# 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]](#footnote-2). 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 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]](#footnote-3) с дейност разработка, поддръжка и съпровождане на софтуерни продукти. В тази връзка е необходимо да се очертае обхвата на понятието „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 вурху 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 асоциираме 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]](#footnote-4), водено от предприемач[[4]](#footnote-5) 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.

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]](#footnote-6) (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]](#footnote-7) 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):

– 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]](#footnote-8) 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, 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]](#footnote-9). 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.

## 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]](#footnote-10):

– 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]](#footnote-11) 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 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]](#footnote-12)** | **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 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]](#footnote-13). 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 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]](#footnote-14). 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]](#footnote-15) 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 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]](#footnote-16) 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 стартираща компания може да се групират в следните групи[[16]](#footnote-17):

– 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 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]](#footnote-18) (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]](#footnote-19). 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.

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]](#footnote-20), техните параметри и статуса им;

– Потребности от ресурси, време и разпределението им по проекти 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
   1. 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:
   1. Създава се минимално функциониращ продукт (MVP) (Duc et al., 2016);
   2. Регулярна обратна връзка от клиентите;
   3. Тестване на версии с група клиенти за валидация (Split or A/B Testing);
   4. Ползват се действени показатели (Actionable Metrics) водещи пряко до управленски решения;
   5. Структуриран процес за корекции в модела (Pivot);
4. Опростен и повторяем процес:
   1. Проблем и решение (Problem/Solution Fit) с цел намиране на MVP;
   2. Продукт и пазар (валидиране на бизнес модела) (Product/Market Fit);
   3. Разрастване (Scale);
5. Кратки цикли итерации с три стъпки за създаване на артефакти:
   1. Създаване (Build) – създава се артефакт (MVP, модел и други) с хипотези;
   2. Тестване (Measure) – тест с показатели; обратна връзка от клиенти;
   3. 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]](#footnote-21).

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.

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.

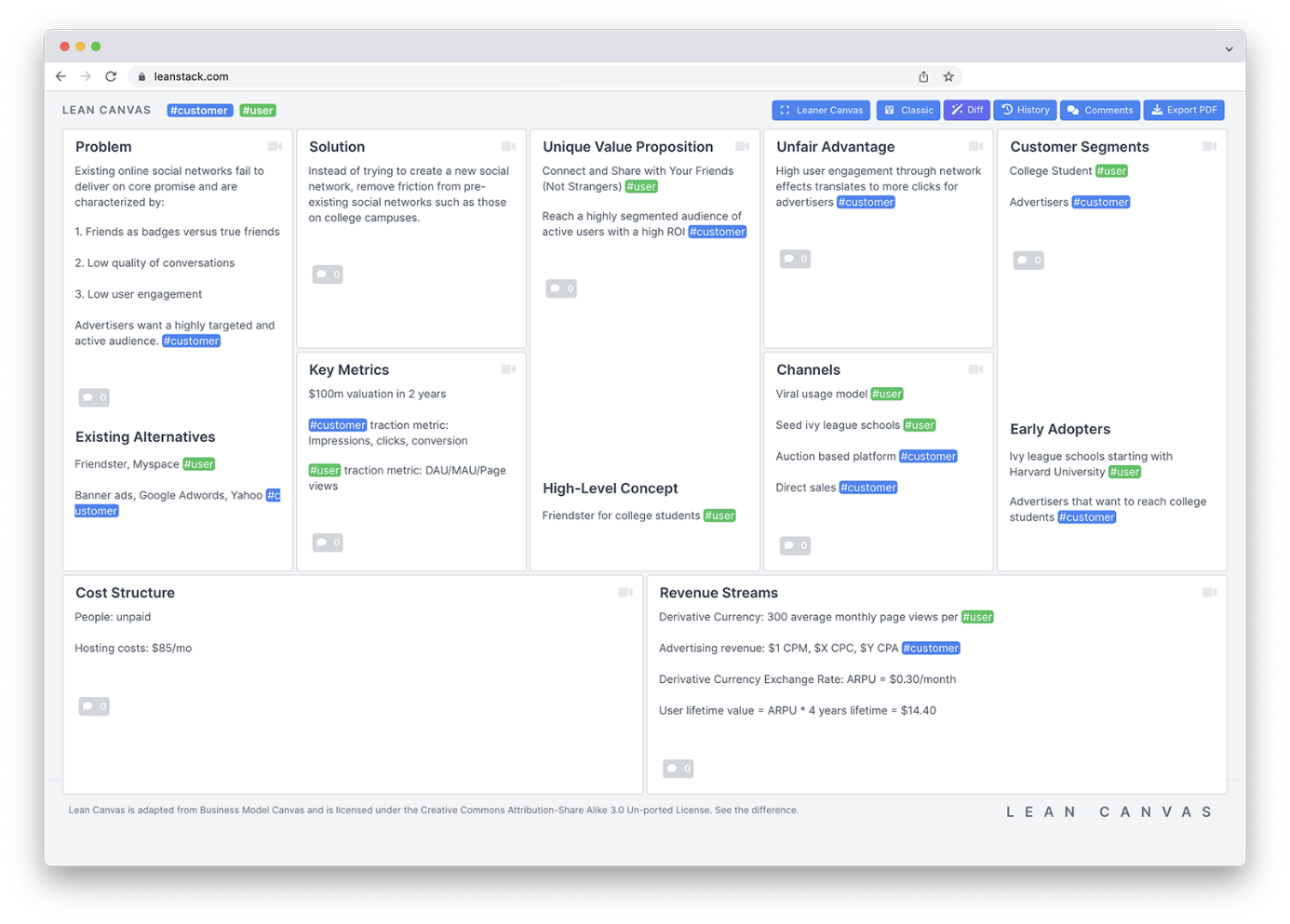
Таблица 1.2.  
 Бизнес модели според тип участници и вид на услугата.  
(адаптирано от автора по Филипова и др., 2017; Croll, 2013)

| **№** | **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]](#footnote-22), 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).

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.

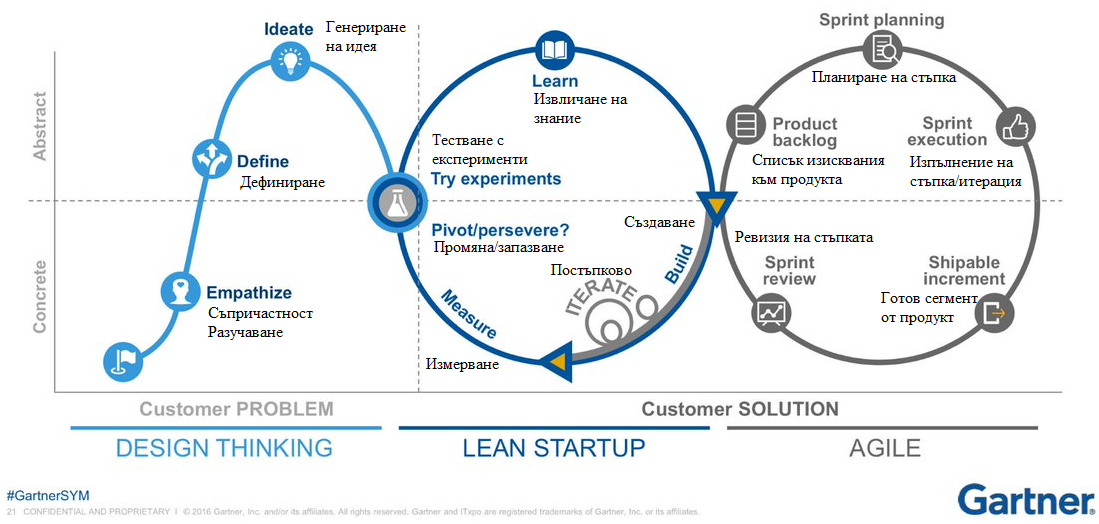
Важен параметър, който трябва да се има предвид, е източникът на изисквания за продукта, яснота и променливост на изисквания към продукта. Яснотата има значение, дали могат да се определят в началото, дори с риск изменение или напълно неясни и добивани от опита. Яснотата е важен параметър, тъй като някои 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**, койтое 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]](#footnote-23). 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]](#footnote-24).

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]](#footnote-25). 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 инкрементална разработка възприемат промяната като неизбежност и следват еволюционен адаптивен процес. Те използват обратна връзка с потребителя[[25]](#footnote-26), 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 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]](#footnote-27). 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:

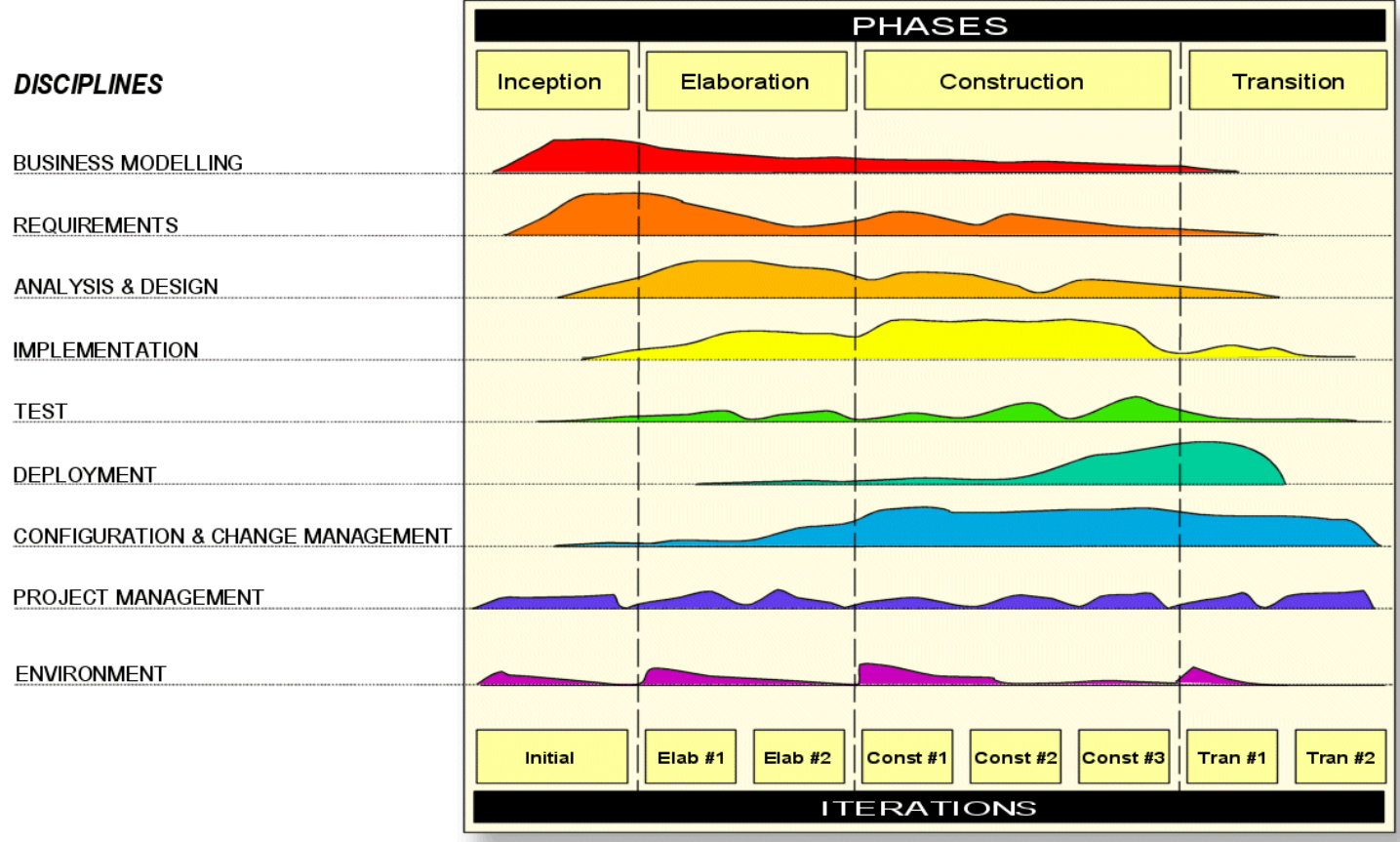
– Фазите са с итерации, при което с приоритет са високо рискови и високо стойностни елементи;

– 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 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]](#footnote-28). Взаимовръзка между фази, итерации и дисциплини е дадена на фиг. 1.3.



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

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]](#footnote-29). Процесът е в два вида стъпки – реализации[[29]](#footnote-30) 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 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” (М), „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]](#footnote-31), 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]](#footnote-32) 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 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]](#footnote-33). 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 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.

# 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 Калянджиев”, 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 Калянджиев.
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**

1. 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]
2. Икономи, П. (2018). Тайните на ефективното лидерство. Мениджър. <https://manager.bg/liderstvo/trayna-taynite-na-efektivnoto-liderstvo> [23.08.2022]
3. 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]
4. 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]
5. 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]
6. European Commission. (2019). SME definition. <https://ec.europa.eu/growth/smes/business-friendly-environment/sme-definition\_en> [01/10/2019]
7. Fowler M. (2019). Agile Software Guide. <https://martinfowler.com/agile.html> [10.12.2020]
8. Graham, P. (2012). Startup = Growth. September 2012, <http://www.paulgraham.com/growth.html> [01/02/2020]
9. 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>
10. IBM Cloud Education, (2020). Three-Tier Architecture, <https://www.ibm.com/cloud/learn/three-tier-architecture> [20.12.2021]
11. Iqbal, M. (2021). Uber Revenue and Usage Statistics Business of Apps. <https://www.businessofapps.com/data/uber-statistics/#6> [Accessed: 01/11/2021]
12. 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]
13. 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]
14. 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].
15. Robehmed, N. (2013). What Is A Startup? *Forbes*, <https://www.forbes.com/sites/natalierobehmed/%202013/12/16/what-is-a-startup/> [16.12.2020].
16. Tracy, B. (2015). Leading and Motivating. <http://keithlee.com/leading-motivating-by-brian-tracy/> [23.08.2022]
17. 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]
18. 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]

1. Имаме предвид т.нар. „икономика на споделянето”. [↑](#footnote-ref-2)
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." [↑](#footnote-ref-3)
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. [↑](#footnote-ref-4)
4. Под „предприемач” разбираме един или повече предприемачи, основаващи своя компания. [↑](#footnote-ref-5)
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). [↑](#footnote-ref-6)
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. [↑](#footnote-ref-7)
7. Гъвкавият дизайн дава възможност за лесна промяна и адаптация на продукт или услуга към различни предпочитания. [↑](#footnote-ref-8)
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. [↑](#footnote-ref-9)
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). [↑](#footnote-ref-10)
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. [↑](#footnote-ref-11)
11. Възложителят възлага официално, но детайлите по заданието е възможно да се определят съвместно и със специализираната помощ от друг отдел или служител. [↑](#footnote-ref-12)
12. Целенасоченото надхвърляне на предварителните очаквания на клиента е известно в маркетинга като „очароване на клиента“ (customer delight). [↑](#footnote-ref-13)
13. ISO/IEC/IEEE 15288:2015 Systems and software engineering - System life cycle processes. [↑](#footnote-ref-14)
14. Пенева, П., Александрова, Я., Армянова, М. Бизнес информационни системи: Учебник за дистанционно обучение. Варна: Наука и икономика, 2013, 274. с.27. [↑](#footnote-ref-15)
15. Някои автори (Larman, 2011) споменават дори изобретяване на нов продукт с процес на учене. [↑](#footnote-ref-16)
16. Цитираните групи и други допълнителни са дефинирани в (DeMarco & Lister, 2013) [↑](#footnote-ref-17)
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. [↑](#footnote-ref-18)
18. Според (Tracy, 2015) основните качества на лидера са: визионерство/проницателност, почтеност, смелост, реализъм и отговорност. [↑](#footnote-ref-19)
19. Планирани или възложени продукти, договорени проекти за разработка на продукти; [↑](#footnote-ref-20)
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. [↑](#footnote-ref-21)
21. Гъвкавата вътрешна среда е работна среда, която може лесно да се адаптира към външните (за организацията) условия. [↑](#footnote-ref-22)
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“. [↑](#footnote-ref-23)
23. Универсалността и разширението позволяват моделиране на всякакви системи, обекти и процеси с тях. [↑](#footnote-ref-24)
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. [↑](#footnote-ref-25)
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. [↑](#footnote-ref-26)
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). [↑](#footnote-ref-27)
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. [↑](#footnote-ref-28)
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. [↑](#footnote-ref-29)
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". [↑](#footnote-ref-30)
30. В литературата има редица примери за успешно прилагане и на подход DDD - Domain-Driven Design (Evans & Szpoton, 2015) при разработка на системи. [↑](#footnote-ref-31)
31. Терминът „продуктивен” е най-подходящ, защото ключовото е доставка на най-голяма ценност за клиент. [↑](#footnote-ref-32)
32. Терминът е Value Stream Map, идващ от производството и в контекста на настоящото изложение можем да го приемем като жизнен цикъл на продукта. [↑](#footnote-ref-33)