Wskaźniki stosowane w procesie programowania i realizacji projektów
Opracowanie
Styczeń 2009 r.

Rational software

Kiedy można zakończyć testowanie?

Nowy model wymiernego określania ryzyka wprowadzania na rynek nowego oprogramowania w porównaniu z kosztami dalszego testowania

Murray Cantor, starszy inżynier, sekcja ds. oprogramowania Rational, dział oprogramowania IBM (IBM Software Group)

Michael Lundblad, kierownik programu odpowiedzialny za zarządzanie jakością, sekcja ds. oprogramowania Rational, dział oprogramowania IBM (IBM Software Group)

Avik Sinha, pracownik badawczy zajmujący się automatyzacją testów oraz pomiarami i oceną działania oprogramowania, dział badań IBM (IBM Research)

Clay Williams, kierownik ds. badań naukowych w dziedzinie nadzoru, dział badań IBM (IBM Research).

Spis treści

Wprowadzenie

Koszty przestojów związanych z oprogramowaniem wynoszą w przybliżeniu około 300 mld USD rocznie[1]. Weźmy pod uwagę następujące przykłady:

- *We wrześniu 2007 r. w Atlancie nastąpiła awaria systemu komputerowego przetwarzającego plany lotów, przygotowane przez pilotów, i wysyłającego je do kontrolerów ruchu lotniczego. Agencja przekierowała funkcje systemu do innego komputera w Salt Lake City, który uległ przeciążeniu na skutek zwiększonej ilości danych, co jeszcze pogłębiło problem*[2];
- *Operator telekomunikacyjny wydał przez sześć miesięcy 3 mln USD na pomoc techniczną do oprogramowania zamiast na rozwój nowych aplikacji*[3];
- *Firma oferująca ubezpieczenia zdrowotne traci 20 mld USD rocznie z tytułu utraconych możliwości biznesowych oraz prac związanych z usuwaniem problemów z wydajnością systemu produkcyjnego*[4].

Z badań i z doświadczenia wynika, że koszty znajdywania i usuwania defektów rosną wykładniczo wraz z jakością kodu. Czyli po usunięciu każdego defektu rosną koszty znalezienia i usunięcia następnego. Oznacza to, że praktycznie nie ma możliwości stworzenia oprogramowania pozbawionego wad. Ze zrozumiałych względów dyrekcja firm oczekuje produktów, których defekty będą miały minimalny wpływ na użytkowników – pod względem funkcjonalnym i finansowym. Ale taka pewność – oraz zwiększona zdolność do unikania awarii takich, jak te opisane powyżej – sporo kosztuje. Na testowanie oprogramowania wydaje się rocznie setki milionów dolarów. W rzeczywistości bezpośrednie doświadczenia IBM ze współpracy z firmami z całego świata oraz dane zebrane przez pracowników badawczych współpracujących z setkami firm pokazują, że większość firm poświęca na kontrolę jakości[5] 25% lub nawet większą część czasu i pieniędzy przeznaczonych na tworzenie oprogramowania.

Najważniejsze informacje

Menedżerowie projektów potrzebują sposobu mierzenia ryzyka biznesowego związanego z wprowadzeniem na rynek oprogramowania w porównaniu z kosztami dalszego testowania.

Kluczowym punktem w każdym procesie programowania jest moment, w którym kończy się testowanie i firma przystępuje do realizowania pierwszych wdrożeń. Jest to ten punkt w cyklu życia każdego projektu programistycznego, w którym menedżer programu musi zadać kilka bardzo praktycznych pytań:

- *Czy dalsze wydawanie pieniędzy na kontrolę jakości ma sens?*
- *Czy dalsze testy nie będą kosztowały więcej, niż są tego warte?*
- *Czy oprogramowanie jest gotowe do wprowadzenia na rynek?*
- *Skąd wiadomo, kiedy można zakończyć testowanie?*

W niniejszym opracowaniu przedstawiono nowatorskie podejście umożliwiające udzielenie odpowiedzi na te pytania – jest to nowy model służący do wymiernego określania ryzyka biznesowego związanego z wprowadzeniem oprogramowania na rynek w porównaniu z kosztami dalszego testowania.

Sformułowanie właściwego pytania – ryzyko techniczne a ekonomiczne

Po pierwszej analizie pytanie o to, kiedy należy zakończyć testowanie, wydaje się czysto techniczne. Firmy zazwyczaj stosują kryteria dotyczące zakończenia procesu testowania oparte na takich czynnikach, jak odsetek pomyślnie zakończonych testów na kompletność zestawu funkcji oraz liczba defektów pozostałych jeszcze na różnych poziomach istotności. Niektóre zespoły ds. kontroli jakości mierzą również jakość za pomocą takich wskaźników, jak:

- *gęstość defektów i ścieżka schodzenia (glide path);*
- *zmienność wymagań;;*
- *częstość zmian w kodzie;*
- *średni czas między awariami (mean time to failure – MTTF) i średni czas naprawy podczas testów regresyjnych i obciążeniowych;*
- *zakres testowania;*
- *użyteczność, niezawodność, wydajność/skalowalność i łatwość serwisowania.*

Najważniejsze informacje

Stosowanie samych tylko środków technicznych może spowodować pominięcie kluczowych konsekwencji ekonomicznych dalszego testowania w porównaniu z wprowadzeniem oprogramowania na rynek.

Podstawowa formuła zaczyna się od obliczenia oczekiwanych kosztów, których chcemy uniknąć oraz oczekiwanej utraty korzyści.

Techniczne środki testowania są niezmiernie istotne, ale często są stosowane na ślepo. Jeśli stosuje się je bez szerszego kontekstu, można pominąć ważne konsekwencje ekonomiczne związane z podjęciem decyzji o kontynuowaniu testowania lub o wprowadzeniu oprogramowania na rynek.

Decyzja o wprowadzeniu oprogramowania do sprzedaży musi uwzględniać nie tylko kryteria techniczne, ale również szanse biznesowe, które stwarza wprowadzenie aplikacji w planowanym terminie. Zbyt późne wprowadzenie oprogramowania na rynek może oznaczać utratę potencjalnych przychodów lub oszczędności. Natomiast zbyt wczesna premiera może oznaczać dla firmy ryzyko w postaci utraty reputacji, zakłóceń w działaniu firmy oraz wysokich kosztów serwisu. Aby znaleźć złoty środek, firmy powinny wziąć pod uwagę szereg czynników, w tym:

- *ilość udoskonaleń, których można oczekiwać w wyniku dalszych inwestycji w testowanie;*
- *zmniejszenie ryzyka ekonomicznego w porównaniu z utratą korzyści w wyniku zbyt późnego wprowadzenia oprogramowania do sprzedaży.*

Rozpoczęcie od wzoru podstawowego

Załóżmy, że dla danej wersji oprogramowania (*build*) istnieje możliwość oszacowania strat firmy z tytułu utraconych przychodów i poniesionych kosztów serwisu związanych z usunięciem usterki w oprogramowaniu. Nazwijmy je *oczekiwanymi kosztami do uniknięcia* (*expected cost avoidance* – ECA). W przypadku dalszego testowania należałoby oczekiwać, że wartość wskaźnika ECA będzie maleć. Jeśli nie, to po co wydawać pieniądze na testowanie?

Z drugiej strony jednakże dalsze testowanie oznacza przesunięcie daty premiery, co powoduje utratę korzyści biznesowych. Dlatego też należy również zmierzyć utratę takich korzyści finansowych. Nazwijmy ten wskaźnik *oczekiwaną utratą korzyści* (*expected value lost* – EVL). Wskaźnik EVL może rosnąć, gdy planuje się wprowadzenie nowej funkcji, i maleć wraz z przesuwaniem premiery.

Najważniejsze informacje

Przy wszelkich inwestycjach w dalsze testowanie należy uwzględnić kwestie zarówno techniczne, jak i ekonomiczne.

We wczesnej fazie cyklu programowania wskaźnik ECA jest prawdopodobnie znacznie wyższy niż wskaźnik EVL. Produkt realizuje jeszcze niewiele zadań, ale generowałby mnóstwo błędów wymagających usunięcia. W rezultacie *oczekiwane ryzyko biznesowe* (*expected business risk* – EBR), liczone według wzoru EBR = ECA – EVL, ma wysoką wartość dodatnią. Ostatecznie, przy dobrym wykonaniu, wskaźnik EBR powinien zbliżać się do zera, a nawet osiągnąć wartość ujemną.

Nawet i w takiej sytuacji jednak wciąż aktualne pozostaje pytanie: „Czy obniżenie wartości wskaźnika EBR uzasadnia dalsze testowanie?". Oznaczmy koszty testowania jako C_T. Gdy stosunek EBR/C_T osiągnie wartość 1 lub mniejszą od 1, staje się oczywiste, że inwestycje w dalsze testowanie nie przynoszą już dalszych korzyści, a nawet mogą spowodować efekt wręcz przeciwny.

Analiza sposobów korzystania z oprogramowania w celu uzyskania bardziej precyzyjnej odpowiedzi

Przedstawione powyżej rozumowanie zakłada, że można rzeczywiście określić wartość wskaźnika EBR. Jak pokażemy, istnieją praktyczne, a wręcz proste metody obliczenia ryzyka biznesowego w kategoriach wskaźników ECA i EVL. Aby zastosować to rozumowanie, należy starannie przeanalizować sposoby korzystania z systemu w celu opracowania prawidłowych testów, które nie tylko obejmą wystarczająco szeroki zakres sposobów wykorzystania i kodu, ale także przetestują inne czynniki mogące mieć konsekwencje ekonomiczne po wdrożeniu, takie jak bezpieczeństwo, wydajność i niezawodność. W rezultacie planowanie testów powinno obejmować aspekty zarówno techniczne, jak i ekonomiczne.

Najważniejsze informacje

Każdy element modelu FURPS ma konsekwencje techniczne i ekonomiczne; aby móc określić wzorce, w niniejszym opracowaniu poddano analizie niezawodność.

Określenie różnych rodzajów ryzyka ekonomicznego wynikających z problemów jakościowych

Rozważając różne rodzaje ryzyka ekonomicznego i korzyści związane z wdrożeniem oprogramowania, warto rozpocząć od analizy typowego zestawu czynników mających wpływ na jakość, znanych jako model FURPS – Features (*funkcje*), Usability (*użyteczność*), Reliability (*niezawodność*), Performance (*wydajność*) i Supportability (*łatwość serwisowania*):

- ***Features*** *— czy oferujemy maksymalne możliwe korzyści? Większość zespołów zajmujących się testowaniem wykonuje rzetelną robotę, przeprowadzając testy funkcjonalne i regresyjne.*
- ***Usability*** *— czy użytkownicy mogą wydajnie wykonywać określoną pracę? Zazwyczaj ten obszar jest badany w ramach testów odbiorczych.*
- ***Reliability*** *— czy częstotliwość awarii oprogramowania oraz odtwarzanie po tych awariach podczas testów lub w fazie produkcyjnej powodują dodatkowe nieuzasadnione koszty operacyjne lub obciążenia?*
- ***Performance*** *— czy czasy reakcji oprogramowania i/lub jego przepustowość są wystarczające, aby zaspokoić potrzeby firmy w zakresie produktywności?*
- ***Supportability*** *— czy defekty można wyizolować i usunąć jak najniższym kosztem?*

Każdy z tych czynników ma określoną specyfikę techniczną oraz konkretne konsekwencje ekonomiczne. Ich szczegółowe omówienie wykracza poza zakres tego opracowania. Możemy natomiast zastosować przyjęte rozumowanie do niezawodności, aby opracować wzorzec, który – po wprowadzeniu odpowiednich korekt – można będzie równie dobrze zastosować względem innych czynników. Dlatego zaczniemy od następujących pytań: „Jakie znaczenie dla firmy ma kwestia niezawodności?" i „Jakie są koszty awarii systemu?".

Najważniejsze informacje

Do wyekstrapolowania wskaźnika awarii po okresie praktycznego testowania można wykorzystać funkcję gęstości awarii.

Zastosowanie równania w obszarze testowania niezawodności

Zastosowanie równania w obszarze testowania niezawodności wymaga rozwiązania dwóch problemów – określenia wskaźnika ECA i określenia wskaźnika EVL. Zajmiemy się tym w poniższych akapitach.

Wykorzystanie funkcji gęstości awarii (*failure density function – fdf*) do określenia wskaźnika *ECA*

Aby zmierzyć statystyczny wpływ na ryzyko biznesowe w kategoriach wskaźnika ECA, potrzebna jest teoria umożliwiająca wyekstrapolowanie wskaźnika awarii po okresie testów praktycznych. Teoria taka jest dostępna i bazuje na funkcji gęstości awarii (*fdf*) opisanej przez rozkład statystyczny. W przeciwieństwie do krzywej o kształcie dzwonu opisującej rozkład normalny, rozkład prawdopodobieństwa awarii w czasie[6] dla funkcji *fdf* wygląda następująco:

$$= \lambda e^{-\lambda t} \quad \text{dla } t \geq 0$$
$$= 0 \quad \text{dla } t < 0$$

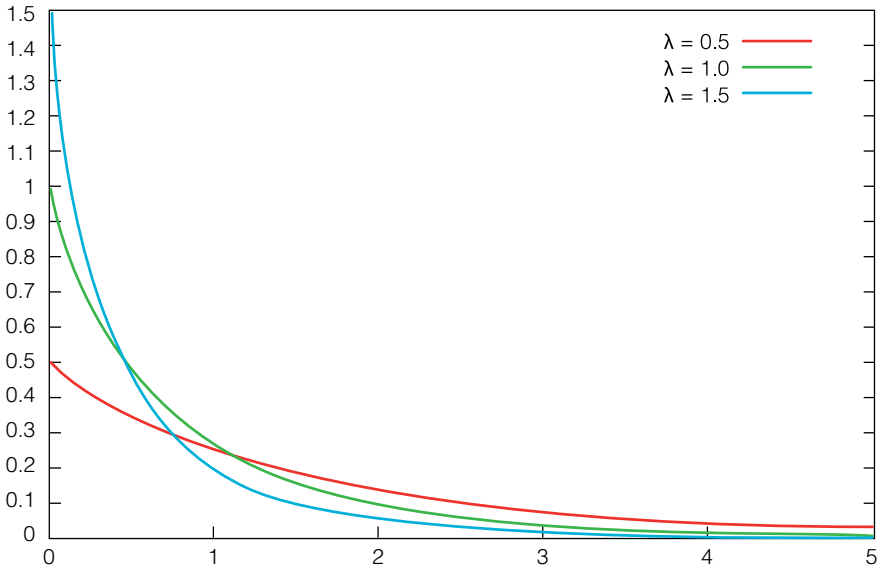
Wykres funkcji *fdf* dla współczynników λ dotyczących różnych wersji oprogramowania przedstawiono na rysunku 1 poniżej. Na rysunku tym oś x reprezentuje czas, zaś oś y reprezentuje gęstość awarii w danym czasie. Prawdopodobieństwo awarii przed danym punktem w czasie (t) jest określone przez pole obszaru pod krzywą w przedziale od 0 do t. Zastosowanie niektórych obliczeń z pierwszego roku daje następujący wynik:

$$P[0,t] = 1 - e^{-\lambda t}$$

gdzie $P[0,t]$ oznacza prawdopodobieństwo wystąpienia awarii przed czasem t.

Najważniejsze informacje

Program jest bardziej niezawodny, gdy bardziej prawdopodobne jest, że awarie wystąpią w dalszej przyszłości.



Rysunek 1. Przykładowe funkcje gęstości awarii (przykład IBM).

Testowanie i naprawa powinny spowodować spadek wartości współczynnika λ, powodując w efekcie spłaszczenie i przesunięcie rozkładu funkcji gęstości awarii w prawo. Zmniejszenie wartości współczynnika λ oznacza, że bardziej prawdopodobne jest, iż awarie wystąpią w dalszej przyszłości, co oznacza, że program jest bardziej niezawodny.

Aby uzyskać przybliżoną wartość współczynnika λ dla danej wersji, należy wykonać następujące czynności:

- *Wykonać wiele testów systemu w różnych warunkach i przy różnych obciążeniach. Wykonanie wymaganej liczby testów może być czasochłonne i kosztowne, tak więc zespół musi opracować sposoby wydajnego tworzenia wystarczającej liczby przebiegów przy różnych obciążeniach i przy różnej kolejności wywoływania funkcji, aby wygenerować odpowiedni rozkład. Są dwa typowe podejścia do tego zagadnienia: testowanie automatyczne i masowe testy wersji beta. W obydwu przypadkach należy zarejestrować czas do wystąpienia awarii dla każdego przebiegu oraz dane związane z debugowaniem dla danej awarii.*

Najważniejsze informacje

Wartość programu w momencie wprowadzenia na rynek jest na ogół zależna od czasu – im późniejsza jest premiera, tym mniejsza jest wartość programu.

- *Utworzyć histogram czasów do wystąpienia awarii. Znormalizować histogram poprzez podzielenie każdej z jego wartości przez liczbę testów. W rezultacie powstaje tabela z procentowymi wartościami wskaźników awarii w określonym czasie.*
- *Wykreślić krzywą na podstawie tabeli, aby uzyskać rozkład funkcji fdf.*

Gdy znamy już współczynnik λ, możemy oszacować prawdopodobieństwo awarii między czasem 0 a dowolnym czasem t poprzez obliczenie pola powierzchni obszaru pod krzywą w przedziale od 0 do t. Należy zauważyć, że jest to prawda nawet dla czasu w odległej przyszłości.

Na podstawie każdego z rozkładów funkcji fdf dla poszczególnych wersji oprogramowania można określić wskaźniki prawdopodobieństwa straty biznesowej i wpływu na obsługę serwisową, aby uzyskać wartość wskaźnika ECA.

Ustalenie korzyści oferowanych przez oprogramowanie wprowadzane na rynek – określenie wskaźnika EVL.

Aby zmierzyć EVL, firma musi umieć określić wartość programu w momencie jego wprowadzania na rynek. Wartość ta jest na ogół zależna od czasu – im późniejsza jest premiera produktu, tym mniejsza jest jego wartość. Na przykład:

- *dostawa w ramach umowy z klauzulami o karach i/lub premiach obliczanych po dokonaniu odbioru;*
- *wprowadzenie systemu na rynek, gdy zysk zależy od czasu wprowadzenia.*

W drugim z ww. przypadków dane muszą zostać dostarczone przez dział marketingu, gdyż poza odpowiednio dobranym czasem premiery ważne mogą być również inne aspekty wprowadzenia oprogramowania na rynek[7].
W każdym przypadku do podjęcia decyzji o tym, czy należy kontynuować/zakończyć testowanie, konieczne jest określenie kwotowej wartości udostępnianego programu.

Najważniejsze informacje

Na samym początku trzeba zadać sobie pytanie: w jaki sposób firma lub branża definiuje niezawodność produktu?

Dostosowanie równania do specyfiki testowania niezawodności

Określone kwotowo straty biznesowe poniesione w wyniku wad niezawodnościowych są bezpośrednio związane z rodzajem opracowywanego oprogramowania lub systemu oraz branży, dla której się je opracowuje. Należy więc zacząć od zadania pytania: „Co firma rozumie pod słowem »niezawodność«?".

Prostym przykładem jest sytuacja, z którą styka się codziennie wiele osób i firm: często restartuje się system operacyjny laptopa po kilku dniach jednoczesnego używania różnych aplikacji, aby uniknąć zachowań odbiegających od normy, takich jak niska wydajność, błędy ekranowe lub nawet zablokowane ekrany, które mogą wystąpić później w wyniku ciągłego używania komputera. Programista opracowujący system operacyjny wie, że koszty awarii są wystarczająco niskie i że system operacyjny spełnia standardy akceptowalnej niezawodności, jeśli średni czas między awariami jest dłuższy niż akceptowalny czas między restartami.

Inna przykładowa sytuacja występuje rzadziej i może powodować znacznie wyższe koszty. Jeśli system obsługuje centralę telefoniczną biura, każda awaria może „kosztować" dziesiątki tysięcy dolarów. W tym przypadku prawdopodobieństwo pojedynczej awarii w roku musi być mniejsze niż 10^{-3} lub 10^{-4}.

Korzystając z opisanej wcześniej analizy funkcji gęstości awarii, przeanalizujemy teraz przykład z działu IBM Research Lab, aby uzasadnić to założenie.

Mierzenie niezawodności w kategoriach czasu do wystąpienia awarii

Niezawodność jest mierzona jako czas między awariami w trakcie testowania lub eksploatacji w terenie. Aby przetestować niezawodność, trzeba wykonać dużą liczbę równoczesnych przebiegów testowych dla serii wersji oprogramowania przy różnych obciążeniach i z losowym wywoływaniem funkcji. Czas do wystąpienia awarii może być dla każdej wersji oprogramowania inny. Tak więc dla każdej wersji można utworzyć histogram czasu do wystąpienia awarii. Każda wersja ma inny histogram.

Najważniejsze informacje

Istnieje możliwość wyekstrapolowania i obliczenia prawdopodobieństwa awarii dla dowolnego przedziału czasu w przyszłości.

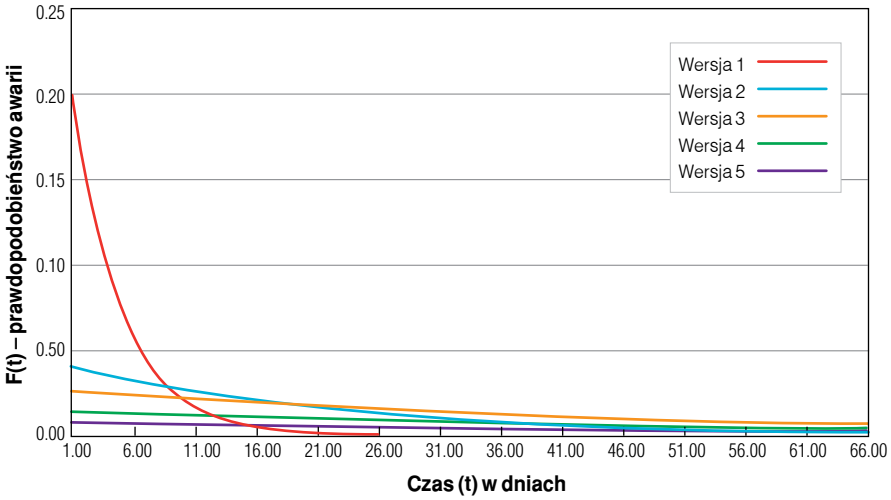
Na przykład dla danej wersji można wykorzystać testowanie automatyczne w celu przeprowadzenia 10 000 testów niezawodnościowych. Każdy test wprowadza wersję w proces równoległy. Monitorowano i rejestrowano liczbę awarii procesów, które wystąpiły w ciągu dnia. Cały eksperyment był obserwowany przez okres czterech dni (patrz tabela poniżej).

	Wersja 1	Wersja 2	Wersja 3	Wersja 4	Wersja 5
Dzień 1	31	10	5	3	2
Dzień 2	125	49	24	13	10
Dzień 3	453	233	112	63	42
Dzień 4	5896	1333	650	330	210

Tabela 1. Częstość awarii dla każdej wersji (przykład IBM).

Mimo tego, że eksperymenty niezawodnościowe trwały tylko cztery dni (i niektóre przebiegi nie zakończyły się niepowodzeniem w wyznaczonym czasie), istnieje możliwość wyekstrapolowania i obliczenia prawdopodobieństwa awarii systemu dla dowolnego przedziału czasu w przyszłości.

Zastosowanie procesu ekstrapolacji do danych w tabeli 1 prowadzi do uzyskania następującego wykresu:



Rysunek 2. Wyekstrapolowane prawdopodobieństwo awarii (przykład IBM).

Najważniejsze informacje

Z chwilą, gdy znane jest prawdopodobieństwo awarii, można oszacować koszty jej obsługi.

Dla wersji 1 współczynnik $\lambda = 2{,}63 \times 10^{-3}$, a prawdopodobieństwo wystąpienia awarii systemu przez ciągły okres maksymalnie jednego miesiąca określa się poprzez obliczenie pola powierzchni obszaru pod ekstrapolowaną krzywą zgodnie ze wzorem $F(t) = 1 - e^{-\lambda t} = 99{,}96\%$. Należy zauważyć, że istnieje możliwość oszacowania tego prawdopodobieństwa awarii, nawet jeśli test nie będzie mógł trwać przez cały miesiąc.

Należy również pamiętać, że dodatkowe wskaźniki zwiększenia niezawodności w wyniku testowania i napraw można zmierzyć z wykorzystaniem różnych środków statystycznych, np. średniej wartości rozkładów funkcji fdf dla poszczególnych wersji. Na przykład średni czas między awariami to wartość średnia rozkładu funkcji fdf dla danej wersji. Wyliczoną wartość średniego czasu między awariami dla wersji 1–5 przedstawiono w tabeli 2.

	Wersja 1	Wersja 2	Wersja 3	Wersja 4	Wersja 5
Średni czas między awariami (w dniach)	3,80	22,56	48,54	95,79	149,51

Tabela 2. Wykstrapolowany średni czas między awariami (przykład IBM).

Mierzenie ryzyka biznesowego

Teraz rozważmy różne rodzaje ekonomicznego ryzyka związanego z testowaniem. Gdy znamy już prawdopodobieństwo awarii w dowolnym przedziale czasu, możemy również oszacować koszty obsługi tych awarii. Koszty te zależą od różnych zmiennych biznesowych, takich jak koszty robocizny i przeznaczenie systemu. Dla każdej wersji należy oszacować następujące wielkości:

- *oczekiwane koszty serwisu w przypadku udostępnienia oprogramowania już teraz (m) – obliczone poprzez pomnożenie kosztu naprawy przez częstość awarii;*
- *oczekiwane straty biznesowe wynikające z awarii systemu w przypadku udostępnienia oprogramowania już teraz (bl).*

Najważniejsze informacje

Gdy przyrost korzyści ekonomicznych i spadek ryzyka w wyniku testowania zbliżają się mocno do linii poziomej, ale wciąż jej nie osiągają, należy rozważyć, ile korzyści się traci poprzez opóźnianie wprowadzenia nowej wersji na rynek.

Straty biznesowe bl_0, bl_1 i koszty serwisowania m_0, m_1 w punktach czasu t_0 i t_1 są zmiennymi losowymi, gdyż można je tylko oszacować. Przy założonych podstawowych stratach biznesowych na poziomie 20 mln USD pomnożenie tej kwoty przez prawdopodobieństwo awarii w zadanym przedziale czasu daje zmienną bl. Strumień kosztów serwisu jest również podany w postaci szacowanego założonego zestawu zmiennych.

Kontynuując przykład, poprzez dodanie do tabeli 2 oczekiwanych (tzn. średnich) kosztów serwisu i strat biznesowych otrzymujemy tabelę 3.

	Wersja 1	Wersja 2	Wersja 3	Wersja 4	Wersja 5
Średni czas między awariami (w dniach)	3,80	22,56	48,54	95,79	149,51
Koszty serwisu	1 300 000 USD	650 000 USD	1 000 000 USD	300 000 USD	200 000 USD
Straty biznesowe	19 992 46 USD	14 710 482 USD	9 219 881 USD	5 377 919 USD	3 636 197 USD

Tabela 3. Ryzyko biznesowe wdrażania (przykład IBM).

Na podstawie tych zmiennych losowych można określić oczekiwane straty spowodowane przez wady niezawodnościowe. Dla dłuższego testowania wskaźnik ECA oblicza się na podstawie wzoru:

$$ECA = \text{wartość średnia}\ (bl_1 + m_1) - \text{wartość średnia}\ (bl_0 + m_0)$$

Dane te sugerują, że można by zakończyć testowanie, ale nie ma co do tego pewności. Przyrost korzyści ekonomicznych i spadek ryzyka w wyniku testowania zbliżają się mocno do linii poziomej, ale wciąż jej nie osiągają. Należy też rozważyć, ile korzyści w kategoriach nowych przychodów traci się w wyniku opóźniania premiery. W tym celu należy rozważyć wartość samego systemu dla każdej wersji.

Najważniejsze informacje

Uwzględnienie obecnych przychodów netto pozwala zrównoważyć wysokość kosztów do uniknięcia poprzez wydłużenie testowania z utratą korzyści na skutek opóźnień.

System lub oprogramowanie może dostarczać korzyści przez pewien przedział czasu w przyszłości, należy więc obliczyć obecną wartość netto (*net present value* – NPV), która będzie traktowana jako obecne przychody netto (*net present revenue* – NPR) ze sprzedaży udostępnionego systemu lub oprogramowania. Użyteczna forma równania służąca do tego celu wygląda następująco:

$$NPV = \sum_{i=1}^{n} \frac{R_i}{(1 + r_R)^i}$$

Gdzie R_i to oczekiwany strumień przychodów lub korzyści (powiedzmy kwartał po kwartale), a r_R to stopa procentowa związana z wartością pieniądza w czasie (powiedzmy oprocentowanie obligacji skarbowych). W przypadku prac programistycznych na ogół każda z wartości strumienia R_i jest zmienną losową, podobnie jak funkcja fdf.

Ważnym czynnikiem, który należy uwzględnić w odniesieniu do wskaźnika NPR systemu, jest fakt, że przejawia on tendencję spadkową w przypadku opóźniania premiery, ale może rosnąć wraz z rozbudowywaniem funkcjonalności. Uwzględnienie wskaźnika NPR pozwala nam zrównoważyć wysokość kosztów możliwych do uniknięcia poprzez wydłużenie testowania z utratą korzyści na skutek opóźnień wynikłych z przedłużenia okresu testowania.

Wracając do naszego przykładu — gdybyśmy uzyskali z działu marketing informację, że dla danego kwartału Ri wynosi według szacunków 5 mln USD, a r_R = 6%, wskaźnik NPR dla systemu dla różnych dat premiery wyglądałby tak, jak przedstawiliśmy to w tabeli 4.

	2008 r. – II kw.	2008 r. – III kw.	2008 r. – IV kw.	2009 r. – I kw.	2009 r. – II kw.
∑(NPR)	21 061 818 USD	17 325 528 USD	13 365 060 USD	9 166 963 USD	4 716 981 USD

Tabela 4. Wskaźnik NPR (przykład IBM).

Najważniejsze informacje

Jeśli oczekiwane ryzyko biznesowe przyjmuje wartość ujemną, może nadszedł już czas na zakończenie testowania.

Podobnie wartość wskaźnika EVL na skutek możliwości utraconych w okresie testowania można określić według następującego wzoru:

$$EVL = \text{wartość średnia}(NPR_0) - \text{wartość średnia}(NPR_1)$$

Ogólna wartość wskaźnika EBR wynikająca z wydłużenia testowania wynosi:

$$EBR = ECA - EVL$$

Dla wersji 2–5 przykładowego systemu obliczone wskaźniki ECA, EVL i EBR przedstawiają się więc następująco:

	Wersja 2	Wersja 3	Wersja 4	Wersja 5
ECA	5 931 987 USD	5 140 601 USD	4 541 962 USD	1 841 722 USD
EVL	3 736 291 USD	3 960 468 USD	4 198 096 USD	4 449 982 USD
EBR	2 195 696 USD	1 180 132 USD	343 866 USD	-2 608 260 USD

Tabela 5. Obliczone wartości wskaźników ECA, EVL i EBR (przykład IBM).

Należy zauważyć, że wskaźnik EBR dla wersji 5 przyjął wartość ujemną. Oznacza to z pewnością, że nadszedł już czas na zakończenie testowania. W następnym rozdziale przeanalizujemy tę kwestię bardziej szczegółowo.

Kryteria dotyczące zakończenia testowania – ryzyko biznesowe a koszty testowania

Takie podejście stwarza podstawy do udzielenia odpowiedzi na pytanie: „Kiedy należy zakończyć testowanie?”. Zależnie od poziomu zrozumienia różnych rodzajów ryzyka i korzyści w celu uzyskania odpowiedzi można zastosować podejście statystyczne do zmiennych losowych. Aby maksymalnie uprościć odpowiedź, w dalszej części artykułu będziemy już operować wyłącznie wartościami średnimi.

Najważniejsze informacje

Obliczając oczekiwane korzyści z testowania, można określić, kiedy firma wydaje więcej niż oszczędza – a także kiedy wydłużenie czasu testowania może zwiększyć oszczędności.

Poniżej przestawiono dwa sposoby udzielenia odpowiedzi na postawione pytanie:

1. W pewnym punkcie procesu wskaźnik EBR może przyjąć wartość ujemną. W tym punkcie wartość utraconych potencjalnych przychodów zaczyna przewyższać koszty do uniknięcia, co oznacza, że należy zakończyć testowanie i rozpocząć sprzedaż.
2. Znając koszty testowania CT, można określić oczekiwane korzyści z testowania (EVT) jako stosunek oszczędności do kosztów:

$$EV_T = EBR/C_T$$

Wartość mniejsza niż 1 wskazuje, że firma wydaje więcej niż oszczędza, natomiast wartość większa niż 1 oznacza oszczędności wynikające z wydłużenia testowania. Dlatego z tabeli 6 wynika, że należałoby zakończyć testowanie na wersji 5, dla której EVT < 0.

	Wersja 2	Wersja 3	Wersja 4	Wersja 5
EVT	5,63	1,97	1,91	-21,74

Tabela 6. Kiedy należy zakończyć testowanie (przykład IBM).

Wraz z obliczeniem EVT zakończyliśmy analizę ekonomiczną. Stwierdziliśmy, że w tym przykładzie tracimy więcej pieniędzy na pokrywanie kosztów testowania i z powodu straconych potencjalnych przychodów, niż potencjalnie oszczędzamy poprzez unikanie ryzyka awarii. W takim momencie trzeba sobie zadać kolejne pytanie: „Czy są jeszcze jakieś inne przyczyny skłaniające do utworzenia kolejnej wersji?". Nie jest to decyzja mechaniczna, ale dane te faktycznie pomagają pokierować myśleniem. Następny krok determinują już inne czynniki, takie jak kultura korporacyjna. Decydenci mogą chcieć zwiększyć swój poziom pewności, zalecając utworzenie jeszcze jednej wersji.

Najważniejsze informacje

Model sprawowania nadzoru pomaga zadecydować „kto, co, kiedy, dlaczego i jak" w procesie podejmowania decyzji.

Nadzór nad jakością: aspekty organizacyjne i decyzyjne

Przed podjęciem decyzji o zakończeniu testowania należy też rozważyć kwestie sprawowania nadzoru nad jakością. Nadzór ten oznacza określenie „kto, co, kiedy, dlaczego i jak" w procesie podejmowania decyzji[8]. W tabeli 7 podsumowano czynniki wymagające rozważenia dla każdego aspektu modelu sprawowania nadzoru nad testowaniem.

Pytanie	Znaczenie	Czynniki do rozważenia
Kto? (role)	Kto decyduje o zakończeniu testowania? Z kim konsultowana jest ta decyzja? Kto jest informowany o decyzji?	• Dyrektor ds. jakości • Analityk ds. kontroli jakości (analizy statystyczne)
Co? (decyzje)	Jakie artefakty z procesu testowania stanowią dane wejściowe do procesu podejmowania decyzji?	• Plany i wyniki testów • Wykresy
Kiedy? (określanie terminów i tworzenie harmonogramu)	Do kiedy należy podjąć decyzję?	• Analiza funkcji *fdf* w fazie testowania systemu
Dlaczego? (reguły)	Jakie reguły i procedury wprowadzono, aby kierować procesem podejmowania decyzji?	• $EV_T = EB/C_T$, gdzie $EV_T < 1$, oznacza, że wydatki na testowanie przewyższają kwoty, które firma spodziewa się zaoszczędzić poprzez zmniejszenie ryzyka biznesowego.
Jak? (artefakty)	Jakie plany testów, środki i wyniki umożliwiają podejmowanie lepiej ugruntowanych decyzji w zakresie analizy ryzyka biznesowego?	• Wykresy funkcji *fdf* • Oczekiwana wartość kosztów do uniknięcia $ECA = (bl_1 + m_1) - (bl_0 + m_0)$ • Oczekiwana wartość utraconych korzyści EVL = wartość średnia (NPR_0) – wartość średnia (NPR_1) • Oczekiwane ryzyko biznesowe $EBR = ECA - EVL$ • Oczekiwany wskaźnik korzyści $EV_T = EBR/C_T$

Najważniejsze informacje

Dyrektor ds. jakości może stymulować opracowanie kompleksowej strategii zapewniania jakości i kompleksowe podejmowanie decyzji.

Analityk ds. kontroli jakości może dostarczać analizy niezbędne do podjęcia decyzji o kontynuowaniu lub zakończeniu testowania.

Jakość oprogramowania może wywierać wpływ na całą działalność biznesową firmy, tak więc problem nadzoru nad jakością często wymaga zaangażowania ze strony dyrekcji i szczebla kierowniczego przedsiębiorstwa. Być może nadszedł też czas, aby rozważyć wprowadzenie dwóch nowych stanowisk, które nie istnieją w większości firm – dyrektora ds. jakości oraz analityka ds. kontroli jakości.

Obecnie zespoły ds. kontroli jakości koncentrują się na testowaniu i dostarczają wynikającą z tego ocenę elementów modelu FURPS do wykorzystania przy podejmowaniu decyzji. Ale ktoś, właśnie dyrektor ds. jakości odpowiedzialny konkretnie za zarządzanie jakością oprogramowania o newralgicznym znaczeniu przez cały cykl jego życia, musi stymulować opracowanie kompleksowej strategii kontroli jakości oraz kryteriów podejmowania decyzji.

Ponadto tradycyjne zespoły ds. kontroli jakości nie mają w swoim składzie nikogo z wykształceniem matematycznym, kto byłby w stanie wykonywać analizy omówione w niniejszym artykule. Specjalny analityk ds. kontroli jakości mógłby natomiast dostarczać szczegółowe analizy niezbędne do podjęcia lepiej ugruntowanych decyzji o kontynuowaniu lub zakończeniu testowania.

Ponadto obliczenia takie stanowiłyby podstawę do podejmowania decyzji związanych z nadzorem, które podejmuje się po wdrożeniu oprogramowania. Na przykład średni czas między awariami oraz dane probabilistyczne uzyskane w trakcie testowania niezawodności mogą być niezwykle przydatne podczas planowania cyklów serwisowania oprogramowania i systemu. Na takich danych są na przykład oparte cykliczne okresy serwisowania w branży lotniczej. Testowanie niezawodności może również być źródłem użytecznych informacji, które można zamieścić w dokumentacji dotyczącej rozwiązywania problemów, przeznaczonej dla użytkownika. Awarie związane z wydajnością skalowania mogą stanowić podstawę do podejmowania decyzji dotyczących monitorowania systemu produkcyjnego.

Najważniejsze informacje

Porównanie biznesowego wpływu wprowadzenia produktu na rynek w danym momencie z kosztami dalszego testowania może dostarczyć rzetelnych informacji o tym, kiedy najlepiej jest zakończyć testowanie.

Podsumowanie

Decyzja o zakończeniu testowania ma aspekty zarówno ekonomiczne, jak i techniczne. Typowe testy oparte na modelu FURPS dają rzetelny wgląd w potrzeby, proces i status wyników technicznych. W celu pełnego zrozumienia gotowości nowego oprogramowania do wprowadzenia na rynek należy jednakże porównać też biznesowy wpływ wprowadzenia produktu na rynek w danym momencie z kosztami dalszego testowania i koniecznych napraw.

Wykazaliśmy, że dostępne są techniki zapewniające uzyskanie niezbędnych do tego danych. Nie jest niespodzianką, że wymagają one pewnych umiejętności analitycznych oraz uwzględnienia nowych czynników w sprawowaniu nadzoru nad procesem udostępniania nowych wersji oprogramowania. Dzięki zastosowaniu tych metod oraz sprawowaniu nadzoru nad procesem wprowadzania nowych wersji kierownictwo firmy może jednak stworzyć rzetelne, ekonomiczne podstawy do udzielenia odpowiedzi na podstawowe pytanie, jakie zadają sobie programiści: „Kiedy należy zakończyć testowanie?”.

Więcej informacji

Aby dowiedzieć się więcej o produktach i rozwiązaniach do zarządzania jakością IBM Rational®, należy skontaktować się z przedstawicielem IBM lub Partnerem Handlowym IBM albo odwiedzić stronę:

ibm.com/software/rational/offerings/quality

1 The Standish Group, Comparative Economic Normalization Technology Study, CHAOS Chronicles v12.3.9, 30 czerwca 2008 r.
2 „FAA Computer Glitch Causes Flight Delays", Tescom, http://www.tescom-intl.com/site/ en/tescom.asp?pi=465&doc_id=2923.
3, 4 Forrester Research, „Performance-Driven Software Development", Carey Schwaber, luty 2006 r.
5 Walker Royce, Software Project Management: A Unified Framework, (Addison-Wesley Professional, Indianapolis, 1998 r.).
6 M. Modarres, M. Kaminskiy i V. Krivtsov, Reliability Engineering and Risk Analysis: A Practical Guide (Marcel Dekker Inc., Nowy Jork, 1998 r.).
7 M. Cantor, "The Value of Development", artykuł w przygotowaniu.
8 M. Cantor i J. Sanders, „Operational IT Governance", IBM developerWorks®, maj 2007 r. (http://www-128.ibm.com/developerworks/rational/library/may07/cantor_sanders/index.html).

Podziękowania

Autorzy niniejszego opracowania pragną podziękować Davidowi Lubanko, Patrickowi Manciniemu i Haroldowi Mossowi z IBM za pomocne rozmowy oraz za przekazywanie opinii na temat tekstu w trakcie jego opracowywania.

IBM Polska Sp. z o.o.
ul. 1 Sierpnia 8
02-134 Warszawa, Polska
Tel. 022 878 67 77
Internet: www.ibm.com/pl/software
e-mail:software@pl.ibm.com

Wydrukowano w Polsce
Styczeń 2009 r.