СОФИЙСКИ УНИВЕРСИТЕТ “СВ. КЛИМЕНТ ОХРИДСКИ”

ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА

КУРСОВА РАБОТА

Agile методологии за разработка на софтуерни решения

Изработили:

Велина Иванова Ташева, ф.н. M24077

Иван Пламенов Колев, ф.н. M25878

София, 2018г.

Съдържание

Какво е Agile

Agile са вид методологии за разработка на софтуерни решения, при които изискванията, спецификациите и решенията на проблемите се развиват и надграждат чрез съвместни усилия на самоорганизиращи се екипи, съставени от хора с различни фунцкии и техните клиенти.

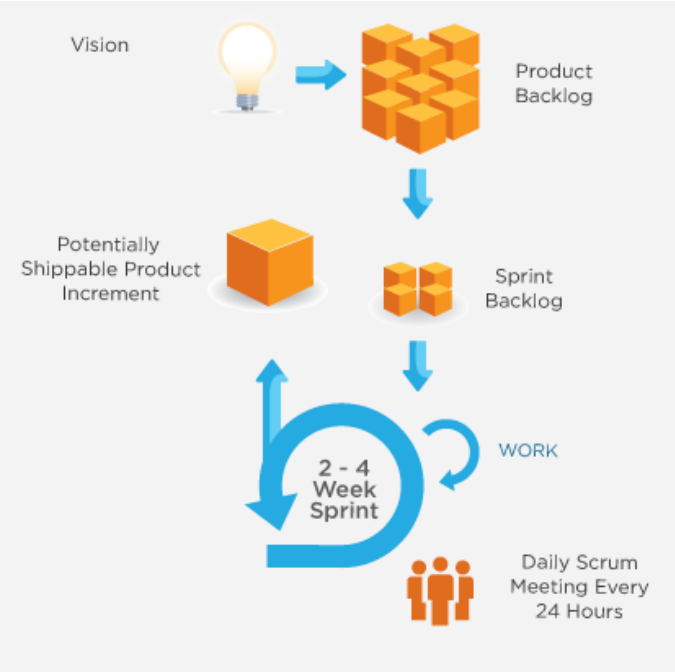
Agile методологиите насърчават адаптивното планиране, еволюиращата разработка на софтуера, ранното доставяне, итеративния подход и непрекъснатото подобряване. Те поощтряват бързото и гъвкаво реагиране на промени.

Терминът *agile* в контекста на разработването на софтуер е популяризиран през 2001 година в The Agile Manifesto. Принципите и ценностите, залегнали в този манифест, са извлечени от и подкрепят широка гама от методи за разработване на софтуер, включително Scrum и Kanban.

Agile методологиите се фокусират върху доставянето на високо-качествен работещ софтуер. Доставянето се извършва често и консистентно, като в същото време се минимизират загубите и се увеличават ползите за бизнеса.

Основни стъпки в Agile

Има много различни имплементации на agile методологии, но се забелязват някои основни стъпки, които се спазват при всички от тях:



Проучване

Важно е да бъдат разбрани вижданията и бизнеса на клиента, когато се започва нов проект. Проектите, които ще се разработват по agile методология започват с поредица от проучвателни сесии и изследвания за да бъдат разбрани целите на

клиента, предизвикателствата, естеството на бизнеса и т.н. В тези сесии взимат участие важни членове на екипа, включително клиента, ръководител на проекта, дизайнер и разработчик с цел да се подсигури общо виждане за продукта в целия екип.

Product Backlog

По време на периода на проучване, екипът работи заедно за да създаде product backlog от високо ниво. Това е списък с всички фунционалности, които биха били полезни на клиента и неговите потребители. Собственикът на продукта работи заедно с клиента за да приоритизира тези фунционалности, като по този начин се определя реда, в който те ще бъдат разработени, тествани и доставени. Фактът, че клиентът определя приоритетите позволява на екипа да се фокусира върху доставката на фунционалностите, които носят повече стойност на клиента.

Итерации

След като се подсигури, че екипът разбира визията на клиента и има създаден backlog с фунционалности, екипът започва разработката и доставянето чрез серии от ограничени във времето итерации, наречени спринтове. Всеки спринт е с продължителност между една и четири седмици, в зависимост от размера и продължителността на проекта. В края на всеки спринт се доставя пакет от работещи фунционалности, които са част от целия backlog.

Продължаване на цикъла

При нужда се провеждат допълнителни спринтове, за да се доставят нови функционалности. При всяка следваща итерация се включват обратна връзка от предишни итерации и бета тестове на потребителите.

Всеки следващ спринт е едновременно итеративен, т.е. осигурява подобрения в работата, завършена в предишни спринтове и надграждащ, т.е. добавя нови функционалности към системата.

Внедряването на agile методологиите в процеса на разработка на софтуер може да има сериозно влияние върху цялостния успех на проекта. Обратната връзка и подобренията се случват достатъчно често, така че да бъдат поправени малките нередности преди те да се превърнат в големи проблеми. Комуникацията по време на целия процес също се подобрява чрез agile подхода към управлението на проекти. При този вид методологии постигаме удовлетворение на клиентите чрез бърза доставка на полезен софтуер. Agile ни позволява промяна на спецификациите, дори и в късните фази на проекта, както и устойчиво развитие, което успява да поддържа постоянно темпо. Проектите се изграждат около мотивирани хора, на които се има доверие и се обръща непрекъснато внимание на техническото съвършенство и добрия дизайн.

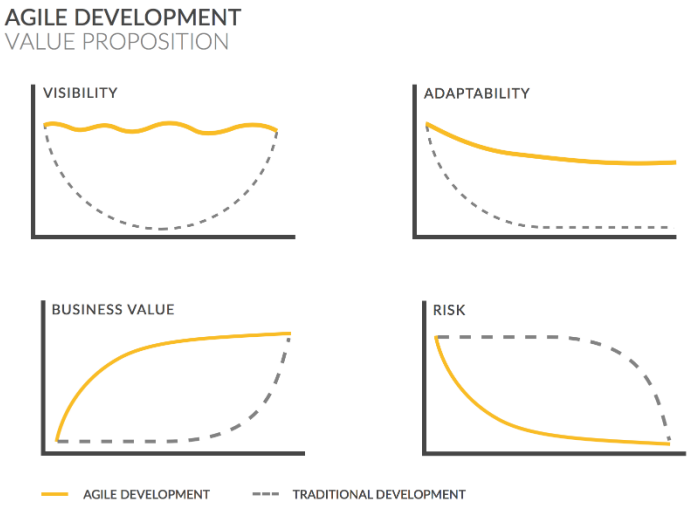
Като цяло agile методологиите предоставят елегантен и ефективен модел за успешното разработване на софтуер.

Сравнение между agile и традиционен метод

Преди да разгледаме предимствата на agile управлението на проекти, нека направим сравнение между традиционен и agile начин на работа.

При разработката на софтуер често говорим за "традиционен модел" на работа, който се отнася до “waterfall” модела. Този модел е много различен от agile, тъй като не е итеративен. Waterfall е по-скоро процес, при който може да се види напредъка "протичащ" през различни фази. Всъщност това е последователен модел, който обикновено започва от анализ на изискванията, преминава през проектиране, разработване, тестване и поддръжка.

Изображението по-долу илюстрира разликата в стойността, която предоставят двете методологии и как agile осигурява видимост и адаптивност в началото на процеса и значително намалява рисковете по време на проекта.



Agile процесът отговаря перфектно на нуждите на клиентите. По време на целият цикъл се насърчава тяхното участие, което предоставя видимост и прозрачност, показвайки реалният прогрес на проекта.

Позли от прилагане на Agile

Ползите от прилагането на agile са много и разнообразни. Те помагат на екипите да се справят с обичайните проблеми от процеса на разработване на софтуер, като хаос в резултат на непрекъснато променящите се изисквания, подценяване на необходимото време за разработване и тестване, подценяване на ресурсите и разходите.

Удовлетвореност на клиентите

Agile предоставя множество възможности за ангажиране на заинтересованите страни – клиента и екипа - преди, по време и след всеки спринт. Чрез включването на клиента във всяка стъпка от процеса на разработка на проекта има висока степен на сътрудничество между клиента и екипа, което предоставя повече възможности на екипа да разбере наистина визията и потребностите на клиента. Предоставянето на работещ софтуер на кратки интервали и на ранен етап увеличава доверието на клиента в способността на екипа да предоставя

висококачествен софтуер и го насърчава да бъде по-активно ангажиран в проекта.

При agile собственикът на продукта винаги участва активно в процеса на разработка, и по този начин напредъкът в развитието е много ясно видим. Също така гъвкавостта и възможностите за промени са от изключително значение за клиентите. В края на всеки спринт се представят новите фунционалности, което позволява много рано да бъдат идентифицирани несъответсвията в очакванията на клиентите и разработения софтуер. Тъй като доставянето на продукта става бързо и често, клиента получават ранен достъп до софтуера.

Всички тези фактори предполагат ангажираност и удовлетвореност на клиентите.

Контрол и прозрачност

Agile подходът предоставя уникалната възможност на клиентите да участват във всички аспекти на проекта – от приоритизиране на фунционалности и планиране на итерации до често преглеждане на работещия софтуер. По този начин те имат контрол върху това какви трябва да бъдат следващите стъпки от жизнения цикъл на продукта.

Това предоставя прозрачност на процеса, видимост върху различните трудности по време на разработването и ясна визия за напредъка на продукта. Но това също така изисква клиентите да разбират, че виждат продукта в процес на разработка, т.е. е възможно да попаднат на различни дефекти, но въпреки това те получават всички останали ползи от това да виждат и използват софтуера на ранен етап.

Предвидимост и намален риск

Новите фукнционалности се доставят бързо и често, с висока степен на предвидимост, поради факта, че се работи по план с фиксирани спринтове с дължина от една до четири седмици. Това също така предоставя възможност да бъде направен бета тест на софтуера по-рано от планираното в случай, че това ще донесе по-голяма стойност на бизнеса. Тъй като всеки спринт е с фиксирана продължителност, цената за него е предвидима и ограничена до количеството работа, която може да бъде извършена от екипа в рамките на това време. В резултат на фиксираната дължина на спринта, в комбинация с оценките, които са предоставени на клиента преди всеки спринт, той може по-лесно да разбере приблизителната цена на всяка функционалност. Това подобрява взимането на решения относно приоритета на задачите и идентифицирането на необходимостта от допълнителни итерации.

Agile технологиите на практика премахват шансовете за абсолютен провал на проекта. Още в самото начало, след приключване на първия спринт клиентът разполага с работещ продукт, така че нито един agile проект не се проваля напълно. Разработването под формата на спринтове подсигурява, че времето между първоначалната инвестиция в проекта и установяването дали проектът ще бъде успех или провал е сравнително кратко.

Ранното генериране на приходи при самофинансиращи се продукти, позволява на клиентите да започнат проект с малки първоначални капитали. Agile дава свобода, когато трябва да бъдат направени промени в изискванията. Те могат да бъдат реализирани на много ниска цена, поради честотата, с която се правят нови подобрения.

Agile методологиите позволяват адаптиране към нуждите и предпочитанията на клиента по време на процеса на разработване. Agile обикновено използва user stories, в които се дефинират критерии за приемане на работата (acceptance criteria). Обикновено характеристиките на продукта се дефинират именно чрез тези критерии, които са изцяло фокусирани върху бизнеса.

Посредством фокусирането на изискванията върху нуждите на реалните клиенти, всяка фунционалност добавя стойност за потребителите. Това също предоставя възможност на потребителите да правят бета тестове на софтуера след всяка итерация, което осигурява ценна обратна връзка в ранните етапи на проекта. Благодарение на това екипът има възможност навреме да реагира и да направи промени, ако това е необходимо.

Високо качество

Разбиването на проекта на по-малки, управляеми единици позволява на екипа да се съсредоточи върху висококачествено разработване, тестване и съвместна работа. Също така, чрез честата доставка на работещ софтуер и честото провеждане на тестове и прегледи по време на всяка итерация, качеството се подобрява, тъй като бързо се откриват и поправят различните дефекти и рано се идентифицират несъответствията в очакванията.

При процеси, които следват принципите на agile методологиите тестването е интегрирано в самия цикъл на разработка, което означава, че има редовни проверки, които верифицират, че продуктът работи по време на разработката. Agile позволява на собственика на продукта да прави промени, ако е необходимо, и съответно екипът е осведомен в случай, че има някакви проблеми.

Провеждането на спринт ретроспектив срещи, позволява на agile екипа непрекъснато да подобрява процесите, качеството, комуникацията и начина си на работа. По време на тези срещи се идентифицират различни проблемни области в процеса на работа и се дефинират съответните стъпки за справяне с тези проблеми. Това позволяване непрекъснато развитие и подобряване на работата и качеството.

Фокусиране върху необходимостите на бизнеса

Тъй като клиентът определя приоритетите на необходимите функционалности, екипът навреме разбира кое е най-важното за бизнеса на клиента и съответно може да достави модулите, които осигуряват най-голяма бизнес стойност.

По време на всяка итерация екипът трябва да бъде изцяло фокусиран върху това да достави подмножеството от функционалности, които са били одобрени и съгласувани с клиента в началото на итерацията. Все пак през това време клиентът има възможността да променя приоритетите и да усъвършенства цялостната концепция на продукта. Нови или променени изисквания могат да бъдат планирани за следващата итерация, като по този начин се предоставя възможност за въвеждане на промени в рамките на няколко седмици.

Практиката показва, че интегрирането на agile методологиите в процеса на разработка на софтуер предоставя решения, доставени навреме, с по-високо качество и с по-висока степен на удовлетвореност на клиентите.

Agile е мощен инструмент за разработка на софтуер, който не само предоставя много ползи на разработващия екип, но и осигурява редица важни бизнес ползи за клиента. Agile помага на екипите по проекта да се справят с много от най-често срещаните проблеми в процеса на разработка (като разходи, предвидимост на плана и разширяване на обхвата) по по-контролиран начин.

Чрез реорганизиране и ревизиране на дейностите, свързани с разработването на софтуер, agile постига същите цели, но по по-лек и по-целенасочен начин.

The Agile Manifesto

През февруари 2001 г. е създаден agile манифестът за разработка на софтуер.

Принципи, заложени в agile манифеста:

1. Нашият най-висок приоритет е да задоволим нуждите на клиента чрез ранно и постоянно доставяне на стойностен софтуер.
2. Приветстваме променящите се изисквания, даже и в напреднал стадий на разработка. Agile процесите прегръщат промяната в името на конкурентното предимство на клиента.
3. Често доставяне на работещ софтуер - между две седмици и два месеца - с предпочитание към по-кратките срокове.
4. Хората на бизнеса и разработчиците трябва да работят заедно ежедневно през цялото време на проекта.
5. Проекти се изграждат от мотивирани личности. Дайте им средата и подкрепата, от които се нуждаят и им гласувайте доверие, че ще свършат работата.
6. Най-ефективният и най-ефикасен метод за предаване на информация към и вътре в екипа от разработчици е разговорът лице в лице.
7. Работещият софтуер е основната мярка на прогреса.
8. Agile процесите насърчават непрекъснатата разработка. Спонсорите, разработчиците и потребителите трябва да могат да поддържат постоянен ритъм безсрочно.
9. Постоянното внимание към техническо усъвършенстване и добрият дизайн подобряват гъвкавостта.
10. Простотата - изкуството да се максимизира работата, която не е нужно да се върши - е от изключително значение.
11. Най-добрите архитектури, изисквания и дизайни произлизат от самоорганизиращи се екипи.
12. През равни интервали от време, екипът обсъжда как да стане по-ефективен, след което настройва работата си в съответствие с взетото решение.

Видове Agile методологии

Agile събира в себе си различни подходи.
Всички Agile подходи (методологии) практикуват бизнес анализ, но само няколко изрично определят такава роля. Основната характеристика на всеки гъвкав подход е привеждането му в съответствие с ценностите и принципите на “Agile Manifest“-а. Един agile екип може да прилага един или комбинация от няколко подхода, което дава възможност за по-ефективна работа, като се има предвид вида на проекта и работната среда.

Extreme Programming (XP)

Този подход се фокусира върху процесите свързани с техническото разработване и се характеризира с програмиране по двойки, Test Driven Programming и други. XP често се използва в комбинация с някои от Agile мениджмънт подходите.

XP практиките са базирани на набор от ценности и водещи принципи.

Ценности:

- *Комуникация* - тя е вкоренена в XP – много от практиките не могат да бъдат извършени без комуникиране. Задачи като планиране и оценяване трябва да бъдат изпълнени заедно и са базирани на вербален диалог.
- *Обратна връзка* – процес, който се подобрява чрез промени, трябва да получава обратна връзка, винаги когато е възможно. Обратната връзка в XP се случва на много нива. При метод на работа на къси итерации екипите имат възможност за постоянна обратна връзка от страна на клиента.
- *Опростеност* - тази ценност се обобщава с фразата - “направи го по най-простия начин, така че да работи“.

Принципи:

- Бърза обратна връзка
- Приемане на простотата
- Надграждащи се промени
- Възприемане на промените
- Качествена работа

Практики:

- *Игра на планиране* - основната дейсност по планиране в XP се случва на ниво release и итерация.
- *Малки release-и* - XP екипите се стремят да създават release-и с работеща функционалност възможно най-бързо
- *Метафора* - създаване на общо разбиране от страна на целия екип за системата, нейните елементи и техните взаимоотношения.
- *Прост дизайн* - Дизайнът приложен от разработчиците трябва да бъде възможно най-елементарен, изпълняващ само изискванията, необходими за покриване на функционалностите.

- *Тестване* - всчики сторита трябва да бъдат тествани автоматизирано. Автоматизираните функционални тестове, с критерии за приемане, се идентифицират като част от всяко user story.
- *Рефакториране*
- *Програмиране по двойки* - двама програмисти на един компютър
- *Колективна собственост* – програмистите могат да подобряват всяка част от кода по всяко време.
- *Continuous Integration*
- *40 часова работна седмица*
- *На страната на клиента* - реалният клиент трябва да работи заедно с екипа от програмисти.
- *Стандарти за програмиране*

Scrum

Scrum е най-популярният agile метод, достаигайки 72% популярност. Scrum е подход за управление на процеси, базиран на емпиричен контрол на процесите. Работата се извършва в серии от итерации с фиксирани дължини, наречени sprint-ове. Те могат да продължават месец или по-малко. В края на всеки спринт екипът трябва да предостави работещ софтуер с достатъчно добро качество, за да може той да бъде предоставен на клиента. Ключови елементи са прозрачност, инспекция и адаптиране.

Прозрачност: дава ясен поглед на всички stakeholder-и и клиенти, както и на всички заинтересовани лица.

Инспекция: проверява своевременно колко добре напредва продукта.

Адаптиране: корекции на процеса, с цел да се минимизира отклонението от крайната цел.

Методът се състои от набор от стойности, роли, дейности и артефакти, които формират цялостният подход за доставяне на продукт.

Scrum роли

Scrum master – той е отговорен за ценностите, принципите и практиките. Той улеснява въвеждането на скръм, чрез непрекъснато обучение и напътствия към скръм екипа. Различава се от традиционната роля, отговорна за на управление на проекти, по това че не упражнява контрол и няма власт над екипа. Той действа наставнически като треньор. Той ръководи процеса и улеснява използването на скръм в екипа, с цел да се позволи самоорганизация, придобиване на умения и високи постижения. Той работи и на ниво организация, като се стреми да намалява пречките за приемането на специфичен за организацията скръм подход.

Отговорности:

- *Успех* на scrum порцеса.

- *Установяване* на scrum практики и правила, защита на екипа и премахване на препятствия.
- *Гарантира* че скръм екипа е напълно функционален и продуктивен.
- *Осигурява тясно сътрудничество* във всички роли и функции.
- Гарантира че скръм процеса е следван, включително ефективните дейли скръм срещи, спринт ревюта, ретроспекции и срещи за планиране.
- Води и тренира организацията при приемане на скръм.
- Подпомага product owner-а и намира ефективни техники за мениджъмънт на product backlog-а.

Product owner - упълномощена централна фигура в ръководството на продукта.

Отговорности:

- Определянето и приоритизирането на product backlog
- Увеличване на стойността на продукта като цяло
- Идентифициране на относителната стойност на product backlog елементите.
- Споделя визията на бизнеса с екипа и визията на екипа с бизнеса.
- Дефинира наличен бюджет.
- Дефинира цели за спринтовете и release-ите.
- Участва в срещите за планиране на спринтове и release-и.
- Разработва product backlog елементите точно на време
- Приема product backlog елементите
- Приема спринт/release.
- Преценява кога да се направи release.
- Определя характеристиките на продукта чрез product backlog елементите.
- Прави график за разработка на product backlog
- Променя product backlog и приоритизира всеки спринт, когато е необходимо.
- Осигурява възвръщане на инвестицията.

Екип разработчици - фундаментално различен спрямо традиционните екипи като в waterfall например.

Отговорности:

- Работят с product owner, за да се гарантира, че product backlog-а е разбран и реализиран адекватно.
- Дефинира архитектурата, като в същото време поддържа качеството на високо ниво.
- Самоорганизиращ се екип, с припокриващи се функции и без преопределени роли.
- 7 +/- 2 души, всички с умения необходими за изпълняване на product backlog-а
- Много комуникация лице в лице.
- Отговаря за организирането на задачаите и ангажиментите.
- Има власт да прави каквото е необходимо за да си изпълни ангажимента.
- Демонстрира резултатите от работата си пред product owner-а и stakeholder-ите.

- Има право да прави всичко, съобразено със стандартите, за да достави продукта и приетите изисквания в рамките на спринта/release-а.

Дейности и артефакти:

Планиране на спринтове - преди всяка дейсност по разработване, всеки спринт започва със сесия, в която екипът планира своята работа. По време на тази среща, позната като планиране на спринт, скръм екипът натрупва знания и product owner-а дефинира максимално product backlog-а. Scrum екипът одобрява добавените неща в backlog-а.

Daily scrum - синхронизираща среща с комуникация лице в лице, която е съществена в Agile. Екпът разработчици участва в повтарящи се срещи, които се провеждат ежедневно в съгласувано време. Времетраенето на тези срещи е до 15 минути. Тези срещи позволяват на членовете на екипа да разберат текущия прогрес, да синхронизират дейностите си и да споделят информация за тях. Стандартния план на дневния скръм е улеснен от scrum master-а. Планът се състои от 3 прости въпроса, на които всеки отговаря:

- Какво направих вчера?
- Какво смятам да направя днес?
- Имам ли някакви пречки?

Спринт ревю - в хода на всеки спринт, екипът следва подход към разработването, съсредоточен върху постигането на целта на спринта чрез ангажираност и самоорганизация. Всеки спринт завършва с две допълнителни дейности за инспектиране и адаптиране: преглед на спринта и ретроспекция. Особено в срещите за спринт ревю, екипът има отлична възможност да подобри комуникацията между своите членове и останалите членове от организацията.

Ретроспекция – последната дейност в scrum преди края на спринта - от съществено значение за определяне на възможностите за предствавяне и подобряване на сътрудничеството и вземането на решения. Също така укрепва взаимоотношенията между екипа.

Kanban

Не изисква фиксирани итерации (повторения). Работата се движи като процес на развитие през непрекъснат поток от дейсноти. Ключова функция е да се ограничи размерът на работата в процесите. Екипът работи само върху фиксиран брой задачи във всеки един момент. Екипът може да започне работа по нова задача, само когато тя е в списъка за от следващи задачи и предната задача е завършена.

Kanban е подход на непрекъснато подобряване на продукта, който се съсредоточава върху гладкостта и бързия поток на работа. Организационно канбан е използван за всякакви креативни и научни дейности. Използва се при редакця на видео, рекламиране, подбор на персонал, финанси, продажби маркетинг и други.

Канбан не е Agile метод за софтуерна разработка. Той е алтернативен път към усотойчивост или метод за повишаване на устойчивостта на услугите, насочен

към подобрение на устойчивостта на бизнеса. Също така, той е описан като метод за повишаване на организационната устойчивост.

Практики в ядрото на Kanban:

- Визуализиране на работата, работния процес и бизнес рисковете. Визуализира се всяка стъпка по веригата - от идеята до готовия софтуер.
- Ограничаване на работата в прогрес (WIP). Ограничава се количеството задачи, по които се работи във всеки един момент.
- Политиките стават изрични - целият мениджмънт, риск мениджмънт и политиките за процеса, които се прилагат, трябва да бъдат документирани.
- Измерва се и се менажира работния процес – преходите между стъпките в процеса се наблюдават и измерват. Това дава картина на историята на работния процес.
- Имплементират се цикли на обратна връзка.
- Подобряване на сътрудничеството и развитието.

Kanban спрямо Scrum

Kanban често се счита, че контрастира спрямо Scrum.

Kanban	**Scrum**
Подход за подобряване на предоставянето на услуги, използвайки еволюционно усъвършенстване	Подход към комплексното разработване на продукти
Няма дефинирани роли или процеси	Стандартни роли и процеси, ясно дефинирани
Подобряващ, започващ с каквото съществува и итерира, уважава това което има	Революционен, действа се както скръм изисква без значение какво е било преди това
Ограничава работата в процес и позволява времето да варира	WIP никога не надминава количеството, което може да бъде завършено от екипа в спринта. Ограничава времето и варира работата за да пасне във времето

Други agile методологии

Crystal Clear

Част от семейството е crystal clear методологията, която е дефинирана и базирана върху премахване на бюрокрацията, добавя твърдост и цвят. Твърдостта се отнася до бизнес критичност или потенциал за причиняване на вреда, което се равнява на по-голяма строгост и предвидимо планиране, тъй като критичността се увеличава. Цвета се отнася до тежестта на проекта в редица измерения, включително броя на необходимите хора и рискови елементи в проекта.

Disciplined Agile Delivery (DAD)

Процес за взимане на решения, който вкючва идеи от различни други Agile подходи. Той е предназначен да поддържа проекта от начало до край. DAD практиките не са задължителни и позволяват на екипите да персонализират своите подходи и итерации.

Еволюционен меницжъмнт на проекти (EVO)

EVO е метод за мениджмънт на проекти. Разработва и доставя системата поетапно. Той се фокусира върху количествената стойност за много stakeholder-и и планира прираст, базиран на доставяне на тази стойност (която може да бъде измерена). Използва таблици за оценка на въздействието, като формална техника за оценка на решението, и за възможността да бъде доставен продуктът на множество stakeholder-и за дадена цена.

Dynamic Systems Development Method (DSDM)

Подход, който се фокусира върху фиксирана цена, качество и време в началото. Непредвидените разходи се поемат чрез вариране на крайните функционалности. За менажиране на работата се използват MoSCoW приоритизация и кратки периоди с ясно дефинирани резултати.

Feature Driven Development (FDD)

Фокусира се на ценните функционалности за клиента при разработването на софтуер. Като пример е следването на високо равнище на обхват на приложението. Прави се списък с функционалности и всичко от планиране, дизайн до разработване се изпълняват на база на този списък.

Scaled Agile Framework® (SAFe™)

Методология за имплементиране на agile практики с голям бизнес обхват. Подчертава индивидуалните роли, екипи, дейности и артефакти, необходими да се увеличи agile от екип до програма към голямо бизнес равнище.

Заключение:

Кои Agile методологии бихте препоръчали за използване в практиката на вашата фирма? Защо?

Както видяхме по-горе всички Agile методологии отговарят на различни нужди и изисквания на клиентите. Някои на прозрачност на работата, някои на бързина на завършеност на готовия продукт, други ограничават цената, като варират с качеството и функционалността на продукта.
Бихме препоръчали на фирмата да бъде наистина гъвкава, да приоритизира добре изискванията на клиентите си, за да избере тази методология или комбинация от методологии, които най-много отговарят на изискванията на клиентите ѝ.

Ресурси

1. https://apiumtech.com/blog/agile-project-management-benefits/
2. https://www.seguetech.com/8-benefits-of-agile-software-development/
3. http://agilemanifesto.org/
4. https://www.seguetech.com/what-is-agile-software-development/
5. BABOK v3 A Guide To The Business Analysis Body Of Knowledge
6. Agile Foundations Principles, practices and frameworks by Peter Measey and Radtac