Jarosław Żeliński

Synteza pojęć i wzorców MOF, MDA, PIM, MVC i BCE

Zintegrowany model struktury procesu projektowania aplikacji

Streszczenie: Publikacje, w tym podręczniki akademickie, zawierają wiele niespójności w opisach zastosowań metod i wzorców architektonicznych kryjących się pod skrótami MOF, MDA, PIM, MVC, BCE. Skuteczna analiza oraz następujące po niej projektowanie oprogramowania, szczególnie gdy są to projekty realizowane w dużych zespołach, wymaga standaryzacji procesu wytwórczego i stosowanych wzorców i frameworków. W pracy tej podjęto próbę uporządkowania systemu pojęć opisujących ten proces i stosowanych do opisu wzorców architektonicznych. Przeprowadzono analizę kluczowych pojęć MOF i MDA, wzorców MVC i BCE, stworzono spójny system łączący je w jedną całość.

Słowa Kluczowe: architektura oprogramowania, wzorce projektowe

1. Wprowadzenie

Celem badań było zweryfikowanie obecnego stanu metod projektowania i opracowanie spójnego systemu pojęć i wzorców projektowych w obszarze projektowania logiki oprogramowania, jako jej abstrakcyjnego modelu. Wiele publikacji na temat analiz i projektowania, w obszarze inżynierii oprogramowania, przywołuje nazwy wzorców projektowych MVC i BCE oraz model PIM (patrz OMG MDA, OMG MOF 2016). Z uwagi na, nie raz, nie małe rozbieżności w interpretacji tych metod i wzorców, autor podjął próbę uporządkowania ich wzajemnych zależności.

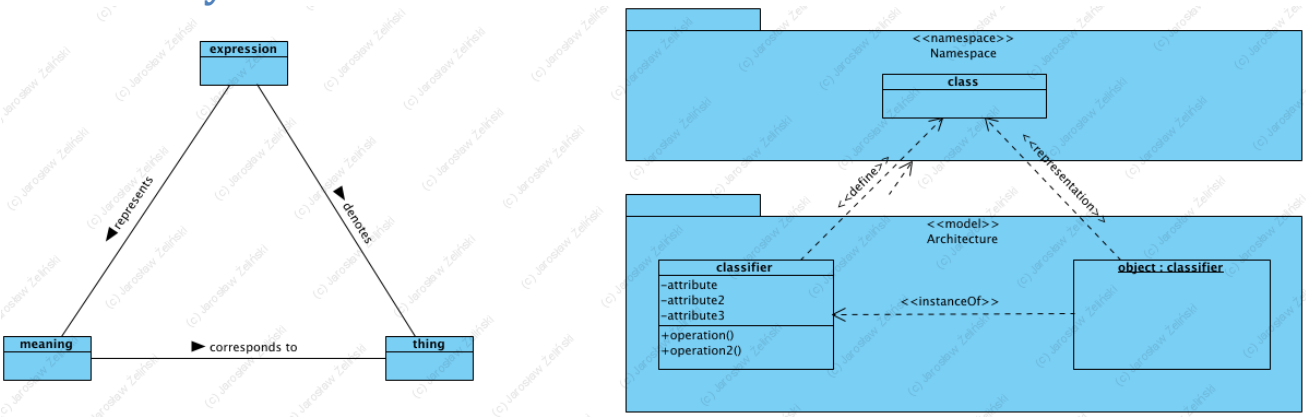
2. Metody

Wykorzystano systemy notacyjne Object Management Group (OMG.org). Specyfikacja MOF opisuje trzy poziomy abstrakcji: M1, M2, M3 oraz poziom M0 czyli realne rzeczy (patrz struktura poziomów abstrakcji, OMG MOF 2016). M0 to realny system, poziom M1 to abstrakcja obiektów tego systemu (jego model) . Poziom M2 to związki pomiędzy klasami tych obiektów (nazwy ich zbiorów) czyli metamodel systemu. Poziom M3 to meta-metamodel poziom opisujący metodę modelowania z użyciem nazwanych elementów o określonej semantyce i syntaktyce.

Proces analizy i projektowania został oparty na specyfikacji MDA (OMG MDA). Proces ten ma trzy fazy rozumiane jako tworzenie kolejnych modeli: CIM, PIM, PSM oraz fazę tworzenie kodu. Model CIM jest dokumentowany z użyciem notacji BPMN (OMG BPMN 2013) i SBVR (OMG SBVR 2017). Są to odpowiednio: modele procesów biznesowych oraz modele pojęciowe i reguły

biznesowe. Modele PIM i PSM dokumentowane są z użyciem notacji UML (OMG UML 2017). Pomiędzy modelem CIM a PIM ma miejsce ustalenie listy usług aplikacyjnych (reakcje systemu), których mechanizm realizacji opisuje model PIM. Standardowym wzorcem używanym do modelowania architektury aplikacji jest wzorzec MVC. Komponent Model tego wzorca jest modelowany z użyciem wzorcza architektonicznego BCE.

2.1. Semiotyka a UML



Rys. Semiotyka a UML

Semiotyka, jako nauka o symbolach i tym co one znaczą, daje nam narzędzie pozwalające określić zależności pomiędzy obiektem (thing), jego nazwą (expression) i definicją pojęcia reprezentowanego tą nazwą (lub znakiem, meaning). Zależności te nazywane są trójkątem semiotycznym. Na rys. Semiotyka a UML trójkąt ten zobrazowano po lewej stronie (na podstawie OMG SBVR 2017).

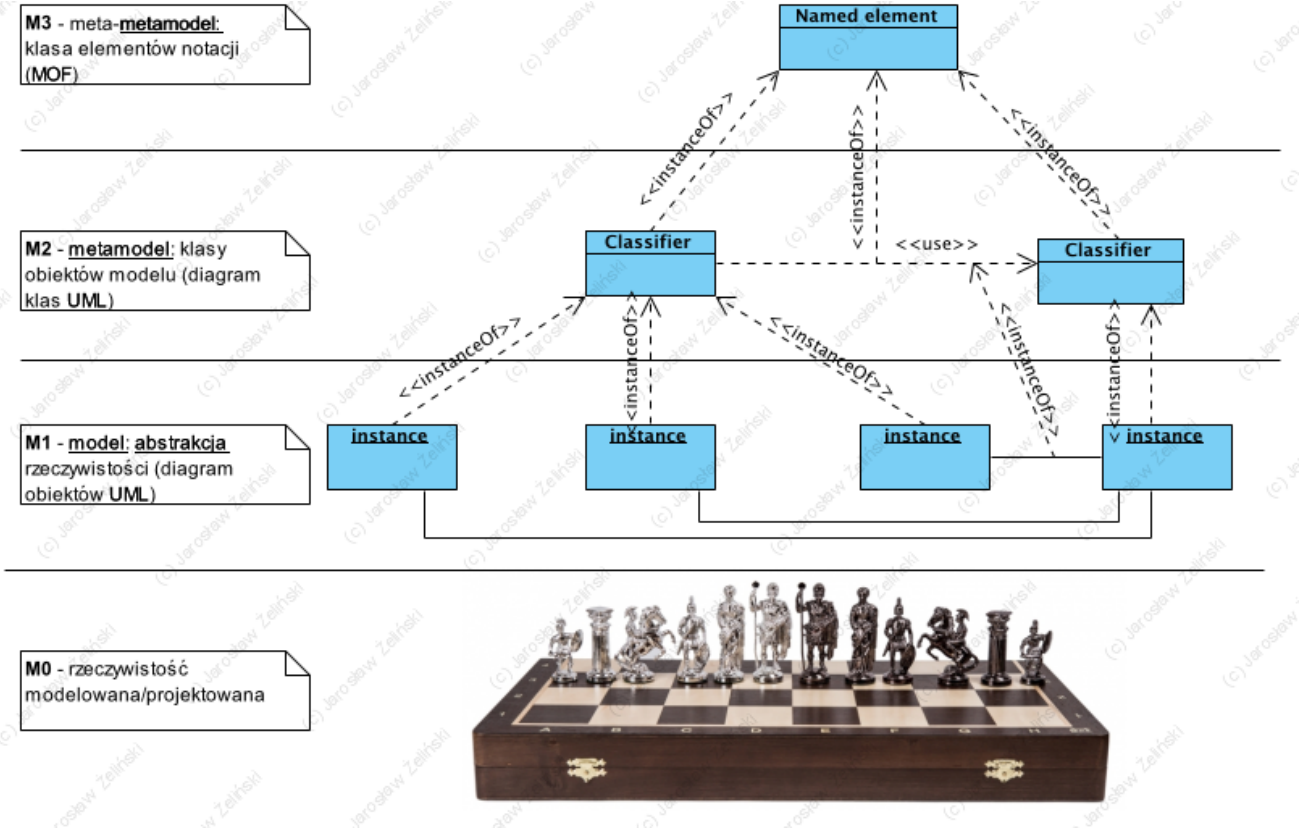
Notacja UML (OMG UML 2017) operuje pojęciami instancji, klasyfikatora i klasy. Po prawej stronie Rys. Semiotyka a UML pokazano odpowiednik trójkąta semiotycznego wyrażony w tych terminach.

Notacja UML operuje także ogólnym pojęciem struktury (structure), jest nią zawartość każdego poprawnego diagramu UML. Struktury mogą wyrażać model pojęciowy (Namespace) lub model (także metamodel) architektury systemu (np. oprogramowania) w postaci schematu blokowego (Architecture).

Modele pojęciowe to struktury oparte na taksonomiach, obrazujące związki pojęciowe (związki pomiędzy nazwami) między nimi. Pojęcie class reprezentuje nazwę (expression) zbioru desygnatów. Pojęcie classifier odnosi się do definicji (meaning) tego pojęcia. W UML definicją klasy obiektów jest zestaw ich wspólnych cech czyli atrybutów i operacji. Pojęcie object odnosi się do desygnatów (thing). W notacji UML, do nazwy obiektu, po dwukropku, dodajemy nazwę klasyfikatora, który go definiuje (klasy do której należy).

W modelach pojęciowych UML wykorzystuje się związki generalizacji (taksonomie) i asocjacje (związki semantyczne między nimi), zaś w modelach architektury: związki kompozycji (związek całość-część), związki użycia (związek zależności od) i związki realizacji (związek między specyfikacją a jej implementacją). Modele architektury wyrażają strukturę i mechanizm działania systemów modelowanych.

2.2. Specyfikacja MOF Poziomy abstrakcji



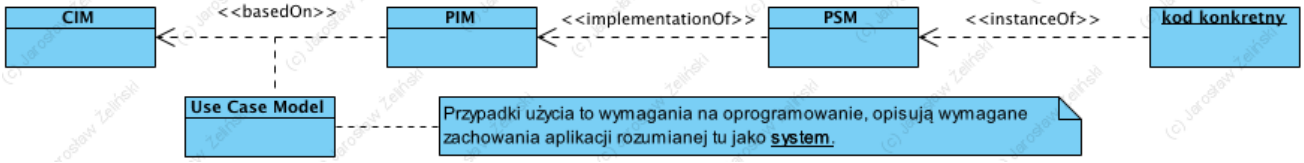
Rys. Specyfikacja MOF Poziomy abstrakcji

Specyfikacja MOF opisuje cztery podstawowe poziomy tworzenia abstrakcji i modeli. Poziom oznaczony M0 to tak zwany realny świat czy to co jest przedmiotem modelowania. Poziom M1 to model będący abstrakcją tego świata, z reguły ujętą z określonej perspektywy. Model, jako abstrakcja, nie jest uproszczeniem. Abstrakcja to obraz określonej rzeczywistości w określonym kontekście, czyli z pominięciem nieistotnych w tym kontekście szczegółów. Poziom M1 to abstrakcje desygnatów pojęć, którymi to pojęciami nazywamy byty świata rzeczywistego.

Poziom M2 operuje klasami (zdefiniowane i nazwane zbiory) obiektów warstwy abstrakcyjnej M1. Jest to metamodel warstwy M1 (metamodel: model zbudowany z klas obiektów warstwy M1). Poziom M3 to warstwa definiująca elementy warstwy M2 jako jedną klasę elementów modeli (jest to meta-metamodel). To opis systemu notacyjnego.

Na poziomach M1 i M2 można tworzyć typy obiektów i typy klasyfikatorów. Warstwy te to tak zwane warstwy pośrednie, a typy klas i obiektów oznacza się w UML stereotypami. Narzędzie to - stereotypowanie - służy do tworzenia tak zwanych profili czyli dziedzinowych grup typów obiektów i klas (tak powstają typy diagramów np. diagram komponentów czy diagram komunikacji). Notacja UML to poziomy M1 i M2 wg. MOF.

2.3. Specyfikacja MDA i model PIM



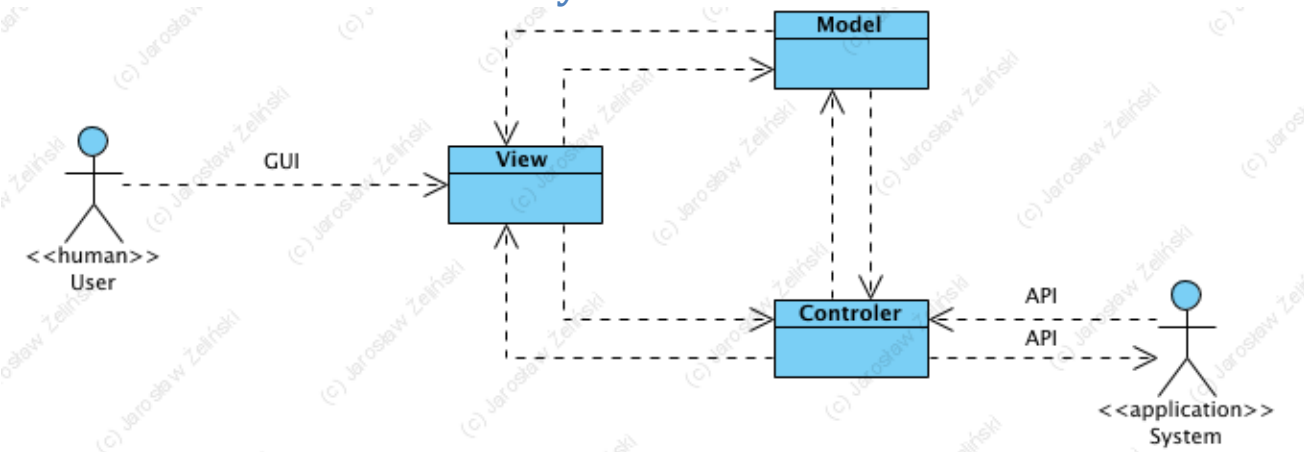
Rys. Specyfikacja MDA i model PIM

Jedną z podstawowych cech MDA jest rozdzielenie kontekstów tworzonych modeli (boundary context, separation of concerns patrz Fowler). Specyfikacja mówi: "MDA oddziela logikę biznesową i aplikacyjną od podstawowej technologii platformy. MDA operuje trzema kontekstowymi typami modeli: CIM (Computation Independent Model), PIM (Platform Independent Model) oraz PSM (Platform Specyfic Model).

Model CIM jest budowany w kontekście działań "świata rzeczywistego i jego reguł biznesowych (patrz MOF poziom M0). Model ten gromadzi opisy struktur informacji (dokumenty) i system pojęć służących do klasyfikowania informacji i działań (namespace). Na podstawie modelu CIM i umowy o zakres (przypadki użycia jako wymagania) powstaje model PIM. Jest on budowany w celu stworzenia opisu mechanizmu działania oprogramowania. Model ten abstrahuje od implementacji. Model przypadków użycia (Use Case Model), jako model pomocniczy (supplementary model, patrz OMG UML 2017), ma na celu określenie roli oprogramowania (usługi aplikacyjne) w modelu biznesowym CIM.

Model PSM to opis implementacji oprogramowania. W stosunku do modelu PIM, zawiera dodatkowe elementy będące konsekwencją tak zwanych wymagań pozafunkcjonalnych (jakość użytkowania, bezpieczeństwo, itp.).

2.4. Wzorzec architektoniczny MVC



Rys. Wzorzec architektoniczny MVC

Wzorzec architektoniczny MVC (ang. Model View Controller, patrz Fowler, Fowler 1997) zakłada podział architektury aplikacji na trzy kluczowe komponenty: Model, który realizuje całość logiki dziedzinowej aplikacji czyli zarządzanie strukturami informacji oraz realizację ich zachowania i przetwarzania,Controler, który realizuje wszelkie techniczne aspekty aplikacji oraz View odpowiadający za obsługę dialogu aplikacji z aktorem, jakim jest tu User. Komunikacja pomiędzy komponentami MVC potencjalnie zachodzi w dwie strony: View przekazuje żądania do komponentu Model ale ten może też prezentować użytkownikowi swój stan niezależnie od tych

wywołań. View może żądać usług od Controler(np. dostępu do usług zewnętrznej aplikacji System). Model także może żądać danych z zewnątrz, itp..

Większość literatury przedmiotu pomija fakt, że integracja aplikacji z innymi odbywa się za pośrednictwem komponentu Controler, który pełni tu rolę separacji komponentu Model od otoczenia aplikacji (inne aplikacje: tu System).

Użytkownik (człowiek) korzysta z aplikacji wywołując jej usługi, ta interakcja jest jednostronna (aplikacja nie wywołuje usług człowieka). Zewnętrzne systemy (Systemj) wywołują usługi aplikacji, zaś aplikacja może korzystać z usług zewnętrznych systemów.

2.5. Wzorzec architektoniczny BCE

Rys. Wzorzec architektoniczny BCE

Wzorzec BCE (Boundary, Control, Entiry, czytaj [BCE]) to wzorzec architektoniczny znany od lat 90-tych. Zakłada podział kodu realizującego usługę aplikacyjną (przypadek użycia systemu) na trzy separowane komponenty: boundary odpowiadający za obsługę dialogu z użytkownikiem, control odpowiadający za realizację logiki biznesowej oraz entity odpowiadający za utrwalanie informacji). Wzorzec obecnie używany także do projektowania tak zwanych mikroserwisów (pojedynczych usług aplikacyjnych).

Stosowanie tego wzorca ułatwia analizę i projektowanie, gdyż narzuca separowanie i hermetyzację tych trzech odpowiedzialności. Jest zgodny z dobrą praktyką projektowania architektury obiektowej OCP (ang. Open Close Principia) mówiącą, że dobra architektura jest otwarta na rozszerzenia i zamknięta na zmiany.

Wzorzec ten, na etapie analizy i projektowania, jest stosowany do projektowania komponentu Model wzorca MVC (opis w dalszej części). Zgodnie z paradygmatem obiektowym wszystkie te komponenty mają operacje (interfejsy).

3. Rezultaty

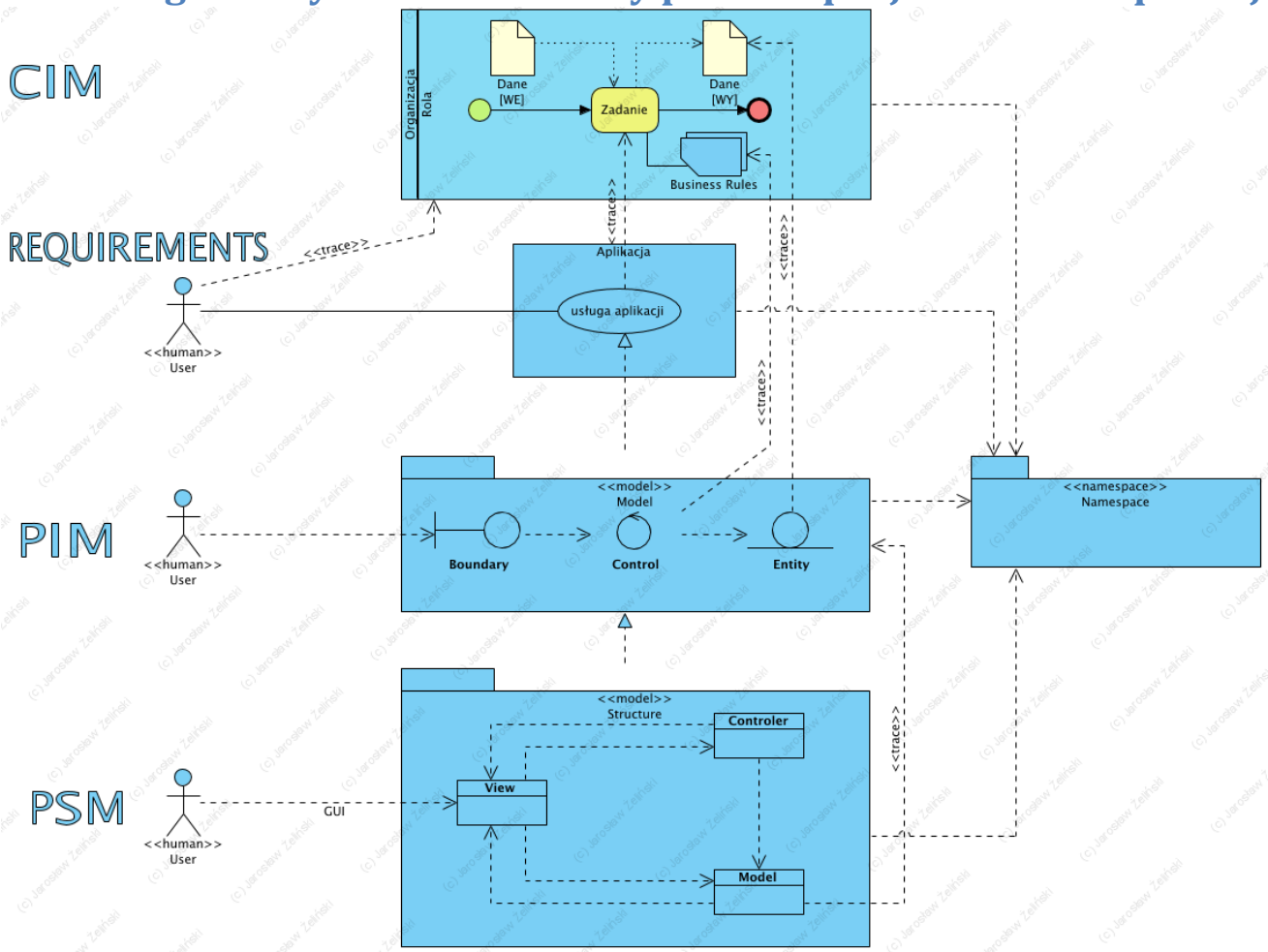
Opisano spójny system pojęciowy oparty na definicjach specyfikacji MOF, MDA (PIM) (patrz OMG.org) oraz na modelach wzorców architektonicznych MVC i BCE. Osiągnięto rezultat pozwalający jednoznacznie określić odpowiedzialność komponentów tych wzorców a także wyznaczyć granice pomiędzy projektowaniem a implementacją. Praca w całości opiera się na podejściu zorientowanym na modelowanie w inżynierii (MDE). Definicje pojęć zebrano na końcu dokumentu.

3.1. Założenie uproszczające

Rys. Założenie uproszczające

Jak wskazano w części Wzorzec architektoniczny MVC tylko komponent Model realizuje logikę dziedzinową, komponenty View i Control są transparentne dla logiki dzedzinowej. Z rozdziału Specyfikacja MDA i model PIM wynika więc, że model PIM zawiera cały i tylko opis realizacji logiki dziedzinowej. Można więc przyjąć bezpieczne założenie upraszczające, zobrazowane na diagramie powyżej: abstrakcja aplikacji na etapie tworzenia modelu PIM, do celów analizy i projektowania realizacji przypadków użycia (usług aplikacyjnych) może składać się wyłącznie z komponentu Model opisującego mechanizm realizowania usług aplikacyjnych (model dziedziny systemu, ang. domain model).

3.2. Zintegrowany model struktury procesu projektowania aplikacji



Rys. Zintegrowany model struktury procesu projektowania aplikacji

Zobrazowanie na jednym schemacie blokowym opisanych modeli wymaga użycia związków <<trace>> do pokazania wywodzenia (śladowanie) elementów kolejnych modeli.

Model CIM to model biznesowy opisujący analizowaną dziedzinę. Jest to model procesów biznesowych (tu notacja BPMN) realizowanych w ramach modelowanego obszaru działania ludzi. Procesy (aktywności tworzące określoną wartości lub produkty zwane biznesowymi) i reguły biznesowe opisują mechanizm działania np. organizacji. Jeżeli organizacja jako tak opisany system współpracujących w określonym celu ludzi, ma zawierać element będący oprogramowaniem (aplikacja), to wymagania na tę aplikację modelujemy jako przypadki użycia, na tym etapie Aplikacja jest tak zwaną czarną skrzynką. Model taki wyrażamy jako przypadki użycia w notacji UML. Istotnym elementem jest tu jeden wspólny biznesowy słownik pojęć reprezentowany

jako Namespace. Wszelkie klasy atrybuty, operacje itp. to nazwy pochodzące (i zdefiniowane w niej) z tej przestrzeni pojęciowej.

Kolejny etap pracy nad projektowaniem oprogramowania to stworzenie modelu tak zwanej białej skrzynki, czyli modelu mechanizmu działania tej usługi aplikacyjnej, opisującego sposób powstania reakcji (efektu) w odpowiedzi na bodźce. Mechanizm ten modelujemy jako model PIM (architektura wyrażona jako diagram klas lub komponentów w notacji UML). Jest to to wyłącznie model mechanizmu realizującego tak zwaną logikę dziedzinową. Na tym etapie abstrahujemy od elementów nie realizujących logiki dziedzinowej. Architektura modelu PIM tu jest budowana w oparciu o opisany już wzorzec architektoniczny BCM. Model ten stanowi komponent Model wzorca MVC.

Model PSM, to plan implementacji z użyciem konkretnej technologii, obrazujący pozostałe elementy wzorca MVC czyli komponentu View realizującego szczegóły komunikacji z użytkownikiem oraz komponentu Controler tego wzorca.

Modele obiektowe (obiektowy paradygmat) nie odnoszą się bezpośrednio do danych jako takich, a do ich nazwanych struktur (odpowiadają one elementom Dane w modelu CIM powyżej). Innymi słowy obiekt entity przechowuje określone informacje ale nie odnosimy ich do atrybutów obiektów. Wartością jednego atrybutu obiektu może być np. nawet bardzo bogata struktura XML zawierająca wiele tak zwanych pól danych (reprezentująca Dane na modelu procesów biznesowych BPMN). Z perspektywy operacji obiektu, nie ma to także znaczenia, bo żądanie jako wywołanie operacji klasy, może dotyczyć całego zestawu danych a nie jednego "pola" (powszechną złą praktyką jest implementowanie pól baz danych jako atrybutów obiektów i budowanie interfejsu dostępu do nich w postaci zestawu operacji set/get dla każdego takiego atrybutu).

Opisany tu system pojęć i elementów architektury, jako całość, stanowi spójną strukturę modeli pozwalającą na śladowanie, czyli wywodzenie kolejnych elementów modeli szczegółowych z ich ogólniejszych poprzedników.

4. Dyskusja

Literatura przedmiotu zawiera opisy tworzenia opisanych modeli, jednak spotkać można rozbieżności w opisach ich architektury i definicji poszczególnych elementów modeli. Najczęściej spotkać można zbyt rozdrobnione modele przypadków użycia oraz tezy mówiące, że komponent entity wzorca BCE to tylko dane (baza danych) zaś komponent control realizuje dziedzinowe reguły biznesowe. Zdaniem autora takie tezy prowadzą to niespójności, których konsekwencją jest brak możliwości jednoznacznego wywodzenia kolejnych elementów modeli począwszy od modeli CIM, przez model PIM do postaci PSM. Efekt tu zaprezentowany jako Zintegrowany model struktury procesu projektowania aplikacji, byłby niemożliwy do uzyskania bez założeń porządkujących pojęcia w obszarze stosowania notacji UML i BPMN, zaprezentowanych w tej pracy. Założenia te są zgodne ze specyfikacjami użytych tu notacji.

5. Dalsze prace

Autor rozwija opisany system integracji modeli obiektowych w kierunku pozwalającym na stworzenie metody tworzenia dziedzinowych (domenowych) metamodeli, jako profili UML w warstwie M2. Celem jest opracowanie medodyki projektowania oprogramowania w sposób iteracyjno-przyrostowy opartej na metamodelowaniu, pozwalającej testować wykonywalność aplikacji jeszcze przed rozpoczęciem jej implementacji.

6. Bibliografia

Nazwa	Opis
BCE	Agile Development with ICONIX Process, People, Process, and Pragmatism, Rosenberg, Don, Collins-Cope, Mark, Stephens, Matt, © APRESS 2005
Berliński, Penc-Pietrzak 2004	Inżynieria Projektowania Strategii Przedsiębiorstwa, Lechosław Berliński, Ilona Penc-Pietrzak, Wydanie: Difin, Warszawa 2004
BERTALANFFY 1969	Bertalanffy L. (1969) General System Theory, wyd. George Braziller Inc.
Beynon-Davis 2004	Inżynieria systemów informacyjnych, Paul Beynon-Davis, Wydanie: Warszawa, WNT 2004
BGH 1998	R. Breu, R. Grosu, F. Huber, B. Rumpe, W. Schwerin. Systems, Views and Models of UML. In: The Unified Modeling Language, Technical Aspects and Applications. Martin Schader, Axel Korthaus (eds.) Physica Verlag, Heidelberg. 1998. www.se-rwth.de/publications
BONDECKA 2015	Bondecka-Krzykowska I. (w Murawski R. 2015, Filozofia matematyki i informatyki), Co to jest komputer, uwagi ontologiczne. wyd. Copernicus Center Press Sp. z o.o. Kraków 2015.
Cempel 2008	Teoria i Inżynieria Systemów, Czesław Cempel, Wydanie: Czesław CEMPEL, Poznań 2008
Dennis 2005	Systems Analysis and Design with UML Version 2.0, Alan Dennis, Barbara Halley Wixom, David Tegarden, Wydanie: Second edition, John Wiley and Sons, Inc. 2005, USA
DIAZ IGI 2014	Advances and Applications in Model-Driven Engineering, Vicente García Díaz (University of Oviedo, Spain), Juan Manuel Cueva Lovelle (University of Oviedo, Spain), B. Cristina Pelayo García-Bustelo (University of Oviedo, Spain) and Oscar Sanjuán Martinez (University of Carlos III, Spain), 2014, ISBN13: 9781466644946|
Fowler	Architektura systemów zarządzania przedsiębiorstwem. Wzorce projektowe, Martin Fowler, Wydanie: Helion Gliwice
Fowler 1997	Analysis Patterns. Reusable Object Models, Martin Fowler, Wydanie: Addison-Wesley, 1997
Graham 2004	Metody obiektowe w teorii i praktyce, Ian Graham, Wydanie: WTN, Warszawa 2004
Hofmeister 2006	Tworzenie architektury oprogramowania, Christine Hofmeister, Robert Nord, Dilip Soni, Wydanie: WNT, Warszawa 2006
KARAGIANNIS 2016	Domain-Specific Conceptual Modeling, Concepts, Methods and Tools, Herausgeber: Karagiannis, Dimitris, Mayr, Heinrich C., Mylopoulos, John (Eds.), wyd. Springer 2016
LOSEE 2001	John Losee, Wprowadzenie do filozofii nauki, wyd. polskie Prószyński i s-ka 2001,
Malinowski	Logika ogólna, Grzegorz Malinowski, Warszawa, PWN 2010.
Martin, Odell 1997	Podstawy metod obiektowych, James Martin, James J.Odell, Wydanie: WNT, Warszawa 1997
MNATI 2018	Gianfranco Minati, Eliano Pessa, Ignazio Licata, General System Theory: Perspectives in Philosophy and Approaches in Complex Systems, MDPI, 9 lip 2018
OMG BPMN 2013	Business Process Model and Notation (BPMN) Version 2.0.2, 2013 , OMG http://www.omg.org/spec/BPMN
OMG MDA	Object Management Group spec. Model Driven Architecture, (https://www.omg.org/mda/)

Nazwa	Opis
OMG MOF 2016	OMG (2016) Meta Object Facility (MOF) Core Specification Version 2.5.1 https://www.omg.org/spec/MOF
OMG SBVR 2017	Semantics of Business Vocabulary and Business Rules Version 1.4 https://www.omg.org/spec/SBVR/
OMG SysML 2017	OMG Systems Modeling LanguageTMVersion 1.5, OMG Document Number: formal/2017-05-01
OMG UML 2017	OMG (2017) Unified Modeling Language® (OMG UML®) Version 2.5.1 https://www.omg.org/spec/UML/
Perdita Stevens 2007	UML. Inżynieria oprogramowania. Wydanie II, Perdita Stevens, Wydanie: Helion 2007
SEARLE 1990	Searle, John R. "Is the Brain a Digital Computer?" Proceedings and Addresses of the American Philosophical Association, vol. 64, no. 3, 1990, pp. 21–37. JSTOR, JSTOR, www.jstor.org/stable/3130074. https://www.jstor.org/stable/3130074
Shalloway 2005	Projektowanie zorientowane obiektowo. Wzorce projektowe, Alan Shalloway, James R. Trott, Wydanie: Gliwice, Helion 2005
Zullighoven 2005	Object-Oriented Construction Handbook, Heinz Zullighoven, Wydanie: Elsevier Inc. 2005

7. Kluczowe pojęcia metodyczne

Nazwa	Opis
abstrakcja	opis z pewnej nazwanej perspektywy lub w pewnym kontekście, polegający na pominięciu nieistotnych w tym kontekście szczegółów, co pozwala na spojrzenie z określonego punktu widzenia i na wyższym poziomie ogólności (źr. OMG spec. MDA)
BCE	analityczny wzorzec architektoniczny, (ang. Boundary, Control, Entity), zakłada modelowanie struktury dowolnej aplikacji z użyciem trzech bloków konstrukcyjnych: Boundary (granica systemu, interfejs), Control (sterowanie, jedyne miejsce realizacji logiki systemu), Entity (trwały byt, miejsce przechowywania informacji, danych, pamięć); w UML są stereotypy klas lub obiektów, wszystkie powinny mieć operacje, modele abstrakcyjne mogę nie mieć atrybutów; wzorzec może być stosowany także do obiektowego modelowania organizacji.
BMM	Business Motivation Model, notacja opublikowana przez organizację Object Management Group (https://www.omg.org/spec/BMM/), dostarcza model pojęciowy do analizy, tworzenia, dokumentowania biznesplanu i metod zarządzania oraz system symboli pozwalający tworzyć diagramy modelujące elementy misji i wizji, strategii i celów biznesowych organizacji, notacja BMM odwołuje do notacji BPMN i SBVR a także do modeli struktur organizacyjnych
BPMN	Business Process Model and Notation, system pojęciowy i graficzna notacja pozwalająca modelować i dokumentować w graficznej formie procesy biznesowe i procedury; pozwala także na modelowanie i dokumentowanie współpracy miedzy organizacjami. (specyfikacja: http://www.omg.org/spec/BPMN/)
CIM	ang. Computation Independent Model, model (perspektywa) niezależny od

Nazwa	Opis
	systemu obliczeniowego, rozumiany jako model abstrahujący od technologii (źr. OMG.org)
elementarny proces biznesowy	zadanie wykonywane przez jedną osobę, w jednym miejscu i określonym czasie, w odpowiedzi na określone zdarzenie biznesowe; zadanie to prowadzi do uzyskania mierzalnej wartości biznesowej; po jego wykonaniu dane są w spójnym stanie; (przypadki użycia powinny realizować elementarne procesy biznesowe, cytat z : UML i wzorce projektowe, Craig Larman)
fakt	to, co zaszło lub zachodzi w rzeczywistości (źr. SJP PWN), w notacji SBVR element łączący pojęcia w jeden kontekst
informacja	wiadomość o czymś lub zakomunikowanie czegoś; dane przetwarzane przez komputer (źr. sł. j. polskiego PWN)
komponent	część składowa, stanowiąca zamknięty konstrukcyjnie element systemu, mogąca być odrębnym przedmiotem dostawy lub realizacji, cechująca się określonym zachowaniem (interfejs), własnym cyklem życia, mogąca być wymieniona na inną, o tych samych cechach zewnętrznych, bez wpływu na pozostałe elementy systemu i system jako całość (na podst, UML, v.2.5.1)
MDA	ang. Model Driven Architecture (architektura zorientowana na modele), architektura oparta na otwartych standardach OMG (Object Management Group), pozwala na odseparowanie w analizie i projektowaniu logiki biznesowej, logiki aplikacji i jej implementacji (źr. www.omg.org/mda)
MDE	Model-driven engineering (MDE), metodologia analizy i projektowania zorientowana na tworzenie modeli danej dziedziny, w tym modeli pojęciowych i innych, budowanych z wielu różnych perspektyw w celu opisania problemu i jego specyfiki; bazuje na stosowaniu abstrakcji i modeli mechanizmów rządzących daną dziedziną zamiast detalicznych wyliczeń i algorytmów.
metamodel	generalizacja modelu, innymi słowy jeżeli model to abstrakcja określonej rzeczywistości, to metamodel jest uogólnieniem określonej klasy (zbioru) takich modeli (abstrakcji), dany metamodel systemu reprezentuje wszystkie systemy zgodne z tym metamodelem, metamodel to klasa modeli, metamodel definiuje semantykę i syntaktykę określonej klasy modeli
model	abstrakcyjna reprezentacja produktu lub systemu, pozwalająca sprawdzić i zweryfikować jego cechy przed rozpoczęciem produkcji, stanowi sobą system założeń, pojęć i zależności między nimi, pozwalający opisać (modelować) określony aspekt rzeczywistości w określonym kontekście; model nie jest uproszczeniem, jest abstrakcją (na podstawie Modele a analizie systemowej, Władysław Findejsen, Jakub Gutenbaum); może zostać wyrażony wzorami matematycznymi lub schematem blokowym (model nominalny, abstrakcyjny), lub jako uproszczona konstrukcja rzeczywista (model realny) (patrz także model Encyklopedia), sł. j. polskiego: konstrukcja, schemat lub opis ukazujący działanie, budowę, cechy, zależności jakiegoś zjawiska lub obiektu;
model biznesowy	Według M. E. Portera model biznesowy jest opisem działalności przedsiębiorstwa, która zapewnia mu zyski. Sprowadza się to do określenia roli organizacji w rynkowym łańcuchu wartości, w jakim działa oraz opisania mechanizmutworzenia tej wartości wewnątrz organizacji. Modele takie wyraża się obecnie w postaci procesów biznesowych i struktur oraz ich dynamiki. Typowe elementy modelu biznesowego: rynkowy nadrzędny

Nazwa	Opis
	łańcuch wartości, model wewnętrznego łańcucha wartości, model rynkowych pięciu sił, także kontekstowa analiza SWOT (źr. M.E.Porter Strategia konkurencji).
moduł	wyodrębniony element architektury rozwiązania (systemu informatycznego) pełniący ustaloną funkcję, łatwy do określenia i wykorzystania jako część większej całości (sł. j.polskiego).
MOF	ang. Meta Object Facility, struktura podstawowych poziomów abstrakcji modeli definiowana przez OMG (OMG.org)
MVC	(model, widok, sterowanie) uznawany za podstawowy, wzorzec architektoniczny opisujący trzy kluczowe komponenty aplikacyjne: model opisujący realizację logiki dziedzinowej, widok opisujący realizację komunikacji z użytkownikiem, sterowanie (kontroler) opisujący realizację sterowania i integracji aplikacji
PIM	(ang. Platform Independent Model) model opisujący logikę działania aplikacji w oderwaniu od technologii implementacji
procedura	wyodrębniony element architektury rozwiązania (systemu informatycznego) pełniący ustaloną funkcję, łatwy do określenia i wykorzystania jako część większej całości (sł. j.polskiego).
proces	przebieg następujących po sobie i powiązanych przyczynowo określonych zmian (źr. s.j.p. PWN)
proces biznesowy	aktywność jako zadanie lub chronologiczny łańcuch zadań, przetwarzająca określony stan wejścia procesu w stan jego wyjścia; wymaga użycia zasobów i może być ograniczony określonymi regułami (opr. własne na podstawie M.E. Porter "On Competition" oraz Rummler, Brache "Improving Business Processes") (patrz także zadanie oraz procedura)
przypadek użycia	reprezentuje określone, inicjowane przez aktora (użytkownika), zachowanie lub zestaw zachowań systemu, którego celem jest z góry określony i przydatny dla aktora rezultat, może zawierać możliwe warianty jego podstawowego zachowania, w tym wyjątkowe zachowanie takie jak obsługa błędów, oznacza konkretne (w celu uzyskania konkretnego efektu) użycie Systemu; kluczowe pojęcia skojarzone z Przypadkami Użycia to Aktor i System (na podstawie UML v.2.5.1., www.omg.org).
PSM	(ang. Platform Specific Model) model opisujący architekturę implementacji aplikacji w kontekście technologii użytej do implementacji
SBVR	Semantics Of Business Vocabulary And Rules, (https://www.omg.org/spec/SBVR/), specyfikacja OMG definiująca metodę tworzenia reguł biznesowych oraz budowanie biznesowego słownika pojęć, specyfikacja zawiera także opis notacji do tworzenia tak zwanych diagramów faktów, są one graficzną reprezentacją biznesowego słownika pojęć i jednocześnie metodą modelowania i kontroli spójności domenowego systemu pojęć (ang. namespace).
SOA	ang. Service Oriented Architecture, pol. architektura (aplikacji) zorientowana na usługi (patrz także usługa aplikacji)
system	dowolny twór pojęciowy lub fizyczny, składając się z części wzajemnie powiązanych i oddziaływających na siebie (na podst Bertalanfyy, Ackof), zbiór elementów i związków między nimi zdolny to realizowania określonych celów (www.omg.org, Model Driven Engineering); SJP: układ elementów mający określoną strukturę i stanowiący logicznie uporządkowaną całość, uporządkowany zbiór twierdzeń, poglądów,

Nazwa	Opis
	tworzących jakąś teorię, log. całościowy i uporządkowany zespół zdań połączonych ze sobą stosunkami logicznego wynikania.

Spi treści

Jarosław Żeliński
independent expert, scientist researcher
j.zelinski@it-consulting.pl
+48 608 05 90 20
I invite you to contact me regarding this publication.