Доставка на софтуер - особености и процеси

Тема №1

Място на доставката в цялостния жизнен цикъл на софтуера

Съдържание

1 Въведение

Курсът "Доставка на софтуер - особености и процеси" обръща огромно внимание върху тази малка, но изключително важна част от целия процес за създаването на софтуерни решения и тяхната пазарна реализация. Но, за да имаме еднакво разбиране какво е "доставка" и каква е цялостната картина (обкръжението), в която тази доставка се случва, ние ще започнем курса именно с тези въпроси.

В настоящата тема ще си припомним или ще се запознаем за първи път с идеята за жизнен цикъл на софтуера. След което ще изясним понятието доставка и ще го поставим във въпросния жизнен цикъл, за да определим кога точно се случва. А дали доставката не оказва съществено влияние върху множество етапи на жизнения цикъл? Дали ясно видимото и проявление в него не е само една малка част от присъствието на доставката в жизнения цикъл? Тази и следващите теми от курса целят именно да отговорим на тези въпроси и да се запознаем в детайли с важността и обхвата на доставката на софтуер.

2 Какво е жизнения цикъл на софтуера?

Жизненият цикъл на софтуера поставя рамка за систематизиран подход, покриващ разработката на софтуерен продукт от появата на идеята, през създаването му и появата му на пазара до поддръжката. Той е съставен от редица фази, всяка от които има своите задачи, цели и артефакти, които продуцира.

По отношение на фазите, има огромно разнообразие по отношение на брой, наименование, паралелност, цикличност, състав и много други. Това се дължи на редица фактори - разнообразни модели и методологии, разлики между авторите, разлики появили се в практиката, вътрешно-фирмени процеси, естество на продукта, качество на процесите и други.

Въпреки това, за целите на курса, основно можем да разгледаме следните дейности:

- **Извличане и анализ на изискванията** - фаза, в която се правят необходимите проучвания с цел да се установят с максимална пълнота и точност изискванията, които клиенти, потребители и приложната област / заобикалящата среда имат към продукта или услугата, която искаме

да разработим. Трябва да се изясни какъв е очаквания резултат от нашата дейност - какви са целите на продукта, за кого е предназначен и други. На база това може да се определи и планира следващата работа. Но не е достатъчно само да съберем необходимата информация. Тя трябва внимателно да се анализира. Да се открият неясноти, противоречия и други дефекти. Изискванията трябва да се приоритизират. Необходимо е да се оцени и до колко те са реалистични, особено взимайки предвид ресурсни ограничения като време и бюджет. Резултатите от тези задачи следва да се документират по ясен начин, така че да се използват от следващите фази.

- **Дизайн** - фаза, която често се определя само като изготвяне на дизайн на софтуерната архитектура. И докато това действително е част от тази фаза и то такава с огромно значение, към дизайна е редно да се включат и други не по - малко важни задачи. Такива са проектиране на потребителското изживяване (как се взаимодейства с продукта или услугата), както и тясно свързаните дейности на дизайн на графичен потребителски интерфейс, дизайн на програмните интерфейси (APIs), дизайн на конзолните интерфейси (CLIs). Не всеки продукт има нужда от всяка от тези дейности, но в случай, че са приложими е важно да се изпълнят качествено. Често може да се гледа на тези дейности като част от архитектурата и работа на архитектите. Такова схващане обикновено води до негативни последствия. От изключително значение при проектирането на потребителското изживяване е валидацията на направените дизайни и прототипи с реалните целеви потребители на продукта. Тази обратна връзка е критична, тъй като без такава валидация, създаването на качествено потребителско изживяване практически граничи с невъзможното. Очакван резултат е документация на всички дизайни, включващи диаграми, описания, прототипи и друга информация, която би била полезна за следващите фази.

- **Имплементация** - имплементация на всички компоненти на системата (frontend, backend и каквото още е необходимо), според спецификациите от фазата на дизайн. Резултатът е имплементираната система - код, скриптове, SQL заявки и всякакви други артефакти, изграждащи софтуерния продукт.

- **Тестване** - фаза, в която имплементирания софтуер бива проверен за дефекти. Това се прави на база положителни сценарии, симулиране на невалидни сценарии и възникнали грешки, проверки на база нефункционални изисквания. Целта е да сме сигурни, че разработения продукт отговаря на дефинираните изисквания.

- **Внедряване** - фазата, която прави продукта фактически достъпен за клиентите и потребителите. Реализацията ѝ зависи много от типа софтуер, но в общия смисъл софтуера става или достъпен за изтегляне и/или инсталиране за клиентите/потребителите, или се инсталира от производителя на някаква среда (било то на клиента или на производителя), към която се осигурява достъп с цел използване на софтуера.

- **Поддръжка** - фаза, в която се отстраняват дефекти, открити след началото на експлоатацията на продукта, предоставя се консултация относно употреба, конфигурация и други аспекти, може и да се предоставят малки подобрения към продукта (тук трябва да се внимава и често е по - добре да се предоставят чрез нова версия под формата на ново преминаване през фазите).

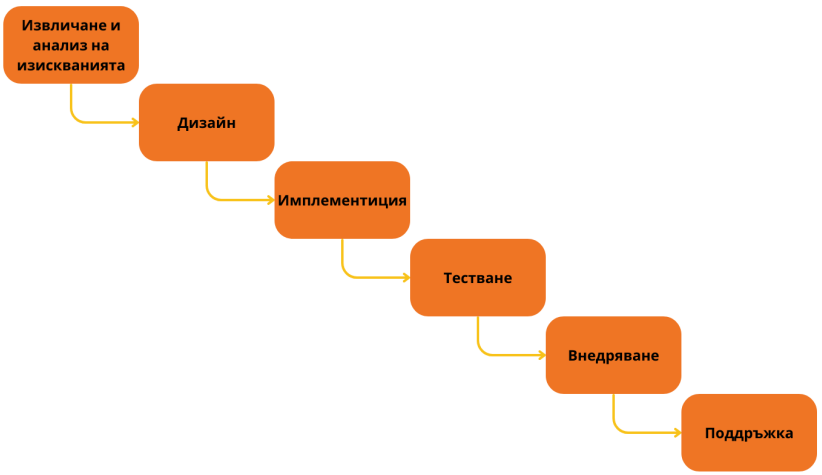
Наличието на ясен модел или методология за жизнения цикъл на софтуера е критично за успехът и постоянството при разработката на софтуерни продукти и услуги. То спомага да се управляват ефективно рисковете, да се минимизира възможността за допускане на пропуски, да се управляват по-лесно процесите, да има съвместимост между различните екипи и продукти в една организация и други.

На кратко, с цел пълнота, ще се спрем и на основните характеристики на някои от популярните методологии и модели.

2.1 Модел на водопада (waterfall)

Моделът на водопадът е структуриран подход за разработка на софтуер, където проектът преминава през последователни и разделени фази – от изискванията и дизайна до разработването, тестването и пускането в експлоатация. Всяка фаза завършва преди да започне следващата, като предоставя ясна структура и документация на процеса. Този модел е подходящ, когато изискванията са добре определени и не се очакват съществени промени по време на разработката. Моделът не предвижда

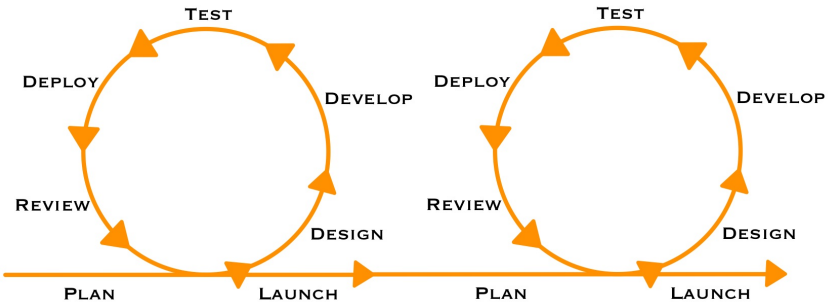
обратна връзка, итерации и междинни версии и като цяло е с ограничена гъвкавост. И въпреки това, моделът на водопада е силно разпространен и до ден днешен, особено в среди с по-слабо ниво на иновации.



Фиг. 1: Модел на водопадът

2.2 Гъвкави методологии (agile)

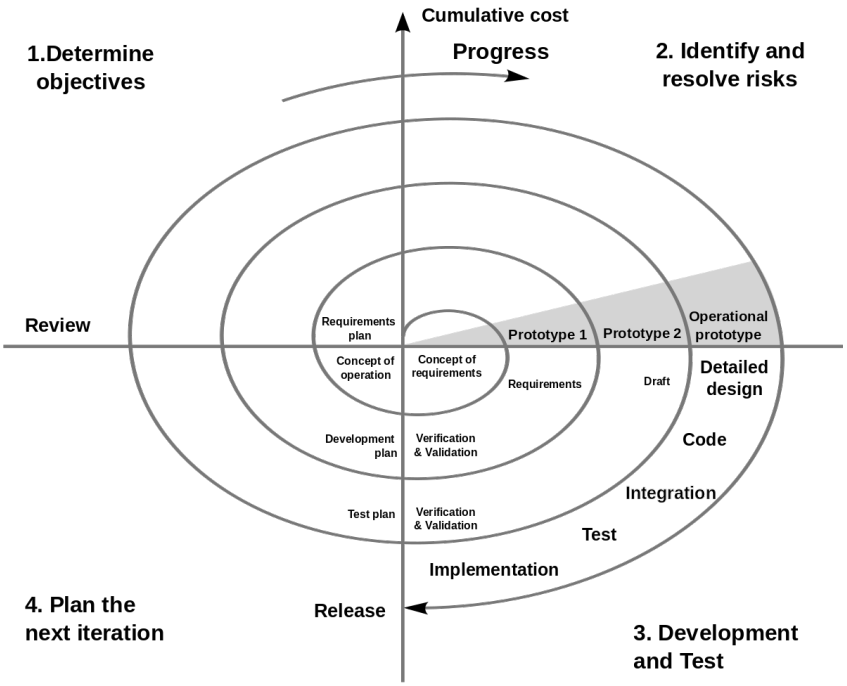
Гъвкавите методологии са подходи за разработка на софтуер, които се основават на гъвкавост, итерации и сътрудничество. Те са създадени, за да се справят по-добре с променящите се изисквания и да позволят по-голяма адаптация по време на разработката. Вместо последователни стъпки, гъвкавите методологии се фокусират върху гъвкав процес и постоянно сътрудничество със заинтересованите страни. Някои примери са SCRUM, Kanban, XP. Гъвкавите методологии се отличават със своя подход, фокус и инструменти, но общата им черта е способността да се адаптират към променящите се изисквания и да предоставят стойност на клиента по възможно най-ефективен и устойчив начин.



Фиг. 2: Гъвкави методологии

2.3 Спираловиден модел (spiral)

Спираловидният модел е методология за разработка на софтуер, която комбинира елементи от адаптивността на гъвкавите методологии със стратегическото планиране и рисковете на традиционните подходи. Този модел се нарича "спираловиден", защото процесът на разработка се изпълнява под формата на итерации, като на всяка се изпълняват определени дейности. В основата на спираловидния модел стоят четири основни фази, които се изпълняват във всяка итерация: Планиране, Анализ на рисковете, Разработка, Оценка на резултата и подготовка за следваща итерация. Спираловидният модел е подходящ, когато проектите са сложни, с високи рискове и изискванията на клиента са склонни да се променят. Този модел спомага предотвратяването на проблеми и грешки, но също би предизвикал нуждата от по-големи инвестиции свързани с управление и планиране.



Фиг. 3: Спираловиден модел

3 Понятието "доставка на софтуер"

Доставката на софтуер е изключително важна. Това е процесът, в резултат на който клиентите и потребителите получават възможност да боравят с нашия софтуерен продукт. Но най - важното, което трябва да отбележим, е че тук не говорим само за физическият достъп (това да могат да го достъпят на уеб адрес, да могат да го инсталират от Google Play, да могат да го изтеглят и инсталират на компютъра си и т.н.). Това със сигурност влиза в доставката и е най - очевидната част от нея. Но в допълнение на това, чрез процеса по доставка на софтуер ние искаме да подпомогнем до максимална степен успешната реализация на софтуерния ни продукт като подсигурим, че той се предоставя в съответствие с правните изисквания (нормативната уредба), изискванията на приложната област (индустриални изисквания), утвърдени стандарти и организационните изисквания (на клиента и на производителя). Също така този процес цели самото предоставяне на софтуера да е систематично и чрез установени и стандартизирани канали, намалявайки до минимум възможността за грешки. Процесът на доставка има за цел и да подсигури наличието на необходимите сертификати, които биха позволили или улеснили пускането на пазара на софтуерния продукт или услуга.

Специално съблюдаването на съответствието с регулаторните рамки и утвърдените стандарти е от огромно значение за реализацията на софтуерните продукти и услуги. Без това да е на лице софтуера не би бил годен за употреба. Това би довело до отдръпване на клиентите, тъй като това би изложило и тяхната дейност на риск (или би я направило напълно невъзможна). В същото време е напълно възможно да се наложат санкции или да се предприемат юридически мерки от страна на държавни органи, в резултат на което може да се понесат поражения като финансови загуби, репутационни загуби и наказателна отговорност. Не се обръща необходимото внимание на тези аспекти, поради което са по - силно застъпени в този курс.

4 Мястото на доставката в жизнения цикъл на софтуера

Ако се абстрахираме от предходната секция вероятно бихме заключили, че процеса на доставка съвпада или се съдържа във фазата на внедряване. И действително тази фаза е фактическата доставка и в резултат на нея софтуерът ни става достъпен за употреба. Но, взимайки предвид и разясненията ни относно същността на доставката от предходната секция, установяваме че доставката на софтуер играе много сериозна роля и във фазата на събиране и анализ на изискванията, където ще трябва да се съобразим с множеството рамки, в които функционираме. Указва съществено

влияние и върху начина на осъществяване на другите фази (например чрез налагане на правила върху процеса на разработка). Фактически за добре изпълнена доставка на софтуера е необходимо да се осъществи наблюдение и контрол върху протичането на целия жизнен цикъл на софтуера.

5 Обобщение

В заключение, на доставката трябва да гледаме като по - абстрактен процес на високо ниво, който се грижи за цялостната успешна поява на софтуерния продукт на своя целеви пазар, в съответствие с всички регулаторни рамки, стандарти и процесуални изисквания. Не добре изпълнената доставка създава риск софтуера да е негоден за употреба, като това не е задължително да е по технически причини. Именно за това в следващите лекции ще обърнем специално внимание на различните аспекти, които се включват в този процес и импликациите им върху продуктите и услугите, които разработваме.