Kamil ŻYŁA

ANALIZA ROZWIĄZAŃ ZWIĄZANYCH Z JĘZYKAMI MODELOWANIA DLA URZĄDZEŃ MOBILNYCH POD KĄTEM INTERAKCJI UŻYTKOWNIKA Z APLIKACJĄ

STRESZCZENIE *Inżynieria sterowana modelami jest obecnie dynamicznie rozwijającą się dziedziną inżynierii oprogramowania, której głównym zadaniem jest zwiększenie abstrakcji oraz uproszczenie procesu wytwarzania oprogramowania. Dynamika jej zmian zaczyna dorównywać popularności urządzeń mobilnych. Celem artykułu jest zidentyfikowanie i podsumowanie osiągnięć inżynierii sterowanej modelami w dziedzinie modelowania aplikacji dla urządzeń mobilnych z punktu widzenia interakcji użytkownika z aplikacją oraz ocena ich przydatności, a także zaproponowanie innowacyjnej graficznej notacji osadzonej w istniejących realiach.*

Słowa kluczowe: *inżynieria sterowana modelami, technologie mobilne, języki modelowania aplikacji mobilnych*

1. WSTĘP

Inżynieria sterowana modelami (MDE) jest aktualnie jedną z dynamicznie rozwijających się dziedzin inżynierii oprogramowania. Badaniami nad nią zajmują się znaczące centra badawcze takie, jak m.in.: Politecnico di Milano, Ecole des Mines de Nantes, Technical University of Vienna. Również firmy w branży wytwarzania oprogramowania spostrzegły potencjał dziedziny, a wśród nich m.in. IBM i MetaCase [2, 6].

mgr inż. Kamil ŻYŁA
e-mail: kamilz@cs.pollub.pl

Politechnika Lubelska, Wydział Elektrotechniki i Informatyki

PRACE INSTYTUTU ELEKTROTECHNIKI, zeszyt 260, 2012

Jednocześnie wzrasta znaczenie urządzeń mobilnych – już od 2009 roku telefon komórkowy jest najczęściej kupowanym urządzeniem elektronicznym na świecie [3]. W dodatku warto zauważyć, że coraz większą część rynku obejmują tzw. smartfony (w 2010 roku stanowiły 3/4 sprzedanych telefonów) i tablety [1]. Ciągła ekspansja tego rynku sprzyja rozwojowi platform programistycznych oraz zapotrzebowaniu na szybkie metody wytwarzania oprogramowania.

Siła obydwu trendów sprawia, że warto zastanowić się nad rozwiązaniami MDE mogącymi znaleźć zastosowanie w procesie wytwarzania aplikacji na urządzenia mobilne.

2. PODSTAWOWE POJĘCIA MDE

Obecnie w ramach inżynierii sterowanej modelami wyróżnia się dwa główne podejścia do procesu wytwarzania oprogramowania: MDA (ang. *Model-Driven Architecture*) – sformalizowane podejście zaproponowane przez OMG (ang. *Object Management Group*) oraz MDSD (ang. *Model-Driven Software Development*) – mniej sformalizowane podejście ukierunkowane na szybką implementację. [2]

Niezależnie od podejścia, oprogramowanie reprezentuje się przy pomocy modeli zbudowanych z wykorzystaniem graficznych bądź tekstowych języków modelowania. Wspomniane języki są zazwyczaj dostosowane do dziedziny aplikacji i są wtedy nazywane językami dziedzinowymi (ang. DSLs – *Domain-Specific Languages*). Model zdefiniowany przy pomocy języka dziedzinowego nazywa się modelem dziedzinowym (ang. DSM – *Domain-Specific Model*) [2].

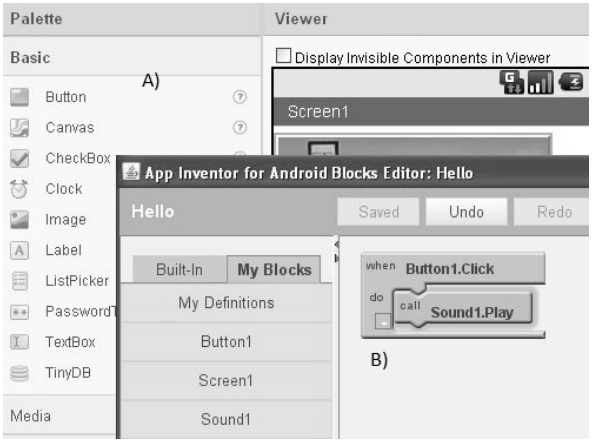
Sformalizowany opis metody tworzenia modeli, będący „modelem modelu", jest nazywany metamodelem i jest zbiorem konceptów (elementów, procesów itp.) związanych z określoną dziedziną. Jeśli model jest abstrakcją świata rzeczywistego, to metamodel jest abstrakcją opisującą właściwości takiego modelu oraz określającą podstawowe jego elementy [4].

Kod wynikowy aplikacji jest generowany na podstawie odpowiednich modeli – w przypadku złożonych systemów informatycznych kod wynikowy może być wypadkową wielu modeli różnego typu. W związku z tym ważne jest utrzymanie spójności modeli z kodem. W tym celu zmiany powinny być wprowadzane w modelu, a nie kodzie aplikacji, a następnie propagowane przez generator [2].

3. GRAFICZNE ROZWIĄZANIA MDE W KONTEKŚCIE MODELOWANIA APLIKACJI MOBILNYCH

App Inventor jest narzędziem modelowania aplikacji mobilnych na platformę Android opracowanym przez firmę Google. Obecnie opieka nad nim została przekazana Massachusetts Institute of Technology. Modelowanie aplikacji przebiega dwuetapowo – opracowanie interfejsu aplikacji mobilnej przy pomocy App Inventor Designer, a następnie reprezentacja akcji związanych z elementami interfejsu przy pomocy App Inventor Blocks Editor [5].

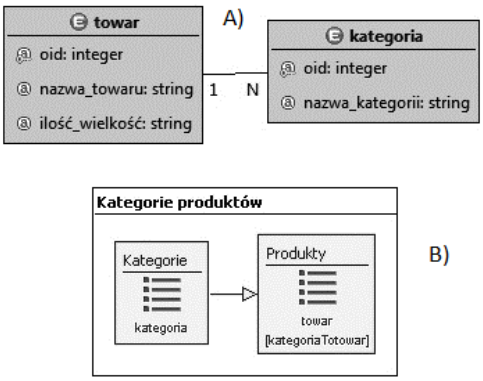
Na rysunku 1 przedstawiono fragmenty modelu interfejsu i logiki biznesowej aplikacji na platformę Android, która w odpowiedzi na kliknięcie przycisku odtwarza plik dźwiękowy. Sekcja A przedstawia edytor WYSIWYG (ang. *What You See Is What You Get*) służący do modelowania interfejsu. Sekcja B przedstawia narzędzie modelowania działania aplikacji, w którym dostępne komponenty symbolizują m.in. metody, zdarzenia, stałe, pętle, itp.



Rys. 1. Widok App Inventor Designer (A) oraz Blocks Editor (B)

WebML (ang. *Web Modeling Language*) jest narzędziem modelowania aplikacji internetowych zarządzających dużymi ilościami danych opracowanym i rozwijanym przede wszystkim przez Politecnico di Milano oraz Web Models. Na proces powstawania aplikacji składają się specyfikacja wymagań funkcjonalnych i niefunkcjonalnych, opracowanie modelu danych oraz opracowanie modelu hipertekstu. Każdy z etapów korzysta z charakterystycznych dla siebie rozwiązań [2].

Na rysunku 2 przedstawiono fragment modelu danych oraz modelu hipertekstu aplikacji zarządzającej sprzedażą. Model danych (sekcja A) jest typowym diagramem związków encji i przedstawia zależność jeden do wielu

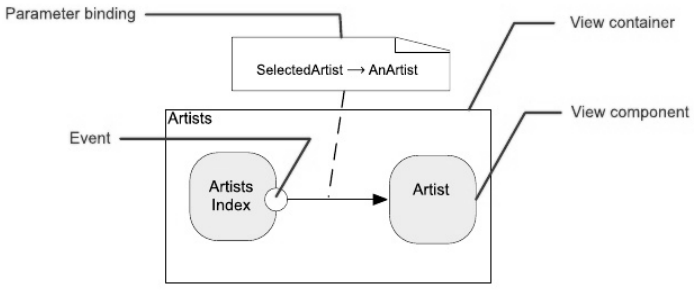


Rys. 2. Przykładowy model danych (A) i hipertekstu (B) [2]

pomiędzy kategorią a towarem. Natomiast model hipertekstu (sekcja B) przedstawia stronę „Kategorie produktów" wyświetlającą kategorie produktów, a po wyborze kategorii przez użytkownika produkty w tej kategorii.

IFML (ang. *Interaction Flow Modeling Language*) jest graficznym językiem dziedzinowym, będącym w trakcie procesu standaryzacji OMG, służącym do uogólnionego opisu interakcji użytkownika z aplikacją. Na potrzeby języka wydzielono dwa główne obszary aplikacji – logikę biznesową i widok użytkownika. IFML obejmuje swoim zakresem: interakcję pomiędzy komponentami widoku użytkownika, interakcję pomiędzy użytkownikiem a komponentami widoku użytkownika, rozmieszczenie komponentów widoku użytkownika i ich powiązania z obiektami biznesowymi [9]. Pomimo swojego uniwersalizmu język IFML ma silne korzenie w języku WebML, służącym do modelowania aplikacji internetowych.

Na rysunku 3 przedstawiono model ekranu aplikacji (kontenera) wyświetlającego informacje o artystach. Jeśli wewnątrz kontenera zajdzie zdarzenie polegające na wybraniu artysty z listy (komponent *Artists Index*), to zostaną wyświetlone szczegółowe dane o wybranym artyście (komponent *Artist*). Pomiędzy komponentami jest przekazywana informacja o wyborze.



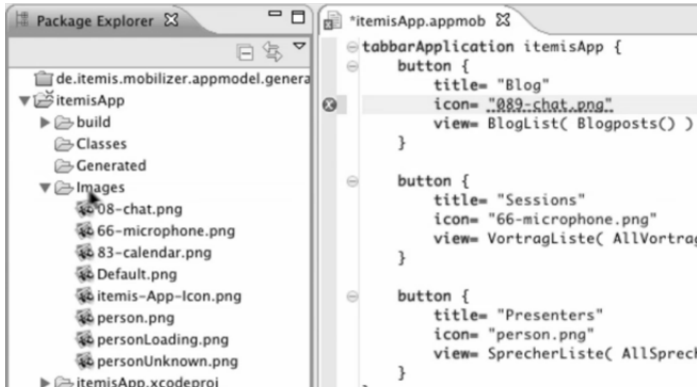
Rys. 3. Przykład modelu w języku IFML [7]

Przedstawione rozwiązania posiadają wady w dziedzinie modelowania aplikacji mobilnych z punktu widzenia ich interakcji z użytkownikiem. AppInventor jest dedykowany platformie mobilnej, ale poza tym stanowi nie-

wielką abstrakcję dla instrukcji języka programowania. WebML z kolei służy do modelowania aplikacji internetowych, więc nie pokrywa zagadnień typowych dla aplikacji mobilnych. Niemniej blisko mu do idei modelowania interakcji z użytkownikiem, a wynikowe aplikacje internetowe mogą zostać dostosowane do platform mobilnych poprzez odpowiednie szablony. IFML jest zaś projektowany pod kątem modelowania interakcji z użytkownikiem i wyrasta z tradycji wielu istniejących rozwiązań, ale poprzez swoją ogólność nie uwzględnia specyficznych potrzeb urządzeń mobilnych.

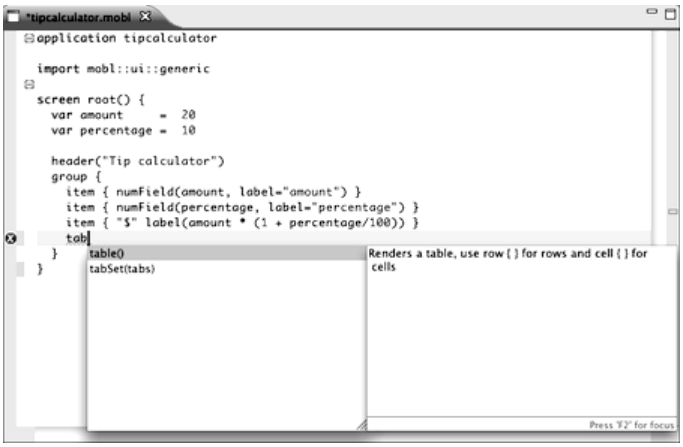
4. TEKSTOWE ROZWIĄZANIA MDE W KONTEKŚCIE MODELOWANIA APLIKACJI MOBILNYCH

Applause jest jednym z tekstowych rozwiązań służących do modelowania aplikacji na platformy mobilne, w tym iPhone, Android i Windows Phone. Wykorzystuje ono język dostosowany do aplikacji mobilnych zarządzających dużą ilością danych, który jest następnie tłumaczony do kodu (czytelnego dla człowieka) w języku Objective-C, Java, C# lub Python [8]. Na rysunku 4 przedstawiono krótki fragment „modelu" aplikacji umożliwiającej wyświetlanie informacji o blogach.



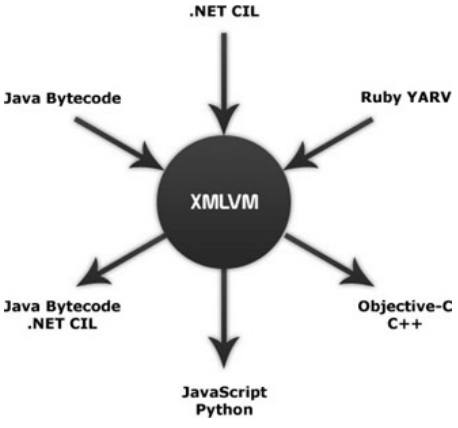
Rys. 4. Fragment „modelu" aplikacji mobilnej wykonany w Applause [8]

Mobl jest otwartym tekstowym językiem modelowania wspomagającym tworzenie aplikacji internetowych, dla urządzeń mobilnych, w oparciu o HTML 5. Powstał w odpowiedzi na problemy z: niedoskonałościami języka JavaScript i języków definiowania interfejsu użytkownika, asynchronicznym API oraz powiązaniem danych z elementami aplikacji [10]. Na rysunku 5 przedstawiono fragment „modelu" aplikacji będącej kalkulatorem napiwków.

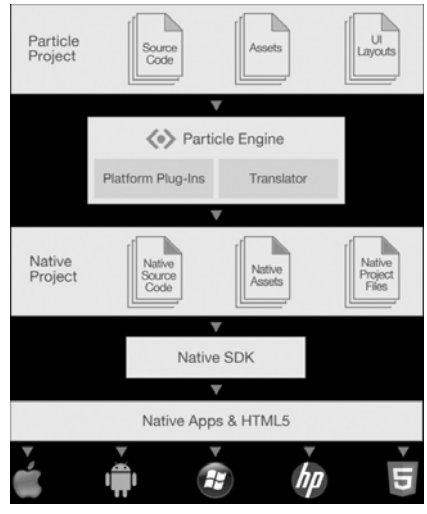


Rys. 5. Fragment „modelu" aplikacji mobilnej wykonanego w mobl [10]

Kolejne podejście do tworzenia aplikacji mobilnych polega na napisaniu kodu w pewnym języku programowania np. Java, a następnie wykorzystanie narzędzi do transformacji tego kodu na kod aplikacji mobilnej w innym języku. Do tej kategorii narzędzi można zaliczyć m.in. XML VM oraz particlecode. Pierwsze z nich dostarcza mechanizmy do przedstawienia programów w postaci dokumentów XML, w celu ułatwienia ich parsowania i translacji do innej postaci (rys. 6) [12]. Drugie z nich umożliwia tworzenie aplikacji mobilnych (natywnych oraz internetowych) w językach obiektowych takich, jak Java i ActionScript3, a następnie dostosowanie ich do różnych platform mobilnych (rys. 7) [11].



Rys. 6. Idea narzędzia XML VM [12]



Rys. 7. Idea narzędzia particlecode [11]

Rozwiązania tekstowe wymagają dużej dyscypliny i mogą być trudne do opanowania. Często uzyskanie modelu o podobnej funkcjonalności przy pomocy graficznej notacji zorientowanej na użytkownika byłoby (w przeciwieństwie do notacji tekstowej) równie mało skomplikowane, jak uzyskanie modelu przedstawionego na rysunku 2.

5. ZAŁOŻENIA AERGIA MODELING LANGUAGE

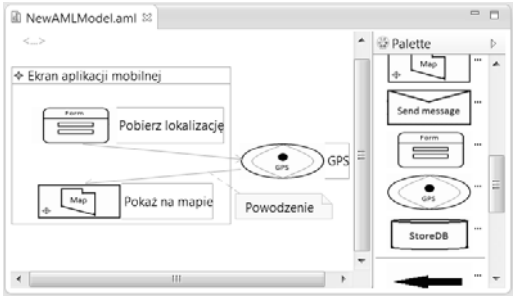
Dostępne rozwiązania MDE są niewystarczające dla potrzeb modelowania aplikacji mobilnych z uwzględnieniem interakcji użytkownika z aplikacją. W związku z tym istnieje potrzeba stworzenia łatwej do nauczenia graficznej notacji, wykorzystującej specyficzne możliwości urządzeń przenośnych, której elementy (komponenty) połączone siecią zależności pozwolą odzwierciedlić istotę aplikacji (zbudować model aplikacji) nawet ludziom z ograniczoną wiedzą programistyczną i znajomością UML (ang. *Unified Modeling Language*).

Powyższe wymagania ma szansę spełnić graficzny język DSL (notacja) o roboczej nazwie Aergia Modeling Language (AML), nad którym pracuje autor niniejszego artykułu. Podstawowe składowe AML to:

1. Komponenty reprezentujące aktywności, które mogą zostać wykonane przez aplikację mobilną, np. odczytanie pozycji GPS, zapisanie do bazy danych, wyświetlenie treści w określony sposób.
2. Połączenia (linki) łączące poszczególne komponenty i wykorzystywane do przekazywania wartości lub tworzenia łańcuchów operacji.
3. Łańcuchy operacji będące zespołami komponentów symbolizujących pewne akcje, których kolejność wykonania jest określona przez połączenia.
4. Kontenery umożliwiające grupowanie elementów notacji.

W związku ze specyfiką urządzeń mobilnych, elementy notacji uwzględniają m.in. następujące obszary: operacje na mobilnych bazach danych, usługi lokalizacyjne, usługi multimedialne, usługi społecznościowe, graficzny interfejs użytkownika, przetwarzanie treści, sensory urządzenia, usługi komunikacyjne (np. bluetooth i wi-fi), itp.

Edytor umożliwiający projektowanie aplikacji z wykorzystaniem przedstawionej notacji jest tworzony jako rozszerzenie środowiska Eclipse, zbudowane w oparciu o EMF (ang. *Eclipse Modeling Framework*) i GMF (ang. *Graphical Modeling Framework*), będące składowymi EMP (ang. *Eclipse Modeling Project*).



Rys. 8. Przykład modelu w języku AML

Na rysunku 8 przedstawiono przykładowy model aplikacji mobilnej w języku AML wykonany w edytorze tworzonym przez autora niniejszego artykułu, w oparciu o środowisko Eclipse. Aplikacja na żądanie użytkownika pobiera informacje o jego bieżącym położeniu, a następnie (jeśli udało się je uzyskać) przedstawia je na mapie.

6. WNIOSKI

Na podstawie analizy istniejących rozwiązań MDE pod kątem modelowania aplikacji dla urządzeń mobilnych można stwierdzić, że szczególnie obiecujący jest język AML tworzony przez autora niniejszego artykułu.

W przeciwieństwie do AML, istniejące rozwiązania nie łączą w sobie dostosowania do specyfiki urządzeń mobilnych oraz prostoty użytkowania. Część z nich jest trudna w odbiorze dla projektanta lub wymaga długiej nauki, część jest jedynie graficzną abstrakcją języka programowania, więc nie spełnia wymogu projektowania z punktu widzenia interakcji z użytkownikiem. Kolejne zaś spełniają to wymaganie, ale nie są dedykowane dla urządzeń mobilnych.

LITERATURA

1. Hatalska N.: Penetracja smartfonów w Polsce – dane za 2011. http://hatalska.com/ /2012/02/13/penetracja-smartfonow-w-polsce-dane-za-2011/ (stan na 12.04.2012 r.).
2. Kęsik J., Żyła K.: Współczesne Technologie Informatyczne – Technologie MDE w projektowaniu aplikacji internetowych. Wydawnictwo Politechniki Lubelskiej, Lublin 2011.
3. Konecki T.: Rzeczywistość rozszerzona – lepsza wersja świata. http://nt.interia.pl/news/ /rzeczywistosc-rozszerzona-lepsza-wersja-swiata,1651714 (stan na 12.04.2012 r.).
4. Stahl T., Völter M.: Model-Driven Software Development: Technology, Engineering. Management. Wiley 2006.
5. Żyła K.: Wykorzystanie mechanizmów lokalizacji urządzenia mobilnego w oparciu o Google App Inventor. Logistyka 3/2012, Instytut Logistyki i Magazynowania, Poznań 2012.
6. Żyła K., Kęsik J.: Podsumowanie i kierunki badań nad MDE na Politechnice Lubelskiej. Kompetentny absolwent informatyki 2012, Polskie Towarzystwo Informatyczne, Lublin 2012.

7. Brambilla M.: Experiences and requirements for a User Interaction Modeling Language, 28.03.2012.
8. Strona domowa narzędzia applause, github.com/applause (stan na 10.06.2012 r.).
9. Object Management Group: Interaction Flow Modeling Language (IFML) RFP. OMG Document: ad/11-12-06.
10. Strona domowa języka mobl, www.mobl-lang.org (stan na 10.06.2012 r.).
11. Strona domowa narzędzia particlecode, www.particlecode.com (stan na 14.06.2012 r.).
12. Strona domowa narzędzia XML VM, xmlvm.org/overview (stan na 14.06.2012 r.).

Rękopis dostarczono dnia 29.08.2012 r.

ANALYSIS OF MODELING LANGUAGES DEDICATED FOR MOBILE DEVICES FROM THE USER INTERACTION POINT OF VIEW

Kamil ŻYŁA

ABSTRACT *Nowadays Model-Driven Engineering (MDE) is dynamically evolving domain of software engineering. Its main goal is to improve abstraction level and simplify the process of software development. In parallel the role of mobile platforms and need for dedicated applications increased significantly. The main goal of this paper is to identify and summarize achievements of MDE in the field of modeling mobile applications, analyze their usefulness and propose innovative graphical notation fulfilling the technology gap.*

Keywords: *Model-Driven Engineering, mobile technologies, DSLs for modeling mobile applications*

Mgr inż. Kamil ŻYŁA – asystent w Instytucie Informatyki Politechniki Lubelskiej. Zajmuje się inżynierią sterowaną modelami, technologiami mobilnymi oraz bazami danych. Uczestnik i współautor projektu „MDE Expertise – Exchanging knowledge, techniques and experiences around Model Driven Engineering education".