ANDRZEJ SOBCZAK*

KRYTERIA I METODY EWALUACJI ARCHITEKTURY SYSTEMÓW OPROGRAMOWANIA[1]

Definicyjne ujęcie ewaluacji

W chwili obecnej nie ma jednoznacznego ujęcia zagadnienia ewaluacji. W opracowaniu *Ewaluacja w administracji publicznej – funkcje, standardy i warunki stosowania* przytoczona została definicja M. Pattona, wieloletniego prezesa Amerykańskiego Towarzystwa Ewaluacyjnego, który określił ewaluację jako proces systematycznego gromadzenia informacji na temat działania, właściwości i rezultatów, personelu oraz produktów, które są wykorzystywane przez specjalistów do redukowania niepewności wokół działania i poprawienia jego efektywności, a także służą podejmowaniu decyzji z uwagi na to, co owe działania, personel lub produkty robią oraz czego dotyczą [5, s. 6–7]. H. Komorowska określa ewaluację jako proces zbierania danych i ich interpretację w celu podejmowania decyzji [7, s. 44]. Charakteryzuje ona ten proces poprzez wskazanie następujących jego cech:

- motywem podjęcia ewaluacji jest potrzeba rozwiązania określonego problemu;
- celem ewaluacji jest podjęcie racjonalnej decyzji;
- analiza podczas ewaluacji ma charakter idiograficzny;
- eksplanacja w ewaluacji jest sprawą wtórną, ważniejszy jest opis i ocena;
- autonomia prac podejmowanych w czasie ewaluacji jest ograniczona poprzez wskazanie celu ewaluacji przez zamawiających;
- rezultaty ewaluacji muszą być przydatne dla zamawiającego.

Na potrzeby niniejszego artykułu przyjmuje się, że ewaluacja jest to systematyczne badanie wartości albo cech konkretnego (...) obiektu z punktu widzenia przyjętych kryteriów w celu jego usprawnienia, rozwoju lub lepszego rozumienia [4, s. 5].

---

* Katedra Informatyki Gospodarczej, Szkoła Główna Handlowa, Al. Niepodległości 162, 02-554 Warszawa.

1 Praca została wykonana w ramach projektu badawczego nr N115 010 32/01443, finansowanego przez Ministerstwo Nauki i Szkolnictwa Wyższego.

Pojęcie architektury systemu oprogramowania

I. Graham postawił tezę, że „nie ma dotychczas jasnej i jednoznacznej definicji tego, czym jest architektura systemu oprogramowania" [6, s. 360]. Najlepszym na to dowodem jest fakt zgromadzenia na stronie internetowej Software Engineering Institute Carnegie Mellon University blisko 50 definicji – zaczerpniętych z literatury oraz zaproponowanych przez internautów – terminu „architektura systemu oprogramowania" [10].

Standard IEEE 1471–2000 *Recommended Practice for Architectural Description of Software-Intensive Systems* definiuje architekturę systemu oprogramowania jako: fundamentalną organizację systemu zawierającą w sobie: elementy, relacje między tymi elementami oraz środowisko i pryncypia projektowania i ewolucji tej organizacji. Przy czym: „fundamentalna organizacja" oznacza zasadnicze, ujednolicone koncepcje i zasady; określenie „system" obejmuje system oprogramowania, platformę, linię produktową; środowisko rozumiane jest jako projektowy, operacyjny lub programistyczny kontekst systemu.

L. Bass, P. Clements i R. Kazman definiują architekturę systemu oprogramowania jako strukturę (lub zbiór struktur) tego systemu, która składa się z określonych elementów, widzianych z zewnątrz właściwości tych elementów oraz związków między nimi [2, s. 360].

W. Chmielarz definiuje architekturę systemu jako układ współpracujących ze sobą następujących elementów: struktury oprogramowania, fizycznej struktury danych, mechanizmów gwarantujących bezpieczeństwo danych. Dodatkowo W. Chmielarz zauważa, że: „pełna specyfikacja problematyki architektury systemu [informatycznego – przypis autora] wymaga jeszcze ponadto uwzględnienia w rozważaniach zagadnień sprzętowych oraz związanych z organizacją procesu przetwarzania" [3, s. 131].

Na koniec przytoczona zostanie definicja, która została przyjęta na potrzeby niniejszej pracy. P. Kroll i P. Kruchten rozpatrują natomiast architekturę jako zbiór znaczących decyzji dotyczących struktury systemu, czyli: wyboru elementów strukturalnych i ich interfejsów, z których składa się system, zachowań określających współpracę tych elementów, zorganizowania tych elementów strukturalnych i behawioralnych w większe podsystemy, stylu architektonicznego, który ma wpływ na kształt tej struktury.

Znaczenie ewaluacji architektury oprogramowania

Bazując na ogólnej definicji ewaluacji, można wskazać, że ewaluacja architektury systemu oprogramowania jest to systematyczne badanie cech tejże architektury z punktu widzenia przyjętych kryteriów ewaluacji w celu jej usprawnienia, rozwoju lub lepszego rozumienia.

Ewaluacja architektury może zostać przeprowadzona w różnych momentach procesu tworzenia systemu oprogramowania. Jeśli architektura jest jeszcze w stanie zalążkowym, możliwa jest ewaluacja pierwszych (najczęściej krytycznych) decyzji

architektonicznych. Jeśli projektowanie architektury zostało zakończone, ale prace implementacyjne na jej podstawie nie rozpoczęły się jeszcze, wskazana jest ewaluacja całej architektury.

Uzasadniona jest również ewaluacja architektury systemu odziedziczonego (*legacy systems*), który podlegać będzie procesowi doskonalenia. Innym obszarem ewaluacji jest jej zastosowanie podczas nabywania dużego systemu oprogramowania u dostawcy zewnętrznego. Ewaluacja pozwala nie tylko ocenić, czy realizuje on określone potrzeby organizacji (można dokonać ewaluacji kilku konkurencyjnych rozwiązań i wskazać to, które w największym stopniu odpowiada potrzebom organizacji), ale umożliwia zrozumienie jego architektury.

L. Clements, P. Kazman i R. Klain przytaczają opinię, że „architektura złożonego oprogramowania (...) stanowi całą serię trudnych decyzji, których zmiana jest bardzo kosztowna; sukces rozwoju i ewolucji [systemów oprogramowania – przypis autora] jest uzależniony od dokonywania prawidłowych wyborów związanych z konstrukcją ich architektury" (8, s. 11). W związku z tym zastosowanie skutecznej metody ewaluacji zaprojektowanej architektury stanowi w istotnym stopniu o sukcesie lub porażce całego przedsięwzięcia. W szczególności przeprowadzenie ewaluacji pozwala na [2, s. 281–286]:

- efektywniejsze zarządzanie środkami finansowymi; ewaluacja – jak to było wskazane wcześniej – nastawiona jest na cele utylitarne; jej przeprowadzenie przekłada się średnio na zmniejszenie o 10% kosztów prowadzonego przedsięwzięcia;
- lepsze udokumentowanie architektury; w ramach przygotowań do ewaluacji następuje przegląd i wniesienie poprawek do istniejącej dokumentacji architektury; bardzo często okazuje się, że istniejący opis jest albo zbyt krótki, albo (coraz częściej – w dobie narzędzi CASE – się to zdarza) zbyt rozbudowany;
- uchwycenie uzasadnienia; ewaluacja pozwala na uzyskanie i udokumentowanie odpowiedzi w zakresie podejmowanych decyzji architektonicznych, co jest niezwykle istotne podczas dalszego rozwoju systemu oprogramowania;
- wczesne wykrycie problemów dotyczących istniejącej architektury; ewaluacja pozwala na zidentyfikowanie sytuacji problemowych (takich jak na przykład problemy związane z pielęgnowalnością, przenoszalnością, wydajnością systemu) i podjęcie niezbędnych kroków zaradczych;
- weryfikację wymagań; ewaluacja – ze względu na fakt, iż podczas jej trwania zaangażowani są przedstawiciele różnych grup interesariuszy – stanowi platformę do dodatkowej weryfikacji wymagań; wynikiem tego jest najczęściej uświadomienie sobie sprzeczności występującej między wymaganiami (na przykład: z jednej strony wysoka wydajność i bezpieczeństwo, a z drugiej niskie koszty) i przeprowadzenie negocjacji w tym zakresie;
- udoskonalone zarządzanie architekturą; jeżeli ewaluacja architektury stanie się standardową praktyką, wówczas w organizacji rozwija się atmosfera do stosowania działań na rzecz zapewnienia jakości w ramach całego procesu tworzenia systemów oprogramowania.

Czynniki wpływające na skuteczność ewaluacji

L. Bass, P. Clements i R. Kazman wskazali na niezbędne ich zdaniem warunki, które pozwalają zakończyć ewaluację sukcesem [2, s. 281–286].

- Jasno wyrażone cele i wymagania wobec ewaluowanej architektury. Wymagania wobec architektury stają się kryteriami oceny i jedynie one pozwolą na rzetelną ewaluację architektury oraz sformułowanie przydatnych dla zamawiającego wniosków.
- Ustalony zakres ewaluacji. Należy jednoznacznie określić, jaki fragment systemu oprogramowania będzie poddawany ewaluacji – czy cały system, czy może tylko jego kluczowe komponenty.
- Efektywność kosztowa. Zamawiający ewaluację (po konsultacji z ekspertami) powinni dobrać sposób ewaluacji do jej zakresu. Metody, które są właściwe dla dużego zakresu ewaluacji, mogą nie być efektywne kosztowo w wypadku mniejszych przedsięwzięć.
- Dostępność kluczowego personelu. Niezbędne jest zapewnienie dostępu do głównego architekta systemu lub innej osoby, która zna przesłanki podjęcia określonych decyzji architektonicznych. Niektóre metody ewaluacji wymagają także dostępu do kluczowych interesariuszy (czasami wręcz reprezentantów kierownictwa organizacji) tworzonego systemu.
- Kompetentny zespół ewaluatorów. Wskazane jest wykorzystanie zewnętrznego zespołu, wyspecjalizowanego w procesie ewaluacji (dzięki temu efekty ewaluacji będą postrzegane jako obiektywne).
- Kontrolowane oczekiwania. Niezbędne jest precyzyjne określenie celów ewaluacji, co będzie jej wynikiem, jakie obszary będą badane (a jakie nie), ile czasu i zasobów przedsięwzięcia pochłonie ewaluacja oraz komu zostaną przekazane jej wyniki.

Kryteria ewaluacji architektury

Projektowana architektura może być ewaluowana w trzech obszarach.

Obszar cech jakościowych, które zbudowany na podstawie architektury system powinien zapewniać. W szczególności chodzi tutaj o zapewnienie realizacji określonej funkcjonalności – tj. zdolności systemu do pracy, do której został on przeznaczony. Do innych atrybutów jakościowych zalicza się:

- Niezawodność – zdolność wykonywania systemu oprogramowania do wykonywania przewidywanych zadań w ustalonych warunkach. Może być ona mierzona prawdopodobieństwem, z jakim oczekuje się, że oprogramowanie poprawnie realizuje swoje funkcje w określonym przedziale czasu.
- Dostępność – procent czasu, w którym cały system oprogramowania lub dany jego obiekt jest sprawny.
- Wydajność – zdolność systemu oprogramowania do szybkiego wykonywania przewidzianych zadań, przy efektywnym wykorzystaniu zasobów. Wydajność nie jest tożsama z efektywnością, gdyż ta ostatnia cecha uwzględnia także elementy

związane ze sprzętem (takie jak na przykład: czas wykorzystania procesora, wielość pamięci zewnętrznej i wewnętrznej, zajętość urządzeń zewnętrznych i telekomunikacyjnych).

- Łatwość konserwacji – cecha systemu oprogramowania odzwierciedlająca to, jak łatwo jest w tym systemie dokonywać zmian.
- Zdolność do współdziałania – cecha systemu oprogramowania umożliwiająca współpracę z innymi systemami, w tym również takimi, które nie były przewidywane na etapie projektowania.
- Zabezpieczenie – zdolność systemu oprogramowania do przeciwdziałania niepowołanemu dostępowi do kodu systemu oraz zasobów danych. W ramach zabezpieczeń występują zagadnienia autentykacji i autoryzacji.
- Bezpieczeństwo – zdolność systemu oprogramowania do osiągania akceptowalnie małego poziomu ryzyka dla wystąpienia zdarzenia skutkującego utratą życia lub zdrowia i dużymi stratami finansowymi lub zanieczyszczeniem środowiska.
- Użyteczność – zdolność systemu oprogramowania do działania w określonych warunkach, w określony sposób i tak, aby było zrozumiałe, łatwe do nauczenia i atrakcyjne w użyciu.

Obszar spełnienia wymagań biznesowych dotyczących systemu. W ramach tego obszaru identyfikuje się następujące atrybuty:

- Czas wprowadzenia systemu oprogramowania, który ma zostać zbudowany na podstawie opracowanej architektury, na rynek.
- Koszty i korzyści finansowe dla organizacji związane z wprowadzeniem danego systemu oprogramowania.
- Przewidywalny czas eksploatacji systemu oprogramowania.
- Rynek, na który ma być wprowadzony dany system oprogramowania.
- Łatwość integracji z istniejącymi już w organizacji rozwiązaniami.

Trzeci obszar ewaluacji jest konstytuowany przez grupę atrybutów, które odnoszą się do architektury jako do produktu. Jest ona wówczas charakteryzowana za pomocą:

- koncepcyjnej jednorodności – polegającej na zapewnieniu spójnej wizji architektury na wszystkich jej poziomach;
- poprawności i kompletności;
- możliwości realizacji architektury w formie działającego systemu w określonym przedziale czasowym, w ramach określonego budżetu i przy istniejących zasobach ludzkich.

Charakterystyka istniejących metod ewaluacji architektury

Metody ewaluacji architektury systemów oprogramowania można podzielić na trzy grupy: metody bazujące na pytaniach, metody pomiarowe oraz metody hybrydowe (łączące cechy dwóch pierwszych rodzajów).

Metody bazujące na pytaniach polegają na przeprowadzeniu określonych eksperymentów myślowych, przez co mogą być stosowane już na wczesnych etapach prac nad architekturą (w tym w zakresie podejmowania decyzji architektonicznych).

W ramach tej grupy stosowane są kwestionariusze, listy kontrolne oraz scenariusze. Kwestionariusz ma postać listy ogólnych pytań, często o charakterze otwartym. Pytania mogą dotyczyć na przykład sposobu tworzenia i dokumentowania architektury oraz odnosić się bezpośrednio do architektury jako produktu końcowego pewnego działania. Lista kontrolna jest zestawem szczegółowych pytań, które zostały opracowane na bazie doświadczeń uzyskanych z ewaluacji systemów z tego samego obszaru dziedzinowego. Pytania te dotyczą najczęściej poszczególnych funkcji lub atrybutów jakościowych systemu. Scenariusze mają postać opisu interakcji interesariusza systemu (najczęściej jest to użytkownik systemu) bezpośrednio z samym systemem. Scenariusze tworzone są na potrzeby konkretnego systemu, z tego też powodu są one tworzone przez interesariuszy podczas procesu ewaluacji architektury. Istnieje możliwość ponownego wykorzystania scenariuszy w procesie ewaluacji architektury innych systemów – w takim przypadku stają się one składowymi list kontrolnych.

Poniżej pokrótce scharakteryzowane główne metody bazujące na pytaniach [1, 6, 9]:

- Metoda analizy architektury oprogramowania (Software Architecture Analysis Method, SAAM) była pierwszą (powstała w 1993r.) udokumentowaną, powszechnie stosowaną metodą analizy architektury systemów oprogramowania. Pierwotnie została ona opracowana w celu analizy architektury pod względem jej modyfikowalności. Jednak w rzeczywistych warunkach metoda SAAM okazała się również możliwa do zastosowania odnośnie do innych atrybutów jakościowych (takich jak przenośność, rozszerzalność, integralność), a także do identyfikacji konfliktów między wymaganiami architektonicznymi.
- Metoda SAAMCS (SAAM Founded on Complex Scenarios) stanowi rozszerzenie metody SAAM i koncentruje się na czynnikach ryzyka związanych ze złożonością scenariuszy. Jest ona stosowana do finalnej wersji architektury, opisanej w szczegółach. W ujęciu tym system oprogramowania rozpatrywany jest nie w izolacji, lecz jako zintegrowany z otoczeniem. Opis architektury dzielony jest tutaj na makroarchitekturę i mikroarchitekturę.
- Metoda SAAMER (Software Architecture Analysis Method for Evolution and Reusability) stanowi rozszerzenie metody SAAM i koncentruje się na analizie aspektów związanych z dwoma atrybutami jakościowymi: łatwością rozwoju i ponownym użyciem.
- Metoda ARID (Active Reviews for Intermediate Designs) dedykowana jest prowadzeniu ewaluacji architektury w jej wczesnej fazie projektowania – zwłaszcza w zakresie modułów i komponentów. Opiera się ona na technice przeglądów z zastosowaniem scenariuszy oraz na czynnych przeglądach projektów (ang. Active Design Reviews). Osoby biorące udział w ewaluacji dobierane są w taki sposób, aby reprezentowani byli przyszli użytkownicy zarówno systemu, który powstanie na bazie architektury, jak i jej dokumentacji (tj.: programiści, integratorzy, testerzy).
- Metoda ATAM (Architecture Tradeoff Analysis Method) zapewnia przeprowadzenie oceny, w jakim stopniu dana architektura systemu oprogramowania spełnia określone wymagania jakościowe, oraz umożliwia analizę wpływu na

siebie zdefiniowanych wymagań architektonicznych, a także identyfikację kompromisów, jakie są wymagane w celu zapewnienia koegzystencji tych wymagań. Wskazuje ona, w jaki sposób należy identyfikować konflikty i ich rozwiązania w ramach architektury systemu.

- Metoda CBAM (Cost-Benefit Analysis Method) bazuje na metodzie ATAM i umożliwia ocenę problemów technicznych i ekonomicznych związanych z projektowaniem architektury oraz służy do modelowania kosztów i korzyści związanych z decyzjami architektonicznymi. CBAM zaczyna się w miejscu, w którym ATAM się kończy, i w istocie zależy od produktów będących wynikami zastosowania metody ATAM.
- Metoda QAW (Quality Attribute Workshop) umożliwia zrozumienie problemów poprzez wydobycie atrybutów jakościowych wymagań za pomocą scenariuszy testowych. Metoda ta powstała na podstawie metody ATAM, ale w przeciwieństwie do niej może być stosowana do oceny różnych propozycji architektury, które nie są jeszcze do końca dopracowane. Pozwala to na usprawnienie procesu komunikacji między architektami a kluczowymi interesariuszami, wczesne poznanie konsekwencji pewnych decyzji architektonicznych – w tym rozpoznanie kształtu tworzonego systemu oraz ulepszenie lub zmianę przyjętych rozwiązań.
- Metoda SBAR (Scenario-Based Architecture Reengineering) pozwala na ocenę wpływu przyjętej architektury na osiągnięcie zidentyfikowanych atrybutów jakościowych na system. W ramach tej metody używane są cztery techniki: scenariuszowa, modelowania matematycznego, symulacji, wnioskowania na podstawie doświadczeń (experience-based reasoning). Technika scenariuszowa jest zalecana do oceny atrybutów związanych z rozwojem systemów – tj. konserwowalnością oraz ponownym użyciem. Symulacje uzupełniają technikę scenariuszową w zakresie weryfikacji wydajności i odporności na uszkodzenia. Modelowanie matematyczne pozwala na analizę statycznych elementów architektury oraz ocenę niezawodności systemu w warunkach dużego obciążenia. Ostatnia technika obejmuje wnioskowanie logiczne bazujące na zgromadzonych wcześniejszych doświadczeniach. Ma ona bardziej miękki, subiektywny charakter. Wykorzystuje się tutaj wiedzę ukrytą osób zaangażowanych w prace nad architekturą – tj. architektów systemu.
- Metoda ALPSM (Architecture Level Prediction of Software Maintenance) dedykowana jest do analizy konserwowalności (rozumianej jako łatwość konserwacji) systemów oprogramowania poprzez obserwację wpływu scenariuszy zmian na ich architekturę. ALPSM możliwa jest do zastosowania w przypadku istniejących już rozwiązań. W ramach tej metody zaangażowani są jedynie projektanci systemów. Określają oni tzw. profile konserwacji definiowane jako zbiory scenariuszy zmian – już dokonanych i planowanych do realizacji. Scenariusz opisuje akcję lub sekwencje akcji, które związane są z wprowadzeniem określonej zmiany do systemu.
- Metoda SACAM (The Software Architecture Comparison Analysis Method) umożliwia przeprowadzenie analizy porównawczej architektury systemów oprogramowania. Można ją zastosować do wskazania tej architektury, spośród kilku

propozycji, która w największym stopniu będzie spełniać wymagania biznesowe sformułowane dla systemu oprogramowania. Przy czym przeprowadzona analiza bazować będzie na atrybutach jakościowych, a nie tylko związanych z kosztami. Po drugie metoda ta zorientowana jest na cele biznesowe, które powinien realizować system. Dodatkowo metoda ta pozwoli na analizę typu „co–jeżeli" – czyli jak zmiany na poziomie celów przełożą się na architekturę systemu.

Przedstawiane do tej pory metody ewaluacji architektury miały charakter jakościowy. W wielu wypadkach podejście takie jest jednak niewystarczające (zwłaszcza przy ewaluacji architektury systemów, które będą pracować w architekturze trójwarstwowej i z założenia są przeznaczone dla bardzo dużej grupy użytkowników). Dlatego podejmowane są wysiłki na rzecz opracowania ilościowych metod ewaluacji architektury. Wyróżnia się tutaj dwa podejścia: miary oraz eksperymenty [1, s. 11].

Miary wskazują na te miejsca w architekturze, w których z dużym prawdopodobieństwem będą zachodziły zmiany lub gdzie wprowadzanie zmian będzie mocno utrudnione. Do najczęściej wykorzystywanych miar zaliczyć można: głębokość struktury (liczba warstw enkapsulacji danego obiektu), używanie modułów (liczba modułów, które używa dany element pośrednio lub bezpośrednio), liczbę klastrów komponentu (liczba klastrów, z których składa się analizowany obiekt), liczbę zdarzeń (liczba synchronicznych i asynchronicznych zdarzeń, na które reaguje dany element).

Inną metodą ewaluacji architektury jest przygotowywanie prototypów (składają się na nie komponenty, które mają postać struktur szczątkowych) lub symulacji i przeprowadzanie z ich pomocą eksperymentów. Przykładem podejścia bazującego na symulacji jest metoda LQN (Layered Queuing Network), w której następuje transformacja architektury do modelu warstwowego, zbudowanego z wykorzystaniem kolejek; model ten opisuje interakcje między poszczególnymi komponentami i czasy przetwarzania niezbędne do wykonywania tych interakcji; metodę tę wykorzystuje się m. in. do ewaluacji architektury systemu pod kątem kryteriów wydajnościowych [8].

Podsumowanie

Przedstawione w artykule rozważania pozwalają stwierdzić, że dysponując zaprojektowaną architekturą, mamy możliwość wnioskowania o systemie, zanim zostanie on zbudowany. Ma to istotną wartość dla organizacji, gdyż pozwala na stosunkowo mało kosztowne ograniczenie ryzyka porażki całego przedsięwzięcia. Z tego powodu uważa się, że ewaluacja architektury powinna być standardową częścią każdego architekturocentrycznego procesu tworzenia systemów oprogramowania.

Literatura

1. Abowd G., Bass L., Clements P., Kazman R., Northrop L., Zaremski A., *Recommended Best Industrial Practice for Software Architecture Evaluation*, Technical Report CMU/SEI-96-TR-025, Carnegie Mellon University, Software Engineering Institute, 13 January 1997.
2. Bass L., Clements P., Kazman R., *Architektura oprogramowania w praktyce*, Wydawnictwa Naukowo-Techniczne, Warszawa 2006.
3. Chmielarz W., *Systemy informatyczne wspomagające zarządzanie*, Aspekt modelowy w budowie systemów, Dom Wydawniczy Elipsa, Warszawa 1996.
4. Ciężka B., Chojecki J., Ratajczak J., *Ewaluacja funduszy strukturalnych*, Polskie Towarzystwo Ewaluacyjne, Warszawa, wrzesień 2003.
5. Ekiert K., *Ewaluacja w administracji publicznej – funkcje, standardy i warunki stosowania*, opracowanie Rządowego Centrum Studiów Strategicznych, Warszawa 2004.
6. Graham I., O'Callaghan A., Wills A., *Metody obiektowe w teorii i praktyce*, Wydawnictwa Naukowo-Techniczne, Warszawa 2004.
7. Komorowska H., *Konstrukcja, realizacja i ewaluacja programu nauczania*, Instytut Badań Edukacyjnych, Warszawa 1995.
8. Kroll P., Kruchten P., *Rational Unified Process od strony praktycznej*, Wydawnictwa Naukowo-Techniczne, Warszawa 2007.
9. Mattsson M., Grahn H., Mårtensson F., *Software Architecture Evaluation Methods for Performance, Maintainability, Testability, and Portability*, Proceedings of the Second International Conference on the Quality of Software Architectures, Västerås, Sweden 2006.
10. http://www.sei.cmu.edu/architecture/definitions.html.