SCRUM – НЕ ЖОРСТКІ ПРАВИЛА, SCRUM – ЦЕ ФРЕЙМВОРК

Слива Є.К.
Науковий керівник – доц. Міщеряков Ю.В.
Харківський національний університет радіоелектроніки
(61166, Харків, просп. Науки,14, каф. Системотехніки,
тел. (057) 702-13-06)
e-mail: yevhen.slyva@nure.ua, телефон +380 99 279 58 52

Scrum in real combat conditions. How to efficiently use scram processes. What we do wrong when we use Scrum. Solving problems with processes using Scrum. How to make software development faster and better. Why scrum can be used well. How Scrum helps developers control development processes. Why you should not take scram as hard and fast rules. We understand the flexibility of the methodology correctly and use this flexibility. Problems that may arise when using scrum. How to solve problems that occur when using scrum. Scrumban via merging scrum and kanban. Why scrum can give you and your project big profit. The new reasons to use scrum.

Сьогодні скрам є найпопулярнішою методологією управління життєвим циклом розробки програмного забезпечення. Скрам дозволяє швидко змінювати пріоритети які важливі для бізнесу. Існує проблема, яка сьогодні доволі часто зустрічається серед тих, хто використовує скрам.

Команди та проекти сприймають цю методологію як жорсткі правила того, яких процесів та правил потрібно жорстко дотримуватись при розробці програмного забезпечення. З таким підходом доволі часто виникає проблема того, що щось йде не так, і щоб вирішити це «не так» потрібно прийняти рішення, яке не задано правилами скраму, або якось суперечить цим правилам. Команди не приймають це рішення бо вони працюють за скрамом, і думають що це буде відхилення від процесу розробки.

Скрам, перш за все, це гнучкий фреймвок, котрий дозволяє налаштовувати себе на різні процеси. Не потрібно сприймати цей фреймворк як щось те, що жорстко встановлює границі. Якщо розширення або зміна границь зробить вашу розробку кращою, швидшою та замовник буде більш задоволений, то чому не зробити це.

Досить часто виникає якась проблема у одному з процесів розробки і учасники процесу думають що в правилах скраму написано як вирішити ту чи іншу проблему, і сподіваються що скрам вирішить проблему за пару годин чи день. Приклад такої проблеми. Розробники на початок кожного спринту мають певну низку задач, котрі мають бути виконані за певний час, залежно від того на скільки часу розрахований спринт. Кожна задача має опис та ціль, що потрібно зробити. У середині спринту виникає проблема, що якась задача недостатньо описана, наприклад, не чіткі

макети, макети які не відповідають дійсності, відсутність макетів, не розписані усі випадки використання функціоналу, не враховується існуючий функціонал.

Таку проблему не можуть вирішити за допомогою існуючих скрам процесів тому що іноді люди не розуміють, що скрам можна налаштувати під себе. Для того, щоб вирішити таку проблему спочатку необхідно сформулювати загальну тестову низку вимог. Усі члени команди повинні вирішити що з цього списку є мінімально необхідним, для того щоб розуміти що задача достатньо описана. Якщо виникають проблеми розбіжності думок, то як раз наша гнучка методологія надає декілька можливих варіантів як вирішувати такі питання, де є розбіжності думок.

Далі необхідні пункти не потрібно одразу жорстоко впроваджувати у процес. Наступний крок це надання пріоритетів кожному с пунктів. Пріоритети створюються для того щоб не впроваджувати усі правила одночасно, вони створюються щоб адаптувати процес під дійсність поступово. Якщо впроваджувати по декілька пунктів з нових правил у процеси розробки, то труднощі під час адаптації будуть значно менші, ніж якщо впровадити зразу усі нові правила.

Канбан також гнучка методологія, яка значно відрізняється від скраму, канбан має набагато менше артифактів аніж скрам, немає спринтів, інкрементів, спринт беклогу, та усього іншого що з’язане зі спринтами. Відсутня багата кількість мітингів.

Скрам змогли змінити настільки, що навіть назва фреймворку дещо змінилася. Здається створили нову методологію. Але люди не створювали нову методологію, вони налаштовували скрам, налаштовували так, як було вигідно їм, вони не побоялися змінити його повністю.

Одною із причин того, чому був створений сам скрам – це бажжаня змінити на той час вже існуючі методології, підходи. Усі те що було дуже відрізнялося одне від іншого. Тому скрам був створений як частинки чогось найкращого серед того що є. Фреймворк не просто увібрав у себе найкращі підходи та рішення. Фреймворк показав що можна змінити усе навкруги, сам скрам ен побоявся змінити все – тому ми не повинні боятися змінювати його.

Ця стаття розповідає про те що Скрам надає можливість правильно налаштовувати процеси та керувати ними. Скрам не задає жорстких обмежень щодо того чи іншого, він дозволяє створювати такі обмеження поступово. Якщо скрам це фреймворк, то і використовувати фреймворк потрібно так як вам зручно, розширяти його.