ПЛОВДИВСКИ УНИВЕРСИТЕТ “ПАИСИЙ ХИЛЕНДАРСКИ”
ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА

Катедра “Компютърни технологии”

# ОСИГУРЯВАНЕ НА КАЧЕСТВО НА СОФТУЕРА (Q.A.)

/упражнения/

гл. ас. д-р Георги Шарков

гл. ас. д-р Мая Стоева

# Съдържание

**МЕТОДИ ЗА ГЪВКАВА СОФТУЕРНА РАЗРАБОТКА**

1. Agile гъвкава софтуерна разработка
2. Скръм (Scrum)

# Agile гъвкава софтуерна разработка

- **Хора и комуникация** – според гъвкавата методология самоорганизацията и мотивацията са важни, като например работата в съвместно помещение и програмиране по двойки.
- **Работещ софтуер** – работещият софтуер е по-полезен по време на срещи с клиента, отколкото представянето на документация по проекта.
- **Сътрудничество с клиента** – софтуерните спецификации не могат да бъдат изцяло изготвени в началото на проекта, затова непрекъснатото участие на всички заинтересовани страни е ключово.
- **Адресиране на промените** – гъвкавата методология за разработка се фокусира над бърза реакция към промените и непрекъснато развитие

# Agile гъвкава софтуерна разработка

***The Agile Manifesto се основава на дванадесет принципа:***

1. Удовлетворение на клиентите чрез бърза доставка на полезен софтуер
2. Промяна в спецификациите е възможна, дори и в късните фази на проекта
3. Често предоставяне на работещ софтуер (в периоди от седмици, а не месеци), напр. MVP (Minimum viable product) версии
4. Работещият софтуер е основната мярка за напредък

# Agile гъвкава софтуерна разработка

***The Agile Manifesto се основава на дванадесет принципа:***

5. Устойчиво развитие, което успява да поддържа постоянно темпо
6. Тясно, ежедневно сътрудничество между бизнес служители и разработчици
7. Разговорите лице в лице са най-добрата форма на комуникация (съвместна локация)
8. Проектите се изграждат около мотивирани хора, на които се има доверие

# Agile гъвкава софтуерна разработка

Гъвкавите методи разбиват задачите на малки стъпки с минимално планиране, без да засягат дългосрочното планиране на проекта. Итерациите (етапите) стават на кратки срокове (timeboxes), които обикновено траят от една до четири седмици. През всяка итерация екипът, съставен от хора с различни функции, работи по всяка една от функциите: планиране, анализ на изискванията, проектиране, разработка, тестване и проверка при приемане.

# Agile гъвкава софтуерна разработка

В края на итерацията работещият софтуер се представя пред заинтересованите страни. Така се намалява цялостния риск и проектът може да бъде адаптиран бързо към промени. Една итерация може да не добавя достатъчно функционалност, за да обоснове пускане на пазара, но целта е да съществува работещо решение (с минимални грешки) в края на всяка итерация.

С цел да се завърши даден софтуер или функционалност, могат да бъдат нужни множество итерации.

# Agile гъвкава софтуерна разработка

Независимо кои дисциплини за програмиране се изискват, всеки екип съдържа представител на клиента. Той се назначава от заинтересованите страни и действа от тяхно име като има личен ангажимент да бъде на разположение на разработчиците за отговори на въпроси, възникнали по време на дадена итерация.
В края на всяка итерация, заинтересованите страни и представителят на клиента преглеждат заедно напредъка на проекта и оценяват приоритетите с оглед оптимизиране на възвращаемостта на инвестициите (ROI) и съобразяване с нуждите на клиента и целите на компанията.

# Agile гъвкава софтуерна разработка

Обща характеристика на гъвкавите методи за разработка са ежедневните срещи относно напредъка на проекта или т.н. стенд-ъп срещи (stand-ups). По време на кратката среща, членовете на екипа докладват пред всички какво са направили предишния ден, какво възнамеряват да правят днес и има ли нещо, което ги възпрепятства да свършат задачите си.

# Agile гъвкава софтуерна разработка

Специфични инструменти и техники, като например, непрекъсната интеграция, автоматизирани Unit тестове, програмиране по двойки, разработка чрез тестове (test-driven development), шаблони за дизайн (design patterns), дизайн според сферата (domain-driven design), преработка на код (code refactoring) и други техники, често се използват за подобряване на качеството и повишаване на гъвкавостта на проекта. В гъвкавите методологии за разработка на софтуер се използват информационни радиатори (напр. дъска с листчета) – те онагледяват физически прогреса на проекта на видно място в офиса, където минувачите могат да го видят.

# Scrum

**Scrum** е процес, използван при изготвяне и управление на големи проекти. Той е разработен с цел дългосрочното планиране на изработването на даден продукт да бъде улеснено значимо. За разлика от типичния мениджмънт чрез контрол и командване, при Scrum процесите се набляга на обратната връзка и се дава повече власт на хората, вършещи „черната работа".

# Scrum

**Scrum** е процес, използван при изготвяне и управление на големи проекти. Той е разработен с цел дългосрочното планиране на изработването на даден продукт да бъде улеснено значимо. За разлика от типичния мениджмънт чрез контрол и командване, при Scrum процесите се набляга на обратната връзка и се дава повече власт на хората, вършещи „черната работа".

# Scrum – ключови роли

**Product Owner / Продуктен Собственик**

Продуктният собственик е отговорното лице за това организацията да добавя стойност (към продукти и услуги) за клиентите. Той е гласът на клиента и прави всякакви неща свързани с него. Той добавя ъпдейти от клиента, приоритизира задачите и вписва всичко в „product backlog"- а. В SCRUM екипа такава роля трябва да има само един човек. Ролята на продуктния собственик не се препоръчва да се смесва с тази на SCRUM господаря. От друга страна той може да бъде човек от развойния екип.

# Scrum – ключови роли

**Scrum Master / Scrum Господар**

Scrum Господарят е отговорен да бъдат премахвани пречките пред това екипът да изпълни договорените за спринта задачи и да постигне желаните за спринта цели. Той не е лидер на екипа, а нещо като служещ на екипа лидер. Следи за изпълнение на правилата и за това нещата да се случват по концепциите на SCRUM. SCRUM господарят се грижи за това екипът да не бъде разсейван със странични фактори и за това всичко да е подчинено на целите на спринта.

# Scrum – ключови роли

**Development Team / Развоен екип**

Екипът, който създава продукта, върши „черната работа" – анализира, прави архитектурата, пише код, тества го, извършва работа по техническа комуникация, документира и т.н. Състои се от 3 до 9 души, които имат различни и сходни умения по множеството работа. Той има задачата да доставя увеличен брой или подобрени парчета софтуер на края на всеки спринт. Тези парчета трябва да са по някакъв начин подходящи за изпращане или продажба. Така във всеки един момент проектът може да бъде спрян и продаден. Развойният екип се самоорганизира, дори и да трябва да си сътрудничат с отдел/организация за управление на проекти.

# Scrum – ключови роли

**Product Owner, Scrum Master и Development Team образуват заедно т.нар. Scrum Team.**

# Scrum – помощни роли

1. Stakeholders / Клиенти или Вендори
2. Managers / Мениджъри

Матрицата помага да се планират всички заинтересовани лица от проекта.
По-конкретно се планира какви да бъдат те по време на всяка от фазите (*при по-детайлният план – на всяка от дейностите*) на проекта: **R**esponsible, **A**ccountable, **C**onsultant, **I**nformed и **F**acilitate.

# Литература

www.agilemanifesto.org/

# БЛАГОДАРЯ ЗА ВАШЕТО ВНИМАНИЕ!