Rozdział 1

Wstęp

W ciągu ostatnich paru lat, w informatyce można zaobserwować wyjątkowo szybki rozwój tematyki przetwarzania w chmurze obliczeniowej (ang. Cloud Computing), a co za tym idzie, zainteresowanie stylami i wzorcami architektonicznymi, które w efektywny sposób są w stanie wykorzystać potencjał aplikacji rozproszonych. Takie podejście prezentuje między innymi idea mikro-serwisów, czyli małych, niezależnych komponentów komunikujących się między sobą prostymi protokołami sieciowymi[**?**]. Najczęściej stosowanym rozwiązaniem jest używanie komunikacji HTTP w konwencji REST, gdyż zostało to zarekomendowane przez prekursorów idei rozproszonych mikrousług - Martina Fowlera oraz Jamesa Levisa[**?**].

Oczywistym jest, że wraz ze wzrostem skomplikowania i rozmiaru systemu bazującego na architekturze mikroserwisowej, powstaje coraz więcej usług, które komunikują się ze sobą nieustannie. Nieodzownym elementem testowania takiego oprogramowania jest więc sprawdzanie punktów styku między usługami (and. end points) pod kątem ich poprawności i zgodności ze specyfikacją. Zgodnie z wiedzą autora, w systemach zgodnych z architekturą mikroserwisową, testy jednostkowe skupiają się na funkcjonalnościach każdej z usług, a walidacja poprawności zwracania i wymiany danych jest często ignorowana. W środowisku pracy zorganizowanym wokół mikroserwisów krytyczna jest komunikacja między zespołami pracującymi nad połączonymi mikrousługami. Taka komunikacja ma na celu uzgodnienie sposobów integracji wielu komponentów w postaci mikrousług, a jej efektem jest ustalenie i spisanie dokumentacji technicznej, będącej podstawą pisania nowych konsumentów danej usługi. Specyfikacja taka powinna być spójna i logiczna, dlatego często stosuje się języki formalne, które w swojej semantyce wykluczają lub minimalizują ryzyko sprzeczności lub niedomówień.

1.1 Cel pracy

Niniejsza praca ma na celu zaprezentowanie modelu narzędzia do generowania testów jednostkowych połączeń między mikrousługami na podstawie ich specyfikacji zapisanej w języku RAML, zaimplementowanie tego modelu w języku C#, a także dokonanie weryfikacji istniejącego systemu opartego o komunikację REST w celu oceny skuteczności zaproponowanego rozwiązania.

1.2 Struktura pracy

W rozdziale pierwszym zaprezentowany jest zarys obecnej sytuacji w dziedzinie rozwoju oprogramowania oraz problemy jakie stawia przed nimi szybki rozwój aplikacji rozproszonych. Został tam też zamieszczony opis zakresu pracy, która jest odpowiedzią na problemy stające przed zespołami rozwijającymi nowoczesne systemy informatyczne. Częścią tego rozdziału jest również niniejsze przedstawienienie struktury pracy.

Rozdział drugi ma na celu wprowadzenie czytelnika w tematykę postawionego problemu oraz opisanie głównych pojęć związanych z tematyką pracy. Przeanalizowany zostanie styl architektoniczny oparty o mikrousługi, uwidaczniając przyczyny jego powstania, zasady działania oraz najbardziej znaczące zalety. Nieuniknione będzie również zidentyfikowanie słabych stron i procesów,

które według publikacji, mogłyby je redukować. Pamiętając o komunikacji mikrousług, opisana zostanie najczęściej używana metoda ich komunikacji czyli interfejsy REST z użyciem protokołu HTTP. W dalszym ciągu rozdziału nastąpi przybliżenie potrzeby dokumentacji interfejsów programistycznych (*API*), korzyści z nich płynących oraz standardów. Ta sekcja zostanie opisana na przykładzie języka RAML w wersji 1.0. W końcowej części tego rozdziału zaprezentowany zostanie koncept testowania jednostkowego, zasady pisania takich testów oraz korzyści z nich płynące.

Rozdział trzeci zawiera opis proponowanego przez autora rozwiązania. Począwszy od modelu tworzonego narzędzia, poprzez projekt architektury zaprezentowane zostanie rozwiązanie problemów związanych z rosnącą popularnością komunikacji międzyusługowej. Rozdział uwzględnia również zmiany kultury pracy i dyscypliny deweloperskiej, która jest niezbęna przy stosowaniu takich rozwiązań procesowych. W końcowej części rozdziału spojrzymy na rezultaty implementacji, przedstawiając główne funkcjonalności.

W rozdziale czwartym skupimy się na weryfikacji proponowanego rozwiązania. Przybliżony zostanie istniejący system oparty o komunikację sieciową HTTP/REST. Następnie opisana zostanie metodologia przeprowadzania testów, czyli kryteria testowania oraz sposób weryfikowania poprawności działania stworzonego rozwiązania. W ostatniej sekcji tego rozdziału zaprezentowane zostaną otrzymane rezultaty testów i weryfikacji.

Ostatni rozdział zawiera anailizę rezulatów testów wyszczególnonych w rozdziale czwartym‘, a następnie kreślone zostaną wyciągnięte wnioski na podstawie analizy wyników. W dalszej części, wybiegając w przyszłość, zidentyfikowane zostaną najważniejsze ścieżki udoskonalenia narzędzia nakreślając tym samym kierunki rozwoju całej dziedziny poruszanej w niniejszej pracy. W ostatniej sekcji całej pracy zostanie ona podsumowana.