Agenci programowi jako metodologia tworzenia oprogramowania

Marcin Paprzycki

Computer Science Department , Oklahoma State University, Tulsa, OK 74106 USA

marcin@cs.okstate.edu

Streszczenie

Często powtarzaną jest teza, że inteligentni agenci programowi będą kolejną rewolucją w informatyce. Ma to dotyczyć w szczególności zastosowań agentów jako metodologii tworzenia oprogramowania. Jak łatwo się przekonać, zapowiadany przełom nie nastąpił. W poniższym tekście podjęta zostaje próba odpowiedzi na pytanie, dlaczego tak się nie stało i co można zrobić, aby to uległo zmianie.

1 Wprowadzenie

Od wielu lat mówi się, że rozwiązania technologiczne bazujące na inteligentnych agentach programowych będą kolejną rewolucją w informatyce. Ma to dotyczyć nie tylko sposobu w jaki komunikujemy się z komputerami [TBL2001, HEN1999, MAE1994], ale również metodologii tworzenia oprogramowania [GRI1999, JEN2001]. Niestety, jak się łatwo przekonać, ów rewolucyjny przełom, przepowiadany od 1994 roku przez „wizjonerów agentyzmu", nie nastąpił (pomimo rosnącej liczby konferencji, sesji specjalnych, warsztatów, publikacji itp.). Nie jest tak, że włączając komputer kontaktujemy się z agentem osobistym, który wyświetla wyselekcjonowane wiadomości, przedstawia plan dnia, a na podstawie prognozy pogody, programu dnia i naszych upodobań doradza np. jak się ubrać itp. (agent-idealny kamerdyner). Podobnie, gdy tworzymy oprogramowanie dla sklepu internetowego nie budujemy go używając ogólnodostępnych modułów „agentowych" (np. agent-specjalista od reklamy, agent-zarządzający zamówieniami itp.). Nie jest też niestety tak, że możemy powierzyć zarządzanie siecią komputerową inteligentnemu agentowi, który na przykład uzgodni z agentami lokalnymi dla węzłów sieci, jaki jest najlepszy harmonogram instalacji nowej wersji oprogramowania (agent-administrator sieci).

Powstaje więc pytanie, dlaczego istnieje tak niewiele systemów składających się z inteligentnych agentów programowych i co można zrobić, aby sytuacja ta uległa zmianie. Wprawdzie istnieje bardzo wiele możliwych zastosowań agentów (i niektóre z nich zostaną wymienione w sekcji 3), ale w kontekście tej pracy interesuje nas głównie wykorzystanie ich do tworzenia oprogramowania. Naszym celem jest zarysowanie tematyki i pozostawienie czytelnikowi oceny, czy technologia agentowa jest przyszłością programowania. W tym celu postępować będziemy jak następuje. Zaczniemy od przedsta-

wienia istniejących definicji i wybranych zastosowań inteligentnych agentów programowych. Następnie przedstawimy najczęściej wymieniane zalety i wady systemów agentowych. Stąd przejdziemy do omówienia pozytywnych tez Jenningsa [JEN2001] i krytyki Nwany i Ndumu [NWA1999]. Na zakończenie porównamy istotne dla tworzenia oprogramowania cechy czterech systemów agentowych: Aglets, Concordia, Grashopper i Voyager.

2 Definicja agenta programowego

Jak się okazuje nie istnieje ogólnie przyjęta definicja agenta programowego. Poniżej przedstawiamy cztery popularne definicje. Dla każdej z nich podajemy autora i rok publikacji oraz w przypisach wersję oryginalną.

Definicja 1. Cokolwiek, co może być uznane jako obserwujące otoczenie poprzez sensory i działające w ramach tegoż otoczenia poprzez efektory (Russell i Norvig, 1995)[1].

Definicja 2. Jednostki programowe podejmujące działania w imieniu użytkownika lub innych programów, w pewnym stopniu niezależnie lub autonomicznie, które działając stosują pewną wiedzę lub reprezentację celów lub potrzeb użytkownika (IBM, 1997)[2].

Definicja 3. Zamknięty system komputerowy znajdujący się w pewnym otoczeniu, posiadający umiejętność elastycznego działania w tymże otoczeniu, działania polegającego wypełnieniu celów dla jakich został stworzony (Wooldridge, 1997)[3].

Definicja 4. Autonomiczny system znajdujący się w dynamicznym otoczeniu działający niezależnie od narzucanych przezeń ograniczeń i wypełniający w jego ramach zbiór celów lub poleceń, dla których został stworzony (Maes, 1998)[4].

Nawet pobieżna analiza powyższych definicji pokazuje, że wprawdzie istnieje pomiędzy nimi pewne podobieństwo, to są one w sposób istotny rozbieżne. Nie jest tylko przypadek wynikający z wyboru tych a nie innych definicji. Jak można się przekonać z pracy [GAL2001], jest to niestety sytuacja typowa. W [GAL2001] autorzy podsumowali literaturę dotyczącą systemów agentowych i ustalili, że nie istnieje jeden, ogólnie przyjęty model systemu agentowego. Mamy natomiast do czynienia z szeroką gamą cech przypisywanych agentom i systemom agentowym. W tabeli 1. prezentujemy zbiór wybranych cech, pochodzących tak z pracy [GAL2001] jak i z innych źródeł. Aby uniknąć nieporozumień w tłumaczeniu terminów angielsko-języcznych, dla każdej cechy podajemy również jej nazwę oryginalną.

Tabela 1. Cechy przypisywane agentom i systemom agentowym

[1] Anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors (Russell and Norvig, 1995)

[2] Software entities that carry out some set of operations on behalf of a user or another program with some degree of independence or autonomy, and in so doing, employ some knowledge or representation of the user's goals or desires (IBM, 1997)

[3] An encapsulated computer system that is situated in some environment and that is capable of flexible action in that environment in order to meet its design objectives (Wooldridge, 1997)

[4] Autonomous system situated in a dynamical environment acting independently of its restrictions and fulfilling in it a set of goals or directives for which it was created (Maes, 1998)

Cecha	Źródło	Cecha	Źródło
reaktywność	reactiveness	rozumowanie oparte o zgromadzoną wiedzę	reasoning based on collected knowledge
ukierunkowanie na osiąganie celów	goal orientation	Umiejętność przemieszczania się	mobility
Autonomia	Autosomy	bycie godnym zaufania	reliability
Umiejętność dostosowania	Adaptivity	oddziaływanie	Interactivity
umiejętność uczenia	learning ability	umiejętność przewidywania	Proactivity
umiejętność porozumiewania się	ability to communicate	umiejętność rozumowania	capacity for reasoning
umiejętność współdziałania	capacity for cooperation	inteligencja	intelligence

Warto zwrócić uwagę, że cechy wymienione w tabeli 1. są bardzo specyficzne. Między innymi nie określają one odrębnych właściwości, a wręcz przeciwnie, ich definicje wzajemnie się zawierają. I tak, zwykle przyjmuje się, że umiejętność uczenia się czy też umiejętność rozumowania, są częścią definicji inteligencji. W podobny sposób, umiejętność dostosowania jest bardzo ściśle związana z umiejętnością przewidywania. Jak pokazano w pracy [GAL2001], w praktyce badacze wybierają podzbiór powyżej wymienionych cech i ew. uzupełniają go o dodatkowe cechy i używają tego podzbioru jako definicji agentów programowych i/lub systemów agentowych. Wynikiem tego postępowania jest jeszcze większe „zamieszanie definicyjne". W tym kontekście warto wspomnieć, że istnieje jeszcze inne podejście do problemu. Na przykład niedawno opublikowana książka [LIU2001] nie zawiera definicji agentów programowych. Są one omawiane na takiej zasadzie jak w staropolskiej definicji konia: „koń, jaki jest, każden widzi", tylko tym razem jest to: „agenci programowi są tym, o czym jest ta książka".

Podsumowując, wszystko wskazuje na to, że teoria i praktyka inteligentnych agentów programowych okazuje się nie być dziedziną okrzepłą na tyle, aby wypracować powszechnie akceptowalną terminologię. Ponadto warto dodać, że nie istnieje również ogólnie przyjęta typologia agentów i systemów agentowych (np. ze względu na ich zastosowania czy też inne cechy). Biorąc to pod uwagę, pozostawimy nasze rozumienie agentów programowych na poziomie intuicyjnym i bazowanym na powszechnie znanych przykładach agentów, takich jak agenci ubezpieczeniowi, czy też agenci turystyczni. Tak więc jako agenta będziemy rozumieli oprogramowanie, które jest w jakimś stopniu autonomiczne, reprezentuje czyjeś interesy (użytkownika lub innego oprogramowania – w tym innych agentów) i oddziałuje z otoczeniem i/lub innymi agentami.

3 Zastosowania agentów programowych

Pomimo braku ściśle określonej terminologii agentów programowych odnaleźć można w bardzo szerokiej gamie projektów badawczych jak i wstępnych próbach implementacji praktycznych zastosowań. Jednym z możliwych sposobów podziału sfery istniejących zastosowań jest wyróżnienie: (a) agentów jako narzędzi personalizacji, (b) zastosowania agentów związane z tworzeniem oprogramowania dla systemów rozproszonych (w tym w szczególności zastosowania związane z Internetem), oraz (c) zastosowania agentów do modelowania systemów złożonych (ang. complex systems). Przedstawimy teraz niektóre z zastosowań występujące we wszystkich trzech kategoriach. Oczywiście możliwe są również inne sposoby podziału systemów agentowych, jak i istnieje znacznie więcej praktycznych zastosowań rozważanych w ramach programów badawczych na uczelniach i w laboratoriach.

3.1 Agenci w zarządzaniu informacją

Wspomaganie użytkownika. Do kategorii tej należą agenci mający za zadanie inteligentne doradzanie użytkownikowi systemu. Typowym, choć funkcjonalnie niezbyt udanym, przykładem takowego systemu jest „spinacz-doradca" w programie Microsoft Word'97. Dzięki ciągłemu doskonaleniu systemów komputerowych (mając na uwadze zarówno rozwój sprzętu jak i oprogramowania) realne stały się już eksperymenty z tworzeniem animowanych agentów doradców / przewodników np. w bankowości elektronicznej, na firmowych stronach WWW czy też w sklepach internetowych.

Zarządzanie pocztą elektroniczną. Agenci tego typu mają za zadanie kompleksowe zarządzanie nadchodzącą poczta elektroniczną. W tym, na przykład, sortowanie jej do folderów, nadawanie priorytetów, automatyczne odpowiedzi na rutynowe zapytania itp.

Zarządzanie organizacją dnia. Zadaniem tego typu agentów jest organizacja dnia użytkownika. Są one odpowiedzią na wzrastającą liczbę zadań i zebrań (zwłaszcza w przypadku managerów wszystkich szczebli). Ich celem jest m.in. negocjowanie z agentami reprezentującymi pozostałych uczestników planowanego zebrania, kiedy może się ono odbyć (kiedy wszyscy mają „wolny" czas).

Indywidualizowane dostarczanie informacji. Agenci tej grupy stanowić mają odpowiedź na wzrastającą geometrycznie ilość dostępnej w sieci informacji. Mają one filtrować informację w taki sposób, aby dostarczona została kompletna, wymagana informacja i tylko potrzebna informacja. Agenci tego typu, w celu osiągnięcia stawianych przed nimi celów, współpracować mogą z agentami poszukującymi i przeglądającymi (sekcja 3.2).

Agenci monitorujący lub zarządzający. Jest to klasa agentów, których zadaniem jest monitorowanie zjawiska i ew. podejmowanie działań w imieniu użytkownika. Prostym agentem (monitorującym) będzie agent śledzący ceny akcji i informujący użytkownika w momencie wystąpienia oczekiwanego zdarzenia (np. cena akcji BRE osiągnęła poziom 141.00 PLN). Agent zarządzający natomiast będzie nie tylko obserwował ceny akcji, ale również dokona zakupu (lub sprzedaży) akcji, kiedy osiągnięty zostanie

wymagana cena.

3.2 Agenci w systemach rozproszonych

Agenci poszukujący informacji. Są to agenci, których zadaniem jest odnalezienie ściśle określonej informacji na podstawie określonej sekwencji słów kluczowych lub też na podstawie pytań zadawanych w języku naturalnym. Jest to typ agenta, który może zostać wywołany i wydelegowany przez agenta „osobistego".

Agenci przeszukujący Internet. Ich zadanie polega na przeszukiwaniu Internetu w celu odnalezienia informacji, która może być przydatna dla użytkownika. W oczywisty sposób są bardzo podobni do agentów poszukujących z tą różnicą, że agenci poszukujący mają zwykle za zadanie odnalezienie ściśle określonej informacji, a agenci przeszukujący mają za zadanie odkrycie różnorodnej przydatnej informacji. Podobnie jak poprzednio, jest to rodzaj agenta, który może być wywołany i wydelegowany przez agenta „osobistego".

Agenci e-biznesu i m-biznesu. Są to agenci, których zadania związane są z różnorodnymi funkcjami systemów handlu Internetowego. Należą do tej kategorii na przykład agenci poszukujący najlepszej ceny towaru (a więc mogą do tej kategorii należeć również wybrani agenci poszukujący), agenci negocjujący cenę zakupów czy też agenci zarządzający zawartością magazynów. Choć bardzo podobne, funkcje niektórych agentów wspomagających systemy handlu Internetowego działające w środowisku składającym się z komputerów przyłączonych bezpośrednio do sieci (e-biznes) różnią się od agentów wspomagających funkcje bazowane na urządzeniach mobilnych (m-biznes). Różnice te wynikają z faktu, że urządzanie mobilne może być odłączone od sieci przez pewien nieokreślony czas. Równocześnie jakość połączenia może być bardzo zła a transmisja danych bardzo powolna.

Agenci zarządzający siecią. Jest to bardzo liczny zbiór agentów, których rola polega na wykonywaniu różnych funkcji związanych z zarządzaniem komputerami połączonymi w sieć. Może to być na przykład zarządzanie aktualizacją oprogramowania, wykrywanie ataków czy też optymalizacja przepływu informacji.

3.3 Agenci w modelowaniu systemów złożonych

Jak do tej pory zainteresowani byliśmy bezpośrednimi zastosowaniami systemów agentowych. Jednakże systemy te służą bardzo często jako narzędzia do modelowania zjawisk świata rzeczywistego. Mogą to być agenci, którzy są wykorzystani bezpośrednio w modelowaniu, ale modelowanie to może stać się punktem wyjścia do praktycznych zastosowań (a więc przejścia, na przykład, do kategorii „agenci w systemach rozproszonych"). Do takich zaliczyć można między innymi:

Agenci modelujący zachowania w czasie negocjacji cenowych. W tym przypadku agenci reprezentują różnorodne strategie składania zleceń i używani są jako narzędzia wspomagające modelowanie rzeczywistych zachowań uczestników licytacji. Celem jest zarówno odkrycie nowych strategii licytacji jak i opracowanie nowych klas agentów,

którzy będą w stanie autonomicznie reprezentować klientów w handlu elektronicznym.

Agenci modelujący proces zarządzania produkcją. Celem badań jest opracowanie skutecznych strategii zarządzania kompleksową strukturą przedsiębiorstwa. Długoplanowym zadaniem jest stworzenie infrastruktury agentowej wspomagającej zarządzanie i/lub będącej w stanie przejąć autonomiczną kontrolę nad procesami biznesowymi mającymi miejsce w przedsiębiorstwie.

Istnieją też zastosowania agentów, które mają na celu modelowanie skomplikowanych procesów społecznych i biologicznych. Możemy więc mieć do czynienia na przykład z modelami, w których agenci wyposażeni są w przekonania i cele i muszą współpracować, aby osiągnąć wyznaczony cel.

W kontekście tego artykułu jesteśmy zainteresowani głównie zastosowaniem agentów jako metodologii tworzenia oprogramowania, a więc wybranymi agentami z sekcji 3.1 i 3.2. Przedstawimy teraz najczęściej wymieniane zalety i wady systemów agentowych tego typu.

4 Podstawowe zalety i wady systemów agentowych

4.1 Zalety

W literaturze przedmiotu wymienianych jest wiele potencjalnych zalet technologii agentowych. Jeden z popularnych argumentów „za" brzmi: agenci stanowią najlepszą odpowiedź na gwałtowny wzrost ilości dostępnych w Internecie informacji. To właśnie agenci personalni, na podstawie wiedzy o naszych zainteresowaniach będą w stanie filtrować zawartość Internetu dostarczając nam: dokładnie tej informacji, która jest potrzebna, tylko tej informacji, która jest potrzebna i wówczas, gdy jest ona potrzebna.

Inny argument przemawiający za technologiami agentowymi oparty jest na obserwacji, że systemy agentowe składać się mają z inteligentnych i posiadających zdolność uczenia się (adaptacji) elementów. W dodatku elementy te będą w stanie komunikować się ze sobą, a więc wymieniać doświadczenia i dzięki temu w dodatkowy sposób wspomagać ewolucję systemu mającą na celu dostosowanie do środowiska. Stąd to właśnie systemy agentowe stanowią najlepszą możliwą platformę do budowy inteligentnych systemów adaptacyjnych.

Zauważmy również, że użytkownicy systemów komputerowych stają się coraz bardziej mobilni i konieczne jest stworzenie oprogramowania, które będzie również mobilne. Mówi się więc, że to właśnie mobilne systemy agentowe są najlepszą platformą wspomagania mobilnych użytkowników. Fakt ten jest szczególnie widoczny w przypadku tworzenia oprogramowania wspomagającego podróżnych (np. Internetową agencję turystyczną, gdzie personalni agenci turystyczni mogą przemieszczać się wraz z podróżniczką).

Ostatni argument jest najbardziej istotny w kontekście tej pracy. Twierdzi się, że systemy agentowe mogą stać się nową metodologią tworzenia oprogramowania. Szczegóły tego rozumowania omówione zostaną w sekcji 5.

Poza argumentami bazującymi na konieczności dokonania zasadniczych zmian w różnych aspektach tworzenia oprogramowania i dowodzącymi, że to technologie agentowe są najlepszym rozwiązaniem, systemy agentowe mają wiele pomniejszych zalet. Po pierwsze, weźmy pod uwagę systemy, w których skład wchodzą agenci mobilni, czyli tacy, którzy mogą przemieszczać się pomiędzy komputerami. Łatwo wówczas zauważyć, że umożliwia to znacznie lepsze wykorzystanie dostępnych zasobów. Jeśli dany komputer nie posiada dostatecznej mocy obliczeniowej, to agent mobilny może się przemieścić na inny komputer i tam dokończyć przetwarzanie informacji. Możliwym jest też na przykład utworzenie kopii agenta (klonowanie) i rozesłanie ich do innych komputerów w sieci, w celu dokończenia podzielonych na mniejsze części obliczeń. Zwróćmy tutaj uwagę na fakt, że jest to podejście bardzo zbliżone i komplementarne do tak popularnych ostatnio obliczeń gridowych (ang. grid computing). Po drugie, systemy agentowe z minimalną choćby autonomią agentów wpasowują się naturalnie w krajobraz wypełniony urządzeniami mobilnymi, których połączenie z siecią jest krótkotrwałe, powolne lub złej jakości. W tym przypadku naturalnym jest lokalne uruchomienie agenta, który następnie migruje do sieci i podejmuje działania w czasie, gdy lokalne urządzenie zostaje odłączone (zauważmy, że agent może być niewielkich rozmiarów, co ułatwia przesłanie, natomiast zasoby konieczne do wykonania zadania dostępne będą „w sieci"). Warto również zaobserwować, że podejście to umożliwia zaoszczędzenie baterii urządzeń mobilnych i częściowo uniezależnia system od tymczasowego braku dostępu do sieci. Po trzecie, jedną z podstawowych funkcji systemów agentowych jest możliwość tworzenia kopii agentów, co ułatwia wprowadzenie redundancji do systemu, a tym samym ograniczenie jego zawodności.

Należy tutaj zaznaczyć, że w literaturze przedmiotu wymienianych jest wiele zalet podejścia agentowego i nie jest celem tego artykułu wymienienie wszystkich. Natomiast jest również znanym fakt, że systemy agentowe nie pozostają bez krytyki. Przedstawimy teraz kilka bardziej popularnych krytyk podejścia agentowego.

4.2 Krytyka podejścia agentowego

Jedna z najpoważniejszych krytyk podejścia agentowego może zostać podsumowana w następujący sposób. Cokolwiek może zostać opracowane i zaimplementowane w postaci systemu agentowego, może zostać zaprogramowane również w inny sposób. Można w miarę prosty sposób wykazać, że zdecydowana większość funkcji dostępnych w istniejących systemach agentowych może zostać zaimplementowana przy pomocy jednego ze znanych już istniejących mechanizmów. Na przykład, mobilność zastąpiona może być przez wykonywanie funkcji na odległość (ang. remote procedure calling), a klonowanie jest zbliżone koncepcyjnie do wielostrumieniowości w systemie operacyjnym Unix (gdzie każdy strumień jest kopią pozostałych).

Są również tacy, którzy zwracają uwagę na fakt, że systemy agentowe jako idea istnieją już bardzo długo, a intensywne badania nad nimi trwają przynajmniej od początku lat 90-tych. Biorąc to pod uwagę trudno jest zrozumieć fakt, że do tej pory nie ma znaczących implementacji systemów agentowych. Wskazywać to może na istnienie istotnych problemów z samym podejściem i oznaczać, że wiara w zbawienne skutki technolo-

gii agentowych jest tylko tymczasową modą, która z czasem zaniknie (jak wiele innych mód, od których świat informatyki nie jest wolny).

Jeszcze inna krytyka powraca do głównego wątku zarysowanego w sekcji 2. i ma swoje źródło głównie w kręgach badaczy reprezentujących teoretyczne podejście do informatyki. Wskazują oni na fakt, że teoria systemów agentowych przez prawie 20 lat nie była w stanie wypracować precyzyjnej definicji agenta programowego i systemów agentowych. Dla tychże badaczy stanowi to dowód na istnienie istotnych problemów z dyscypliną jako taką.

Jedną z bardziej pragmatycznych krytyk jest zwrócenie uwagi na istotne problemy z bezpieczeństwem, które mają swoje źródło w zastosowaniu technologii agentowej. Łatwo jest zauważyć, że agent to potencjalny mobilny koń trojański mogący odwiedzić wiele komputerów. Agent to też reprezentant użytkownika. Jako taki zawiera on istotne informacje o jego preferencjach (informacje, które są konieczne, aby być w stanie tegoż użytkownika reprezentować). Stąd konieczne jest stworzenie mechanizmów chroniących komputery przed agentami (a zauważmy, że agenci, będąc w stanie się powielać i odwiedzając komputery połączone mogą w bardzo prosty sposób stać się wirusami lub być ich nosicielami), agentów przed komputerami, które odwiedzają, agentów przed innymi agentami mającymi na celu uzyskanie informacji o ich zawartości, jak również komunikację pomiędzy agentami reprezentującymi danego użytkownika. Konieczność utworzenia skutecznych mechanizmów zabezpieczających wszystkie powyższe aspekty technologii agentowej czyni ją znacznie mniej atrakcyjną.

Na zakończenie warto wspomnieć o dosyć powszechnym problemie związanym z popularnością technologii agentowej. Pobieżne zapoznanie się z istniejącą literaturą jak i tekstami nadsyłanymi na konferencje wskazuje, że badacze umieszczają agentów w scenariuszach, które zupełnie takowych nie wymagają. Zwiększa to poczucie, że technologie agentowe są w rzeczywistości tylko modą, która przeminie.

4.3 Odpowiedzi na krytykę

Wydaje się, że możliwym jest przedstawienie racjonalnych odpowiedzi na powyższą krytykę. Wprawdzie odpowiedzi te nie przekonają zagorzałych krytyków, ale takowi mogą zostać przekonani tylko, jeśli nastąpi dalszy rozwój technologii agentowych oraz zaczną się pojawiać zaimplementowane i skutecznie działające zastosowania, które ponadto zaistnieją poza laboratoriami badawczymi.

Oczywistym jest, że w relatywnie młodej i aktywnie rozwijającej się dziedzinie będzie wiele projektów badawczych starających się wykorzystać panującą modę. Z czasem projekty te zostaną wyeliminowane w procesie naturalnej selekcji. Stan aktualny wielu dziedzin informatyki pokazuje też, że możliwym jest prowadzenie badań, projektowanie i implementowanie systemów, bez precyzyjnych definicji. Czas na precyzyjne definicje nastąpi w sposób naturalny wraz z rozwojem dyscypliny.

Problemy związane z bezpieczeństwem różnorodnych aspektów systemów agentowych nie różnią się zasadniczo od konieczności zapewnienia bezpieczeństwa systemom opartym o inne technologie. Tak więc ten aspekt technologii agentowej, podobnie jak

podobieństwo agentów do wirusów, nie powinny stanowić zasadniczej przeszkody w rozwoju systemów agentowych.

Wprawdzie rzeczywiście jest tak, że wszystko to co można zaimplementować w ramach systemu agentowego może być zaimplementowane w inny sposób, ale jest to sytuacja podobna do faktu, że wszystko to co można zaprogramować w Fortranie można też zaprogramować w asemblerze. To co jest istotne to fakt, że technologia agentowa może stanowić zunifikowane podejście do tworzenia oprogramowania. W podobny sposób jak technologia obiektowa zastępuje programowanie w językach takich jak Fortran czy C, tak programowanie agentowe może w przyszłości zastąpić programowanie obiektowe.

Na zakończenie zauważmy, że tworzenie systemów agentowych jest związane z tworzeniem oprogramowania dla bardzo skomplikowanych systemów i wymaga zapewnienia współpracy pomiędzy technologiami, które były tworzone niezależnie od siebie i w związku z tym w chwili obecnej nie są przygotowane do współpracy. Ponadto systemy takie zawierają elementy pochodzące z bardzo wielu dziedzin. Dla przykładu weźmy pod uwagę tworzenie oprogramowania dla Internetowej agencji turystycznej [ANG2002]. W tym przypadku konieczna jest integracja teorii, narzędzi i technologii pochodzących z następujących dziedzin (poniższa lista nie jest wyczerpująca ale skutecznie reprezentuje wielostronność problemu):

- marketing i personalizacja,
- zarządzanie wiedzą,
- systemy czasu rzeczywistego,
- zarządzanie rozproszonymi zasobami,
- systemy ekspertowe,
- komunikacja człowiek-komputer,
- ontologie i ich zastosowanie,
- uczenie maszynowe,
- systemy rozproszone,
- odkrywanie wiedzy,
- zarządzanie i komunikacja w sieci
- filtrowanie informacji ze źródeł nieustrukturalizowanych.

To właśnie poziom komplikacji zadania stojącego przed twórcami oprogramowania bazowanego na systemach agentowych jest, być może, jedną z głównych przyczyn tego, że w chwili obecnej istnieje tylko niewiele zaimplementowanych systemów testowych.

5 Agenci w tworzeniu oprogramowania

Jak do tej pory omówiliśmy definicję i zastosowania agentów programowych, jak również argumenty za ich stosowaniem i popularną krytykę agentowego podejścia do tworzenia oprogramowania. Istnieje jednakże bardziej interesujący argument, dlaczego technologia agentowa stanowi ciekawą alternatywę dla innych metod tworzenia oprogramowania. Argument ten przedstawiony został przez Jenningsa w pracy [JEN2001]

i skierowany jest na tworzenie oprogramowania dla dużych skomplikowanych systemów (ang. complex software systems). Jennings pokazuje najpierw, w jaki sposób stopień skomplikowania objawia się w budowanym oprogramowaniu, a następnie argumentuje, w jaki sposób oprogramowanie oparte o technologię agentową pomóc może w opanowaniu tego problemu. Teza Jenningsa opiera się na pokazaniu jak, w naturalny sposób, dekompozycja systemu na współdziałających agentów odpowiada standartowym technikom opracowanym przez badaczy inżynierii oprogramowania do budowy dużych systemów. I tak, zgodnie z [BOO1994] najważniejszymi metodami umożliwiającymi opanowanie wzrostu skomplikowania tworzonego oprogramowania są: dekompozycja, abstrakcja i organizacja. Dekompozycja, to podział problemu na mniejsze części, które mogą być oprogramowane relatywnie niezależnie od innych części. Abstrakcja to proces definiowania uproszczonego modelu problemu, który podkreśla pewne aspekty problemu, równocześnie ukrywając inne (w danym momencie mniej istotne). Natomiast organizacja, to proces zarządzania oddziaływaniami pomiędzy komponentami powstającymi w wyniku dekompozycji i abstrakcji. Wydaje się, że biorąc pod uwagę przedstawione do tej pory podstawy teorii systemów agentowych, w miarę prosty sposób widać, jak powyższe techniki mogą być wyrażone w tych właśnie terminach. Tak więc, w swojej pracy Jennigs argumentuje, że: (a) wyrażenie problemu w terminach niezależnych agentów jest efektywną metodą podziału problemu (abstrakcja), (b) podstawowe abstrakcje związane z podejściem agentowym są naturalnym sposobem modelowania skomplikowanych problemów (dekompozycja), (c) podejście agentowe do modelowania i zarządzania organizacją i oddziaływaniami pomiędzy komponentami systemu jest właściwym sposobem modelowania zależności istniejących w dużych problemach i jest też najlepszym podejściem do ich implementacji (organizacja).

Zakładając prawdziwość argumentu przedstawionego przez Jenningsa, powraca pytanie: dlaczego istnieje tak niewiele systemów opartych o technologie agentowe? Wydaje się, że jedna z możliwych odpowiedzi znajduje się w pracy Nwany i Ndumu [NWA1999]. Zanim przedstawimy ich krytykę podejścia do tworzenia oprogramowania przy pomocy agentów programowych warto zwrócić uwagę na kilka faktów. (1) Po opublikowaniu w 2001 r. pracy [JEN2001], Jennings zajął się badaniami nad teorią i praktyką oprogramowania dla agentów prowadzących negocjacje cenowe (sekcja 3.3) i nie powrócił już do teorii systemów agentowych jako metodologii tworzenia oprogramowania. (2) Pisząc artykuł w roku 2000, Jennings powinien był znać tekst Nwany i Ndumu, ale całkowicie go pomija (co może być związane z wysoce krytycznym i czasami wręcz obraźliwym tonem tejże publikacji). (3) Nwana i Ndumu, jako badacze zatrudnieni przez British Telecom, pracowali nad implementacją systemów agentowych (m.in. Internetowej agencji turystycznej), są więc praktykami zastosowań technologii agentowych i stąd różnice w doświadczeniach jak i podejściu do tematu. Naszkicujmy więc najważniejsze problemy wymienione w ich artykule.

1. Problem odkrywania informacji – jaka informacja jest konieczna dla funkcjonowania systemu, gdzie się ona znajduje i jak umożliwić nieprzerwany dostęp do danych, których reprezentacja ciągle się zmienia (np. strony WWW są modyfikowane i w związku z tym informacja, która była dostępna w pewnym miejscu może być już w nim niedostępna, a dostępna na „sąsiedniej stronie").

2. Problem komunikacji – jak sprawić, aby systemy były w stanie komunikować się ze sobą; na przykład w jaki sposób agenci mają komunikować się z dostawcami informacji, gdy każda z tych informacji może być w innej formie i nie istnieje ściśle określony standard komunikacji.
3. Problem ontologii – jak sprawić, aby różne systemy: agenci komunikujący się z agentami jak i agenci oddziałujący z różnorodnymi źródłami informacji, rozumiały się nawzajem na poziomie semantycznym.
4. Problem długowiecznego oprogramowania (*ang. legacy software*) – jak sprawić, aby agenci programowi byli w stanie komunikować się z systemami, które powstały jako samoistne jednostki i nie mają wbudowanych mechanizmów współpracy z innym oprogramowaniem (poza narzędziami współpracy ze ściśle określonymi partnerami; np. system rezerwacji SABRE, który w celu maksymalizacji przepustowości jest w dużej części oprogramowany w asemblerze).
5. Problem rozumowania – w jaki sposób agenci mają przetwarzać uzyskane z Internetu dane, aby wykorzystać je w celu, dla którego zostały pozyskane.
6. Problem monitorowania – jest to problem specyficzny dla systemów wspomagania podróżnych i jako taki ma najmniejszą wagę. Jeśli podróżny ma udać się za miesiąc do Chin, system powinien śledzić wydarzenia na świecie i na przykład poinformować klienta o strajku kontrolerów lotniczych w Paryżu, który to strajk może uniemożliwić podróż.

Zauważmy, po pierwsze, że problemy przedstawione powyżej są znacznie bardziej praktyczne niż problemy omówione w sekcji 4.2. Po drugie, gdy Nwana i Ndumu opublikowali tekst w roku 1999 ich argumenty były oparte o analizę problemów związanych z budową Internetowych agencji turystycznych opartych o technologię agentową. Agencje takie nie tylko nie istniały wówczas, ale nadal nie powstały. Gdy w roku 2001 przeszukaliśmy Internet, to znaleźliśmy 8 niedokończonych projektów i 3 systemy testowe o minimalnej użyteczności. Co więcej, do dzisiaj nie natrafiliśmy na projekt, który byłby sukcesem, pomimo istnienia wieloletnich projektów badawczych fundowanych z grantów Unii Europejskiej. Wskazywać to może na fakt, że problemy przez nich wymienione nie są wcale trywialne.

Można więc powiedzieć, że nawet jeśli problemy postawione przez Nwane i Ndumu wydawać się mogą proste do rozwiązania (większość z nich jest w chwili obecnej w centrum intensywnych badań prowadzonych w wielu ośrodkach naukowych), to jednak ich główna teza, że jest jeszcze bardzo daleko do momentu kiedy będziemy w stanie tworzyć oprogramowanie przy pomocy technologii agentowych, pozostaje w mocy. Być może problem, na który oni wskazują jest inną wersją problemu, który został zarysowany w sekcji 4.3. Nie jest rzeczą skomplikowaną znalezienie pewnego rozwiązania wybranej części jednego z problemów postawionych przez Nwanę i Ndumu. To co jest potrzebne, to całościowe i skuteczne rozwiązanie najważniejszych aspektów wszystkich powyższych problemów. To z kolei oznacza konieczność stworzenia oprogramowania, którego źródła znajdą się w wielu dyscyplinach będących częścią informatyki, zarządzania i marketingu. I to właśnie owa niezbędna multidyscyplinarność rozwiązania stanowi najtrudniejszy do rozwiązania problem.

6 Istniejące otoczenia agentowe

Praca Nwany i Ndumu jest nie tylko krytyczna. Zawiera ona również sugestię, w jaki sposób należy postępować, aby wyjść z obecnej sytuacji kryzysowej. Jak się można spodziewać sugestia ta jest bardzo pragmatyczna; twierdzą oni bowiem, że koniecznym jest odejście od ideałów (tworzenia teorii i założeń wspaniałych systemów agentowych, które będą w stanie zaspokoić nasze najskrytsze potrzeby; takich jak systemy postulowane przez P. Maes [MAE1994] czy J. Hendlera [HEN1999]) oraz przejście do tworzenia i implementacji rzeczywistych systemów agentowych. Systemów bazowanych na istniejących technologiach. Systemów, których implementacja będzie stanowiła niezwykle istotne źródło doświadczeń pozwalających na lepsze zrozumienie istniejących problemów, tworzenie standardów oraz budowę narzędzi następnej generacji systemów agentowych.

Zakładając, że podejście sugerowane przez Nwanę i Ndumu jest słuszne i istotnym jest implementacja oprogramowania opartego o technologię agentową, powstaje pytanie: jakie narzędzia dostępne są dla tworzenia takowego oprogramowania. W chwili obecnej istnieje ponad 80 systemów tworzenia oprogramowania agentowego [ALT2001] a kolejne są nieustannie tworzone. Tak więc, zamiast koncentrować się na najnowszych systemach, porównamy cztery popularne systemy agentowe z punktu widzenia aspektów istotnych dla inżynierii oprogramowania. Te cztery systemy to: Aglets (IBM), Concordia (Mitsubishi), Grasshopper (IKV), Voyager (Object Space). Porównanie to pozwoli nam na zaobserwowanie różnic pomiędzy systemami agentowymi. Materiał przedstawiony poniżej oparty jest o dokumentację i bardzo ograniczoną liczbę eksperymentów i nie powinien być traktowany jako ostateczny wyznacznik zachowania danego systemu. Równocześnie możliwym jest, że w najnowszych wersjach oprogramowania nastąpiły zmiany powodujące, że część z poniższych stwierdzeń jest już nieprawdziwa. Wydaje się, że nawet jeśli tak się stało, nie obniża to wartości poznawczej poniższego porównania.

6.1 Java

Wszystkie cztery systemy (jak i zdecydowana większość istniejących systemów agentowych) zaimplementowane są w języku Java. Jednakże tylko Grasshopper i Voyager oparte są na najnowszej wersji języka Java. W chwili obecnej wiadomym jest, że nie powstanie nowa wersja Aglets, albowiem IBM porzuciło ten projekt (trwają natomiast prace nad otoczeniem agentowym, które będzie jednakże niekompatybilne z Aglets). Jako interesującą ciekawostkę warto dodać, że w roku 2001 spotkałem na konferencji współtwórców Concordii, którzy poinformowali mnie, że Mitsubichi nie jest zainteresowane dostosowaniem Concordii do nowej wersji języka Java.

6.2 Tworzenie agentów na odległość

Trzy z omawianych systemów Voyager, Grasshopper i Aglets umożliwiają tworzenie agentów na innym komputerze przez przesłanie tylko i wyłącznie niezbędnych parame-

trów. Concordia nie ma takiej możliwości. Tak więc w przypadku Concordii, agenci muszą być tworzeni centralnie i muszą przemieszczać się do miejsca, gdzie mają być wykorzystani.

6.3 Przemieszczalność obiektów

Voyager jest jedynym systemem, który umożliwia przesyłanie obiektów bez konieczności przemieszczania agenta transportującego obiekt. Pozwala to systemom budowanym przy pomocy Voyagera na ograniczenie wykorzystania sieci; przesyłana jest bowiem mniejsza ilość informacji.

6.4 Klonowanie agentów

Wszystkie cztery systemy umożliwiają agentowi stworzenie swojej własnej kopii. Pozwala to na zaimplementowanie agentowej formy znanego podejścia do programowania równoległego – model: jeden program wiele danych (ang. single program multiple data; SPMD).

6.5 Przemieszczalność agentów

Rozróżniane są dwa typy przemieszczalności agentów. Słaba przemieszczalność (ang. weak mobility) oznacza, że agent przemieszcza się razem z potrzebnymi danymi, ale nie następuje przemieszczenie stanu agenta. Silna przemieszczalność (ang. strong mobility) oznacza, że również stan agenta zostaje przesłany i agent może kontynuować pracę jak gdyby nie zmienił lokalizacji. Żaden z porównywanych czterech systemów agentowych nie zapewnia silnej przemieszczalności (jednym z niewielu systemów, który daje taką możliwość jest system NOMADS powstały w University of West Florida).

6.6 Mechanizmy komunikacji

Komunikacja między agentami może mieć trzy formy. Komunikacja synchroniczna wymaga potwierdzenia otrzymania wiadomości przez odbiorcę zanim agent nadawca będzie mógł kontynuować pracę. W przypadku komunikacji asynchronicznej agent nadawca wysyła wiadomość i kontynuuje pracę. Wszystkie cztery systemy agentowe umożliwiają oba rodzaje komunikacji. Dodatkowo, Voyager i Grashopper umożliwiają komunikację dynamiczną. Jest to asynchroniczna komunikacja, w której wiadomość podąża za mobilnym agentem.

6.7 Współpraca agentów

Ze wszystkich systemów Concordia posiada najlepsze mechanizmy wspomagania współpracy agentów. Agenci Concordii tworzą grupy, które używają rozproszonych obiektów (dostępnych w języku Java) w celu wymiany informacji. W przypadku otoczenia Aglets agenci mogą negocjować i ustalać terminy „zebrań”. Grasshopper

i Vayager nie posiadają specjalnych mechanizmów wspomagania współpracy między agentami.

6.8 Usuwanie śmieci (ang. garbage collection)

Voyager usuwa agentów i związaną z nimi informację natychmiast po zaniknięciu ostatniego odwołania do danego agenta. Agenci w systemie Grasshopper istnieją do momentu wykonania komendy samozniszczenia do momentu, gdy zostaną usunięci przez inny element oprogramowania (np. innego agenta) lub przez użytkownika. W przypadku Aglets i Concordii agenci muszą zostać usunięci przy pomocy odpowiedniej komendy systemowej.

7 Uwagi końcowe

W powyższej pracy przedstawiliśmy podstawowe informacje związane ze stosowaniem agentów programowych jako narzędzia tworzenia oprogramowania. Staraliśmy się naszkicować obiektywny obraz rzeczywistości, nie poddający się przesadnemu optymizmowi proroków agentyzmu a równocześnie nie popadający w przesadny negatywizm krytyków perspektyw technologii agentowych. W chwili obecnej wydaje się, że wprawdzie krytyka zaprezentowana przez Nwanę i Ndumu była odrobinę przesadzona, to właśnie oni mieli rację w swoim programie pozytywnym. Koniecznym było (i jest) przystąpienie do tworzenia rzeczywistych zaimplementowanych systemów agentowych. Takie systemy zaczynają właśnie powstawać i być może już niedługo naprawdę agent osobisty będzie nas witał, gdy włączymy rano komputer. I być może rzeczywiście już niedługo będziemy budować oprogramowanie dokonując dekompozycji problemu na komunikujących się ze sobą, współpracujących agentów.

Bibliografia

[ALT2001] J. Altmann, F. Gruber, L. Klug, W. Stockner i E. Weippl, *Using Mobile Agents in real World: A Survey and Evaluation of Agent Platforms, Proceedings of the Second International Workshop on "Infrastructure for MAS, and Scalable MAS,"*, Montreal, Canada , May 28 – June 01 2001.

[ANG2002] R. Angryk, V. Galant i M. Paprzycki, *Travel Support System – an Agent-Based Framework, Proceedings of the International Conference on Internet Computing (IC'02)*, CSREA Press, Las Vegas , 2002, 719-725.

[TBL2001] T. Berners-Lee, J. Hendler i O. Lassila, *The Semantic Web Scientific American*, May 2001, http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21.

[BOO1994] G. Booth, *Object--oriented Analysis and Design with Applications*, Addison Wesley, 1994.

[BRE1998] W. Brenner, R. Zarnekow, H. Wittig i C. Schulbert, *Intelligent Software Agents*, Springer-Verlag, 1998.

[GAL2001] V. Galant i J. Tubyrcy, *Inteligentny Agent Programowy*, Prace Naukowe AE Wrocław, 2001, 45-57.

[GRI1999] M.L. Griss, *My Agent Will Call Your Agent ... But Will It Respond?*, Hewlett Packard, 1999, http://www.hpl.hp.com/techreports/1999/HPL-1999-159.pdf.

[JEN2001] N.R. Jennings, *An agent-based approach for building complex software systems*, Communications of the ACM, 44(4), 2001, 35-41.

[HEN1999] J. Hendler, *Is There an Intelligent Agent in Your Future?*, Nature, 11, March 1999.

[LIU2001] J. Liu, *Autonomous Agents and Multi-agent Systems*, World Scientific, 2001.

[MAE1994] P. Maes, *that Reduce Work and Information Overload*, Communications of the ACM, 37(7), 1994, 31-40.

[NWA1999] H. Nwana i D. Ndumu, *A Perspective on Software Agents Research*, The Knowl-edge Engineering Review, 14(0), 1999, 1-18.