УДК 005.8:005.1

МЕТОДИ ПРОТИДІЇ ОПЕРАЦІЙНИМ РИЗИКАМ У ІТ-КОМПАНІІ

***І.С. БЕРЕЗНЯК*[1*], *Н.С. КРАСНОКУТСЬКА*[2]**

[1] *магістрант кафедри менеджменту та оподаткування, НТУ «ХПІ», Харків, УКРАЇНА*
[2] *завідувач кафедри менеджменту та оподаткування, професор, док.-р екон. наук, НТУ «ХПІ», Харків, УКРАЇНА*
[*]*email: ivan.berezniak@gmail.com*

Кінцевою метою ІТ-компанії, що займається розробкою програмного забезпечення, є випуск продукту або його нової версії на ринок. Відповідно, вимогами до операційного процесу в такій компанії є можливість точного планування, з одного боку, та ефективна реалізація планів, з іншого. У зв'язку з тим, що найточніше планування може бути легко зруйноване невмілою реалізацією, саме операційні ризики є загальною проблемою у процесі ефективної роботи будь-якої ІТ-компанії.Невід'ємними складовими операційних ризиків під час розробки програмного забезпечення є два головних фактора – нестабільне середовище та технічний борг. Ці фактори не можуть бути цілком виключені, але процеси можна побудувати таким чином, щоб невелювати їх негативниій вплив на процес розробки програмного продукту.

Нестабільне середовище пов'язане із постійними змінами умов діяльності і спричиняє неповноту вимог до кінцевого продукту. У такому разі процес планування ускладнюється як необхідністю урахування змін, так і доповненням вимог до кінцевої цілі. Найголовнішим ключем до ефективної взаємодії з нестабільним середовищем (за фактом таким є будь-яке реальне середовище) є гнучкість, тобто вміння швидко та високоефективно реагувати на будь-які зміни та вимоги. Відповідно, найбільш поширеною та сучасною методологією планування, що певною мірою враховує операційні ризики розробки програмного забезпечення, є Agile та Scrum як шлях його реалізації.

Суть цієї методології полягає у тому, щоб складати не тільки загальний план на увесь шлях розробки та впровадження програмного продукту, але і розбиття усього часу розробки на короткі відрізки часу (ітерації) [1]. Такі ітерації (зазвичай один-два тижні) складаються з кількох етапів: планування ітерації, розробка та тестування, впровадження, аналіз.На стадію аналізу виносяться проблеми, що були виявлені у процессі ітерації. Це і проблеми реалізації, і впровадження. Аналіз реалізації дає можливість підримувати необхідний технічний рівень продукту. Аналіз впровадження дає інформацію щодо відповідності продукту до вимог. Та дає можливість постійної перевірки вимог на доцільність та ефективність.Саме постійне впровадження та аналіз результатів дозволяє отримувати найсвіжищі результати взаємодії продукту з середовищем та найточніше планування наступних етапів. Також такий підхід дає можливість раннього виявлення помилок.Головним недоліком Agile-

методології є відсутність протидії технічному боргу. Саме доповнення Agile методами роботи з технічним боргом, на наш погляд, дозволить більш ефективно управляти операційними ризиками в IT-компанії.

Технічний борг – загальна назва для сукупності усіх «поганих рішень» (рішень, що містять заздалегідь відомі обмеження та недоліки) і неповноти знань [2]. Технічний борг [3] загалом є поєднанням декількох проблем. Це і вимушені неповноцінні рішення, і невдалі рішення, і недостатня кваліфікація працівників та поширення найактуальнішої інформації про стан продукту між ними. І якщо загальне поширення інформації про стан продукту між працівниками є складовою Agile, то методи опрацювання боргових рішень та докваліфікація працівників взагалі не регламентуються.

У межах цієї роботи розглянемо лише опрацювання боргових рішень. Під час роботи з такими ризиками доцільно враховувати умови виникнення технічних боргів, відповідно до яких борги бувають свідомі та несвідомі.

Свідомим є борг, що був створений із розумінням факту неповноти рішення. Тобто заздалегідь закладена неповноцінність рішення задля досягнення певної миттєвої вигоди.

Несвідомим боргом є виявлена помилка чи недолік. Накоплення боргів на початковому етапі може пришвидшити розробку та випуск продукту. Але при подальшій роботі вони будуть все більше и більше руйнувати будь-які плани та зривати строки. І в кінцевому рахунку можуть навіть призвести до повної неможливості подальшої роботи. Саме тому опрацювання та аналіз наявних технічних боргів повинні бути невід'ємною частиною кожної ітерації.

Факт закладення будь-яких свідомих боргів має бути винесений в аналітичну частину ітерації, де вони мають бути пріоритезовані. Також повинні бути встановлені строки їх обробки.Несвідомі борги мають бути поміщенні як пріоритетні на наступну ітерацію. За умови їх критичності для виконання планів поточної ітерації вони вирішуються відразу.Такий підхід дозволяє тримати борги на найнижчому рівні і тим самим підтримувати роботу в більш оптимальному режимі, коли немає вимушених простоїв, та дозволяє у критичні моменти брати нові борги, щоб відреагувати на нові зміни.Але потрібно завжди пам'ятати, що є кретичний рівень боргів, після досягнення якого проект починає вимагати більше часу на підримання його цілостності. А розвиток може вимагати у декілька разів більше часу та ресурсів. У такій ситуації компанія починає вкладати все більше і більше грошей, а заробляти все повільніше и повільніше.

Список літератури:

1. Agile Methodology [Electronic resource]. – Mode of access: http://agilemethodology.org.

2. *Тепляков, С.* Технический долг / С. Тепляков [Электронный ресурс]. – Режим доступа: http://habrahabr.ru/post/119490.

3. Технический долг разорит вас. Если вы позволите, конечно // Блог компании Alconost Inc. [Электронный ресурс]. – Режим доступа: http://habrahabr.ru/company/alconost/blog/174521.