Programowanie zwinne

Programowanie zwinne (ang. *agile software development*) – początkowo nieznacząca sekta, obecnie najpopularniejsza religia wśród programistów. Jej podstawowym założeniem jest bycie fajniejszym (zwinniejszym) od nie-edżajlowych deweloperów.

Przykazania

Podstawą religii Agile jest stworzony w 2001 roku przez siedemnastu proroków *Agile Manifesto*, stanowiący odpowiednik dekalogu. Oto treść Manifestu:

Jeden z proroków prezentuje *Agile Manifesto*

Zwinnie wytwarzając oprogramowanie i pomagając ziomkom w tym zakresie, odkrywa się lepsze sposoby bycia fajnym. W wyniku tych doświadczeń przedkłada się:

- ***fajni ludzie i melanże*** *ponad procesy i narzędzia*
- ***pisanie ogromnych ilości niezwiązanego wzajemnie ze sobą kodu*** *ponad obszerną dokumentację*
- ***targowanie się z klientem*** *ponad formalne ustalenia*
- ***zmienianie wszystkiego jak popadnie*** *ponad podążanie za planem*

Według Manifestu Edżajl, to co wymieniono po prawej stronie jest reliktem przeszłości i każdy kto te rzeczy stosuje jest dinozaurem. Jesteśmy od was zwinniejsi.

Stand-up meeting

Wśród elementów metodyk zwinnych można wymienić *stand-upy*, czyli zebrania członków zespołu na stojąco. Z założenia niewygoda wynikająca z ciągłego stania ma skutkować krótszymi spotkaniami, niestety wraz ze wzrostem popularności biurek do stania[1] znacząco zwiększyła się wytrzymałość programistów na stanie. Obecnie wykształciły się różne szkoły rozwiązywania tego problemu, od spotkań stanych na rękach, przez stanie bosymi stopami na rozgrzanych węglach aż po spotkania w egzotycznych pozycjach inspirowanych jogą ekstremalną.

Metodyki *edżajlowe*

Na żyznym gruncie mniej lub bardziej spójnego pierdolenia o programowaniu zwinnym wyrosło

jest już *passé* i nikt poważny go nie stosuje.

Współczesny *stand-up meeting*

Guilt Driven Development

Jedna z bardziej skutecznych metodyk. Na początku iteracji każdy członek zespołu składa obietnice, które elementy zrealizuje. Wszystkie obietnice zapisywane są w odpowiedniej bazie, a komputer *nigdy* nie zapomina. Jeśli nie dotrzyma się jakiejś obietnicy, odpowiedni program napomina dewelopera i robi mu wyrzuty komentując jego niesłowność i nieodpowiedzialność. Przykładowe wyjście takiego programu:

> *Obiecałeś do końca iteracji zaimplementować funkcjonalność #4069, a kodu dalej coś nie ma… Zastanów się, co to mówi o Tobie? Czy kiedykolwiek dojrzejesz do bycia odpowiedzialnym dorosłym? Czy Twoja matka byłaby z ciebie dumna? Przysięgałeś na zdrowie matki, że to zrobisz, czy rzeczywiście tak nisko cenisz sobie jej zdrowie?*

Słowem wyjaśnienia – bardziej zaawansowane programy nękające wymagają wprowadzenia wielu informacji o każdym członku zespołu, jego rodzinie, zdrowiu psychicznym, osiągnięciach w nauce i życiu erotycznym. Następnie programista musi przysiąc na coś, co jest wysoko w jego hierarchii wartości (np. na Boga, matkę, rachunek różniczkowy…) dla maksymalnej skuteczności. Warto zaznaczyć, że programy nękające działają także w święta, ba, w dni ustawowo wolne od pracy nękają aż trzy razy częściej.

Wśród skutków ubocznych metodyki GDD można wymienić między innymi większą podatność członków zespołu na załamanie nerwowe, depresję i ogólną zwiększoną śmiertelność. Należy ją stosować wraz z agresywnymi technikami poszukiwania pracowników w celu zapewnienia stałej ilości programistów w projekcie.

Shame Driven Development

Lider zespołu SDD musi zadbać o odzież ochronną dla siebie

Metodyka nieco podobna do Guilt Driven Development, ją również można implementować przy pomocy narzędzi automatycznych, ale daje dużo lepsze efekty, gdy robią to ludzie. Podstawowym założeniem jest rezygnacja z jakichkolwiek wymagań przy commitowaniu i merge'owaniu kodu. Tak, żadnych durnych code review, żadnych unit testów, kod nie musi się nawet kompilować, cała odpowiedzialność spoczywa na programiście. Jeśli w kodzie będą jakieś bugi (a będą), należy delikwenta publicznie wyśmiać i zagonić do

> *Ten gówniany commit został oznaczony jako „stabilny”, ale coś mi się wydaje, że nawet tego nie skompilowałeś!*[2]

> *Oczywiście, sugerowałbym też żeby geniusz który myślał, że czytanie danych BAJT PO J[CENZURA]YM BAJCIE to dobry pomysł powinien zostać poddany wstecznej aborcji. Kto k[CENZURA]a robi tak idiotyczne rzeczy? Jakim cudem nie umarł jako dziecko, skoro prawdopodobnie był zbyt głupi, by znaleźć cycka do ssania?*[3]

I *crème de la crème*:

> *Mauro, ZAMKNIJ [CENZURA] RYJ!*[4]

Skutki uboczne SDD są w gruncie rzeczy podobne do skutków GDD, więc nie będziemy się powtórnie nad nimi rozwodzić.

Bug Driven Development

Bardzo naturalna metodyka, która głęboko rezonuje z każdym pełnokrwistym programistą. Dobremu deweloperowi nie trzeba jej nawet tłumaczyć, bo jest dlań oczywista i intuicyjna. Podstawowym założeniem jest napisanie jak największej ilości kodu w jak najkrótszym czasie, byle się kompilowało, nic więcej, nie musi nawet za bardzo działać. Następnie w kolejnych iteracjach należy łatać bugi do skutku.

Technika *Bug Driven Development* gwarantuje zakończenie projektu w skończonym czasie i budżecie. Niestety, górnym oszacowaniem czasu potrzebnego na wdrożenie całości jest Apokalipsa, dlatego kluczowym jest zachowanie optymizmu i wytrwałe targowanie się z klientem[5].

Stack Overflow Driven Development

Awangardowa metodyka uprawiana głównie przez młodych i ambitnych programistów. Wbrew swej nazwie, SODD nie ma żadnego związku ze strukturą stosu ani jej przepełnieniem – nie trzeba nawet wiedzieć co to jest stos! Na dobrą sprawę to w ogóle nie trzeba nic wiedzieć.

Metodyka *Stack Overflow Driven Development* została ściśle zdefiniowana następującymi krokami:

1. Napisz kawałek kodu i spróbuj go skompilować.
2. Oczywiście że się nie skompilowało. Sprawdź jaki błąd wypluł kompilator i skopiuj tekst błędu do schowka.
3. Wklej ten błąd w wyrocznię i przeczytaj, co mądre głowy na Stack Overflow mają do powiedzenia na temat.
4. Skopiuj losowo wybrany kod ze Stacka do swojego programu

Kroki te należy powtarzać do skutku, czyli do wyklepania koszmarnie zabugowanego i powiązanego sznurkiem od snopowiązałki kodu, albo do załamania nerwowego programisty.

Programowanie ekstremalne

Praojciec wszystkich najciekawszych metodyk zwinnych, otoczony tak wielkim szacunkiem, że zasłużył sobie na własny artykuł.

> *Główny artykuł: Programowanie ekstremalne*

Zobacz też

- coaching
- inżynieria oprogramowania
- programista
- język programowania

Przypisy

1. Kolejny hipsterski wymysł
2. Źródło
3. Źródło
4. Źródło
5. *It's not a bug, it's a feature!*

Zobacz więcej artykułów w **portalu o informatyce**.

CZY NIE WIESZ...

Że odkurzacz jest jednym z największych osiągnięć ludzkości?