на промените и силно позициониране в своята област. Крайният резултат от наличието на добра бизнес стратегия и добра стратегия за веригата за доставки е максимизиране на клиентската стойност и поддържане на устойчиво конкурентно предимство.

Управлението на веригата за доставки е важен фактор за увеличаване на ливъридж на печалбите и паричния поток, повишаване на обслужването на клиентите, намаляване на оперативните разходи и подобряване на финансовото състояние. Също така е важно да бъдете надежден доставчик на клиентите, тъй като клиентите вече имат много възможности по отношение на това с кого работят.

За подобряване на веригата за доставки е важно да се разбере модулът за управление на материалите и неговите подмодули, като SD, логистично изпълнение и планиране на производството. MRP (материални изисквания и планиране на попълване) е инструмент, използван от SAP за проследяване на инвентара на ниво място за съхранение. В рамките на инвентара първото нещо, което трябва да направите, е да разгледате нивата на запасите в модула за управление на инвентара, околната среда, управлението на материалите и подмодулите на околната среда. В рамките на веригата за доставки е важно да започнете с покупките и доставчиците, когато оценявате нивата на запасите.

Продажбите и дистрибуцията (SD) са част от логистичния модул, който поддържа клиентите от оферите до изграждането на клиента. SD се използва за изграждане на S&OP (планиране на продажби и операции) и функционалност за доставка. Логистичното изпълнение е част от модула за управление на материалите и включва складови операции. Управлението на склада се използва за поддържане на инвентара на ниво контейнер, но може да стане неуправляемо, ако не се дисциплинира в транзакциите.

Ключовите фактори за поддържане на верига за доставки в SAP включват вяра в хората и използване на технологии. Стив Джобс заявява, че технологиите не са достатъчни без хора. Преди да научи SAP, лекторът е работил като купувач за компания за гуми и е изградил собствена верига за доставки, използвайки сажди като компонент. SAP е чудесен инструмент, който помага на бизнеса да управлява веригата си за доставки, като предоставя цялата информация на едно място.

Хората, процесите, дисциплината, комуникацията и обучението са ключови фактори за успех във веригата на доставки. SAP е структуриран процес, който изисква последователност и въвеждане на данни, а изходът е

доста последователен. При управлението на веригата за доставки доставчиците трябва да изпълняват и изпращат поръчките навреме, производството трябва да следва правилните стъпки, за да спази графика, а доставката трябва да следва стъпките за доставяне на продукта до краиния клиент.

В заключение, SAP е основен ключов фактор във веригата за доставки, тъй като позволява комуникация между различни елементи, засегнати от веригата за доставки. Чрез разбирането и прилагането на тези фактори организациите могат да създадат и поддържат стабилна верига за доставки, която отговаря на нуждите на техните клиенти и доставчици. SAP е мощен софтуер, който гарантира информационните потоци в цялата организация и ключовите хора са наясно с влиянието на транзакциите. Обучението е друг ключов фактор, но бюджетните ограничения понякога могат да попречат на завършването на проекта. SAP изискава добре обучен персонал, за да бъде ефективен, а обучението може да бъде формално или неформално, като се провежда нагоре и надолу по веригата на проекта. От съществено значение е да се познават процесите нагоре и надолу по веригата и отговорните за тях.

Управлението на веригата за доставки е от решаващо значение за успеха на една организация и SAP може да помогне да се оцени нейното представяне. Неформално това може да се направи чрез наблюдение на нивата на запасите или получаване на обаждане от Производство. От формална гледна точка това може да се направи чрез методи като Lean Supply Chain, както се намира в модела Scor.

Ефективността на SAP може да бъде оценена чрез качествени и количествени мерки, като удовлетвореност на клиентите и качество на продукта. KPI са ключови показатели за ефективност, използвани за

измерване на инвентара, оборота на инвентара, дните в наличност, средния инвентар, нивото на обслужване и точността на инвентара. Стандартните отчети и стандартният и гъвкав анализ също са достъпни за потребителите за достъп до информацията, от която се нуждаят, за да управляват бизнеса си по-добре.

TODO:

Big one - [SAP : Supply Chain Logistics in R/3 \(udemy.com\)](#)

<https://heidelbergmaterials.udemy.com/course/order-to-cash-o2c-practical-guide-for-business-finance>

Order Management Topic

[SAP SD Advanced Training \(udemy.com\)](#)

Make to order

1.2. Възможности за дигитализация на процесите по управление чрез прилагане на облачни технологии

В последните години облачните технологии се превърнаха във водеща тенденция в софтуерната индустрия. Те предоставят нов начин за изграждане на големи и сложни системи, като по този начин използват пълноценно съвременните практики за разработка на високо-качествен софтуер и налична инфраструктура. Това променя начина на проектиране, интегриране и внедряване на системите. Облачно базираните решения са проектирани да приемат бързо промените, да обслужват голям мащаб от хора и да бъдат устойчиви на всякакъв вид натоварване или хакерски атаки (Vettor, 2022).

1.2.1. Определение и качества на облачните системи

Организацията Cloud Native Computing Foundation предлага следното определение: "*Технологиите, базирани на облак, дават възможност на организациите да създават и изпълняват приложения в модерни, динамични среди като публични, частни и хибридни облаци, чрез мрежи от услуги и микроуслуги. Качества на системите са устойчивост, висока наличност и достъпност, мащабируемост и управляемост, които са от критично значение за много от бизнес единиците. Автоматизацията на тези процеси позволява на инженерите да правят промени, с голямо въздействие, но с минимални усилия.*"

Приложенията стават все по-сложни, като изискванията, от страна на потребителите, стават все повече и повече, главно насочени към бърза реакция и иновативни функции. Проблеми с производителността или повтарящи се грешки вече не са приемливи.

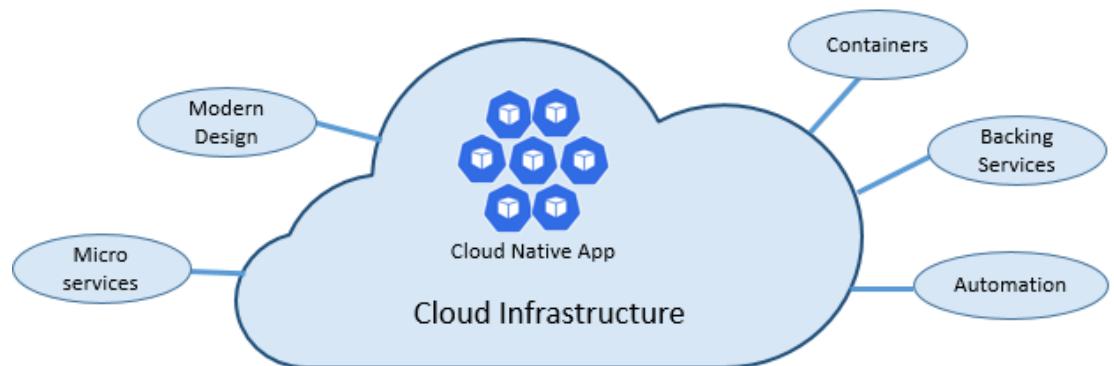
Предимствата на облачните системи поставят бизнеса една стъпка пред конкурентите. Бизнес системите се развиват от способностите на бизнеса да бъдат инструменти за стратегическа трансформация, която ускорява растежа на компанията. Облачно базираните системи се свързват

главно с бързина (Smith, 2022). Незабавното пускане на иновативните идеи на пазара е важна тема за всички модерни компании, например следните компании са приложили успешно тези техники:

- Netflix има над 600 услуги в производствена среда. Стотици пъти на ден се изпълняват нови внедрявания и разгръщания на съществуващи;
- Uber има над 1000 услуги в производствена среда. Обновяват се няколко хиляди пъти всяка седмица;

Както е видно, бизнесът на тези две компании се базира на системи, които се състоят от стотици независими микроуслуги. Този архитектурен стил им позволява бързо да реагират на пазарните условия като постоянно актуализират малки, но важни области. Скоростта на облачния носител се дължат на редица фактори, като на първо място е инфраструктурата на изчислителните ресурси.

На фигура 1.10. са показани пет основополагащи стълба, осигуряващи основата за базирани в облак системи.



Фиг. 1.10. Фундаментни стълбове на облачните системи (Smith, 2022).

Проектирани да процъфтяват в динамична, виртуализирана облачна среда, облачните системи използват широко „платформата като услуга“ (PaaS) изчислителна инфраструктура и управлявани услуги. Те третират основната инфраструктура като за единократна употреба - осигуряват се за минути и се преоразмеряват, мащабират или унищожават при поискване - чрез автоматизация.

Нека разгледаме DevOps концепция наречена: „Pets vs. Cattle“ (Menchaca, 2018). В традиционния център за данни сървърите се третират като домашни любимци (pets), като всеки един от тях представлява физическа машина, която трябва да бъде поддържана. Машабирането се случва като се добавят ресурси към нея. Възникването на проблем в сървър, рефлектира върху всички потребители. Моделът на услугата Cattle е по-различен. При него всеки ресурс се предоставя като виртуална машина или контейнер. Те са идентични и присвояват системни идентификатори като Service-01, Service-02 и т.н. Машабирането се случва като се създадат нови екземпляри. Ако един от тях стане недостъпен, друг поема неговата роля. Облачните системи поддържат този модел. Те продължават да работят, независимо от машините.

Дванадесетфакторното приложение (Wiggins, A., 2017), представено в таблица 1.2, е известна методология за конструиране на облачно базирани приложения. Изготвена от разработчици в Heroku, компания, предлагаша, платформа като услуга, описва набор от принципи и практики, които разработчиците следват, за да създават приложения, оптимизирани за модерни облачни среди. Много практици смятат Twelve-Factor за солидна основа за изграждане на облачни приложения, защото е приложим за всяко уеб-базирано приложение. Системите, изградени на този принцип, могат да се внедряват и машабират бързо, както и да добавят нови или да променят съществуващи функции, за да реагират бързо на пазарните промени.

*Таблица 1.2.
Описание на методологията на дванадесетте фактора
(Wiggins, 2017)*

Factor	Explanation
1 - Code Base	A single code base for each microservice, stored in its own repository. Tracked with version control, it can deploy to multiple environments (QA, Staging, Production).
2 - Dependencies	Each microservice isolates and packages its own dependencies, embracing changes without impacting the entire system.
3 - Configurations	Configuration information is moved out of the microservice and externalized through a configuration management tool outside of the code. The same deployment can propagate across environments with the correct configuration applied.
4 - Backing Services	Ancillary resources (data stores, caches, message brokers) should be exposed via an addressable URL. Doing so decouples the resource from the application, enabling it to be interchangeable.
5 - Build, Release, Run	Each release must enforce a strict separation across the build, release, and run stages. Each should be tagged with a unique ID and support the ability to roll back. Modern CI/CD systems help fulfill this principle.
6 - Processes	Each microservice should execute in its own process, isolated from other running services. Externalize required state to a backing service such as a distributed cache or data store.
7 - Port Binding	Each microservice should be self-contained with its interfaces and functionality exposed on its own port. Doing so provides isolation from other microservices.
8 - Concurrency	When capacity needs to increase, scale out services horizontally across multiple identical processes (copies) as opposed to scaling-up a single large instance on the most powerful machine available. Develop the application to be concurrent making scaling out in cloud environments seamless.
9 - Disposability	Service instances should be disposable. Favor fast startup to increase scalability opportunities and graceful shutdowns to leave the system in a correct state. Docker containers along with an orchestrator inherently satisfy this requirement.
10 - Dev/Prod Parity	Keep environments across the application lifecycle as similar as possible, avoiding costly shortcuts. Here, the adoption of containers can greatly contribute by promoting the same execution environment.
11 - Logging	Treat logs generated by microservices as event streams. Process them with an event aggregator. Propagate log data to data-mining/log management tools like Azure Monitor or Splunk and eventually to long-term archival.

Factor	Explanation
12 - Admin Processes	Run administrative/management tasks, such as data cleanup or computing analytics, as one-off processes. Use independent tools to invoke these tasks from the production environment, but separately from the application.

В книгата *Beyond the Twelve-Factor App* авторът Кевин Хофман описва подробно всеки от оригиналните 12 фактора, като добавя три допълнителни, които отразяват модерен дизайн на облачни приложения.

*Таблица 1.3.
Организационни структури в САП
(Хофман, 2022)*

New Factor	Explanation
13 - API First	Make everything a service. Assume your code will be consumed by a front-end client, gateway, or another service.
14 - Telemetry	On a workstation, you have deep visibility into your application and its behavior. In the cloud, you don't. Make sure your design includes the collection of monitoring, domain-specific, and health/system data.
15 - Authentication/ Authorization	Implement identity from the start. Consider RBAC (role-based access control) features available in public clouds.

Проектирането и внедряването на облачно базирани работни натоварвания може да бъде предизвикателство. Microsoft Well-Architected Framework (Stanford D. et al, 2022) предоставя набор от ръководни принципи, които се използват за подобряване качеството на работното натоварване. Следната таблица представя пет важни стълба на добра архитектурата.

Таблица 1.4.

*Стандартни за добри практики на облачната индустрията
(адаптирано от автора по Croll, 2022)*

Tenets	Description
<u>Cost management</u>	Focus on generating incremental value early. Apply <i>Build-Measure-Learn</i> principles to accelerate time to market while avoiding capital-intensive solutions. Using a pay-as-you-go strategy, invest as you scale out, rather than delivering a large investment up front.
<u>Operational excellence</u>	Automate the environment and operations to increase speed and reduce human error. Roll problem updates back or forward quickly. Implement monitoring and diagnostics from the start.
<u>Performance efficiency</u>	Efficiently meet demands placed on your workloads. Favor horizontal scaling (scaling out) and design it into your systems. Continually conduct performance and load testing to identify potential bottlenecks.
<u>Reliability</u>	Build workloads that are both resilient and available. Resiliency enables workloads to recover from failures and continue functioning. Availability ensures users access to your workload at all times. Design applications to expect failures and recover from them.
<u>Security</u>	Implement security across the entire lifecycle of an application, from design and implementation to deployment and operations. Pay close attention to identity management, infrastructure access, application security, and data sovereignty and encryption.

Облачните системи поддържат ориентирания към микроуслуги архитектурен стил за конструиране на системи. Това е подход за изграждане на сървърно приложение като набор от малки, но високо-качествени подуслуги. Съответно, клиентите, на сървърните услуги, могат да бъдат отделни приложения, които да се поддържат и управляват самостоятелно. Всяка услуга работи в собствен процес и комуникира с други процеси, използвайки различен тип и вид протоколи като: HTTP/HTTPS, WebSockets, AMQP и мн. други. Всеки микросървис притежава специфична бизнес способност. Предимства на това архитектурен стил са:

- Всяка микроуслуга може да бъде проектирана, разработена и внедрена независимо една от друга, което осигурява възможно за независима работа по отделни области на приложението;
- Работата може да бъде дистрибутирана между отделни екипи;
- Проблемите са по-изолирани;
- Позволява използването на различни технологии;

Микроуслугите насырчават фактор #6 от принципите на дванадесетфакторното приложение.

.....

Всяка микроуслуга притежава своя собствена логика и данни, в рамките на автономен жизнен цикъл. Концептуалните модели се различават между подсистемите или микроуслугите. Този принцип е заложен в дизайнът, управляван от домейн (DDD), където всяка услуга притежава свой модел на домейн (данни плюс логика и поведение).

Традиционният (монолитен) подход, използван в много приложения, притежава единична централизирана база данни (или няколко бази), която често е нормализирана SQL база, използвана за цялото приложение и всички негови вътрешни подсистеми. Този подходът изглежда по-прост, позволява повторно използване на обекти. В крайна сметка се стига до огромни таблици, които обслужват много различни подсистеми, включващи атрибути и колони, които в повечето случаи не са необходими.

Монолитните приложения: ACID транзакции и SQL език, като и двете работят във всички таблици и данни, свързани с приложението. Това допринася за сравнително лесно комбиниране данни от множество таблици.

Достъпът до данни, става много по-сложен в архитектурата на микроуслуги. Дори когато използвате ACID транзакции в рамките на микроуслуга или ограничен контекст, е изключително важно да се има предвид, че данните, притежавани от всяка микроуслуга, са частни за тази и трябва да бъдат достъпвани синхронно, чрез нейните API крайни точки (REST, gRPC, SOAP и т.н.) или асинхронно чрез съобщения (AMQP). Капсулирането на данните гарантира, че микроуслугите са слабо свързани и могат да се развиват независимо една от друга. Ако множество услуги имат достъп до едни и същи бази данни, актуализациите на схемата изискват координация. Това би нарушило автономността на жизнения цикъл. Когато един бизнес процес обхваща множество микроуслуги трябва се да използва т.н. „евентуална последователност“. Това е много по-трудно за изпълнение от обикновените SQL съединения. Различните микроуслуги често използват различни видове бази данни. Съвременните приложения съхраняват и обработват различни видове данни. За някои случаи на употреба NoSQL база данни като Azure CosmosDB или MongoDB може да има по-удобен модел, както и да предлага по-добра производителност и мащабируемост от SQL база данни. Микроуслугите често използват смесица от SQL и NoSQL бази данни, което се нарича подход на „полиглотна устойчивост“ (Polyglot Persistence).

Концепцията за микроуслуга произлиза от модела на ограничен контекст (Bounded Context) в управлявания от домейн дизайн (DDD). DDD се занимава с големи модели, като ги разделя на множество BC и определя техните граници. Всеки BC има собствен модел и база данни. По същия начин всяка микроуслуга притежава свързаните с нея данни. В допълнение, BC притежават така нареченият повсеместен език, който помага на комуникацията между разработчици на софтуер и експерти в бизнеса.

Ограничен контекст може да бъде отделна услуга или е просто логическа граница.

.....

Предизвикателства и решения за управление на разпределени данни
Определянето на границите на микросервиз основно предизвикателство, което съсредоточава върху логическите модели на приложението и свързаните с него данни. Всеки контекст може да има различни бизнес термини. Термините и обектите, може да звучат подобно, но понякога имат различни цели в различни контексти. Например, даден потребител може да бъде посочен като потребител на системата, като клиент в контекста на ERP и т.н. Начинът, по който се идентифицират границите между множество контексти на приложения с различен домейн, е сходен с начина за ограничаване границите за всяка бизнес микроуслуга. Зависимостите между микроуслугите трябва да е сведена до минимум.

*Таблица 1.5.
Принципи за проектиране на облачни системи
(адаптирано от автора по Croll, 2022)*

име на български	име на английски	Описание (тук има доста цитати)
Разделяне на грижите	Separation of Concerns	всеки обект и модул трябва да бъде в своя собствена грижа и контекст
Капсулиране	Encapsulation	
Инверсия на зависимостта	Dependency Inversion	
Изрични компоненти	Explicit Components	
Единична отговорност	Single Responsibility	
Не се повтаряйте	Don't Repeat Yourself	
Устойчивост и невежество относно инфраструктурата	Presentation Ignorance	
Ограничени контексти	Bounded Contexts	

В книгата Cloud Native Patterns, авторът Корнелия Дейвис (2019) отбелязва, че „Контейнерите са чудесен инструмент за облачния софтуер“. Фондацията Cloud Native Computing поставя контейнеризацията на микроуслуги като първа стълка в Cloud-Native Trail Map - насоки за предприятия, които започват поддръжка в облака.

За изграждане, доставка и изпълнение на системи, изградени както като монолитни приложения, така и като ориентирани към услуги, се препоръчва използването на контейнеризирани технологии. Контейнеризацията е подход, в сферата на разработката на софтуер, при който кодът на приложение, всички негови зависимости и конфигурации са пакетирани в двоичен файл, наречен изображение. Изображенията са „шаблони“ само за четене и се съхраняват в регистър, който работи като хранили или библиотека за изображения. Изображението се трансформира в работещ екземпляр на контейнер, който може да се стартира, спира, премества и изтрива. Създават се контейнери за различните части от приложението: уеб услуга, база данни, кеширане и др. Точно както транспортните контейнери позволяват транспортирането на стоки, независимо от товарите вътре, софтуерните контейнери се възприемат като стандартна единица за внедряване на софтуер, която може да съдържа различен код и зависимости. Контейнеризирането на софтуера дава възможност на разработчиците и ИТ специалистите автоматично да подновяват новите промени в различни среди. Контейнерите също така изолират приложениета едно от друго в споделена операционна система. Приложения се изпълняват върху хостът на контейнерите. От гледна точка на приложението, инстанцирането на изображение означава създаването на контейнер. Друго предимство на контейнеризацията е мащабируемостта. Разширяването става бързо: създават се нови контейнери за краткосрочни задачи. Контейнерите предлагат предимствата на изолация, преносимост, гъвкавост и контрол в целия жизнения цикъл на приложението.

Управлението на контейнери се извършва със специална софтуерна програма, наречена контейнер оркестратор. Когато работите в мащаб с много независими работещи контейнери, оркестрацията е от същевно значение. Следващата таблица описва общи задачи за оркестрация.

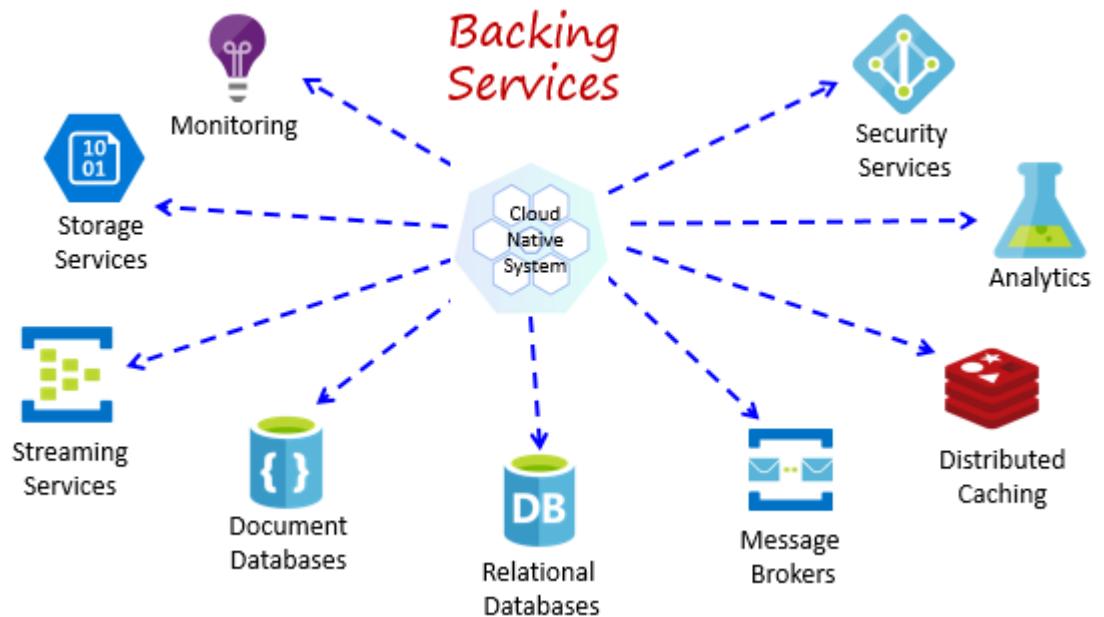
Таблица 1.5.
modo
(modo)

Tasks	Explanation
Scheduling	Automatically provision container instances.
Affinity/anti-affinity	Provision containers nearby or far apart from each other, helping availability and performance.
Health monitoring	Automatically detect and correct failures.
Failover	Automatically reprovision a failed instance to a healthy machine.
Scaling	Automatically add or remove a container instance to meet demand.
Networking	Manage a networking overlay for container communication.
Service Discovery	Enable containers to locate each other.
Rolling Upgrades	Coordinate incremental upgrades with zero downtime deployment. Automatically roll back problematic changes.

Оркестраторите на контейнери възприемат принципите за единократност (*Factor #9*) и едновременност (*Factor #8*) от приложението с дванадесет фактора.

Въпреки че съществуват няколко контейнерни оркестратора, Kubernetes се превърна в де факто стандарт. Това е преносима, разширяема платформа с отворен код за управление на работни натоварвания в контейнери.

Облачните системи зависят от много различни спомагателни ресурси, като хранилища за данни, брокери на съобщения, мониторинг и услуги за идентичност. Тези услуги са известни като поддържащи услуги. Фигура 1.14 показва много общи услуги за поддръжка, които използват облачните системи.



Фиг. 1.11. Спомагателни услуги използвани от облачните системи (Smith, 2022).

Облачните доставчици предлагат богат асортимент от спомагателни услуги. Облачния доставчикът управлява ресурса и носи отговорност за производителността, сигурността и поддръжката. Мониторингът, резервирането и наличността са вградени. Добрата практика за тези услуги е да се третират като прикачен ресурс, динамично свързан с главна микроуслуга чрез конфигурация (URL и идентификационни данни). Това ръководство е изложено в Приложението на дванадесетте фактора, обсъдено по-рано в главата. С този модел поддържащите услуги може да бъдат контролирани без промени в кода. Облачните доставчици предоставят API, за комуникация, чрез патентовани библиотеки, капсулиращи сложността. Добра практика е да се въведе междинен слой или API, който да предлага общи операции, „обвивайки“ кода на доставчика вътре в него.

Това „слабо“ свързване позволява обновявания, без да се налага да се правят промени в кода на основната услуга.

TODO:

[Cloud Computing Essentials \(udemy.com\)](#)

Essentials

[Software Architecture & Design of Modern Large Scale Systems \(udemy.com\)](#)

software-architecture-design-of-modern-large-scale-systems.pdf

[The Complete Guide to Becoming a Software Architect \(udemy.com\)](#)

Kenov like patterns principles (1 glava + 2ra)

[SOLID Principles: Introducing Software Architecture & Design \(udemy.com\)](#)

Solid

1.2.2. Управление на бизнес процесите чрез ориентиран към домейн дизайн

DDD, CQRS, ES, TDD

1.3. Специфики при управление на поръчките от клиенти в производствено предприятие

Система за управление на поръчки е цифров начин за управление на жизнения цикъл на поръчка (George Kokoris, 2018). Тя проследява цялата информация и процеси, включително въвеждане на поръчки, изпълнение и следпродажбено обслужване. Системата предлага видимост както за бизнеса, така и за купувача. Организациите могат да имат поглед върху заявките и доставките в реално време, а клиентите могат да проверяват кога ще пристигне поръчка. Управлението на поръчките започва, когато клиент направи поръчка, и завършва, след като получи своя продукт или услуга. Софтуерната система позволява на бизнеса да координира процеса на изпълнение. Включният работен процес може да се различава в зависимост от нуждите на компанията, но типично включва стъпки като:

- Заявление: Клиентът прави поръчката чрез автоматизирана форма. Член на търговския екип проверява детайлите и потвърждава поръчката;
- Изпълнение: Служител в склада потвърждава подробните за доставката и изпълнява поръчката

1.3.1. Киберсигурност и защита на данните предоставени публично на клиентите

TODO:

[Azure Security Best Practices \(udemy.com\)](#)

Облачните системи осигуряват персонализиран достъп на потребителите. Най-често се формират групи потребители с точно определени права за достъп (например счетоводители, касиери, стоковеди, мениджъри производство, технолози, логистични мениджъри). Всеки нов потребител се причислява към определена група.

Софтуерните приложения трябва да имат познание за потребителя

или процеса, който ги извиква. Потребителят или процесът, взаимодействащ с приложение, е известен като принципал за сигурност, а процесът на удостоверяване и упълномощаване на тези принципали е известен като управление на самоличността (Vettor, 2022). Простите приложения могат да включват цялото им управление на самоличността в приложението, но този подход не се мащабира добре с много приложения и много видове принципали за сигурност. Windows, например, поддържа използването на Active Directory за предоставяне на централизирано удостоверяване и оторизация. Въпреки че това решение е ефективно в рамките на корпоративни мрежи, то не е предназначено за използване от потребители или приложения, които са извън домейна.

Удостоверяването е процес на определяне на самоличността на потребител. Упълномощаването е актът на предоставяне на удостоверено разрешение за извършване на действие или достъп до ресурс (Wike R, 2022).

Мобилни и базирани на JavaScript приложения се изпълняват на ниво клиент. Те трябва да комуникират с публичен API, което означава че не може да се използва защита като VPN. Изпращането на потребителско име и парола (Basic Authentication) при всяка заявка от клиент към API е лоша идея, защото например с анализатор на мрежови протоколи, има вероятност някои да проследи паролата и така да получи достъп до много информация.

Вместо да се изпраща комбинацията от потребителско име и парола при всяко повикване към API, се използват токени. Те представляват съгласие, например съгласие, дадено от потребителя за достъп. Токените се издават от централен компонент за решаване на общи задачи, свързани с идентичността. Отговорност на доставчика на самоличност е да удостовери потребител и безопасно да предостави доказателство на приложение. Често, доставчика на самоличност е и портал към задачи, свързани с потребителя и управлението на акаунта.

Регистрация на потребители, управление на акаунти, прилагане на политики за пароли с правила за силата, блокиране на потребители и т.н. са

задачи към подсистемата за управление на идентичност и достъп. Тя позволява съвместимо използване в различни приложения, както и възможност цялата система да бъде тествана за пробив (penetration test). Важна функционалност е многофакторното удостоверяване, при което потребителят трябва да предостави допълнителна информация (освен име и парола), за да докаже своята самоличност, като например цифров пропуск, издаден от смартфон, който има приложение за удостоверяване.

OAuth2 е отворен протокол, който позволява сигурно оторизиране по стандартен метод от уеб, мобилни и настолни приложения. OAuth2 дефинира токен, който може да бъде поискан от клиентско приложение за да получи достъп до API, като го предаде по HTTP. OAuth определя крайни точки за оторизация, дефинирани по стандарт, като също така посочва как да бъдат използвани от различни типове приложения.

Следната таблица представя различни доставчици на самоличност които прилагат OAuth2.

Таблица 1.6.
modo
(modo)

Okta	
Trust Builder	
Identity Server	

OpenID Connect е слой за идентичност, разширяващ, прилагащ принципите на OAuth2, който помага на приложенията да получат токен за самоличност/достъп. Той се използва за влизане в клиентско приложение, докато същото приложение използва токена за достъп до API.

Фигура 1.12. илюстрира комуникацията за получаване на токен.



Фиг. 1.12. Схема на OpenID Connect (Smith, 2022).

Създава се заявка от клиентското приложение, което пренасочва потребителя към доставчика на самоличност, където потребителят доказва кой е, чрез предоставяне на потребителско име (или например имейл, телефонен номер) и парола. Това действие може да е познато от влизане в сайт чрез Google или Microsoft. Това е началото на OpenID Connect потока. В крайна сметка доставчикът на самоличност създава токен, подписва го и го връща на клиентското приложение. Така то има доказателство за самоличност. OpenID Connect е проектиран да работи за различни видове приложения (уеб приложения от страна на сървъра или от страна на клиента, мобилни приложения, уеб услуги и др.), като дефинира типове клиенти и потоци.

Има два дефинирани типа клиенти: поверителни и публични клиенти. Поверителният клиент в състояние да запази поверителност на идентификационните си данни, които са клиентски идентификатор и тайна. Това не са идентификационните данни на потребителя. В OpenID Connect може да се удостовери както потребител, така и самото клиентско приложение. Идентификационните данни се съхраняват на ниво приложение, като например ASP.NET Core MVC уеб приложение, което работи на уеб сървър. Тъй като това приложение работи на сървъра, то може безопасно да съхранява идентификационните си данни, тъй като не са достъпни от потребителя. Публичните клиенти не са в състояние да запазят

поверителност, тъй като изпълнението на програмата е в браузъра или на мобилно устройство, на ниво потребител. „Тайна“, съхранена в JavaScript или на мобилно устройство, не е тайна. Винаги е достъпна. В частност мобилните операционни системи позволяват използването на хардуерни API за безопасно съхраняване на тайни, но все о се считат за публични клиенти, поради факта, че тайната се съхранява на устройството.

Потокът в OpenID определя как кодът и/или токените се връщат на клиента. В случая разглеждате поток като редица заявки и отговори, често между клиентските приложения и доставчика на идентичност. Пример за такъв поток е сайт, който предлага опция за влизане с акаунт от Microsoft или Google. Браузър пренасочва към сайт, където се предоставя потребителско име и парола. След това се пренасочва обратно към основния сайт. В повечето от тези потоци има набор от заявки и отговори, които се случват, без да бъдат визуализирани от браузъра. OpenID потока се определя в зависимост от изискванията на клиента.

Заявките за оторизация се изпращат до крайна точка на ниво доставчик на идентичност. Тази крайна точка се използва от клиентското приложение за получаване на удостоверяване за токени за самоличност и/или оторизация за токени за достъп от потребителя. Това става чрез пренасочване на клиентското приложение към доставчика на идентичност. Код за оторизация или токен могат да бъдат върнати към клиентското приложение. TLS (SSL) е изискване за OpenID Connect, трафикът трябва да винаги е криптиран. Липсата на криптиран трафик прави потока уязвим към атаки от тип „човек по средата“.

Крайна точка за пренасочване е URI адресът, към който IDP пренасочва обратно клиента. Той е на ниво клиентско приложение и се използва от доставчика на идентичност, за да достави кода за оторизация или токен на клиентското приложение.

Крайна точка за токен е на ниво доставчик на идентичност. Клиентските приложения могат програмно да изискват токени. Обикновено

става без пренасочване чрез методът HTTP POST. Повикванията могат да бъдат удостоверени с идентификатор и тайна за поверителните клиенти.

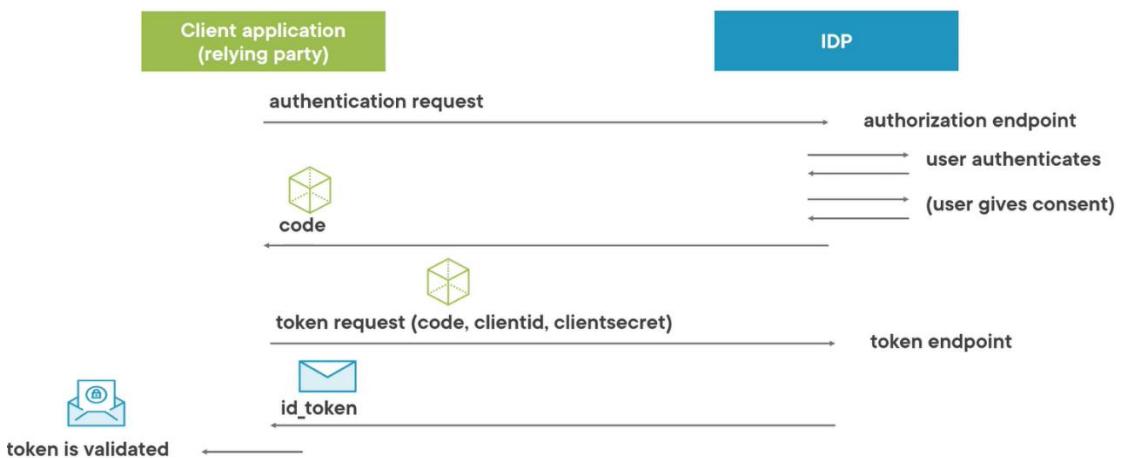
Както за поверителни, така и за публични клиенти се препоръчва използването на вариации на потока „Кода за оторизация“ (Authorization Code Flow) подплатен с PKCE защита (OAuth 2.0 Security Best Current Practice RFC / <https://datatracker.ietf.org/doc/html/draft-ietf-oauth-security-topics-19>). Потокът „код за оторизация“ получава името си от факта, че започва с код за оторизация. Този код за оторизация може да се разгледа като краткотраен идентификационен номер за еднократна употреба, който се използва за проверка дали влезлият на ниво доставчик на идентичност е същият, който е започнал на ниво клиентско приложение. Този поток позволява дълготраен достъп за поверителни клиенти, но за публичните е ограничен. Разрешена е вариация на токен за опресняване. Изборът на правилен вариант на поток е важен, защото може да доведе до уязвимости в сигурността. Технически, много от подходите работят, но повечето доведат до сериозен рисък.

Всеки OpenID Connect поток започва със заявка към крайната точка за оторизация. Пример за URI е представен на следната фигура.

```
https://idphostaddress/connect/authorize?  
client_id=imagegalleryclient  
&redirect_uri=https://clientapphostaddress/signin-oidc  
&scope=openid profile  
&response_type=code  
&response_mode=form_post  
&nonce=63626...n2eNMxA0
```

Фигура 1.12: *URI for the authentication code flow endpoint.*

На първо място е самата крайна точка на ниво IDP. След това са идентификаторът на клиентското приложение (client_id), URI за пренасочване. Това е URI на ниво клиентско приложение, до което бъде доставен отговорът от заявката за оторизация. В случая, зададения обхват показва, че приложението изисква достъп до и до идентификатора на потребителя (openid) и твърденията, свързани с профила, като собствено име и фамилия. След това е стойността на типа отговор (response_type), която определя потока, който се използва. Стойностите, зададени като response_type, са нещата или токените, върнати от доставчика на идентичност към клиента, чрез пренасочване от браузъра. Фигура 1.13 представа модела на този поток.



Фигура 1.12: *authentication code flow model.*

Уеб приложение създава и изпраща заявка за удостоверяване с код. На ниво IDP потребителят предоставя комбинация от потребителско име и парола. В този момент доставчикът на самоличност знае кой е потребителят.

Клиентското приложение все оне. Доставчикът на самоличност по избор иска съгласие от потребителя, като например да изиска разрешение за получаване на достъп до информацията от профила. След това доставчикът на самоличност изпраща обратно към уеб приложението чрез URI пренасочване или POST формуляр. Това е комуникация, която е видима за браузъра. Кода за оторизация може да се разглежда като много краткотраен токен, който е доказателство за удостоверяване. Той е свързан с потребителя, който току-що е влязъл в доставчика на самоличност. В следващата стъпка уеб клиентът извиква крайната точка за токен. Тази заявка не използва пренасочване и следователно не е видима за браузъра. Това е HTTP заявка от сървър към сървър, към която се предава кода за оторизация и комбинация от clientid и clientsecret. Доставчикът на самоличност генерира токен и го връща на клиента в HTTP отговора. На ниво клиент токенът се валидира и извлича идентификатора на потребителя.

Съвременните решения за самоличност обикновено използват токени за достъп, които се издават от защитена услуга/сървър (STS) на принципал за сигурност, след като тяхната самоличност бъде определена (Anil N, 2022).

Удостоверяването чрез токени е механизъм без състояние, тий като никаква информация за потребителя не се съхранява в паметта на сървъра или базата от данни, за разлика от бисквитките (Gichuhi , 2021).

Стандарт (RFC 7519) за уеб приложениета е JSON уеб токен (JWT). Той се състои от три части:

- Заглавна - JSON обект, кодиран във формат base64. Съдържа информация за типа на токена и алгоритъма за криптиране;
- Полезен товар - съдържа информация за текущия потребител (потребителско име, роля и др). Тук не трябва да се включват чувствителни данни, защото лесно се могат да бъдат декодирани със публични сайтове като jwt.io;
- Подпис – Използва се от сървъра, за да провери дали токенът е

валиден. Той се генерира чрез комбиниране на двете части (заглавна и полезен товар) заедно. Базира се на таен ключ, който само сървърът за удостоверяване знае. По този начин злонамерен потребител не може да фалшифицира валиден токен;

1.3.2. Предизвикателства при управлението на клиентските поръчки в производствените организации

Обслужването на клиенти е сложен процес, който включва различни променливи и компоненти. Разбирането на очакванията на клиентите е от решаващо значение за изграждането на ефективна стратегия за обслужване. Международният институт за управление на клиенти (ICMI) идентифицира важни фактори, включително достъпност на услугата, отзивчиви и добре обучени служители, адекватни доставки и други. Тези очаквания непрекъснато се развиват и разбирането им помага на организациите да изпълнят или надхвърлят тези очаквания.

customer service strategy

Ефективната стратегия за обслужване на клиенти е от решаващо значение поради бързо развиващите се очаквания на клиентите. Стратегията се свързва с план за действие, предназначен за постигане на дългосрочна или обща цел, различна от визията. От съществено значение е организациите да имат ясна визия и стратегия, които да ръководят решенията и действията. В много организации има цялостна корпоративна стратегия, стратегии в рамките на конкретни подразделения или бизнес единици. Стратегиите за обслужване на клиенти трябва да отразяват и подкрепят цялостната визия на организацията, но да бъдат достатъчно конкретни, за да ръководят развитието. Например, визията на Уолт Дисни за услуги през 50-те години на миналия век е да създава щастие чрез безопасност, утивост, шоу и ефективност. Стратегията на Disney се фокусира върху ключови действия и поведения, които подкрепят всяко действие, вдъхвайки живот на визията и стратегията чрез действията на всеки служител. Без ясна и ефективна стратегия клиентите и техните проблеми може да не получат вниманието, което заслужават.

Ефективната стратегия за обслужване на клиенти е ориентирана към действия, помагаючи за информирани решения и операции. Тя трябва да поддържа от висшия мениджмънт и да насърчава и приоритизира подобренията на услугите, продуктите и процесите. Успешната стратегия води до увеличаване на приходи, пазарен дял и ангажираност на клиенти и служители. Ентузиазът на клиентите относно организацията е крайната мярка за ефективността на подхода.

Технологиите са от важно значение при разработването на стратегия за обслужване на клиенти, тъй като позволяват на организацията да подкрепят своята визия и да подобрят своите услуги. Основните тенденции в технологиите за обслужване на клиенти включват: администриране на поръчки, проследяване на доставки в реално време.

За да се гарантира, че финансирането подкрепя стратегията, от съществено значение е да се вземат всички решения за финансиране в контекста и да се максимизират междуфункционалните ресурси. SWOT анализът може също така да помогне на организацията да идентифицира силните, слабите страни, възможностите и заплахите, позволявайки инициативи, които използват силните страни и смекчават слабите страни. Като се фокусират върху иновациите, организацията може да създаде по-ефективна и ефикасна стратегия за обслужване на клиенти.

Customer satisfaction

Качеството на услугите е от решаващо значение за устойчивия растеж на една организация. Изграждането на добър екип включва внимателен подбор, съгласуване на организационните ценности и други. Удовлетворението на служителите и клиенти са взаимосвързани и отговорността на всеки е да оправдае и дори надмине очакванията.

[10 Things Every Customer Wants | Inc.com](#)

Изводи и обобщения към първа глава

Дигитализацията е един от стълбовете на трансформацията, преминаващ през всички бизнес операции. Дигитални продукти, насочени към клиентите, целят да помогнат за успех в основния бизнес. Те са един от главните източници за повишаване производителността и конкурентоспособността. Облачните услуги са ключов инструмент на тази трансформация. Те осигуряват възможност за електронен обмен на данни между участниците в логистичния процес чрез съвременни изчислителни ресурси и високоскоростни канали за пренос на данни.

организация на бизнес процесите съобразно възможностите на софтуерния продукт; □ В редица предприятия мениджърите внедряват ERP система, за да нагодят бизнес процесите си спрямо особеностите на ERP системата. Въпреки, че подобен подход изглежда алогичен той се среща в практиката.

В повечето случаи логистичните мениджъри се нуждаят от една „малка“ справка, за да вземат решение. Така например справка за движението на сировини и материали може веднага да покаже онези сировини и материали, които са закупени, но не са вложени в производство (залежават в склада).

Логистичен мениджър може да изведе справка „Продажби по дати“, за да направи тенденция на продажбите (в количествено изражение).

....

В логистичната система възниква информация на различни места. Тази информация следва да се набира, структурира и съхранява в бази от данни. Най-често информацията се въвежда ръчно в средата на ERP (Enterprise Resource Planning) система (наричана на български език автоматизирано или автоматично. Освен това, набраната информация се използва от логистичните мениджъри за вземане на решения.

Някои производствени предприятия получават ежедневно поръчки

от своите клиенти по телефон. Поръчките се въвеждат от служител на производственото предприятие в корпоративна ERP система. В края на деня се извежда справка, съдържаща обобщена информация.

Набирането на информация за поръчки от клиенти от дистрибуторите в определени случаи се извършва чрез мобилни терминали. Тези терминали могат да предават данни към централизирана ERP система онлайн (в реално време) или офлайн. В първия случай логистичният мениджър има по всяко време актуална информация за получените поръчки.

Получаването на заявка от клиент в онлайн система е обичайна практика. Най-често онлайн системата не е директно свързана с ERP системата на търговеца. Обикновено получените заявки се одобряват, преди да се прехвърлят в ERP системата.

Трето, усъвършенстване на мобилните връзки; □ Стремежът на редица предприятия е да накарат своите клиенти да въвеждат поръчки в онлайн система, вместо да звънят по телефона. В този случай се използва „свободното“ време на клиента за въвеждане на поръчка. Редица пицарии предлагат отстъпка при поръчване онлайн.

Така на пример куриерските компании интегрират своите корпоративни системи с уеб сайт, където клиентите могат сами както да създават товарителници, така и да проверят придвижването на своята пратка по номер на товарителница.

.....

Перспективите в развитието им в рамките на логистичната система могат да се обобщят до:

1. Информационна интеграция на основата на Интернет и глобален мониторинг на материалните потоци;

□ Повечето куриерски компании позволяват проследяване на пратка по номер на товарителница. Повечето превозвачи на контейнери (като например Maersk) позволяват онлайн проследяване на контейнер по номер на контейнер.

оптимална маршрутизация; Използването на устройства за навигация в автомобили е обичайна практика. В този случай устройството има вградена GPS антена. Американското правителство плаща 400 млн. долара годишно за поддържане на спътниците, за да може всички потребители да ползват устройствата си с GPS антена бесплатно. Повечето смарт телефони имат вградена GPS антена. Използването на устройства за навигация в автомобили е обичайна практика. В този случай устройството има вградена GPS антена. Американското правителство плаща 400 млн. долара годишно за поддържане на спътниците, за да може всички потребители да ползват устройствата си с GPS антена.(6) определяне на оптималната продължителност на логистичния цикъл;(7) оптимизация на процедурите по набиране, обработка и изпълнение на заявки;(8) оптимизация на параметрите на системите за управление на запаси;(9) оптимален избор на превозвач, експедитор или доставчик.

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.

TODO:

<https://heidelbergmaterials.udemy.com/course/how-to-become-an-outstanding-solution-architect>

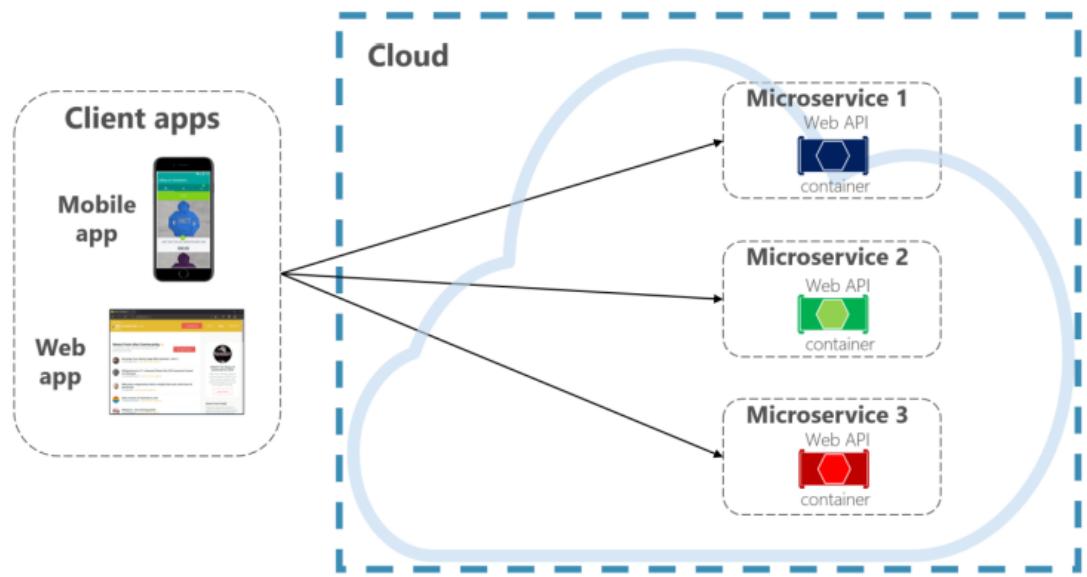
The Process For Designing Architectures 89

2.1. Ключови бизнес процеси и дейности свързани със системата за управление на поръчките

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, базирана на микроуслуги, работеща върху множество процеси и сървъри (хостове). Всяка услуга се изпълнява в отделен процес като контейнер, разположен в кълстер 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.

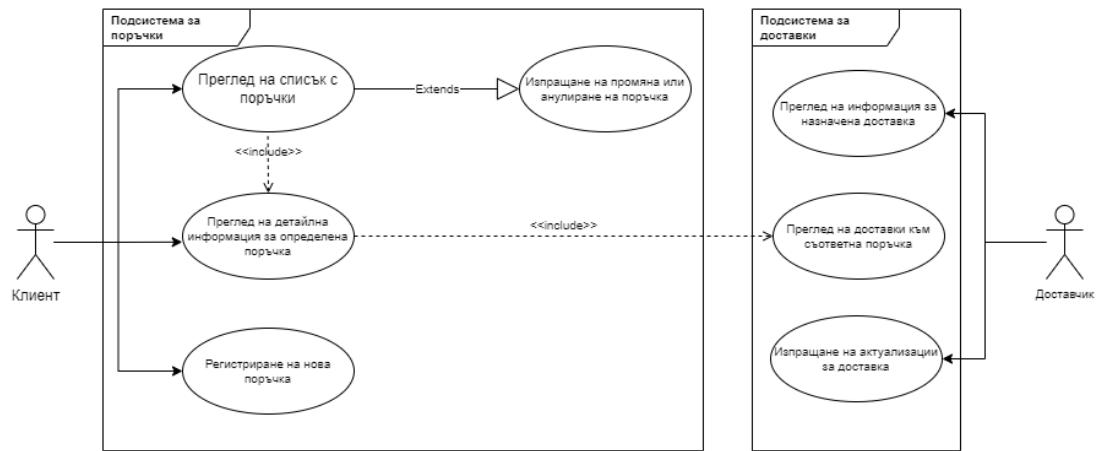


*Фиг. 2.1. Диаграма от високо ниво на главните приложения.
(разработка на автора)*

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.

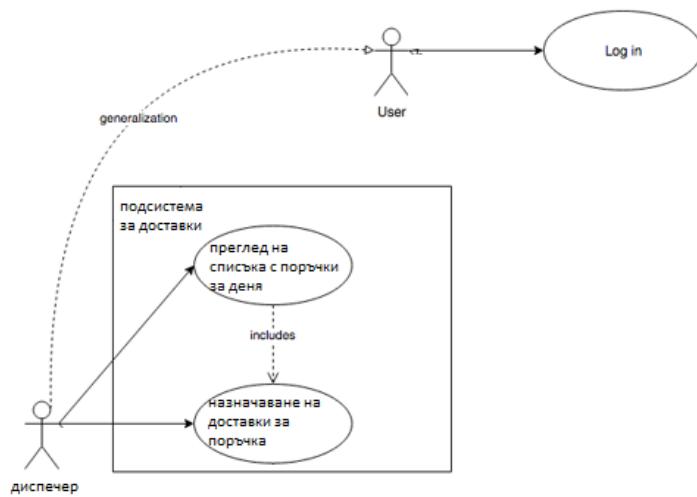
Обхватът на мобилното приложение, насочено към крайните клиенти, включва еcran за вход, интерфейс за текущите поръчки и доставки към тях. Също така панел за създаване или промяна на поръчка. Фигура 2.2 представя процесите под формата на диаграма.



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

Уеб порталът е софтуер, насочен към диспечерите, 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.



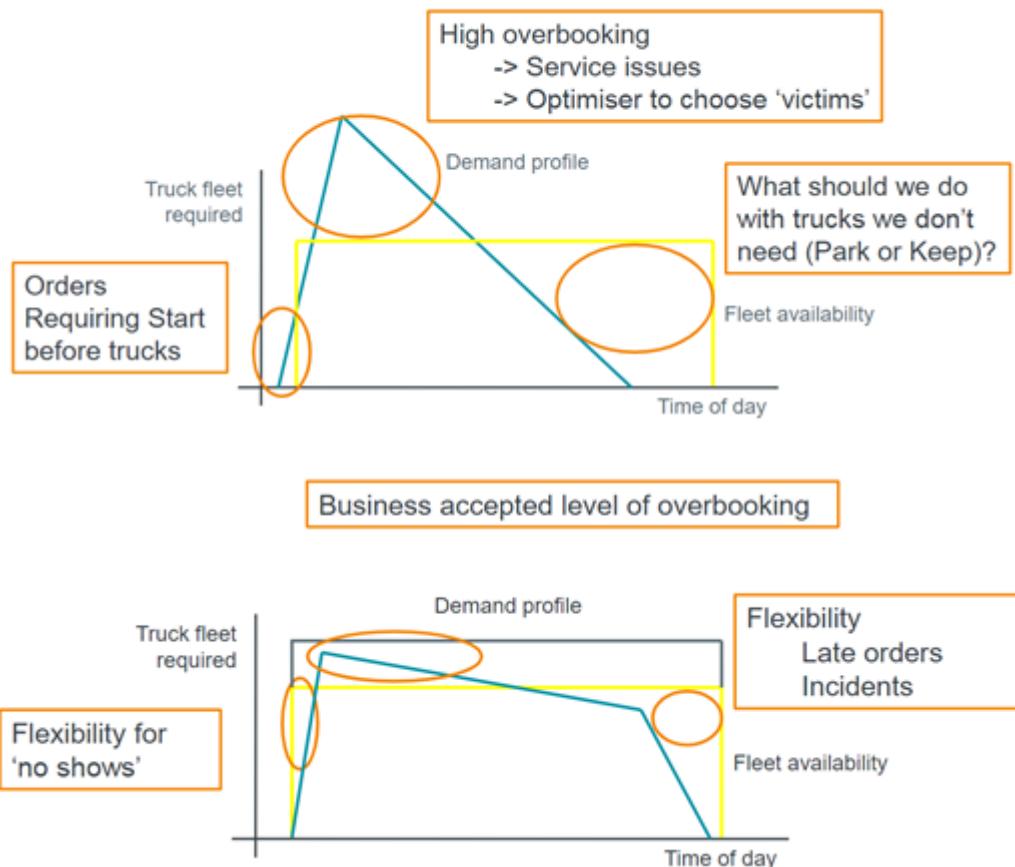
Фиг. 2.2. Диаграма на главен бизнес сценарий (разработка на автора)

Уеб портала прилага усъвършенствани техники като: оптимизация в реално време, оптимизация за оценка на поръчки, оптимизация на предварителното планиране, които се извършват постоянно във фонов режим. Входните данни, идващи от ЕРП, са записите за завод, клиентски местоположения, превозни средства, техните свойства и статуси, поръчки, параметри на оптимизатора и други.

....

Следна фигура представя правила за приемане на поръчки:

What are the rules for taking orders?



Фиг. 2.3. Правила за приемане на поръчки (разработка на автора)

Уеб портала поддържа доклад за **късно зареждане** е Той служи като обратна връзка към диспетерите. В случай, че клиентските поръчки закъсняят, клиентите могат да бъдат извикани проактивно. Целта е ако все пак има закъснение, по-добре е да клиентите да бъдат уведомени предварително.

Уеб портала предоставя ежедневно експортиране на пробега на всеки камион, въз основа на отчет от базата данни. Автоматизиран интерфейс е проектиран и внедрен, за да замени ръчното извлечане на данни. ЕРП предоставя нов функционален модул за получаване на данни за пробега чрез RFC. Веднага след като камион приключи смяната си, всички изминати натоварени и разтоварени маршрути, записани в база данни се експортира със специален идентификатор към ЕРП. Ако това експортиране бива неуспешно, съобщение, съдържащо идентификатора, се появява в полето за съобщения.

2.2. Концептуален модел на системата

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

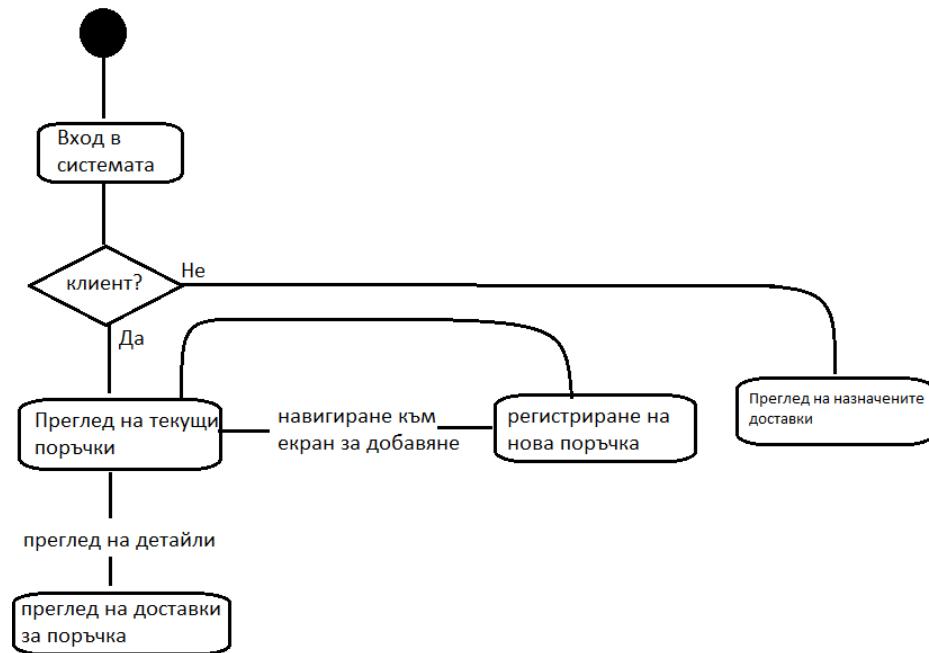
2.2.1. Поведенчески диаграми

Поведенческите диаграми идентифицират как различните елементи взаимодействат помежду си.

2.2.1.1 Диаграми за активност UML

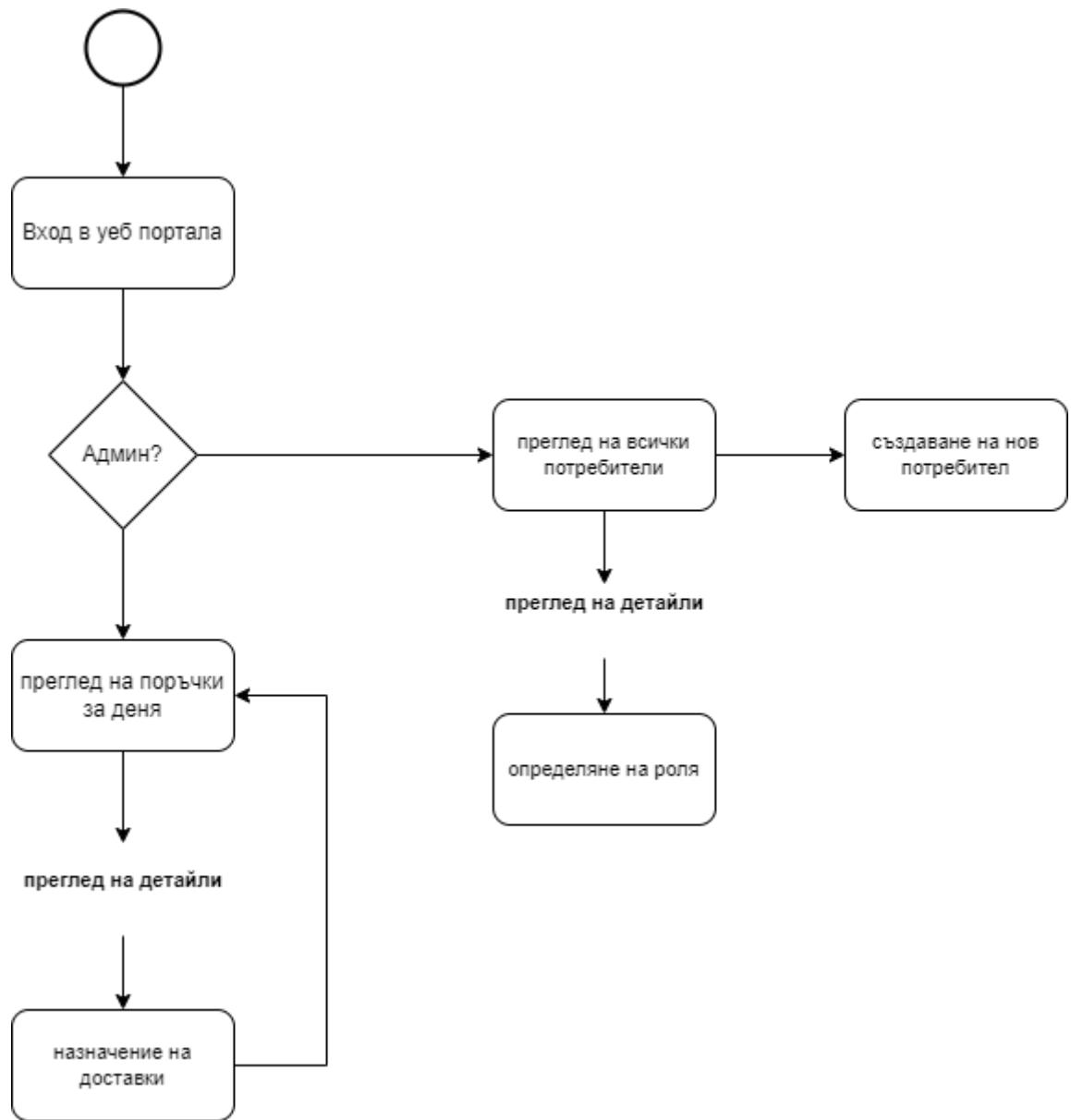
Диаграмите за активност изглеждат много подобни на блок-схемите. Наличието на тези прилики улеснява комуникацията между технически и не-технически лица (stakeholders).

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

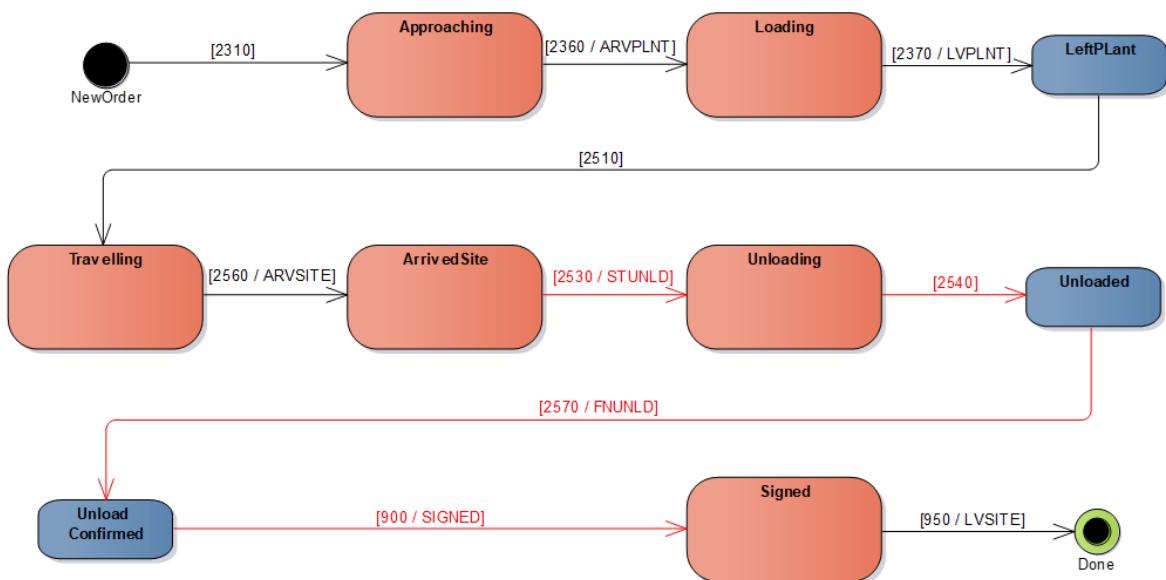


*Фиг. 2.3. Диаграма на активноста за мобилно приложение.
(разработка на автора)*

Следната диаграма изобразява потока от операции в уеб портала:



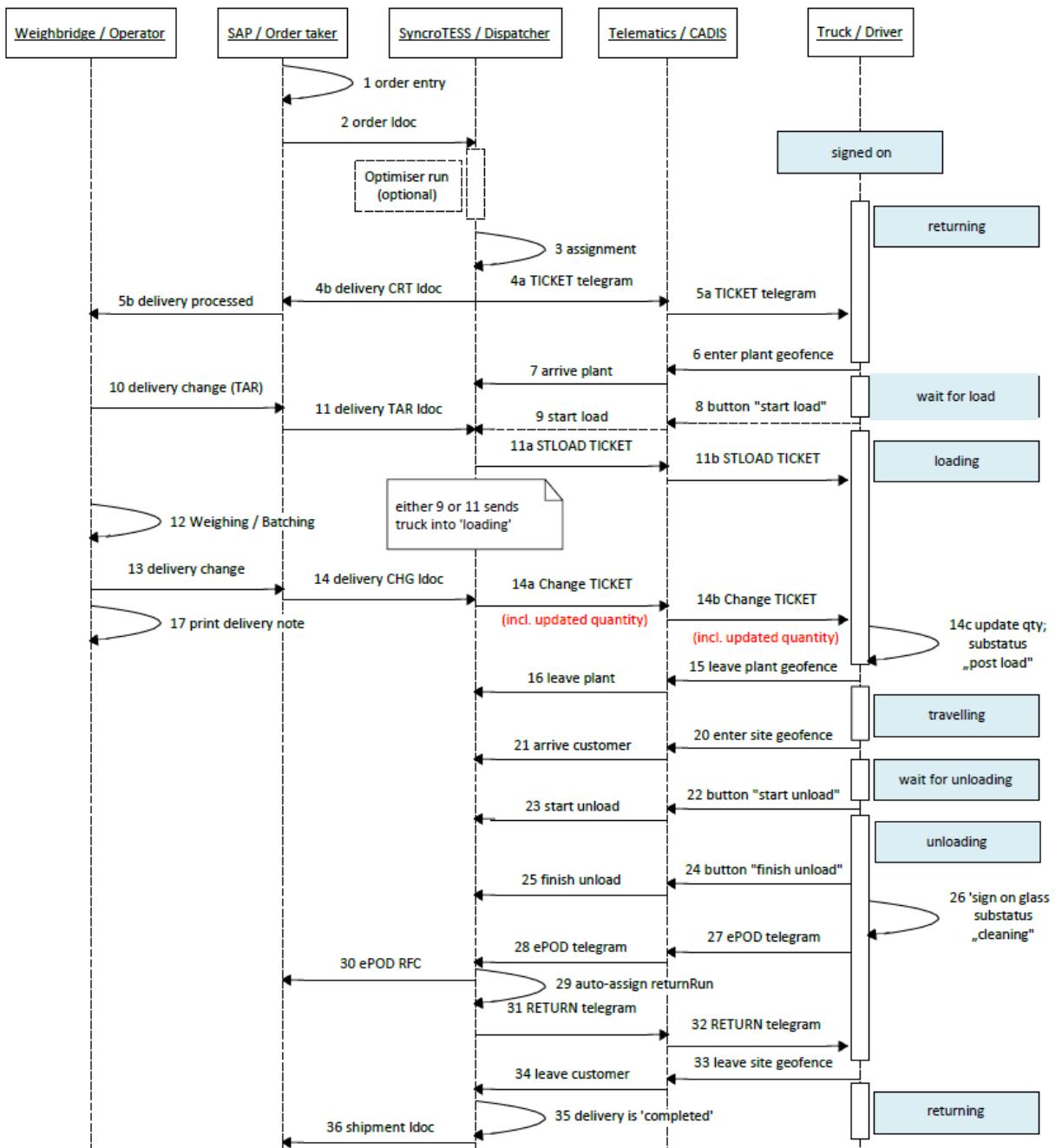
Фиг. 2.4. Диаграма на активност за уеб портал. (разработка на автора)



Фиг. 2.4. Диаграма на активност на поръчка. (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.



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

2.2.2. Структурни диаграми

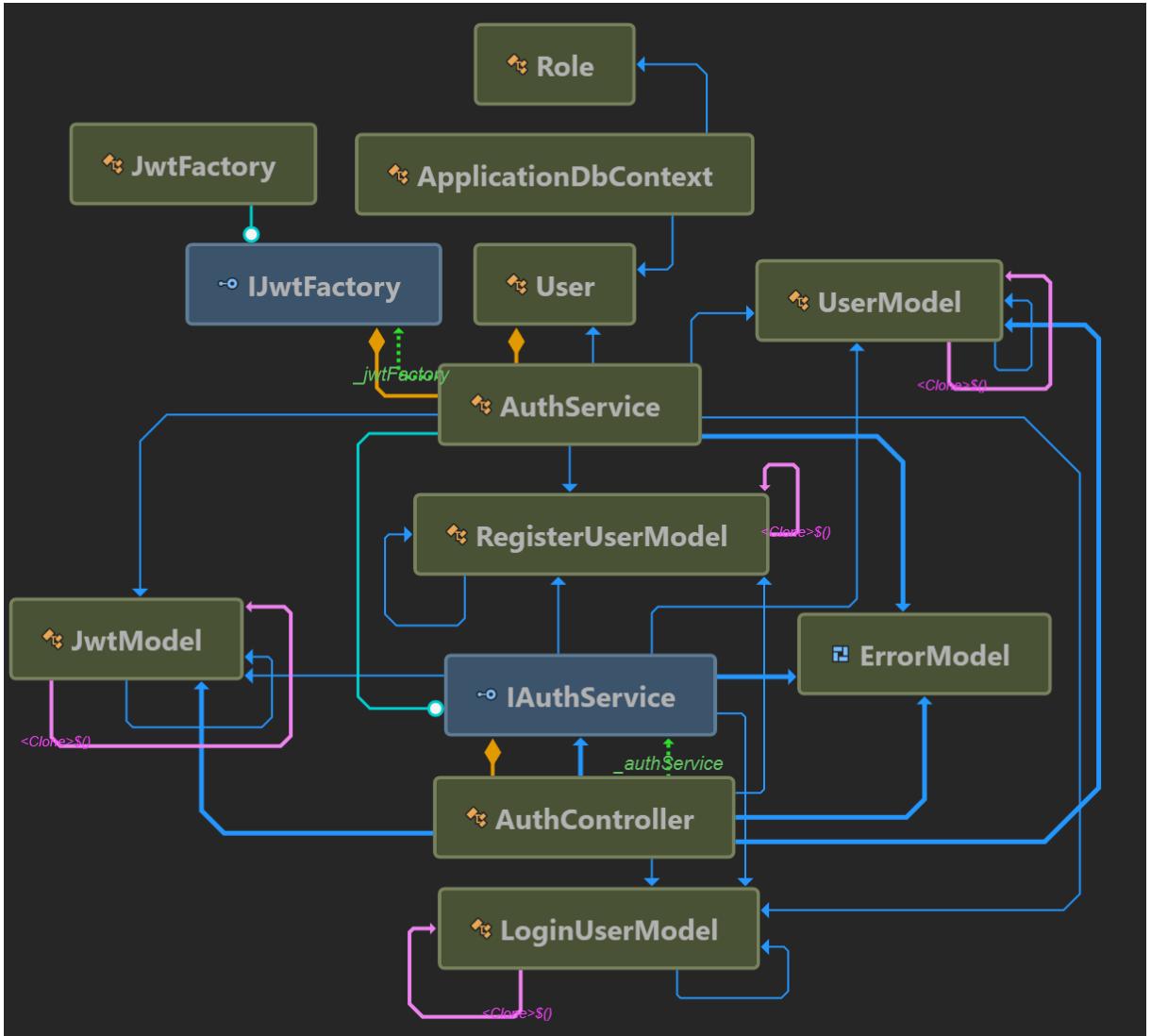
Структурните диаграми помагат за дефиниране цялостната структура на системата, подобно на плана, който определя как изглежда една къща. Структурните диаграми моделират как изглежда системата в архитектурно отношение. Те ни помагат да дефинираме „речника“ на системата, гарантират съгласуваност от заинтересовани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

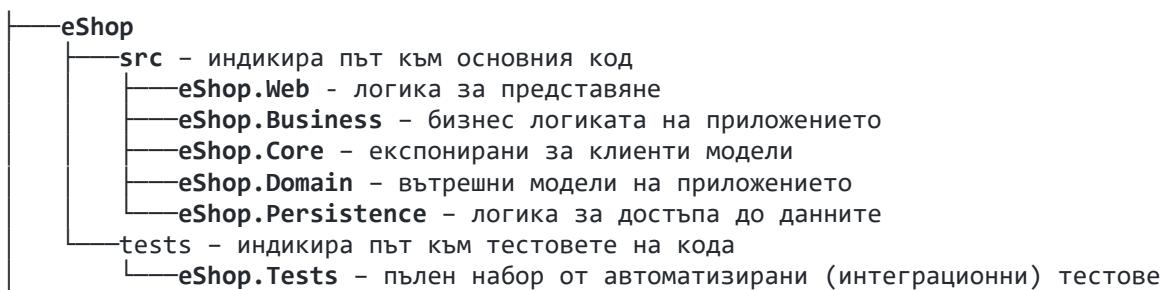
Диаграмите на класове са едни от най-често срещаните, когато става на въпрос за разработката на софтуер. Едно от основните неща, които тези диаграми правят е да идентифицира речника на системата. Например, те определят връзките между обектите, които съответстват на основните съществителни.

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 свойства чрез **UserService**, който капсулира логиката по безопасен за използване начин.

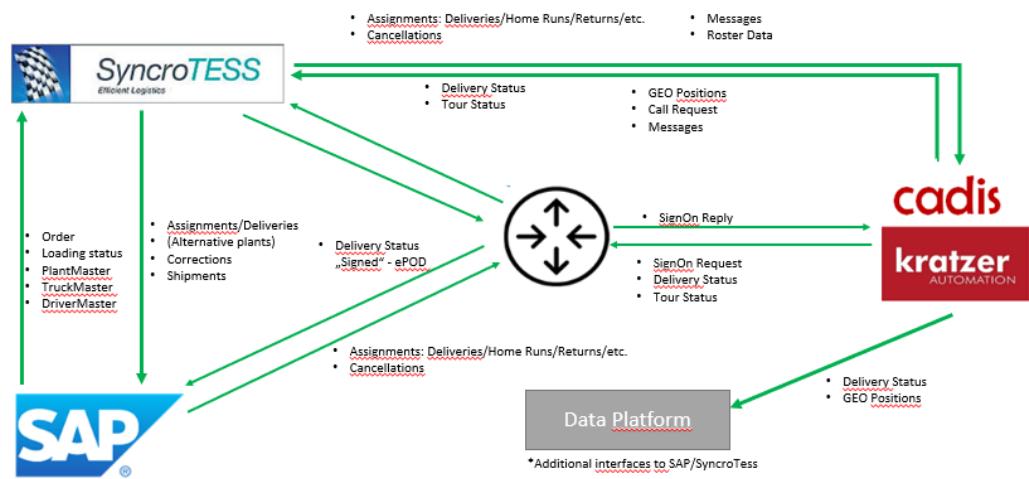


Фиг. 2.8. Интерактивна диаграма на класовете. (разработка на автора)

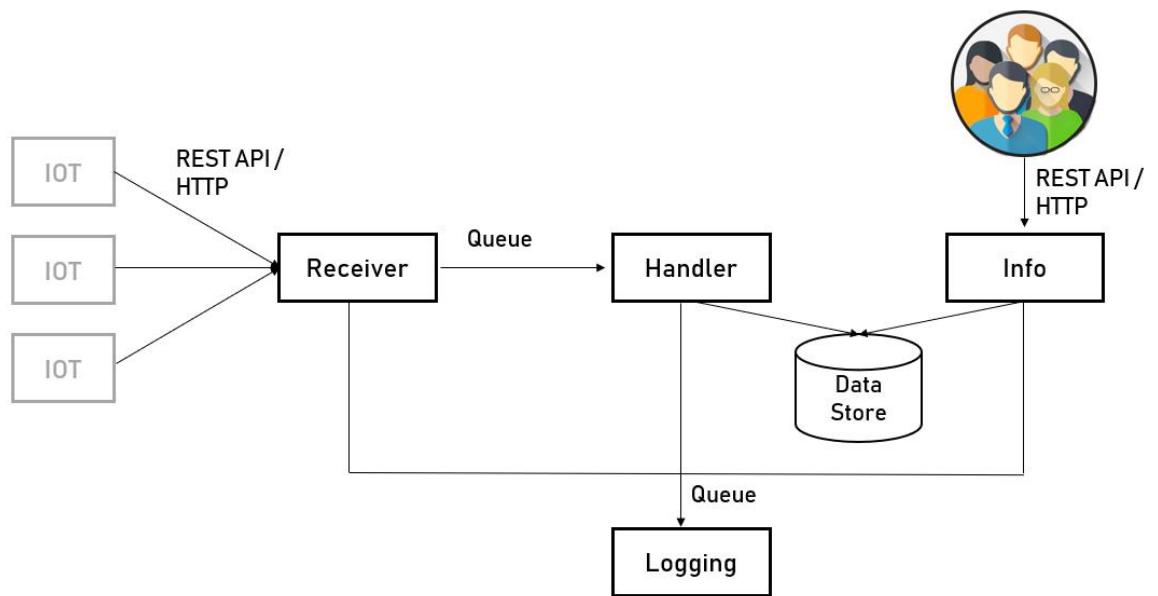
Структурата на папките на приложението е добре оформена, по следния функционален, управляван от домейн дизайн:



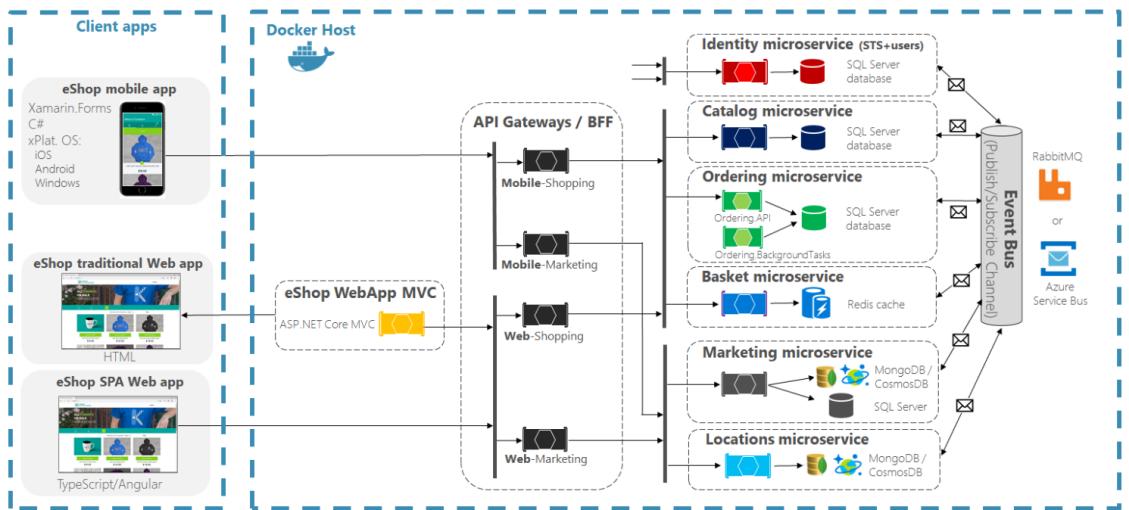
2.2.3. Архитектурни диаграми



Фиг. 2.9. Процесен модел на веригата за доставки. (разработка на автора)



Фиг. 2.10. архитектурна диаграмма на подсистемата за доставки. (разработка на автора)

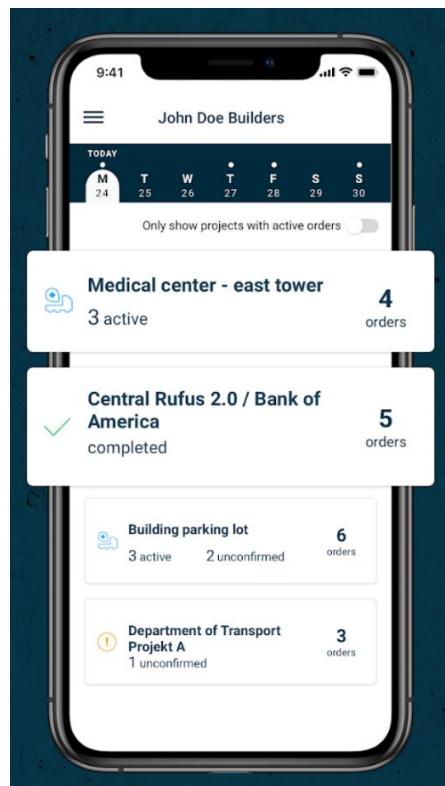


Фиг. 2.11. Микросървисната архитектурна диаграма.
(разработка на автора)

2.3. Функционалност и потребителски интерфейс

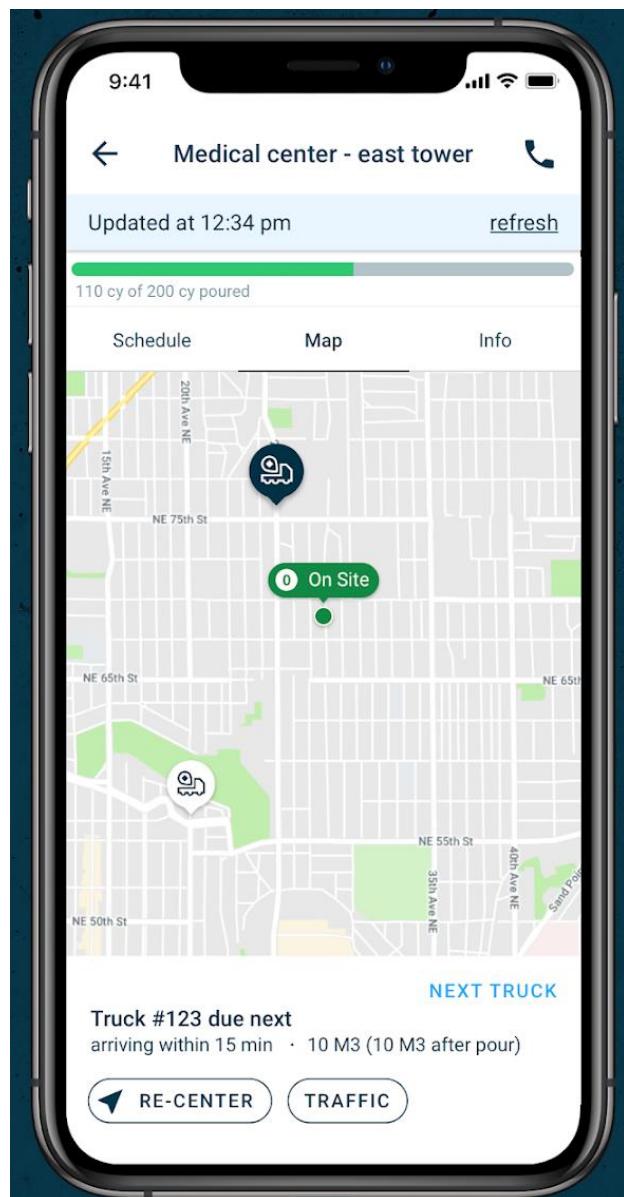
Ясно дефинираните изисквания са основата на успешен проект, тъй като включват набор от процеси като анализ, спецификация и валидиране. Функционалните изисквания са продуктови характеристики, които разработчиците трябва да внедрят, за да позволят на потребителите да изпълнят своите задачи. Като цяло функционалните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

Нека започнем с преглед на характеристиките и изискванията на мобилното приложението. Както беше отбелязано, то представлява приложение за поръчки и проследяване на доставки онлайн. Примерна скица на интерфейса на началния экран, след вход, в на потребителското приложение е даден на фиг. 2.10. Представени са основни елементи на заглавна част – име на текущ потребител, инструмент за избор на дата и списък на текущите поръчки.



*Фиг. 2.10. Скица на основен еcran на приложението.
(разработка на автора)*

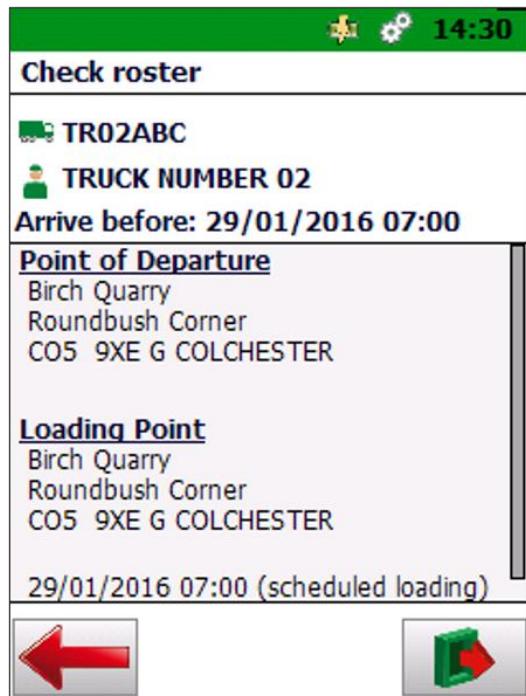
Фигура 2.11. представя детайлна информация за определена поръчка и доставките към нея.



Фиг. 2.11. Скица на еcran за информация за поръчка.
(разработка на автора)

Регистрирането на нови поръчки ще се осъществява чрез еcran в главното меню.

Фигура 2.12 представя използвано на приложението от доставчика. Както беше споменато, мобилното приложение допринася за бързо изпълнение на процесите, сравнително лесно за използване и удобно за работа, чрез функционалностите за достъп до геолокация, навигация, съобщения, телефон. Това допринася за стандарт за сигурност на данни и връзки. Водача може да провери списъка, с предстоящи доставки, назначени към него. Приложението притежава функционалност за уведомяване, в случай че списъкът е експортиран, но няма насрочени задачи.

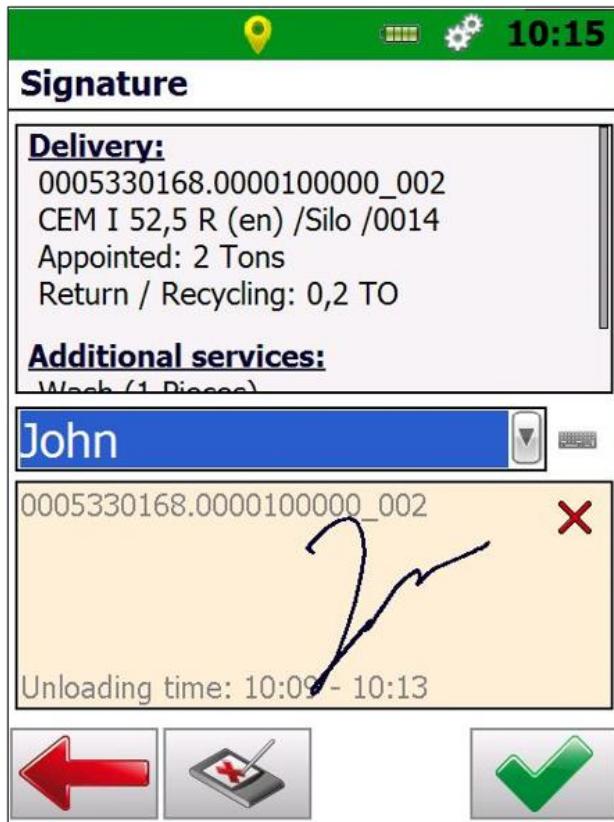


Фиг. 2.12. Скица на начален экран за доставчика. (разработка на автора)

Посоченият еcran съдържа подробности за поръчка, включваща материал, количество, местоположение за товарене и разтоварване и планирани часове. Тъй като, доставката може да бъде анулирана или пренасочена към друга поръчка, остатъкът се докладва и след това бива върнат, използван повторно или отклонен. Също така шофьорът може да съобщи за повреда, като отписването е възможно след изпращане на лог файлове към диспечера.

За да подпомогне автоматизирането и рационализирането на

документацията, приложението поддържа функционалност за електронно доказателство за доставка. Това е процес, който създава документацията, валидираща получаването на стоката от клиента. Традиционно POD се осъществява чрез подпись на клиента на физически документи. В случай на липса на подпись трябва да се посочи причина. Мобилното приложение „улавя“ съответните данни и снимки, като това бива последния етап от доставката. Следната фигура представя экрана за тази функционалност.

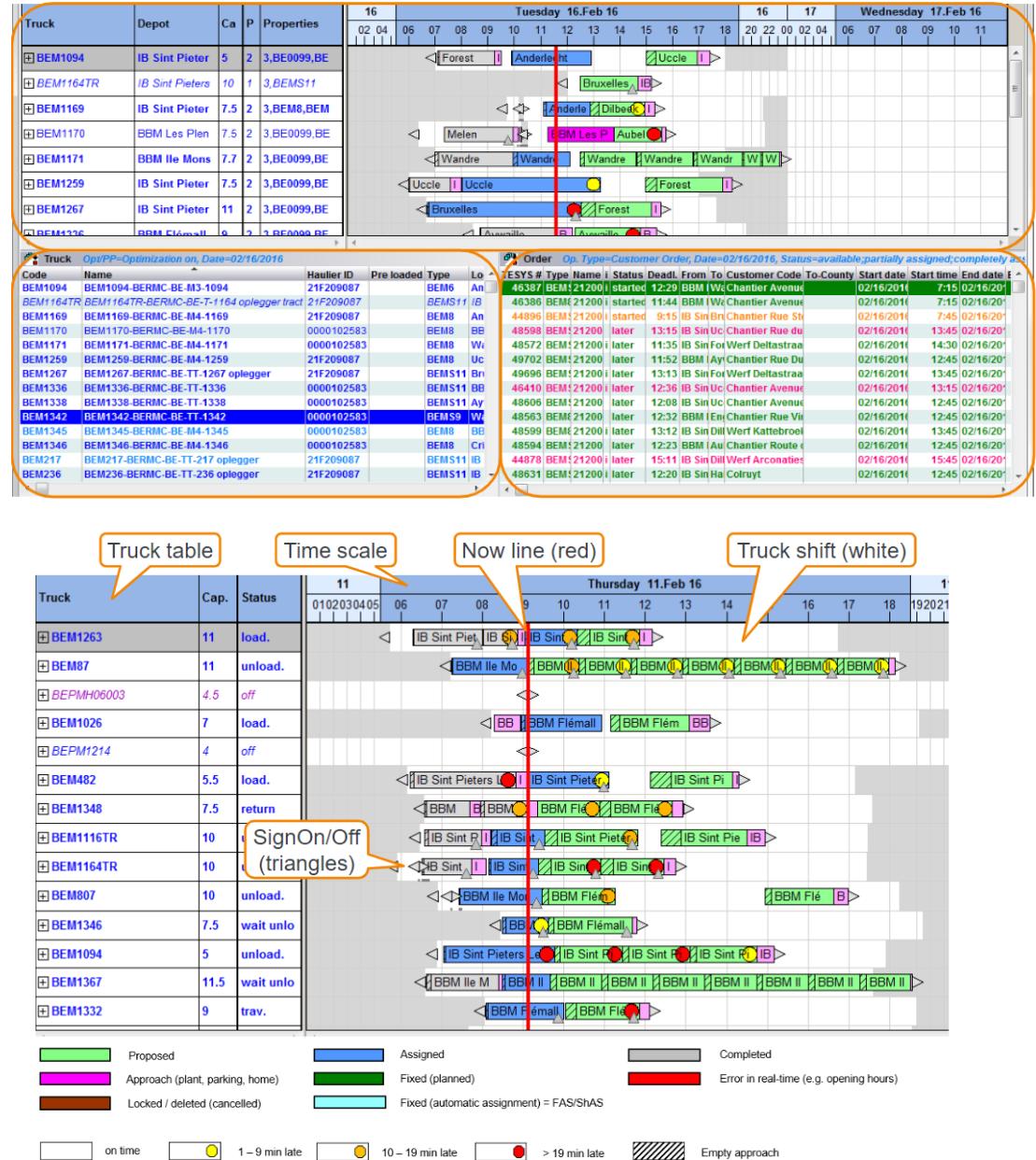


*Фиг. 2.13. Скица на екран за доказателство за доставка (ePOD).
(разработка на автора)*

Когато материалът е доставен, от клиента се иска да потвърди получаването чрез подпись на мобилното устройство, който се предава заедно с допълнителни данни. ePOD документа се изпраща препраща към ERP.

Графичен интерфейс на уеб портала, представен на фигура 2.14., е предназначен за използване от диспечерите за разпределение и планиране на работата. Той представя информация за поръчките, които трябва да бъдат

доставени, като също така дава пълен контрол върху всички превозни средства. Целта към обслужването на клиенти е, да даде представа за организацията през работния ден, да се определи допустимо ниво на резервиране и да се съобщи на поемащите поръчки.



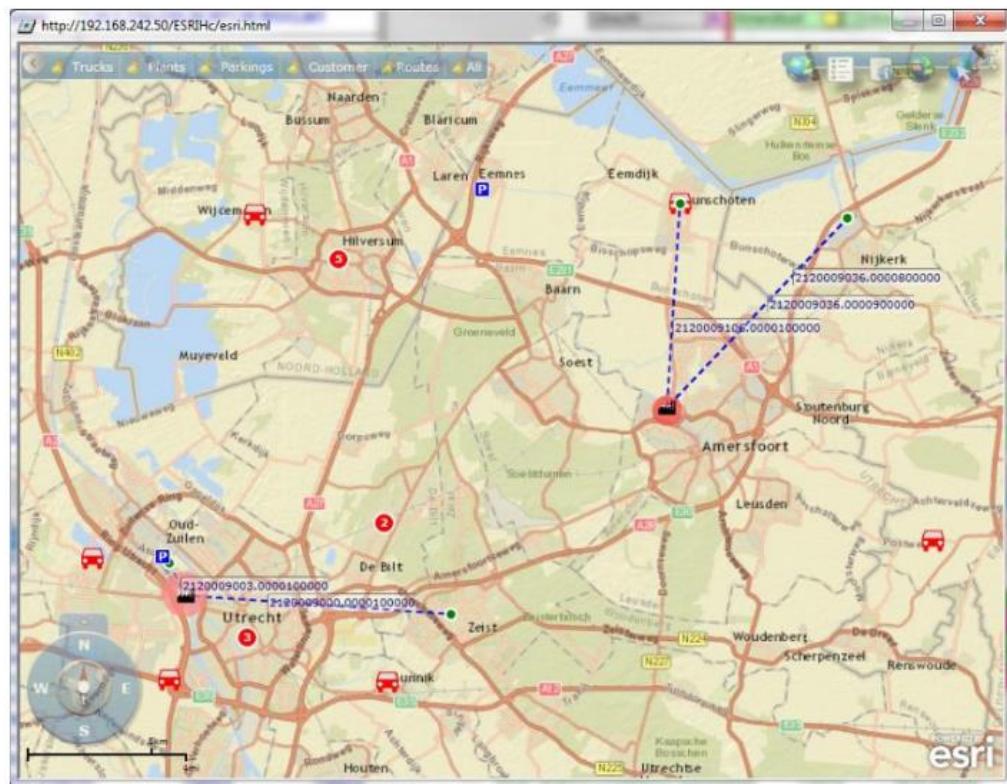
Фиг. 2.13. Главен екран в уеб порталата. (разработка на автора)

Изгледът предлага списък, в който всеки ред съдържа информация за камион със съответните доставки, основни данни и данни за състоянието, всички получени поръчки и планирани доставки. Ширината на колоната

може да се променя чрез пълзгане, в случай, че ширината е твърде малка. Сортирането е възможно по всяка колона (възходящо/низходящо). GUI актуализира на всеки 10 секунди и след всяка транзакция, като анкетира сървъра, който трябва да отговори на всяка заявка за актуализиране поотделно. Сглобяване на всички нови данни изискват изчислителна мощност, затова броят на заявки към сървър е ограничен. Почти пълната оперативна база данни е съхранена в паметта. В оперативната база данни се поддържа само малък набор от данни. Историческите данни не са необходими за оптимизиране и изпращане.

Уеб порталът служи като инструмент, използван за актуализации на състоянията на пристигане/напускане, натоварване и други за камионите, които към определен момент са без дистанционно предаване на данни.

Уеб порталът поддържа функционалност за времето, което отнема на превозното средство за да стигне от точка А до точка Б. По този начин, той служи като инструмент за разстояние и продължителност на пътуването, като например продължителността на пътуването от завода до мястото на клиента. Пътят обратно може да е различен, заради еднопосочни улици. От камиона може да бъде поискано да се върне някъде другаде. Уеб порталът поддържа вътрешна „Матрица за време и разстояние“ (често наричана „Разстояния и продължителности“ за да съхранява заявените времена и разстояния. На следващата фигура е визуализиран изгледат за маршрутизиране.



Фиг. 2.14. Екран за маршрутизиране. (разработка на автора)

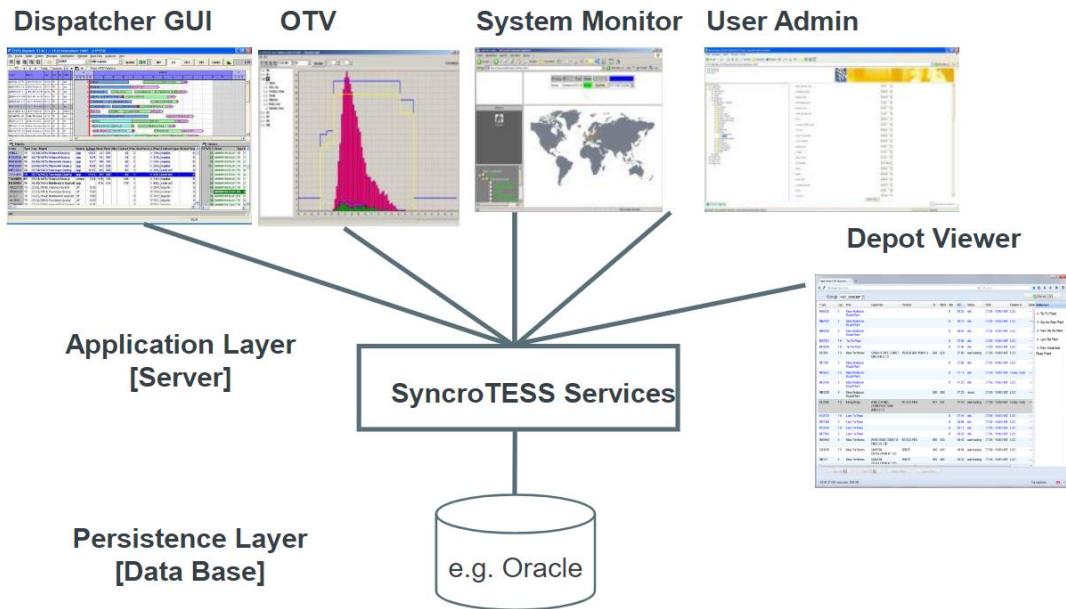
Уеб портала комуникира с камиони и заводи, за текущите статуси на почивки, отчети за състоянието, начало и края на товарите. Информира за задания, анулации и т.н. Уеб портала и мобилното приложение предоставят телематична система, която дава обратна връзка към диспечърите. Уеб портала използва събития за местоположение, за да изгради прогнозната оставаща продължителност на пътуването, от мобилното приложение, което изпраща геокоординати. По подразбиране актуализациите на ETA за отчетено местоположение са на всяка минута, но това може да се промени.

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 /



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 доставки Е неоптимизиран изглед на поръчките• Отразява времената от поръчките• Помалко 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?“

The Material Replenishment(MRP)

За производството на бетон са необходими сировини и SyncroTESS план за група бетони определя търсенето на бетон: ● SyncroTESS експортира търсенето на готов бетон ● на завод ● на продукт ● на интервал от половин час ● SAP използва тази информация, за да изчисли търсенето на сировини ● SAP генерира/актуализира автоматично обобщените поръчки

2.4. Комуникационни модели между програмните интерфейси

Комуникационните технологии са от важно значение за много уеб приложения, в частност системи за електронна търговия или управление на поръчки. Те са част от Световната мрежа, която сама по себе си представлява разпределена система от взаимосвързани ресурси.

Актуалността на изследваната тема се обуславя от тенденцията облачните технологии да се превръщат в инструменти за стратегическа трансформация и дигитализация на бизнеса. Облачните платформи позволяват бърза реализация на иновативните идеи. Това предимство поставя компаниите една стъпка пред конкурентите.

Обект на изследване в настоящия реферат е разпределена информационна система, базирана на микроуслуги, работеща върху множество процеси и сървъри (хостове). Всяка услуга се изпълнява в отделен процес като контейнер, разположен в кълстер от виртуални машини. Приложениета взаимодействват помежду си с помощта на протоколи за комуникация TCP, HTTP и AMQP в зависимост от естеството на работа. Разгърната в облачната платформа Azure, инфраструктурата се управлява от инструмент за оркестрация Kubernetes.

Целта на реферата е да представи информационното взаимодействие между подсистемите за управление, които осъществяват връзка чрез технологии за изпращане и получаване на данни.

Основни задачи, които са поставени:

- да се разгледат актуалните комуникационни модели, които интегрират отделните части на софтуерния продукт;
- да се предложат основни принципи и добри практики при изграждането на облачно базираните приложения;

Основната теза на изследването е свързана с внедряването на бизнес процеси „от край до край“, като същевременно се поддържа последователност и съгласуваност в компонентите на системата. Архитектурата трябва да поддържа важни нефункционални изисквания като: висока степен на достъпност, разширяване на мащаба при увеличаващ се потребителски трафик и други.

Клиентски и сървърни приложения могат да използват различни видове комуникация, насочени към постигането на различни цели. Може да разгранишим два основни типа, които се използват между компонентите на системата: синхронна и асинхронна.

2.4.1. Синхронна комуникация

Уеб услугите са интерфейси, които са предназначени за комуникация между приложения, за разлика от уеб сайтовете, които са насочени към взаимодействието с хората и се достъпват през браузър (Бенджамин, 2012). Клиенти на уеб услуга могат да бъдат както мобилни или базирани в потребителския браузър приложения, така и други уеб услуги.

В синхрония подход клиентът изпраща HTTP заявка към услуга, която я обработва и връща обратно HTTP отговор. Клиентският код може да продължи своята задача само след получаване на отговор. Заявка и отговор имат обща структура:

- Начален ред, описващ текущия HTTP метод, адрес, статус и протокол;
- Заглавни редове (headers) – дават възможност на клиента и сървъра да предадат допълнителна информация;

- Метаданни за двета HTTP компонента;
 - Тяло, съдържащо данни, свързани със заявката или отговора;
- Фигура 1.1 илюстрира примерна HTTP заявка/отговор.

```
GET /html/rfc1945 HTTP/1.1
Host: tools.ietf.org
User-Agent: Mozilla/5.0 (Ubuntu; X11; Linux x86_64; rv:9.0.1) Gecko/20100101
           Firefox/9.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: de-de,de;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Connection: keep-alive
If-Modified-Since: Sun, 13 Nov 2011 21:13:51 GMT
If-None-Match: "182a7f0-2aac0-4b1a43b17a1c0;4b6bc4bba3192"
Cache-Control: max-age=0

HTTP/1.1 304 Not Modified
Date: Tue, 17 Jan 2012 17:02:44 GMT
Server: Apache/2.2.21 (Debian)
Connection: Keep-Alive
Keep-Alive: timeout=5, max=99
Etag: "182a7f0-2aac0-4b1a43b17a1c0;4b6bc4bba3192"
Content-Location: rfc1945.html
Vary: negotiate,Accept-Encoding
```

Фигура 1.1. Примерна HTTP заявка/отговор

Източник: Бенджамин Е, 2012

2.4.1.1. Механизъм за трансфер на репрезентативно състояние

Representational State Transfer (REST) представлява софтуерна архитектура за проектиране на уеб услуги. Представена е през 2000г. като част от дисертацията на Рой Т. Филдинг. Продуктите, използващи REST, са базирани на хипермедија. REST е независим от протоколите на приложния слой, като концептуалните идеи зад него са взети от HTTP и WWW (Прайс, 2022).

Основно предимство на REST е, че той използва отворени стандарти и не обвързва внедряването на API или клиентските приложения с конкретна реализация.

REST API са проектирани около ресурси/бизнес обекти. В системата за управление това са потребители и поръчки. Създаването на поръчка може

да се постигне чрез изпращане на HTTP POST заявка, която съдържа определена информацията. HTTP отговорът показва дали поръчката е създадена успешно или не.

REST е архитектура за моделиране на обекти и операции, които приложението изпълнява. Ресурсите често се групират в колекции. Колекцията е отделен ресурс и притежава собствен идентификатор (Petersen, 2022). Добра практика е URI да се организират в йерархия, като например: <https://manager.com/orders> връща колекцията от поръчки. Всеки елемент в колекцията има свой собствен уникален идентификатор, като например този за конкретна клиентска поръчка може да бъде <https://manager.com/orders/eu.123123.231>. Друг примерен идентификатор: <https://manager.com/orders/eu.123123.231/deliveries> представя доставки за поръчка. Важно да отбележим, че това ниво на сложност може да бъде трудно за поддържане, ако връзките между ресурсите се променят в бъдеще.

HTTP протоколът дефинира редица методи, показани в таблица 1.1, които осигуряват различна семантика, когато се прилагат към ресурс:

Таблица 1.1: Методи на протокола HTTP 1.1.

Метод	Описание
GET	Най-разпространеният метод. Използва се за извличане на репрезентации на ресурси. Информацията се съдържа в отговора.
HEAD	Подобен на GET, но без обект в отговора.
PUT	Заменя ресурса в посочения URI. Тялото на заявката описва ресурса, който трябва да бъде актуализиран.
DELETE	Премахва ресурса в посочения URI.
POST	Създава нов ресурс. Тялото на заявката предоставя подробности за новия ресурс.
OPTIONS	Представя мета данни за ресурс.
PATCH	Извършва частична актуализация на ресурс.

Резултат на конкретна заявка зависи от това дали ресурсът е колекция или отделен елемент. Следващата таблица описва общите конвенции, приети от повечето RESTful реализации, използвайки примера за управление на поръчки.

The following table displays the API actions:

Functionality	Path
Get all the updates for a specific house's devices for a given time range	GET <code>/api/house/<i>houseId</i>/devices/update s?from=<i>from</i>&to=<i>to</i></code>
Get the updates for a specific device for a given time range	GET <code>/api/device/<i>deviceId</i>/updates?from=<i>from</i> &to=<i>to</i></code>
Get the current status of all the devices in a specific house	GET <code>/api/house/<i>houseId</i>/devices/status/current</code>
Get the current status of a specific device	GET <code>/api/device/<i>deviceId</i>/status/current</code>

Таблица 1.2: Общи REST конвенции.

Ресурс	GET	POST	PUT	DELETE
/orders	Извлича всички поръчки.	Създава нова поръчка.	Общо актуализиране на поръчките.	Премахва всички поръчки.
/orders/1	Извлича детайли за поръчка 1.		Актуализира данните за поръчка 1, ако съществува.	Премахва поръчка 1.
/orders/1/ deliveries	Извлича доставките за поръчка 1.	Създава нова доставка за поръчка 1.	Общо актуализиране на доставките за поръчка 1.	Премахва всички доставки за поръчка 1.

В HTTP протокола форматите се определят чрез използване на типове медији, наричани още MIME. За недвоични данни, повечето уеб API поддържат JSON (`application/json`) или XML (`application/xml`) като формат за обмен. Те се използват за представяне на структурирани данни. Например, заявка към посочения по-горе URI за детайли на поръчка, ще върне следния отговор във формат JSON:

```
{
    "orderId":eu.123123.231,
    "orderValue":99.90,
    "productId":1
}
```

Сървърът информира клиента за резултата от заявка чрез използване на предварително зададени “кодове на състоянието”, представени в следната таблица.

Таблица 1.3. Таблица с диапазоните на HTTP кодовете.

Статус код	Тип	Описание	Пример
1xx	Информационен	Изпраща се с подгответелна цел.	100 Continue
2xx	Успех	Заявката обработена успешно.	200 OK
3xx	Пренасочване	Клиентът трябва да изпрати допълнителни заявка.	301 Redirect
4xx	Грешки в клиента	Резултат от грешна заявка, причинена от клиента.	404 Not Found
5xx	Грешка в сървъра	Грешка от страна на сървъра.	503 Service Unavailable

REST е подходящ за CRUD-базирани операции. Клиентите взаимодействат със сървъра през HTTP. REST е широко разпространен и е поддържан от повечето работни рамки като ASP.NET, Symfony, Spring, Node.js и други.

2.4.1.2. Механизъм за заявки към отдалечени процедури

gRPC е високопроизводителна рамка, която позволява дистанционно извикване на процедури (RPC). На ниво приложение, gRPC структурира съобщенията между клиенти и бек-енд услуги. Произхождащ от Google, това е проект с отворен код и част от Cloud Native Computing Foundation.

Клиентско gRPC приложение създава локална функция в уеб услуга, която реализира бизнес операция. Тази локална функция извиква друга функция на отдалечена машина (Vettor, 2022).

В приложенията, базирани на облак, разработчиците често работят

на различни езици за програмиране, рамки и технологии. gRPC осигурява „хоризонтален слой“, който помага за съвместимостта между компонентите.

gRPC използва HTTP/2 като транспортен протокол, който разполага с разширени възможности (Vettor, 2022):

- Двоичен протокол за транспортиране на данни, за разлика от HTTP1.1, който е текстов;
- Поддържа изпращане на множество паралелни заявки през една и съща връзка;
- Компресира съдържанието на съобщенията, което намалява натоварването на мрежата;

Механизмът е базиран на технология с отворен код, наречена Protocol Buffers. Файловете .proto осигуряват висока ефективност и платформено-неутрален формат за структуриране на съобщения. Използвайки междуплатформен език за дефиниране на интерфейс (IDL), разработчиците дефинират “договор” за всяка микроуслуга. Договорът, реализиран като текстов .proto файл, описва методи, входове и изходи. Използвайки прото файла, компилаторът може да генерира както клиентски, така и сървърен код за целевата платформа.

Кодът включва следните компоненти:

- Строго обособени обекти, споделени от клиента и услугата, които представляват елементи от съобщението;
- Базов клас, който отдалечената gRPC услуга може да наследява;

Може да разгледаме следния пример за order_delivery.proto.

Фигура 1.2. Protobuf файл за интегриране на микроуслугата за поръчки.

```
syntax = "proto3"; // версия на синтаксиса

option csharp_namespace = "Manager.Protos";

package order_delivery; // идентификатор на пакета

import "google/protobuf/wrappers.proto";

service OrdersDeliveries {
    rpc GetOrder (GetOrderRequest) returns (NullableOrder); // метод за
извикване
}

message GetOrderRequest { // формат на съобщението
    google.protobuf.StringValue order_nr = 1;
}

message Order { // формат на отговора
    google.protobuf.StringValue order_nr = 1;
}
```

Източник: Разработка на автора

По време на изпълнение всяко съобщение се сериализира като стандартно представяне на Protobuf и се обменя между клиента и отдалечената услуга. Единицата за обем на информация се сериализират в двоичен вид (Smith, 2022).

2.4.2. Асинхронна комуникация

Съществен момент при изграждането на ориентираната към услуги архитектура е интеграцията помежду им. Комуникацията между подсистемите трябва да бъде сведена до минимум. Целта на всеки микросървис е да той бъде автономен и достъпен за потребителя, дори ако другите, които са част от приложението, не работят. Силна зависимост в архитектурата може да опишем в случай, когато трябва да се извърши повикване от една микроуслуга към друга (като изпълнение на HTTP заявка за получаване на данни), за да може да се обработи отговор. Програмният продукт няма да бъде устойчив ако някоя от частите се срине. Освен това, създаването на вериги от заявки/отговори, намалява значително производителността на цялото приложение.

2.1 Базирана на съобщения комуникация

Асинхронните съобщения и управляемата от събития комуникация са от критично значение при разпространението на промените в множество микроуслуги и свързаните с тях домейн модели. Когато настъпят промени, системата се нуждае от начин за съгласуване в различните модели. Решението: евентуална последователност и комуникация, управлявана от събития, базирана на асинхронни съобщения.

Клиент прави заявка към услуга, като ѝ изпраща съобщение. Тъй като това е асинхронна комуникация, клиентът приема, че отговорът няма да бъде получен веднага или че няма да има отговор. Съобщението се състои от заглавие (метаданни като информация за идентификация) и тяло. Съобщенията се изпращат чрез асинхронни протоколи като AMQP. Част от предпочитаната инфраструктура за този тип е брокер на съобщения, който е различен от тези, използвани в SOA. В опростен вариант, той действа като шина.

Налични са различни брокери на съобщения като всеки един има низ за връзка и потребителски достъп. Реалната част на процеса се състои в

крайните точки, които публикуват и получават съобщения, тоест в микроуслугите. Важно правило, което системата за управление се опитва да спазва, доколкото е възможно, е да използва само асинхронни съобщения между вътрешните услуги, а синхронна комуникация: само от клиентските приложения към API Gateway.

Когато се използва асинхронна комуникация, управлявана от събития, микроуслуга публикува интеграционно съобщение, задействащо се когато нещо се случи в нейния домейн и друга микроуслуга трябва да получи информация за това. Пример е промяна на цената в микроуслуга от продуктов каталог. Микроуслугите се абонират за събитията, за да могат да ги получават асинхронно. Когато това се случи, получателите могат да актуализират собствените си обекти. Тази система за публикуване/абониране се реализира чрез по-горе споменатия брокер.

2.2 Съгласуваност между услугите

Високата степен на наличност и толерантност към частични проблеми не могат да бъдат гарантирани на 100% в разпределените системи. От гледна точка на широкомащабните уеб архитектури, това е от важно значение за съхранението в услугите, тъй като тези компоненти запазват текущите състояния на приложението.

Условието при внедряване на бизнес процеси от край до край, е същевременно да се поддържа последователност в услугите.

Важни изисквания за последователността са:

- Нито една услуга не трябва да включва таблици/хранилища за данни от друга и не трябва да извиква директни заявки към тях;
- Предизвикателство при възникване на частична повреда – колкото по-свързани са частите на системата, толкова по-голям проблем.

2.4.3. Комуникационни модели за достъп до бекенда

За да бъдат нещата опростени, клиент от „предния край“ може да комуникира директно с микроуслугите. Следната фигура показва този

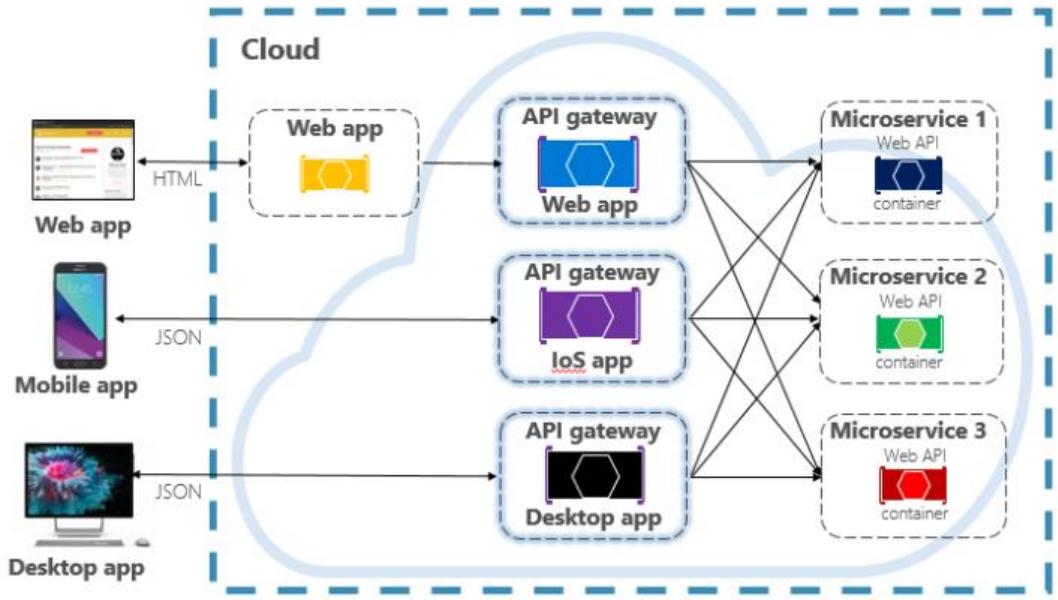
вариант. Този подход се използва, когато различни части от страницата на клиента изискват различни микроуслуги. Всяка микроуслуга има публична крайна точка, която е достъпна от клиентските приложения. Макар и лесна за изпълнение, директната комуникация с клиента би била приемлива само за прости микросервизни приложения. Този модел свързва тясно клиентите от предния край с основните бек-енд услуги, което води до редица проблеми, включително:

- Микроуслугите трябва да бъдат изложени на „външния свят“;
- Междусекторни проблеми като удостоверяване и оторизация;
- Сложен клиентски код - клиентите трябва да следят множество крайни точки и да се справят с възможни неуспехи;

Като надграждане на първия модел за проектиране, облачната инфраструктура позволява да се внедри API Gateway. Тя предоставя единична точка за група микроуслуги. Наподобява модела за дизайн: „фасадата“. Известен е също като „backend for frontend“. Изгражда се за конкретните нужди на клиента. Действа като пълномощник между клиентите и микроуслугите. Може да осигури удостоверяване, кеширане или други. Azure предоставя няколко готови продукта:

- Azure Application Gateway - насочен към .NET, работещ с архитектура, ориентирана към микро услуги, предоставящ унифицирана входна точка към системата. Услугата поддържа възможности за балансиране на натоварването;
- Azure API Management - шлюз, който позволява контролиран достъп до бек-енд услуги, базиран на конфигурируеми правила. Представя уеб портал на разработчиците, които могат да го използват за инспектиране на услугите и анализиране на тяхното натоварване.

Шаблонът е показан на следната фигура:



Фигура 2.2. Комуникация чрез шлюз за приложните програмни интерфейси.

Източник: Smith, S., 2022

API шлюзът може да се превърне в “анти-модел“ като монолитно приложение, съдържащо твърде много крайни точки, обединяващо всички микроуслуги. API шлюзовете трябва да бъдат разделени от логически групи въз основа на бизнес ограничения. Протоколи за пренос на данни могат да бъдат HTTP или gRPC.

TODO:

[Design Microservices Architecture with Patterns & Principles \(udemy.com\)](#)

Design Microservices Architecture with Patterns Best Practices

<https://heidelbergmaterials.udemy.com/course/software-architecture-system-design-practical-case-studies/>

Design a Scalable Ride Sharing Service

System+Design+Practical+Case+Studies++Course+Workbook.pdf

[Fundamentals in Oracle Transportation Management \(OTM\) Cloud \(udemy.com\)](#)

oracle-transportation-management-cloud-ds.pdf

[Software Architecture & Technology of Large-Scale Systems \(udemy.com\)](#)

Software Architecture & Technology of Large-Scale Systems.pptx

(Performance, scalability, reliability, security, deployment, tech stack)

<https://heidelbergmaterials.udemy.com/course/the-complete-cloud-computing-software-architecture-patterns>

more patterns

Глава 3. Изграждане на облачна система за производствено предприятие Holcim Group

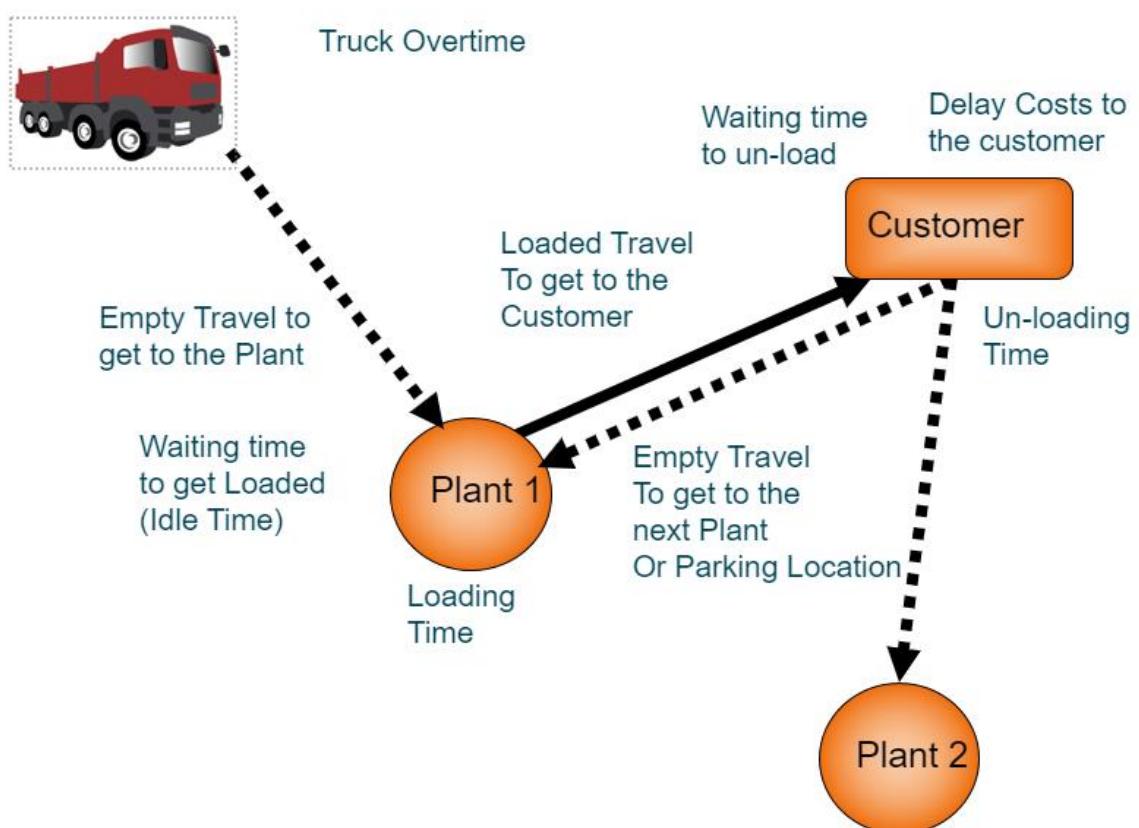
Интрo

3.1. Обща характеристика на дейността на компанията *Holcim Group*

Abvc

3.1.1. Основни бизнес процеси в компанията

Продуктите на компанията се използват за изграждане на къщи, инфраструктура, търговски и промишлени съоръжения, като по този начин отговарят на нуждите на нарастващото световно население за жилища, мобилност и икономическо развитие. Основната дейност включва производство и дистрибуция на цимент, инертни материали, готови бетонови смеси и асфалт.





Преход от хартиени документи към цифрови документи за доставки на бетон. Цифровите документи и цялата информация за доставка ще бъдат достъпни чрез приложението. Някои от плюсовете са:

- Без повече липсващи документи;
- Край на събирането и съхраняването на документи;

Те ще съдържат информация за поръчки, документи, фактури, резултати от тестове и др.

OnSite е нашето приложение, което ви помага да управлявате и проследявате напредъка на вашите конкретни доставки в движение в реално време.

Hub е нашият онлайн портал за управление, проследяване и свързване на цялата информация, свързана с продукта. Вашите поръчки, документи, фактури и протоколи от тестове – всичко това е на едно място.

Приветстваме ви в нашето безхартиено пътуване през нашите дигитални платформи.

3.1.2. Стимулиране на продажбите чрез цифрови технологии

Пакета от приложения стимулира прозрачността на данните, стандартизация в ERP, по-бързо и рентабилено планиране в заводите, създават иновативно решение, ориентирано към потребителите. Услугите с добавена стойност, правят клиентите по-логистично интегрирани и по-добре оборудвани за да посрещнат предизвикателствата, свързани с устойчивостта. Целят да намалят материалните и логистични разходи, въглеродния отпечатък, да подобрят производителността и клиентския опит.

Проследяване на камиона с готов бетон по пътя му към строителна площадка, съхраняване на билети и протоколи едно място, с мобилно и уеб приложение. Благодарение на контрола, циментът се произвежда по по-устойчив и ефективен начин, който намалява отпечатък върху околната среда и осигурява безопасност на работниците. Обслужването на клиенти се рационализира чрез постоянната връзка с бек-офиса и превозвача.

Технологичният пакет се основава на авангардни технологии с отворен код, с най-новите програми и езици. Контейнерни услуги, работещи изцяло в облак и разпределени в множество регионални кълстери.

3.2. Избор на технологични средства за разработка и операции

Тази подточка, ще се опише едно от най-важните решения, най-вече защото е почти необратимо. Основни съображения за изпълнение на задачата са общност в Stack Overflow, популярност според Google Trends и други.

Като метод за вземане на решене, базирано на данни, се представя анализ на разходите и ползите, който представлява процес на сравняване на прогнозираните или очакваните разходи и възможности (). Ако прогнозираните ползи надхвърлят разходите, можете да се твърди, че решението е добро. Показателите, използвани за измерване и сравнение са:

- Тип - статичен или динамичен;
- Зависим от платформа/инфраструктура;
- Обюоност;
- Производителност;
- Крива на обучение;

Таблица 3.3 представят анализ на сървърните технологии, подходящи за изпълнение на заданията.

Таблица 3.3: Сравнение на сървърни технологии за разработка.

	App Types	Type System	Cross Platform	Community	Performance	Learning Curve
.NET	All	Static	No	Large	OK	Long
.NET Core	Web Apps, Web API, Console, Service	Static	Yes	Medium and growing rapidly	Great	Long
Java	All	Static	Yes	Huge	OK	Long
node.js	Web Apps, Web API	Dynamic	Yes	Large	Great	Medium
PHP	Web Apps, Web API	Dynamic	Yes	Large	OK -	Medium
Python	All	Dynamic	Yes	Huge	OK -	Short

.NET е инструмент за разработка на софтуер, който Microsoft създава

за собствената си екосистема от продукти и услуги. Той е стандарт, който разработчиците използват за създаване на софтуерни програми, които биват съвместими с технологичния продукти на Microsoft. Стартоването на .NET Framework се случва през 2002 г., с въвеждането на езикът за програмиране C#. Той предлага библиотеката с GUI настолни приложения, уеб рамката ASP.NET и функцията за достъп до данни ADO.NET. Цялата операция разчита на Common Language Runtime (CLR), тъй като това позволява компилирането и изпълнението на управлявания код. Има много причини, поради които .NET Framework и семейството .NET като цяло се използват, като например: обектно-ориентиран, система за кеширане, интегрирана среда за разработка на Visual Studio, универсални стандарти, общност, автоматичен мониторинг.

Въпреки всичко това, той идва и с доста неуспехи, като: проблеми с обектно-релационна поддръжка, тъй като съществуват опасения относно гъвкавостта на работната рамка по отношение на новите проекти на бази данни и тяхната поддръжка. Неприятен факт също е, че тъй като пакетът .NET е към Microsoft, всички промени или ограничения, които компанията може да наложи, неизбежно ще повлият на проекти, изпълнявани под рамката. Също така и цената на лицензите.

Java е един от най-широко разпространените езици за програмиране.

Причина за това е, че има много предимства, които помагат на програмистите да решават сложни проблеми от реалния свят. Съвместимостта на Java не зависи от операционната система или хардуера, което го прави независим от платформа. Подобно на .NET, Java е обектно-ориентиран език от високо ниво, осигуряващ автоматично освобождаване на паметта, многонишков с ефективна стратегия за разпределение на паметта.

Недостатъците на Java са сравнително слабата производителност, заемане на значителна част от паметта, платен търговски лиценз и други.

В днешно време хората широко използват езика PHP за създаване и разработване на уеб приложения. Сега той се превърна в един от основните езици за разработчиците, докато създават ново приложение. Водещи сайтове като Facebook и Харвардския университет използват езика PHP и го направиха по-популярен и надежден. Използването му се разви драстично през годините и сега хората го използват като прост инструмент за програмиране за разработване на уеб сървъри. Има много добродетели или предимства и няколко недостатъка. Нека ги разгледаме систематично. Някои от най-важните предимства на PHP са следните: отворен код, независимост от платформа, лесно зареждане, удобен за потребителя, стабилен, връзка с база данни, поддръжка на библиотека и други. Въпреки че PHP е фантастичен език за програмиране, той има някои недостатъци като: проблеми със сигурността, лоша производителност, не подходящ за разработка на гигантски приложения, обработката на грешките и други.

Node.js е сървърно приложение като Apache, IIS, TOM и д.р, което изпълнява JavaScript. Самият Node не е изцяло написан на JavaScript, а основно на C. Node е платформа с отворен код за създаване на мрежови приложения в реално време, предоставя асинхронни, управлявани от събития входно/изходни операции. Плюсовете са: асинхронното управление помага за едновременното обработване на заявки, споделя база с код както на сървъра, така и на клиента, активна общност и други. Минусите му са, че Node.js не предоставя възможност за мащабиране, за да се възползва от множеството ядра, работа с релационни бази данни, не е подходящ за задачи, натоварващи процесора и други.

Проучването на StackOverflow () показва, че Python е обявен за втория най-бързо развиващ се език за програмиране на разработчиците след Rust. Тъй като е многофункционален език, той е предпочитан избор за предприемачи, които търсят проекти за машинно обучение и Data Science. От гледна точка на навлизане на пазара на труда, студентите предпочитат Python, тъй като е лесен за разбиране и кодиране. Освен това много организации използват Python за много от своите проекти.

Плюсовете са, че е удобен за начинаещи, притежава голяма общност, както и обширна работна среда на библиотеки, машабирам и други.

Минусите са, че е по-бавен от компилираните езици, притежава висока консумация на памет и други.

Таблица 3.4: Сравнение на мобилни технологии за разработка.

	Native	Hybrid	Cross Platform
Development Language & IDE	iOS – Objective-C or Swift, with X-Code & iOS SDK Android – Java with Android Studio & Android SDK	Thin wrapper around HTML, JavaScript, CSS	Xamarin (C#, Visual Studio) React Native (JavaScript)
Access to Phone's Features	Full control, no limits	Very limited	Catch-up with latest versions
User Experience	Exceptional	Inferior	Good, with limitations

Keep an eye on PWA!

TODO:

<https://heidelbergmaterials.udemy.com/course/microsoft-azure-cloud-architecture-case-studies>

Base cloud – on premise overview

Case studies

[The Cloud Strategy Master Class \(udemy.com\)](https://udemy.com/the-cloud-strategy-master-class)

SWAT anazis

<https://heidelbergmaterials.udemy.com/course/microservices-architecture-the-complete-guide>

ultimate solution all in for microservices

3.4. Приложение на избраните технологии за изграждане на инфраструктурата в облачно базирана среда

По примери и указания от глава 2, тази подточка разглежда осъществяването на опростен във функционално отношение, облачен продукт, демонстриращ използването на .NET, Docker, Kubernetes в облачната среда на Microsoft Azure.

3.4.1 Хранилищата за данни в подсистемите

Различните back-end услуги, използвани от системата, имат различни изисквания за съхранение на данните. Azure предоставя много видове хранилища за данни, които могат да помогнат за поддръжка и извлечане:

Azure SQL Database - Облачно базиран SQL Server. Поведението му е същото като това на основното изпълнение на базата, но предлага и много предимства: репликира в реално време данни в други географски региони, маскира данни за определени потребители, предоставя пълен одит на всички действия, които са се случили върху данните. Услугата е използвана от подсистемите за удостоверяване и каталогът за продуктите.

Azure Cosmos DB е нов вид нерелационна база данни, която работи с механизъм за съхранение и предоставяне на данни, който използва свободен модел, също така включва ниска латентност, репликация на данни в други географски региони в реално време, управление на трафика, автоматично индексиране на данните. Услугата е използвана от маркетинговата част.

Azure Blob представлява хранилище за съхраняване на големи неструктурирани данни. Това могат да бъдат фактури, изображения, видео, файлове и други. Услугата е използвана от подсистемата за поръчки.

Допълнение, Azure предоставя услуги за бази данни MySQL, PostgreSQL и MariaDB като универсално достъпни, мащабируеми, силно защитени и напълно управлявани.

Azure предоставя две хранилища за данни, които са много

подходящи за съхранение на големи количества с цел анализ: Data Warehouse & Data Lake.

На фиг. 6 са показани различните услуги според структурата и характеристиките на данните.

	Database	Cosmos DB	Blob	Table	File	PostgreSQL, MySQL	SQL Data Warehouse	Data Lake Store
Relational data	✓					✓	✓	✓
Unstructured data		✓	✓					✓
Semistructured data		✓		✓				✓
Files on disk					✓			
Store large data		✓	✓		✓	✓	✓	✓
Store small data	✓	✓	✓	✓	✓	✓	✓	✓
Geographic data replication	✓	✓	✓	✓	✓	✓		
Tunable data consistency		✓						

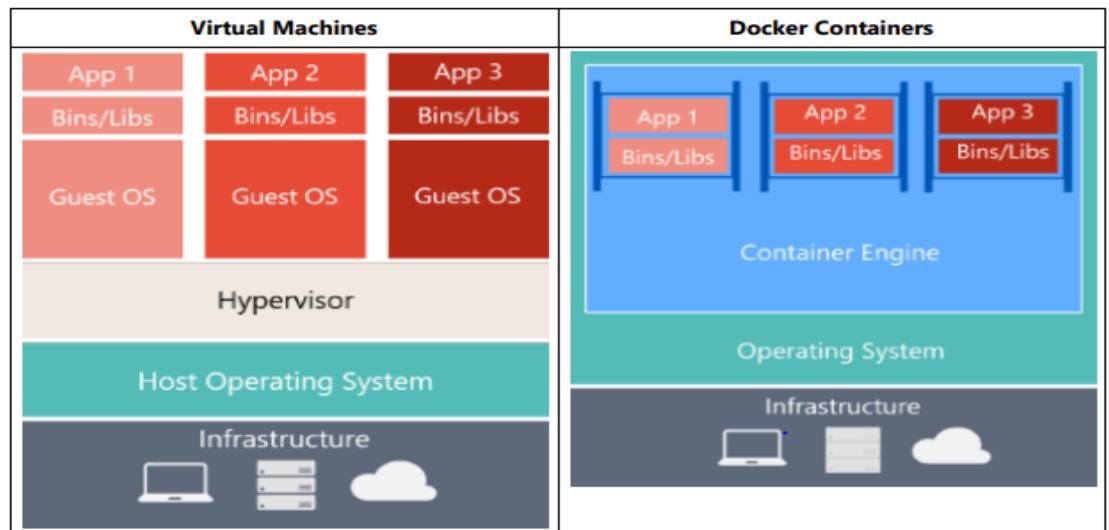
Фиг. 6. Показва коя услуга за данни да се използва при определен сценарий

3.4.2. Софтуерното внедряване и поддръжка в облачна среда

Най-използваната и наложила се като стандарт технология е **Docker**.

Това е проект с отворен код за автоматизиране на внедряването на приложения като преносими, самодостатъчни контейнери, които могат да работят локално или в облака. Също така е компания, която популяризира и развива тази технология. Docker контейнерите могат да работят върху Linux или Windows. Предимства за разработчиците са: ускорено въвеждане на нови програмисти в проекта, премахнете конфликтите в приложението, актуализиране и миграция на софтуера.

На фиг. 7 е представено сравнение между виртуална машина и Docker контейнер.



Фиг. 7. Виртуални машини и Docker контейнерите

Виртуалните машини включват приложението, необходимите библиотеки и пълна операционна система. Изиска пълна виртуализация повече ресурси, повече време за стартиране в сравнение.

Докер контейнерите включват приложението и всички негови зависимости. Те обаче споделят ядрото на ОС с други контейнери, изпълняващи се като изолирани процеси в потребителското пространство на хост операционната система. (с изключение на Hyper-V контейнери, където всеки контейнер работи вътре в специална виртуална машина).

Виртуалните машини имат три основни слоя: инфраструктура, хост,

операционна система, Hypervisor и всички необходими библиотеки. Слоевете в Docker са инфраструктурата, ОС и двигател за контейнери, който поддържа изолация, но споделя основните услуги на ОС. Тъй като контейнерите изискват много по-малко ресурси (например не се нуждаят от пълна ОС), те са лесни за изпълнение, внедряване и започват бързо. Основната цел на изображението е да направи зависимостите еднакви в различните среди. Това гарантирана еднакво поведение на всички среди: локална среда, среда за разработка или продуктивна.

Azure предоставя услуги, които могат да помогнат за постигане на много неща, вариайки от обикновени, като създаване на ново приложение с база от данни – до по-развити като създаване на работни потоци за непрекъсната интеграция (CI) и внедряване (CD). Това са само няколко примера за някои често срещани работни похвати. Много от тях трябва да бъдат създадени индивидуално, но облачната инфраструктура предлага всичко това като услуги. Силата на облака е, че ресурсите са невероятно устойчиви, малко вероятно е аварийно да спрат работа, тъй като центровете за данни са разположени по целия свят, състоящи се от десетки хиляди сървъри. Ако един сървър се повреди, друг поема управлението. Един от най-убедителните аргументи в полза на облака е, че може да разширява мащаба на услуги и ресурси почти безкрайно, в определени моменти, като например "Черен Петък" или голяма маркетингова кампания с промоции и намаления на артикули. Също така, когато натоварването намалее, мащабът може да се намали до обикновените си параметри. Уважавани и опитни облачни доставчици като Microsoft разпознават моделите на използване на нормалните потребители и тези на злонамерените. Интелигентни инструменти за наблюдение, алгоритми за обучение и изкуственият интелект предоставят възможност да откриват атаки. При стаптиране на приложения в Azure едно от първите решения, които трябва бъдат вземети, са планираните за използване услуги:

- Azure App Services - един от най-лесните и мощни начини за хъстване на приложения. Той е предпочитан при монолитната архитектура. Услугите са достъпни и работят в 99,95% от времето. Споделят мощни функции като автоматично мащабиране, внедряване с нулев застой и лесно удостоверяване, позволяват отстраняването на грешки в приложението докато работи в производствена среда (със Snapshot Debugger). По подразбиране приложението ще бъде достъпно в интернет, без да е необходимо да се настройва име на домейн или да се конфигурира DNS. Работи много добре с контейнери.

- Azure Virtual Machines - позволява преместване на съществуващи приложения от виртуални машини, които вече се изпълняват във център за данни. Има много предварително дефинирани изображения, които могат да бъдат използвани като Windows Server, който работи с IIS и има инсталиран и предварително конфигуриран ASP.NET на него, както и собствени софтуерни лицензи (като за SQL Server). Услугата е подходяща за миграция на т.нар. „наследена система“, която да бъде използвана като подсистема или източник на данни.

Следната таблица представя услугите и техните най-чести случаи на употреба:

	App Service Web Apps	App Service Mobile Apps	Azure Functions	Logic Apps	Virtual Machines	Azure Kubernetes Service (AKS)	Container Instances
Monolithic and N-Tier applications	✓				✓ *		✓
Mobile app back end		✓			✓ *		
Microservice architecture-based applications			✓			✓	
Business process orchestrations and workflows			✓	✓			

Фиг. 8. Представя кои услуги на Azure са подходящи за различните типове.