

Гъвкаво Разработване на Софтуер

Ива Кръстева

Институт GATE

Съдържание

Мотивация и развитие

Текущ статус на използването на
гъвкави методи и практики


Манифест за гъвкава разработка на
софтуер

Екстремно програмиране - практики


Скрум – стойности, принципи, процес и
основни елементи

Разработка на софтуер - еволюция

План-базираната софтуерна разработка се основава на контролирани и стриктни процеси, които включват подробно планиране на проекта, спецификация на изискванията, моделиране, разработване и тестване на системата.



Недостатък: Формални и тежка процеси със значителни разходи, произтичащи от изчерпателната документация и формална комуникация, без възможност за своевременна реакция на промени и ранни доставки поради предварителното планиране и доставка на софтуер в края на целия процес.



Решение: Гъвкавите методи се фокусират върху софтуера и неформалната комуникация, и разработват софтуер в поредица от стъпки като имат за цел да намалят бюрокрацията на процеса възможно най-много.

Гъвкава разработка на софтуер - мотивация

Динамична бизнес среда

- **Конкурентно предимство** - Софтуерните продукти трябва да бъдат пуснати на пазара бързо, така че бързото разработване и доставка на софтуер е от съществено значение.
- **Променящи се изисквания** - Гъвкавото софтуерно инженерство се фокусира върху бързото предоставяне на функционалност, реагирайки на променящите се продуктови спецификации и минимизирайки разходите за разработка.

Гъвкава разработка на софтуер - история

- През 90-те години на 20-ти век се появяват „олекотени“ методи за разработка на софтуер като Scrum, Extreme Programming (XP), Feature-Driven Development (FDD), Crystal Family of methods...
- 2001 – Манифест за гъвкава разработка на софтуер
 - Стойности и принципи на гъвкавата разработка

Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler

James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick

Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

Гъвкаво разработване на софтуер – текущ статус

16th State of Agile report by Digital.ai

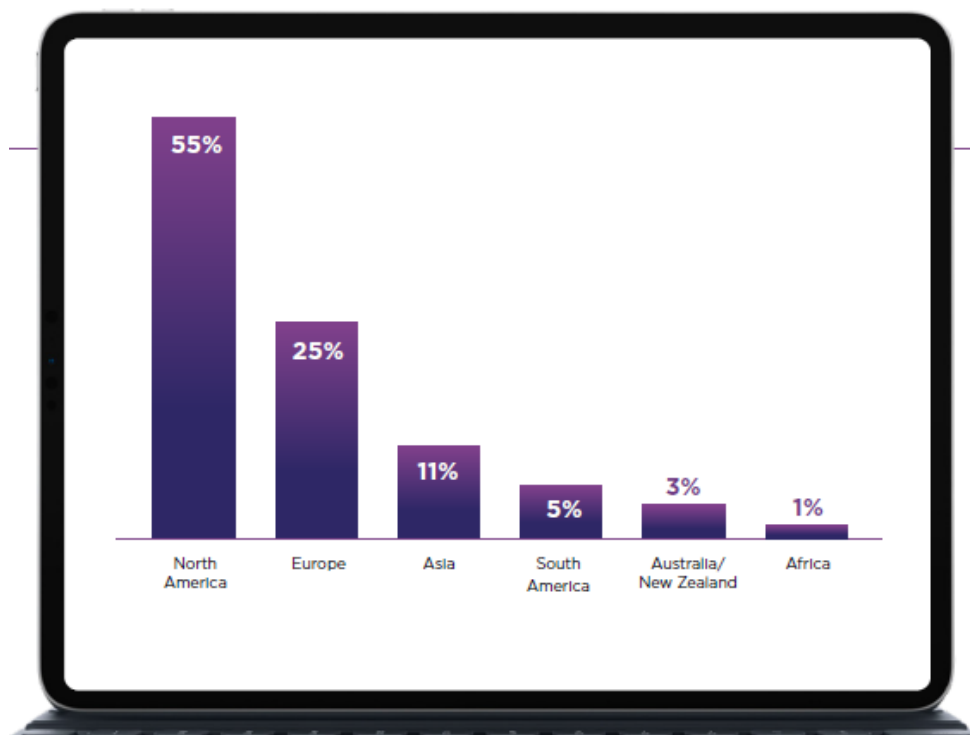
най-дългата непрекъсната серия от ежегодни прегледи на гъвкави методи, техники и практики



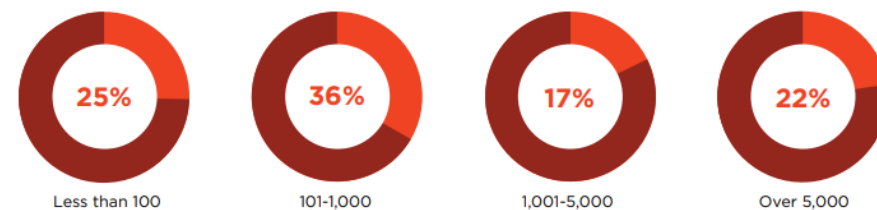
**16th
State of
Agile Report**

16-ти годишен преглед на статуса на гъвкавата разработка - участници

3,220 участници

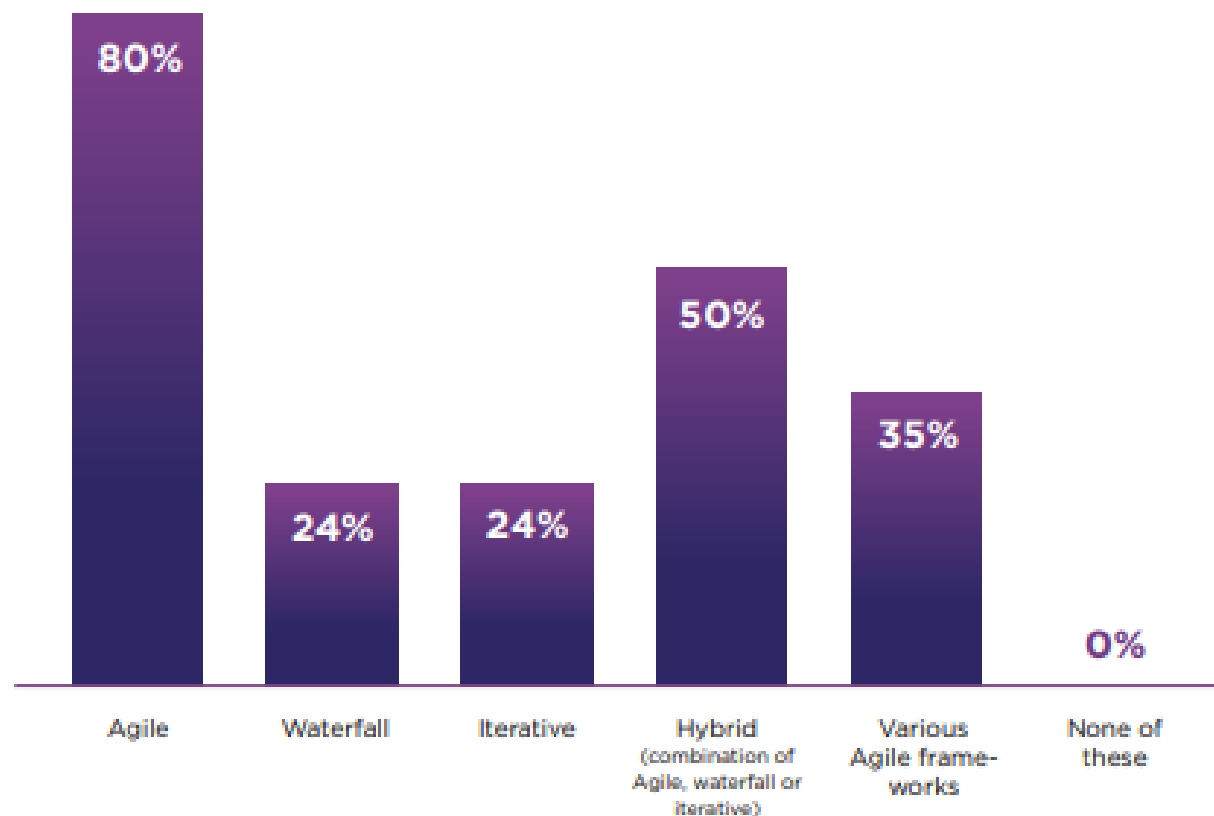


Over one-third of respondents say the software organization in their company is between 101 and 1,000 people and one-quarter say it's less than 100 people



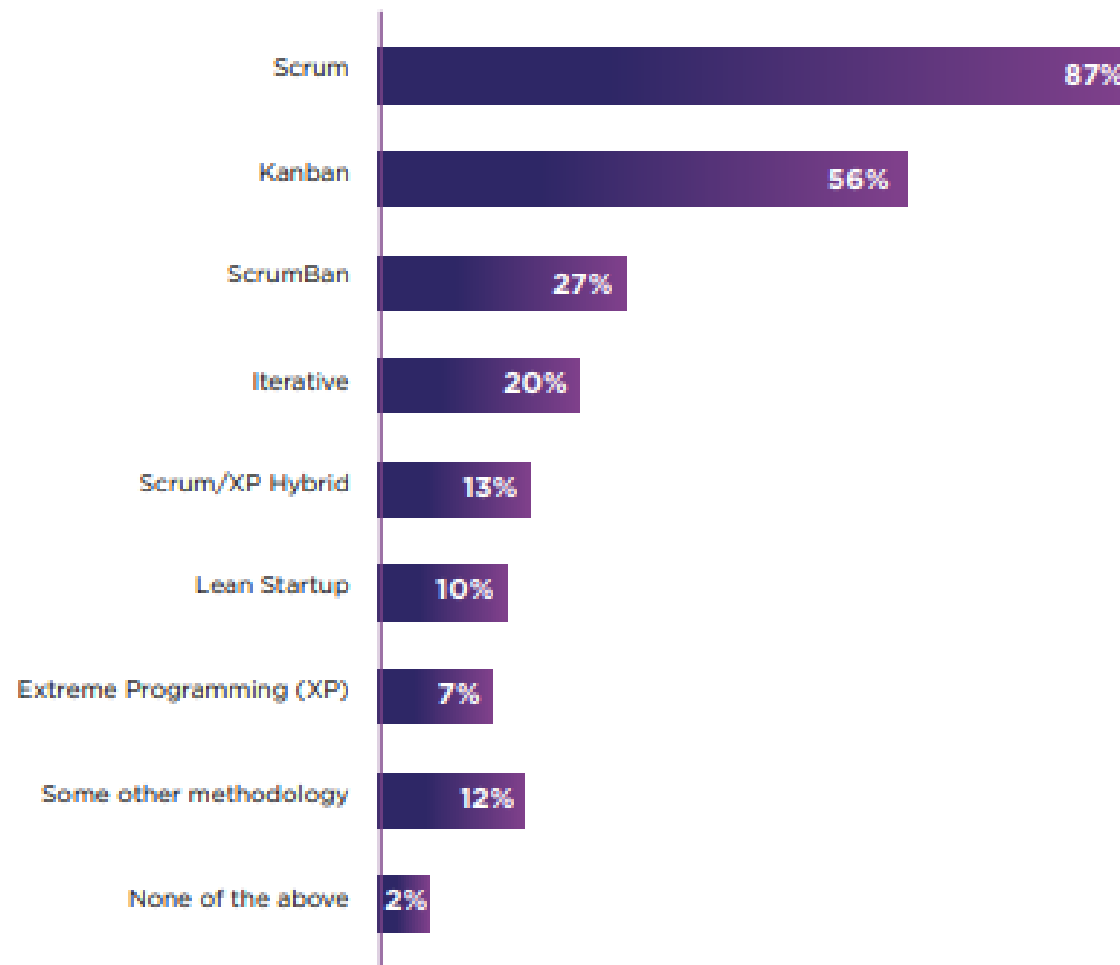
16-ти годишен преглед на статуса на гъвкавата разработка

Подходи за разработка



16-ти годишен преглед на статуса на гъвката разработка

Гъвкави методи



Манифест за гъвкава разработка

Стойности на гъвкавата разработка

Individuals & interactions	over	Processes & tools
Working software	over	Comprehensive documentation
Customer colaboration	over	Contract negotiation
Responding to change	over	Following a plan

That is, while there is value in the items on the RIGHT, we value the items on the LEFT more.



Манифест за гъвкава разработка: Принципи

1

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

2

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

3

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

4

Business people and developers must work together daily throughout the project.

Манифест за гъвкава разработка: Принципи

5

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

6

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

7

Working software is the primary measure of progress.

8

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Манифест за гъвкава разработка: Принципи

9

Continuous attention to technical excellence and good design enhances agility.

10

Simplicity--the art of maximizing the amount of work not done--is essential.

11

The best architectures, requirements, and designs emerge from self-organizing teams.

12

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.



Quizzz

Как гъвкавата разработка помага на организациите да намалят времето за излизане на пазара на нови продукти?

- a. Чрез насърчаване на по-йерархична организационна структура
 - b. Чрез бързо създаване на работещи версии на софтуера и непрекъснатата доставка
 - c. Чрез увеличаване на броя на управленските нива, участващи във вземането на решения
-

Quizzz

Как гъвкавата разработка помага на организациите да подобрят качеството на продуктите?

- a. Чрез наблягане на обширни тестове и документация
- b. Чрез намаляване на акцента върху предоставянето на работещ софтуер и фокусиране повече върху спазването на крайните срокове
- c. Като позволява по-бърза обратна връзка и непрекъснати цикли на подобрене

Quizzz

Как гъвкавата разработка помага на организациите да подобрят морала на екипа и удовлетворението от работата?

- a. Като набляга на екипното сътрудничество и признаването на индивидуалния принос
- b. Чрез насърчаване на силно конкурентна култура
- c. Чрез използване на контролиращ лидерски стил

Quizzz

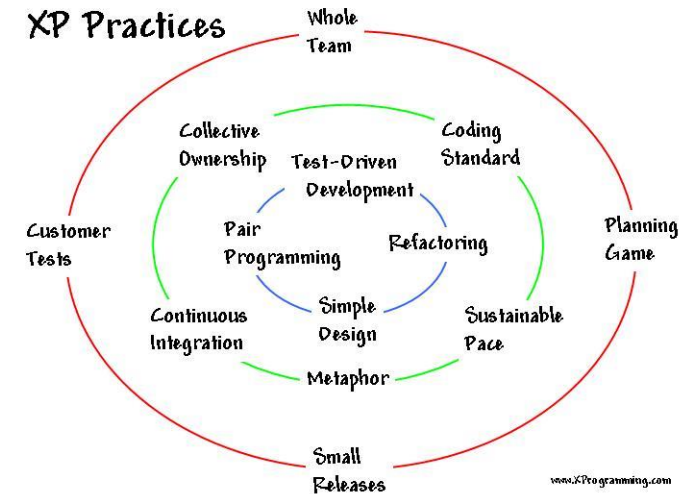
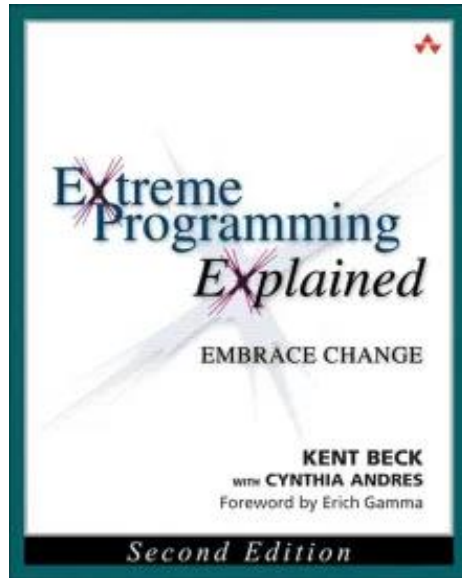
Какво представлява Манифестът за гъвкава разработка на софтуер?

- a. Документ, който описва Гъвкава методология
- b. Ръководство за гъвкаво управление на проекти
- c. Набор от принципи за гъвкава разработка на софтуер

Quizzz

Каква е основната мотивация за появата на гъвкавото разработване на софтуер?

1. Желанието за намаляване на разходите и повишаване на ефективността при разработването на софтуер
2. Необходимостта да се отговори на променливите изисквания на бизнеса и да се осигури конкурентно предимство
3. Необходимостта да се спазват правителствени разпоредби и политически изискванията

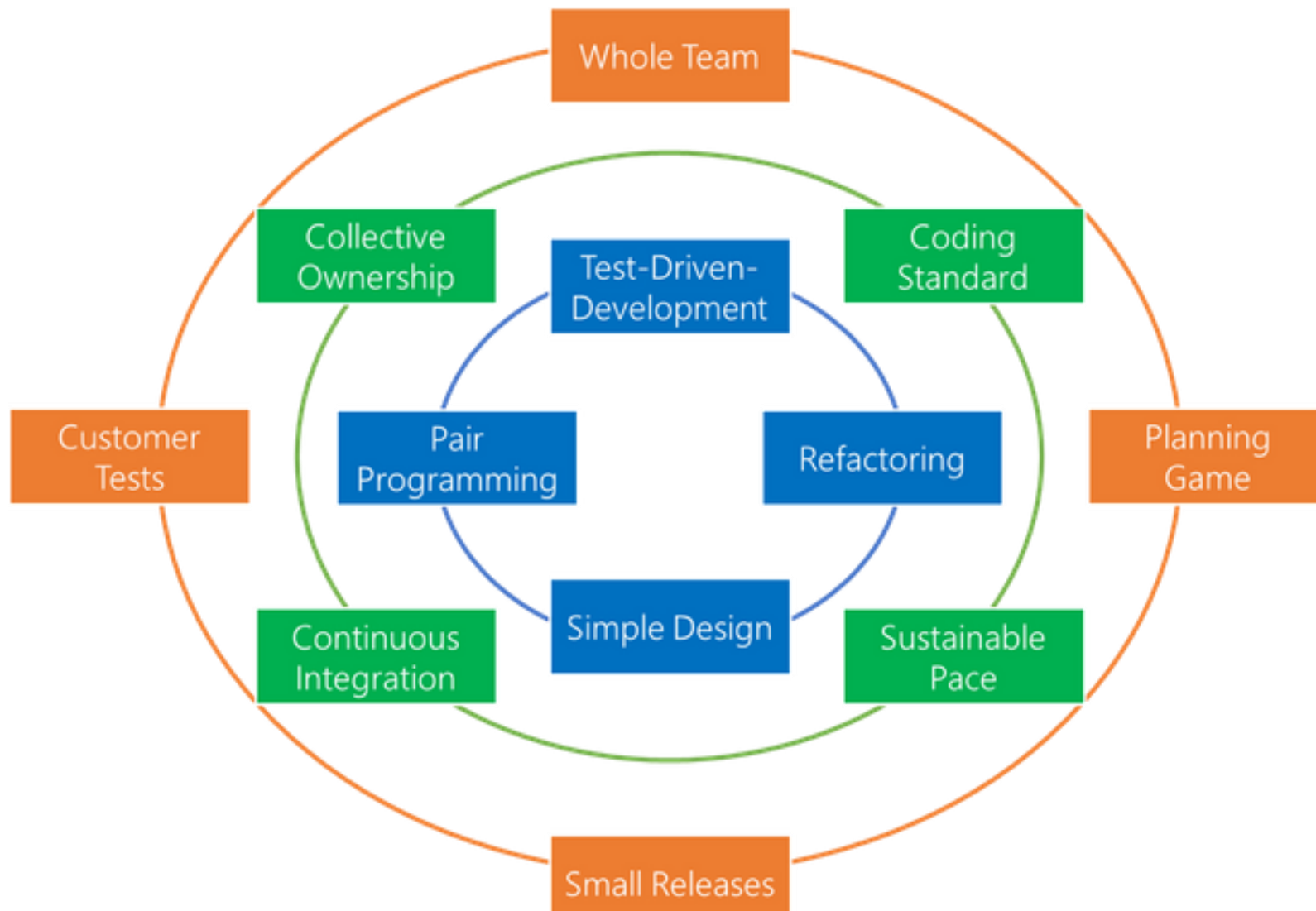


Extreme Programming (XP)

Методологията, която промени културата на разработка на софтуер!!!

Екстремно Програмиране

- Създадена от Кент Бек през 1998 г.
- Използване на признати добри практики до „екстремни“ нива
- Фокусира се върху 12 практики за разработка
- Някои от тези практики се използват широко и до днес



Практики на Екстремното Програмиране

XP практики

Whole team\ On-site customer

- Екипът се състои от всички, участващи в създаването на продукта, и работят заедно (физически!)

Planning game\Incremental planning

- Няма предварително планиране за системата. Това, което трябва да се внедри (изискванията), се установява в дискусии с представител на клиента преди всеки етап/итерация.
- Работата е разделена на потребителски истории. Клиентът оценява бизнес стойността на всяка история, преди програмистите да преценят необходимата работа за тях. След това играта е да се увеличи максимално планираното създаване на бизнес стойност за предстоящата итерация.

Small releases

- Първо се разработва минималният полезен набор от функционалности, които осигуряват бизнес стойност. Доставките на системата са чести и постепенно добавят функционалност към предишната версия.

Customer tests

- Клиентът помага на програмистите да напишат (автоматизираните) функционални тестове.

XP практики

Continuous integration

- Веднага след приключване на работата по дадена задача, тя се интегрира в цялата система и се създава нова версия на системата. Всички модулни тестове от всички разработчици се изпълняват автоматично и трябва да са успешни, преди новата версия на системата да бъде приета.

Collective code ownership

- Всеки програмист е отговорен да развива и поддържа целия код, а не само своята част.

Coding standards

- Използват се конвенции за писане на код, с които всички програмисти са съгласни

Sustainable pace\ 40-hour week

- Извънредната работа не се счита за приемливи, тъй като ефектът е намаляване на качеството на кода и средната производителност на екипа

XP практики

Test-driven\ test-first development

- Вместо да пишат код и след това да тестват този код, разработчиците първо пишат тестовете. Това помага да се изясни какво всъщност трябва да прави кодът и че винаги има налична тествана версия на кода. Използва се автоматизирано тестване както на новата функционалност, така и на старите внедрени функционалности.

Refactoring

- Подобряване на структурата, четливостта, ефективността и сигурността на програмата без да се променя функционалността. Очаква се всички разработчици да преработят кода веднага щом бъдат намерени потенциални подобрения в кода. Това прави кода прост и лесен за поддръжка.

Simple design

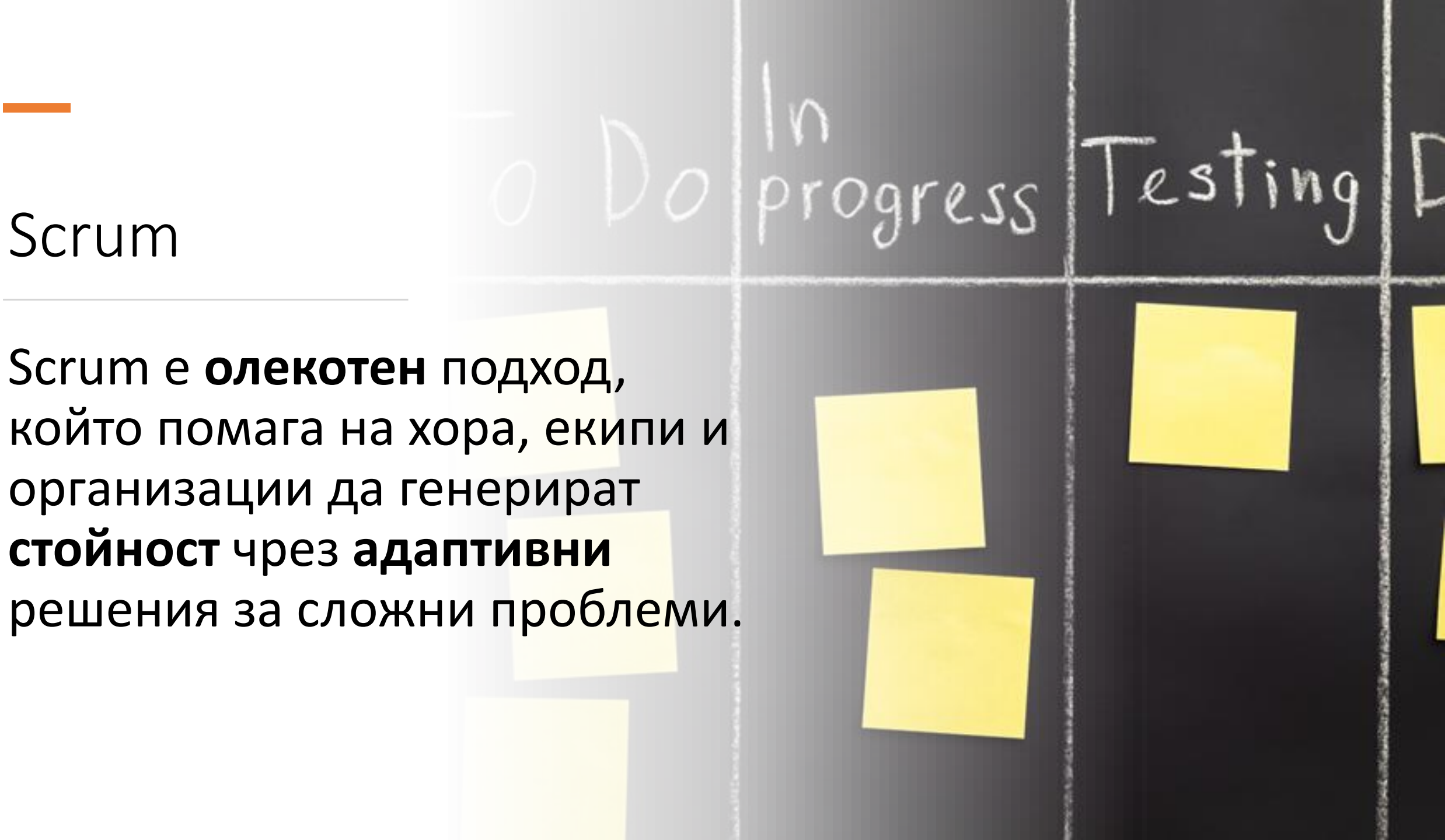
- Прави се най-простият дизайн, който би могъл да работи. Не се планира за бъдеще и не се включват неща, които все още не добавят бизнес стойност.

Pair programming

- Програмистите седят на един и същи компютър по двойки, за да пишат код. Единият пише кода, а другият коментира. Ролите се сменят често.

Scrum

Scrum е **олекотен** подход, който помага на хора, екипи и организации да генерират **стойност** чрез **адаптивни** решения за сложни проблеми.



Принципи на Scrum

Приоритизиране

Адаптивност

Сътрудничество

Самоорганизация

Time-boxing

Основни елементи



Events

Sprint
Sprint Planning
Daily Scrum
Sprint Review
Sprint retrospective



Roles

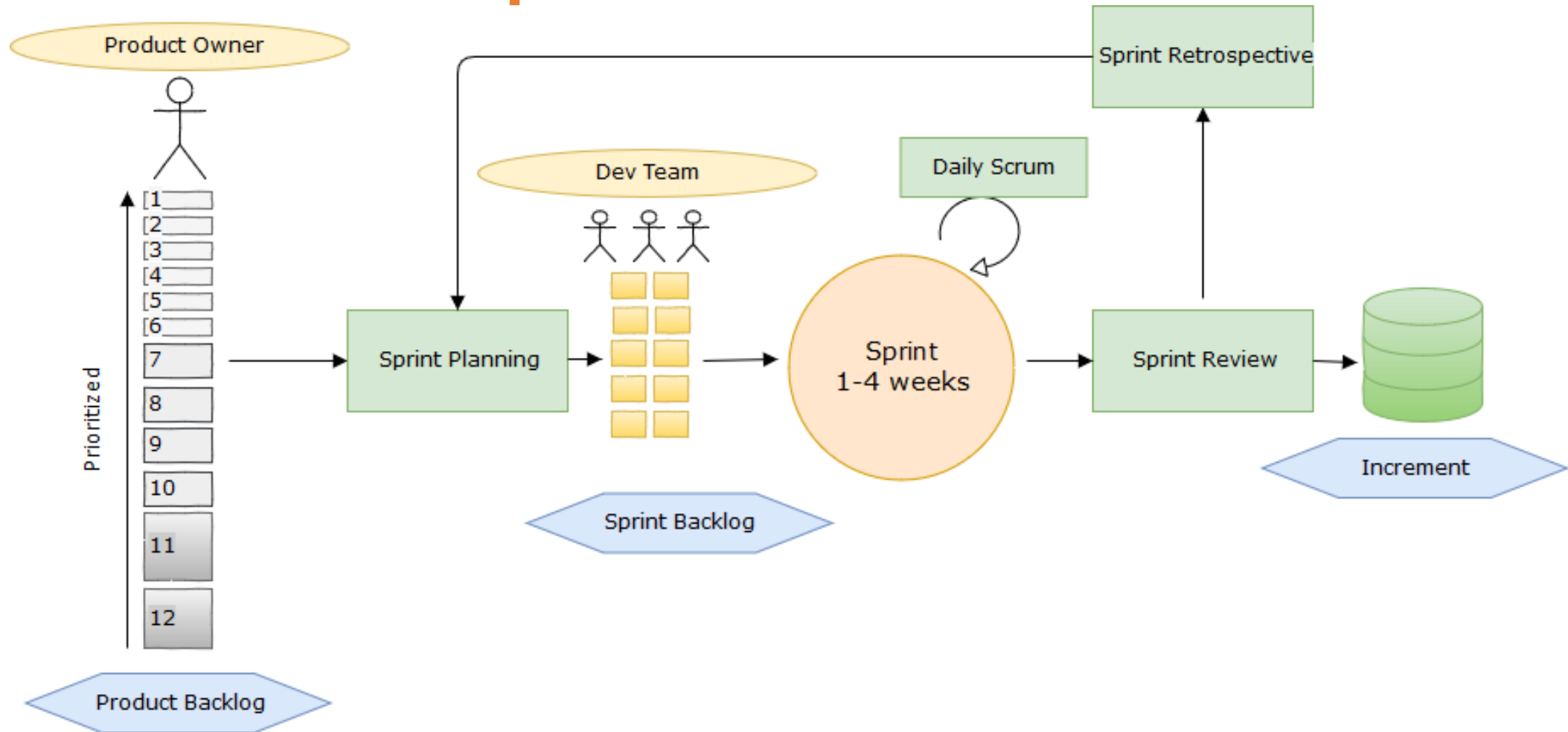
Development team
Scrum Master
Product Owner



Artifacts

Increment
Product Backlog
Sprint Backlog

Scrum процес



Product Owner

- Бизнес ориентиран
- Отговаря за максимизиране на стойността на продукта
- Тази роля принадлежи на един човек
- Отговаря за управлението на Product Backlog
- Отговаря за комуникацията с клиента и други заинтересовани страни в бизнеса
- Планиране и прогнозиране на проекта и представяне на тази информация на всички заинтересовани страни
- Приоритизира изискванията в Product Backlog-a

Scrum Master

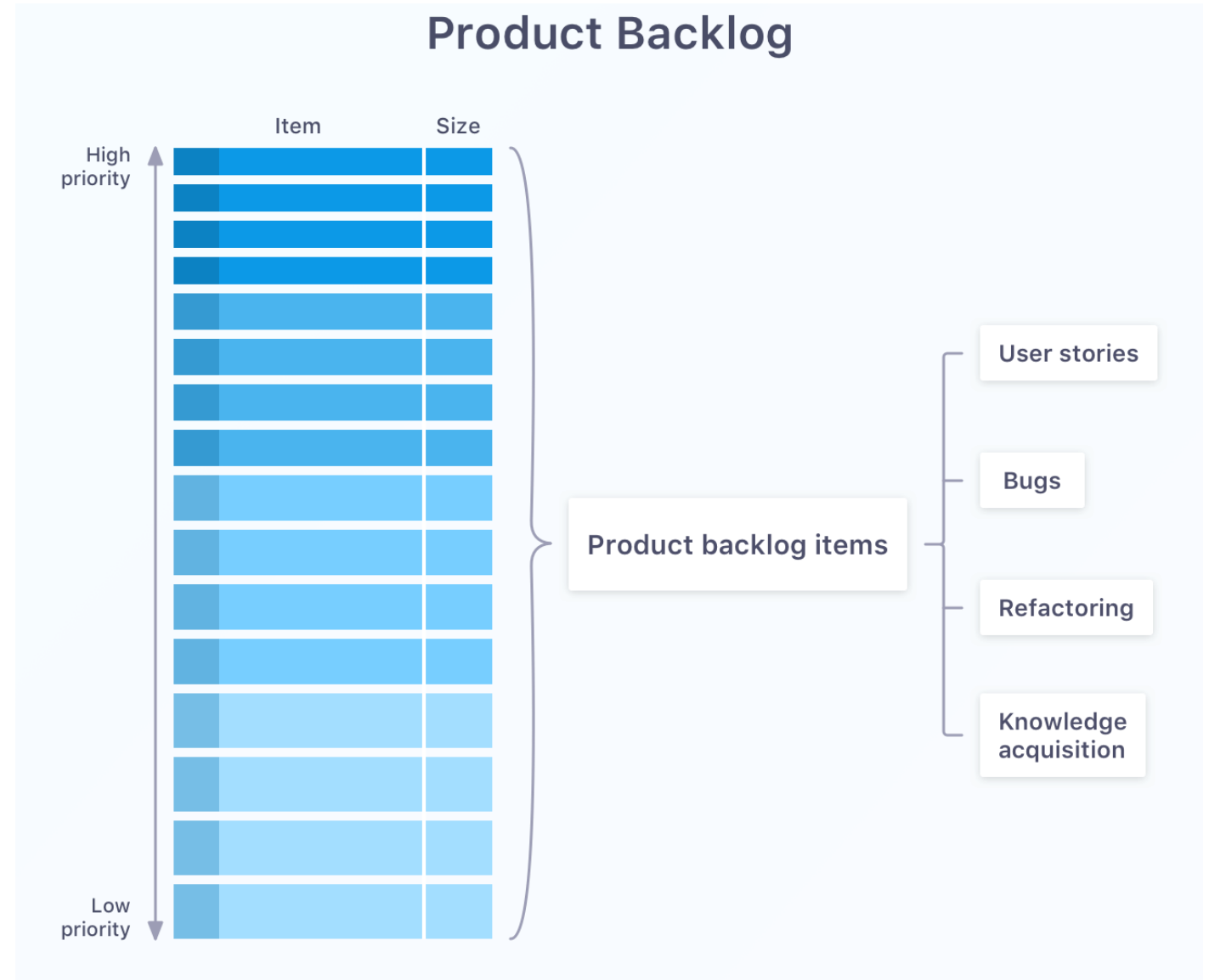
- Експерт в Scrum и помага на екипа да го внедри правилно
- Управлява процеса на Scrum
- Премахва пречките пред екипа за разработка
- Организира Scrum събитията
- Обучава или тренира екипа от разработчиците
- Помага на Product Owner-а
- Помага на организацията в усилията ѝ за внедряване на методологията

Development Team

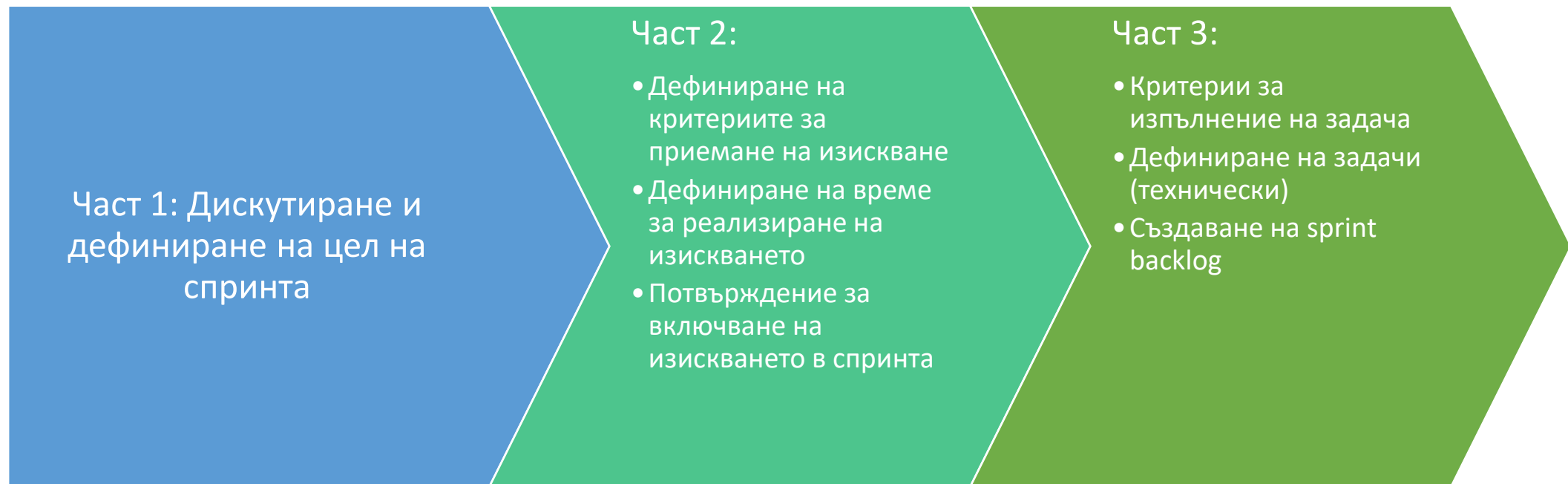
- Експерти в приложната област
- Самоорганизиран екип
- Многофункционален, екипът разполага с всички умения за изпълнение
- Колективна отговорност за всички задачи (колективен ангажимент)
- Мотивация и отдаденост

Product Backlog


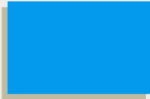
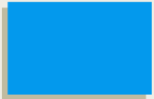


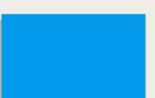

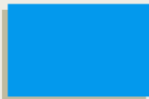
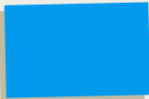
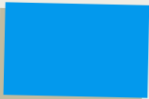
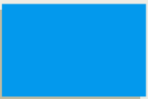
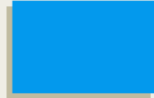
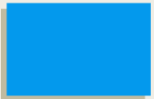

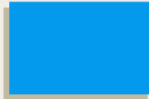

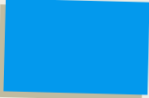
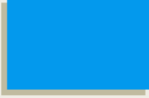

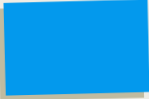
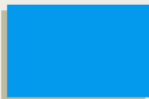
- Екипът знае, че работи по най-важните елементи
- Бизнесът знае, че най-важните характеристики ще бъдат направени първи



Sprint Planning



Sprint Backlog

PBI	TO-DO	IN PROG.	DONE
			   
	 	  	
	   		
	 		

Daily Scrum

- Проверява напредъка на екипа като целта е да се установят проблеми и да се адаптира процеса, ако е необходимо
- 15 минути
- 3 въпроса
 - Какво направих от предишната среща?
 - Какво планирам да завърша до следващата среща?
 - Имам ли пречки по пътя си?



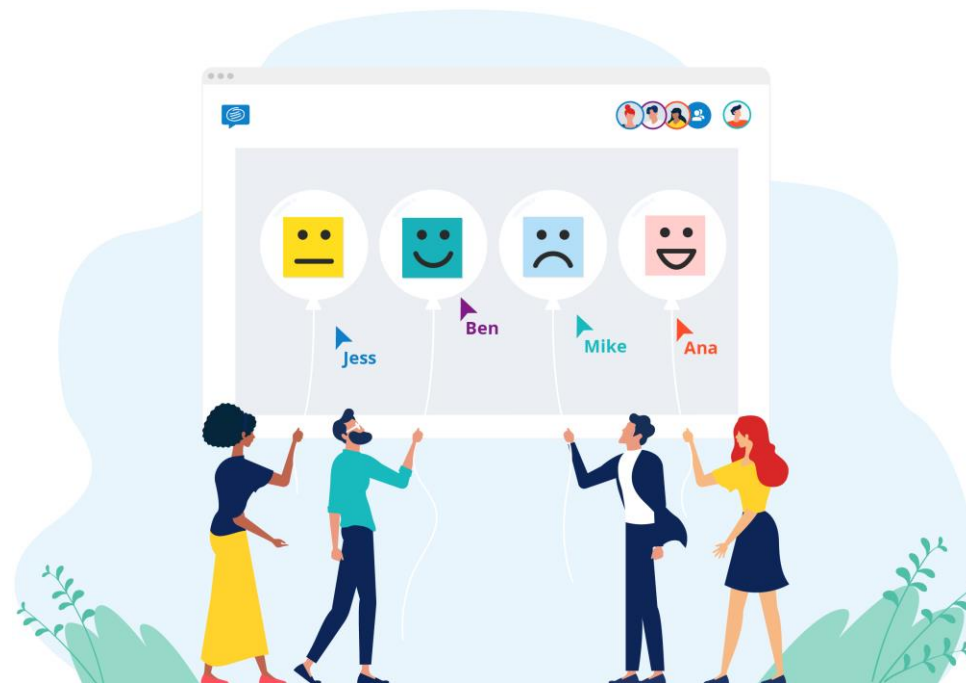
Sprint Review

Екипът демонстрира
разработената
функционалност

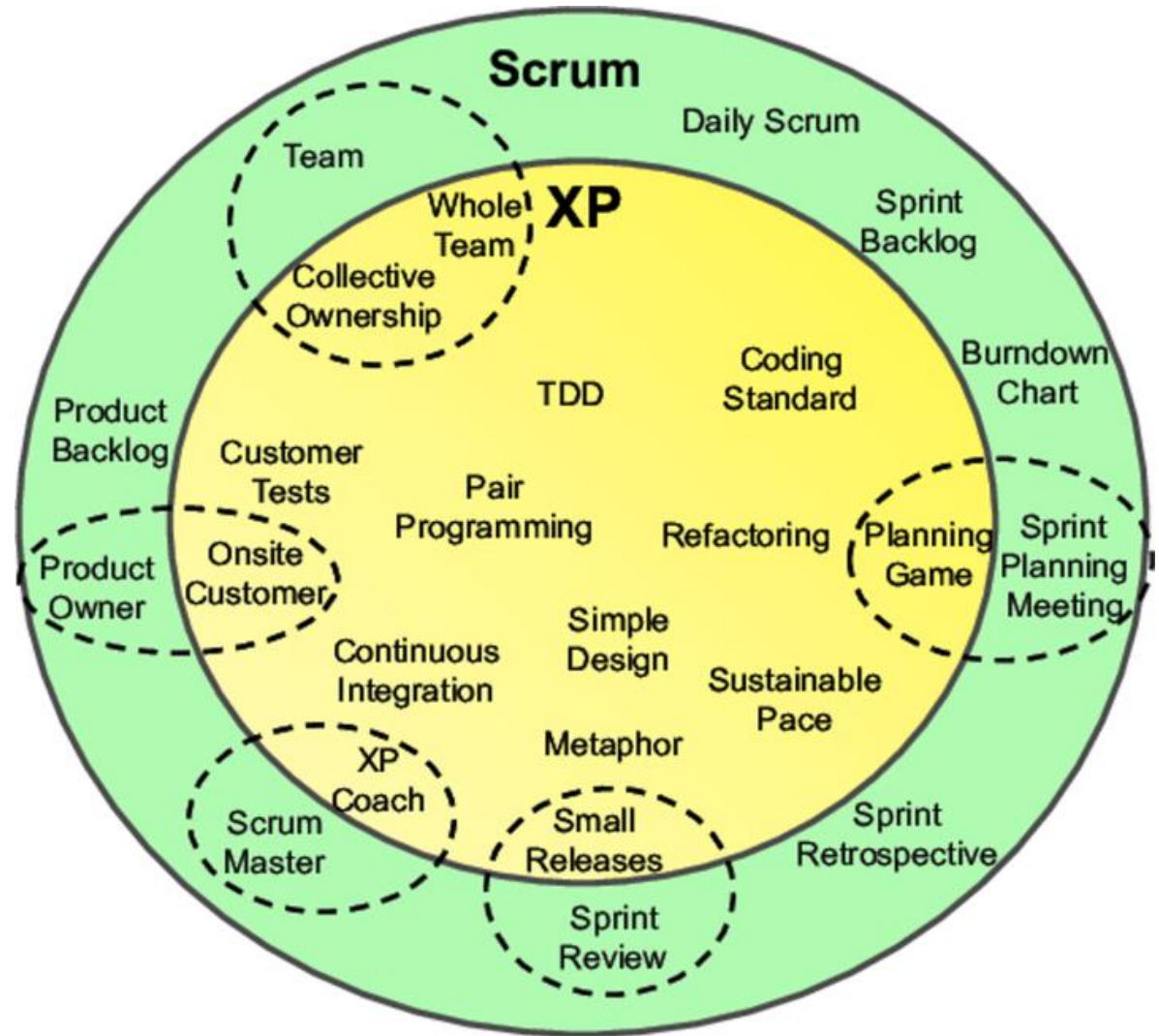
РО и заинтересованите
страни обсъждат, задават
въпроси и се споразумяват
за промени, които да се
отразят като нови елементи
в Product Backlog-a

Sprint Retrospective

- Ревизия на извършената работа и задаване на предложения за това какво може да се подобри
- Различни техники
 - Какво премина добре? (Continue)
 - Какво не работи добре? (Stop)
 - Какви подобрения ще въведем? (Start)



XP and Scrum





Quizzz

В края на срещата за планиране на спринта всеки елемент от Sprint Backlog-а трябва да бъде собственост на член на екипа за разработка.

1. Вярно
 2. Невярно
-

Quizzz

Кое твърдение най-добре описва Sprint Review?

1. Това е преглед на работата на разработчиците
2. Това е преглед на работата на разработчиците, Scrum Master-а и Product Owner-а.
3. Това е демонстрация и проверка на свършената работа
4. Това е преглед на това какво е минало добре и какво не е минало добре



Quizzz

Какъв е резултатът от Sprint Review?

- Списък с подобрения, които екипът ще внедри
 - Общо разбиране за това какво може да бъде доставено в Increment-а и как ще бъде постигната работата, необходима за доставяне на Increment-а
 - Общо разбиране на това, което е направено, напредъка към доставка на версия и ревизиран Product Backlog
 - Общо разбиране за напредъка към целта на спринта и как напредъкът върви към завършване на работата в Sprint Backlog
-

Study case

Ще формираме Sprint Backlog. Екипът предпочита да избере 100 точки за работа за първия спринт, но собственикът на продукта вярва, че трябва да избере поне 150 точки. Какво да правим?

1. Трябва да го обсъдим и да постигнем обща основа
2. Трябва да е 100 точки
3. Трябва да е 150 точки
4. Scrum Master трябва да реши

Study case

Ще възложим на Джон, нашия маркетинг мениджър, ролята на собственик на продукта; Джон не е експерт в разработката на софтуер. Трябва ли вместо това да изберем друг човек?

1. Да, имаме нужда от експерт в софтуерната разработка, способен да комуникира с клиента
2. Да, имаме нужда от експерт в софтуерната разработка, който е част от екипа
3. Не, не е необходимо той да е експерт в софтуерната разработка, стига да получава експертна помощ, когато е необходимо
4. Не, не е необходимо той да е експерт в софтуерната разработка, той просто трябва да е бизнес ориентиран

Study case

Ще изберем Мери или Марк за ролята на Scrum Master. Мери познава много добре Scrum, но е много млада и няма опит в реалния свят. Марк не познава Scrum, но има осем години опит в управлението на ИТ проекти. Кой е по-добрият избор за ролята на Scrum Master?

1. Мери, защото тя познава Scrum и не трябва да управлява проекта
2. Мери, защото познава Scrum и скоро ще научи управление на проекти
3. Mark, защото той познава управлението на проекти и не е нужно да познава Scrum
4. Mark, защото той познава управлението на проекти и скоро ще научи Scrum

Study case

Никой в настоящия състав на екипа не знае как да тества професионално част от софтуера и ние трябва да тестваме всяка част от софтуера, докато я разработваме. Какво да правим?

1. Добавете друг човек към екипа за разработка, който е професионален софтуерен тестер
2. Помолете тестовия отдел, който предоставя услуги на други проекти на компанията, да се справи с тестовете на този проект
3. Възложете тестовете на друга компания
4. Твърде рано е да вземете решение за задача, която предстои да приключи

The background of the slide is a close-up, shallow depth-of-field photograph of a large number of wooden question marks. The question marks are made of light-colored wood and are scattered across the frame, with some in sharp focus in the foreground and others blurred in the background. A horizontal white bar is positioned across the middle of the image, containing the text.

Въпроси