AUTOMATYKA • 2005 • Tom 9 • Zeszyt 1–2

Andrzej Kononowicz*, Małgorzata Żabińska**

Dystrybucja obiektów dydaktycznych bazująca na technologii agentowej

1. Wprowadzenie

Pod pojęciem e-learning rozumie się wykorzystanie nowych, specyficznych technologii informatycznych w nauczaniu. Kierunek ten przeżywa obecnie swój wielki rozkwit. W dzisiejszej rzeczywistości wiedza stanowi cenny kapitał. Nieustanny postęp techniczny dezaktualizuje naszą wiedzę, która musi być nieustannie uzupełniana. E-learning jest odpowiedzią na coraz większe zapotrzebowanie edukacyjne społeczeństwa. Z oferty serwisów e-learning mogą korzystać studenci tradycyjnych uniwersytetów, traktując go jako formę uzupełniającą do tradycyjnych kursów. Zalety e-nauczania odkryły również prywatne firmy, które zyskały możliwość szybkiego szkolenia pracowników, bez narażania się na koszty wynikające z eksternistycznych kursów. Dzięki systemom e-learning pracownik może uzupełniać swoją wiedzę w dowolnym czasie, zakresie i nie opuszczając przy tym miejsca pracy. E-learning stanowi często jedyną szansę na dokształcenie dla osób niepełnosprawnych lub tych, którym wiek, zamieszkanie, sytuacja rodzinna lub zawodowa nie pozwala podjąć tradycyjnych studiów.

Jedną z największych zalet e-learning jest możliwość personalizacji procesu kształcenia, która pozwala uczniom o różnych uzdolnieniach, zainteresowaniach i posiadanej wiedzy, dostosowywać tryb nauki do własnych potrzeb [18]. Nietrywialnym zadaniem jest techniczna realizacja tych założeń. Pomysłem realizacji zadania tego typu są „obiekty dydaktyczne" (*learning objects*) [9, 22]. Treść przedmiotu nauczania dzielona jest na mniejsze, spójne pod względem merytorycznym części, które mogą być – podobnie jak obiekty w programowaniu – odpowiednio dobierane i łączone w większe całości. Obiekty dydaktyczne, gromadzone w specjalnych bazach danych, mogą być udostępniane uczącym się poprzez Internet.

Niemałym wyzwaniem jest opracowanie techniki skutecznego wyszukiwania przez studentów interesujących ich obiektów dydaktycznych. Tradycyjne wyszukiwarki internetowe, działające jedynie na poziomie syntaktycznym, nie zapewniają dostatecznie wysokiej jakości dostarczanych wyników kwerend użytkownika. W celu zwiększenia skuteczności

* Zakład Bioinformatyki i Telemedycyny, Collegium Medicum, Uniwersytet Jagielloński, Kraków
** Katedra Informatyki, Akademia Górniczo-Hutnicza w Krakowie

zaczęto stosować komputerowe opisy semantyki publikowanych materiałów (poprzez metadane). Jednym ze sposobów opisu znaczeń jest tworzenie sieci semantycznych. Sieć semantyczna jest to graf skierowany, którego poszczególne węzły stanowią pojęcia, a krawędzie oznaczają relację między pojęciami.

W ostatnich latach wielki rozgłos zyskał zapoczątkowany przez Tima Berners-Lee projekt Semantic Web [2]. Projekt ten ma na celu stworzenie, w ramach obecnej sieci Internet, globalnej sieci semantycznej. Publikowane w sieci Internet materiały, których znaczenie zostało opisane w zrozumiały dla systemów komputerowych sposób (za pomocą technik projektu Semantic Web, takich jak RDF [10, 16, 29], OWL [28]), stanowi wymarzone pole działania mobilnych agentów programowych. Agenci programowi są autonomicznymi systemami komputerowymi, które postrzegając i oddziałując ze swoim otoczeniem, dążą do wykonania zleconych im zadań [6, 11]. Mobilni agenci są bytami mogącymi przemieszczać się pomiędzy rozproszonymi systemami komputerowymi. Dzięki projektowi Semantic Web agenci zyskają dostęp do olbrzymiej bazy wiedzy, w której będą mogli dużo skuteczniej niż dotychczas wyszukiwać informacje, wykorzystując techniki sztucznej inteligencji.

2. Agenci programowi w e-learning

Celem agentów programowych w e-learning jest wspomaganie uczących się w procesie ich kształcenia. Pierwotnie agenci w e-learning wykorzystywani byli jako element inteligentnego interfejsu użytkownika[1)]. Agenci tego typu nazwani zostali agentami peda-gogicznymi [1]. Agenci pedagogiczni posiadają niekiedy antropomorficzną postać graficzną [22]. Zadaniem ich jest obserwacja postępów w nauce studenta i motywowanie go/jej do dalszej pracy poprzez udzielanie zindywidualizowanych rad dotyczących przedmiotu nauczania, technik uczenia się oraz sposobu obsługi programu edukacyjnego. Bardziej zaawansowani agenci pedagogiczni opracowują dla swoich studentów indywidualnie dopasowane programy nauczania.

Technologia agentowa w e-learning może być wykorzystana również jako warstwa pośrednicząca w komunikacji między kształcącymi się a magazynami obiektów dydaktycznych, dostępnymi w sieci Internet lub w lokalnych sieciach akademickich. Agenci programowi, dzięki samodzielnym obserwacjom zachowań swych zleceniodawców, a także poprzez gromadzenie i przechowywanie ich preferencji, mogą usprawniać proces wyszukiwania materiałów dydaktycznych.

Obecnie podejmowane są próby stworzenia ogólnej architektury systemów e-learning wykorzystujących technologie agentowe [4, 5, 7, 19, 20, 21]. Prace te koncentrują się zwykle na stworzeniu ontologii mogących być wykorzystane przez agentów w e-learning. Trwają również prace nad definicją różnych typów agentów mogących mieć zastosowanie w edukacji. Celem tworzenia tego rodzaju modeli jest umożliwienie w przyszłości prostej współpracy systemów tej klasy. Niniejszy artykuł dotyczy drugiego z wymienionych zagadnień.

[1)] Rozgraniczenie między inteligentnym interfejsem użytkownika a agentem programowym bywa niekiedy trudne i stanowi źródło nieporozumień.

Chen i Mizogutchi przedstawili w swojej pracy [5] szkic wieloagentowej architektury dla inteligentnych systemów edukacyjnych, w której występuje pięć typów agentów. Agenci ci to *Learning Material Agent*, *Learning Support Agent*, *Learner Modeling Agent*, *Learner Model Agent* oraz *Interface Agent*.

Learning Material Agent odpowiedzialny jest za wiedzę z dziedziny nauczania. Agent ten posiada informację na temat zawartości materiałów dydaktycznych i ich struktury. Funkcją agenta typu *Learning Support Agent* (zwanego też *Pedagogical Agent*) jest planowanie procesu dydaktycznego dla osoby uczącej się. Agent ten kontaktuje się z agentem typu *Learning Material Agent* w celu pobrania informacji na temat dziedziny nauczania, a następnie na podstawie zgromadzonych danych planuje ćwiczenia dla osoby uczącej się. W razie wystąpienia trudności w zrozumieniu materiału stara się pomóc uczącemu się przez udzielanie dalszych wskazówek.

Celem *Learner Modeling Agent* jest stworzenie modelu osoby uczącej się na podstawie jej interakcji z systemem. W przypadku wystąpienia trudności z tworzeniem modelu, agent ten może zwrócić się do agenta typu *Learning Support Agent* w celu postawienia uczącemu się dodatkowych pytań, które wyjaśnią wątpliwości. *Learner Model Agent* komunikuje się z zewnętrznymi agentami w celu wymiany informacji o uczących się. *Interface Agent* odpowiedzialny jest za komunikację między systemem a osobą uczącą się. Agent ten obserwuje zachowanie uczącego się. Zebrane informacje przekazywane są agentom typów *Learner Modeling Agent* i *Learning Support Agent*.

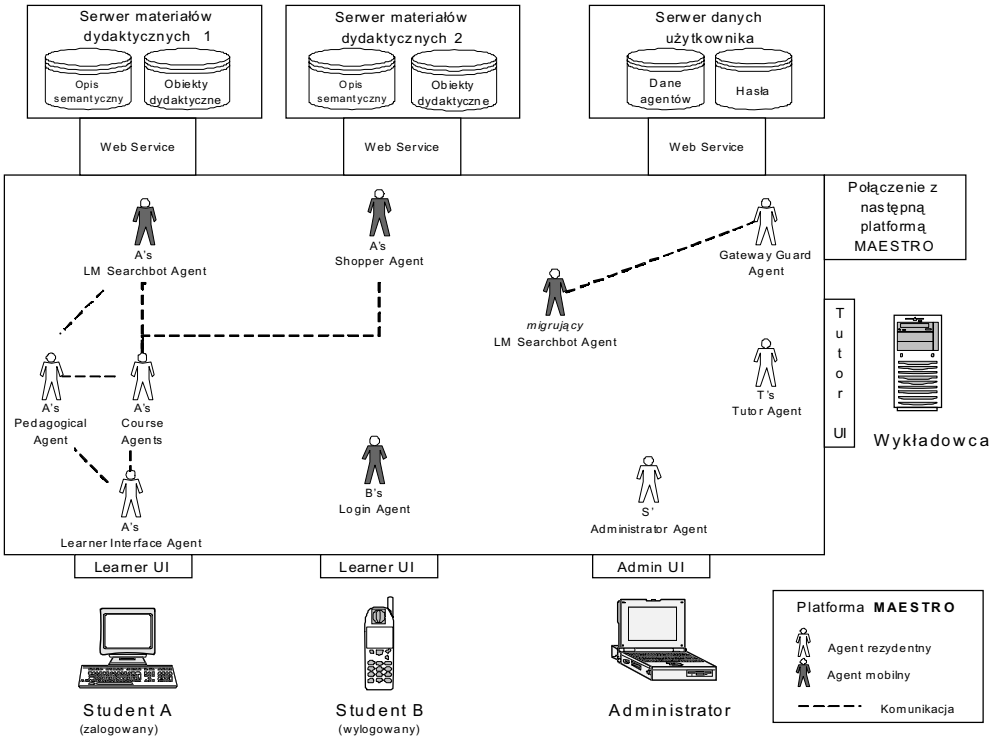
3. Architektura MAESTRO

Zaproponowana w ramach pracy [13] architektura MAESTRO modyfikuje i rozszerza model przedstawiony przez Chen i Mizogutchi w [5]. Architektura MAESTRO obejmuje nie tylko zagadnienia podziału zadań pomiędzy agentów, ale również środowisko, w którym przyjdzie im działać. Problem ten nie był uwzględniony w wymienionej pracy autorstwa Chen, Mizogutchi i dlatego ponownie przeanalizowaliśmy podział zadań agentów w systemie e-learning. Wnioski z tych rozważań zaprezentowano w następnym rozdziale. Natomiast w niniejszej pracy, w odróżnieniu od artykułu [5] nie rozważono zagadnienia opracowania wspólnej ontologii dla systemów e-learning. Opieramy się tutaj w znacznym stopniu na powszechnie uznanych zbiorach metadanych takich jak LOM [15, 23, 15]oraz standardach definicji metadanych takich jak RDF [30].

Środowisko pracy agentów w architekturze MAESTRO składa się z szeregu wzajemnie połączonych platform. Platformy MAESTRO stanowią rozszerzoną wersję klasycznej platformy agentowej. Nowymi elementami platformy są serwery materiałów dydaktycznych oraz serwery danych użytkownika co przedstawiono na schemacie architektury MAESTRO (rys. 1), natomiast charakterystyka zaproponowanych typów agentów została zaprezentowana w rozdziale 4.

Serwery materiałów dydaktycznych służą do przechowywania obiektów dydaktycznych. Obiektami dydaktycznymi mogą być np. strony internetowe w formacie HTML lub XML, aplety Javy, animacje Macromedia Flash, interaktywne gry, sekwencje audio lub wi-

deo. Obiekty dydaktyczne pogrupowane są w moduły. Moduły dydaktyczne odpowiadają jednostkom nauczania znanym z tradycyjnych form edukacji: np. statystyka, biologia molekularna czy sieci komputerowe. Każdy obiekt dydaktyczny posiada pewne właściwości opisane za pomocą metadanych.



Rys. 1. Architektura MAESTRO

Opis modułu stanowi sieć semantyczna zawierająca trzy rodzaje węzłów:

1) węzły pojęć,
2) węzły obiektów,
3) węzły struktur.

Węzły połączone są szeregiem relacji. Węzły pojęć przedstawiają terminy z dziedziny tematyki modułu (np. mediana, średnia arytmetyczna – dla modułu statystyka, aminokwas – dla modułu biologia molekularna czy protokół TCP/IP – dla modułu sieci komputerowe). Liczba możliwych typów relacji została celowo zawężona, by uprosić tworzenie modeli, jak również ułatwić tworzenie algorytmów wnioskowania agentów programowych. Pojęcia powiązane są między sobą relacjami *subTerm* (relacja dziedziczenia), *sameTerm* (relacja bycia synonimem), *requiresKnowing* (relacja asocjacji, jedno pojęcie wymaga do zrozumienia

znajomości innego pojęcia), *logicalPartOf* (relacja agregacji). Tworzone semantyczne modele modułów nie mają aspiracji pełnego odwzorowanie logiki świata rzeczywistego, gdyż jest to proces niezwykle złożony. Celem tworzenia sieci węzłów pojęć jest jedynie zaprezentowanie logiki procesu uczenia się opisywanego modułu dydaktycznego. Węzły obiektów odpowiadają obiektom dydaktycznym z danego modułu. Węzły tego typu opisane są za pomocą metadanych w standardzie LOM [15, 22]. Relacja *hasLearningMaterial* opisuje powiązania pomiędzy węzłami pojęć i obiektów dydaktycznych. Węzły struktur porządkują węzły obiektów dydaktycznych nadając im hierarchiczną strukturę (np. rozdział, podrozdział, paragraf) – relacja *contains*, jak również porządkując je liniowo – relacja *next*.

Zakładamy, iż opis obiektów dydaktycznych wykonany będzie w technologii RDF (*Resource Description Framework*) [10, 29], który jest specyfikacją tworzenia metadanych rekomendowaną przez W3C [30]. Sposób implementacji serwerów materiałów dydaktycznych może być różny, ważne jest jednak zachowanie wspólnego interfejsu dostępu do serwerów materiałów dydaktycznych, z którego mogliby korzystać agenci. Zasadne wydaje się tutaj wykorzystanie technologii Web Services [31, 32].

Dostęp do modułów dydaktycznych może być ograniczony do pewnej wybranej grupy osób. Przewiduje się możliwość odpłatnego korzystania z modułów dydaktycznych. Ciekawym, jednakże jeszcze nierozwiązanym zagadnieniem jest kwestia bezpieczeństwa transakcji przeprowadzanych przez agentów.

Serwery danych użytkownika zawierają informacje zgromadzone na temat użytkowników platformy.

Rozróżniono trzy klasy użytkowników:

1) uczniowie,
2) nauczyciele,
3) administratorzy.

Na serwerach danych użytkownika przechowywane są hasła użytkowników oraz ich dane personalne. W dotychczasowych pracach nad architekturą skupiono się przede wszystkim na obsłudze osób uczących się. Serwery danych zawierają więc informację dotyczące preferowanego przez uczniów stylu nauczania, dotychczasowych postępów w nauce, jak również subiektywne spostrzeżenia na temat uczących się dokonywane przez agentów (np. uczeń woli praktyczne przykłady od teoretycznych rozważań, uczeń jest systematyczny, uczeń sporadycznie korzysta z systemu i łatwo się niecierpliwi).

Platformy MAESTRO mogą być wzajemnie łączone, co zapewnienia większą różnorodność dostępnych materiałów dydaktycznych. Mobilni agenci mogą przemieszczać się pomiędzy platformami w poszukiwaniu nowych źródeł obiektów dydaktycznych. Niezwykle cenną właściwością tego rozwiązania jest decentralizacja sposobu zarządzania obiektami dydaktycznymi. Pozwala to w niezależny sposób tworzyć bazy obiektów dydaktycznych, a następnie zespalać zasoby poprzez tworzenie połączeń pomiędzy platformami.

Kluczową kwestią wymagającą dalszych rozważań jest zapewnienie bezpieczeństwa platformom MAESTRO. Należy uwzględnić możliwość wystąpienia ataków typu „odmowa obsługi" (*Denial of Service*) mogących sparaliżować działanie platformy, jak również agentów podszywających się pod osoby uprawnione do korzystania z zasobów edukacyjnych, do których istnieje ograniczony dostęp.

4. Agenci programowi w architekturze MAESTRO

W zaproponowanej architekturze MAESTRO wyróżniono dziesięć klas agentów:

1) *Interface Agent*,
2) *Login Agent*,
3) *Course Agent*,
4) *Pedagogical Agent*,
5) *Learning Material Searchbot*,
6) *Learner Interface Agent*,
7) *Shopper Agent*,
8) *Tutor Agent*,
9) *Administrator Agent*,
10) *Gateway Guard Agent*.

W niniejszym rozdziale zostaną scharakteryzowane pokrótce cele tych agentów, ich cykl życia, jak również zostanie porównane proponowane rozwiązanie z koncepcją przedstawioną w pracy Chen i Mizogutchi [5]. Podczas analizy podziału zadań agentów w inteligentnych systemach edukacyjnych przedstawionego w pracy Chen i Mizogutchi [5] nasuwają się wątpliwości, co do zasadności istnienia agenta typu *Learner Model Agent*, którego zadaniem jest komunikacja z zewnętrznymi agentami w celu wymiany informacji na temat uczącego się. Funkcję tę może pełnić *Learner Modeling Agent*, którego zadaniem jest tworzenie modelu osoby uczącej się. Model ten wykorzystywany jest jak się wydaje, wyłącznie do ustalenia strategii nauczania. Stąd propozycja, by połączyć tego agenta z agentem typu *Learning Support Agent*. Powstałego w ten sposób agenta nazwaliśmy Pedagogical Agent.

Interface Agent jest agentem integrującym się z aplikacją kliencką. Odpowiada on agentowi o takiej samej nazwie z architektury [5]. Jego zadaniem jest pośredniczenie w komunikacji między użytkownikiem i jego agentami. Tworzony jest w momencie inicjacji aplikacji klienckiej; kończy działanie w momencie zamknięcia aplikacji. *Learner Interface Agent* jest podklasą agenta *Interface Agent*, wyspecjalizowaną do obsługi interfejsu przeznaczonego dla uczniów.

Login Agent obsługuje proces weryfikacji tożsamości użytkownika platformy. Do jego zadań należy przyjęcie od użytkownika hasła (lub klucza), wyszukanie odpowiedniego serwera danych użytkowników oraz poinformowanie użytkownika o wyniku procesu weryfikacji tożsamości. *Login Agent* tworzony jest w momencie podania przez użytkownika hasła i usuwany jest po zakończeniu procesu logowania (niezależnie od jego wyniku). Pozytywne przejście procesu logowania powoduje odtworzenie przyporządkowanego danemu uczniowi agenta typu *Pedagogical Agent*. Agent ten, jak już wspomniano, ma za zadanie dostosowywanie sposobu kształcenia do preferencji użytkownika.

Pedagogical Agent tworzony jest w momencie pierwszego zalogowania się użytkownika w systemie. Po każdym wylogowaniu się użytkownika zebrane przez niego dane zapisywane są w serwerze danych użytkownika. Po ponownym zalogowaniu się użytkownika następuje rekonstrukcja agenta na podstawie danych z serwera danych użytkowników.

Course Agent odpowiedzialny jest za zarządzanie w imieniu użytkownika danymi dotyczącymi wybranego modułu dydaktycznego. Odpowiada on agentowi typu *Learning*

Material Agent z pracy [5], z tą różnicą, iż jego działanie zostało przez nas określone, w znacznie większym stopniu. Agent ten nie posiada całej wiedzy z danej dziedziny, jedynie jej fragment, związany z przeglądanym przez użytkownika obiektem dydaktycznym. W odpowiedzi na zapytania użytkownika, *Course Agent* tworzy szereg agentów typu *Learning Material Searchbot*, które poszukują w serwerach obiektów dydaktycznych informacji, które mogą być przydatne do wyjaśnienia użytkownikowi poszukiwanego pojęcia. Wyniki wyszukiwań agentów typu *Learning Material Searchbot* zbierane są przez agenta typu *Course Agent*, a gotowa lista wyników przekazywana jest agentowi typu *Pedagogical Agent* w celu uporządkowania jej według preferencji użytkownika.

Agent typu Course Agent tworzony jest przy pierwszym połączeniu się użytkownika z nowym modułem dydaktycznym. Agent usuwany jest w chwili zamknięcia przez użytkownika modułu. Po ponownym otwarciu modułu dane agenta typu *Course Agent*, zapisane dotychczas na serwerze danych użytkownika, służą jego rekonstrukcji. Uporządkowana pod względem preferencji użytkownika lista wyników zapytania wysyłana jest agentowi typu *Interface Agent*. Użytkownik wybiera jeden z zaproponowanych mu materiałów. Żądany obiekt pobierany jest z serwera obiektów dydaktycznych przy pomocy agenta typu *Shopper Agent*. W przypadku modułów komercyjnych, na tym etapie następuje transakcja zakupu obiektu.

Shopper Agent tworzony jest w momencie wydania przez użytkownika dyspozycji pobrania materiału dydaktycznego, a ulega likwidacji w chwili przekazania obiektu dydaktycznego agentowi typu *Interface Agent*.

Agent typu *Tutor Agent* zbiera dla nauczyciela informacje dotyczące aktywności i postępów w nauce podległych nauczycielowi uczniów. Tworzony jest w momencie rejestracji nauczyciela w platformie, a usuwany jest w chwili likwidacji konta nauczyciela.

Administrator Agent monitoruje stan platformy i informuje administratora o zauważonych zagrożeniach lub przypadkach łamania regulaminu platformy. Agent tego typu tworzony jest w momencie rejestracji administratora w platformie, usuwany jest w chwili likwidacji konta administratorskiego.

Gateway Guard Agent ma za zadanie decydowanie o przyjęciu lub odmowie przyjęcia agentów zamierzających migrować na zarządzaną przez nich platformę MAESTRO. Tworzony jest w chwili utworzenia platformy MAESTRO i trwa aż do jej likwidacji. Prace nad agentami ostatnich trzech typów – *Tutor Agent*, *Administrator Agent*, Gateway *Guard Agent* – są na bardzo wczesnym etapie koncepcyjnym i wymagają dalszej analizy.

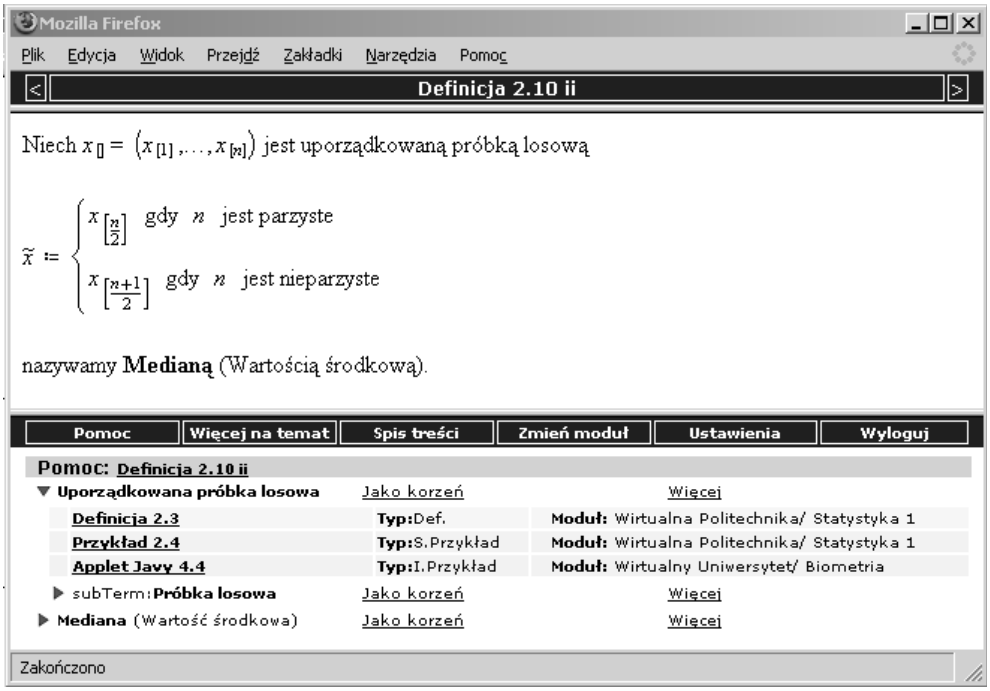
5. Prototyp systemu

W ramach pracy [13] zaprojektowano i częściowo zaimplementowano prototyp aplikacji przeznaczonej dla uczących się. Prototyp ten został zrealizowany jako aplikacja sieciowa w technologii JSP. Na rysunku 2 zaprezentowano fragment interfejsu użytkownika.

Graficzny interfejs użytkownika składa się z czterech części:

1) paska tytułowego,
2) obszaru obiektu dydaktycznego,
3) paska menu,
4) obszaru opcji menu.

Pasek tytułowy zawiera nazwę aktualnie otwartego obiektu dydaktycznego oraz dwa przyciski „>" i „<" prowadzące do poprzedzającego i następującego obiektu dydaktycznego (relacja *previous* i *next*). Obszar obiektu dydaktycznego zawiera treść aktualnie wybranego obiektu dydaktycznego. W przykładzie przedstawionym na rysunku 2 pokazany jest bardzo prosty obiekt dydaktyczny z modułu do nauki statystyki. Obiekt zrealizowany jest jako strona HTML ze wzorami matematycznymi zapisanymi w formacie MathML. Obszar opcji menu zawiera pola zależne od wybranej aktualnie opcji.



Rys. 2. Prototyp graficznego interfejsu w architekturze MAESTRO, dla uczącego się

Pasek menu posiada sześć opcji:

1) *Pomoc,*
2) *Więcej na temat,*
3) *Spis treści,*
4) *Zmień moduł,*
5) *Ustawienia,*
6) *Wyloguj*.

Po wybraniu opcji *Pomoc* (rys. 2) wyświetlane są dostarczone przez agenta typu *Interface Agent* nazwy obiektów dydaktycznych mogących być pomocne użytkownikowi w zrozumieniu pojęć zawartych w aktualnie wybranym obiekcie dydaktycznym. Pomoc ma

strukturę drzewa i zależna jest od relacji, którymi połączony jest aktualny obiekt dydaktyczny z innymi węzłami w sieci semantycznej. Węzły w tym drzewie stanowią węzły pojęć i obiektów dydaktycznych. Pierwszy poziom drzewa stanowią węzły pojęć bezpośrednio powiązane z wybranym obiektem dydaktycznym. Dalsze poziomy drzewa zawierają węzły topologicznie bardziej oddalone od aktualnego węzła obiektu dydaktycznego w sieci semantycznej. Bezpośrednio po wyborze opcji *Pomoc* wyświetlany jest tylko pierwszy poziom drzewa pomocy – wybór trójkątnych ikon powoduje rozwijanie lub zwijanie kolejnych poziomów drzewa.

W przypadku nadmiernego rozrośnięcia się drzewa można ograniczyć przeglądanie do wybranego poddrzewa poprzez wybór węzła pojęcia jako korzenia nowego drzewa (opcja *Jako korzeń* na rys. 2). Opcja *Więcej* ma znaczenie w sytuacji, gdy została ograniczona liczba zwracanych wyników, a w tych z nich, które zostały zwrócone, brak jest szukanych informacji. Wybranie opcji *Więcej* powoduje pokazanie dalszych wyników kwerendy, które nie znalazły się w pierwszej wersji odpowiedzi, ze względu na brak miejsca. Wiersze pomocy związane z obiektami dydaktycznymi obejmują opis informujący o typie obiektu dydaktycznego (np. definicja, statyczny przykład, interaktywny przykład), jak również nazwę modułu, z którego pochodzą (przydatne, gdy przeszukiwanych jest więcej niż jeden moduł dydaktyczny). W przypadku wierszy węzłów pojęć, nazwa pojęcia poprzedzona jest nazwą relacji, w jakiej znajduje się dany węzeł w stosunku do węzła będącego jego bezpośrednim przodkiem w drzewie pomocy.

Opcja *Więcej na temat* zbliżona jest w swoim działaniu do opcji *Pomoc*, istnieją jednak pewne różnice. Opcja *Pomoc* przeznaczona jest dla osób mających trudności w zrozumieniu prezentowanego materiału dydaktycznego, opcja *Więcej na temat* natomiast przeznaczona jest dla osób rozumiejących omawiane zagadnienia, pragnących jednak dowiedzieć się coś więcej na dany temat. W praktyce różnica polega na sposobie przeszukiwania sieci semantycznych, filtracji i sortowaniu wyników. Dla przykładu w trybie pomocy szukane są terminy wymagane do zrozumienia aktualnego pojęcia. W trybie *Więcej na temat* poszukiwane są pojęcia, do których zrozumienia wymagana jest znajomość aktualnie rozpatrywanego pojęcia. Inną różnicą jest to, iż w trybie *Więcej na temat* preferowane są obiekty dydaktyczne o większym stopniu trudności od aktualnego, natomiast w trybie *Pomoc* – o mniejszym poziomie trudności.

Opcja *Spis treści* pokazuje strukturę hierarchiczną wybranego modułu. Odzwierciedla ona relacje typu Structure, pomiędzy węzłami. Opcja *Zmień moduł* zamyka bieżący moduł i pozwala użytkownikowi wybrać nowy moduł dydaktyczny. W opcji *Ustawienia* użytkownik może dostosowywać parametry sposobu uczenia do własnych potrzeb. W praktyce wpływa to na sposób sortowania wyników wyszukiwania obiektów dydaktycznych w sieciach semantycznych. Użytkownik może na przykład zdecydować, iż obiekty dydaktyczne zawierające interaktywne przykłady mają być preferowane w wyszukiwaniu lub to, iż zależy mu przede wszystkim na łatwych zadaniach. Przycisk *Wyloguj* powoduje opuszczenie przez użytkownika systemu.

Graficzny interfejs użytkownika wykonany został przy użyciu ramek HTML. W trakcie działania aplikacji można zmniejszać i zwiększać ilość miejsca okna przeznaczony na obszar obiektu dydaktycznego i opcje menu. Do tworzenia wzorów matematycznych sto-

sowany był język MathML [26]. Do implementacji wykorzystano serwer aplikacyjny Tomcat 4.1 oraz platformę agentową JADE 3.1 [24]. Zaimplementowano prototypowo wybrane funkcje agentów typu Learner Interface Agent (interakcja z użytkownikiem poprzez Servlet) oraz Course Agent i LM Searchbot Agent (pobieranie danych o obiektach dydaktycznych z serwera dydaktycznego). Zrealizowano również elementy serwera obiektów dydaktycznych. Opisy obiektów dydaktycznych w formacie RDF [10, 16, 29] są przechowywane w bazie danych MySQL 4.0 [27] i wyszukiwane za pomocą biblioteki Jena 2.1 [25].

6. Podsumowanie

Analiza zastosowania agentów programowych do celu e-learning oraz prowadzone prace praktyczne skłoniły nas do sformułowania kilku wniosków. E-learning stanowi interesującą formę uzupełniającą tradycyjny sposób uczenia. Agenci programowi mogą być wykorzystani w aplikacjach edukacyjnych w celu dostosowywania procesu dydaktycznego do potrzeb uczących się, jak również w procesie wyszukiwania obiektów dydaktycznych. Sieci semantyczne nadają się dobrze do opisu semantyki materiałów dydaktycznych.

Zaprojektowana architektura MAESTRO łączy w sobie technologie agentowe i koncepcje zastosowania sieci semantycznych do celu efektywnej dystrybucji obiektów dydaktycznych. Zaproponowano i przedstawiono szereg agentów, których zadaniem jest wsparcie procesu nauczania poprzez dostosowanie wyników wyszukiwania do indywidualnych potrzeb osób uczących się, niewidoczne dla użytkownika łączenie zasobów wielu baz obiektów dydaktycznych, monitorowanie postępów w nauczaniu oraz zapewnienie bezpiecznego działania systemu.

Platforma MAESTRO stanowi propozycję rozszerzenia znanych tradycyjnych platform do celu e-learning. Obejmuje ona dwa dodatkowe elementy:

1) serwery materiałów dydaktycznych,
2) serwery danych użytkowników.

Obiekty dydaktyczne dzielone są na moduły, które opisywane są przez sieci semantyczne złożone z trzech typów węzłów (pojęć, obiektów dydaktycznych i struktury).

Planuje się powstanie szeregu aplikacji klienckich przeznaczonych do obsługi platformy MAESTRO, dedykowanych poszczególnym grupom użytkowników. Oprócz tego przewiduje się stworzenie narzędzi dedykowanych dla osób uczących się, pozwalających w prosty sposób wyszukiwać obiekty dydaktyczne, aplikacji dla nauczycieli, których zadaniem jest semantyczne opisanie tworzonych obiektów dydaktycznych, jak również narzędzi dla administratora służących do kontroli funkcjonowania platform oraz zarządzania obiektami dydaktycznymi. Plany obejmują także stworzenie edytora opisów semantycznych dla obiektów dydaktycznych obecnych w architekturze MAESTRO, zaimplementowanie funkcji agentów przeznaczonych dla nauczycieli i administratorów jak również opracowanie systemu związanego z zapewnieniem bezpieczeństwa platformy MAESTRO.

Architektura MAESTRO stanowi dobry punkt wyjścia do podjęcia dalszych badań nad zastosowaniem systemów wieloagentowych w e-learning. W celu pełnej realizacji zapropo-

nowanego rozwiązania wymagane są prace w trzech zasadniczych kierunkach. Z jednej strony należy weryfikować przydatność i efektywność zaproponowanych rozwiązań poprzez implementację wszystkich zaprojektowanych typów agentów. Z drugiej zaś – należy rozwijać aplikacje narzędziowe służące do obsługi platformy (takie jak przedstawiony prototyp aplikacji dla osób uczących się). Natomiast trzeci kierunek prac polega na tworzeniu obiektów dydaktycznych i opisywaniu ich poprzez sieci semantyczne zgodne z architekturą MAESTRO.

Literatura

[1] Bendel O.: *Pädagogische Agenten im Corporate E-Learning*. Univeristy of St. Gallen, 2003 (Ph.D Thesis)

[2] Berners-Lee T., Hendler J., Lassila O.: *The Semantic Web*. In: Scientific American, vol. 284 Issue 5, May 2001, 34

[3] Bigus J, Bigus J.: *Intelligente Agenten mit Java programmieren – eCommerce und Informationsrecherche automatisieren*. München, Addison-Wesley 2001

[4] Cheikes B.: *GIA: An Agent-Based Architecture for Intelligent Tutoring Systems.* In: Proceedings of the CIKM'95 Workshop on Intelligent Information Agents, 1995

[5] Chen W., Mizoguchi R.: *Communication Content Ontology for Learner Model Agent in Multiagent Architecture*. In: Proceedings of the 7th International Conference on Computers in Education, ICCE99, 1999, 95–102

[6] Demazeau Y., Müller J.P.: *Decentralized artificial intelligence*. In: Decentralized A.I., Amsterdam, Elsevier North Holland 1990, 3–13

[7] Fenton-Kerr T.: *GAIA: An Experimental Pedagogical Agent for Exploring Multimodal Interaction*. In: Nehaniv Ch. (Ed.), Computation for Metaphors, Analogy, and Agents. Lecture Notes in Computer Science, Berlin, vol. 1562, 1999, 154–164

[8] Ferber J.: *Multiagentensysteme: eine Einführung in die Verteilte Künstliche Intelligenz*. München, Addison-Wesley 2001

[9] Häfele H.: *E-Learning Standards aus didaktischer Perspektive*. In: Bachmann G., Haefeli O., Kindt M. (Ed.), Campus 2002: Die virtuelle Hochschule in der Konsolidierungsphase, Waxmann, Münster, 2002

[10] Hjelm J.: *Creating the Semantic Web with RDF*. New York, Wiley Computer Publishing, etc., 2002

[11] Jennings N., Wooldridge J.: *Applications of Intelligent Agents*. In: *Agent Technology: Foundations, Applications, and Markets*. Heidelberg, Springer-Verlag 1998, 3–28, http://citeseer.nj.nec.com/jennings98applications.html

[12] Keegan D.: *The future of learning: From elearning to mlearning*. ZIFF-Papiere 119, Zentrales Inst. für Fernstudienforschung, Fernuniv. Hagen, 2002

[13] Kononowicz A.: *Anwendung von Multiagentensystemen im E-Learning*. Praca magisterska pod kierunkiem: Żabińska M., Rakoczy W., Katedra Informatyki, Akademia Górniczo-Hutnicza, Kraków 2004

[14] Luck M., Mcburney P., Preist C.: *Agent Technology: Enabling Next Generation Computing (A Roadmap for Agent Based Computing)*. AgentLink, 2003

[15] Nilsson M., Palmér M., Brase J.: The *LOM RDF binding – principles and implementation*. 3rd Annual Ariadne Conference, Lueven, 2003

[16] Powers S.: *Practical RDF*. O'Reilly&Associates, 2003
[17] Rakoczy W., Żabińska M.: *Cyberprzestrzeń jako środowisko życia mobilnych agentów programowych*. Półr. AGH Automatyka, vol. 7 (1/2), 2003, 245–250
[18] Tadeusiewicz R.: *Dwa cele i dwa modele e-learningu: model minimalnych kosztów masowego kształcenia oraz model maksymalnej jakości kształcenia elitarnego*. Konf. „e-Learning w Społeczeństwie Wiedzy", Kraków, 14 marca 2005
[19] Żabińska M.: *Zastosowanie technologii agentów do egzaminowania*. Półr. AGH Automatyka, t. 5, z. 1/2, 2001, ISSN 1429–3447, 587–593
[20] Żabińska M., Rakoczy W.: *„Ignorance tree" – Examination with the Use of Agents*. In: Proc. of the Workshop MANAM 2000 of 34th International Spring Conference MOSIS 2000, Brejcha M., Štefan J. (Eds), Acta MOSIS, No. 81, May 2–4 2000, Rožnow pod Radhoštěm, Czech Republic, 1st Edition, MARQ, Ostrava 2000, ISBN 80-85988-46-1, 71–76
[21] Żabinska M., Rakoczy W., Cieszkowski M., Gołębiewski G.: *The Multi-Agent System for Examination – A Practical Realization of Ignorance Tree Concept*. In: Štefan J. (Ed.), Proc. of XXIInd International Colloquium ASIS 2000, Sv. Hostyn, MARQ, Ostrava, 2000, 117–124
[22] Advanced Distance Education, http://www.isi.edu/isd/ADE/ade.html
[23] IEEE LTSC – Learning Object Metadata, http://ltsc.ieee.org/wg12/index.html
[24] JADE – Java Agent DEvelopment Framework http://jade.tilab.com/
[25] JENA – http://jena.sourceforge.net/
[26] MathML – http://www.w3.org/Math/
[27] MySQL – http://www.mysql.com/
[28] OWL – http://www.w3.org/TR/owl-features/
[29] RDF (Resource Description Framework) http://www.w3.org/RDF
[30] W3C http://www.w3.org
[31] W3C Web Services Activity http://www.w3.org/2002/ws/
[32] WebServices.org http://www.webservices.org