ИКОНОМИЧЕСКИ УНИВЕРСИТЕТ – ВАРНА
ФАКУЛТЕТ ИНФОРМАТИКА
КАТЕДРА ИНФОРМАТИКА

Светослав Стефанов Иванов

Управление на разработката, поддръжката и съпровождането на софтуер от технологични стартиращи компании

ДИСЕРТАЦИЯ

за присъждане на образователна и научна степен „доктор”
по докторска програма
„Приложение на изчислителната техника в икономиката”

Научен ръководител: доц. д.н. Павел Петров

ВАРНА
2022

СЪДЪРЖАНИЕ

Списък на използваните съкращения

БМ	Бизнес Модел
БСВС	Бизнес Сценарии за Взаимодействие със Системата
ИБМ	Информационно Бизнес Моделиране
ИМТИ	Интегриран Модел на Технологични Иновации
КцБМ	Концептуален Бизнес Модел
МСП	Малки и Средни Предприятия
ОБМ	Обектен Бизнес Модел
РП	Работни Продукти
СУБД	Система за Управление на Бази от Данни
AARRR	Acquisition, Activation, Retention, Referral, Revenue
API	Application Programming Interface
CC	Crystal Clear
CRM	Customer Relationship Management
CSV	Comma-Separated Values
DSDM	Dynamic Systems Development Method
ERP	Enterprise Resource Planning
HML	High, Middle, Low
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
MoSCoW	M – Must have; S – Should have; C – Could have; W – Won't have
MVP	Minimum Viable Product
PMI	Project Management Institute
RAD	Rapid Application Development
RUP	Rational Unified Process
SCM	Supply Chain Management
SCRUM	Systematic Customer Resolution Unraveling Meeting
SOAP	Simple Object Access Protocol
UML	Unified Modeling Language
UP	Unified Process
XML	Extensible Markup Language

Въведение

Бурното развитие на информационните технологии и ефекта им в обществото води до приложението им в различни сектори и функции на бизнеса. Тяхната роля се премества от поддържащ към ключов момент в бизнеса. Информационните технологии позволяват приложението на дигитализация на отделни части от бизнеса, което променя процесите в компаниите, начина на опериране и дори типа на бизнеса. Информационните технологии позволяват и иновации чрез преизползване на бизнес модела на съществуващи бизнеси (например споделено ползване на активи). В резултат на масовото навлизане на информационните технологии в бизнеса и човешкия бит се създадоха предпоставки за възникване на явлението технологични стартиращи компании.

В тази връзка се появиха нови понятия като FinTech, InsurTech, EdTech, MedTech/HealthTech, FoodTech, BioTech, AgroTech, PropTech (недвижими имоти), CleanTech/GreenTech (пестене на енергия, рециклиране и др.) и др., с които да се направи опит да се класифицират компаниите, работещи в определени направления и индустрии. Общото при всички подобни компании е, че се налага разработката на нов софтуерен продукт. В някои случаи този софтуер тясно се интегрира с определен вид хардуер – например, устройство от тип IoT, шлем за виртуална реалност, дрон или автономно летателно средство, автомобил или електромобил, медицинска апаратура, военна техника, и прочие.

В други случаи се преосмислят базови човешки дейности с помощта на технологиите, при което софтуерът автоматично събира и обработва данни, анализира, съветва, прогнозира, визуализира, предлага, комуникира, сключва договори или извършва отчетни операции без необходимостта от специален хардуер, а с помощта на широкоразпространени персонални компютри или мобилни устройства.

Тези компании представляват нова организационна форма, характерна със започване и развитие на нов бизнес, съчетано с висока

динамика на процесите в тази насока. Стартиращите компании се описват в литературата най-вече като нововъзникнала малка компания, която създава нов продукт за посрещане на пазарна потребност или приложение на иновативна технология (Ripsas & Troger, 2015). Една част от технологичните стартиращи компании фокусират своята дейност върху производството на софтуер, при което можем да отбележим, че са налице **два ключови компонента – от една страна това е софтуера, като очакван краен продукт, и от друга – иновационния процес, като обект на управление**.

В последните години **софтуерът** от сложен и високотехнологичен продукт, предназначен за високоплатежни клиенти, **се превръща в масов бизнес и потребителски продукт**, към който обаче се поставят повишаващи се изисквания. За да оцелеят в сложната конкурентна среда, технологичните стартиращи компании е необходимо да съобразят производството на софтуер с тези повишаващи се изисквания. Съвременните изисквания към софтуера включват както предоставяне на желана функционалност, така и покриване на редица критерии за качество и време за пазарна реализация, тъй като те са от ключово значение за дейността на компанията.

Добре известен факт е растежът в софтуерната индустрия. Могат да се посочат редица успешни технологични стартиращи компании, чиито основен предмет на дейност не е разработка на софтуер, но именно чрез специално разработен и използван софтуер се реализира конкретната услуга. Например Юбер (Uber) извърши пробив в начина за осъществяване на пътнически транспорт, като предоставя услугата си чрез софтуер, позволяващ свързване на крайния клиент с превозвача и управление на всички останали детайли. Без да подценяваме важността на бизнес идеята (по своята същност комбинация от споделено пътуване и гъвкава заетост), резултатите са показателни – основана през 2009 г, приходите за 2020 г. са над 11 млрд. долара (Iqbal, 2021).

Подобни примери стават еталон за модел и на тази база се появяват

и успешно функционират редица други компании, които прилагат споделено използване или наемане на ресурси[1]. Друг пример за ролята на софтуера е в намаление на разходите и съответно намаление на цената на услугата – например цената на космически полети чрез софтуер, който използва сателитна и инерционна навигационна система, което дава възможност първата степен на космическа ракета да се преизползва многократно (Фалкон 9 на СпейсЕкс, 2015 г.).

Последната криза с Covid-19 показа ясно, че дигитализацията на различни дейности може да подпомогне бързото решаване на обществен проблем чрез прилагане на виртуална комуникация, работа от вкъщи, онлайн здравни консултации и т.н., и да ускори тази трансформация чрез технологиите (Галоуей, 2021). По някои оценки пазарният ръст на бизнес софтуер и услуги за периода 2021-2028 г. се очаква да е около 11% на годишна база, а в някои сегменти около 50% (GrandViewResearch, 2021).

Вторият ключов компонент е **иновационният процес**, който е част от бизнес развитието на технологичните стартиращи компании. Особено актуален негов вариант е дигитализацията. Дигитализацията на бизнеса усъвършенства и променя процесите в компаниите, като допринася за постигане на по-ниски разходи, по-висока скорост, по-добра управляемост и контрол. В някои случаи се достига до промяна на бизнес модела (disruptive business model) или типа на бизнеса (Christensen et al., 2018; Zanni, 2019). Дигитализацията е приложима и в бита, като го прави по-удобен и спестява разходи. Тоест, приложението на информационни технологии създава предпоставки за множество иновации и развитие на предприемачеството. Поради това, голяма част от технологичните стартиращи компании имат за предмет на дейност създаването на продукти, услуги или решения, пряко или косвено свързан с разработката на софтуер, и съответно най-търсени са специалистите, заети с това.

По определение малките и средни предприятия (МСП) се считат за

[1] Имаме предвид т.нар. „икономика на споделянето”.

гръбнак на икономиката и потенциал за растеж, важен източник на заетост и генератор на нови работни места. Стартиращите компании са тези МСП, които се фокусират над иновация и бърз растеж. Европейската комисия приема редица инициативи за подобряване на икономическите и регулаторни рамки за стартиращи компании като потенциал за икономически растеж и създаване на нови работни места (Bormans et al., 2020). Процесът по създаване и развитие на технологични стартиращи компании е свързан с висока степен на несигурност и риск. Рискът за стартиращи компании често се измерва с ниво на оцеляване (или провал) след петата година от основаването. По някои оценки този показател е под 20% (Ejermo & Xiao, 2014; Santisteban at al., 2021). Това мотивира да се извършат множество изследвания през последните години за възможните фактори за успех при стартиращите компании и рисковете за провал (Bormans et al., 2020).

Гореизложеното по отношение на важната роля на технологичните стартиращи компании за икономиката и същевременно техния нисък процент на оцеляване (Santisteban & Mauricio, 2017; U.S. Bureau of Labor Statistics, 2021), съчетано със сложността и риска при разработката на софтуер, показва **актуалността и важността** на темата за управление на иновационния процес при разработка на софтуер в стартиращи технологични компании. Необходимо е провеждането на изследвания, които да подпомогнат дейностите по управление на разработката, поддръжката и съпровождането на софтуер от този вид компании. В тази връзка, за целите на дисертационния труд, използваме следните работни дефиниции:

1. Под понятието „технологична стартираща компания” (от англ. „technology startup company”) се разбира микро или малко предприятие, водено от предприемач или екип от предприемачи, което има сравнително кратка история (до 5 г.), с цел ефективно създаване на нов софтуерен продукт, да се разработи бизнес модел, да се докаже жизненост (валидиране в пазарна ниша) и да се постигне растеж.

2. Под „продукт” се разбира общо понятие в различни форми – краен продукт, отделен компонент, услуга, решение, стойност, услуга с добавена стойност и т.н. Той се явява основен компонент в конкретно прилаган бизнес модел. В настоящото изследване продуктът, който разглеждаме, е софтуер. Допълнителни дефиниции за софтуер, продукт, услуга, решение могат да се намерят в литературата. Правим уточнението, че в изложението разглеждаме технологични стартиращи компании, които създават продукт, който е ключово свързан с разработка на софтуер.

Характерно за технологичните стартиращи компании е, че те работят в условия на сравнително малко ресурси, ограничен времеви прозорец, ограничени предприемачески и маркетингови възможности, висока степен на несигурност и риск от провал. В началото не са ясни всички детайли по отношение на продукт, пазар и организация. Допълнително дейността е ограничена в т.нар. „предприемачески прозорец на възможности” (Коев, 2016), който се дефинира от интензивното развитие на технологии и иновации. Поради това, компанията трябва да е проактивна, адаптивна към средата, бързо учеща се. Друга съществена особеност е, че са налице поне два „резултата” – валидиран бизнес модел и софтуерен продукт.

Производството на качествен продукт може да се постигне с приложение на подходящи технологии, организация на процесите и тяхното ефективно управление. Последното предполага използването на софтуерна система, подпомагаща процесите по управление. Ето защо, основна **изследователска теза** на дисертацията е, че стартиращите технологични компании имат специфична цел, процеси и организация, и затова предприемачите се нуждаят от специализиран подход за управление, в комбинация с софтуерна система, с помощта на които да постигнат целта си. Разбира се, следва да се отчита, че това е само едно от необходимите условия в развитието на рисковани бизнес начинания.

В тази връзка научно-изследователската **цел на дисертационния труд** е да се разработи проект на софтуерна система, която прилагана с

подходящ подход за управление на процесите, да подпомага успешно дейността на технологичните стартиращи компании. С оглед поставената цел, **основните задачи** за решаване са:

1. Изследване на същността и особеностите на технологичните стартиращи компании, както и някои проблеми, свързани с разработката на софтуерни продукти.

2. Изследване на организационните структури и подходите за управление на софтуерни проекти в технологичните стартиращи компании

3. Разработване на проект на софтуерна система, подпомагаща процесите по управление на софтуерни проекти в технологичните стартиращи компании.

4. Представяне на план за разработка и внедряване на софтуерна система.

5. Апробиране на резултатите от изследването в конкретен обект на приложение.

Обект на дисертационното изследване са технологични стартиращи компании с дейност разработка, поддръжка и съпровождане на софтуерни продукти. Те може да са изцяло собствени търговски продукти в определена форма и реализация (продукт, услуга, компонент и други) или по поръчка от страна на външен възложител.

Предмет на изследването е разработка на нови софтуерни продукти при предприемачески процес с неясни предварително и динамично променящи се във времето изисквания и параметри. Поради това насочваме изследването към подходящи подходи за управление на процеса по разработка на софтуер и проектиране на софтуерна система, която може да подпомага технологичните стартиращи компании в тази дейност.

За **методологична основа на изследването** в дисертацията са ползвани различни научноизследователски методи, от тях с най-голямо значение са: проучване и събиране на данни, сравнение, анализ и синтез, систематизиране (класифициране и типизиране), индукция и дедукция,

моделиране и научна абстракция.

За описание на последователността на процесите е използван системен подход към изследвания обект и предмет. Онагледяването на различни факти и данни се осъществява с помощта на графичния, схематичния и фигуралния подход. Комбинираното прилагане на изследователски методи и подходи е с цел достигането на крайни констативни изводи и представянето на препоръки.

Като се има предвид, от една страна, факта, че така поставената тема е всеобхватна и за нейното изследване може да се приложи и друг изследователски подход, а от друга страна, съобразявайки се с поставените цел и задачи, се налага да се дефинират следните **ограничения**:

1. Извън обхвата на изследването са въпроси, свързани с правните и регулаторни рамки, на които трябва да се подчинява дейността на технологичната стартираща компания като стопански субект (търговец по смисъла на Търговския закон).

2. Не са обхванати практико-приложните проблеми на технологичната стартираща компания от общ характер за всяка една компания – връзки с бизнес партньори, институции, управление на собственост, капитал, човешки ресурси и прочие. В дисертационния труд фокусът е единствено върху проблемите, свързани с управлението на софтуерни проекти.

3. В по-широк смисъл, целевата дейност на една технологична стартираща компания може и да не е свързана с производство на софтуер. Съответно тези компании не са обект на изследването.

Основните резултати от дисертационния труд са апробирани на научни конференции и публикувани в сборници с доклади (5 бр.) и научни списания (3 бр.).

Глава 1. Теоретични основи на софтуерните технологични стартиращи компании

1.1. Същност и особености на технологичните стартиращи компании

Известно е, че ключова роля при създаването и функционирането на технологични стартиращи компании има т.нар. предприемач. Неговата основна дейност е да организира различни по своята същност и структура работни процеси за постигането на определена цел. Организирането, като основен дял в теория на управлението, разглежда въпроси, свързани с начините, чрез които една организация може да постигне своите цел и задачи.

Всяка предприемаческа дейност работи при специфични условия, което включва ограничения в ресурси, време, предприемачески и пазарни възможности, висока степен на несигурност и риск, конкуренция от други компании, интензивно развитие на технологии и иновации. Затова е необходимо управлението да е на високо ниво, включително и по отношение на подходяща организираност на екип, задачи, ресурси, и взаимодействие с външната среда, за да може да се оползотворят открилите се възможности (Scarborough, 2013).

Обект на дисертационното изследване са технологични стартиращи компании[2] с дейност разработка, поддръжка и съпровождане на софтуерни продукти. В тази връзка е необходимо да се очертае обхвата на понятието

[2] В изложението понятието "компания" е използвано като еквивалент на понятието "предприятие", използвано в "Закон за малките и средните предприятия", "Закон за насърчаване на инвестициите", "Закон за защита на конкуренцията", "Закон за счетоводството", "Закон за корпоративното подоходно облагане", Регламент (ЕС) № 651/2014 и други нормативни актове. Също така използваме понятието "компания" и като еквивалент на понятието "търговец", използвано в "Търговския закон", чл.1, ал.1: "Търговец по смисъла на този закон е всяко физическо или юридическо лице, което...". Пак според същия закон, чл.7, ал.1: "Фирма е наименованието, под което търговецът упражнява занятието си и се подписва."

„стартираща компания” в контекста на настоящото изследване. Аналогични понятия могат да се срещнат под наименованията технологичен стартъп (technology startup), стартиращо предприятие, софтуерна компания. За да бъдат те ясно обособени, посочваме **критерии**, по които те се отличават от всички останали компании (предприятия).

Правната база в Европейския съюз и България използва термина предприятие (enterprise) и разделя предприятията в четири категории – микро, малки, средни и големи. Разделянето е на базата на броя наети хора и стойността на активите или годишният оборот от продажби. Използвайки тази класификация, се фокусираме основно над микро и малки предприятия – до 50 наети и до 10 милиона евро активи или оборот (European Commission, 2019).

Възможни методи за определяне са измерване **годишните приходи или ниво/етап на инвестиция**, както и периодът от време „преди да стане голяма компания” (Blank, 2013). Тях можем да смятаме за аналогични и да използваме дадения по-горе критерий за микро и малки предприятия. Допълнително, за да се обособят стартиращите технологични компании, е необходимо да се постави ограничение за тяхната възраст. Дефинициите в литературата са различни – до 3 години, до 5 или 10 години, „имат малък опит или никаква оперативна история” (Blank, 2013). Краткият цикъл на иновации и технологично развитие, особено в ИТ, налагат да изберем за критерий възраст до 5 години.

Друг критерий за дефиниране на стартиращи компании може да е **ключовата цел в бизнеса**. В литературата има различни дефиниции, като тук представяме някои от по-популярните:

– Ефективно разработва и валидира мащабируем бизнес модел (Ries, 2011);

– Достига до повторяем, мащабируем, печеливш бизнес модел (Blank & Dorf, 2012);

– Прилага жизнеспособен бизнес модел за посрещане на пазарна

потребност или проблем (Blank, 2013);

– Разработва нов продукт, прилага нов бизнес модел, доказва жизненост и потенциал за растеж (Robehmed, 2013);

– Достига задоволително съвпадение продукт-пазар (product-market fit), като набира клиентска база, където фокусът е върху търсене на клиенти, адаптиране на продукт и бизнес модел (Alvarez, 2017);

– Млада компания с иновативен бизнес модел и/или иновативни технологии, демонстрираща значим растеж в брой заети и оборот (Ripsas & Troger, 2015);

– Движенията, които изследват пазара за възможности, а не движенията, които използват съществуващите позиции, са преобладаващи (Katila et al., 2012).

От така дадените дефиниции можем да отделим някои несъществени различия между авторите. Например, някои от тях споменават продукт, а при други това се подразбира. Една част споменават и конкретна причина за продукта – да се достигне задоволително съвпадение продукт-пазар (посрещане на пазарна потребност) и адаптиране на продукт и бизнес модел, който приляга (пасва) на пазарните условия (така нареченият в английската литература Customer Development Model) (Blank, 2013; Alvarez, 2017).

Вариант на този подход е търсенето на продукт с приложение на нова технология, която да намира пазарна реализация. Гореспоменатите автори се обединяват около възгледа, че следва да се разработи бизнес модел и да има „съвпадение" на бизнес модел и пазар, което се нарича още валидиране. Тоест, има процес на „изследване на пазара" и намиране на бизнес модел, който е валиден за този пазар. Това позволява на компанията да съществува – жизненост.

В определенията не се посочва дали пазарите са нови или съществуващи. Но отличителното е, че тяхната цел съдържа това да „изследват пазара за възможности, а не движенията, които използват

съществуващите позиции, да са преобладаващи" (Katila et al., 2012). Вторите „движения" са характерни за установените компании с ясен продукт, пазар, модел и се фокусират да оптимизират процеси и структури за установените продукти и пазари (срещан като Product Development Model). Поради това смятаме, че този критерий – търсене на жизнеспособен валидиран бизнес модел и модела на поведение, е един от най-важните, за да се отличи коя компания е стартираща.

Важен фактор, който следва да се разгледа и се описва от много автори (Ries, 2011; Graham, 2012; Blank, 2013; Heitman, 2014; Ripsas & Troger, 2015) по темата за стартиращи компании, е т.нар. **цел за растеж** (компанията се стреми да стане по-голяма). В по-точно описание, можем да цитираме следното от дефинициите „мащабируем бизнес модел", „растеж", „растеж на брой заети и оборот". Бланк описва няколко вида новосъздадени компании и описва които могат да се определят като стартиращи (startup). В тях не се включват компаниите, които планират да останат малки и задоволяват само личните потребности на основателите, като например работещи на свободна практика, семеен бизнес или обслужване на други бизнеси. Затова приемаме като подходяща за случая дефиницията „търси растеж и мащабируем модел" (Blank, 2013). Под мащабируем бизнес модел следва да се разбира модел, който позволява повторяемост на модела, ефективно увеличава размера на бизнеса (бързо и с икономии от мащаба) и осигурява растеж на компанията.

В този смисъл, подкрепящо е определението за стартираща компания на Пол Греъм основател на „Y combinator" – „Стартиращата компания е проектирана да расте бързо. Единственото съществено нещо е растежът. Всичко друго, което асоцииираме със стартиращата компания, произлиза от растежа" (Graham, 2012). Казано по друг начин, изначално целта включва планиран растеж и бизнес модел.

Друг критерий, по който можем да отличим стартиращи компании, е **ориентираността към иновация, нов продукт, нов бизнес модел**. В

литературата по мениджмънт може да се намерят описани много форми на предприемаческа активност. Изключваме тези, които заимстват дейност или продукт на друга компания, използват или дублират готов продукт, бизнес модел, технология и търговска марка с високо ниво на покритие, като например представителство, франчайз, бизнес придобиване, копиране на конкурент и други. В настоящия труд се фокусираме над компании със свой продукт и бизнес модел, както и компании, които използват познати компоненти, бизнес модел, процеси, но има степен на уникална комбинация от тях.

Следващ критерий за отличаване, който може да се посочи, е **видът на продукта – софтуер или продукт, свързан с разработване на софтуер**. Голяма част от технологичните стартиращи компании имат предмет на дейност продукти, услуги или решения, свързани с разработката на софтуер (Bormans, 2020). За целите на дисертационния труд, използваме класическото определение за понятието „софтуер" на американската асоциация IEEE. Тя гласи, че софтуерът представлява компютърни програми, процедури, правила и евентуално придружаваща документация, както и данни, отнасящи се до функционирането на компютърна система (IEEE, 1983). Еквивалентно наименование за софтуер е програмен продукт. Той има специфични свойства, като разнообразие от форми, функции, абстрактност, уникалност, състав, качество, надеждност, производителност и т.н.

За поставената тема по-интересни са тези характеристики относно процеса на създаването, като уникалност, ресурсоемкост, висок риск и качество, мултидисциплинарност, актуалност и приложимост, поддръжка. Под **уникалност** е прието да са разбира, че създаването на софтуер изисква значими първоначални усилия преди появата на първи работещ екземпляр, разпространение на дефекти (наричани още бъгове) във всички копия и подмяната на по-стари с по-нови версии. Характеристиката **ресурсоемкост** е налична поради влагане на много усилия, преди да се

излезе готово за продажба първото копие. **Мултидисциплинарност** съществува поради необходимостта от различни специалисти. Под висок риск се разбира сложност на производството, пропуски в изисквания и качество на продукт, неспазване на срок за доставка и други.

Под **качество** следва да са разбира огромният комплекс от изисквания към продукта, за да бъде актуален на тенденции, стандарти, практики и потребителски изисквания. Под **актуалност и поддръжка** имаме предвид, че всеки софтуер съществува в дадена среда и с оглед на интензивното развитие на иновациите и софтуера, следва да поддържа актуално по-нови версии за намерени бъгове, уязвимости, нови тенденции и актуализирани версии в хардуера и софтуера, от които зависи.

Като последен критерий приемаме **същността на създателите на стартиращи компании – един или повече предприемачи**. За краткост ползваме термина „предприемач" и за едно лице и за екип предприемачи. В някои източници се ползва и понятието „основатели". Ролята на предприемача е ключова при създаването и функционирането на технологични стартиращи компании. Има редица изследвания (Manev et al., 2012; Prohorovs et al., 2019; Bormans, 2020; Santisteban et al., 2021; Marconatto et al., 2021), които доказват, че основният фактор за успех и растеж на микро, малки и средни предприятия е свързан с предприемача и неговите: лична цел, мотивация, ориентация към пазара, способност за растеж и професионално управление, контактите и социалните мрежи. Това доказва допълнително важността и актуалността на настоящата теза, и необходимостта от знание и инструментариум, като специфична софтуерна система, която да подпомага управлението на стартиращата компания. В някои случаи предприемачите са мениджъри от голяма компания, на които се възлага развитието на изцяло ново поделение за работа в дадената насока и се използват сходни подходи за развитие.

За целите на дисертационния труд, на база така дефинираните критерии, можем да използваме следната работна дефиниция за понятието

„технологична стартираща компания” – новосъздадено технологично микро или малко предприятие[3], водено от предприемач[4] или екип предприемачи, което има кратка история (до 5 г.), с основна цел да се създаде нов софтуерен продукт, да разработи ефективно и валидира мащабируем бизнес модел за него, така че да докаже жизненост и осигури растеж. Разработката на продукта е свързана, пряко или непряко, с разработване на софтуер. Начинът за постигане е чрез иновация в продукт, технология, бизнес модел или организация, и откриване на пазарни възможности.

Организационната правна форма на технологичната стартираща компания с предприемач и екип, които разработват нов софтуерен продукт и бизнес модел, могат да варират според обстоятелствата. Обичайно, това е самостоятелен нов търговец (по смисъла на „Търговския закон”) или ново бизнес подразделение в съществуваща голяма компания, което има същата цел. В контекста на условията в България, при преобладаващи микро и малки софтуерни предприятия, за съществуващите такива е трудно да започнат нов продукт с екип повече от микро предприятие. Обичайно практиката показва, че стартиращите компании са микро или малки предприятия, докато те не докажат значим растеж, при който и самата организация се разраства.

В литературата съществува аналогичен термин от английски език наречен „startup” или „start-up”. Приемаме, че стартираща компания и „startup” са сходни, като вече уточнихме, че критериите за иновация, жизненост и растеж се имат предвид за стартиращи компании. Друга особеност е, че често „startup” компаниите са проектирани да са обект на външно финансиране и придобиване с цел ускоряване на растежа или продажба с цел печалба. Тази тема е извън обхвата на настоящата теза.

[3] Имаме предвид микро или малко предприятие по смисъла на закона за МСП и дефиницията на Европейската комисия за предприятие с до 50 наети и оборот/активи до 10 милиона евро.

[4] Под „предприемач” разбираме един или повече предприемачи, основаващи своя компания.

Когато споменаваме продукт, за целите на настоящата дисертация ние имаме предвид обобщено понятие като резултат от дейността на компанията. То може да приема различни форми. В литературата за иновации, стартиращи компании, бизнес моделиране има различни термини. Такива са продукт (Ries, 2011), решение (Maurya, 2012), услуга, стойност или стойност за потребителя (Blank, 2013; Morgan & Licker, 2020). За да се избегне това разминаване, използваме обобщеното понятие „продукт" като резултат от иновационния процес за стартираща софтуерна компания.

Подобна дефиниция е „всичко, което клиентите изпитват от тяхното взаимодействие с компанията може да се счита за част от продукта на компанията" (Ries, 2011). Това определение е адекватно и в контекста на модерните тенденции в софтуерните продукти да има комплекс от компоненти или фамилия продукти, които реализират представата на клиента като продукт и са силно взаимосвързани. Трябва да отбележим, че продуктът може да е вследствие на решение на проблем (започващи от пазарна потребност, проблем на клиент) или търсене на приложение за изобретение. Поради тази причина, считаме за уместно при дефиницията за продукт да не се спираме на конкретната му форма – компонент, услуга, решение, стойност, услуга с добавена стойност и т.н.

Продуктът също може да се дефинира като технологична иновация, базирана на иновация с информационни технологии. Терминът „технология" може да дефинираме като „знания, умения и артефакти, които могат да се използват за разработка на продукти и услуги, както и за системите за производството и доставката им" (Burgelman et al., 2008). За термина „иновация" считаме за удачна дефиниция, която е приета в изследване за софтуерната индустрия (Edison et al., 2013) – „производство или приемане, усвояване и използване на новост с добавена стойност в икономическата и социалната сфера; обновяване и разширяване на продукти, услуги и пазари; разработка на нови методи на производство;

създаване на нови системи за управление; тя е едновременно процес и резултат“[5] (Crossan & Apaydin, 2010). От тези дефиниции може да се изведе и съвкупният термин **„технологична иновация“**. Технологията сама по себе си не е от съществено знание. Важно уточнение е, че **технологията се превръща в иновация чрез успешната пазарна реализация във форма на продукт**. По отношение на изходната точка, от която се търси продукт, има два основни подхода. Първият е основан на създаване на нов продукт за клиентски проблем – пазарна потребност (Blank, 2013; Alvarez, 2017). Той се решава с определени технологии. Вторият е създаване на продукт, който цели приложение на технология[6] и впоследствие се търсят пазарни потребности.

За да бъдат успешни технологичните стартиращи компании, следва да се имат предвид условията на работа, ресурсите, с които се разполага, и целта в бизнеса. Редица компоненти като продукт, пазар, бизнес модел не са установени. Те също може да целят нови пазари, за които са характерни слаба структурираност, непредсказуемост и ограничено разбиране за клиенти, конкуренти и сегменти. Поради това „движенията, които изследват пазара за възможности, … са особено важни” (Katila et al., 2012). От гледна точка на предприемачеството, най-характерни за тях са ограничения в следните области (Adler, 2011; Ries, 2011; Maurya, 2012; Blank, 2013; Croll & Yoskovitz, 2013; Rauch & Hulsink, 2015; Santisteban & Mauricio, 2017):

---

[5] Друга популярна дефиниция на понятието е приетата от OECD и приложена в "Иновационна стратегия за интелигентна специализация 2014-2020 г.", разработена от Министерството на икономиката и одобрена с Решение на МС №384 от 13.07.2017 г.: „Иновация е въвеждане в употреба на някакъв нов или значително подобрен продукт (стока или услуга) или производствен процес, на нов метод за маркетинг или на нов организационен метод в търговската практика, организацията на работните места или външните връзки, които създават пазарни предимства и при това повишават конкурентоспособността на фирмите” (по Manuel d’Oslo 3e édition OECD/European Communities 2005).

[6] Важно е да се открои изходната точка, защото подходът и методите в двата случая са различни. От обхвата на изследване са изключени дейности като износ на ресурси, на дейност (outsourcing) и други, които по същество не са създаване на собствен продукт или услуга.

– Ограничени ресурси;

– Прозорец на време и възможности;

– Новост и неяснота;

– Несигурност и висок риск от провал;

– Силна конкуренция.

Ограничените ресурси са обичайно тези, с които предприемачите започват дейността. Тук се включват не само финансови и материални ресурси, а и лични цели и мотивация, интелектуален капитал (човешки ресурси, знания, компетенции, опит в технологичен, пазарен и организационен аспект), сработване с екипа, умения за управление на процеси и бизнес организация. Известно е, че често стартиращите компании са слаби, уязвими и търсят ресурси за осъществяване на идеите на предприемачите. Редица изследвания показват, че от изключителна важност за налагане на пазара са: опит в индустрията, опит със стартираща компания и компетентност в областта (Colombo & Grilli, 2010; Preisendorfer et al., 2012; Prohorovs et al., 2019; Santisteban et al., 2021). Съществуват агенти и механизми, чрез които предприемачите могат да набавят липсващи им ресурси (обучение, привличане на хора с ресурси и опит, партньорство, трансфер на ноу-хау, финансиране на етапи и други).

Предприемачеството в дадено направление е възможно в определен времеви интервал – т.нар. „предприемачески прозорец на възможности" (Коев, 2016), след което конкуренцията на съответната новопоявила се пазарна ниша се засилва значително. При интензивното развитие на технологии, иновации и глобализация, този времеви интервал става все по-кратък. Новостта и неяснотата, като особености, следват от липса на предварителна и детайлна яснота за продукта, пазара, бизнес модела и организацията, която да се създаде. Има случаи, в които тези елементи са ясни като предварителна рамка, но в процеса на работа и пазарна валидация се оказват невалидни или неефективни. От това следва и постановката за условия на несигурност и висок риск от провал.

Доказателство за високия риск е и високото ниво на провал (U.S. Bureau of Labor Statistics, 2021). Рискът за стартиращи компании често се измерва с ниво на оцеляване след петата година от основаването. По някои оценки този показател е под 20% оцеляване в световен план (Preisendorfer et al., 2012; Ejermo & Xiao, 2014; Santisteban at al., 2021), а по други оценки под 5-10% за период от около 20 г.

Това мотивира извършването на множество изследвания за причините за провал, възможностите да се смекчат рисковете при стартиращите компании, както и систематизиране на възможните фактори за успех по етапи на развитие (Santisteban & Mauricio, 2017). Без да навлизаме в детайли, можем да споменем три възможни фактора, подходящи за изследването – умения и опит в областта, умения за управление на бизнес и пазарна валидация.

Относно силната конкуренция може да се отбележи, че много технологични стартиращи компании се оказват пряко или в последствие обект на силна конкуренция от големи компании, които имат нужните ресурси, организация, пазарно влияние. Често това е придружено и от агресивно поведение, като ценови дъмпинг, изкупуване на самата компания или на нейни конкуренти, копиране на продукта или негови характеристики, и други.

Специфичните условия, от гледна точка на индустрията „разработка на софтуер”, в синтезиран вид са бързо и динамично развитие на технологиите и пазара, сложност на продукта, многообразие на компоненти и доставчици, сложност на изискванията за качество, сериозна инвестиция преди първата продажба, неяснота във всички изисквания в началото, сложност на процеса по разработка, необходимост от регулярна поддръжка и обновление на продукта, важност на човешките ресурси и висока цена, мултидисциплинарност, интензивна работа със знание, управление на създаването и съхраняване на знанието (Paternoster et al., 2014; Giardino et al., 2015).

Причините за успех или провал на стартиращите компании са често дискутирана тема в литературата. Често проблемът и неговото решение (продукт) се оценява като пазар и валидира емпирично преди изработването му. Това е ключово и редица източници по предприемачество и стартиращи компании го посочват (Ries, 2011; Maurya, 2012; Blank, 2013; Croll, 2013; Schmitt et al., 2018; Hoff, 2019; Leatherbee & Katila, 2020).

Самата необходимост от пазарна валидация налага да се вземат решения, които са ключови в изработването на решение на проблем, което е основа за продукт на компанията – да се открие проблемът на потребителя (още наричан и пазарна потребност), дори и неявен, и да се избегнат възможни грешки и коригират отклонения при търсенето на технологични решения на проблема. Пазарната потребност и нейното откриване е известно в англоезичната литература като метод Customer Development (Alvarez, 2017; Kittlaus & Fricker, 2017; Silva et al., 2020). Съобразяването с нея, откриването и дефинирането на клиентски профил и пазар, налагат гъвкав дизайн[7] и ефективно откриване на знания, за да се съобрази предприемачът с липсата на явност в потребностите и изискванията, да се адаптира към несигурността, динамиката и ограниченията във времето.

Като особеност на разработката на софтуер в технологична стартираща компания трябва да отбележим и спецификата на самата дейност. По принцип, две важни концепции в успешната разработка на софтуер са участието на потребителя (user involvement) и описание и стабилност на изискванията. За стартиращата технологична компания тези концепции не са валидни или не са в този си вид.

В началото, потребителят или клиентът може да не е ясен,

[7] Гъвкавият дизайн дава възможност за лесна промяна и адаптация на продукт или услуга към различни предпочитания.

изискванията да не са явни, да не са известни или да се променят по време на търсенето на пазарна ниша. Обратната връзка от потребителя/възложителя не съществува или не е винаги директна. Тя обичайно не е чрез директно задаване на въпроси и получаване на отговори. Възможно е тя да се получи с наблюдение, изследване, анализ на данни, експерименти за валидиране на хипотетично изискване и други методи. Допълнително, всяко изискване следва да има и анализ на маркетинговия и финансов ефект, тъй като неговото включване може евентуално или да привлече, или да не привлече клиенти, но със сигурност изисква ангажирането на ресурси за самото му разработване и внедряване.

Важен аспект, който според нас е съществен по темата, е за предприемача (или екип предприемачи съдружници) в стартираща компания. Различни автори споделят необходимостта от определени психологически атрибути, социокултурна среда, предприемаческа нагласа, мотивация, владеене на начини, умения и инструменти за предприемаческо откриване, организиране и лидерство, както и съответно обучение (индивидуално или групово) (Preisendorfer et al., 2012; Prohorovs et al., 2019; Santisteban et al., 2021). За целта следва преди обичайният процес или заедно с него, един предприемач да е преминал стъпки на валидиране и обучение за това да е подходящ и подготвен за предприемачество. Следва да има и обучение за умения за работа в екип съдружници с цел работа успешно в продължителен период и с нужни ключови компетенции. Например, за софтуерна компания е подходящо екипът да има и поне минимални компетенции по технологии, маркетинг и бизнес развитие.

Както е видно от дефиницията за стартираща технологична компания, тя трябва да развие и валидира ефективен бизнес модел. Пазарната реализация на един продукт се осъществява чрез специфичен бизнес модел. Разработката на бизнес модел е част от иновационния процес. Както отбелязват някои изследователи – „забелязваме за почти всички успешни основатели … тяхната способност за работа на много

конкретно и много абстрактно ниво … те се опитват да открият най-добрия бизнес модел" (Croll & Yoskovitz, 2013).

Състоянието на предприемачеството в България в ИТ сектора, в сравнение с другите европейски страни, се развива с темпове, отговарящи на бизнес средата[8]. Най-много стартиращи компании има в гр.София. Към средата на 2022 г. сред успешните можем да споменем: Opencode Systems (https://opencode.com/) – доставчик на телекомуникационни решения за мобилна мрежа, Microweber (https://microweber.com/) – CMS платформа с отворен код за създаване на уебсайтове чрез технологията „drag and drop", ScaleFocus (https://scalefocus.com/) – ИТ поддръжка, SessionStack (https://sessionstack.com/) – интерактивно съвместно сърфиране и инструменти за анализ за екипи за поддръжка, Flyver (https://flyver.co/) – софтуер за управление на дронове, Connect Machines To Web (https://cm2w.net/) – дистанционно наблюдение и контрол на хардуерно оборудване, Ontotext (https://www.ontotext.com/) – обработка на неструктурирани данни, Tool Domains (https://edoms.com/) – управление на домейни, Software Group (https://www.softwaregroup.com/) – дигитално банкиране и др.

Извършените проучвания по дисертационния труд разкриват до голяма степен същността на технологичните стартиращи компании от правна и научна гледна точка. За да се изпълни целта на изследването, е необходимо да бъдат идентифицирани основните проблеми, свързани с разработката на софтуерни продукти, с които може да се сблъскат технологичните стартиращи компании.

[8] Налични са голям брой специализирани сайтове за новини за стартиращи компании в Европейския съюз, САЩ и по света. Например: https://www.eu-startups.com/, https://www.gemconsortium.org/, https://www.seedtable.com/, https://www.finsmes.com/, https://www.siliconrepublic.com/, https://www.crunchbase.com/, https://sifted.eu/, https://siliconcanals.com/, https://tech.eu/ и др.

1.2. Методични проблеми, свързани с разработката на софтуерни продукти

За понятието „софтуерни продукти" (software product) приемаме определението на IEEE – съвкупност от компютърни програми, процедури, правила и евентуално придружаваща документация, както и данни отнасящи се до функционирането на една компютърна система (IEEE, 1990). Дейността на технологичната стартираща компания включва спецификация, разработка, внедряване, продажба, поддръжка на софтуер, както и услуги за него, но акцентът е на производството и поддръжката. Дейността може да се класифицира като[9]:

– производство и продажба на собствени търговски продукти;

– собствени компоненти за влагане в продукти на други разработчици;

– производство по възлагане от клиенти (външни възложители);

– индивидуални услуги[10] за настройка, адаптация и внедряване на продукт;

– поддръжка на съществуващи и внедрени продукти от горните категории.

Всеки вариант има специфика по отношение на възложител, задание, финансиране, резултат, жизнен цикъл, повторяемост, и прочие, а дейността на технологичната стартираща компания може да бъде смесица от различни варианти с различни продукти. Всеки един може да се реализира в един или серия проекти. Проектите, както е известно, имат за основни параметри обхват, качество, цена и време, които се определят в

[9] Вариантите се използват под различни подварианти в софтуерната индустрия и някои автори ги делят на продукти и услуги. Има тенденции за преминаване в бизнес с продукти, предоставящи услуги (Ескенази и Манева, 2006).

[10] Има се предвид индивидуални услуги над/със съществуващ продукт. Други автори включват и други услуги (напр. продукт на външен клиент), но тук визираме услуги над съществуващ работещ продукт.

договор за разработка с външен клиент или задание за разработка на собствен продукт. Обща характеристика на вариантите е дадена в табл.1.1.

Таблица 1.1.
Видове дейности в технологична стартираща компания (разработка на автора)

Вид дейност	Възложител[11]	Заданието се определя от	Резултат	Повторяемост и обновяване	Специфична дейност по продукта
Собствени продукти	Р-л екип / отд. „Маркетинг”	Р-л екип / отд. „Маркетинг”	Продукт готов за продажба	Постоянна поддръжка и обн.	Всички
Собствени компоненти	Р-л екип / отд. „Маркетинг”	Р-л екип / отд. „Маркетинг”	Компонент за продажба	Постоянна поддръжка и обн.	Всички
Клиентски проект	Клиент	Клиента (или с проектант)	Софтуер по задание	Няма или с нов договор	Спецификация и поддръжка са опции
Индивидуални услуги	Клиент	Клиента (или с проектант)	Внедрен продукт	Няма	Спецификация, документация, внедряване
Поддръжка	Клиент / отд. „Продажби”	Клиент / отд. „Маркетинг”	Услуга по спецификация	Срочно / при необходимост	Документация, проучване, корекции

Забележка: Възможно е да има обособени отдели „Маркетинг” и „Продажби” с основна задача за навлизане на определени пазарни ниши. В случай, че няма обособени отдели, то ръководителят на екипа и клиентът изпълняват техните функции.

Софтуерният продукт за външен клиент е по възложен проект (от директен контакт, търг, надстройка на друг разработен продукт, експеримент, други). Често подобни проекти се поемат от технологични стартиращи компании с цел да се провери идея, да се съберат изисквания за продукт, да се изучи клиента с цел последваща повторяемост и стандартизация. Параметрите се определят в договор преди същинската работа по проекта. Обхватът и спецификацията се задават от клиента или се изготвят от аналитик в технологичната стартираща компания.

Понякога обхватът на заданието не е дефинирано всестранно и изчерпателно в договора, тъй като при началните етапи негови части не могат да се дефинират изцяло или са описани без детайли. На база личен

[11] Възложителят възлага официално, но детайлите по заданието е възможно да се определят съвместно и със специализираната помощ от друг отдел или служител.

опит можем да твърдим, че нерядко клиентът желае да промени / допълни изисквания, когато види начина, по който действа софтуерът, или дадените изисквания не са адекватни. Нивото на качество се задава индивидуално за проекта (Janes et al., 2017; Desyatirikova et al., 2017; Khosravi & Nilashi, 2018; Jain et al., 2019; Kaur, 2020), като то може да е индивидуално уговорено между компаниите, или да е на базата на национален или международен стандарт. Времето за изпълнение на проекта може да е различно, но според според нас, обичайно е от една седмица до една година. По ограниченията за обхват, цена и време, проектите могат да се разделят например на: фиксирани като обхват, цена и време; фиксирани, но с опцията за промени в изискванията за време и цена; и променливи по обхват, време, цена на база изразходван ресурс.

Независимо от типа, задължение на компанията е да изпълни проекта в договорен обхват, качество, време и бюджет. Въпреки, че има типове с променливи параметри, в зависимост от клиента и случая, всеки тип проект си има краен (възможно е и неизвестен в началото) финансов, времеви и психологически лимит (спрямо договора). Допълнително предимство на компанията може да бъде, ако тя работи освен за изпълнение на проект, но и за степента на удовлетворение или дори „очароване на клиента“ с решение надвишаващо договореното[12]. Това е важно, ако разработваният продукт е ключов за клиента или ако има потенциал както за добра референция, така и за продукт, от който може да последват свързани поръчки за обновление и поддръжка, така и на отправна точка за нова пазарна ниша. Тук е необходима проницателна преценка за свързаните с това реална възможност и рискове.

Собствените софтуерни продукти са инициатива на компанията, която има отговорността за планиране, спецификация, изпълнение и

[12] Целенасоченото надхвърляне на предварителните очаквания на клиента е известно в маркетинга като „очароване на клиента“ (customer delight).

финансиране. Изискванията са от вътрешен възложител (р-л продукт, маркетингови специалисти), но се допълват в различна форма с обратна връзка от потребители. Качество и цена (бюджет) се определят също от компанията според избраното позициониране на пазара и възможности.

Собственият софтуерен продукт предлага по-големи възможности за печалба с мултипликация на продажбите, но е редно да се очаква повече динамика в изисквания, процес на разработка и параметри, повече предизвикателства. Последните включват избор на точна пазарна ниша и момент на пускане, адекватност на бъдещи потребителски потребности, технологии и стандарти за покриване, по-високи стандарти за качество, широк кръг потребители, съвместимост с различни среди, динамика в развитие на технологии, състезание с конкуренти за пускане на продукт, технологично остаряване, силен производител да заеме нишата със своя пакет, кратък период за пазарни продажби и др. В контраст с индивидуалния продукт, собственият продукт изисква по-добра документация, повече поддръжка за краен потребител, както и вътрешна документация за поддръжка. Реализацията в проекти е обичайно в един начален по-голям проект за дефиниране на минимална версия и след това по-кратки за следващи версии.

Собственият софтуерен компонент е аналог на собствения софтуерен продукт. Компонентът е междинен продукт, готова „част" включвана в софтуерни продукти. Обичайно потребителите им са професионалисти от бранша (разработчици, програмисти, дизайнери, проектанти). От това следва, че документацията и поддръжката са ориентирани към професионалисти. Положителният ефект за технологичната стартираща компания е продажба на специализиран пазар и възползване от пазарни ниши на други „производители" на краен продукт. Детайли за компонентите са дадени по-долу.

Индивидуалните услуги са еднократно действие и услуга за определен клиент (външен възложител). Те са извън обсега на настоящето

изследване. В някои случаи, когато услугата е по-сложна, те може да се интерпретират като работа по проект на външен клиент. Разликата е, че става дума за услуги по настройка, внедряване или интеграция на съществуващ продукт (собствен, от външен проект или на трета страна).

Поддръжката на продукт е обичайно за определени в договор срок и задължения. Тя може да е постоянна с наблюдение или поддръжка при заявен проблем от клиента. Отличителното е, че се работи регулярно по мониторинг или по оплаквания за дефекти. Може да включва и минимално усъвършенстване на продукт.

Често към процеса на поддръжка на софтуер се включва и процеса на съпровождане**,** но в зависимост от контекста на употреба на понятието, между двата процеса може да има разлика. Според международните стандарти за жизнения цикъл на софтуерните системи, поддържането включва дейности по осигуряване функционирането на системата[13]. От друга страна, според други автори: „Съпровождането е процес на постоянно наблюдение върху работата на системата с цел разкриване на проблеми и своевременното им отстраняване.“[14] При обявяване на обществени поръчки в нашата страна, под „съпровождане“ се има в предвид съпровождане на потребителите, т.е. обучение и подпомагане на персонала при усвояване особеностите на работата с нова софтуерна система.

Можем да обобщим, че поддръжката е свързана с осигуряване на нормална работа на изграден и внедрен продукт, а съпровождането е свързано с внасяне на промени за коригиране на установени грешки, адаптация към нова среда, операционна система, хардуер, за подобрение на съществуващи и добавяне на нови функционални възможности. Много често значението на съпровождането се пренебрегва от специалистите по

[13] ISO/IEC/IEEE 15288:2015 Systems and software engineering - System life cycle processes.

[14] Пенева, П., Александрова, Я., Армянова, М. Бизнес информационни системи: Учебник за дистанционно обучение. Варна: Наука и икономика, 2013, 274. с.27.

информационни технологии, тъй като те предпочитат да работят по създаването на нов софтуер (на база личен опит).

Според нас, при съпровождането е подходящо да се добави регулярното наблюдение и документиране, които имат за цел да се проследява функционирането на продукта, установяване на нередности и описване на промените. Основни задачи са:

– Разучаване на продукт, изготвяне на документация за поддръжка, обучение;

– Регулярно наблюдение на поддържания софтуер;

– Приемане на оплаквания от клиенти и комуникация с тях;

– Анализ на проблемите, проектиране на измененията, програмиране и тестване;

– Интегриране и системно тестване на продукта;

– Описване на промените в експлоатационна и съпровождаща документация;

Поддръжката и съпровождането са трудоемки дейности, поради което са създадени множество средства за автоматизация, като модулно и интеграционно тестване, управление на конфигурации, документиране, интегрирани и лесно адаптируеми среди за разработка, и други.

Разработката на софтуер е високо интелектуален труд по създаване[15] на нов продукт. За разлика от стандартното производство, където продуктът е еднотипен, тук се оформя изцяло нов продукт. Динамиката на изисквания и технологии внася допълнителна сложност. Управлението на сложността и иновации в продукта могат да се получат с подходящ екип специалисти, процес, среда и управление. В този смисъл се откроява важността на **управлението на човешки ресурси**. Някои съществени характеристики на човешките ресурси в технологична

[15] Някои автори (Larman, 2011) споменават дори изобретяване на нов продукт с процес на учене.

стартираща компания може да се групират в следните групи[16]:

– Екип професионалисти, които ефективно създават иновативен и качествен продукт;

– Необходимост от нови знания, умения и качества на специалистите;

– Висок дял на разходите за човешки ресурси и тенденция за ръст на възнаграждения и мобилност;

Специфични изисквания към управлението по повод на мотивацията на човешките ресурси и важността на първата група, идва от търсения резултат – качествен нов продаваем продукт. Тук трябва да отчетем и ролята на обучението, което можем да разделим в три групи:

– за професионални умения, технологии и новости за конкретна среда;

– в адекватни подходи, методи, техники за усъвършенстване процеса на разработка;

– подобрение на работа чрез самообучение (обратна връзка и натрупване на опит).

Технологичното развитие на средствата за разработка, за автоматизация в програмиране и тестване изместват фокуса на дейностите и уменията. Затова, трябват не просто програмисти, а съзидателни личности, готови на автономност, отговорност и постоянно обучение. Счита се, че за да се работи успешно в бизнеса, трябва определена ценностна система и култура. Това поставя важният въпрос за подбора на екип. В този смисъл, усилията на българските технологични стартиращи компании трябва да са насочени към избягване и преодоляване на негативни черти на българската култура. Например, възможно е при типичната мотивационна структура, след постигане на безопасност, да следва ред на добрите връзки и поддържане на статуквото, т.е. липса на

[16] Цитираните групи и други допълнителни са дефинирани в (DeMarco & Lister, 2013)

мотивация за поемане на риск, новаторство и акцент над качеството.

Важността на човешките ресурси се обуславя и поради финансови причини, тъй като те имат преобладаващ дял в разходите за краен продукт и тенденция към повишаваща се цена и мобилност на професионалистите. Според едно изследване, съотношението в производителност на труда между „най-добри" и „най-слаби" е 10:1, а най-значимо обстоятелство е с кого работиш в екип[17] (DeMarco & Lister, 2013). Това налага по-голям акцент над ефективното управление. Без да навлизаме в детайли, обръщаме внимание на областите, които са важни за всяка организационна и методическа промяна. На първо място е *подбор и организационна култура*. Следва *усещането за лична сигурност*, за да може комуникацията в екипа и процеса да върви нормално с фокус над работата.

Важно е мястото на *мотивацията на екипа*, която е широкоспектърен процес с много условия (не само парични стимули). Съществуват множество теории и модели за мотивация, а правилното им прилагане може да доведе до повишаване на производителността и организационното представяне. За дейността на компанията е важно и ефективното лидерство (Трейси, 2011), което да насочва екип и да взема крайните решения[18]. Лидерът в екипа трябва да интегрира знание и разбиране за бизнес потребности, технологии и употребата им. Неправилно е той да е зает с административно управление на задачи и отделен от екипа (Икономи, 2018; Даскал, 2018). Можем да обобщим, че човешкият ресурс е елементът с най-голяма сложност за управление, но единственият, който може да допринесе за значими достижения в технологичната стартираща компания.

---

[17] Изследването е от състезания. Показва, че най-добрите са 2-2,5 пъти по-добри от средното ниво, а минимално значение (в проценти) имат показатели като: език за програмиране (от едно и също ниво), опит (над 6 мес.) и заплата. Допуснати дефекти и по-добра работна среда водят като тенденция до по-добро време/резултат.

[18] Според (Tracy, 2015) основните качества на лидера са: визионерство/проницателност, почтеност, смелост, реализъм и отговорност.

Въпросът с ефективното **управление на финансовите ресурси** стои пред всяка компания. Особена тежест има при технологичната стартираща компания, тъй като обичайно в началото разполага с ограничени финансови ресурси и не може да разчита на тях като съществено конкурентно предимство. Нещо повече – при разработка и промоциране на нови продукти компанията рядко може да разчита на дългосрочно солидно финансиране. Затова доброто финансово управление и управление на разработката е належащо, а то е неразривно свързано с доброто управление на човешките ресурси.

Както посочват редица автори (Duff, 2013; Martin, 2017; Jolselt, 2019) обществото навлезе в ера, наречена информационно общество. Тя се характеризира с изравняване ролята на информацията с другите ресурси, широк достъп на всеки член от обществото до информационни ресурси (без класифицираните), условия за по-пълноценно използване на човешките интелектуални способности. Това изисква информатизация на управлението на стопанските системи с цел въвеждане на нова форма на управление с по-висока ефективност и оптимизиране на процеси.

Успешното управление на всяка компания в днешно време зависи от софтуерната система. Тя трябва да предоставя информация за навременно, адекватно и ефективно вземане на решения. Изясняването на информационните потребности е необходимо за обхвата очакван от софтуерната система и задачите й. За **процеса на разработка** е нужна информация за:

– Вид, размер и статус на ресурсите на компанията – материални, финансови, човешки;

– Продукти в процес на разработка[19], техните параметри и статуса им;

– Потребности от ресурси, време и разпределението им по проекти

[19] Планирани или възложени продукти, договорени проекти за разработка на продукти;

и етапи;

– Изисквания към всеки продукт, документация, проектни варианти;

– Състояние на продукта и реализация на изискванията в разработвания софтуер;

– Статус на проекта и процеса, с който се изпълнява;

– Представяне на екипа като цяло в проекта и индивидуално;

– Разход на ресурси и време в проекта;

– Изпълнение и реализация на продукта (и негови версии);

– Обратна връзка от клиента и крайните потребители (ако са различни).

За **поддръжката на продукти** се изисква информация за:

– Договорена поддръжка и данни за клиенти;

– Оплаквания (заявки) на клиенти за проблеми и дефекти, статус;

– Статус на заявките и направеното по тях; Удовлетвореност на клиентите;

– Методи/тестове за регулярно наблюдение, резултати и открити дефекти;

– Екип – статус, отговорности по поддръжката и по заявки; представяне на екипа;

– Разходи по поддръжката и отделни проблеми;

– Съотношение на приход от продукт/поддръжка и разходи за тази дейност;

– Внедряване на промени и нови версии, документация за потребител и поддръжка.

Съществуват различни **методологии за разработка на софтуер**, подходящи за стартиращ бизнес. Интерес представляват т.нар. „леки методологии”, които по същество са опростени и с ниско ниво формализация и верификация (Филипова и др., 2017).

Пример за методологии, които са и в същото време популярни сред

специалисти, разработващи софтуер, са SCRUM (Fowler, 2019) и XP (eXtreme Programming) (Larman, 2011). Друга подходяща методология за разработка на нов софтуерен продукт е LSD (Lean Software Development) (Poppendieck & Cusumano, 2012). Общото между тях е, че продуктът се изгражда постъпателно по етапи (releases) и стъпки (iterations), като на всеки етап има реализация на частично завършен /преработен/ продукт.

В процеса на работа планът може да се изменя (стъпката започва с планиране), но в рамките на една стъпка той е по-скоро фиксиран. Характерна е регулярната обратна връзка от клиента, работа в малки екипи, ревизия на направеното и процеса в края на всяка стъпка. Член на екипа има роля да следи за спазване на принципите, протичане на процеса и работа на екипа с клиента.

Отделните подходи изискват създаване на артефакти (елемент работа и документация, част от продукта и междинни резултати) и имат определена терминология. Жизненият цикъл на разработка се състои от начални етапи, стъпки на изграждане и етап на завършване/внедряване на продукта. Например, за стегната разработка (Lean), етапите са: проучване, одобрение на проекта, предварително проектиране (архитектура), итерации за сегменти от продукта (всяка с проектиране, програмиране, тестване и верифициране с клиента) и завършва с внедряване.

Във всеки подход „потока работа” трябва да се „определя от заявки на клиента” с акцент над ефективност и скорост. За целта може да се използват инструменти за обратна връзка с клиента, приоритети, наблягане на емпиризма и „ученето” в процеса на разработка и други (Fowler, 2018). Важен елемент са знания, опит и възприемчивост на предприемача към методологията и на този етап се смята, че е много вероятно предприемачи от софтуерния бранш да познават някоя от така изброените методологии.

В следствие на така изложените принципи и ограничения на предприемаческия процес в стартираща компания, са разработени методологии за нов бизнес и продукт. Дефинирани са поредица от стъпки

за откриване на проблем, решение, пазар, валидиране, корекция и други. От гледна точка на стартиращи компании в англоезичната литература за startup са развити различни специализирани методологии, производни на по-общи методологии за т.нар. „стегната работа“.

Интерес представлява адаптация на общия подход „Стегната разработка” (Lean Development), която за разработка на стартиращ бизнес е наречена „Стегнато стартиране” (Lean Startup) (Ries, 2011). В този подход основна концепция е, че с определени принципи и инструменти екипът трябва да се концентрира над доставяне на „стойност” (в смисъла на удовлетворени ценни потребности) на потребителя във възможно най-кратко време и високо качество, което се постига чрез ефективен „поток на стойността”. В методологията Lean Startup са вложени следните принципи и инструменти (Ries, 2011; Maurya, 2012; Blank, 2013; Alvarez, 2014):

1. Основен резултат е бизнес моделът, а не само новият продукт
    a. Инструмент Lean Canvas – схема на бизнес модел за стартираща компания на база схемата на Остервалдер (Business Model Canvas) (Osterwalder & Pigneur, 2010);
2. Фокус над създаващите стойност дейности;
3. Интензивна обратна връзка от клиенти:
    a. Създава се минимално функциониращ продукт (MVP) (Duc et al., 2016);
    b. Регулярна обратна връзка от клиентите;
    c. Тестване на версии с група клиенти за валидация (Split or A/B Testing);
    d. Ползват се действени показатели (Actionable Metrics) водещи пряко до управленски решения;
    e. Структуриран процес за корекции в модела (Pivot);
4. Опростен и повторяем процес:
    a. Проблем и решение (Problem/Solution Fit) с цел намиране на MVP;

b. Продукт и пазар (валидиране на бизнес модела) (Product/Market Fit);
c. Разрастване (Scale);

5. Кратки цикли итерации с три стъпки за създаване на артефакти:
   a. Създаване (Build) – създава се артефакт (MVP, модел и други) с хипотези;
   b. Тестване (Measure) – тест с показатели; обратна връзка от клиенти;
   c. Извличане на знание (Learn) – с тестване на хипотези се извлича знание;
6. Ползване на методи за гъвкава разработка (Agile Engineering);

Основна разлика между методите Lean Startup и Lean LaunchPad е в началната идея – базирана на технология или на клиентски проблем. В процеса има интеграция на изобретяване (Design Thinking), процес за лесно стартиране (Lean Startup) и гъвкава методология за разработка на софтуер[20].

Изброените модели на предприемачески процес са приложими и за технологични (софтуерни) иновации. Разликата е в набора от използвани инструменти за реализацията им и дали те могат да се използват изцяло. Нещо повече, възможно е етапите на лесно стартиране и гъвкава разработка да са паралелни и взаимно зависими (Mullins & Komisar, 2013; Moran, 2016), а именно – разработката на бизнес модел и продукт да текат паралелно, като се обменя между тях информация относно изисквания за продукта, разработен междинен продукт, тестове с продукта и клиенти и корекция.

[20] Т. нар. „гъвкава разработка“ използва „гъвкава методология“, която най-общо казано представлява съвкупност от техники за управление разработката на софтуер, при които основната идея е, че разработката на софтуер е динамичен процес, при който дългосрочното планиране има ниска ефективност. В хода на работа по даден продукт, е необходимо разработчиците постоянно да поддържат обратна връзка с клиента и непрекъснато да адаптират разработката си, съобразно ново постъпващите изисквания.

Разработени са и други модели на предприемачески процес със съответна прилика. Някои автори посочват инструмент „верига на иновационната стойност" и модел за управление на иновацията с 4 етапа – създаване на идея, избор на проект, разработка и комерсиализация (Robehmed, 2013). Други автори представят различни модели за технологични иновации и създават интегриран модел на технологични иновации (ИМТИ). В него бизнес моделът е интегрална част от иновационния процес с основни елементи – маркетингов продукт, технологично решение и канал за продажби. ИМТИ има три фази (Русева, 2015):

1. Генериране и избор на идея – резултат е маркетингов продукт

2. Валидиране и моделиране на бизнес идея – резултат е бизнес модел

3. Технологично решение – създаване като минимално необходим продукт (MVP);

Според същия автор софтуерното решение е последен етап в процеса, защото поради високата скорост на развитие в бранша е невъзможно създаване на конкурентно предимство чрез технологично решение (Русева, 2015).

Съществуват различни класификации на **бизнес модели**, описани в литературата за електронен бизнес (е-бизнес) и електронна търговия (Rainer & Cegielski, 2013; Laudon & Traver, 2019). Много от тези модели имат елементи, които се реализират чрез виртуални компоненти, като например сайт, виртуална софтуерна услуга, виртуализация на хардуера и т.н. Бизнес моделите могат да се класифицират по различни критерии. Първо, това е тип на участниците – бизнес, краен потребител, администрация и други (B2B, C2C, B2C, C2B и т.н.). Второ, за различните типове има различни модели според вида на услугата и начина на предоставяне. Едно примерно изброяване на модели за типове B2C и B2B са дадени в табл. 1.2.

Таблица 1.2.
Бизнес модели според тип участници и вид на услугата.
(адаптирано от автора по Филипова и др., 2017; Croll, 2013)

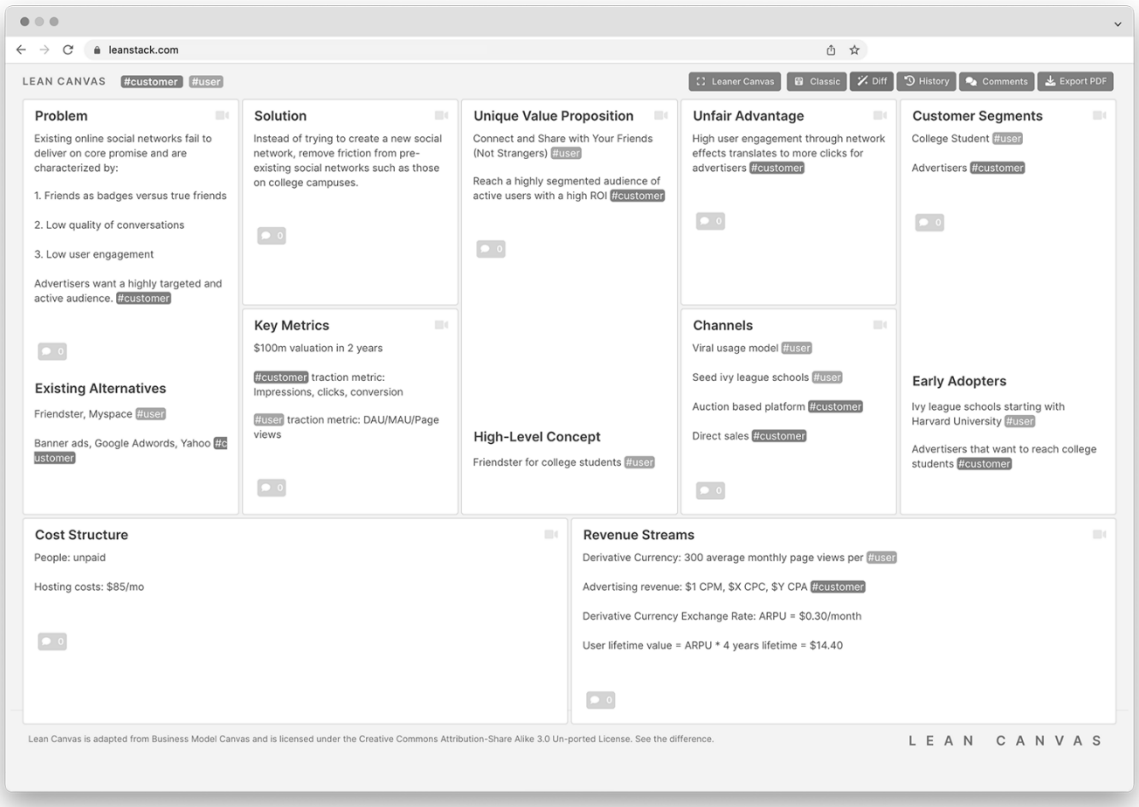
№	Модел по тип/подтип (Филипова и др., 2017)	Група модели по Крол/Йосковиц (Croll, 2013)
	Бизнес към потребител (B2C)	
1	Е-магазини	Електронна търговия
2	Е-молове	Електронна търговия
3	Портали	Електронна търговия
4	Доставчици на съдържание	Медия сайт; Приложение
5	Транзакционни брокери	Електронна търговия
6	Създатели на пазарна среда	Пазари
7	Доставчици на услуги	Софтуер като услуга
8	Платформи по интереси	Потребителски генерирано съдържание
9	Е-аукциони	Пазари
10	Сайтове с динамични цени	Пазари
	Бизнес към бизнес (B2B)	
1	Системи за поръчки	Пазари
2	Електронен обмен на данни	
3	Електронни витрини	Електронна търговия
4	Електронни пазарни платформи Е-дистрибутори Онлайн пазари за ресурси Борси	Пазари; Електронна търговия
5	Частни индустриални мрежи	

Трети подход е бизнес моделите да се групират от друга гледна точка на модел за генериране на приходите (Croll, 2013), канали за продажба (Maurya, 2012; Blank, 2013), индивидуализация на решението и други. Правим уточнението, че това е само един компонент на бизнес моделите. Например, някои автори идентифицират основни елементи на бизнес модела (маркетингов продукт, технологично решение и канал за продажби) и предоставя класификация на бизнес модели за стартиращи софтуерни компании по две скали – индивидуализация на решението и канали за продажба (Русева, 2015).

Примерна класификация на бизнес модели специално за софтуерен продукт, според модел на предоставяне на услугата и генериране на приходи (виж табл. 1.2): Електронна търговия – когато се купува стока/услуга от сайт на търговец; Софтуер като услуга – когато софтуера се ползва/доставя на поискване; Безплатно приложение – предоставя се безплатно; плаща се за екстри; Медия сайт – публикуване на съдържание и приходи от реклама; Потребителски генерирано съдържание – пример:

социални мрежи; Пазари – потребители търгуват на електронен пазар; пример: борси, аукциони.

Независимо от вида на модела, за предприемача са по-важни неговите конкретни компоненти и подходящия начин да се определят, изпълнят, валидират. Възможно е да се ползват базови модели за описание на бизнес модели (Osterwalder & Pigneur, 2010), бизнес планове, но за стартиращия бизнес е важно да се концентрира върху най-важното. От тази гледна точка, в литературата има описани методи и инструменти за описание на модели. Например, в съгласуваност с посочените методологии за разработка на стартиращ бизнес и принципите на стегната разработка, някои автори описват инструмент за дефиниция, наречен схема за стегнато стартиране (виж фиг.1.1) за разработка на бизнес модел (Maurya, 2012; Link, 2016; Nidagundi & Novickis, 2017).



Фиг. 1.1. Схема за стегнато стартиране (Lean Canvas) за разработка на стартъп бизнес модел. Източник: Leanstack <https://leanstack.com/lean-canvas>, [23.08.2022]

Схемата за стегнато стартиране (Lean Canvas) за разработка на бизнес модел за стартиращи компании има 9 свързани елемента и се представя обичайно като схема на една страница. Базирана е на схемата за бизнес моделиране (Business Model Canvas), създадена от Alex Osterwalder (https://www.alexosterwalder.com/). Основните елементи на схемата са следните: проблем (problem), решение (solution), уникалната комбинация от продукти и услуги, която има стойност за клиента (unique value proposition), нечестни предимства (unfair advantage), потребителски сегменти (customer segments), ключови метрики (key metrics), канали за дистрибуция (channels), структура на разходите (cost structure), източници на приходи (revenue streams). Предлага се и онлайн инструмент за нейното създаване – https://leanstack.com/lean-canvas.

В заключение, може да се обобщи, че в настоящата точка се разглеждат основните проблеми, свързани с разработката на софтуерни продукти, като са разгледани особеностите на софтуерния продукт за външен клиент, собствените софтуерни продукти, собствения софтуерен компонент, индивидуалните услуги, поддръжката и съпровождането на продукт. Разгледани са специфични въпроси касаещи управлението на: човешките ресурси, финансовите ресурси и процеса на разработка. Изследвано е използването на различни методологии за разработка на софтуер, различни модели на предприемачески процес и процесът на създаване на бизнес модел, като е представена схемата за стегнато стартиране (Lean Canvas) за разработка на стартъп бизнес модел. Налага се изводът, че управлението на софтуерни проекти в технологичните стартиращи компании е сложен процес, който има нужда от допълнително изследване.

1.3. Управление на софтуерни проекти в технологичните стартиращи компании

За да бъде бизнес моделът на стартиращата компания успешен, то трябва да се създаде организация за управление, която да го изпълнява и подобрява. Организирането се разглежда като ключово понятие в теория на управлението. Организациите може да се определят като „целенасочени социални образувания, които са проектирани като съзнателно структурирани и координирани системи от дейности и които са свързани с външната среда" (Daft, 2020). Създадената организация на управление може да постигне дефинирана цел, която е над възможностите на един човек чрез по-производителни методи – разделение на труда, по-мащабни и съвременни технологии, икономии от разходите и други.

От тази гледна точка технологичните стартиращи компании изпълняват специфични дейности и чрез резултатите от тях се постига целта. От своя страна, структурата определя поведението на технологичните стартиращи компании, а то е от изключително значение, тъй като създава привързаност към общи цели и ценности и оказва влияние при свързване на служителите с външната среда, тъй като резултатът от дейността се изявява извън организацията.

За стартиращите технологични компании вземането на решения е основно в условия на риск и неопределеност. Затова по-детайлното дефиниране на управленските функции може съществено да подобри работата, тъй като е възможно да се избегнат някои проблемни ситуации на по-ранен етап, което е и по-оптимално с оглед цялостното функциониране на организацията в дългосрочен план.

При разработка на софтуер широко приложение намират методите на емпиризъм и принцип на обратна връзка:

– процесите се движат на принципа „тегленето" (известен и като Kanban);

– ръководителят присъства на място, сред служителите и където се създава продуктът;

– цели се винаги и всичко, което е възможно да се тества и оценява, включително и за оценка на служителите, което може да не е подходящо за началните етапи на функциониране на стартиращата компания, поради липса на време и ресурси;

– стратегическото управление е с процесен, а не с аналитичен подход – стратегията се формира от активна и гъвкава вътрешна среда[21], постепенно, при взаимодействие с външната;

– експериментална адаптация чрез учене от опита, вместо предварително дългосрочно прогнозиране.

Тези принципи надграждат класическите теории за управление в направление за работа в по-малки пазари, по-гъвкави модели продукти, при по-нестабилна външна среда. На тази база се създава моделът за учещата се организация – учене чрез опит и системни групи и процеси за усъвършенстване, модел на бенчмаркинг – сравнение с лидер в бранша, модел на двойния кръг – за получаване на обратна връзка от партньорите и конкурентите за промени в средата, модел на мрежата – външната среда не е само обобщено пазарът, а комплексна мрежа от взаимодействия с много организации в различни измерения; предлагане на решения на база малки постоянни групи, постоянно усъвършенстване.

Въз основа на тези постановки впоследствие се изгражда методологията Lean Production (стегнато производство) и през 90-те години на 20-ти век Lean Development (стегната разработка). Тя е адаптивна към промени, базирана на опита и интегритета на всички части от организацията, с високо качество и интегритет продукция (Poppendieck & Cusumano, 2012).

[21] Гъвкавата вътрешна среда е работна среда, която може лесно да се адаптира към външните (за организацията) условия.

Освен организационната структура, съществен въпрос е и използваният **механизъм на координация и управление на проекти**. Разработката на софтуер по своите характеристики принадлежи на групата наречена разработка на продукт (Product development). За разлика от обичайното производство в тази група се „произвежда" проект за един продукт с определени изисквания и по-късно той се „размножава" в производствена система, продава, дистрибутира и внедрява (Poppendieck, 2011). Затова в разработката на софтуер се ползват методологии, базирани на общата рамка за разработка на продукт със съответната специализация, принципи, методи и инструменти на работа. Класическият модел е на четири етапа (Kahn, 2012; Zacca & Dayan, 2017):

1. Изработка на изисквания – действията, които продуктът трябва да извършва;

2. Проектиране на продукта – определяне на начините, по които продуктът изпълнява изискванията;

3. Реализиране на продукта – конструиране и тестване на продукта;

4. Комерсиализация – пускане, продажба, внедряване, сервиз.

Всеки етап има различни стъпки и инструменти, които се ползват. Най-рисков етап е този на изискванията, в който има различни подходи според степента на неопределеност и вида на външната среда от която се получават. При избора на методология, следва да се имат предвид няколко критерия специфични за стартиращите компании и предприемаческа активност:

1. Източник на изисквания – пазарни проучвания и обратна връзка от клиент – интервюта, фокус група, клиент прототип и други;

2. Адаптивност към изисквания – да може да се адаптира към промени на изисквания, тъй като те не са изцяло ясни в началото и се развиват във времето и с обратната връзка от клиентите;

3. Адаптивност към времето – да може да се достави продукт навреме. Нещо повече, ред методологии говорят за ползване на междинно

готов минимално функциониращ продукт (MVP), с който да се работи;

4. Адекватна към продукт и нужен екип – преценявайки по класификация за критерии цена на загуба и размер на екип.

Важен параметър, който трябва да се има предвид, е източникът на изисквания за продукта, яснота и променливост на изисквания към продукта. Яснотата има значение, дали могат да се определят в началото, дори с риск изменение или напълно неясни и добивани от опита. Яснотата е важен параметър, тъй като някои класически методологии като Stage-Gate model (Waterfall) са подходящи за стабилна среда с ясни и непроменливи изисквания.

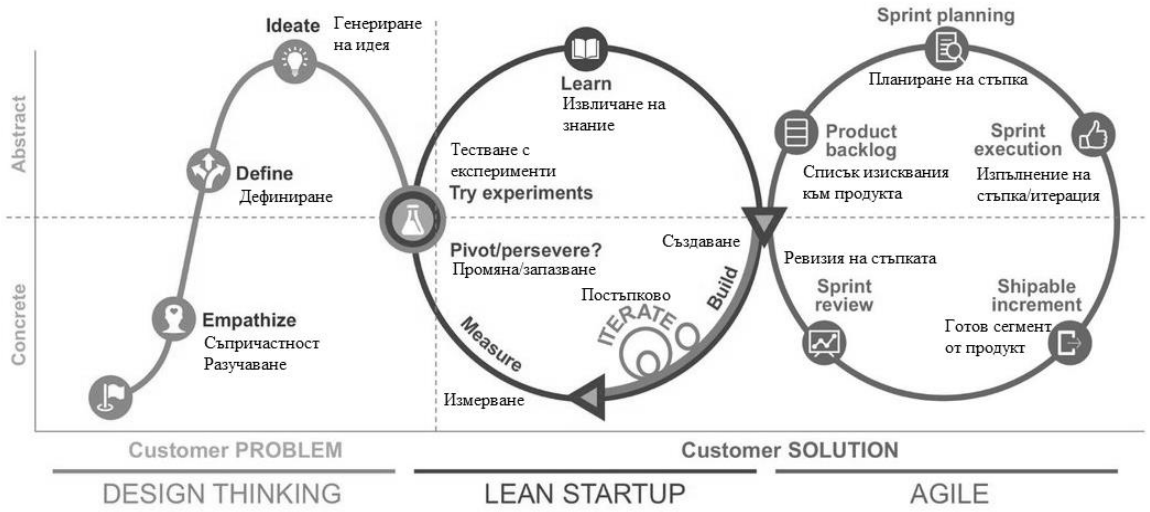
Трети групи, основно от групата „стегната разработка" са проектирани да работят в неяснота, излизане на пазара, променлива среда. Също така, методологията и нейните практики трябва да са познати на участниците.

Другият важен елемент на дейността на стартираща софтуерна компания, който в едни случаи предшества, а в други случаи се разработва паралелно с продукта, е разработката на бизнес модел. При така изложените ограничения на предприемаческия процес в стартираща компания, има развити методологии за разработка на бизнес модел с пазарна валидация. В тях има стъпки на откриване на проблем, решение, пазар, валидиране, корекция и други.

От гледна точка на стартиращи компании в англоезичната литература за startup са развити различни методологии производни на стегната разработка (Lean Development) и адаптирани за случая. За разработка на стартиращ бизнес е наречена „Стегнато стартиране" (Lean Startup, Lean Launchpad). В него основна концепция е, че на отделни етапи и с определени инструменти екипът трябва да се концентрира над доставяне на „стойност" (в смисъла на удовлетворени ценни потребности) на потребителя във възможно най-кратко време и високо качество, което се постига чрез ефективен „поток на стойността" и постоянен цикъл на

тестване, валидиране на хипотези и учене от опита и тестове.

На фиг.1.2 е показана примерна конструкция на процес, съчетаващ три етапа – изобретяване, разработка на бизнес модел и разработка на технологично решение. Във втори и трети етап се използват гъвкави методологии, като те могат да са последователни във времето, но по-удачно е да протичат паралелно. Фигурата е с посочени термини на английски и български език. Кръговете изобразяват постъпкова разработка (с итерации) и връзка като данни и процеси. Възможна е повторяемост в един инструмент, обратна връзка и връщане назад в процеса, когато се установи, че дадена идея, изискване или хипотеза не могат да се изпълнят или не са подкрепени от тестове.



Фиг. 1.2. Диаграма на процеса в Lean Startup, интегрирана с разработка на софтуер. Източник: Gartner, Inc. <https://medium.com/@SteveGlaveski/the-difference-between-design-thinking-lean-startup-and-agile-5cf07b117562>, [09.10.2020]

Практики за координация, които се ползват често в тези подходи и са често срещани в литературата и практиката, са обобщени в табл. 1.3.

Таблица 1.3.
Обобщение на практиките за координация, описани в теорията и често срещани в практиката
(разработка на автора)

Практика за координация	Описание
Разделяне на проектни малки групи с ръководител	Продуктът се декомпозира на модули или части, всяка група се занимава с една част
Разделяне на разработката по време на стъпки (итерации)	Всяка добавя или изменя в продукта отчасти с всички дейности
Регулярни срещи за синхронизиране в екип	Планиране, ежедневна работа, край на етап или проект за обратна връзка
Прилагане на метод за „Управление на проект“	Задачи, ресурси, разпределение в екип и по време, вкл. отчети за екипно, стъпково, лично представяне, споделена проектна документация
Свързваща роля с „бизнес възложител”	Въвеждане на роля „бизнес аналитик” със задача спецификация на изисквания
Свързваща роля „проектен мениджър”	Управлява всичко за проекта по избран метод за управление на проекта
Свързващите роли са в екипа	Роли „бизнес аналитик“, „проектен мениджър“ и други са част от екипа и участват във всички срещи и решения на екипа
Координиране на решения с опции	При вземане на решение за взаимно зависими елементи, винаги се предлага група опции за избор и се стиковат от двете страни
Синхронизиране на интерфейси	Разделяне на продукта по модули и екипи, а ръководители на отдели и архитекти координират „свързване на модулите” да е ясно, устойчиво и малко изменчиво
Разработка на слоеве	Софтуерът се разработва на слоеве, така че системата да е устойчива на промени в един слой без нужда от корекции в друг (опаковане на модули)
Информационни радиатори	Информацията за статуса на проект/продукт е достъпна за всички, чрез визуално онагледяване на прогреса на видно и леснодостъпно място
Информационни табла	Най-важните оперативни отчети са на видимо място за всеки според правомощията и актуализирани постоянно
Работа по двойки	Двама специалисти работят по една задача
Работа в един офис един екип	За пряка хоризонтална комуникация
Ревю на код, взаимно одобрение	Първичен контрол на качеството чрез одобрение от колеги и контрол на спазването на стандарти на работа
Регулярна/автоматизирана интеграция на продукт	При всяка промяна или ежедневно от текуща работа продуктът се асемблира готов за тест
Регулярни/автоматизирани системи за тестване	Регулярна обратна връзка за качеството на продукта и нужда от корекции
Виртуален офис	Споделяне на информация в екипа независимо от работното място
Групи и срещи по технологии	За обмен на това какво се работи, как, какви проблеми има и технологичен обмен по функционален признак
Матрични структури	Специалистът е във функционален отдел, в който се развива професионално, но основно работи в проект с проектен ръководител
Следване на стандартни процеси	Процесът е добре дефиниран и неговото изпълнение от всички участници осигурява взаимна координация
Отворени стандарти за технологии и решения	Ползват се публично достъпни и популярни стандарти, технологии, решения, инструменти, с цел икономия на развой, по-лесна интеграция на модулите, обучение

Съвременното разбиране за успешно управление утвърждава необходимост както от изграждане на добри бизнес процеси, така и на подходяща и ефективна софтуерна система. Софтуерната система помага на технологичните стартиращи компании за въвеждане на нови форми на управление с по-висока ефективност и за оптимизиране на процеси. Тя трябва да предоставя информация за навременно, адекватно и ефективно

вземане на решения. Удачно е да се направи обзор на целите й и съвременните тенденции. От тази гледна точка, информационното осигуряване има следните роли:

1. Предоставяне на информация за управлението на: софтуерен процес в рамките на софтуерен проект; софтуерния бизнес по функции и задачи.

2. Автоматизация по вземане на решения за управление: средства за подпомагане вземането на решения и автоматично вземане на решения от системата (например разпределение на работа и време по проект от разполагаемото и очаквано време).

Тенденциите при разработката, доработката и съответно прилагането на софтуерните системи са обусловени както от технологичните промени в хардуера и софтуера, така и от повишаващите се изисквания от страна на бизнеса. Основни направления за прилагане на софтуерни системи за управление на бизнеса според нас са: употреба на системи за управление на бизнеса (ERP) – те покриват много функции с настройваемост, цялостен поглед и възможност за оптимизиране на бизнес процеси; приложение на системи за връзка с клиентите (CRM) – за по-добро познаване на клиента, потребностите му, история, покупки, целеви маркетингови усилия; системи за управление на веригите за доставка (SCM) – ефективно и интегрирано управление (по тип бизнес) на процесите по снабдяване, производство и дистрибуция; групова работа – екипа работи заедно чрез системата съобразно отговорностите си; централизация на данните и достъпа до системата (технология клиент-сървър); приложение на интернет технологии с възможна мобилна мрежова работа; бездокументални (с електронен вход/изход) информационни системи; електронни форми на бизнес и интеграция с основния бизнес софтуер; виртуализация на информационната система, достъп на екипа чрез виртуални методи и средства до информационната система (Илиев и др., 2010); автоматизация на управленски процеси чрез

залагане на правила и политики; общи стандарти за формат и обмен на данни, отвореност/съвместимост на системи; използване на лесен за употреба и настройка потребителски интерфейс; възможности за индивидуализация чрез настройка; възможност за преизползване чрез директен програмен достъп (API / SOAP).

Могат да се добавят и други детайли, но по-важни са тенденции, касаещи тезата. Акцентът в тезата е към система, подпомагаща управлението на софтуерни разработки (проекти и продукти), както и специфична част от управлението на софтуерния бизнес.

1.4. Подходи за управление при разработката на софтуер

Редица автори предлагат **структурирани подходи** на разработка (Curtis & Cobham, 2008; Rowley & Hartley, 2017), които изискват декомпозиция на системата на функции, подходящо структуриране, наличие на пълна спецификация в самото начало на разработката и минимални промени в последствие, цялостно завършване на една фаза преди започване на следващата. Реализира се в хронологично последователни фази за разработка на продукта – определяне на системата (цел, задачи, обхват, реализуемост и изисквания), проектиране, програмиране, тестване и интегриране, използване (внедряване и поддръжка). В проектирането се описват функции и структура на системата, процеси, логическа структура на данните, интерфейс на системата.

Структурата на софтуерната система се представя като йерархия от модули с техните свойства – вход и изход, функция, механизъм за реализация на функции, вътрешни данни (Nuchprayoon & Phuaksaman, 2018). В следващите фази продуктът се кодира, тества и внедрява. Вариант на структурираните подходи е каскадният модел на жизнения цикъл, който

има възможност за връщане в предходен етап за корекции. Подходът е приложим за силно структурирани проекти в области с устойчиви изисквания. Като основен недостатък може да се посочи растяща цена за корекция на грешки в предходна фаза, голямо натоварване на екипа при тестване и ниска адаптивност спрямо т.нар. гъвкави подходи.

Друг често използван подход в практиката е **обектно ориентираният подход**, който е пряко свързан с еволюцията на обектно-ориентираното проектиране и програмиране. При него системата се разглежда като съставена от обекти (групирани по тип като класове) с тяхното статично състояние и поведение в динамика. Разработваната система се описва чрез обектно-ориентирани модели[22]. При обектно ориентираният подход специфични дейности са: анализ, проектиране, програмиране и тестване. В резултат се моделират статични и динамични характеристики на системата чрез система от класове и обекти, с тяхната структура, връзки и поведение.

Проектирането на продукта е на две нива – системно (външно, за софтуерната архитектура, подсистеми, потребителски интерфейс и логическа структура данни) и обектно (вътрешно, за класове, обекти, интерфейсно описание, операции и взаимодействия между тях). След като проектът е готов следва програмиране на обектно-ориентиран език за програмиране. Тестването се извършва по класове (модулно) и тестване на група класове (интеграционно). Ключово в подхода е създаването на обектно-ориентирани модели (архитектурно центрирани), които се описват на език за моделиране. За целта може да се използва UML, който е отворен за разширение към други системи[23].

[22] Системата се описва чрез обектно-ориентирани модели чрез изграждане на йерархия от класове и обекти, които да представят нейните статични структура и състояние, както и да реализират динамично поведение чрез взаимодействие между обектите с помощта на съобщения (Booch, 2018). Моделите представляват т.нар. „изгледи на системата“.

[23] Универсалността и разширението позволяват моделиране на всякакви системи, обекти и процеси с тях.

Техниките за управление при обектно-ориентиран подход могат да се разделят на управление на процес, управление на проект и продукт, и управление на персонал. При управление на процес се счита, че най-голям резултат може да се получи от процес, движен от сценарии за взаимодействие на потребителя със системата, който е основан на модели, и е итеративен и инкрементален (Booch, 2018). Допълнително може да се използва рекурсивно-паралелен подход, който разработва последователно разширяващ се прототип (с всички фази за всеки прототип) до изграждане на краен продукт (Wautelet, 2020). Управлението на персонал и проект е организирано в роли – стратези, програмисти на компоненти и на приложения. Първите задават насока, съветват и координират. Вторите проектират, разработват и тестват компоненти. Когато са готови, поставят ги в библиотека от компоненти, за да могат програмистите на приложения да ги ползват в конкретните продукти (Dwivedi & Rohilla, 2017).

Гъвкавите подходи са вид подходи, широко прилагани в съвременната практика. При малки компании често има предизвикателства за кратки срокове, конкурентна цена и променливи изисквания, съкратен технологичен и пазарен живот на продуктите. Подходите за ускорена разработка, в които влизат и гъвкави подходи, се появяват през 90-те години на миналия век след установяване на слаба ефикасност и ниски резултати при класически прецизно дефинирани и напълно управляеми процеси на разработване, при усъвършенстването на продуктите с повече функции и опции, увеличаване на конкуренцията и повишаване на изискванията за качество. В този смисъл, гъвкавите подходи са предназначени да отговорят на съвременните изисквания за разработка на софтуер и нови продукти, приложения с богат интерфейс, в интернет и мобилна среда, електронни форми на бизнес и т.н. Аналогия има с проектирането на нови фабрични продукти, където се налагат при дизайна

и разработката[24]. Основната разлика спрямо строго формализираният процес е, че разработката на софтуер е дизайн на нов продукт и процесът се оприличава по-скоро на „откриване", а не на масово производство (предвидимо, с детайлни спецификации в началото) (Conboy & Carroll, 2019). При разработката на софтуер възможността да се предвижда и управлява се появява с натрупване на опит по проекта, при което може да се очаква известна непредвидимост и промени в изискванията към продукта в хода на разработка.

Гъвкавите подходи решават проблема с адаптация към промените (Bass & Haxby, 2019) чрез работа на стъпки с обратна връзка от клиента, които са итеративни, инкрементални, еволюционни и адаптивни, т.е. продуктът се подготвя на отделни етапи и „израства" постепенно. Итерацията, като базов компонент, представлява самостоятелна времева фаза от проект, в която се извършват всички дейности по анализ, проектиране, програмиране, тестване на планирана функционалност и завършва с някакъв частично завършен (интегриран и тестван) продукт за нея. Последната итерация реализира изцяло завършен продукт. Итеративната разработка е и инкрементална (с натрупване), като следващи итерации добавят функционалност и „продуктът расте". Важно правило в повечето гъвкави подходи е итерацията да е фиксирана времево, като периодът винаги се спазва за сметка на функционалност с нисък приоритет. Продължителността на отделни итерации може да е различна (Oorschot et al., 2018).

Подборът на функционалност за реализация в една итерация следва приоритет, или движен от риска – първи са по-рисковите функционалности, или движен от клиента – първи са тези с най-голяма бизнес стойност (Gren et al., 2020). Като цяло, подходите за итеративна

[24] Имаме предвид процеса на проектиране и разработка на нов продукт, а не самото му фабрично производство. Първото е уникален проект и процес, а второто е повторяем по задание, което е резултат на разработката.

инкрементална разработка възприемат промяната като неизбежност и следват еволюционен адаптивен процес. Те използват обратна връзка с потребителя[25], възможност за промяна на изисквания и корекции, което е възможност за адаптация. За да има предвидим и стабилен процес се налага като ограничение промените да са възможни преди фиксиране на изискванията за итерация и невъзможни (или минимални) по време на разработка в итерация.

Повечето модерни подходи за итеративна инкрементална разработка започват с разработка на визия, най-важните ключови изисквания, анализ и проектиране на ключова архитектура. При планиране на усилия, разходи и време трябва да се отчита ефекта „конус на несигурност" като отклонение между планирано преди започване на работа и краен резултат по проект. Обикновено в началото има само общ обхват, прави се една или няколко итерации, и когато „конусът" се свие, планът се коригира като реалистичен и възможните отклонения в края са по-малки (McConnell, 2019).

Гъвкавите подходи се базират на следните по-съществени принципи:

– ранна, непрекъсната и в срок доставка на ценен за потребителя софтуер;

– първичният измерител за прогрес и постижение е работещият софтуер;

– постоянна обратна връзка за продукта от клиента;

– клиентът работи редовно с екипа има с него пряка връзка;

– вниманието е насочено към отлични решения, добър дизайн и простота на решенията;

– необходим е екип с мотивация, свобода и компетенции да свърши

[25] Под потребител се има предвид всяка форма на краен потребител или проверяващ - клиент, краен потребител, тестери, потребителска фокус група и т.н., в зависимост от продукта и реалните крайни потребители.

работата;

– регулярно екипа коригира поведението си към по-добра ефективност.

В хронологичен план с най-дълга история е подходът **Evo** (Evolutionary Value Delivery) – еволюционен подход (Abbas et al., 2008; Clarke et al., 2018). Характерно при него е, че се набляга на кратки цикли с еволюционно формирани изисквания, дизайн и доставка, и адаптивно планиране, движено от ценност на функциите. Използва се език за спецификации от високо ниво, измерими показатели за прогрес и числово дефинирани количествени изисквания. Изисква ясно дефиниране, сбитост, оценка и измерване на представянето като качество, капацитет, разход на ресурси, наблюдавани по време на дизайн и итерации (Hanssen, 2011).

Подходът **UP** (Unified Process, унифициран процес) и търговският му вариант RUP, са създадени от авторитетни изследователи в областта на обектно-ориентираното моделиране[26]. Подходите се характеризират с универсалност и постъпателна разработка в кратки фиксирани итерации (Kroll & MacIsaac, 2006; Anwar, 2014; Kiv et al., 2017; Ozkan et al., 2020). Жизненият цикъл се състои от четири фази с итерации – начална $^{(1+)}$, развитие $^{(2)}$, конструиране (много), преход$^{(2)}$. По-съществените особености на подхода са:

– Фазите са с итерации, при което с приоритет са високо рискови и високо стойностни елементи;

– Архитектурно ориентиран – представяне на системата чрез UML модели;

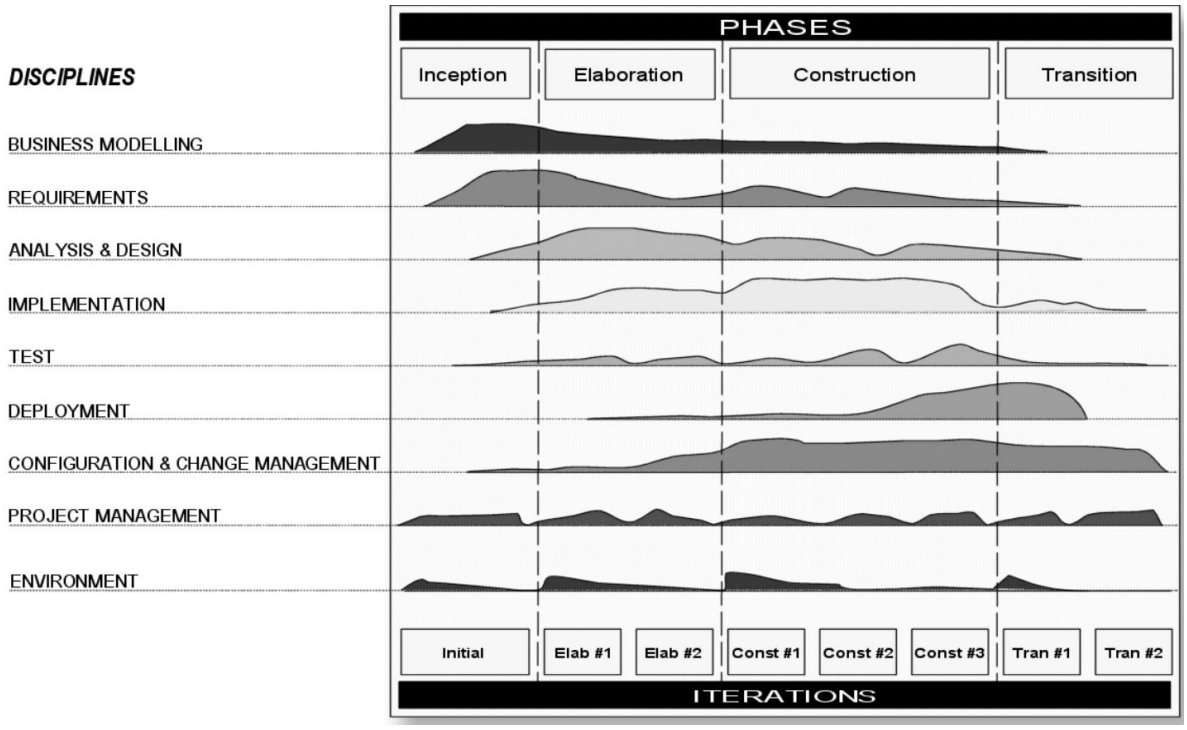
– Приспособяване на промените рано в проекта; Екипа работи заедно като цяло;

– Подходът представлява по същество фреймуърк за създаване на

[26] Подходът UP е разработен от Jacobson, Booch и Rumbaugh като базов подход, а подходът RUP го реализира в търговски продукт с шаблони за всички артефакти (информационни продукти).

индивидуален процес.

За всяка фаза се дефинира модел с ясно предназначение и набор от диаграми. Следват се сценарии за взаимодействие със системата (ако областта позволява). Началната фаза започва с визия и модел на взаимодействието със системата (ако е движен от тях процес). За развитие се използва концептуален, за конструиране – логически и за прехода – физически модел. Прецизиране на моделите в следваща фаза/итерация е възможно. Подходът дефинира 6 основни и 3 допълнителни дисциплини (дейности, потоци от работа), които работят с артефакти (информационни продукти). За всяка фаза е дефинирано приложението на дисциплини и артефакти[27]. Взаимовръзка между фази, итерации и дисциплини е дадена на фиг. 1.3.



Фиг. 1.3. Фази, итерации и дисциплини в UP/RUP (Anwar, 2014).

[27] Артефактите (дефинирани са около 50) са информационни продукти. UP и RUP предвиждат ползване на визия, списък рискове и още няколко, но само минимално необходимите. Екипът подбира за конкретния проект кои да са те.

Подходът UP/RUP предвижда използването на практики като съгласувана архитектура, визуално моделиране за комуникация, придържане към използване на компоненти, управление на изисквания и промени, постоянна проверка на качество и др., включително от други методологии (напр. за работа в екип и тестване). Планирането в подхода е на две нива – общ проектен план и итеративен план за всяка итерация. Подходът UP може да се използва от всички видове софтуерни проекти, тъй като представлява среда с множество опции, от която се прави индивидуален процес за проект или организация (Larman, 2011).

Подходът **SCRUM** (Systematic Customer Resolution Unraveling Meeting) е създаден като опростен подход, подходящ за различни видове проекти. Най-характерното за него са т.нар. „ценности", които са: ангажираност с цели, отговорност и автономност за изпълнение, фокус на екипа, отвореност и прозрачност на проект, екипна отговорност, респект, кураж и доверие на ръководството. Подходът е гъвкав, при което самият екип избира нивото (минимално възможно). Счита се, че оптимални резултати се получават когато размерът на един екип е до седем човека и работи в едно общо помещение.

Множество екипи могат да работят по един софтуерен проект и представителите им да осъществяват ежедневни срещи. Състои се от четири фази – планиране, организиране (staging), разработка и пускане (release). Във фаза „планиране" се установява визията, очакванията, подсигурява се финансиране, определят се и се оценяват изисквания (Product Backlog items). Във фаза „организиране" се идентифицират изискванията, подбират се приоритети за първа итерация и се подготвят архитектура и прототипи. Във фаза „разработка" се изгражда системата в серия от 30 дневни итерации (sprints). Клиентът работи с екипа и дава списъка с изисквания и приоритета им, а програмистите работят по изисквания в списъка за итерация (Sprint Log) по приоритет.

Всяка итерация започва с планиране, дефиниране и оценка на работата по задачи в нея (Sprint Backlog). Следва работа по тях, но не се добавя работа в започната итерация. Ежедневно се започва с 15 минутна среща (scrum) за разясняване на статуса. Екипът разработва и тества ежедневно с интеграционен и регресионен тест. В края на итерацията се прави преглед с реална демонстрация пред клиента (Sprint Review). Фаза „пускане" включва внедряване, документация, продажбени дейности. Може да се комбинира с подход UP – SCRUM е подобрен вариант на UP и XP (Maximini, 2015; Larman & Vodde, 2016).

Подходът **XP** е създаден като лека методология за малък екип (до 10 човека), разработващ софтуер при неясни или бързо изменящи се изисквания. В основата на подхода са залегнали пет основни принципа – бърза обратна връзка, простота, инкрементални промени, ангажиране с промените и качествена работа; и 10 допълнителни принципа. Практиките, които работят само ако се прилагат в комплект, са (Mangalaraj et al., 2009; Stellman & Greene, 2014; Sohaib et al., 2019):

– Планиране – определяне на реализация на база бизнес приоритети и технически възможности;

– Малки реализации в рамките на 1-2 месеца;

– Метафора – изисквания се описват с проста аналогия с цел разбираемост от всички;

– Просто проектиране – винаги системата да има прост проект, без сложности;

– Тестване – модулни тестове преди кодиране, а клиентът прави приемни тестове;

– Рефакторинг – преструктуриране към простота, гъвкавост и без излишества;

– Програмиране по двойки – двойка програмисти пише код на един компютър;

– Колективна собственост – всеки може да променя всеки код;

– Непрекъсната интеграция – при завършване на една част, тя се интегрира веднага;

– 40-часова работна седмица – нормална работна седмица, без претоварване;

– Клиентът при разработчика – компетентен клиент работи с екипа ежедневно;

– Стандарти за кодиране – код се пише съгласно добре дефинирани правила.

При подхода XP екипът работи заедно в една стая, с пряка връзка помежду си и с представител на клиента на място. Провежда ежедневни срещи със специални въпроси. Ролите и отговорностите са сравнително добре разпределени: „клиентът“ дава изисквания, приоритет, разяснява, приема продукта с тест; „мениджърът“ изпълнява организационни задачи, осигурява екип и ресурси; „водачът/треньорът“ е комуникативен и компетентен, с фокус над правилния процес и работа; „проследяващият“ има грижа за метрики, срокове, проследява изпълнението на задачите; „програмистът“ работи в двойка и редува тестване, кодиране и рефакторинг[28]. Процесът е в два вида стъпки – реализации[29] и итерации.

В реализациите клиентът задава желан набор от функционалности, поставя приоритет и подбира за изпълнение според приоритет и оценки на екипа. Реализацията се разделя на итерации със стриктен срок за изпълнение, при които итерации програмистите определят малък брой истории и задачи за изпълнението им. След приключване на реализация, клиентът проверява с приемни тестове продукта.

Подходът **DSDM** (Dynamic Systems Development Method) е създаден

[28] Рефакторингът представлява процес на „почистване” на програмния код чрез премахване на дублиращ се код и редактиране с цел програмният код да бъде лесен за четене и поддръжка при следващите итерации.

[29] В литературата се използва и думата рилийз (от release), но по своята същност това представлява итерация с доставка, т.е. издаване на работещ продукт в края на итерацията. Затова, според нас, е по-правилно да се използва термина "реализация".

в опит да се определи обща стандартна рамка за доставка на софтуерни продукти, базирани на разбирането за бърза разработка на софтуер (RAD – Rapid Application Development). Подходът се използва като база за управление, изпълнение и мащабиране на софтуерни проекти. Ключови принципи са: бизнес потребности, активно включване на потребител, вдъхновени екипи, честа доставка, интегрирано тестване и сътрудничество с възложител. Изискванията се планират на високо ниво при старта с приоритет в степените (схема MoSCoW) – „трябва да има" (M), „би трябвало, ако може" (S), „може, но не е критично" (C), „може да се отложи" (W). Критичните изисквания се приключват задължително в кратки фиксирани итерации (Plonka et al., 2014).

Подходът **FDD (**Feature-Driven Development – разработка, движена от характеристики) използва модели с кратки итерации за всяка характеристика (малък, но много полезен, „в очите на клиента", резултат). При подхода FDD се започва с разработка на цялостен модел и следва серия от двуседмични итерации. Основната идея е да се проектира и разработва с дейности: преглед област, дизайн, проверка на дизайн, кодиране, тест на код, интеграция. Практиките са модел на областта[30], собственост на код, екипна работа, инспекции, управление на конфигурации, регулярно интегриране, цялостно тестване, видимост на прогрес и резултати. Подходът е подходящ при работа в големи екипи (Mahdavi-Hezave & Ramsin, 2015).

Подходът **LSD**[31] е разработен като адаптация на принципи и практики от движението Lean Enterprise за фабричното производство и разработката на продукти в разработката на софтуер (Farid et al., 2017). Основната концепция на подхода е, че с определени принципи и

[30] В литературата има редица примери за успешно прилагане и на подход DDD - Domain-Driven Design (Evans & Szpoton, 2015) при разработка на системи.

[31] Терминът „продуктивен" е най-подходящ, защото ключовото е доставка на най-голяма ценност за клиент.

инструменти екипът трябва да се концентрира върху доставяне на „стойност” (удовлетворени ценни потребности) за потребителя във възможно по-кратко време и високо качество, което се постига чрез ефективен „поток на стойността”[32]. Основните принципи са следните:

– Елиминиране на загубите – премахване на седем пречки за ефективен „поток”;

– Впускане в обучение (процеса има експериментиране, изобретяване, учене с опит)

– Вземане на решения възможно най-късно, когато са необходими и с по-нисък риск;

– Доставка възможно най-бързо – бързо изпълнение на продукта в итерациите;

– Вдъхновяване на екипа – подходящ нов стил на работа с екипа;

– Вграден на интегритет – добър съгласуван външен и вътрешен дизайн на продукт;

– Поглед над цялото – цялостен поглед и оптимизация на системата, а не частични.

Подходът дава насоки и 22 инструмента за тяхното постигане. По-важни сред тях са: елиминиране на загубите (седем специфични типа) и построяване на карта на „потока на стойността”, изобразяваща какво трябва да се свърши/постигне, за да се получи продукта. Използва се жизнен цикъл от гъвкавите подходи – начални етапи по проучване, одобрение на проекта, предварително проектиране (архитектура), след това итерации за сегменти от продукта (всяка с проектиране, програмиране, тестване и преглед с клиента на сегмента), и накрая – внедряване.

Задължително „потокът” трябва да се „тегли от заявки на клиента” с

[32] Терминът е Value Stream Map, идващ от производството и в контекста на настоящото изложение можем да го приемем като жизнен цикъл на продукта.

акцент над ефективност и скорост в потока от работа при разработка. За тази цел се използват инструменти: обратна връзка с клиента, приоритети, синхронизация, наблягане на емпиризма и „ученето" в процеса на разработка, и т.н. (Poppendieck & Cusumano, 2012). Фокусът при вземане на решения е към подход за решения за сложни продукти с множество опции, ориентиран към личностите и малките екипи като бърз и ефективен подход. Специално внимание се отделя на проектирането, вдъхновяване на екипа, подобрение на процеса и др.

В литературата са описани и множество **други гъвкави подходи**, като: „Адаптивна софтуерна разработка на Хайсмит" (Highsmith, 2013), подходящ за големи производители на среди за разработка (например Microsoft); подходи за ускорено разработване и т.нар. „слято разработване" (Boyer et al., 2021); подход „Разработване с прототипи" като алтернатива на структурния/каскадния модел, с цел намаление на ефекта от грешки в проектирането (Tanvir et al. 2017); подход „Разработване с участие на потребителя", чрез който посредством съвкупност от техники и целенасочени действия се осигурява активното участие на потребителя във всички фази на процеса; подход „Разработване с мултиплициране", който залага на използване на съществуващи софтуерни елементи (идеи, алгоритми, модели и най-често софтуерни компоненти) в нови софтуерни продукти; подход „Разработване на надежден софтуер" за разработване на софтуер с висока критичност, при който целта е минимизиране на дефекти чрез детайлни формални спецификации, проектиране с капсулиране (само необходимо за задачата), механизми за осигуряване на качество със систематична верификация и валидизация, стриктно тестване след тях, следене за опасни конструкции и други.

За нуждите на технологичните стартиращи компании по-подробно по-горе са разгледани само по-важните подходи, които според нас, са и по-подходящи за малки екипи.

Изводи и обобщения към първа глава

1. От изследване на темата за същността и управлението на технологичните стартиращи компании, мястото им в управлението, направеният исторически и теоретичен обзор можем да заключим, че въпросите, свързани с управлението на технологичните стартиращи компании, са съществени.

2. Въз основа на стандартно известни организационни структури, практики за координация и комуникации, можем да предложим подходящи варианти за организиране на стартираща софтуерна компания.

3. Можем да заключим, че посочените методологии за разработка на софтуер покриват повечето стъпки на общия предприемачески процес. Изборът на методология, модел и решения следва да се прецени според конкретни обстоятелства (проблем, ресурси, мисия на предприемача, продукт и пазар, рискове). Възможни са комбинации с основен процес от изброените и за конкретни елементи – методи, инструменти и показатели от други подходи, така че постъпково с разработка на идея, бизнес модел и софтуерен продукт да се достигне до стандартните етапи на управление на бизнес.

4. Стартиращите технологични компании имат специфична дефиниция, цел и трябва да разработят както собствен продукт, така и адекватен бизнес модел, за да докажат жизненост и да постигат растеж.

5. Предприемачите е необходимо да бъдат подпомогнати с обучение, ментори и софтуерна система, така че да намерят решения на типични проблеми в началните етапи на развитие на организацията.

6. Софтуерната система на технологичната стартираща компания трябва да включва компоненти, които отговарят на следните основни изисквания:

– да осигурява необходимите информационни ресурси;

– да е адекватна на избрания подход за производство и поддръжка на софтуер;

– да следва съвременните тенденции в тази насока;

– да се интегрира със специфични средства за производство и поддръжка на софтуер;

– да се интегрира с другите компоненти на софтуерната система на технологичната стартираща компания.

Глава 2. Софтуерна система за управление на производството на софтуер в технологични стартиращи компании

2.1. Концептуален модел на софтуерната система

2.1.1. Основни бизнес процеси и дейности при системата за разработка, поддръжка и съпровождане на софтуер

Системата за управление на производството на софтуер в технологични стартиращи компании е необходимо да обхваща всички аспекти на управлението, където информацията за състоянието на процесите е от съществено значение за постигане на целите или правно задължителна. За малките компании са характерни по-ниската степен на формализация, опростени процедури и правила. За целите на нашето изследване приехме ограничението, че се разглеждат технологични стартиращи компании, чиято основна дейност е разработка на софтуерни продукти и тяхната поддръжка. В резултат на проучванията на световния опит при управление на процесите за разработка на софтуер в първа глава, считаме за особено подходящи т.нар. „леки методологии” с акцент над реализация на т.нар. „ценен продукт към клиента”, когато се работи с малък екип и ограничени ресурси. Затова считаме, че проектът на софтуерна система трябва да е концентриран само върху основната дейност – производство и поддръжка на софтуерни продукти, както и най-необходимите детайли за нея. Останалите части от общата софтуерна система на технологична стартираща компания може да се реализират с готов модел или продукт на сравнително ниска цена и лесно внедряване.

От бизнес гледна точка, при концептуалния модел се залага схващането, че компанията разработва софтуерни продукти, независимо дали са за клиент или собствени. Софтуерните продукти се реализират

посредством работа по индивидуални проекти[33], като всеки продукт има отделни варианти или версии. Всяка версия се разработва в отделен проект, който е част от жизнения цикъл по разработката и съпровождането на цялостния продукт. Изискванията и други данни могат да се дефинират и преизползват в отделни проекти, като са валидни за конкретна версия на продукта.

При това положение, функционирането на софтуерната система е насочено към подпомагане на съвместната работа на екипа, което включва дейностите по управление на: продукт, екип, проект, изисквания и характеристики, клиенти и оферти/договор за проект, процес на изпълнение на проекта и участие на екипа, работа по етапи и итерации. Допълнително може да се обхванат в детайли и дейностите по: създаване на работна документация, извършване на тестове, осигуряване на обратна връзка и отразяващо подобрение, планиране и отчет на усилията в проекта, разход на ресурси, отчети за статус на проект и итерация, прогнози за завършването им, отчети за завършен продукт и ефективност (на усилията, на произведен продукт и очакван/постигнат резултат).

От административна гледна точка предвиждаме системата да поддържа различни роли потребители, които имат индивидуални права за достъп до отделните компоненти. Потребителите на системата може да получават уведомления за събития по ел. поща и лична лог страница с директни връзки, като всеки елемент може да се свързва към външни документи или ресурси.

Важен момент са функциите за поддръжка на продукт, като дефиниция на условия, следене на заявки от клиенти, дефекти и резултати от регулярен мониторинг. Те се превръщат в характеристики за обновяване на продукта, задачи за изпълнение или дефекти за поправка. За членовете

[33] Това съображение е на база факта, че успешен продукт води след себе си подобрения и поддръжка.

на екипа е необходимо да се генерират справки относно назначените им задачи/дефекти, като внедрените версии се отчитат с номер версия и кратко описание на включени промени. Считаме за удачно системата да предоставя функционалност за отчети на ефективността на поддръжката и удовлетвореност на клиентите.

Относно приоритета на свойствата приемаме, че с най-висок приоритет са полезността на софтуерната система за управление на процеса и нейното лесно използване. Допълнително приемаме, че приоритета на свойствата „качество на процес", „бърза доставка" и „гъвкавост в дизайн" е по-висок от свойствата „представяне", „скорост", „интерфейс" и други маловажни за технологичните стартиращи компании.

По отношение на обхват на софтуерната система, приемаме че разработка на софтуер трябва да покрива определен набор от функционалност, като се избягва голямата сложност и влизане в чисто технически въпроси относно проектиране, програмиране, тестване и интеграция, управление на документи, конфигурации и т.н. За тях може да се ползва специализиран софтуер по индивидуална преценка. Необходимо е да се включат само базов набор функции и инструменти, така че потребителят да има само най-необходимото, за да достави полезен продукт на клиента и да изпълни клиентски проект успешно. В същото време, е необходимо да съществува възможност да се ползват шаблони – за процес, за ценност, за риск, други – с цел да се реализира специфичен индивидуален процес.

От така направения обзор, според нас, на концептуално ниво софтуерната система е необходимо да обхваща следните области, описващи нейния обхват:

1. Продукти – разработвани продукти, техните цели, кратко описание – списък основни версии на продукта – вариант, номер, особености;

2. Проекти – проектите, които се разработват:

– кратко въведение и описание на цели; идентификатор на продукт/основна версия;

– ограничения и условия, срок, бюджет, особености;

3. Изисквания – изисквания / характеристики за проект:

– йерархична структура и тематично групиране на характеристики;

– актуален статус вследствие на промяна или изпълнение в продукт;

– подреждане (ранжиране) за стойност/риск на характеристики по индивидуална скала;

– подреждане, оценка и работа в задачи с характеристики от едно ниво[34];

4. Клиенти, оферти и договор за проект:

– възложители и външни участници (възложител, потребител);

– оферти/планове с параметри и планирани усилия в различни варианти;

– договор и параметри на проект – качество, срок, цена, условия, др.;

– заявки за промени и фактически промени на параметрите на договор;

5. Процес – формат на изпълнение на проекта и участие на екипа:

– шаблони на процес с етапи, итерации и дефинирана документация;

– етапи на процеса, работа по тях, документация;

– реализации и итерации във времето; характеристики и задачи в тях;

– подреден (ранжиран) списък рискове (за включени характеристики в една реализация);

– участие на екипа с роли, персонални задачи, начални и текущи

[34] Характеристиките са в йерархична структура, но „единица” за оценка, разпределение, възлагане и изпълнение в задачи, и отчетност, са характеристики само от едно базово ниво.

оценки;

– обратна връзка от клиент, заявки, пречки, дефекти, приемни тестове;

– отразяващо подобрение – обратна връзка, мерки, изводи, задачи;

6. Планиране и отчитане на вложен труд и разходи:

– оценяване на вложен труд в идеални и реални мерки, и варианти – например: идеални точки, очаквани часове работа, най-вероятна и сигурна оценка[35];

– отчет за съотношение на планирано време и разполагаемо в екипа;

– коригиране на оценки по характеристики в реализации на проекта;

– текущо регулярно отчитане на оставащо време по задачи;

– лично отчитане на усилията по задачи и характеристики;

– опростено отчитане на разхода на финансови и времеви ресурси;

7. Отчети за статус на проект и итерация:

– лично табло за персонална информация и статус според роля;

– статус на проект/реализация/итерация с графика за очаквано завършване[36];

– светофарна сигнализация за проблемност на проект/итерация;

– отчети за оставаща работа –задачи, проблеми, дефекти, тестове и др.;

– отчети за завършеност – списък с изпълнение на характеристики, списък със следващи очаквани реализации и демонстрации с дата, и др.;

8. Отчети за ефективност и точност:

– ефективност на усилията – възвръщаемост „% стойност” към „% усилия”;

---

[35] За по-реалистичен план, оценката може да е с избрана мярка за сложност и в идеално време очаквана работа. За целта може да се направят най-вероятна и песимистична оценка (включваща проблеми/преработка).

[36] Очакваното завършване е отчет на база текущи оценки за оставаща работа в една итерация, реализация или проект, като включва прогноза за очаквано завършване с оптимистичен, линеен и песимистичен вариант.

– степен на преработване на характеристиките;

– точност на преценките по задачи и характеристики, лични и екипни;

– финансов отчет за разходи по проект/продукт и ефективност;

– съотношение на проектен срок и бюджет с актуални/прогнозни данни;

9. Екип – екипът, с които се разполага, участието на членовете му в проекти и достъп до софтуерната система:

– налични специалисти, уменията им, тарифни ставки, история;

– работно време и параметри, данни за лична преценка на усилията;

– присъединяване към проект с определена роля;

– отворени възможности за автономна работа на екипа;

– уведомления чрез e-mail с директни линкове и личен лог съобщения;

10. Достъп до системата:

– поддържане на различни роли с различни права за достъп;

– наличие на роля с пълен достъп и роля с административни задачи;

– персонализирани екрани и достъп на база ролята в проекта;

– програмен достъп с API до компонентите за лесна настройка;

11. Документация и работни продукти:

– дефиниран минимален пакет шаблони работни продукти за подхода;

– работни елементи с линкове към външни документи или ресурси;

– възможно е използването на външна система за управление на конфигурациите;

12. Поддръжка:

– описание и структуриране на изисквания, условия и мониторинг;

– следене на заявки от клиенти и резултати от регулярен мониторинг;

– конвертиране заявка/резултат в дефект, характеристика и задача

за обновяване;

– внедрени версии – номер версия и кратко описание на направени промени;

– отчети за ефективност на поддръжката и удовлетвореност на клиент.

2.1.2. Информационно бизнес моделиране на системата

Така представеният обхват от желана функционалност представлява концептуалната рамка на софтуерната система. По отношение на проектирането приемаме за съществени компоненти следните дейности: детайлизиране на изисквания, анализ и проектиране на информационен модел. Тъй като системата се базира на работа с потребители, според нас е подходящо да се използва архитектурно центриран модел, базиран на сценарии за взаимодействие. Затова е разработен бизнес модел (БМ) на системата и се придържаме към утвърден подход чрез използване на профил за информационно бизнес моделиране (ИБМ), базиран на универсалния език за моделиране и описание на софтуерни системи UML като признат международен стандарт за обектно-ориентиран език за модели[37]. За целите на ИБМ, според нас, е подходящо да се използва UML с разширение чрез специализация с профил на IBM RUP за бизнес моделиране (Fellmann et al., 2018; Beynon-Davies, 2018; Musulin & Strahonja, 2018; Sadowska & Huzar, 2019; Wirtz, 2019). Следвайки този подход, може да се открояят следните елементи за софтуерната система:

1. **Бизнес сценарии за взаимодействие със системата (БСВС)** – отразяват цел и обхват на бъдещата софтуерна система, взаимодействие с потребители и какви услуги осигурява.

[37] UML има възможности за детайлно описание на взаимодействие с външни партньори и системи, както и за вътрешна архитектура и поведение на ИС. За целите на ИБМ използваме предимството на UML за разширение чрез специализация на същия метод и елементи в желаната област на моделиране.

2. **Концептуален бизнес модел (КцБМ)** – показва начина, по който служители и обекти, които те управляват, се свързват (статично и динамично), за да се реализират съответните БСВС.

3. **Обектен бизнес модел (ОБМ)** – част от КцБМ, отразяваща само бизнес същности (характеристиките и връзките им) и не включва отговорности на бизнес работниците.

При разработката на модела се ограничаваме единствено над основния БСВС. В него участници са описаните по-горе роли потребители, а описанието на **бизнес сценария за взаимодействие със системата** относно разработката на софтуер е в следните стъпки:

1. Описание на продукт – вътрешен възложител, координатор или главен проектант описва продукта или негова версия с неговата визия.

2. Дефиниране на проект – вътрешен възложител или координатор дефинират проект, условия, ограничения, характеристики и изисквания. Те също дефинират шаблоните, които ще се ползват в проекта за процес, оценка и т.н. Възложител или потребител дефинира списък със сценарии или характеристики.

3. Планиране на проект – възложител ранжира характеристиките; главен проектант и екипа[38] определят вложения труд за тях и ги съобразяват с капацитета. Получава се оферта и се дефинира договор/план с параметри.

4. Организиране на разработка – главен проектант/координатор дефинира процес по избран шаблон, екип, етапи, реализации. С помощта на екипа се дефинират първи итерации и задачи, списък рискове, и се разпределя времето.

5. Планиране на реализация – главен проектант/координатор с възложител актуализират плана с реализации, разпределят характеристики

[38] Има се предвид само екип от компанията – главен проектант, програмисти, тестери, редактори, координатор;

по реализации, добавят и оценяват нови характеристики. Екипът описва итерации, задачи и преценки за тях, списък с рискове, необходима документация и други.

6. Работа по задачи – екип на компанията работи по задачи, ползва лично табло за статус, променя статус на задачи, отчита разход на време и средства, дефинира нови задачи, проблеми и дефекти, поема и обработва дефекти и рискове. Ежедневно записва изразходвано и оставащо време за работа по задачи, и промяна на статус. Преглежда справки за статус и очаквано завършване.

7. Приключване на итерация – координатор/екип отчита резултатите и записва изводи от отразяващото подобрение. Модифицира се плана за следващи итерации и задачи, преглежда справки за статус и очаквано завършване. Екипът коригира задачи и оценки в следващи итерации (координатор в реализации).

8. Приключване на реализация – координатор/екип отчита резултатите, оформя продукт, преглежда справки за статус, дефинира пречки, дефекти, рискове и модифицира план за реализации след обсъждане с възложител/потребители; координаторът записва резултати от отразяващо подобрение, а екипът дефинира задачи за следващи итерации. Коригира се документацията.

9. Внедряване – главният проектант/програмистите изпълняват задачи за финални корекции и дефекти, внедряване и приемни тестове. Програмистите оформят версия и внедряват. Редактор оформя документацията.

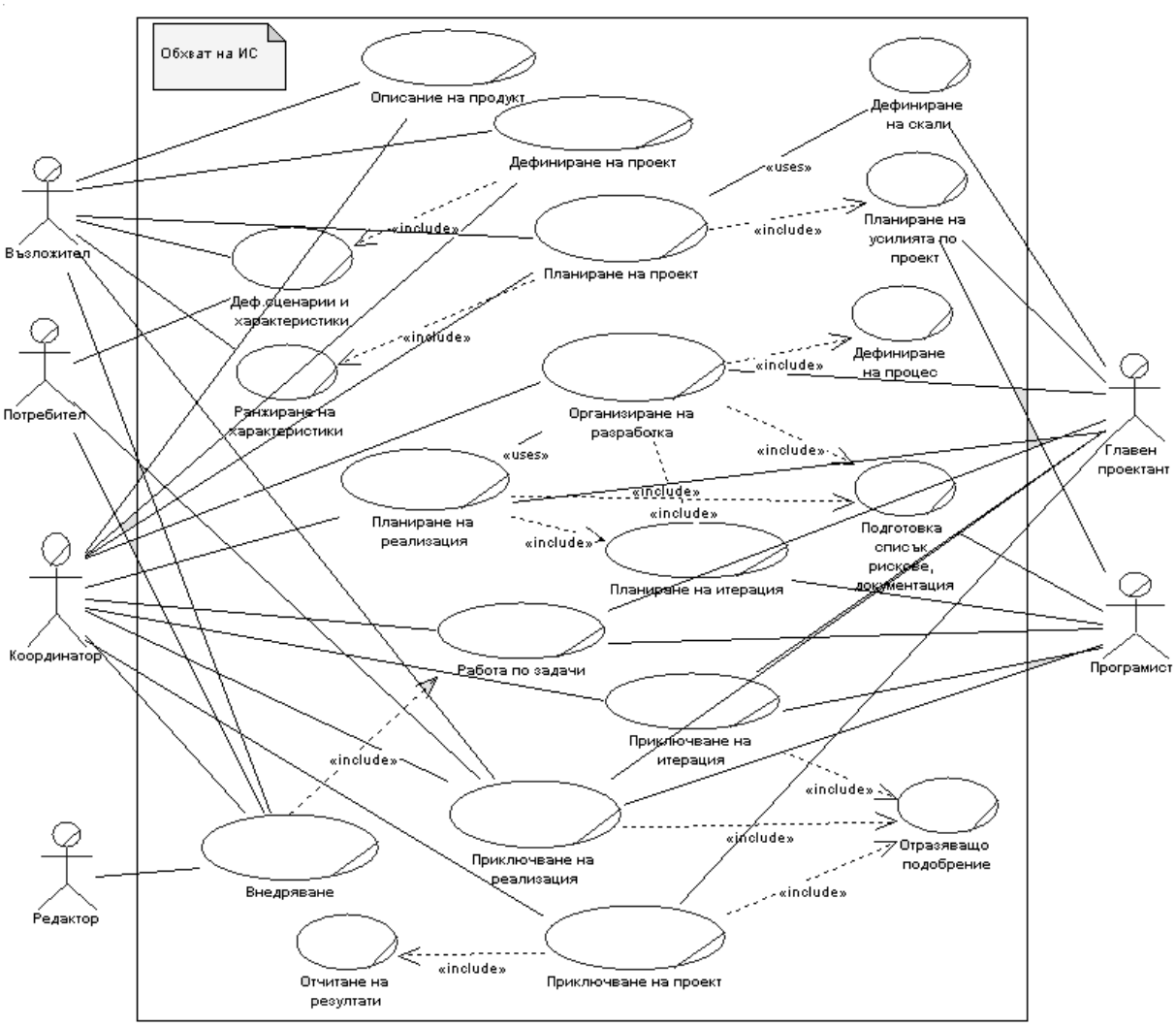
10. Приключване на проект – координатор/екип отчита резултатите, правят се отчети за ефективност и финансов резултат от проекта. Дефинират се резултати от отразяващо подобрение. Дефинират се задачи с лична отговорност в друг проект за подобрение относно подход/конвенции за разработка.

11. Общи административни задачи – координаторът работи по общи

административни задачи, настройва системата, подготвя информационни табла и други.

Така описаният главен сценарий е показан на фиг. 2.1[39]. Той описва взаимодействието между бизнес актьори и софтуерната система при разработка на софтуер. Сценариите са представени с овални форми. По-големите са сценариите в главния БСВС, а по-малките са някои интересни детайлизации към основните. Връзките между актьори и сценарии показват извършваните действия. Връзката „uses" показва, че един сценарий използва друг, а „include" показва, че един сценарий включва като част от себе си друг сценарий.

[39] При създаване на моделите в труда е използван софтуер Enterprise Architect на компанията Sparx Systems (https://sparxsystems.com/). Ползваната анотация от графични елементи е съгласно спецификацията за UML 2.5.



Фиг. 2.1. Диаграма на главен бизнес сценарий за взаимодействие „Разработка на продукт”. (разработка на автора)

Главният сценарий относно поддръжка на софтуерен продукт е следния:

1. Дефиниране на поддръжка – координатор/главен проектант дефинира изисквания, условия, мониторинг при поддръжката;

2. Следене на заявки – възложител/координатор попълва заявки от клиент; програмист извършва мониторинг и попълва заявка за некоректен резултат;

3. Конвертиране на заявки – заявките се преобразуват в дефекти, задачи или характеристики за обновяване на продукт и се назначават за програмист;

4. Опис на внедряване – координатор/програмист описва внедрени

версии;

5. Отчет за поддръжка – координатор/възложител/потребител описва мнение за удовлетвореност от внедрена версия и оказана поддръжка.

Главните сценарии може допълнително да се детайлизират в подчинени БСВС. По отношение на **концептуалния бизнес модел** (КцБМ), в него се изгражда обектен модел, представящ бизнес същностите с информационен характер. Бизнес същностите и основните им свойства са следните:

1. Продукт – разработван продукт с цел, описание и списък с версии по вид и номер.

2. Продуктова версия – версия на даден продукт с вид, номер и кратко описание.

3. Проект – разработван проект за даден продукт; описва се име, код, въведение, цел, идентификатор на продукт и версия, условия, ограничения, особености, целеви срок и бюджет.

4. Списък сценарии – работен продукт като „Actor-Goal List” или „Use Cases”, който описва действията – актьор, име, цел, стъпки, автор, пореден номер.

5. Характеристика – изискване за продукт/проект с йерархична структура, теми, статус за актуалност/изпълнение, ранг за стойност и риск, приоритет, оценки за реализации.

5. Клиент – външен клиент възложител.

6. Представител на клиент – лице представител на клиент с роля в проекта.

7. Проектен план – план/оферта/договор с условия, ограничения, качество, цена, срок, планирани усилия за характеристики и параметри за „конус на несигурност” (резерви).

8. Служител – член на екипа в технологична стартираща компания; име, история и текущи данни за позиция, умения, ставка, документация,

параметри за лична преценка на усилията, роля(и) в проекта.

9. Участник – член на екипа (служител или представител на клиент) с роли в проекта.

10. Роля – именувана роля за участник в проекта – определя персонализация и достъп.

11. Екипна структура – назначение на участник в проект и роля/роли.

12. Етап – етап от проект – име, вид, статус, планов/актуален срок (дължина, от-до).

13. Реализация – вид етап, който притежава описание на реализирания в края продукт.

14. Реализирано приложение – версия на приложение реализирано към клиент.

15. План за реализация – схема на етапи и реализации (със срок) в проекта със списък характеристики подредени за ранг/зависимост и списък реализирани приложения.

16. Оценка за усилия – оценка за усилията на член на екипа по дадена характеристика/работа. Дава се в единици от мярка за най-вероятен и сигурен.

17. Списък рискове – работен продукт за проект описващ рискове за дадена реализация и евентуално характеристика, подреден по приоритет. Рисковете са с наименование, ранг в реализация, вероятност по скала, опис на вероятен резултат, статус и връзка с елемент.

18. Итерация – итерация в дадена реализация, със срок (от, интервал от време, до), списък със задачи.

19. План за итерация – работен продукт със задачи и други елементи в итерация.

20. Задача – работа за реализация на характеристика от определен участник; има име, описание, статус, начална и текуща оценка, оценка за оставащо време за завършване.

21. Тест – приемен тест като подвид задача, която да се изпълни в дадена итерация.

22. Проблем – описан проблем в работата на софтуера – има заглавие, характеристика, тип, статус, приоритет, контекст, вход, изход, автор, отговорник и други.

23. Дефект – вид задача създадена от проблем за поправяне със статус, модул, решение.

24. Заявка – заявка/идея за продукта в даден проект, която има име, тип и описание и може да бъде превърната по-късно в нова характеристика, дефект, задача или отпада.

25. Подобрение – заключение от отразяващото подобрение към реализация/итерация. Има списък мерки с тип, описание, автор и отговорник. Може да се превърне в задача от проект.

26. Оставащо време – очаквано време на член от екипа за работа до края на задача.

27. Разход на време – отчитане на дадена дата, период от време, времето за което член от екипа е работил по дадена задача (за една характеристика) и коментар по това.

28. Календар – разполагаемо работно време в часове по дни на различни участници.

29. Финансов разход – отчита кога, кой, какъв тип, за какво и колко разход е направил.

30. Записка – бележка/съобщение от софтуерната система или участник за събитие или друго относно проект или друга същност. Има тип, заглавие, съобщение, време, статус, уведомление. Може да се ползват и като бележки за дизайн от автора на документ (работен продукт).

31. Документ – документ на външен файл, с който елемент от системата има връзка.

32. Таг – списък от индивидуално дефинирани и именувани

стойности за един обект[40].

33. Рангова скала – за подреждане на обекти за стойност[41], риск, вид, приоритет и т.н. Скалата има списък опции със свойства като код, име, ред, тежест, коментар.

34. Процесен шаблон – именуван и описан списък с етапи за даден тип процес и дефиниция на скалите, които се ползват. Етапите имат име, дължина на срок и описание.

В компонента „поддръжка на продукт" са налични множество сходни дефиниции на същности както в компонента „разработка на продукт". Характеристиките са изисквания в опростен линеен списък с теми, заявка и тест, и имат някои разлики. Различни **бизнес същности** са следните:

– Мониторингов тест – тест за регулярна проверка на продукта;

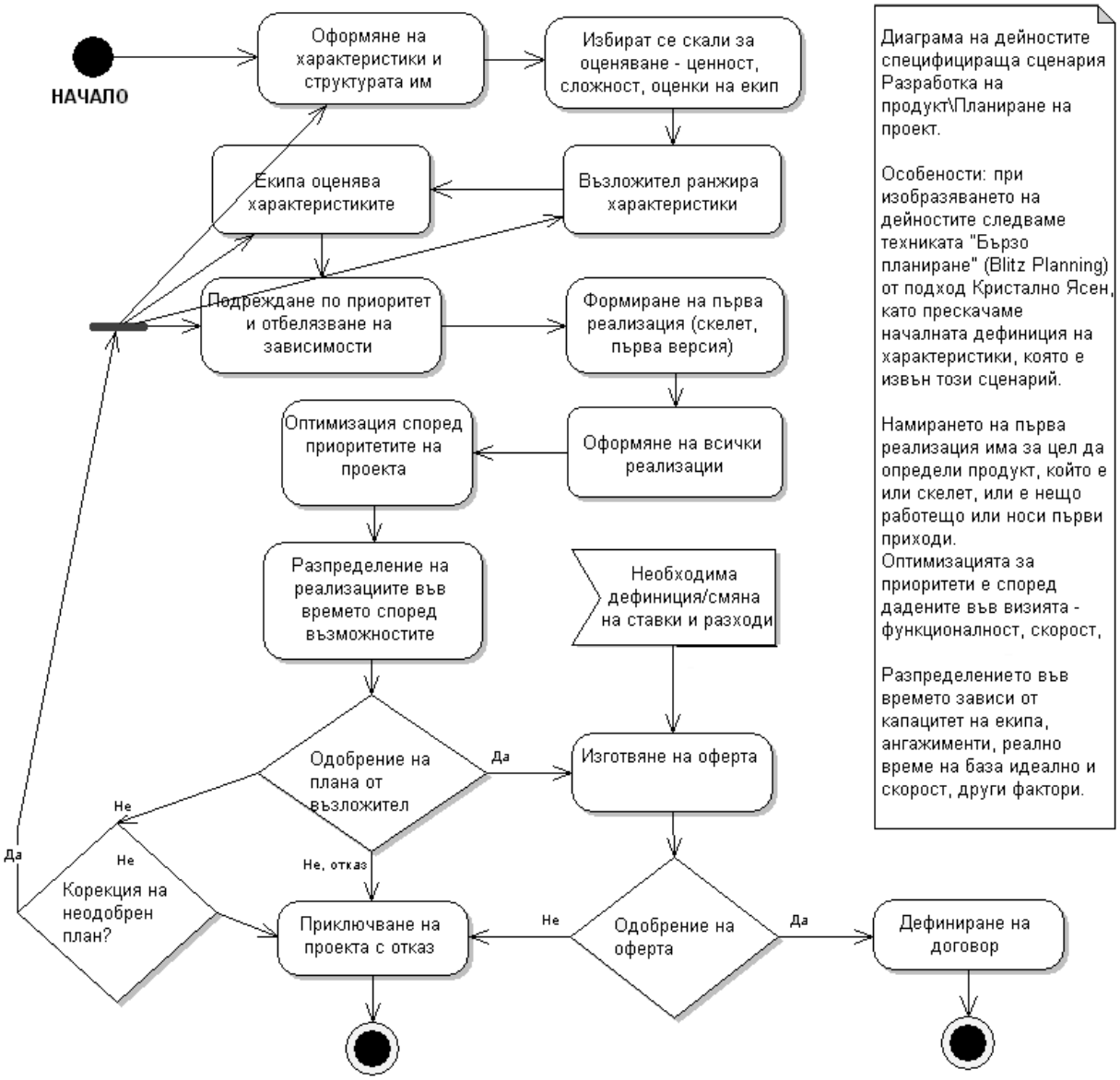
– Заявка за поддръжка – заявка за необходима промяна/корекция/проверка за нередност – може да се конвертира в задача/характеристика/дефект.

По отношение на **реализацията на БСВС** са разработени диаграми на взаимодействията (на последователност) и колаборативни диаграми, с които се показва взаимодействието между бизнес работници (с роли в технологичната стартираща компания – координатор, главен проектант, програмист, редактор, тестер), и рамките на техните отговорности. Техните отговорности са основа за формиране на функциите на системата и персонализираният достъп за всяка роля. Бизнес актьорите са външните участници – възложител и потребител.

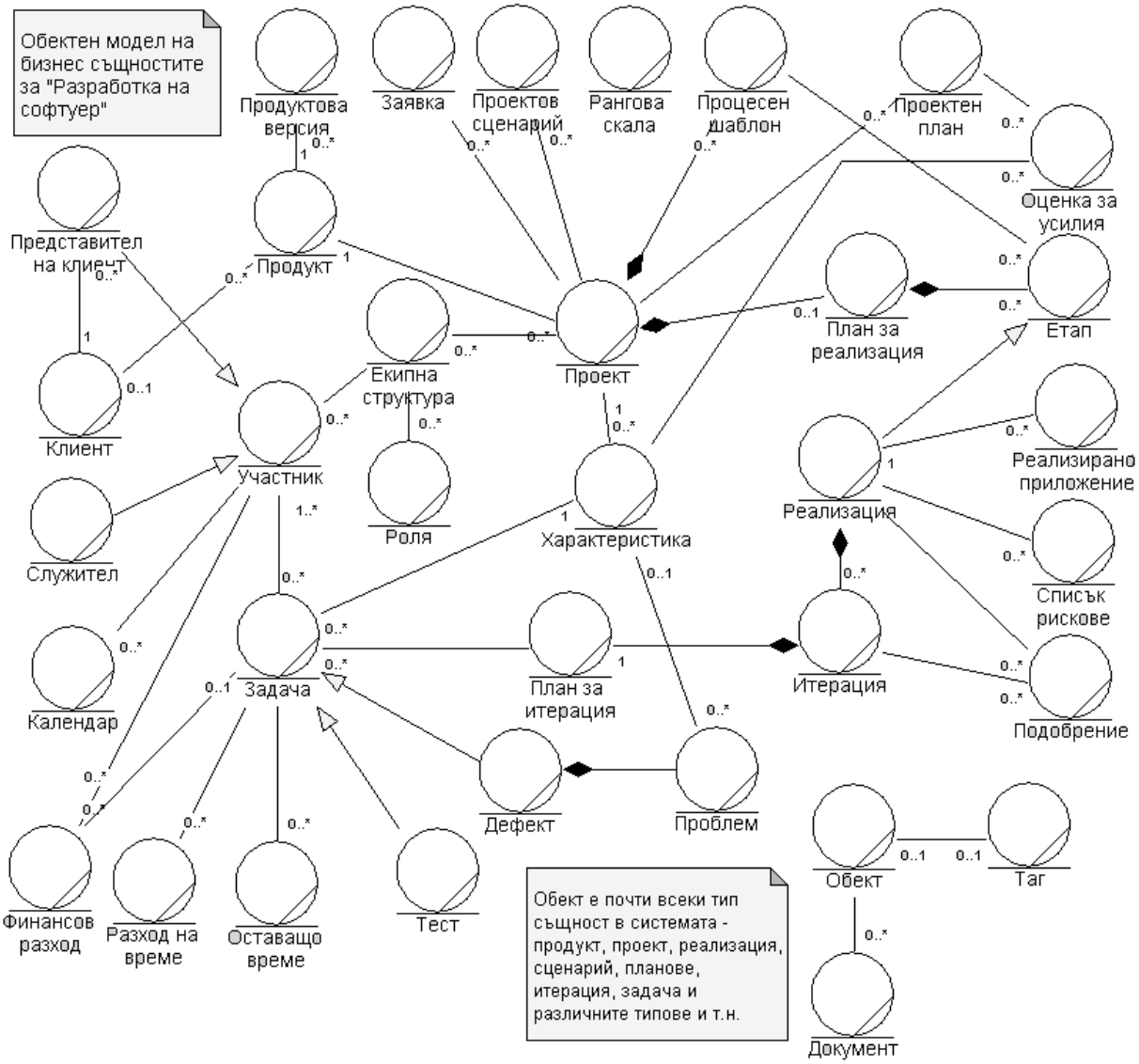
На фиг. 2.2 са представени детайлите в отделни бизнес сценарии с диаграма на дейностите и е показано описването с дейности на даден

[40] Това е структуриран (XML, named pairs) списък от нестандартни данни с име и стойност. Потребителят/ИС ги добавя за/от интеграция с външен продукт или допълнителни записки.
[41] За рангова скала за ценност на характеристика е подходящо да се използва например HML (high, middle, low), MoSCoW от DSDM, и др.;

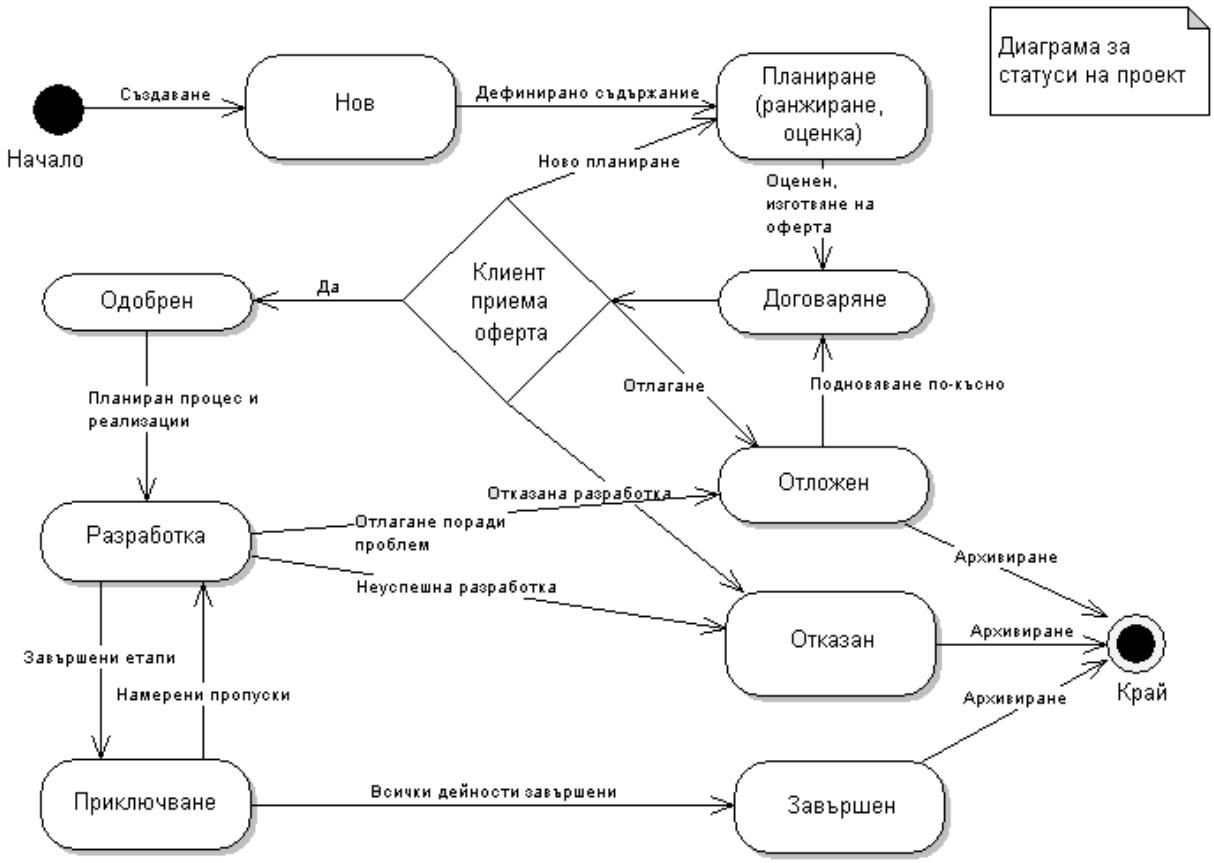
сценарий. Детайлното представяне на бизнес същностите в обектния модел е дадено на диаграмата на фиг. 2.3. Тя показва и връзките между тях. За да се отразят състоянията на тези бизнес същности и събития, при които те се променят, е използвана диаграма на състоянията. Примери относно компонентите „състояния за проект” и „реализация” са дадени на фиг. 2.4 и фиг. 2.5 чрез диаграма за възможни състояния на същност „Проект” и събитията, при които тези състояния могат се променят. Тези диаграми са основа и за изработване на списъка от статуси за дадения обект, както и логиката, която е залегнала в приложението за възможните преходи.



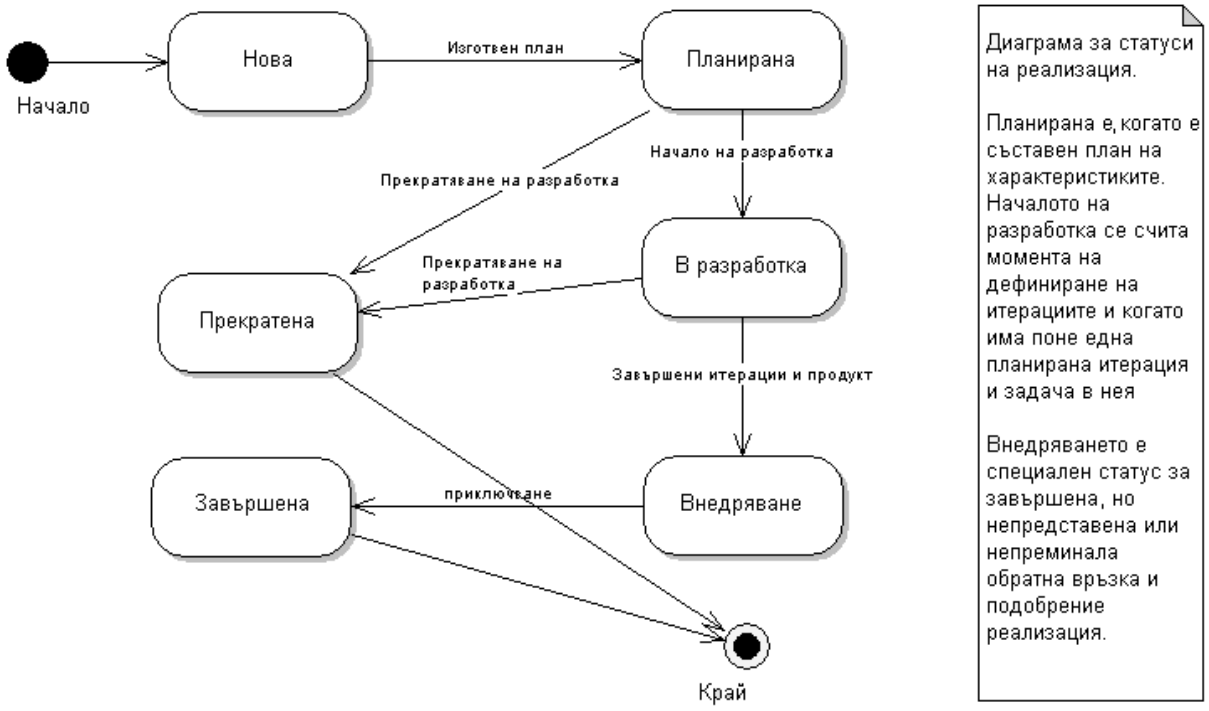
Фиг. 2.2. Диаграма на дейности за БСВС „Разработка на продукт”/”Планиране на проект”. (разработка на автора)



Фиг. 2.3. Обектен модел на бизнес същностите „Разработка на продукт" – диаграма на бизнес същности и техните връзки с други бизнес същности. (разработка на автора)



Фиг. 2.4. Диаграми на състоянията на бизнес същност „Проект”. (разработка на автора)



Фиг. 2.5. Диаграми на състоянията на бизнес същност „Реализация” (разработка на автора)

В съответствие с изложения в първа глава обзор на текущото състояние на информационните системи – тенденции за групова работа (чрез система съобразно отговорности), централизация на данни и достъп, бездокументални и интернет технологии, приемаме за изграждане на софтуерната система да се използва технология тип „клиент-сървър”. Всеки участник в процеса работи на софтуерен клиент, а базата от данни, логиката и обработката се извършват основно на един или няколко сървъра. Според нас, в случая трислойната архитектура за реализирането й е подходяща – представителен слой за потребителски интерфейс, слой за бизнес логика и слой за данни (Microsoft Patterns & Practices Team, 2009; IBM Cloud Education, 2020). В този случай на трислойна структура, софтуерната система е необходимо да се структурира в отделни компоненти, които са разположени в отделните слоеве. Детайлизация на компонентите е дадена по-нататък в изложението, като е описана композицията при реализация на софтуерната система. Реалният потребител – член на екипа, контактува само с представителния слой чрез входно-изходен интерфейс на уеб приложение.

2.1.3. Възможности за усъвършенстване на информационната база

Изхождайки от принципа, че софтуерната система е средство, с което може да работи паралелно целия екип, според нас, е необходимо да се заложи възможност за участие и достъп на клиент (негов представител) до софтуерната система. С директен уеб достъп възложител или потребител може да види реализираните версии и характеристики, готови за тестване, някои отчети и прогнози за завършване, и т.н. Адаптацията на предложения подход води до спестяване на време за комуникация на екипа в технологичната стартираща компания. При експортно ориентираните технологични стартиращи компании е от полза и наличието на

многоезикова поддръжка.

В проекта за софтуерна система е необходимо да се предвидят и автоматизация на процеси: първо, да притежава възможност за дефиниране на процес от шаблон, и второ, да са заложени правила за автоматична промяна на статус и уведомления. Правилата може да се поставят за характеристики, етапи, рискове и други, в случай, че за тях се дефинират свързани елементи – задачи, дефекти и т.н., или всички такива бъдат приключени. На следващ етап на разработка в перспектива софтуерната система е възможно да се добави функция за автоматичен начален избор на характеристики по реализации на база приоритет в стойността/риска, оценки за тях, процес и срок на проекта, и капацитета работно време.

По отношение на **пренасянето на данни и съхранението**, следва да се използват общоприети стандарти и мрежови протоколи за комуникация, както наложили се в практиката системи за управление на бази данни (СУБД) – релационни, така и от тип NoSQL. За съхраняване на работните продукти е подходящо да се разчита на външно хранилище на документи – публична или частна облачна услуга, или локална система за управление на конфигурациите, която да пази историята на промените. Възможно е прилагането и на хибриден подход. Перспективно направление е системата да има възможност за индивидуализация и промяна на списъците за подреждане (ранжиране) и шаблони за процеси. Така технологичната стартираща компания може да изработи собствени варианти на процес. За правилното им прилагане потребителят може да използва заложените в софтуерната система инструкции за употреба на шаблони и скали.

По отношение на **интегриране с други системи** – в практиката минималните очаквания на потребителите са да е предвиден експорт на данни от отчети и списъци във формат на Excel таблици. Чрез възможност за програмен достъп до библиотеки/услуги, тази функционалност може да се интегрира с външни средства. Например средства за управление на

изисквания, за тестване, среди за разработка (така че работните елементи от софтуерната система да се появят в средата за разработка на програмиста, а също и от там да се получат междинни данни за извършвана или завършена работа). Друг пример е онлайн система за заявки за поддръжка, промени и дефекти, (автоматизирано) публикуване на отчети в сайт на компанията, или интеграция с външен софтуер за управление на изисквания, управление на конфигурации, управление и отчитане на време и т.н. Така дефинираните свойства се използват/попълват, а липсващите в системата се добавят в тага в една избрана структура – XML или именувани двойки.

По отношение на **потребителския интерфейс** съществува тенденция, при съвременните бизнес информационни системи, той да е лесен за употреба и гъвкав (лесен за настройване). Поради високата динамика на смяна на устройства и парадигми за дизайн считаме, че в началния етап на развитие на софтуерната система не е удачно да се прилага високо качество на интерфейса, каквато е практиката при масовите продукти. Достатъчни е да се ползват базови възможности на средата за разработка за удобен интерфейс.

Като се отчита, че технологичната стартираща компания има малък екип, ниска формализация на подход и сложност на проекти, е удачно интерфейсът да притежава следните параметри:

– малък размер – оптимизиран за малко данни[42]/съдържание, като избягва страниране;

– простота – опростени данни/съдържание, кратко, най-необходимото, без сложност;

– стандартен формат – стандартен формат на работната площ и еднотипни екрани;

[42] Например, на база лични наблюдения и опит, можем да твърдим, че характеристиките в малък проект са обичайно 3-4 групи по 10-12 с няколко детайла. Общо до 50-100 на брой. Структурирането и разделението по реализации намалява тази бройка.

– меню с две нива – постоянно главно меню и подменю за избрана опция;

– лично табло – за секция; персонализирано; има справки/връзки за най-важното;

– лесна трансформация – лесна трансформация на елементи и смяна на статус/ред[43];

– лесен достъп до инструкции за опции, статуси, рангове;

– лесно прикрепяне на записки – записки, таг, документи лесно се задават за обект;

– избягване на мултизадачна работа – фокус над един проект/задача в един момент;

– бързо и лесно отчитане на работа и време;

– автоматичен работен филтър – по подразбиране се показват само работни елементи;

– синхронизация при екипна работа – регулярно интерфейса да е синхронизиран;

За приложението, което се достъпва от потребителите, е удачно да се използва един основен макет на интерфейс с няколко статични елемента, главно статично меню, подчинено меню, работна площ за повечето екрани и списъци с бързи връзки за най-важните, спешни и нови елементи свързани с текущите продукт, проект, функция и потребител. Детайлно този въпрос е развит в по-нататък в изложението.

Първичното отчитане в настоящата софтуерна система е ръчно от участниците в проекта. Данни се въвеждат във входни екрани от участниците лично и ръчно. На първоначалните етапи от функционирането си софтуерната система може да не предлага автоматична отчетност, както е например в eScrum (Newmark et al. 2018; Dixit & Bhushan, 2019). Това се

[43] Има се в предвид работа с един-два клика на мишката, един-два бързи клавиша и съкратено въвеждане на допълнителни данни.

компенсира с простота и удобство при въвеждането на данни. Също така може да има известна компенсация за отчитане на времето, ако се попълва лога за отчет на работното време със записки, които показват кой и кога е работил със системата и така по-лесно член на екипа може да си спомни и укаже по какъв проект и задача, кога и какво е работил.

Много от екраните за личен отчет трябва да могат да работят както с отговорен потребител, така и с координатора, за да може програмистите да са заети с работата си по същество и с по-малко административна работа. Удачно е извеждането да е само в електронен формат – на екран, експорт в HTML уеб страница или данни в Excel-файл. Изключения може да има в следните наложителни случаи:

– интеграция с външни продукти, които да правят електронен трансфер на данни;

– разпечатване на документи за възложителя, където се изисква писмена форма (оферта в търг, приложение към договор и т.н.);

– разпечатването на документи за т.нар. „информационни радиатори" съгласно подхода.

Предложения концептуален модел на софтуерна система е опростен за целите на технологичните стартиращи компании. Той подпомага компанията за ефективното управление и изпълнение на нейните проекти. Вниманието при софтуерната система се концентрира над разработката на проекти във връзка с разработката на продукти.

Функционалността на системата се различава от готовите продукти на пазара в следните направления (пропускаме разликите в базовите подходи и стриктното предназначение на предлаганата софтуерна система за технологични стартиращи компании):

– опростява работата със системата, като се избягва сложността на готови продукти като „VersionOne" (https://www.collab.net/products/versionone) и eScrum (https://www.digibelt.com/en/escrum-prod-5);

– акцентира се върху асинхронното, а не върху синхронизирано управление на множество проекти и екипи („VersionOne”, и др.);

– зависимостта между задачи е премахната – считаме това и по-горното за излишна сложност[44];

– важно предимство е простота на екраните, които наблягат на спецификата за малък обем и фокус;

– в резултат на опростяването на функционалността е избегнато използването на „работна площ” с влачене и синхронизирани екрани;

– вградено рефлексивно подобрение и мерки по отговорно следене за изпълнението им в същия или отделен проект;

– техническа възможност за интеграция с външни продукти чрез външен програмен достъп (но на този етап няма интеграция с конкретен външен софтуер или среда за разработка);

– опростени средства за прогнозиране на очакван край, като алтернатива на обширните отчети;

– отчет за ефективност както в ScrumWorks (https://www.collab.net/community/scrumworks);

– вградено е поддържане на данни за човешки ресурси и следене на промените при тях;

– изработка на оферта/план, което при други системи може да липсва;

– специална секция за продуктово следене;

– съпровождане на продукти (което имат само част от горните примери).

Предлаганият концептуален модел на софтуерна система се фокусира над разработката на софтуерни проекти във връзка с

[44] Според нас, в малък екип е излишна сложност да се отделят време и ресурси с твърде детайлно планиране на ниво итерация. Включено е планиране на зависимости между характеристиките на ниво проект, което трябва да се съобрази при разпределението на характеристиките в реализациите. В по-голям проект и екип работата може да се преструктурира, така че да се намалят проблемите със зависимостите.

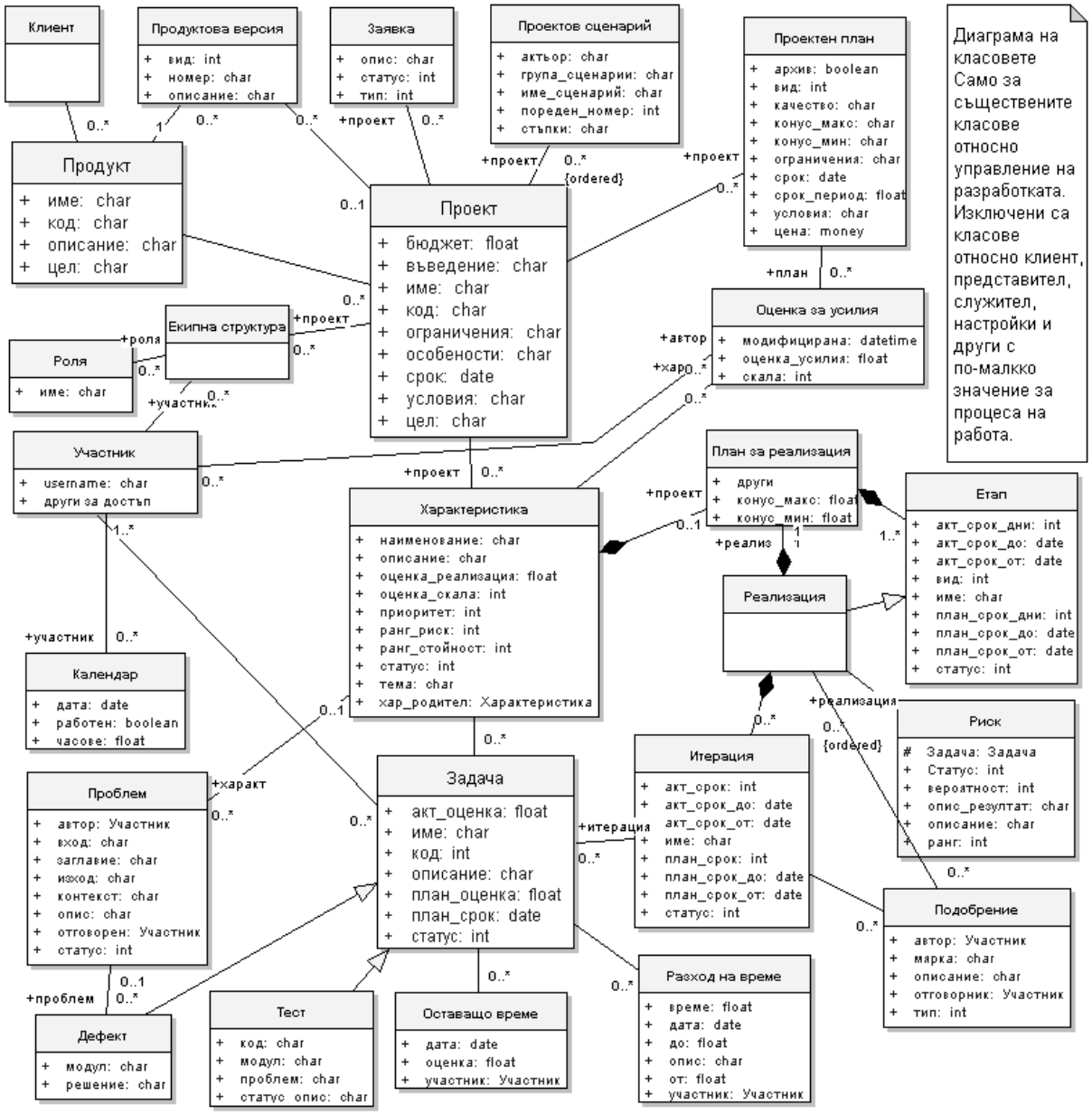
разработката на продукти, прилагайки гъвкав подход (подход за лесна адаптация към промените). Той е опростен за целите на технологичните стартиращи компании и очакваме да подпомогне компанията за ефективното управление и изпълнение на нейните софтуерни проекти.

2.2. Логически модел на системата

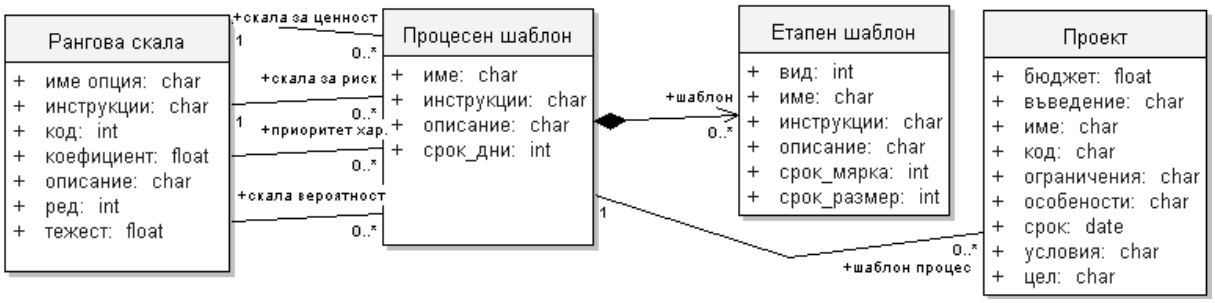
Логическият модел е разработен на базата на изложения концептуален бизнес модел в концепцията за софтуерната система. За постигането на целта, в началото следва да се обособят и развият бизнес същностите като класове със съответни свойства и връзки.

2.2.1. Диаграми на класове, реализиращи бизнес същностите

Диаграма на основните класове и техните връзки са показани на фиг. 2.6 – съществените класове за разработка, формирани от бизнес същностите, и фиг. 2.7. Под основни разбираме тези класове, които се отнасят пряко към управление на разработката. Изключени са детайли по служители, клиенти, договори, поддръжка и т.н. Добавена е диаграма за класовете, обслужващи шаблони за процеси и индивидуални скали за ценност, риск, вероятност, степен и други. Компонентите в нея помагат за настройката на предварителни макети за различни процеси. Като цяло тези и сходни диаграми са и основа за разработване на база данни за софтуерната система и скелет за класовете в бизнес логиката на софтуерната система.



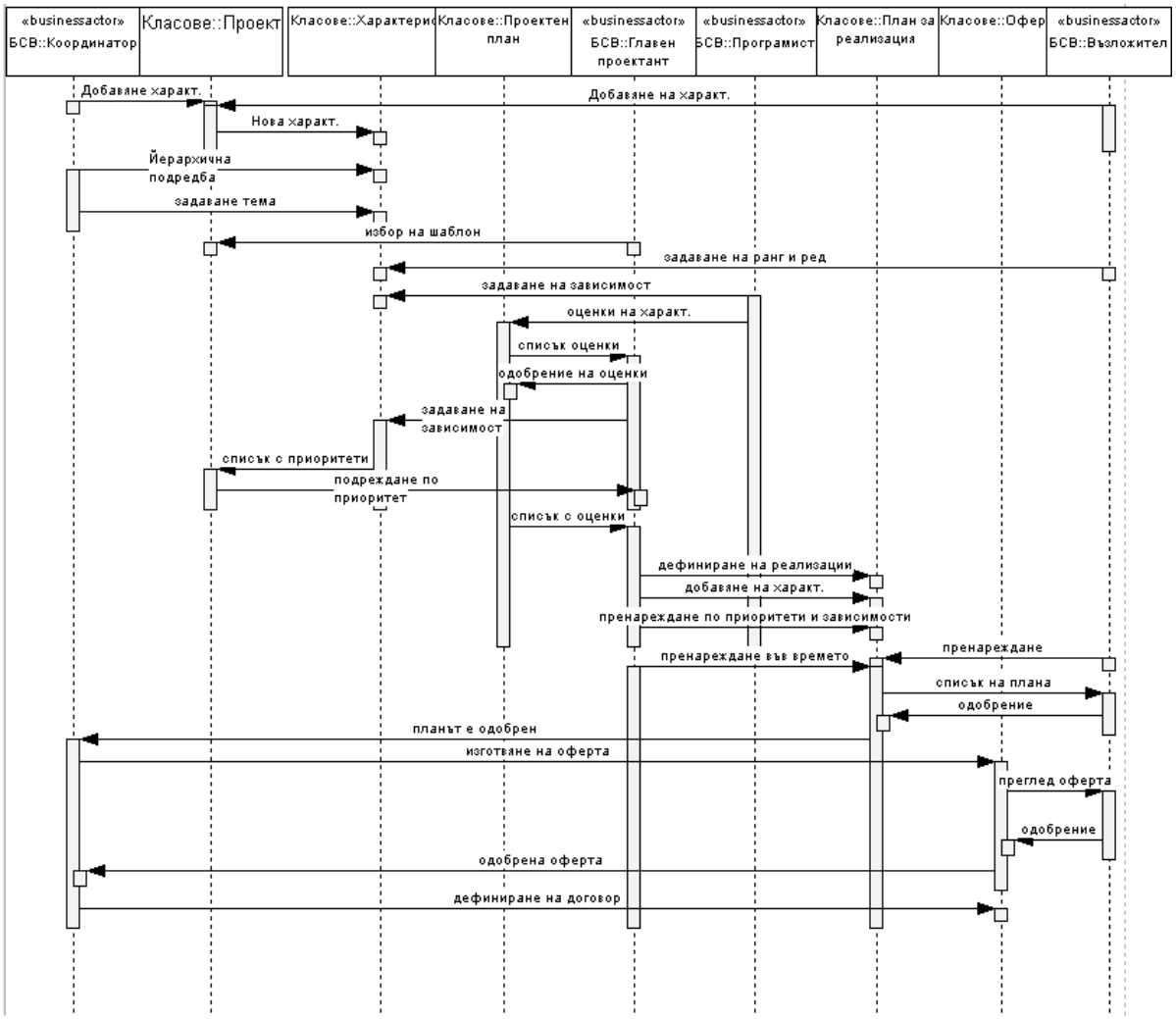
Фиг. 2.6. Диаграма на класовете и техните връзки за „Разработка на продукт”. (разработка на автора)



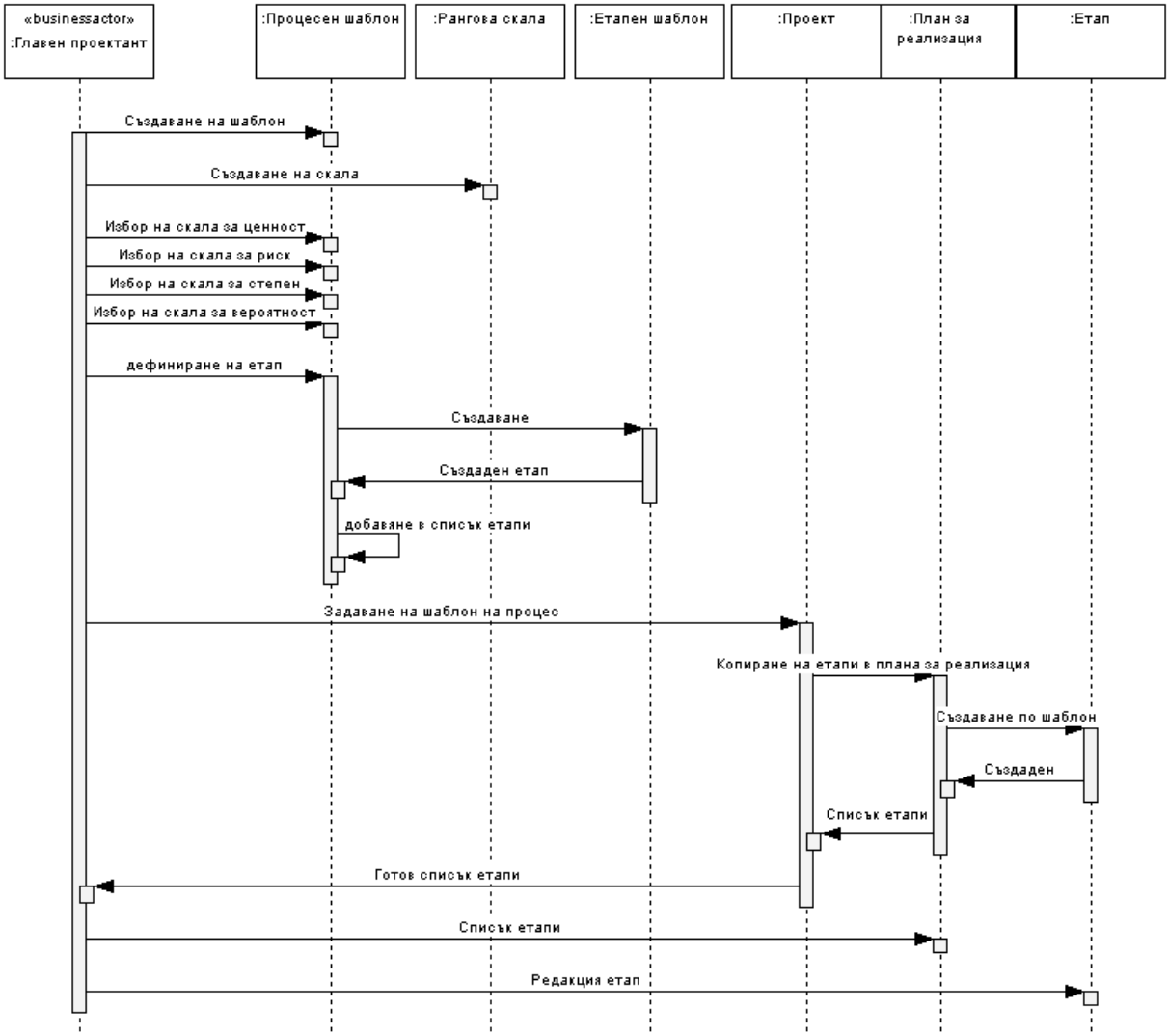
Фиг. 2.7. Диаграма на съществените класове за шаблони, формирани от бизнес същностите. (разработка на автора)

2.2.2. Диаграма на бизнес сценарии за взаимодействие със системата

По отношение на динамичните аспекти са разработени диаграми на дейности, реализиращи сценариите, където се показва начинът, по който протича процесът чрез дейности (фиг. 2.2). Тези дейности се детайлизират в диаграми на последователност и комуникационни диаграми (колаборативни, на сътрудничество) за класовете, за да се покаже начинът, по който дейностите се реализират чрез взаимодействие между класовете във времето и се осъществява съвместната им работа. За пример е дадена диаграма на последователност на фиг. 2.8 по отношение на дейността „планиране на проект” на база диаграмата за дейност на фиг. 2.2, т.е. това представлява диаграма за поведение на класове, с които се реализира дейността по БСВС „Планиране на проект”. Друга диаграма с голямо значение е дадена на фиг. 2.9 за последователност на реализацията на механизма за шаблони за процес, за рангови скали за ценност, риск, вероятност, степен и други, за да се реализира специфичен процес и подреждане. Фигурата е реализирана чрез диаграма за последователност на действията.



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



Фиг. 2.9. Диаграма за реализация на шаблони за процес.
(разработка на автора)

На тази база може да се определят операциите на отделните класове, което спомага както да се завърши дефиницията за интерфейса на бизнес логиката, така и функционалността на системата като необходим интерфейс. Допълването с диаграми на състоянията за статуса на класовете, които бяха представени в информационния модел на концепцията (фиг. 2.4), също е от съществено значение. Те посочват жизнения цикъл на обектите и необходимата логика при управлението им съгласно техния статус и събития.

2.2.3. Разположение на компонентите на системата по слоеве и насоки за изработка на работни продукти

По отношение на **компонентите на системата**, за да се реализира софтуерната система съгласно избраната технология, тя е структурирана в отделни логически компоненти. Компонентите са модули, които осигуряват определена функционалност като част от софтуерната система. Тези части са разположени на отделните логически слоеве в използваната тук трислойна архитектура. Технологично, на физическо ниво, тези компоненти се разполагат на съответен хардуер или виртуална машина – сървър за база данни, приложен сървър за бизнес логика и уеб достъп, и клиентски машини. При адаптацията на предложения подход се осигурява разделяне на слоевете и функциите с възможност за капсулиране, по-добра защита и поддръжка, и оптимизация на натоварването. По изключение може да се съвмести сървър за база данни и приложен сървър на една машина, ако е защитена от мрежови атаки. Описание на компонентите, разпределени по слоеве и хардуер, е дадено в табл. 2.1.

Таблица 2.1.
Разположение на компоненти по слоеве и хардуер
(разработка на автора)

Слой	Хардуер/ВМ	Компонент	Пояснения
Слой за данни	Сървър за БД	База данни на софтуерната система	За съхранение на данни относно продукти, проекти, процеси , потребители и други от софтуерната система. Реализира се с релационна БД.
		Хранилище за документи	За съхранение на документация. Може да е портал на проекта, от рода на MS SharePoint, система за управление на конфигурации или прост файлов сървър.
	Приложен сървър	Компонент за достъп до данни	Контролира и подпомага достъпа до данни в БД от бизнес логиката.
Слой за бизнес логика	Приложен сървър	Компонент за бизнес логика	Предоставя функции на логическо ниво и спазване на правила в софтуерната система. Контролира достъпа до данни.
		Компонент за автоматизация на процеси	Подпомага автоматизацията, прилага правила за промени и вземане на решения, както и правила за уведомление.
		Компонент за потребителски достъп	Потребители, оторизация, назначения, права за достъп.
		Компонент за уведомления	За уведомяване на потребителите. Генериране лично/ автоматично от софтуерната система, изпращане под различна форма – лог, email, други.

Слой	Хардуер/ВМ	Компонент	Пояснения
		Компонент за генериране на отчети	Генериране на данните за отчети и секции от лични табла. Генериране на отчетите с настройки.
Представителен слой	Приложен сървър	Уеб сървър	Сървър за хостване на сайт и уеб услуги.
		Портал за работа на потребители	Сайт за достъп от потребители.
		Компонент за програмен достъп	За програмен достъп към уеб услуги.
		Компоненти за потребителски интерфейс	Компоненти за интерфейс – генериране на графики, изобразяване на отчети, AJAX и други интерфейсни.
	Клиентска машина	Браузер за връзка с уеб приложение	Браузер за връзка с уеб сървъра и уеб приложението на софтуерната система.

Предлаганият от нас подход изисква изработка на **работни продукти (РП)** и при технологична стартираща компания те може да са под формата на документи, но част от тях са вградени в предлаганата софтуерна система. Някои са заимствани от фундаментални, според нас, разработки (Cockburn, 2021), други са преработени или добавени от нас на база личния опит по работа върху различни софтуерни проекти. Списъкът на работните продукти е следния:

– Визия – съдържа една-две страници описание на проект и цели, и списък на приоритети (критични, възможни или жертвани) за характеристики на софтуера[45];

– Екипна структура – обозначава начините, по които отделните членове на екипа комуникират – вградено е в системата с потребители, роли и назначения на потребители към проект с дадени роли;

– Конвенции за работа – списък от правила, практики и обичаи, които екипът спазва в хода на съвместната работа[46];

– Списък сценарии – списък от тип „Actor-Goal” за сценарии с

[45] Предлагаме да се разглеждат като вид насока в какво да се фокусира екипа – скорост, функции и т.н., като формират визията на софтуерната система.

[46] Посочва се избрания подход за разработка, както и указания за възлагане на роли, за кодиране, собственост на код, преглед на код, начин на работа и отчитане в итерации, преглед от потребител, управление на конфигурации и др.

отговорници, вграден в софтуерната система;

– Сценарии за взаимодействие – представен е във форма на модел или текстово описание – конкретно за тази софтуерна система това са бизнес сценарии на взаимодействие от ИБМ, дадено в концепцията;

– Модел на потребителски роли – списък роли в изработвания софтуер, аналогия на роли в софтуера, бизнес актьори и работници в ИБМ;

– Списък характеристики – списък на характеристики с подреждане и зависимости в свободна форма като списък или мрежови график – вградени са като списък/дърво в софтуерната система;

– Изисквания – списък изисквания в свободен текст от потребителите – вграден като списък и описание на характеристики в софтуерната система[47];

– Оферта – включва параметри за договор/план – срок, цена, условия, ограничения, списък с характеристики, обобщени оценки – добавена е от автора и вградена в софтуерната система;

– Договор – условия и параметри на договора с приложен план – добавен е от автора;

– План за реализации – срокове и разпределени характеристики – вграден в софтуерната система;

– Проектен статус – списък или диаграма с реализации, статуса им и бележка какво е направено; вградено в софтуерната система, но може да се подготви като „информационно табло” с маркер за статус;

– Списък рискове – съдържа списък на рискове по приоритет; описва се за вероятност, отговор, промяна на приоритет и засегнати итерации – вградено в софтуерната система като „информационно табло”;

– План/статус на итерация – списък на задачи и статуса им в текуща итерация;

[47] По наше мнение е подходящо и препоръчително в компанията да има приети общи шаблони за еднотипен софтуер, които ориентират клиента с въпроси.

– План за прегледи – списък на предстоящи реализации за уведомяване на екип и потребители; вградено в софтуерната система като списък реализации – може да е „информационно табло”;

– Програмистки продукти – продукти от проектант/програмист, които са за проектиране/реализиране на софтуера[48]; Вид, форма, детайли и съхранение на документите се избират от екипа (или описани в конвенциите); Софтуерната система поддържа само връзки към файлове;

– Тестове – списък автоматизирани тестове или описание за ръчни тестове за код и за приемане на софтуерния продукт от възложител – софтуерната система поддържа само списък на групи или на приемни тестове, които да се извършат (с описание или връзка към документ);

– Отчет на дефекти – списък с получени грешки при работа на софтуера в резултат на тест или забелязан проблем – софтуерната система ги поддържа като проблеми, а програмист конвертира проблемите в дефекти и ги поправя, като за софтуерната система са от тип задачи с работа за вършене;

– Помощен текст – от редактора помощен текст за ползване в системата;

– Потребителска помощ – структурирано описание от редактор на софтуера[49];

– Ръководство за потребителя – упътване от редактора за обучение на потребители[50];

– Резултати от отразяващо подобрение – обичайно списък в три групи за това, което „Работи добре”, „Ще опитаме да променим” и

[48] Включват модел на областта (например обектно ориентиран модел с UML), системна архитектура, дизайн на екрани, други програмистки скици и модели, сорс код, инсталационен пакет, дизайн, бележки.

[49] Потребителската помощ може да има стандартен формат на структура за всички точки, например: йерархия в глави, раздели, точки с цел, описание, важно, изисквания, връзки; може да се автоматизира с експорт на реализирани характеристики.

[50] Разликата с потребителската помощ е, че не е структурирана помощ по теми, а е във вид на учебник или ръководство.

„Нерешени проблеми” – вградено е в софтуерната система.

Някои РП могат да са информационни табла за обща/постоянна видимост, а други могат да са вградени в софтуерната система като документи, съхранявани в хранилището за документи.

2.3. Функционалност и потребителски интерфейс

По отношение на **функционалност и потребителски интерфейс**, представени в предходната глава е дефиниран в детайли интерфейса на потребителското приложение. Примерна скица на потребителския интерфейс е даден на фиг. 2.10 със следните основни елементи:

– Заглавна част – лого, заглавие, текущ потребител и дата, връзка за помощ;

– Главно меню – статично меню с опции видими според правата на потребителя;

– Контекстно подменю според избраното в главното меню и ролята;

– Маркер за наличие на нови съобщения, както и списък на спешни необработени такива;

– Списък на уместни и свързани проблеми, рискове, мерки за подобрение относно проекта и други данни свързани с текущата функция, роля или потребител;

– Списък на уместни/свързани съобщения относно текущия екран;

– Работна площ – място за работа на потребителя с екрани от опции на подменю;

– Завършваща лента – данни за продукт, версия, доставчик, връзки за помощ, обратна връзка за поддръжка и контакт с доставчик на софтуерната система.

Фиг. 2.10. Скица на основен екран на приложението.
(разработка на автора)

Потребителят има достъп до заглавната част, главното меню и завършващата лента винаги. В повечето случаи е видимо контекстното подменю, списъци с новини и свързани неща. След вход в системата с оторизация, се показва началният екран. Той също притежава гореизброените елементи, но на мястото за работна площ има лично (персонализирано) табло с кратки бързи справки за статус, цветна (светофарна) сигнализация, евентуално предстоящи срокове, кратки списъци с връзки за текущи проекти, задачи, проблеми и други най-важни елементи. Тези елементи обичайно са кратки, като данните са подредени

по приоритет или дата и ограничени като брой (например 5-те най-важни, 10-те най-закъснели).

Списъците имат бързи връзки към детайлен екран за съответния обект и функция. Личното табло дава само най-важната информация за проект/продукт, статуса му и процеса. При избор на опция от главното меню се появява подменю със списък опции, а на работната площ може да се появи или работният екран на първа опция от подменюто (обичайно преглед на данни) или вид лично табло специфично за избраната опция в главното меню. потребителя. Главното меню и подменютата имат видими опции, според правата за достъп на ролите на потребителя. Личното табло има секции, съгласно ролите на потребителя, а данните в него може да се филтрират за конкретния потребител.

В екраните за отделни функции, разположени в работната площ, е налице обичайна последователност. Най-отгоре са разположени секции и бързи връзки към други екрани в специална лента за страници/връзки. Под нея се разполага обща информативна секция за избран продукт, проект, итерация или друго, и/или евентуален филтър за бърз преход (ако е необходим). Под тях е самият екран за преглед/редакция, като в долния му край обичайно са бутоните за действия. За всички списъци има бутони/ връзки в дясната страна за стандартни бързи операции като избор, преглед, обновяване, приключване и т.н.

Списъците се подреждат по предназначението си или с клик на етикетите на колоните. Списъците по принцип нямат страниране, освен ако не са дълги (над 50 реда). Софтуерната система и екраните й предвиждаме да имат възможност да запомнят последен избор на потребителя и да предоставят филтър за това (например последен избран проект). В друг екран, който изисква избор и филтър, то той е избран автоматично за последното. Така преминаването между функции със запазен контекст улеснява потребителя, интерфейса и създава фокус за работа.

Относно отчетите, е удачно да се приложи гъвкава (лесно адаптируема) система, позволяваща както извикване на отчет от специален екран за заявка на отчет, така и директно от връзка в друг екран с параметри по подразбиране. В екрана за заявка на отчет се избира групата и вида отчет, задават се параметри като продукт, проект, реализация, итерация, служител, срок и евентуално друг частен критерий за даден отчет. Възможен е критерий за групиране на отчетите по време – дневни, седмични, месечни, тримесечни. Отчетът може да се заяви на екран или данните му да са изведени в CSV формат. При извеждането на екран той показва заглавие, избрани параметри и групиране, таблица с данните и секция за графика.

Основните функции и менюта с кратък коментар за тях са представени в табл. 2.2 – главно меню, подчинени менюта, както и свързани екрани.

Таблица 2.2.
Структура на менюта и потребителски интерфейс в клиентското приложение.
(разработка на автора)

Главно/ Подчинено меню	**Описание**	**Връзки с други екрани**
Начало	Главно лично табло със секции за сигнализация, списък и статус на достъпни проекти, графика за очакван край и степен изпълнение, други най-важни области.	
Проекти	Преминава в Проекти / Списък на мои проекти	
Задачи	Преминава в Итерации / Списък на мои задачи	
Проблеми	Преминава в Проекти / Списък на проблеми	
Дефекти	Преминава в Проекти / Списък на дефекти	
Съобщения	Преглед на лога със съобщения касаещи потребителя.	Добавяне/изпращане на ново съобщение.
Продукти	Показва лично табло	
Лични	Лично табло с текущи продукти и проекти	Списък продукти, нов продукт, Мои проекти, Добавяне на проект
Списък продукти	Азбучен списък с филтър – архивирани, търсене	Детайли за продукт, Нов продукт, Нова/ Редакция версия, Редакция клиент
Нов продукт	Може да се дефинира нов продукт.	
Клиенти	Списък клиенти	

Главно/ Подчинено меню	Описание	Връзки с други екрани
Списък клиенти	Азбучен списък на клиенти с филтър – текущи проекти, архивирани, търсене	Редакция/ Добавяне на клиент, Списък/ Добавяне на представител, Списък продукти, Нов продукт, Списък проекти, Нов проект
Списък представители	Азбучен списък представители с филтър – текущи проекти, архивирани, търсене	Редакция на представител, Редакция на клиент, Добавяне на клиент
Проекти	Активни проекти	
Активни	Азбучен списък на проекти	Списък проекти с филтър, Редакция проект, Добавяне на проект, Преглед реализация, Преглед итерация, Преглед / добавка на записки и документи; Отчети.
Списък проекти	Азбучен списък на проекти с филтър за име, код, срок, клиент, др.	Почти същите без списък; Отчети.
Добавяне на проект	Дефиниране на нов проект, условия и ограничения, избор на шаблон за процес и на рангови скали	
Редакция проект	Филтър за проект, Редакция на данни, срок, условия	Процес, Редакция шаблони за процес, Редакция на рангови скали
Процес	Филтър за проект, Списък етапи, Редакция на процес с етапи и данни за тях, избор на рангови скали (ако статуса позволява промени)	Редакция шаблони за процес, Редакция на рангови скали,
Сценарии	Филтър за проект. Дефиниция на списък сценарии (работен продукт „Action Goal List”), преобразуване в характеристики.	Връзка с Характеристики
Характеристики	Филтър за проект и тема на характеристики. Йерархичен списък характеристики. В един екран обновяване, добавяне, изтриване, преподредба, смяна на статус (ако статуса на проекта позволява).	Преглед и добавка на записки и документи,
Подреждане на характеристики	Филтър за проект и тема на характеристики. Списъкът с характеристики се подрежда за стойност и риск по избрана скала. Задават се и зависимости от вида – X преди Y.	Редакция на рангови скали
Оценка на усилия	Филтър за проект и тема на характеристики. Оценка на характеристиките за сложност и усилия.	
Договори	Филтър за проект. Списък договори, промяна на договор, добавяне на нов, задаване на параметри, смяна на статус, одобрение, приключване или отказ, справка за усилия според оценките.	Редакция договор, Нов договор, Оферта за проект
Реализации	Преглед реализация	
Преглед	Филтър за проект и реализация с автоматичен избор на текущата. Показва кратка статистика – брой характеристики, завършени и време за оставащи, др. Кратък списък итерации и рискове, както и данни за реализацията и прогноза за очаквано завършване.	Редакция реализация, Характеристики в реализация, Планиране на реализация
Редакция	Филтър за проект. Списък на реализации. Редакция на избрана или добавяне на реализация, дефиниране на срок и други параметри.	
Характеристики	Филтър за проект и за реализация (по подрзбиране текуща), за тема на характеристики. Йерархичен списък на всички и линеен списък на избрани характеристики. Възможност за преглед и ако статуса на реализацията позволява – промяна на избраните и статуса им.	Планиране

Главно/ Подчинено меню	Описание	Връзки с други екрани
Планиране	Филтър за проект и автоматичен избор на текуща реализация. Филтър за тема на характеристики. Йерархичен списък на всички и линеен списък на избрани характеристики. Възможност за добавяне, изтриване и преоценка на избраните (ако статуса го позволява).	
Списък рискове	Филтър за проект и реализация, автоматичен избор на текуща реализация. Списък рискове за реализацията подреден по ранг. Добавяне, редакция и премахване от списъка, смяна на ранг.	Назначаване на задача за риск
Подобрение	Филтър за проект и реализация, автоматичен избор на текуща реализация. Списък мерки от подобрението по групи и възможност за редакция, добавяне, махане, назначаване на задачи от тях.	Назначаване на задача за подобрение
Итерации	Преглед (с формата на лично табло)	
Преглед	Филтър за проект, реализация и итерация с автоматичен избор на текущи. Статистика за задачи и дефекти (брой, завършени, оставащо време), проблеми, тестове, рискове, оставащо време и др.	Редакция итерация, Задачи в итерация, Дефекти в итерация, Планиране на итерация
Редакция	Филтър за проект, реализация и итерация с автоматичен избор на текущи. редакция на итерация, срок и т.н.	
Планиране	Филтър за проект, текуща реализация и итерация избрани автоматично. Списък задачи, тестове и дефекти. Възможност за пренареждане, оценка на задачите. Справка общо и по потребител за съотношение капацитет време/назначени задачи коригирани с параметрите за индивидуална преценка.	Добавяне на задача, на тест или дефект. Добавяне на задача за риск.
Подобрение	Филтър за проект и реализация, автоматичен избор на текуща реализация. Списък мерки от подобрението по групи и възможност за редакция, добавяне, махане, назначаване на задачи от тях.	Назначаване на задача за подобрение
Работа	Преглед (с формата на лично табло за текуща работа)	
Преглед	Филтър за проект, реализация и итерация са автоматично избрани текущите. Показва сигнализация за статус на итерация, статистика за задачи (брой и завършени), дефекти (общо и завършени), списък на задачи и дефекти подредени по приоритет на характеристики или по дата на завършване, списък проблеми, рискове, и други.	Преглед, добавяне и редакция на задача, преоценка на задача, добавяне на дефект, добавяне на проблем
Задачи	Филтър за проект, реализация и итерация са автоматично избрани текущите. Списък задачи по ред или приоритет. Бързи връзки за смяна на статус, преоценка, оставащо време.	
Дефекти	Сходно на задачи. Бърза добавка, смяна статус, преоценка, време.	
Проблеми	Сходно на задачи. Бърза добавка, смяна статус и конверсия.	
Отчети	към Филтър на отчети	
Преглед	Филтър за избор на група отчети и конкретен отчет. Филтър за продукт, проект, реализация, итерация с автоматичен избор на текущите. Филтър за клиент, период. Избор групиране.	Генератор на отчет – генератор, който според избрания отчет и параметри го генерира като таблици с данни, графики.
Администрация	Празен екран.	
Шаблони	Редакция и добавяне на шаблони.	Нов шаблон.
Рангови скали	Редакция и добавяне на на рангови скали. Защита при употреба.	Нова скала.
Списъци	Редакция на списъци използвани в системата с възможна промяна.	Нов списък.
Профил	Преглед и редакция на профил на потребителя	

Главно/ Подчинено меню	Описание	Връзки с други екрани
Редакция профил	Преглед на данните и лично обновяване на профила на абоната	
Смяна на парола	Смяна на паролата с потвърждение на старата	

Таблицата е възможно да се допълва и разширява, т.е. тя не е изчерпателна и не детайлизира всички функции, но според нас дава съществена представа за главните екрани и посочва допълнителните.

Изводи и обобщения към втора глава

1. Разработен е концептуален модел на софтуерна система, управляваща основните процеси и дейности при разработката, поддръжката и съпровождането на софтуер.

За малките компании са характерни по-ниската степен на формализация, опростени процедури и правила. Затова по отношение на обхват на софтуерната система, приемаме че разработка на софтуер покрива определен набор от функционалности, като се избягва голямата сложност и влизане в чисто технически въпроси относно проектиране, програмиране, тестване и интеграция, управление на документи, конфигурации и т.н. За тях може да се ползва специализиран софтуер по индивидуална преценка.

При разработката на модела сме се ограничили единствено над основния бизнес сценарии за взаимодействие със системата, за който са разработени диаграми на взаимодействията и колаборативни диаграми, с които да се показва начина на взаимодействие между служителите с различни роли в технологичната стартираща компания и обхвата на техните отговорности. Техните отговорности са основа за формиране на функциите на системата и персонализираният достъп за всяка роля.

2. Разработен е логически модел на софтуерна система.

Логическият модел е разработен на базата на предложения

концептуален бизнес модел и за реализацията му в началото са обособени и развити бизнес същностите като класове със съответни свойства и връзки.

Глава 3. Изграждане и използване на системата за управление разработката на софтуер във фирма БитПайъниърс Блек Сии ООД

3.1. Организация на дейността на фирма БитПайъниърс Блек Сии ООД

Моделът на софтуерна система, подпомагащ разработката на софтуер в технологични стартиращи компании, е разработен във втора глава. За да се апробира приложимостта на разгледаната в детайли архитектура и предлаганата в настоящото изследване система, тя следва да бъде приложена в реална работна среда. Едновременно с това, е необходимо и да се очертаят границите на различните подходи за разработка на софтуер в технологични стартиращи компании, които също имат важно значение при организацията на дейността.

За целта е необходимо да се избере фирма от тип технологична стартираща компания – обект на приложение, която е с подходящ предмет на дейност, притежава известен начален опит в разработката на софтуер и има потребност от внедряване на подобна система. Компанията БитПайъниърс Блек Сии ООД, която авторът на настоящото изследване има възможността да познава добре по отношение не само предмета на дейността, но и организацията на работа, е подходящ обект на приложение. Поради това, че има сключено споразумение за неразкриване на конфиденциална бизнес информация (Non-Disclosure Agreement), каквато е обичайната практика в този висококонкурентен бизнес, то данни по отношение на конкретни клиенти, договори, цени и прочие няма да бъдат предоставени. Изложението третира само професионални въпроси от научна гледна точка.

Компанията БитПайъниърс Блек Сии ООД (BitPioneers Black Sea

OOD, https://bitpioneers.com/en/black-sea/) е основана през 2019 г. в гр. Варна с предмет на дейност разработка на софтуер и софтуерни продукти, разработка на мобилни приложения и уеб сайтове, покупка и продажба на софтуерни продукти, консултантска, рекламна и информационна дейност, и др. Основен собственик на капитала е немската компания BitPioneers GmbH (https://bitpioneers.com/) със седалище в гр. Хановер, която е основана през 2016 г. Българският клон активно работи по различни проекти на компанията-майка, която е специализирана в изпълнението на софтуерни проекти в областта на разработването на приложения за виртуална реалност (virtual reality) и разширена реалност (augmented reality). Екипът на компанията се състои от ИТ специалисти, компютърни експерти и разработчици на видеоигри. BitPioneers имат опит в разработването на рекламни игри, насочени предимно към средния ценови сегмент. Разработват първото си собствено заглавие на симулационна игра от тип стратегия Sphere: Flying Cities[51]. С цел навлизане и разпространение на този продукт на пазара се създава и нова компания – Hexagon Sphere Games UG (https://hexagon-sphere.com/), каквато е обичайната практика за определени собствени проекти с цел набиране на допълнителни инвеститори или подобряване на обслужването на потребителите. Като демонстрационни мобилни приложения за платформа Android са създадени приложенията Print Experience Streetart[52] и FUTUREADY AR[53] с възможности да се разширяват в областта на виртуалната и добавена реалност.

Дейността на компанията е специализирана в следните направления:

1. Дигитално събиране на данни. Екипът на компанията се опитва,

[51] За повече информация виж рекламен клип Sphere - Flying Cities Gameplay (PC), https://www.youtube.com/watch?v=OIZNmhskeqI

[52] Print Experience Streetart - https://play.google.com/store/apps/details?id=com.bitpioneers.pestreetart

[53] FUTUREADY AR - https://play.google.com/store/apps/details?id=com.bitpioneers.futuready

заедно с клиентите си да идентифицира тесните места и да намира прости и ефективни решения. Затова тясната съвместна работа с клиентите е ключов момент за въвеждането на подобни цифрови решения. Изследването на бизнес процесите и отчитане отзивите на потребителите е водещо при разработката на подходящи софтуерни продукти. Много често се използва модулен софтуер, който се адаптира към нуждите на потребителите.

2. Дигитализиране на работни процеси. Дигитализацията на работните процеси дава възможност на компаниите да разкриват нови възможности за тяхното оптимизиране, както и да ускорят изпълнението им. Дигиталното свързване на производствен капацитет и знания, дава възможност процесите да се извършват по-оперативно, ефективно и повече ориентирани към клиентите, което се превръща в решаваща движеща сила за повишаване на конкурентоспособността на средните компании. При дигитализацията на работните процеси, на различните видове потребители на софтуерната система се предлага различна „гледна точка“ към данните за процесите, така че потребителите на системата могат да се концентрират върху съществената, за конкретния момент, информация. По този начин може да се подобри проследимостта на стъпките на процеса и да се избегнат или автоматизират излишните работни стъпки, което от своя страна позволява на служителите да се концентрират върху важните и отговорни задачи, и да се повиши тяхната ефективност и удовлетвореност.

3. Отдалечена комуникация и виртуални събития. Отдалечената работа и комуникация позволяват на компаниите да реагират съобразно обстоятелствата бързо на ново появяващи се бизнес възможности и кризи, като се предоставя оперативен достъп до нови пазари, както и отговор на външни и вътрешни промени. Въпреки това, реалното общуване – ръкостискане и комуникация лице в лице има много предимства. Компанията БитПайъниърс позволява на своите клиенти да прехвърлят тези силни страни в дигиталния свят, като правят продуктите осезаеми и

комуникацията лична и не усложнена чрез проектирането на индивидуални решения за цифрова комуникация. За прилагането на подобни решения, се използват, комбинират и адаптират различни софтуерни модули. Възможностите варират от мащабируеми потоци с телевизионно качество до видеоконферентни и виртуални формати. Те могат да се комбинират с чатове, съдържание за изтегляне и онлайн гласуване. Друго направление е създаването на виртуални 3D светове, които „оживяват" предлаганите от клиентите продукти или услуги, при което потребителите могат да се потопят в събития, търговски панаири или търговски зали чрез разширена или виртуална реалност и по този начин да се свържат с доставчика.

Както се вижда от профила на компанията БитПайънииърс, тя е представител на технологичните стартиращи компании, занимаващи се с разработката на софтуер. Работи се както по собствени проекти, така и по поръчка, като динамично се сформират екипи, които да работят по конкретните проекти. Прилага се подходът за аутсорсинг и т.нар. „офшорно програмиране". Служителите работят дистанционно на принципа „виртуален офис". При подобни ситуации, освен разработката и експлоатацията на собствено разработена софтуерна система, специфично предназначена за вътрешните нужди на компанията, на преден план излизат и въпросите, свързани с използването на конкретен подход при управление процесите по разработката на софтуер, както и въпросите, свързани с възможността за използване на виртуални инструменти, които да допълват основната софтуерна система.

В заключение, може да се обобщи, че прилагането на софтуерната системата се характеризира със значителна комплексност и затова в следващите точки от изследването, спираме своето внимание на плана за реализация и внедряване на системата, и особености при експлоатация, с цел оптимизиране на процеса на внедряване на софтуерната система в компанията БитПайънииърс.

3.2. Организационни аспекти при реализация и експлоатация на системата

3.2.1. План за реализация и внедряване на системата

Един от основните моменти при разработката на софтуерната система е планът за реализация и внедряване. Както е известно, функционалността на софтуерните системи е възможно динамично да се променя във времето, при което се налага извършването на допълнителни дейности. Поради това е необходимо, както планиране на реализациите и версиите, така и характеристики, които се реализират в тях, заедно с разпределение във времето. На база изложеното до момента, е разработен съкратен вариант на работния продукт на база сценарии за взаимодействие. При необходимост той може да се разширява и допълва според потребностите и стратегическите виждания на ръководството.

Планът за реализация е представен в табл. 3.1. Заложените срокове са ориентировъчни и са подходящи за малки до средни по размери организации.

Таблица 3.1.
План за реализации на проекта на софтуерната система за управление производството на софтуер в технологични стартиращи компании.
(разработка на автора)

Етап / Реализация	Идея на етапа	Включени сценарии / характеристики	Примерен срок
Детайлно проектиране	Детайлизация на проект, оглеждане на 360°, конвенции.	Етап за детайлно проектиране с детайлизация на проекта на база текущия проект, концепцията и избран подход	Една седмица
Скелет на проекта	Скелет на всички компоненти и минимум работещ сайт достъп, меню и тестова редакция на продукти.	База данни с примерни данни и без оптимизация; Софтуерен проект и физически компоненти само със скелет на класовете; Настройки на сървъри, добавени базови компоненти. Структура на интерфейс; Логване на потребител; преглед/дефиниция на продукт;	Две седмици
Компоненти за логика и данни	Разработка на компоненти за данни и бизнес логика	Преглед/дефиниция на продукти; Достъп до данни; Усъвършенствани с новите модули и проиграване на модулите;	Две седмици

Етап / Реализация	Идея на етапа	Включени сценарии / характеристики	Примерен срок
Дефиниране на проект и план	Всички функции за дефиниране на проект, съдържание, процес и план; изпитване компонентите;	Сайт, в който потребител може да дефинира продукт, проект, характеристики, и т.н., подреждане и оценка на характеристики, избор на процес и скали за, дефиниция реализации, разпределение на характеристики, изготвяне на план;	Три седмици
Работа по задачи и тестване	Планиране и изпълнение на итерации и задачи, тестване, проблеми, дефекти	Дефиниране на итерации, задачи, списък рискове и задачи по тях, смяна на статус, минимален план и отчет на време по тях; дефинира проблеми, дефекти, конвертиране, обработка на дефекти и рискове;	Три седмици
Детайлен отчет в работата	Детайлизация на план и отчет по работата и итерациите	Записва изразходвано и оставащо време работа по задачи. Лично табло за статус. Отчитане разход на време и средства. Записки и документация към задачи, итерации и всички обекти. Записва изводи от отразяващото подобрение в итерация;	Две седмици
Приключване на реализация	Да може да се приключи реализация и отчетност. Да се оформи отразяващо подобрение.	Справки за статус и очаквано завършване. Корекция на оценки за итерации/реализации, дефиниция на реализирани продукти. Записва изводи от отразяващото подобрение в реализация. Назначава задачи от тях. Добавяне на заявки от клиент, трансфер в характеристики.	Две седмици
Пълен работен цикъл и администрация	Изчистване за пълен цикъл разработка. Администриране на клиенти и служители.	Подобрение за тестване и записки, експорт на списък характеристики за лесна документация. Тагове към всички обекти. Подобрени екрани за отчетност на време и средства. Клиенти, представители, администрация на потребители, настройки, продцеси, скали и т.н. Детайлни данни за служители – история, ставки, умения, календар за време и др.	Три седмици
Договаряне	Да може да се изготви оферта за клиент и дефинира договор;	Добавяне на нов договор или оферта, определяне усилията за тях, съобразяване с капацитет. Изготвяне на Оферта, отпечатване, дефинира договор/план с параметри; детайлно планиране на времето по реализации/итер и съответствие с капацитет.	Две седмици
Отчетност		Компонент за отчети, стандартни отчети за ефективност, приключване на проект. Подготвя информационни табла и други отчети.	Две седмици
Поддръжка	Дефиниране на поддръжка, мониторинг и работа по поддръжка	Дефиниране на поддръжка – координатор/главен проектант дефинира изисквания, условия, мониторинг при поддръжката; Следене и попълване на заявки за поддръжка. Конвертиране на заявки в дефекти, задачи или характеристики за обновяване. Опис на внедрени версии.	Две седмици
Финални	Други екстри; приключване	Потребител описва мнение за удовлетвореност от внедрена версия и оказана поддръжка. Отчети за дефекти, дял поправени, за поддръжка	Две седмици

Планът е за около 26 седмици (или 6 месеца) и подлежи на детайлизация. Целта е в първите три месеца да има годна система за оперативна работа по проекти. Когато след три месеца е готова, екипа

може с реални данни да изпита системата.

По отношение на процеса по **внедряване** са набелязани комплекс от основни стъпки и мерки, които трябва да се извършат при внедряване на софтуерната система в дадена технологична стартираща компания:

– Проучване и гарантиране, че технологичната стартираща компания и екипът й е готов за прилагане на подхода;

– Разработване на индивидуален подход, процес, техники, работни продукти в компанията;

– Бързо тестово проиграване на горните, за да е сигурно, че работят;

– Инсталиране на хардуер – сървъри, клиентски станции, мрежа, защита и т.н.

– Инсталиране на софтуер – системен и базов софтуер, приложения на софтуерната система;

– Настройка на потребители, права, шаблони за процес, скали, нови опции и т.н.

– Въвеждане /трансфер/ на данни от текущи и тестови проекти;

– Интеграция със съществуващ софтуер (ако е нужна)

– Проиграване и тестване с екипа в реални условия; финални корекции и настройки;

– Одобрение на внедрената софтуерна система от ръководството;

– Въвеждане на всички данни за текущи проекти и за желани исторически данни;

– Защита на средата и софтуерната система, създаване на план за архивиране и за справяне с рискове.

След внедряването, всяка система изисква съответни мерки преди и по време на нейната **експлоатация**. Набелязани са комплекс от мерки по използване и поддържане на софтуерната система. Тъй като предлаганият от нас подход за разработка на софтуер отделя специално внимание на работата в екипа и с софтуерната система работят участниците в екипа, то следва да се включат мерки по подготовка на екипа и използване на

софтуерната система от него. Тези мерки следва да се включат и като част от конвенциите (правилата) за работа по проекти. За първите приемаме, че могат да включват следните обичайни мерки в отделните етапи от отделните роли:

– **Администриране и координация**: администратор с помощта на главен проектант и координатор настройват софтуерната система, шаблони на процеси, скали и други опции; администратор и координатор осигуряват на всеки потребител адекватен достъп и данни за служителите, включително квалификация, история, работно време, ставки и т.н.; координатор и възложител се грижат за адекватно дефинирани продукти/проекти.

– В **началото** на всеки софтуерен проект се оглежда заданието според случая и планира проекта: екип/координатор описва условия, характеристики и параметрите им; екипът използва техники за оценяването му; координатор, възложител и главен архитект планират реализациите; изготвяне на оферта, дефиниране на договор и се планира детайлно първа реализация.

– **Ежедневно** екипът в технологичната стартираща компания има следните задължения: след 15 мин. среща се отразява всичко от срещата за статуса на задачи/проект, проблеми и бележки, лично или от координатор; преглеждат се всички новопоявили се новини, проблеми, рискове и т.н., обработват се и се назначава отговорен за тях; в началото на работа се отбелязва започването и темата на работа; в края на задача и ден се записва изработеното време и очакваното време до завършване на задачата или задачите, по които се работи.

– Поне веднъж или два пъти **седмично**: преглежда се статуса на всеки проект, всички рискове, проблеми, новини; преглеждат се отчетите за очаквано завършване и прогнози; при проблем (сигнализация в жълто или червено, или друго) се обсъждат мерките, които трябва да се вземат и се поставят лични задачи за справяне; при значим проблем с извънредна

работа, забавен ход, или необходима промяна на договор, част от работата се спира до предоговаряне; преглеждат се всички висящи новини, проблеми, рискове, дефекти и други записки, обработват се и се назначава отговорен за тях; базата от данни се архивира и съхранява на отделен сървър или носител; да не се допуска да има задачи, които не са отчетени и оценени.

– В **края на всяка итерация и реализация**: дефинира се продукта за реализация; записват се бележките от обсъждането с клиента; отразяват се в плана; допълва се отчетност по итерация, задачи, записки, характеристики и др.; извеждат се отчети за представяне в завършената итерация/реализация; отразяват се резултатите на отразяващото подобрение; дават се задачи; БД се архивира на технически носител и се съхранява извън офиса.

– В **края на всеки проект**: допълва се незавършената отчетност по проекта; описват се реализирани продукти, версия; финансови отчети; отразяват се резултатите на отразяващото подобрение; дават се задачи; ако са необходими корекции в процес и правила, предефинират се; проектът се приключва и архивира.

– При **аварийни ситуации**, когато софтуерната система не може да работи: екипът продължава работа по същество на проекта; ползват се работни продукти ръчно; администраторът коригира софтуерната система веднага; в най-близко време данните от работните продукти се пренасят в софтуерната система.

По отношение на **оперативния мениджмънт на екипа**, работещ с софтуерната система, в първа глава е изяснена важността на човешките ресурси и управлението им в технологичната стартираща компания, и бяха представени необходимите качества за работа с гъвкави подходи[54].

[54] Имат се предвид подходите, базирани на Manifesto for Agile Software Development, https://agilemanifesto.org/

Идеологията и духа на тези подходи изискват голяма тежест да се отдава на работата в екипа. Както отбелязват някои автори (Fowler, 2019) гъвкавите подходи не са подходящи за всички ситуации и изискват определени предпоставки и мерки. Необходимо е да се изяснят някои съображения и изисквания към екипа на технологичната стартираща компания и външните участници, и това трябва предварително да се има предвид.

Първо, участниците трябва да са готови/подходящи за прилагане на гъвкав подход.

Второ, участниците трябва да познават процеса и използваната методология за разработка. Според нас, една подходяща техника за подготовка е т. нар. „Тренинг в мини процес". В нея се проиграва процесът и методологията с най-прост примерен продукт, евентуално два пъти по 1-2 часа, за да са наясно новите членове на екипа с методологията. Още по-добър резултат може да се получи, ако те могат да проиграят, заедно с това, и работата с софтуерната система.

Трето, софтуерната система трябва да се познава и да може да се ползва. Въпреки, че целият екип може да взима участие в системата, повечето работа относно процеса и по-сложните функции касаят роли координатор и главен архитект. Затова тяхното обучение и трениране е ключово. То е и ключово, защото те се грижат за екипа и го насочват. Програмистите се очаква да работят „рамо до рамо", да имат близка комуникация и да ползват съвети от останалите, така че те могат бързо да бъдат обучени на място или да се само обучават един от друг. Системата е проектирана така, че да дава възможност за бързо получаване на кратка помощ и инструкции за специфичните опции.

Четвърто, потребителите трябва да имат минимум технически умения за работа с клиентско уеб приложение. Може да се смята, че в днешни условия това не е проблем за вътрешния екип, и малко вероятно за външни възложители/потребители. Необходимо е само кратко въведение в

начина на работа с приложението. Ако възникне проблем или потребители не са допуснати до софтуерната система, то координаторът с подходяща комуникация с тях може да изпълни тези задачи. Администраторът и вероятно координаторът е нужно да владее администрацията на продукта и да познава администриране на СУБД MS SQL Server или MySQL, и уеб сървър IIS, Apache или nginx, което включва регулярно архивиране на БД (Куюмджиев, 2019), настройки за достъп и оптимизация.

Пето, важни са уменията да се дефинира и пише точно, ясно и кратко, и да се избягва прекалена формалност. Софтуерната система и подходът са замислени като опростено средство и е излишна загуба на време извършването на повече административна работа. И не на последно място, справяне със стреса и намаляване на причините за него. Например, наличие на прекалено натоварване в задачи, видимост на прекалено много несвършена работа и др. Някои прости мерки в това отношение са полезни – интерфейсът показва прогрес, личното табло го показва и скрива дългите списъци над определен размер, фокус в работата, избягване на многозадачност и добавянето на нови характеристики поне в започната итерация, както и ритуали подкрепящи прогреса.

По отношение на **администрирането на системата** може да се приеме, че то трябва да се извършва само от ролите администратор и главен проектант. Важен компонент в софтуерната система е административна част в клиентското приложение и менюта „Администрация” и „Профил”, като чрез първото може да извършва:

– администрация на потребители – добавяне, активация, промяна на данни и парола;

– администрация на служители – пълни данни съгласно модела за служител и историята му – данни за контакт, квалификация, възнаграждение, параметри за работата;

– назначаване на потребител към проект с определена роля;

– дефиниране на шаблон за процес;

– дефиниране на рангова скала;

– дефиниране на списъци или добавяне на опции в разширяеми списъци;

– поправка на често срещани грешки – местене на задачи между итерации, характеристики между реализации, проблеми между проекти, корекция лог и подобни дефекти;

– логическо архивиране – изключване от работа на елементи маркирани като архивирани и улеснено цялостно архивиране на проект/продукт.

Чрез меню „Профил” потребителят може сам да сменя данните си и паролата си.

В хода на експлоатация на софтуерната система могат да възникнат проблеми от различни естество – хардуер, софтуерно осигуряване и др., и да прекратят ползването ѝ. **Възможните рискове и адекватни противодействия** са следните:

– срив в БД – регулярно архивиране на БД и процедура по възстановяване;

– срив в сървър – подмяна или местене на приложенията на нов; смяна на настройки; виртуализация;

– пробив в сигурността – ползване на сигурни протоколи – SSL/TLS, VPN, забранено кеширане, сигурност на работни станции и уеб сървър, ползване на силни пароли, и т.н.

– друг срив на системата – работата по проектите може да продължи с шаблони от работни продукти във файлове, а по-късно да бъдат въведени в софтуерната система;

За противодействие на рисковете е необходимо регулярно да се извършва преглед на следните мерки: изработване на план за справяне с рискове относно софтуерната система; дефиниране на специален проект за технологичната стартираща компания и процеса; регулярно документиране в специалния проект на проблеми, дефекти, рискове и отговорно поставяне

на задачи за разрешаването им; регулярно отразяващо подобрение в планове и конвенции за рискове.

Включването на управление на риска като реален проект и отговорното му възприемане под формата на задачи с пряка отговорност е предпоставка за по-надеждна работа.

Като потребители на софтуерната система могат да са технологични стартиращи компании с размер на екип от 2 до 9-12 човека. При по-голям персонал, екипът за един проект трябва да бъде в лимита и да няма пряка връзка и зависимости между проектите, в случай че член на екипа участва в няколко проекта едновременно. Едно възможно решение за по-голям екип, работещ по един софтуерен продукт, е разделение в по-малки екипи по отделни проекти със синхронизирани изисквания. Примерно, подобни проекти може да са относно:

– общи компоненти на продукта, валидни за всички версии;

– разработка на отделни версии на продукта;

– отделен екип да прави съпровождане на продукт;

– средства за продажби, автоматизация и обслужване на клиенти;

– индивидуални внедрявания на база основния продукт.

Синхронизацията между проектите може да се осъществява по следните няколко начина. Първият е, чрез синхронизиран дизайн на интерфейсите (Poppendieck, 2014) като инструмент. Втора идея е от подхода SCRUM да се правят регулярни ежедневни срещи между главните проектанти на отделни екипи. Аналогична идея съществува и в подхода LSD, където може да се приложи инструмент за вземане на подобни решения базиран на множество опции. Също така, възможна е ротация на членовете на екипи в различни области с цел пренасяне на опит, развитие и натрупване на разнообразен опит, разнообразяване в работата и други.

Софтуерната система се прилага за различни проекти – на външен възложител или собствени, с фиксирани или гъвкави параметри. Според нас, е възможно приложението в разработка на интелектуални продукти

различни от софтуер, но само ако избраният подход е удачен за тях. По-съществените ползи са:

– видимост на продукта и проекта в една система;

– лесна приспособимост в разработките и други ползи от инкрементален адаптивен подход;

– усъвършенстване на бизнес процесите;

– повишаване на качеството на управление чрез усъвършенстван процес;

– измерване на резултатите от дейността с опростени отчети;

– увеличаване на производителността и по-добра ефективност на специалистите;

– опростена система на работа;

– автоматизация за повечето работни продукти;

– формално вградени механизми за подобрение, добро качество и удовлетворение на клиентите;

– подходящо структуриране на персонала и разтоварване от административна работа;

– средства за разпределение на работата по тежест във времето;

– средства за прогнозиране спазването на сроковете.

Когато основната дейност на технологичната стартираща компания е производство и поддръжка на софтуер с конкурентно качество, цена, време за доставка, както и съпровождане, то управлението на компанията трябва да организира адекватно софтуерно производство. За целта е необходимо да се направи избор на подход и процес, като се опишат залегнали техники и практики, както и въпроси на управлението и работа в екип. Важен момент тук е изборът на подходящи подходи за организация на софтуерния процес, адекватни за обекта на управление, на чиято база да се осъществява управлението и чрез софтуерната система.

3.2.2. Особености при експлоатация

Разработването на система за управление разработката на софтуер е продължителен и сложен процес, който се нуждае от планиране и спазване на определени принципи. Прилагането на софтуерната система налага както технологични, така и организационни промени, които са породени и от необходимостта от прилагане на нова схема на управление, изискваща по-тясно сътрудничество между компанията и клиента. Необходимо е при изграждането на софтуерната система да се спазва и принцип за преизползваемост, т.е. повторната употреба на готови компоненти, като още в началото се предвидят възможности за повторно използване на компонентите чрез подходящо проектиране.

В разглежданата компания БитПайъниърс разработването на софтуерната система задоволява следните бизнес потребности:

– Автоматизиране на тези дейности по разработване на софтуер, които не са осигурени с подходящи инструменти;

– Обединяване на съществуващите системи в обща система за управление на разработка на софтуер;

– При тестването предлагаме да се използват автоматизирани средства за управление на тестването. Голяма част от потребителските изисквания се описват в текстови документи и впоследствие ръчно се преобразуват в тестови сценарии, при което използването на автоматизирани средства за управление на изискванията повишава производителността. Едновременно с това се повишават възможностите за бърза адаптация спрямо често променящи се и непредсказуеми изисквания от страна на клиента. Част от процесите по планиране на даден софтуерен продукт може да продължат да се извършват ръчно, съобразно конкретната обстановка.

– Прилаганите подходи в БитПайъниърс дават възможност за лесна бъдеща интеграции с нови софтуерни инструменти или за получаване/изпращане на данни, постъпващи от други бизнес партньори

при съвместна работа по проекти (например при аутсорсинг).

Разработването на софтуерната система решава посочените проблеми като осигурява относителна независимост между използвани платформи, софтуерни средства и програмни езици при разработването на дадено софтуерно решение. За целта се създават нови компоненти, които да предлагат необходимата функционалност.

При използването на система за управление разработката на софтуер в технологична стартираща компания е необходимо да се избере и съответен подход за разработка, подходящ за конкретната ситуация. Според нас, ако се използва подходяща софтуерна система, но се прилага неподходящ подход за управление на процесите на разработка, то няма да се постигне висока ефективност на работа и дори да се стигне до ситуация на неефективна работа по проекта. Поради тази причина приемаме, че изборът на подход при разработката на софтуер е също толкова важен, колкото е и използваната софтуерна система, прилагана за автоматизиране на отделните дейности по разработката на софтуер.

В практиката широко са разпространени няколко подхода, всеки със своите предимства и недостатъци при прилагане в различни ситуации. Затова обоснованият избор на подход може да се окаже от голямо значение за реализацията на софтуерен продукт от технологичната стартираща компания или за нейния провал.

В случаи на малка компания, като БитПайъниърс, с голям дял индивидуални („бутикови”) разработки, според нас, е подходящо преобладаващо да се използват т.нар. „гъвкави методологии” (виж Таблица 3.2).

В повечето случаи възможностите на БитПайъниърс да поема проекти с извънредно висока променливост на изисквания, непрестанно участие на потребителя на място и висок формализъм, са ограничени. В тези случаи, според нас, е подходящо прилагането на опростен подход за малък екип с ниска до средна формализация (по-висока за собствен

продукт). Времево, проектите са за 1-12 месеца. По своя срок на изпълнение и размер проектите условно можем да ги разделим в следните категории:

– микро поръчки, със срок на изпълнение в рамките на 1-2 седмици с много малко изисквания;

– малки или прости проекти, със срок на изпълнение до 3 месеца (външен проект, малко компоненти);

– големи, по-сложни проекти с повече модули и документация, със срок на изпълнение от 3 до 12 месеца. Още по-големи проекти може да се се разделят на няколко големи проекта. Тук влизат почти всички проекти за продукти/компоненти.

В случаи на **микро поръчки**, предвид ограниченията, според нас, подходящи са методологиите SCRUM, CC, LSD и DSDM. Според нас, методологията UP в тези ситуации не е подходяща, тъй като SCRUM се явява опростен вариант на UP. Същото се отнася и за методологията XP, поради не толкова голямата променливост на изисквания. За малки проекти трябва опростена адаптация на CC или SCRUM с опростени механизми. Според нас, най-удачен подход е CC, комбиниран с някои практики на SCRUM/LSD/XP като се разчита на кратки реализации и итерации (или сливането им), със срокове на проекти под 3 месеца. По този начин по-силен акцент може да има над техники за работа в екипа и гъвкавост.

За **малки или прости проекти** предвиждаме да се използват CC, SCRUM, LSD или DSDM, но с повече реализации/итерации, формализация и повече работни продукти, тъй като след като продуктът е по-голям, то и изисква последваща поддръжка и преизползване за компоненти. В този случай трябва да има отличителна разлика между реализации и итерации – реализация приблизително всеки месец, с 2-3 итерации в нея.

За **големи проекти** – продукт или компонент, предлагаме да се приложи CC, SCRUM или LSD. Целта е да има иновативен и по-

формализиран процес, с повече моделиране и документация, тъй като произведеният продукт трябва да се подновява и поддържа. За продукт предлагаме да има наличен един минимален пакет модели и документация, както и по-широк набор задължителни практики. Според нас е подходящо реализациите да са на един или два месеца с няколко итерации в тях. За още по-голям продукт е необходим съответно и по-формализиран процес, но без загуба на иновативност. За него може да се ползва SCRUM, LSD или CC с елементи от UP/DSDM.

Услугите по поддръжка предвиждаме да се реализират като малък проект за клиент с лек подход, минимум реализации и изисквания. Поддръжката има друг формат на работа, спрямо разработката, и се реализира в проект с лек стандартизиран подход и документация. Итерациите в поддръжката са едностепенни и може, обичайно, да са едномесечни. Ако е необходимо обновяване на продукт, изискванията се добавят към дадена итерация или се прави нов проект. За разлика от предходните, тук може да се добавят нови изисквания по време на итерацията, обичайно като промени или дефекти, като вторите се обработват възможно най-скоро в самата итерация. Поддръжката е натоварена с обновяване на модели, продукт, документация за поддръжка и за потребителя. В табл. 3.2 са показани избраните подходи.

Таблица 3.2.
Адекватни подходи за отделните дейности на компанията БитПайъниърс
(разработка на автора)

Вид дейност	Подход за малки проекти	Процес, фази, итерации за малки проекти	Подход за големи проекти	Фази и итерации на жизнен цикъл
Клиентски проект	CC, SCRUM с кратки цикли	Лек; Начална, Проектиране, 1-2 седмични итерации, Внедряване	SCRUM, CC, LSD, DSDM	Начална, Проектиране, 4-6 седмични реализации, Внедряване
Собствени продукти	Малко вероятно, SCRUM, CC,LSD	Планиране, Проектиране, 2-4 седмични реализации, Внедряване	SCRUM/UP, CC, LSD, DSDM	Планиране, Проектиране, 4-6 седмични реализации, Внедряване, Поддържане
Компоненти	Същите	Същите с модификация	SCRUM, CC, LSD	Същите с модификация

Вид дейност	Подход за малки проекти	Процес, фази, итерации за малки проекти	Подход за големи проекти	Фази и итерации на жизнен цикъл
Поддръжка	Модификация с регилярни и възник.задачи	Месечна поддръжка с итерации и реализации по необходимост	Модификация с регулярни и възник. задачи	Месечна поддръжка с итерации и реализации по необходимост

Възможно е да се избере комбинация от два подхода. Например единият може да се прилага при малки клиентски проекти, а вторият може да бъде изцяло за вътрешни продукти (и компоненти), както и за най-големите клиентски проекти. Причината е, че често успешни големи проекти (по-рядко малки) продължават в някои от следните възможности[55]:

– поддръжка поне една или две години след завършване;

– заявки за обновяване в следващите една до две години;

– основа за разработване на нови компоненти – например за чести проблеми;

– продукт или вертикална система за преизползване, който компанията БитПайъниърс може да изгради като собствен на база опита си в проекти;

– база за преизползване на готови решения в дадена област и проблем;

За малък проект реалистично се очакват само следващи микро заявки или малки проекти за обновяване, или дори компоненти на тяхна основа (ако технологичната стартираща компания реши), но рядко поддръжка. Поради това, според нас, е подходящо подходите да се разделят в две основни категории – една за малки проекти и една за продукти и големи проекти. За всеки конкретен проект БитПайъниърс трябва да определи границата между двете, както и случаите, при които е подходящо да се приложи определен подход. Според нас, за малки проекти най-подходяща е методологията СС. За големи проекти и продукти

[55] По лични наблюдения, опит и мнение на автора.

препоръчваме да се избере СС с повече техники и практики, както и елементи заимствани от методологиите SCRUM, UP, DSDM и LSD. В този случай, акцентът може да е върху характеристиките иновативност и качество, гъвкавост и „компромиси”, за се вмести БитПайъниърс в пазарни и вътрешни ограничения, както и да създаде максимално „очароващ продукт“. От организационна гледна точка с висока степен на важност са въпросите, свързани с **процеси, техники, практики и работа в екип**.

В заключение, може да се обобщи, че в зависимост от конкретната ситуация са подходящи различни подходи при разработката на софтуер, с цел работата по тези проекти да е организирана по оптимален начин в компанията БитПайъниърс. Заключенията, в резултат на изследванията, могат да се използват и от други сходни по размер и дейност компании. В следващите точки от изследването, спираме своето внимание върху необходимия инструментариум за физическа реализация и управление на разработването на системата и възможностите за използване на готови виртуални инструментални средства.

3.3. Физическа реализация на системата

3.3.1. Избор на технологични средства за реализация на системата

За физическата реализация на софтуерната система за управление на производството на софтуер в технологични стартиращи компании е необходимо да се подберат конкретни технологични средства за разработка и софтуерни компоненти. Предвиждаме софтуерната система да се интегрира с програмното осигуряване, използвано в обекта на приложение и по този начин да се реализира логическия модел, предложен във втора глава.

От голямо значение за изграждане на софтуерната система са следните етапи, които включват определяне на инструменталните средства

и приложното програмно осигуряване, които ще се използват:

1. Изграждане на слоя с данните:

– избор на релационна СУБД, която ще се използва, придобиване и инсталиране;

– изграждане на структурата на БД.

2. Изграждане на слоя за бизнес логиката:

– избор на програмен език, среда за програмиране и софтуерен фреймуърк за ускоряване разработката на програмните модули;

– разработване на програмните модули, които осъществяват връзката между слоя с данните и потребителския интерфейс;

– прилагане на тестове за осигуряване на качеството на софтуера.

3. Изграждане на представителния слой:

– избор на облачна услуга, хостинг услуга или уеб сървър, инсталиране и конфигуриране;

– избор на софтуерен фреймуърк за разработване на динамични уеб страници;

– разработване на уеб интерфейса на системата и на базовите интерфейсни компоненти.

Изборът на системно програмно осигуряване и по-специално изборът на операционна система е важен момент при разработката на софтуерни системи, но на този етап този въпрос остава на заден план. От една страна, наличен е софтуер за виртуални машини, който дава възможност под управлението на една ОС да работят множество виртуални компютри, всеки със своя собствена ОС, а от друга страна включването на Windows Subsystem for Linux (WSL) като един междинен слой, дава възможност за стартиране на Linux изпълними файлове под Windows 10 и Windows 11, които са най-разпространени сред десктоп системите. От тази гледна точка, следните базови въпроси и проблеми са извън обхвата на труда, тъй като в изложението се концентрираме само върху софтуерната част от една информационна система и освен това в

самата компания има налична техника, която може да се използва споделено. Това са въпросите по: избор и закупуване на компютърна техника - сървъри, мрежови устройства и инсталиране; осъществяване на връзката с Интернет - скорост, избор на доставчик; системно програмно осигуряване - избор и закупуване на операционна система за сървъри и клиенти, инсталиране, конфигуриране и поддържане.

За да се реализира практически софтуерната система е необходимо следния софтуер (програмно осигуряване) и инструментариум от средства:

1. Системен софтуер:

– сървърна операционна система: MS Windows Server 2019/2022 (налична във фирмата);

– клиентска операционна система: Microsoft Windows 10/11 (налична във фирмата), или друга, на която може да се ползва браузър със спецификацията, дадена по-долу;

2. Базов софтуер:

– система за управление на БД: Microsoft SQL Server Standard 2017/2019;

– уеб сървър на приложен сървър: Microsoft IIS 10;

3. Потребителски софтуер:

– интерфейс: браузър (Google Chrome, MS Edge, Mozilla FireFox, Opera, или друг с поддръжка на HTML5, CSS 3.0, JavaScript/ECMAScript 2021;

– програмна среда: ASP.Net 4.8/ASP.Net Core 6.0, C# 10.0 и VB.Net 16.9+ – за уеб приложение;

– програмни компоненти:

• ASP.Net AJAX – за поддръжка на AJAX интерфейс;

• FusionCharts XT за графики (http://www.fusioncharts.com/);

• вътрешни компоненти за библиотеки относно отделни слоеве.

– интерфейс за програмен достъп – уеб услуги за достъп (SOAP

API).

В заключение, предложения софтуер може да е предпоставка за реализиране на софтуерната система, но не е напълно достатъчен. Необходимо е използването на допълнителен инструментариум за управление на разработването на системата.

3.3.2. Инструментариум за управление на разработването на системата

Развитието на ИТ индустрията, повишаването сложността на софтуера, появата на нови методологии и процеси за разработване води до налагането на софтуерни инструменти за управление на разработването на софтуерни продукти. В най-общ план тези средства може да се класифицират по следния начин:

1. Утвърдени универсални инструменти за управление на проекти (MS Project 2019/Online, Scitor Project Scheduler 8.5, Oracle Primavera P6, Project Libre и др.) с богат набор характеристики и настройка с добавки, които покриват всякакви проекти от всякакви области. Обичайно подхода им е на база PMI[56] с добре дефиниран проект.

2. Специализирани за управление на проекти по даден подход (MS Visual Studio Team Services (VSTS)/Azure DevOps Services, Jenkins, Jira и др.). Те имат подобрени средства и техника само за него, понякога простота на работа и специфичен интерфейс, уеб достъп, възможност за преработка на код и БД. Някои са само добавка към горните.

3. Специализирани за софтуерни проекти/продукти по определена методология, обвързани с една или група методологии. Характерни черти са тясната специализация със софтуерна методология, покриват специфични функции за софтуерното производство, съвместна работа на

[56] Project Management Institute подход с правила за работа по проекти на база стандартен дефиниран процес.

екипа в системата с различни роли и права за достъп. Някои от тях са уеб базирани, дават възможност за индивидуална настройка или модификация, или интеграция с утвърдени продукти за управление на проекти (например MS Project).

Съществува голямо разнообразие на продукти във всяка една категория и ние считаме за подходящо, в случай че има съответни финансови средства, използването на MS Project 2019/Online. При липса на достатъчно средства считаме за подходящо използването на специализирани средства от нисък клас, които притежават общи полезни средства – InstantTeam (instant-team.com), Helix ALM (perforce.com), Project Tracker (projectmanager.com), и др. Всички тези продукти имат различен начин на действие, но общата функционалност – подход, групова работа, мобилност, уеб достъп, опции за индивидуализация и модификация, интеграция с популярни продукти са налице, т.е. те имат множество функции за проект, задачи, групова работа, дефекти и т.н.

Според нас, използването на MS Azure DevOps Services има следните предимства, които го правят подходящ за използване при разработката на конкретната система:

- управление на ниво продукт с отделни, добре дефинирани реализации;

- удобен специфичен визуален интерфейс с влачене, менюта, страници, подредба;

- приоритет на функционалности на база % бизнес тежест;

- тематично групиране на характеристики и филтриране/отчитане на база темите;

- следене на задачи в итерация, отчитане на отработено време, проблеми, бъгове;

- специфични лични списъци задачи, календар работно време;

- йерархична организация на характеристики за програма и влагането им в множество продукти, поставяне на тежест на групи; Отчети

за постигната бизнес стойност;

- уеб базирана система за отчети и споделянето им; настройваем генератор отчети;

- тематични отчети за скорост, променливост изисквания, постигната бизнес ценност, възвращаемост като съотношение на % бизнес тежест към % вложени усилия;

- графика за очаквано завършване и прогнозиране с линеен тренд, тренд от оптимистичен и песимистичен опит; предсказване на очакван край с емпирични данни за прогрес;

- интеграция с продукти за следене на дефекти като BugZilla и JIRA; експорт в Excel, пълен програмен достъп през SOAP API, e-mail уведомления.

Относно използването на софтуерни инструменти за специфични функции на производството и поддръжката на софтуер, при разработката на системата, могат да се използват инструменти в следните направления:

- интегрирани среди за разработка: MS Visual Studio, Eclipse и др.;

- управление на софтуерните конфигурации: MS VSS, Subversion, CVS, Git;

- автоматизирано тестване: XUnit, JUnit, FITnesse, HP Quality Center и др.;

- непрекъсната интеграция: MSVS Team Foundation, CruiseControl и др.;

- управление на дефектите: Bugzilla, Quality Center, JIRA, TestTrack и др.;

- управление на изисквания: FeaturePlan, CaseComplete, AcceptRequirements, Enterprise Architect, Borland CaliberRM и др.

3.3.3. Възможности за използване на виртуални инструментални средства

Виртуализацията е процес, възникнал в резултат на компютъризацията, която се е утвърдила с информатизацията на обществото. Този процес е обективен и характерен за информационните технологии и информационните системи. В контекста на настоящето изследване, под виртуализация разбираме реализация, която осигурява ресурси и услуги, които в действителност не съществуват или не съществуват в начина, по който са представени. Съществуват различни методи и технологии за виртуализация на информационните системи, като виртуализация на хардуера, на софтуерните услуги и комуникации, базирани на интернет (Сълов, 2014; Филипова и др., 2017).

Класическата теория приема системата за управление като единство на два компонента – субект на управление и обект на управление. Тази идея се развива в добавяне на трети компонент, информационната система. Тя заедно със субекта на управление формират системата за управление на организацията. В теорията информационната система се разделя на компоненти, а именно система за икономическа информация основана на информационната база и система за обработка на данни. На свой ред, информационната база се състои от информационен фонд и методи за организацията му (Илиев и др., 2010). На тази основа ние можем да дефинираме виртуалната софтуерна система като софтуерна система, в която са приложени технологии и инструменти за виртуализация, така че информационната база е организирана с виртуални средства и системата за обработка на данни също ползва виртуални среди, технологии, инструменти.

В резултат на виртуализацията се появява нова виртуална организационна форма в бизнеса, наречена бизнес мрежа (б’мрежа). Б’мрежата е обособена система от доставчици, дистрибутори, доставчици на търговски услуги, доставчици на инфраструктура и клиенти, които

извършват основна част от бизнес комуникациите и транзакциите си чрез Интернет (Илиев и др., 2010). Според нас, със сигурност, една стартираща компания като БитПайъниърс може да бъде част от подобна б'мрежа, да ползва предимствата на организацията в тази мрежа за създаване или взаимстване на бизнес модел и създаване на продукт. Част от виртуалните информационни системи и инструменти могат да бъдат също от доставчици на услуги, инфраструктура, търговски услуги и дори клиенти в б'мрежата.

Виртуализацията на информационната система и отделни инструменти може да бъде полезна в няколко насоки, за да се осъществи по-ефективно предприемачески процес.

Като **първи аспект**, най-важно за всяка компания е фокусът към стойността, която добавя виртуализацията към продукта и процеса на неговата разработка. Разработката на софтуер е разработка на интелектуален продукт. За да бъде създаден е нужен интелектуален капитал на компанията, който се състои от човешки капитал, капитал клиенти (потребителски капитал) и структурен капитал (Илиев и др., 2010).

Стартиращата компания има и капитала на предприемачите и техните ресурси в тези три направления. Чрез богатството на достъпни ресурси от данни, знания, мрежи и услуги, които предоставя Интернет, компанията може да увеличи капитала си в трите направления. По отношение на човешкия капитал, тя може да получи знание и информация за желаната сфера, за процеса, за обучение, да наема сътрудници за дадени задачи. За потребителския капитал може да се използват инструменти за проучване на пазар, конкурентно разузнаване, потребители, доставчици, партньори. За структурния капитал, изразяващ се в бизнес процеси и модели за удовлетворяване на пазарни изисквания, може да се проучат съществуващи бизнес модели, отзиви и препоръки за приложението им. Увеличението на интелектуалния капитал е предпоставка за успех в

иновационната дейност и то може да се постигне чрез управление на знанието и споделяне на знание. За това важна роля играе приложението и използването на информационни технологии подпомагащи тази дейност.

От казаното следва, че капиталът във всичките му форми е важен за технологичната стартираща компания и трябва да се мобилизира за насърчаване на по-висока производителност, която е чрез създаване (иновиране) на знание и споделяне на знания. Споделянето на знания може да се разглежда като организационна иновация, която има потенциала да генерира нови идеи и да развива нови бизнес възможности чрез социализация и учене на знания (Ling, 2012; Lee, 2016).

Втори аспект е начинът на фокусиране в своята основна дейност, която се осъществява с организационно-технологичен процес на десагрегация на елементите на веригата на стойността. Взема се решение за елементите, които да се запазят и за елементите, които да се предоставят на други компании с цел оптимизация. С последваща реагрегация се сглобяват елементите, за да се получи стойностно предложение. След това се прилага мултипликация, за да бъде ефективно осъществено. Поради наличието на множество видове модели на е-бизнес взаимодействие (B2B, B2C, B2G, B2E, C2B, E2B, G2B) (Croll, 2013; Laudon, 2017), според нас е подходящо, компаниите като БитПайъниърс да стартират дейността си, използвайки виртуални модели на взаимодействие с партньори и среда. В тези модели конкуренцията между доставчици за нови клиенти и цена на услуга калкулирана според потребление създава прозорци от промоционални услуги за стартираща компания в ранни етапи на дейност.

Трети аспект е, че този процес може да се приложи и към софтуерната система на стартиращата компания. Към нея може да се приложи виртуализация, десагрегация на подсистеми и организиране на транзакциите, последваща реагрегация за цялостна информация за бизнес модела и оценка на приноса.

Като **четвърти аспект**, стартиращите компании като

БитПайъниърс могат да осъществят дейности от своя предприемачески процес с минимум ресурси. Такива възможности за тях са:

– Да ползват на ниска или нулева цена достъпно знание и бази от данни;

– Да ползват услуги/инструменти от доставчици на ниска цена според потреблението;

– Да ползват безплатни услуги и продукти за началния период на разработка;

– Да имат лесен виртуален достъп до глобални пазари и отдалечени клиенти.

Пети аспект е организация на дейността в компанията чрез средства за виртуален офис и е-бизнес. Виртуален офис за организация е офис, съществуващ в компютърно симулирана среда, в който всички типични офис процеси се извършват чрез използването на компютърни системи. При ограниченост на ресурси, той е алтернатива за стартираща организация. Друга важен момент е, че софтуерът и процесът на разработката му могат да се ползват лесно от глобализацията чрез системите за виртуален офис и е-бизнес (Илиев и др., 2010).

Както е известно, е-бизнесът е форма на организация на бизнеса, при която бизнес процесите, обмяната на информация, финансовите транзакции и прочие дейности се автоматизират чрез информационни системи. Той дава възможности както за разширяване на базата от клиенти, доставчици, географски обхват, прилагане на електронна търговия и маркетинг (Sulova, 2018; Vasilev & Kehayova-Stoycheva, 2019), засилване на вертикалната интеграция и други.

Предвид аргументите, изложени по-горе, можем да твърдим, че използването на виртуализация при експлоатацията на софтуерната система може съществено да улесни стартиращата софтуерна компания. Затова считаме за полезно да се изследват и опишат съществуващи виртуални инструменти и информационни системи, които могат да се

използват в предприемаческия процес, като се разделят по етапи и по дейности. За всяка дейност посочваме виртуални инструменти, с които тя да се извърши или да бъде подпомогната. Под виртуални инструменти разбираме онези информационни системи, които са реализирани чрез виртуализация и са приложени принципи за виртуален офис (Илиев и др., 2010):

– Работа в среда на Интернет;

– Дислокация на работните места;

– Експлоатация без посредничество на програмисти;

– Процесорна обработка на информацията;

– Достъпност през уеб компонентът браузър и уеб сървър.

Списъкът с инструменти[57], съставен от нас, е примерен и посочва възможност, покритие на задачи, евентуално разнообразие, като няма претенцията да е изчерпателен.

Първата фаза, която разглеждаме, е „Подготовка". Тя има за цел валидиране на предприемач, обучение в предприемачество и подготовка на офис и среда за работа. Предприемачите трябва да увеличат своят капитал на тази фаза. Става въпрос основно за човешки капитал с подходящи знания и умения и в по-малка степен структурен капитал[58] чрез въвеждане на подходящи дейности (процеси) на комуникация и споделяне на знания с екип, ментори и други предприемачи. Отделните дейности и подходящите инструменти са представени в табл. 3.3. Както е видно, има много достъпни виртуални инструменти за тези дейности на минимална цена.

[57] По наше мнение, популярни в практиката системи с общо предназначение като виртуални инструменти към 2021 г. са: Jira (https://www.atlassian.com/jira) , Basecamp (https://basecamp.com/), Zoho (https://www.zoho.com/), Trello (https://trello.com/), Asana (https://asana.com/), Podio (https://podio.com/), Todoist (https://todoist.com/) и др.

[58] Под „структурен капитал" имаме предвид ИТ и другата инфраструктура, която подпомага човешкия капитал в работата му.

Таблица 3.3.
Връзка между дейности и виртуални инструменти за фаза „Подготовка”.
(разработка на автора)

№	Дейност	Инструменти
1	Годност за предприемачество	Онлайн инструменти за (само)оценка: psychometrictest.org.uk/entrepreneur-test/ queendom.com/tests/access_page/index.htm?idRegTest=2287, humanmetrics.com/entrepreneur и много други.
2	Обучение / Тренинг	www.udemy.com/the-lean-startup/ (Eric Ries) eu.udacity.com/course/how-to-build-a-startup--ep245 , fi.co, www.coursera.org/lecture/innovative-entrepreneur/lean-startup-mvps-UyZvW, www.startups.co/platform, scu.edu/mobi/about, www.startups.co/education, www.enterpreneur.com,
3	Комуникация с други предприемачи, ментори, екип	Професионални социални мрежи: linkedin.com, twitter.com, shapr.com; Средства за комуникация: E-мейл; Чат и споделяне (Skype, Viber, Slack), видеоконференции (Hangout, GoToMeeting, Skype), отдалечен достъп (GoToMeeting, LogMeIn); Консултации: your-startup-guru.com, други;
4	Подготовка на виртуален офис	Виртуален офис: HotOffice, eRoom, InTandem поддържащ екип и документи, комуникация, презентации и други: Общи офис приложения: Google Suite, Microsoft Office 365 Средства за презентация Проекти: bitrix24.eu, atlassian.com, trello.com, samepage.io; Инструменти за е-бизнес, като например онлайн плащания; Асистенти: zirtual.com, jobs.bg, zaplata.bg, upwork.com Онлайн хостинг: многообразие от хостинг услуги
5	Подготовка на инструменти за стартиращи компании	startupstash.com, www.startups.co/platform, eznumbers.com, 10000startups.com, startup-playbook.com,

След подготовката или паралелно с подготовката, когато тя има обучение, свързано с реална разработка, е фазата „Изобретяване”. В някои методологии тя се нарича „Генериране на идея”, „Design Thinking” и други. На първо място са включени дейности по разучаване или още наричани разузнаване. Ролята им тук е да се увеличи познанието за клиенти и техни проблеми, състояние, възможности, пазарни условия и т.н. Включени са дейности по дефиниране на проблем, генериране на идея, стратегически анализ, избор и представяне на идея.

В табл. 3.4 е представен списък с дейности на фазата „Изобретяване” и съответните инструменти. Трябва да отбележим, че тук покритието за дейността не винаги е пълно и разнообразно. Причините са, че фаза „Изобретяване” е едновременно, и творческа, и има необходимост от обстоен анализ, за който са налични средства за представяне и търсене на информация. Изборът на данни и източници, инструменти, избор на

идея и решения остават на отговорност на предприемача. Характерно е, че фазата винаги завършва с необходимост от подготовка за реално представяне на идеята пред различни аудитории – собствен екип, инвеститори, съветници, клиенти. Цел на това представяне е да се намерят ресурси, да се получи обратна връзка и намерят потенциални клиенти.

Таблица 3.4.
Дейности и виртуални инструменти за фаза „Изобретяване”.
(разработка на автора)

№	Дейност	Инструменти
1	Разучаване	
А	Избор на индустрия	Публични източници за пазара и тенденции, годишни отчети (gartner.com, marketresearch.com, idg.com, други).
Б	Конкурентно разузнаване	Пазарни анализи: gartner.com, marketresearch.com Публични регистри, статистика: trademap.org, investigativedashboard.org Преса: builtinla.com/2018/01/16/50-la-startups-watch-2018; Форуми, блогове, сайт, електронни борси; Социални мрежи: linkedin.com, twitter.com, shapr.com
В	Интервюта с клиенти * Лични интервюта, но за отдалечен пазар се ползват и виртуални средства.	За отдалечени клиенти и глобален пазар: социални мрежи (linkedin.com, twitter.com, pinterest.com), търсачки за публикации и въпроси (buzzsumo.com, quora.com, craigslist.com) и анкети (docs.google.com/forms, startupstash.com/forms-surveys, aytm.com, jotform.com)
2	Дефиниране на проблем	Виртуален офис за документи, презентации, бележки; публични източници за реализирани идеи, за формат на описание: fi.co/madlibs, leanstack.com/leancanvas;
3	Генериране на идея	milanote.com, curator.co, mindmeister.com
4	Стратегически анализ	Документи във виртуален офис; публични източници за сектора, тенденции и прогнози. Специализирани диаграми: swotanalysis.com, online.visual-paradigm.com,
5	Избор на идея	Средства не могат да бъдат представени;
6	Представяне на идея (пред екип, съветници, инвеститори и клиенти)	office.live.com/start/PowerPoint.aspx, slideshare.com, docs.google.com/presentation, Видео презентация: biteable.com/presentations, knovio.com, moovly.com

След фаза „Изобретяване” следва фаза „Бизнес модел”. В нея основните задачи са създаване на бизнес модела и неговото пазарно валидиране. Бизнес моделът съдържа концепция за продукта, именувана с различни термини, като: „продукт”, „уникално стойностно предложение” или „минимално функциониращ продукт” (MVP) (Ries, 2011; Maurya, 2012; Alvarez, 2014; Pompermaier et al., 2019; Melegati et al. 2020).

Валидиране на бизнес модела[59] се прави регулярно спрямо определен пазар от потенциални или реални клиенти. Извършват се тестове за валидност, събират се данни от тях с цел анализ, извличане на знания и вземане на решения за корекция в модела и технологичното решение. Затова показателите, които се избират, трябва да са пряко свързани с дейността и да водят пряко до вземане на решения. В литературата за различни бизнес модели са описани различни системи от показатели (Adler, 2011; Blank, 2013; Croll, 2013):

– AARRR за измерване на нужните действия към клиенти на Дейв МакКлър;

– Мотори на растежа (Engines of Growth) на Ерик Рийс;

– Схема за лесно стартиране (Lean Canvas) на Аш Мория;

– Пирамида на растеж (Growth Pyramide) на Шон Елис;

– Фуния за продажби (Long Funnel) – преход между етапи в процес на продажба;

– Платформа за анализ (Lean Analytics Framework) на Крол и Йосковиц.

Изборът и използването им зависи от процеса, но за всички изпитването на бизнес модела се оценява регулярно. Инструменти за тази фаза също са налични и са изведени в табл. 3.5.

Таблица 3.5.
Дейности и виртуални инструменти за фаза „Бизнес модел”.
(разработка на автора)

№	Дейност	Инструмент
1	Създаване	Проучване на модели, (не)успешни опити, показатели: leanstack.com, leananalyticsbook.com, startups.co,bizplan.com; Съвети от ментори: clarity.fm, fi.co, професионални социални мрежи и средства за комуникация: таблица 3.5; Виртуален офис с документи за описание: таблица 3.4; MVP чрез списък изисквания или прототип: в таблица 3.7;

[59] В случая имаме предвид процеса на установяване дали избрания целеви пазар има потребност от разработвания софтуерен продукт.

№	Дейност	Инструмент
2	Тестване * инструментите зависят от бизнес модела, подходът за оценка, показатели и инструменти. Дадените са примерни.	Създаване на БД клиенти: mailchimp.com, launchrock.com, Анализ на уеб трафик: indicative.com, manageengine.com, analytics.google.com, analytics.angelfishstats.com/, adobe.com/analytics/adobe-analytics.html; Маркетинг анализ на поведение: clicky.com, kissmetrics.com, shopmetrics.com, retentiongrid.com, adobe.com/marketing-cloud.html, metrilo.com, marketingplatform.google.com; Анкети: docs.google.com/forms, jotform.com, startupstash.com/forms-surveys, marketingplatform.google.com; Тестове: adobe.io/apis/experiencecloud/target.html, abtasty.com/, optimizely.com,
3	Извличане на знание	Обработка на тестови данни със за стандартен анализ: docs.google.com/spreadsheets, tableau.com/products/cloud-bi, sas.com/en_us/solutions/cloud-computing.html, turbo.net/hub/powerbi (Power BI online), sisense.com;
4	Корекция	Средства не са посочени.
5	Представяне	За презентации (виж таблица 3.10); За диаграми: таблица 3.9 и 3.12;

Може да се отбележи, че има разнообразни виртуални инструменти и източници на данни за тази фаза. Инструменти не са посочени за дейност „Корекция”, защото тя е резултат от анализ на резултати и вземане на решения за промяна.

Дейностите във фаза „Бизнес модел” и следващата фаза „Разработка на софтуер” се изпълняват постъпково с гъвкава методология, последователно или паралелно по избор на предприемача. Изборът зависи от степента на необходимост от прототип, реално или имитационно изпълнение на изисквания за минимално функциониращ продукт (MVP) във времето. Относно разработката на софтуер, използваните инструменти зависят от избран бизнес модел, архитектура на софтуера и технология за реализиране. В повечето случаи продуктите се реализират чрез многослойно архитектурно решение с отделни взаимно свързани слоеве, при което всеки слой има множество компоненти. Съществуват базови платформи за разработка (frameworks) от един доставчик и допълване с компоненти от други доставчици. Приложимо е не само за компоненти, които липсват, но и за такива, които са по-ефективни от стандартно заложените в платформата. Затова виртуалните инструменти може да са „твърдо” свързани с конкретна платформа или да са по-широко приложими.

В табл. 3.6 са представени стандартни основни и спомагателни дейности за разработка на софтуер, описани в литературата (Филипова и др., 2017).

Таблица 3.6.
Дейности и виртуални инструменти за фаза „Разработка на софтуер”.
(обобщено и доразвито от автора)

№	Дейност	Инструмент
1	Анализ	Виртуален офис за документи; Средства за проектиране (дейност Проектиране) Описание на изисквания: atlassian.com/software/confluence
2	Проектиране	Средства за моделиране: lucidchart.com, cacoo.com, sparxsystems.com/products/procloudserver/, dragon1.com/products/dragon1-connect-edition, products.office.com/en-us/Visio/, axure.com;
3	Реализация * Забележка: множество от средите поддържат интеграция и на други дейности и на широк спектър продукти.	Кодиране: codeanywhere.com/, aws.amazon.com/cloud9/, koding.com/, progress.com/kinvey, azure.microsoft.com/en-us/products/visual-studio-code, aws.amazon.com/lambda, telerik.com/app-development; Бази данни: azure.microsoft.com/en-us/services/sql-database/, azure.microsoft.com/en-us/product-categories/databases/, aws.amazon.com/products/databases; Бизнес логика: aws.amazon.com/lambda, azure.microsoft.com/en-us/product-categories/compute/; Уеб сървър: azure.microsoft.com/en-us/product-categories/web, Облачни услуги: Доставчиците имат много услуги – от API, изчисления, балансиране, опашки до изкуствен интелект; Хранилища за модули: nuget.org, npmjs.com, github.com, openwrap.org, incubator.apache.org/npanday,chocolatey.org; Потребителски интерфейс: angular.io, reactjs.org, vuejs.org, startbootstrap.com, mockplus.com, jqueryui.com, jqwidgets.com, riot.js.org, semantic-ui.com, webix.com;
4	Тестване – Верификация и Валидация	Автоматизирана интеграция и инсталиране на тестови среди: github.com, jenkins.io, gitlab.com, circleci.com, cloudbees.com, azure.microsoft.com/en-us/services/devops/; Системи за тестове: gurock.com/testrail, seleniumhq.org, techexcel.com/products/devtest/, getxray.app, ranorex.com, getzephyr.com; Автоматизирани тестове: ranorex.com, https://percy.io/, Сътрудници тестери: utest.com/about-us,
6	Внедряване	Средства за автоматизирано инсталиране: jenkins.io, azure.microsoft.com, www.gitlab.com, cloudbees.com, ;
7	Поддръжка	atlassian.com/software/jira/service-desk, atlassian.com/software/jira/ops,
8	Управление на проекта	azure.microsoft.com/en-us/services/devops, atlassian.com/software/jira, atlassian.com/software/trello, github.com/features/project-management, gitlab.com,
9	Управление на конфигурации и промени	gitlab.com, atlassian.com/software/crucible,
10	Конфигуриране на среда за работа	Избор на инструменти и настройка: azure.microsoft.com, Среди за програмиране, управление на проект (погоре); Хранилище: github.com, atlassian.com/software/bitbucket

Включени са примерни виртуални инструменти за разработка на

многослойно уеб приложение, основно с технологии и продукти, свързани с облачните платформи за разработка на компаниите Microsoft (Azure) и Amazon (AWS), като може да използват възможностите, които се предлагат и от други доставчици на облачни услуги (напр. Google, Alibaba Cloud, IBM Cloud, Oracle Cloud, Salesforce и др.). От резултатите е видно, че съществува голямо разнообразие от доставчици и средства, и покритие на дейности. Популярни в практиката са интеграции между инструменти. Според нас, особен интерес представляват три модела за интеграция между инструменти:

– Основният продукт е настолен, но се интегрира с много виртуални инструменти (Пример Microsoft Visual Studio с пакети и облачни услуги);

– Заложена интеграция чрез събитие от друг инструмент;

– Интеграция на инструменти чрез допълнително частно разработена обработка на събития.

В обобщение на представеното, и както е видно от табл. 3.3, 3.4, 3.5 и 3.6, за почти всяка дейност са налични виртуални инструменти, предоставяни като услуги. Въпрос на преценка на предприемача е кои от тях и доколко са удобни за даден продукт, идея и бизнес модел. По отношение на цената за използване, повечето инструменти имат нулева или ниска цена – безплатни версии, безплатни за микро екипи (до 5-10 потребителя) или цената е според брой потребители или обем консумация, което представлява голямо улеснение за една стартираща компания като БитПайъниърс, тъй като предполага ниски разходи за употреба. Съществен проблем е многообразието от инструменти за употреба и изборът на подходящ инструмент. В противовес на това, множество от продуктите имат функции, които покриват повече от една дейност и/или могат да се интегрират с трети инструменти за други дейности, което е улеснение за изграждане на една интегрирана среда за работа в стартираща компания.

От изложените резултати от проведените изследвания за

виртуализация, можем да направим обобщение, доколко дейностите в предприемаческия процес могат да се извършат с виртуални инструменти. Според нас, е необходимо да се съобразим с два основни показателя – покритие и разнообразие. Първият показател дава отговор на въпроса, доколко за задачите в дейността са разработени виртуални инструменти. Вторият показател – разнообразие, представлява степен на възможен избор от различни достъпни инструменти. За тяхно представяне използваме относителна скала с числа от 0 до 3, която е дадена в табл. 3.7, като синтез от двата показателя. Предлаганата от нас скала не отразява съотношение, а по-скоро групиране по тези два показателя с цел по-ясно отразяване и представяне на степента на покритие на задачи по дейности.

Таблица 3.7.
Скала на покритие на дейност с виртуални инструменти.
(разработка на автора)

	Разнообразие от различни достъпни инструменти		
Покритие на дейността от налични виртуални инструменти	*Ниско*	*Средно*	*Високо*
Липсва	0	0	0
Ниско	1	1	2
Средно	1	2	3
Високо/Пълно	2	3	3

Следвайки дадените показатели и дадената скала в табл. 3.7, можем да обобщим резултатите за покритие на дейностите от разглежданите инструменти (табл. 3.8).

Таблица 3.8.
Покритие на дейности по фази с виртуални инструменти.
(разработка на автора)

Подготовка	**Изобретяване**	**Бизнес модел**	**Разработка на софтуер**	
Годност	Разучаване	Създаване	Анализ	Проект

2	2	2	2	2
Обучение	Проблем	Тестване	Проектиране	Конфигурации
2	1	3	2	2
Комуникация	Идея	Знание	Реализация	Среда
3	1	2	3	3
Вирт. офис	Стратегич. анализ	Корекция	Тестване	Внедряване
3	1	0	3	2
Среда	Избор на идея	Представяне	Валидация	Поддръжка
1	0	2	2	2

В табл. 3.8 са включени общо 25 дейности. Дейност „Представяне на идея” и „Представяне на бизнес модел” са обединени, поради покритие на инструментите. Както е видно от таблицата, за две дейности няма посочени инструменти, а 4 дейности са с ниско, 13 са със средно и 6 с високо покритие и разнообразие. Данните от таблицата показват, че за 75% от дейностите имат налични виртуални инструменти със средно или високо покритие, и разнообразие. За останалите 25%, или липсват виртуални инструменти, или разнообразието е малко. Това ни дава основание да потвърдим в значима степен нашето първоначално предположение, че с виртуални инструменти могат да се изпълняват голяма част от задачите по разработката на нов продукт в стартираща софтуерна компания. Най-ниско е покритието във фаза „Изобретяване” и най-високо в „Разработка на софтуер”. Определено се забелязват възможности за иновация в инструментите за стартиращи компании в две посоки – увеличаване на покритието на задачи по дейности и интеграция в единна среда.

Изводи и обобщения към трета глава

1. При използването на система за управление разработката на софтуер в технологична стартираща компания е необходимо да се избере и подходящ подход за разработка. Прилагането на подходящ подход за

управление на процесите на разработка очакваме да осигури постигането на висока ефективност на работа и да се окаже съществен фактор за успех на проекта, по който се работи. Поради тази причина изборът на подход при разработката на софтуер е също толкова важен, колкото е и използваната софтуерна система, прилагана за автоматизиране на отделните дейности по разработката на софтуер.

2. Разработен е план за реализация, внедряване и експлоатация. Един от основните моменти при разработката на софтуерна система е планът за реализация, който включва както планиране на реализациите и версиите, така и характеристики, които се реализират в тях, заедно с разпределение във времето. Разработен е съкратен вариант на работния продукт на база сценарии за взаимодействие. При необходимост той може да се разширява и допълва според потребностите и стратегическите виждания на ръководството.

3. Производството на софтуерни продукти е специфична дейност и в зависимост от ситуацията е подходящ точно определен подход. Според нас, в повечето случаи за разглеждания обект на приложение е подходящо прилагането на гъвкав подход за управление на софтуерните проекти, базиран на подход СС и полезни добавки от SCRUM, LSD, UP, DSDM, XP и др. Предложени са няколко варианта – лек за малки (клиентски) проекти и услуги, и нормален за по-големи клиентски проекти и собствени продукти.

4. Използването на виртуални инструментални средства при експлоатацията на софтуерната система може съществено да улесни стартиращата софтуерна компания. С тази цел са изследвани и систематизирани съществуващи виртуални инструменти и информационни подсистеми, които могат да се използват в предприемаческия процес, по отделни етапи и дейности. За всяка дейност са посочени виртуални инструменти, с които тя може да се извърши или да бъде подпомогната.

Заключение

Настоящият дисертационен труд разглежда въпроси, свързани с управление на производството и поддръжката на софтуерни продукти в технологична стартираща компания. Основно място заемат проблемите на развитие на технологичната стартираща компания чрез добре организирано производство на качествен софтуер. Описана е специфичната дейност и са изведени проблемите пред технологичната стартираща компания по отношение на ключови области – стратегия и пазарно позициониране, производство и поддръжка на софтуер, управление на ресурси (човешки, финансови и информационни).

За малките компании са характерни по-ниската степен на формализация, опростени процедури и правила. Затова по отношение на обхвата на софтуерната система, приемаме че разработка на софтуер трябва да покрива определен набор от функционалности, като се избягва голямата сложност и влизане в чисто технически въпроси относно проектиране, програмиране, тестване и интеграция, управление на документи, конфигурации и т.н. За тях предлагаме да се ползва специализиран софтуер по индивидуална преценка.

При разработката на модела сме се ограничили единствено над основния бизнес сценарии за взаимодействие със системата, за който са разработени диаграми на взаимодействията и колаборативни диаграми, с които да се показва начина на взаимодействие между служителите с различни роли в технологичната стартираща компания и обхвата на техните отговорности. Техните отговорности са основа за формиране на функциите на системата и персонализираният достъп за всяка роля.

Основно качество на предлагания концептуален модел на софтуерна система е да има възможности за много насоки за развитие и затова най-важните му параметри са:

– оптимизирана работа за малки софтуерни проекти с едностепенни

цикли;

– синхронизация в екипа, автоматизация на решения и действия за смяна на статуси;

– подобрено управление на времето, несигурността и оценките; автоматизирана подредба на ресурси, характеристики и задачи във времето и на база връзки и капацитет;

– уведомления за проблеми, когато се очаква заложените параметри да са нереални;

– вариантност по тип и двуезикова поддръжка за документация.

Логическият модел е разработен на базата на предложения концептуален бизнес модел и за реализацията му в началото са обособени и развити бизнес същностите като класове със съответни свойства и връзки.

Един от основните моменти при разработката на софтуерната система е планът за реализация, който включва както планиране на реализациите и версиите, така и характеристики, които се реализират в тях, заедно с разпределение във времето. Разработен съкратен вариант на работния продукт на база сценарии за взаимодействие. При необходимост той може да се разширява и допълва според потребностите и стратегическите виждания на ръководството.

При използването на система за управление разработката на софтуер в технологична стартираща компания е необходимо да избере и съответен подход за разработка, подходящ за конкретната ситуация. Необходимо е да се прилага подходящ подход за управление на процесите на разработка, за да се постигне висока ефективност на работа и да се избегне ситуация на провал на проекта, по който се работи. Затова изборът на подход при разработката на софтуер е също толкова важен, колкото е и използваната софтуерна система, прилагана за автоматизиране на отделните дейности по разработката на софтуер.

Производството на софтуерни продукти е специфична дейност и в

зависимост от конкретната ситуация е подходящ един или друг подход. Според нас, в повечето случаи за разглеждания обект на приложение е подходящо прилагането на гъвкав подход за управление на софтуерните проекти, базиран на подход CC и полезни добавки от SCRUM, LSD, UP, DSDM, XP и др. Предложени са няколко варианта – лек за малки (клиентски) проекти и услуги, и нормален за по-големи клиентски проекти и собствени продукти.

Използването на виртуализация при експлоатацията на софтуерната система може съществено да улесни стартиращата софтуерна компания. С тази цел са изследвани и описани съществуващи виртуални инструменти и информационни системи, които могат да се използват в предприемаческия процес, по отделни етапи и дейности. За всяка дейност са посочени виртуални инструменти, с които тя може да се извърши или да бъде подпомогната. Нашите изследвания показват, че за 75% от дейностите имат налични виртуални инструменти със средно или високо покритие, и разнообразие, което означава, че с виртуални инструменти могат да се изпълняват голяма част от задачите по разработката на нов продукт в стартираща софтуерна компания. Най-ниско е покритието във фаза „Изобретяване” и най-високо в „Разработка на софтуер”.

Използвана литература

1. Галоуей, С. (2021) *Посткорона. От криза към възможност*. Изток-Запад, София.
2. Ескенази, А., & Манева, Н. (2006). *Софтуерни технологии.* София: КЛМН.
3. Закон за защита на конкуренцията. Държавен вестник, София, бр.17 от 26.02.2021 г.
4. Закон за корпоративното подоходно облагане. Държавен вестник, София, бр. 95 от 8.12.2015 г.
5. Закон за малки и средни предприятия. Държавен вестник, София, бр. 17 от 1.03.2016 г.
6. Закон за насърчаване на инвестициите. Държавен вестник, София, бр. 61 от 11.08.2015 г.
7. Закон за счетоводството. Държавен вестник, София, бр.19 от 05.03.2021 г.
8. Илиев, П., Сълов, В., & Петров, П. (2010). *Виртуални системи.* Монографична библиотека „Цани Каляджиев", Варна: Наука и икономика.
9. Коев, Й. (2016). *Кратък курс по предприемачество*. Варна: Стено.
10. Куюмджиев, И. (2019). *Методологически и технологични аспекти при архивирането на бази от данни*. Варна: Наука и икономика, Библ. Проф. Цани Каляджиев.
11. Пенева, П., Александрова, Я., & Армянова, М. (2013) *Бизнес информационни системи*. Варна: Наука и икономика.
12. Регламент (ЕС) № 651/2014. Европейска комисия, 16.06.2014 г.
13. Сълов, В. (2014). *Производителност и ефективност на компютърните системи*. Варна : Унив. изд. Наука и икономика.
14. Трейси, Б. (2011). *Ефективното лидерство*. София: Скорпио.
15. Търговски закон. Държавен вестник, София, бр. 64 от 18.07.2020 г.
16. Филипова, Н., Парушева, С., & Александрова Я. (2017). *Основи на информационните системи*. Варна: Унив. изд. Наука и икономика,
17. Abbas, N., Gravell, A. M., & Wills, G. B. (2008). *Historical roots of agile methods: Where did „Agile thinking" come from?*. In International

conference on agile processes and extreme programming in software engineering. Springer, Berlin, Heidelberg, pp.94-103.

18. Adler, C. (2011). *Ideas are overrated: startup guru Eric Ries' radical new theory*. Wired. 30.08.2011.
19. Alvarez, C. (2017). *Lean customer development: Building products your customers will buy*. O'Reilly.
20. ANSI/IEEE Standards Coordinating Committee. (1983). IEEE standard glossary of software engineering terminology (IEEE Std 729-1983). NY: IEEE.
21. Anwar A. (2014). A review of RUP: Rational unified process, *International Journal of Software Engineering (IJSE), 5*(2), pp.8-24.
22. Bass, J. M., & Haxby, A. (2019). Tailoring product ownership in large-scale agile projects: managing scale, distance, and governance. *IEEE Software, 36*(2), 58-63.
23. Beynon-Davies, P. (2018). Characterizing business models for digital business through patterns. *International Journal of Electronic Commerce. 22*(1), pp.98-124.
24. Blank, S. (2013). *The Four Steps to the Epiphany*. K&S Ranch Publishing.
25. Blank, S., & Dorf, B. (2012). *The Startup Owner's Manual: The Step-By-Step Guide for Building a Great Company*. K & S Ranch Publishing.
26. Booch, G. (2018). The history of software engineering. *IEEE Software, 35*(5), 108-114.
27. Burgelman, R. A., Christensen, C. M., & Wheelwright, S. C. (2008). *Strategic management of technology and innovation*. McGraw-Hill/Irwin.
28. Christensen, C., McDonald, R., Altman, E. & Palmer, J. (2018). Disruptive Innovation: An Intellectual History and Directions for Future Research. *Journal of Management Studies, 55*(7), pp.1043-1078
29. Clarke, P., O'Connor, R., & Yilmaz, M. (2018). *In search of the origins and enduring impact of agile software development*. In Proceedings of the 2018 International Conference on Software and System Process, pp.142-146.
30. Cockburn, A. (2021). *Design in Object Technology: „Class of 1994" (Series on Object-Oriented Design)*. Salt Lake City: Humans and Technology Press.

31. Colombo, M., & Grilli, L. (2010). On growth drivers of high-tech start-ups: Exploring the role of founders? human capital and venture capital. *Journal of Business Venturing, 25*(6), pp. 610-626.
32. Conboy, K., & Carroll, N. (2019). Implementing large-scale agile frameworks: challenges and recommendations. *IEEE Software, 36*(2), 44-50.
33. Croll, A., & Yoskovitz, B. (2013). *Lean Analytics*. O'Reilly Media.
34. Crossan, M., & Apaydin, M. (2010). A multi-dimensional framework of organizational innovation: A systematic review of the literature. *Journal of management studies, 47*(6), pp.1154-1191.
35. Curtis, G., & Cobham, D., (2008). *Business information systems: Analysis, design and practice*. Pearson Education.
36. Daft, R. (2020). Organization theory & design. Cengage learning.
37. DeMarco, T., & Lister, T. (2013). *Peopleware: productive projects and teams*. Addison-Wesley.
38. Desyatirikova, E., Belousov, V., Zolotarev, V., & Lavlinskaia, O. (2017). *Design process of software quality management*. In 2017 International Conference Quality Management, Transport and Information Security, Information Technologies (IT&QM&IS). IEEE. pp.496-499.
39. Dixit, R., & Bhushan, B. (2019). Scrum: An Agile software development process and metrics. *Journal on Today's Ideas-Tomorrow's Technologies, 7*(1), pp.73-87.
40. Duc, A., & Abrahamsson, P. (2016). *Minimum viable product or multiple facet product? The role of MVP in software startups*. In International Conference on Agile Software Development. Springer. pp.118-130.
41. Duff, A. (2013). *Information society studies*. Routledge.
42. Dwivedi, R., & Rohilla, V. (2017). *Empowering Agile Method Feature-Driven Development by Extending It in RUP Shell*. In Advances in Computer and Computational Sciences. Springer, pp.729-739.
43. Edison, H., Bin Ali, N., & Torkar, R. (2013). Towards innovation measurement in the software industry. *Journal of Systems and Software, 86*(5), pp.1390-1407.
44. Ejermo, O., & Xiao, J. (2014). Entrepreneurship and survival over the business cycle: how do new technology-based firms differ?. *Small Business Economics, 43*(2), pp.411-426.

45. Evans, E., & Szpoton, R. (2015). *Domain-driven design*. Helion.
46. Farid, A. B., Helmy, Y. M., & Bahloul, M. M. (2017). Enhancing lean software development by using DevOps practices. *International Journal of Advanced Computer Science and Applications, 8*(7), pp.267-277.
47. Fellmann, M., Koschmider, A., Laue, R., Schoknecht, A., & Vetter, A. (2018). Business process model patterns: state-of-the-art, research classification and taxonomy. *Business Process Management Journal, 25*(5), pp. 972-994
48. Giardino, C., Paternoster, N., Unterkalmsteiner, M., Gorschek, T., & Abrahamsson, P. (2015). Software development in startup companies: the greenfield startup model. *IEEE Transactions on Software Engineering, 42*(6), pp.585-604.
49. Gren, L., Goldman, A., & Jacobsson, C. (2020). Agile ways of working: a team maturity perspective. *Journal of Software: Evolution and Process, 32*(6), e2244.
50. Hanssen, G. K. (2011). Agile software product line engineering: enabling factors. *Software: Practice and Experience, 41*(8), pp.883-897.
51. Heitmann, J. (2014). The Lean Startup-A pragmatic view on its Flaws and Pitfalls (Bachelor's thesis, University of Twente).
52. Highsmith, J. (2013). *Adaptive software development: a collaborative approach to managing complex systems*. Addison-Wesley.
53. Hoff, J. (2019). Requirements practices in software startups. *Scholarly Horizons: University of Minnesota, 6*(1), pp.1-6.
54. IEEE Standards Coordinating Committee. (1990). IEEE standard glossary of software engineering terminology (IEEE Std 610.12-1990). Los Alamitos. CA: IEEE Computer Society.
55. ISO/IEC/IEEE 15288:2015 Systems and software engineering - System life cycle processes.
56. Jain, P., Sharma, A., & Ahuja, L. (2019). A customized quality model for software quality assurance in agile environment. *International Journal of Information Technology and Web Engineering (IJITWE), 14*(3), pp.64-77.
57. Janes, A., Lenarduzzi, V., & Stan, A. (2017). *A continuous software quality monitoring approach for small and medium enterprises*. In Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion. pp. 97-100.

58. Jolselt, J. (2019). Information society: its meanings and implications. Journal of Library, *Science Education and Learning Technology, 1*(1), pp.181-190.
59. Katila, R., Chen, E., & Piezunka, H. (2012). All the right moves: How entrepreneurial firms compete effectively. *Strategic Entrepreneurship Journal, 6*(2), pp.116-132.
60. Kaur, A. (2020). A systematic literature review on empirical analysis of the relationship between code smells and software quality attributes. *Archives of Computational Methods in Engineering, 27*(4), pp.1267-1296.
61. Khosravi, A., & Nilashi, M. (2018). Toward software quality enhancement by Customer Knowledge Management in software companies. *Telematics and Informatics, 35*(1), pp.18-37.
62. Kittlaus, H., & Fricker, S. (2017). *Software product management: The ISPMA-Compliant Study Guide and Handbook*. Springer.
63. Kiv, S., Heng, S., Wautelet, Y., & Kolp, M. (2017). *Towards a goal-oriented framework for partial agile adoption*. In International Conference on Software Technologies. Springer, pp.69-90.
64. Kroll, P., & MacIsaac, B. (2006). *Agility and Discipline Made Easy: Practices from OpenUP and RUP*. Pearson Education.
65. Larman, C. (2011). *Scaling Lean And Agile Development*. Pearson Education.
66. Larman, C., & Vodde, B. (2016). *Large-scale scrum: More with LeSS*. Addison-Wesley Professional.
67. Laudon, K., & Traver, C. (2019). *E-Commerce 2019: Business, Technology and Society* (15th Ed.), Pearson.
68. Leatherbee, M., & Katila, R. (2020). The lean startup method: Early-stage teams and hypothesis-based probing of business ideas. *Strategic Entrepreneurship Journal, 14*(4), pp.570-593.
69. Lemon, N., & Finger, G. (2020). *Digital technology*. In Teaching Early Years. Routledge. pp.143-166.
70. Ling, Y. (2012). A study on influence of intellectual capital and intellectual capital on complementarity on global initiatives. *Electronic Journal Knowledge Management, 10*(2), pp.154-162.
71. Link, P. (2016). *How to Become a Lean Entrepreneur by Applying Lean Start-Up and Lean Canvas?*, Innovation and Entrepreneurship in

Education (Advances in Digital Education and Lifelong Learning, Vol. 2), Emerald Group Publishing Limited, pp. 57-71.

72. Mahdavi-Hezave, R., & Ramsin, R. (2015). *FDMD: Feature-driven methodology development*. In 2015 International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE), IEEE. pp.229-237.
73. Manev, I., Manolova, T., Gyoshev, B., & Harkins, J. (2012). Social capital and strategy effectiveness: an empirical study of entrepreneurial venturesin a transition economy. *Современная конкуренция, 6*(36). pp.57-70.
74. Mangalaraj, G., Mahapatra, R., & Nerur, S. (2009). Acceptance of software process innovations–the case of extreme programming. *European Journal of Information Systems, 18*(4), pp.344-354.
75. Marconatto, D., Teixeira, E., de Oliveira Santini, F., & Ladeira, W. J. (2021). Characteristics of owners and managers in different countries: a meta-analytical investigation of SMEs' growth. Journal of Small Business and Enterprise Development.
76. Martin, W. (2017). *The global information society*. Taylor & Francis.
77. Maurya, A. (2012). *Running lean: iterate from plan A to a plan that works. The lean series* (2nd ed.). O'Reilly.
78. Maximini, D. (2015*). Scrum Culture. Introducing Agile Methods in Organizations*. Springer.
79. McConnell, S. (2019). *More Effective Agile: A Roadmap for Software Leaders*. Construx Press.
80. Melegati, J., Chanin, R., Sales, A., Prikladnicki, R., & Wang, X. (2020). *MVP and experimentation in software startups: a qualitative survey*. In 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). IEEE. pp.322-325.
81. Moran, A. (2016). *Managing Agile*. Springer.
82. Morgan, J., & Liker, J. (2020). *The Toyota product development system: integrating people, process, and technology*. Productivity press.
83. Mullins, J., & Komisar, R. (2013). *Getting to Plan B: Breaking Through to a Better Business Model*, Harvard Business Press.
84. Musulin, J., & Strahonja, V. (2018). Business model grounds and links: towards enterprise architecture perspective. *Journal of Information and*

Organizational Sciences. 42(2), pp.241-269.

85. Newmark, R. I., Dickey, G., & Wilcox, W. E. (2018). Agility in audit: Could scrum improve the audit process?. *Current Issues in Auditing, 12*(1), pp.18-28.
86. Nidagundi, P., & Novickis, L. (2017). Introducing lean canvas model adaptation in the scrum software testing. *Procedia Computer Science, 104*, pp.97-103.
87. Nuchprayoon, K., & Phuaksaman, C. (2018). Evaluation of Key Success Factors in Project Management of Information Systems and Selection of Operators using Analytical Hierarchy Process. *International Journal of Applied Engineering Research, 13*(7), pp.5515-5521.
88. Oorschot, K. E., Sengupta, K., & Van Wassenhove, L. N. (2018). Under pressure: The effects of iteration lengths on Agile software development performance. *Project Management Journal, 49*(6), pp.78-102.
89. Osterwalder, A., & Pigneur, Y. (2010). *Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers*. John Wiley and Sons.
90. Ozkan, N., Gok, M. , & Kose, B. (2020). *Towards a Better Understanding of Agile Mindset by Using Principles of Agile Methods*. In 2020 15th Conference on Computer Science and Information Systems (FedCSIS), IEEE, pp.721-730.
91. Paternoster, N., Giardino, C., Unterkalmsteiner, M., Gorschek, T., & Abrahamsson, P. (2014). Software development in startup companies: A systematic mapping study. *Information and Software Technology, 56*(10), pp.1200-1218.
92. Plonka, L., Sharp, H., Gregory, P., & Taylor, K. (2014). *UX design in agile: a DSDM case study*. In International Conference on Agile Software Development. Springer, pp.1-15.
93. Pompermaier, L., Chanin, R., Sales, A., & Prikladnicki, R. (2019). *MVP Development Process for Software Startups*. In International Conference on Software Business. Springer. pp.409-412.
94. Poppendieck, M. (2011). Principles of lean thinking. *IT Management Select, 18*, pp.1-7.
95. Poppendieck, M. B., & Poppendieck, T. D. (2014). *The lean mindset: ask the right questions*. Pearson Education.

96. Poppendieck, M., & Cusumano, M. A. (2012). Lean software development: A tutorial. *IEEE software, 29*(5), pp.26-32.

97. Preisendorfer, P., Bitz, A. & Bezuidenhout, F. (2012). Business Start-ups and their prospects of success in South African Townships. *South African Review of Sociology, 43*(3), pp.3-23.

98. Prohorovs, A., Bistrova, J., & Ten, D. (2019). Startup Success Factors in the Capital Attraction Stage: Founders' Perspective. *Journal of East-West Business, 25*(1), pp.26-51.

99. Rainer, R., & Cegielski, C. (2013). *Introduction to Information Systems*. Wiley Publ.

100. Rauch, A., & Hulsink, W. (2015). Putting Entrepreneurship Education Where the Intention to Act Lies: An Investigation Into the Impact of Entrepreneurship Education on Entrepreneurial Behavior. *Academy of Management Learning & Education. 14*(2), pp.187-204.

101. Ries, E. (2011). *The Lean Startup*. Crown Publishing Group.

102. Robbins, S., Coulter, M., & DeCenzo, D. (2020). *Fundamentals of management*. Pearson.

103. Rowley, J., & Hartley, R. (2017). *Organizing Knowledge: An Introduction to Managing Access to Information* (4th ed.). Routledge.

104. Sadowska, M., & Huzar, Z. (2019). Representation of UML class diagrams in OWL 2 on the background of domain ontologies. *E-Informatica software engineering journal, 13*(1), pp. 63-103

105. Santisteban, J., Mauricio, D. (2017). Systematic Literature Review of Critical Success Factors of Information Technology Startups. *Academy of Entrepreneurship Journal, 23*(2). pp.1-23

106. Santisteban, J., Mauricio, D., Cachay, O. (2021). Critical success factors for technology-based startups. *International Journal of Entrepreneurship and Small Business, 42*(4), pp.397-421.

107. Scarborough, N. (2013). Essentials of Entrepreneurship and Small Business Management, USA: Prentice Hall.

108. Schmitt, A., Rosing, K., Zhang, S., & Leatherbee, M. (2018). A dynamic model of entrepreneurial uncertainty and business opportunity identification: Exploration as a mediator and entrepreneurial self-efficacy as a moderator. *Entrepreneurship Theory and Practice, 42*(6), pp.835-859.

109. Silva, D., Ghezzi, A., de Aguiar, R., Cortimiglia, M., & ten Caten, C. (2020). Lean Startup, agile methodologies and customer development for business model innovation: A systematic review and research agenda. *International Journal of Entrepreneurial Behavior & Research, 26*(4), pp.595-628.

110. Sohaib, O., Solanki, H., Dhaliwa, N., Hussain, W., & Asif, M. (2019). Integrating design thinking into extreme programming. *Journal of Ambient Intelligence and Humanized Computing, 10*(6), pp.2485-2492.

111. Stellman, A., & Greene, J. (2014). *Learning agile: Understanding scrum, XP, lean, and kanban*. O'Reilly

112. Sulova, S. (2018). *Integration of Structured and Unstructured Data in the Analysis of E-commerce Customers*. In International Multidisciplinary Scientific GeoConference: SGEM, 18(2.1), pp.499-505.

113. Tanvir, S., Safdar, M., Tufail, H., & Qamar, U. (2017). *Merging prototyping with agile software development methodology.* In International Conference on Engineering, Computing & Information Technology (ICECIT 2017), pp.50-54.

114. Vasilev, J., & Kehayova-Stoycheva, M. (2019). Sales Management by Providing Mobile Access to a Desktop Enterprise Resource Planning System. *TEM Journal, 8*(4) Serbia : UIKTEN-Assoc. for Inform. Communication Technology, pp.1107-1112.

115. Wautelet, Y. (2020). *Using the RUP/UML business use case model for service development governance: A business and IT alignment based approach*. In 2020 IEEE 22nd Conference on Business Informatics (CBI), Vol. 2, pp.121-130.

116. Wirtz, Bernd W. (2019). *Digital business models: Concepts, models, and the alphabet case study*. Springer.

117. Zacca, R., & Dayan, M. (2017). Entrepreneurship: an evolving conceptual framework. *International Journal of Entrepreneurship and Innovation Management, 21*(1-2), pp.8-26.

Интернет източници

118. Даскал, Л. (2018). 7 стратегически умения за по-ефективно лидерство. Мениджър. <https://manager.bg/liderstvo/trayna-7-nachina-da-bdete-po-efektivni-lideri> [23.08.2022]

119. Икономи, П. (2018). Тайните на ефективното лидерство.

Мениджър. <https://manager.bg/liderstvo/trayna-taynite-na-efektivnoto-liderstvo> [23.08.2022]

120. Русева, Р. (2015). *Моделиране на технологични иновации. Интегриран подход за моделиране на технологични иновации*, дисертация, СУ, ФМИ, <https://fmi.uni-sofia.bg/sites/default/files/dissertation_work_of_phd/radostina_ruseva_phd_thesis_final_print_for_pdf.pdf> [10.10.2018]

121. Bormans, J., Privitera, M., Bogen, E., Cooney, T. (2020). European Startup Monitor 2019/2020. European Startup Network. <http://www.europeanstartupmonitor2019.eu/EuropeanStartupMonitor2019_2020_21_02_2020-1.pdf> [02.03.2021]

122. Boyer, S., Sharp, J., Matthews, A., Stollery, P. (2021). *Fusion development approach to building apps using Power Apps*. Microsoft. Ebook: <https://docs.microsoft.com/en-us/powerapps/guidance/fusion-dev-ebook/> [11.11.2021]

123. European Commission. (2019). SME definition. <https://ec.europa.eu/growth/smes/business-friendly-environment/sme-definition_en> [01/10/2019]

124. Fowler M. (2019). Agile Software Guide. <https://martinfowler.com/agile.html> [10.12.2020]

125. Graham, P. (2012). Startup = Growth. September 2012, <http://www.paulgraham.com/growth.html> [01/02/2020]

126. GrandViewResearch, Business Software And Services Market Size, Share & Trends Analysis Report By Software, By Service, By Deployment, By End-use, By Enterprise Size, By Region, And Segment Forecasts, 2021 – 2028. <https://www.grandviewresearch.com/industry-analysis/business-software-services-market>

127. IBM Cloud Education, (2020). Three-Tier Architecture, <https://www.ibm.com/cloud/learn/three-tier-architecture> [20.12.2021]

128. Iqbal, M. (2021). Uber Revenue and Usage Statistics Business of Apps. <https://www.businessofapps.com/data/uber-statistics/#6> [Accessed: 01/11/2021]

129. Lee, M. (2016). Knowledge management and innovation management: best practices in knowledge sharing and knowledge value chain. *International Journal of Innovation and Learning*, <https://www.researchgate.net/profile/Ming-Chang_Lee2/

publication/292671860> [25.10.2018]

130. Microsoft Patterns & Practices Team, (2009). *Microsoft Application Architecture Guide*. Microsoft Press; Second edition, <https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ff650706(v=pandp.10)> [20.12.2021]

131. Ripsas, S., Schaper, B., & Troger, S. (2015). A startup cockpit for the proof-of-concept. *Handbuch Entrepreneurship*. <https://link.springer.com/referenceworkentry/10.1007/978-3-658-05263-8_21-1> [01/04/2020].

132. Robehmed, N. (2013). What Is A Startup? *Forbes*, <https://www.forbes.com/sites/natalierobehmed/%202013/12/16/what-is-a-startup/> [16.12.2020].

133. Tracy, B. (2015). Leading and Motivating. <http://keithlee.com/leading-motivating-by-brian-tracy/> [23.08.2022]

134. U.S. Bureau of Labor Statistics (2021). Table 7. Survival of private sector establishments by opening year. <https://www.bls.gov/bdm/us_age_naics_00_table7.txt> [20.12.2021]

135. Zanni, T. (2019). Disruptive companies and business models, KPMG Technology Industry Innovation Survey. <https://assets.kpmg/content/dam/kpmg/es/pdf/2019/10/disruptive-companies-business-models-report.pdf> [01.03.2020]

Списък с публикации по темата на дисертационния труд

Статии

1. **Ivanov, S.**, & Petrov, P. (2020). Business Scenarios for Interaction in the Development of the Software System in a Start-up Software Company. *Economics and Computer Science, 6*(2), Varna: Knowledge and Business, pp.27-37.
2. **Ivanov, S.** (2019). Organization of Activities and Management in Starting Software Firms. *Izvestia Journal of the Union of Scientists – Varna. Economic Sciences Series, 8*(3), Varna: Union of Scientists – Varna, pp.196-207.
3. **Иванов, С.** (2018). Виртуални инструменти за разработка на софтуер в стартираща софтуерна компания. *Известия на Съюза на учените – Варна. Сер. Икономически науки, 7*(3), с.149-160.

Доклади

1. **Иванов, С.** (2020). Информационни системи в стартиращи софтуерни компании. *Икономическа наука, образование и реална икономика: развитие и взаимодействия в дигиталната епоха* : Сборник с доклади от Юбилейна международна научна конференция в чест на 100-год. от основаването на ИУ – Варна: Т. 1, Варна: Наука и икономика, с.702-713.
2. **Ivanov, S.**, & Petrov, P. (2020). Development of Software Systems by Using Interaction Business Scenarios. *10th International Conference on Application of Information and Communication Technology and Statistics in Economy and Education (ICAICTSEE – 2020)* : Conference Proceedings, Sofia: UNWE, pp.411-419.
3. **Ivanov, S.** (2019). Development Stages of Starting Software Company, Problems and Approaches for Software Development. *Information and Communication Technologies in Business and Education* : Proceedings of the International Conference Dedicated to the 50th Anniversary of the Department of Informatics, Varna: Science and Economics Publ. House, 2019, 1, pp.382-396.
4. **Ivanov, S.**, & Petrov, P. (2019). Business Models for Starting Software Companies. 9th International Conference on Application of Information and Communication Technology and Statistics in Economy and Education (ICAICTSEE – 2019) : Conference Proceedings, Sofia: UNWE, pp.170-182.
5. **Иванов, С.** (2009). Средства за управление на софтуерни проекти с гъвкави методи. *Информационни технологии в управлението на бизнеса* : Сборник доклади от международна научна конференция посветена на 40 год. от създав. на кат. Информатика, 16-17 октомври 2009, Варна: Наука и икономика, с.70-76.

Справка за приносните моменти

Основните приносни моменти на настоящето изследване могат да се обобщят по следния начин:

А. В теоретичен план

– направено е изследване на особеностите на стартиращите технологични компании, разработващи софтуер, тяхната дефиниция, управление и организиране на дейността;

– направено е изследване за методичните проблеми, свързани с организация на дейността и подходите за управление, подходящи за стартиращи технологични компании;

Б. В приложен план

– с използването на утвърдени формални средства са разработени концептуален и логически модел на софтуерната система за технологични стартиращи компании, разработващи софтуер;

– предложен е практически план за реализация на софтуерната система;

– аргументирани са софтуерни технологии, които да се използват в зависимост от размера на софтуерния проект, изпълняван от технологична стартираща компания;

– направено е обобщение на софтуерните инструменти, които могат да се използват в отделните етапи на процеса по създаване на софтуер във виртуална среда.