Architektura układu sterującego robotem mobilnym w systemie SOMRS

Grzegorz Terlikowski*, Waldemar Bartyna**

*Instytut Informatyki, Uniwersytet Przyrodniczo-Humanistyczny, Siedlce
**Instytut Podstaw Informatyki, Polska Akademia Nauk, Warszawa

Streszczenie: Zaproponowano nowe podejście (architekturę) do budowy układu sterowania pojedynczym robotem mobilnym. Bazuje ono na paradygmacie SOA, w którym robot widziany jest jako zbiór świadczonych przez siebie usług. W informatyce paradygmat SOA jest uznanym i często stosowanym podejściem do projektowania rozproszonych systemów. W robotyce takim systemem jest niewątpliwie system wielorobotowy. Próba przeniesienia paradygmatu SOA w obszar robotyki ma na celu wykazanie przydatności tego podejścia w robotyce mobilnej. Zaproponowana architektura układu sterującego robotem mobilnym składa się z czterech warstw programowych. Najniższa warstwa, tj. warstwa kontroli urządzeń i agregacji danych, odpowiedzialna jest za kontrolę urządzeń (sensorów, manipulatorów itp.), w które wyposażony jest robot oraz za agregację, przetwarzanie i fuzję pozyskanych z nich danych. Funkcje kolejnej warstwy nawigacji, zwykle implementowane są przez system nawigacyjny robota, który umożliwia m.in. sprawne wyznaczanie i pokonywanie tras. Kontrolery wykonania usług rezydują w warstwie logiki wykonania usług i są odpowiedzialne za realizację poszczególnych usług świadczonych przez robota. W najwyższej warstwie, tj. warstwie zarządzania usługami, znajduje się Menadżer Usług, odpowiedzialny za komunikację systemu robota z pozostałymi komponentami systemu SOMRS oraz za zarządzanie wykonaniem usług na robocie. Na podstawie opracowanej architektury powstał prototyp systemu robota, który został zainstalowany na dwóch robotach typu Pioneer P3-DX. Eksperymenty z udziałem tych robotów pozwoliły na weryfikację przydatności opracowanej architektury w praktycznych zastosowaniach.

Słowa kluczowe: SOA, architektura, usługa, menadżer usług, znaczniki, układ sterujący, mapa obiektowa, SLAM

DOI: 10.14313/PAR_204/118

Proponowana architektura układu sterującego robotem mobilnym powstała jako rezultat cząstkowy projektu o nazwie Robo-enT [1]. Celem tego projektu było opracowanie odpowiednich technologii informacyjnych potrzebnych do zrealizowania idei inteligentnego środowiska (ang. *ambient intelligence*) rozumianego jako SOMRS (*Service-Oriented Multirobot System*).

1. Wprowadzenie

W ostatnim czasie, w środowiskach akademickich, można zauważyć rosnące zainteresowanie systemami wielorobotowymi, gdzie inteligencja (planowanie i kontrola wykonania złożonych zadań) jest często przeniesiona „w górę" poza system pojedynczego robota. Obecnie w informatyce paradygmatem do tworzenia rozproszonych systemów (którym niewątpliwie jest system wielorobotowy) jest architektura zorientowana na usługi SOA (ang. *Service Oriented Architecture*). Paradygmat ten główny nacisk kładzie na definiowanie interfejsów, które ukrywają szczegóły implementacyjne oprogramowania (tzw. usług), dzięki czemu w łatwy sposób można integrować ze sobą różne heterogeniczne środowiska. SOA wnosi do istniejących systemów wielorobotowych nowe spojrzenie – pojedynczy robot (lub inne urządzenie) postrzegany jest jako zbiór usług, które może udostępniać innym komponentom systemu wielorobotowego i w ten sposób współdziałać przy wykonywaniu złożonych zadań.

Podejście usługowe w robotyce jest coraz bardziej popularne, a paradygmat SOA doczekał się kilku znaczących rozszerzeń takich jak: OASIS Devices Profile for Web Services (DPWS) [9] i Service Oriented Device Architecture (SODA) [6]. Podejścia te odnoszą się do systemów budowanych z podłączonych do Internetu urządzeń.

Proponowane podejście do budowy układu sterującego pojedynczym robotem mobilnym jest oparte na paradygmacie SOA. Pojedynczy robot jest tutaj częścią systemu wielorobotowego (SOMRS), w ramach którego świadczy usługi (wykonuje zadania).

2. Podstawowe założenia

Podczas prac nad architekturą układu sterującego robotem mobilnym działającym w ramach systemu SOMRS wzięto pod uwagę następujące założenia:

1. Robot powinien być postrzegany przez system SOMRS nie jako inteligentne urządzenie (którym w istocie jest), lecz jako zbiór usług wykonywanych na żądanie. System pojedynczego robota **nie powinien realizować z własnej inicjatywy złożonych zadań**, lecz wykonywać swoje usługi tylko wtedy, gdy jest takie zapotrzebowanie w SOMRS. Dzięki takiemu podejściu, z punktu widzenia całego systemu SOMRS nie jest ważne, jakie urządzenie wykona daną usługę. W przypadku niektórych usług może to być zarówno robot mobilny, stacjo-

narny manipulator, czy też zespół czujników (np. dalmierzy laserowych).

2. System robota powinien zapewniać **łatwe rozszerzanie** go o **możliwość świadczenia kolejnych usług**. Każdy moduł sterujący wykonaniem danej usługi przez system robota powinien implementować zestaw wymaganych interfejsów programowych. Dzięki takiemu podejściu, do systemu robota można w stosunkowo łatwy sposób dodać kolejne typy usług, jak również je od niego odłączyć (np. by wykonać testy związane z zastąpieniem aktualnej usługi jej nowszą wersją).
3. Robot jako **inteligentne**, **kognitywne** i **autonomiczne** urządzenie powinien samodzielnie **ocenić wykonywalność** zadania oraz jego czasochłonność. Wykonanie zadania należy tutaj rozumieć jako realizację usługi w oparciu o parametry zawarte w treści zadania. Kontrola/monitorowanie wykonania zadania również leży w gestii systemu robota. System ten powinien planować wykonanie zleconego mu zadania oraz reagować na nieoczekiwane sytuacje związane z tym wykonaniem. Ponadto, powinien **samodzielnie pozyskiwać informacje**, które są mu potrzebne do realizacji zadań. Informacje te, mogą być uzyskiwane zarówno z sensorów (wyposażenie robota), jak również mogą być pobrane ze specjalnego komponentu systemu SOMRS, pełniącego funkcję repozytorium map obiektowych.
4. Projektant systemu robota może **wykorzystać** dowolne **istniejące podejścia** przy rozwiązywaniu dobrze znanych problemów takich jak **przetwarzanie** i **fuzja danych** z sensorów, **planowanie tras**, czy reaktywne **omijanie przeszkód**. W tym obszarze projektant systemu robota może wykorzystać dowolne istniejące rozwiązanie lub pokusić się o implementację własnych algorytmów. Z uwagi na taką ogólność proponowanej architektury, w pewnym sensie można ją postrzegać jako dopełnienie lub rozwinięcie istniejących architektur, a nie jako ich alternatywę.

Oprócz powyższych założeń, system robota (jako komponent systemu Robo-enT) musi wykorzystywać hierarchiczną obiektową reprezentację środowiska(wspólną dla wszystkich komponentów systemu) [7] oraz obsługiwać specjalne protokoły komunikacyjne [5].

3. Architektura układu sterującego robotem mobilnym

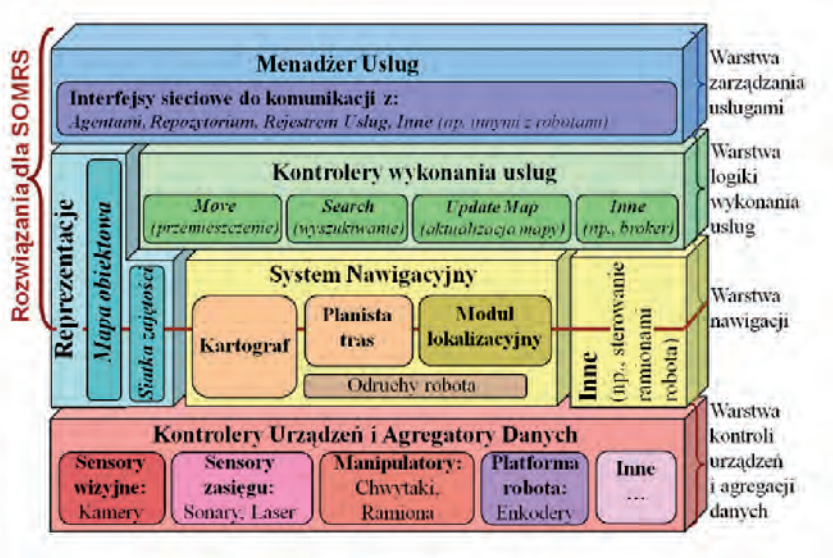
Przedstawiona na rys. 1 architektura reprezentuje podejście do budowy układu sterującego robotem mobilnym od strony technologii informacyjnych, dlatego też podzielona została na warstwy programowe, ze względu na zaimplementowane w nich funkcjonalności. Podział ten, różni się od tradycyjnych podejść, gdzie warstwy są rozpatrywane w kontekście deliberatywności i reaktywności systemu robota. Patrząc w tradycyjny sposób, na proponowaną architekturę, granica deliberatywno-reaktywna wypada gdzieś po środku warstwy nawigacji. Ponadto, cechy deliberatywno-reaktywne mogą wykazywać niektóre moduły zlokalizowane w najniższej warstwie.

Przedstawione na rys. 1 zbiory tzw. kontrolerów wykonania usług oraz modułów z warstwy kontroli urządzeń i agregacji danych, odnoszą się do prototypowego systemu. Zbiory te są ograniczone możliwościami konkretnego robota (wliczając w to jego niestandardowe wyposażenie), dla którego powstał prototyp (tutaj Pioneer P3-DX). Dla innego rodzaju robotów mogłyby być zupełnie inne.

Powyżej **grubej brązowej linii** (rys. 1) znajdują się rozwiązania przeznaczone dla robotów mobilnych będących częścią systemu SOMRS. Rozwiązania te zostały zastosowane w prototypowym układzie sterującym robotem mobilnym i wydają się stanowić dobrą podstawę do budowy układów sterujących dla wszystkich robotów świadczących usługi w systemie SOMRS.

Interesującym rozszerzeniem systemu robota jest implementacja kontrolera wykonania specjalnej **usługi brokerskiej**, który potrafiłby, oprócz sterowania lokalnym robotem, koordynować też działania innego robota (nieustannie się z nim komunikując), realizując w ten sposób ścisłą współpracę obu urządzeń. Usługa brokerska byłaby widziana przez system SOMRS jako zwykła usługa.

Dopuszczalne jest również wyposażenie systemu robota w umiejętność definiowania (w pewnych sytuacjach) zadań do realizacji przez system SOMRS. Dzięki temu robot mógłby np. zlecić systemowi SOMRS otworzenie drzwi (widzianych jako pewna usługa) w momencie, kiedy chciałby przez nie przejechać.



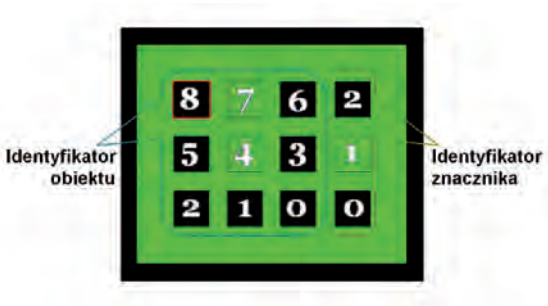
Rys. 1. Architektura układu sterującego robotem w SOMRS
Fig. 1. A robot control system architecture in SOMRS

4. Warstwa kontroli urządzeń i agregacji danych

Warstwa ta zawiera moduły odpowiedzialne za kontrolowanie pracy sensorów, manipulatorów robota oraz za przetwarzanie i fuzję danych przez nie dostarczonych. Wraz z warstwą nawigacji jest ona najbardziej podatna na wykorzystanie w niej znanych rozwiązań(pkt 4 założeń).

4.1. Przetwarzanie obrazu i rozpoznawanie obiektów

Przykładowym komponentem tej warstwy jest moduł obsługujący kamerę. Jego zadaniem jest m.in. przetwarzanie obrazu w celu rozpoznania na nim obiektów. W prototypowym systemie obiekty są rozpoznawane pośrednio przez wykrycie i rozpoznanie przyklejonych do nich znaczników. Znaczniki znacznie upraszczają proces rozpoznawania obiektów, co pozwala skupić się na innych aspektach systemu. W testowym środowisku znaczniki mają postać prostokątnych naklejek. Każdy znacznik koduje identyfikator obiektu w formie graficznych bitów (rys. 2).



Rys. 2. Schemat znacznika
Fig. 2. A schema of a marker

Pełny graficzny identyfikator składa się z dwóch sekcji:
- **identyfikatora obiektu**, który jednoznacznie identyfikuje dany obiekt w mapie obiektowej. Ustawiony ósmy graficzny bit oznacza, że obiekt jest **obiektem statycznym**, czyli takim, którego położenie nie zmienia się w czasie lub ta zmiana jest mało prawdopodobna. Przykładami takich obiektów są ściana i szafa.
- **identyfikatora znacznika**, który jest jego numerem na obiekcie. Każdy obiekt może mieć kilka (w prototypowym systemie do ośmiu) znaczników, które różnią się swoim położeniem na obiekcie (opisanym w mapie obiektowej).

Dzięki odczytanemu identyfikatorowi prototypowy system robota może odwołać się do komponentu systemu SOMRS o nazwie ***Repozytorium***, w celu pozyskania bardziej szczegółowych informacji na temat obiektu. Rozpoznanie znacznika jest równoznaczne z rozpoznaniem obiektu, do którego jest przyklejony.

Oprócz identyfikatora, z przetworzonego obrazu są pozyskiwane informacje, które są wykorzystywane w procesie lokalizowania się robota (podrozdział 5.1). Do takich informacji należą:
- *odległość* obiektywu kamery od środka znacznika – obliczana na podstawie długości krawędzi pionowych znacznika uzyskanych z przetworzonego obrazu oraz znajomości ich rzeczywistych wymiarów,
- *odchylenie* osi kamery od pozycji prostopadłej do płaszczyzny znacznika – obliczane na podstawie proporcji długości obu krawędzi bocznych znacznika wykrytego podczas przetwarzania obrazu,
- *przesunięcie* środka znacznika w stosunku do punktu przecięcia osi kamery z prosta prostopadłą do niej, przechodzącą przez środek tego znacznika – obliczane na podstawie umiejscowienia znacznika na zdjęciu oraz długości krawędzi poziomych znacznika.

Należy pamiętać, że przedstawiony sposób rozpoznawania obiektów jest przykładowy i może zostać zupełnie inaczej zrealizowany w innej implementacji systemu.

4.2. Ekstrakcja cech środowiska z odczytów lasera

Innym komponentem warstwy kontroli urządzeń i agregacji danych w prototypowym systemie jest moduł odczytujący i przetwarzający dane z dalmierza laserowego. Dane pozyskane z tego urządzenia są przetwarzane, w celu wyszukania cech szczególnych otoczenia. Cechami szczególnymi są wykryte linie proste o pewnej długości i ich połączenia (pod kątem zbliżonym do prostego). Linie reprezentują krawędzie ścian i szaf, natomiast połączenia linii – ich kąty i rogi. Do wykrywania linii został wykorzystany znany ze statystyki algorytm *regresji liniowej* [11] (ang. *Linear Regression*). Cechy szczególne otoczenia są wykorzystywane przez system nawigacyjny robota w procesach lokalizowania się i budowy map otoczenia.

5. Warstwa nawigacji

Główną funkcją modułów zlokalizowanych w tej warstwie jest sterowanie urządzeniami, w które wyposażony jest robot, z uwzględnieniem ograniczeń środowiska rzeczywistego. Takie sterowanie zwykle wymaga współpracy kilku urządzeń (głównie manipulatorów i sensorów).

W przypadku robota mobilnego modułem, który nieodzownie wpisany jest w funkcjonalność tej warstwy jest **system nawigacyjny**. System ten, w celu sprawnego sterowania ruchem platformy robota wykorzystuje dane pozyskane z sensorów robota. Dzięki temu robot podczas przemieszczenia może sprawnie omijać napotkane przeszkody.

Głównymi zadaniami realizowanymi przez każdy system nawigacyjny robota mobilnego są:
- *lokalizowanie się* – ustalenie bieżącej pozycji robota w wybranym układzie odniesienia,
- *reprezentacja środowiska* – polega na tworzeniu obrazu (mapy) otoczenia, ewentualnie na nanoszeniu zaobserwowanych zmian na posiadaną mapę,
- *planowanie tras* – wyznaczenie przejazdu od bieżącej pozycji robota do docelowej,
- *realizacja trasy* – podążanie robota po zaplanowanej trasie.

W kontekście reaktywności i deliberatywności, system nawigacyjny w proponowanej architekturze można postrzegać jako klasyczny dwu- lub trójwarstwowy system robota mobilnego, który w warstwie deliberatyw-

nej realizuje m.in. takie wysokopoziomowe cele, jak planowanie tras, lokalizacja na podstawie obiektów, eksploracja środowiska w poszukiwaniu obiektów i pozycjonowanie się względem obiektów. Ponadto, w proponowanej architekturze system nawigacyjny robota można postrzegać jako zasób wykorzystywany do realizacji zadań w warstwie logiki wykonania usług. W porównaniu do istniejących rozwiązań, opisywany system nawigacyjny wnosi nowe elementy w postaci sposobu planowania tras na podstawie mapy obiektowej oraz lokalizowania się na podstawie rozpoznanych obiektów.

Innym przykładowym modułem (niezaimplementowanym w prototypie systemu) zlokalizowanym w tej warstwie może być system sterujący ruchem ramienia (lub ramion) robota podczas chwytania przez nie jakiegoś przedmiotu. Wykonanie tej operacji również wymaga pewnej wiedzy o stanie środowiska, która powinna być uwzględniona przy operacji chwytania. Między ramieniem a chwytanym przedmiotem może zostać wykryta przeszkoda, którą ramię (często o wielu stopniach swobody) powinno ominąć. Chwytanie przedmiotu przy użyciu dwóch ramion o wielu stopniach swobody wymaga zaawansowanego (wyrafinowanego) systemu sterowania, który zapewnia ich synchronizację w czasie i koordynację w przestrzeni. Moduł sterujący ruchem ramienia/ramion można postrzegać jako system nawigacyjny tego ramienia/ramion.

5.1. Lokalizowanie się robota

Aby robot mógł sprawnie świadczyć usługi w swoim środowisku, musi być zlokalizowany. Innymi słowy musi być świadom swojego położenia względem jakiegoś umownego (znanego przez niego) układu odniesienia. Proces lokalizowania się robota jest zadaniem automatycznie realizowanym przez jego system, zaraz po uruchomieniu.

W prototypowym systemie proces lokalizowania się przebiega na dwóch poziomach. Na niższym poziomie system nawigacyjny wykorzystuje dobrze znany z literatury SLAM (ang. *Simultaneous Localization And Mapping*) [4], aby korygować błędy układu odometrycznego, natomiast na wyższym poziomie wykorzystywane są położenia znaczników obiektów w mapie obiektowej. W procesie lokalizowania się opartym na SLAM, wykorzystywane są cechy środowiska (kąty ścian, rogi szaf itd.) wykryte i udostępnione przez moduł najniższej warstwy.

Proces wizyjnej lokalizacji z wykorzystaniem map obiektowych składa się z następujących etapów:

1. *poszukiwanie obiektu* statycznego (dokładnie jego znacznika) w środowisku,
2. *pozycjonowanie się* względem znalezionego obiektu (jego znacznika),
3. *określenie przekształcenia* między globalnym układem współrzędnych robota i układem współrzędnych lokalizacji (pomieszczenia), w której znaleziono obiekt.

Etap poszukiwania obiektu statycznego, polega na sfotografowaniu tych obszarów środowiska, w obrębie których została wykryta (podczas przetwarzania danych z dalmierza laserowego) linia prosta. Wykrycie w skanie laserowym linii, potencjalnie oznacza wykrycie sygnatury ściany lub szafy, a więc obiektu statycznego. Pozycjonowanie (czyli drugi etap) tylko względem takich obiektów jest wiarygodne. W procesie pozycjonowania wykorzystywane są dane uzyskane w czasie przetwarzania obrazu, na którym wykryto znacznik (podrozdział 4.1). Celem pozycjonowania jest prostopadłe ustawienie kamery robota względem znacznika przy jednoczesnym zachowaniu minimalnej odległości od niego. Jest to niezbędne, aby otrzymać jak najdokładniejsze dane dotyczące pozycji robota względem wykrytego znacznika.

Ze względu na trudności, związane z dokładnym określeniem pozycji robota względem znacznika, wynikające z odbić światła, niezbędne okazało się potwierdzenie pozycji znacznika na sekwencji dwóch kolejnych zdjęć. Zdjęcia te są wykonane w trakcie zbliżania się platformy robota do znacznika.

Po zakończeniu etapu pozycjonowania system robota, na podstawie wiedzy o swojej pozycji względem znacznika oraz pozycji znacznika w mapie obiektowej (dane są dostępne w Repozytorium), oblicza zależność (przekształcenie) między własnym globalnym układem współrzędnych a układem współrzędnych pomieszczenia (elementu mapy obiektowej), w którym się znajduje i w którym został wykryty znacznik. Przekształcenie to jest wykorzystywane (podczas świadczenia usług przez robota) do odwzorowań położeń obiektów znanych z mapy obiektowej na położenia w układzie współrzędnych robota i odwrotnie.

5.2. Planowanie tras

Planowanie tras również odbywa się na dwóch poziomach, przy użyciu dwóch reprezentacji środowiska.

- Na niższym poziomie system nawigacyjny wykorzystuje budowaną, przez wewnętrzny moduł kartograficzny, siatkę zajętości (ang. *occupancy grid*). Na jej podstawie wyznaczana jest lokalna trasa, np. od drzwi do drzwi tego samego pomieszczenia. Metodą służącą do wyznaczania trasy jest transformata dystansów [5] (ang. *distance transform*).
- Na wyższym poziomie wykorzystywana jest mapa obiektowa, umożliwiająca wyznaczenie globalnej trasy między różnymi pomieszczeniami. Do tego celu wykorzystywany jest hierarchiczny charakter map obiektowych oraz tzw. **obiekty łączące** (drzwi, przejścia). Obiekty łączące należą jednocześnie do kilku (zwykle dwóch) obiektów złożonych (pokoje, korytarze itp.). Elementami takiej trasy są kolejne obiekty łączące, przez które robot musi przejechać, aby się przemieścić między pomieszczeniem początkowym a docelowym.

Obydwie metody wyznaczania trasy nawzajem się uzupełniają. W celu bezpiecznego podążania po wyznaczonej trasie system nawigacyjny robota wykorzystuje reaktywne zachowania w działaniu zbliżone do metodologii pól potencjałowych [3, 10] (ang. *potential fields*).

6. Warstwa logiki wykonania usług

Warstwa ta zawiera moduły odpowiedzialne za realizację usług świadczonych przez robota i jest kluczowa dla prezentowanej architektury. Moduły te są implementowane w postaci tzw. **kontrolerów wykonania usług**, które

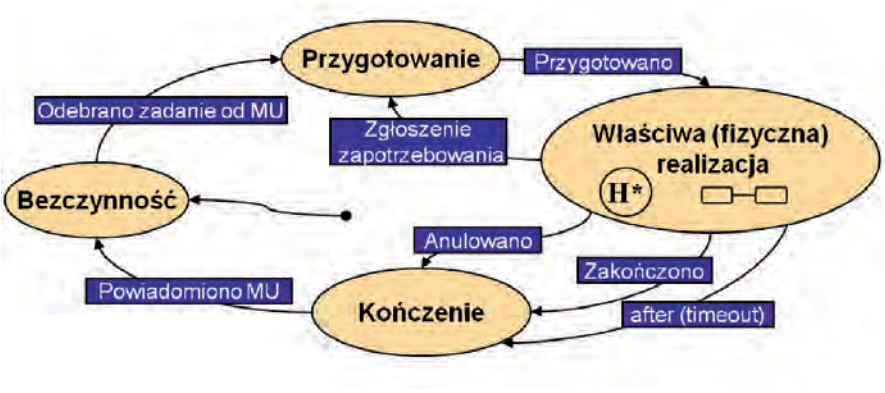
zawierają przepisy (ogólne plany) wykonania poszczególnych usług przez robota. Pojedynczy kontroler jest odpowiedzialny za wykonanie pojedynczego typu usługi. Do realizacji usługi wykorzystywane są funkcjonalności dostarczone przez moduły warstw niższych, tj. *warstwy nawigacji* i *warstwy kontroli urządzeń i agregacji danych*.

Prototypowy system robota został zaimplementowany dla robota typu Pioneer P3-DX. Ograniczenia sprzętowe tych robotów pozwoliły na zaimplementowanie na nim jednej usługi fizycznej – ***move*** (przemieszczenie) oraz dwóch usług kognitywnych – ***search*** (wyszukiwanie) oraz ***update map*** (aktualizacja mapy). Poniższy opis będzie się koncentrował na sposobie realizacji tych właśnie usług.

6.1. Ogólny schemat wykonania usługi

W trakcie realizacji zleconego zadania (czyli realizacji usługi) kontroler wykonania usługi zajmuje się przeprowadzeniem systemu robota przez skończony zbiór jego stanów. Zmiana stanu następuje przez uruchomienie przez kontroler kolejnej funkcji realizującej pożądaną czynność (lub czynności) na danym etapie wykonania zadania. Działanie kontrolerów, podczas wykonywania zadania, można przedstawić w postaci automatów skończonych. Przedstawiony na rys. 3 automat jest ogólny dla wszystkich usług i składa się czterech stanów.

W stanie **Bezczynność** kontroler wykonania usługi oczekuje na zadanie od Menadżera Usług (MU). Otrzymanie zadania, powoduje przejście do kolejnego stanu kontrolera, którym jest **Przygotowanie**. Na tym etapie kontroler przygotowuje się do rozpoczęcia właściwej (fizycznej) realizacji usługi. Przygotowanie zwykle obejmuje aktualizację odpowiedniego fragmentu lokalnej (utrzymywanej przez system robota) mapy obiektowej oraz tworzenie globalnych planów (np. przeszukiwania tras) na podstawie tej mapy. Aktualizacja mapy jest konieczna, aby system robota wykonywał usługę w oparciu o aktualną wiedzę o środowisku. W zależności od potrzeb i sytuacji kontroler może ponownie znaleźć się w tym stanie, np. aktualizując kolejny fragment mapy obiektowej, który jest mu potrzebny do dalszej (patrz H* – głęboka historia na rys. 3) realizacji usługi.



Rys. 3. Schemat przedstawiający ogólne wykonanie usług
Fig. 3. A service realization schema

Przebieg **Właściwej realizacji** usług świadczonych przez robota jest indywidualny dla każdej z nich z osobna, dlatego na rys. 3 stan ten został przedstawiony jako ***stan złożony***. Znaczy to, że jest on rozwijany przez konkretne kontrolery usług (tj. *search*, *updateMap* i *move*). Na etapie Realizacji, wykorzystywane są funkcje warstw niższych, które realizują poszczególne czynności robota i tym samym przyczyniają się do zmiany stanu jego systemu. Rezultat wykonania danej czynności, przez funkcje, która zakończyła swoje działanie (i powiadomiła o tym kontroler), jest analizowany przez kontroler w celu podjęcia decyzji, co do kolejnego stanu robota (tj. jaką funkcję wywołać). W przypadku, gdy wyniki realizacji czynności przez funkcje nie są zgodne z oczekiwaniami, kontroler usługi może posiadać alternatywne sposoby (plany) do osiągnięcia kolejnego pożądanego stanu. Może zostać wywołana funkcja, która poprzednio zawiodła (np. przy wyszukiwaniu obiektu), lecz tym razem z innymi wartościami jej argumentów. Gdy wszystkie sposoby na osiągnięcie kolejnego etapu wykonania zadania (pożądanego stanu) zawiodą, kontroler zaprzestaje realizować usługę. Wykonanie usługi może zostać również anulowane przez MU, który otrzymał takie zadanie od *Agenta*, lub też może upłynąć czas przeznaczony na jej realizacje. ***Agent*** jest specjalnym komponentem systemu SOMRS, zajmującym się wyszukiwaniem usług oraz zlecaniem im zadań do realizacji.

Po zakończeniu fizycznej realizacji usługi, kontroler przechodzi w stan **Kończenie**, gdzie są wykonywane operacje kończące. Operacjami tymi są: (1) powiadomienie MU o rezultacie wykonania zadania (tylko jeśli nie było przez niego anulowane) (2) jeżeli w trakcie realizacji zadania zaobserwowano lub spowodowano zmiany w środowisku, kontroler powiadamia (za pośrednictwem MU) o tym Repozytorium.

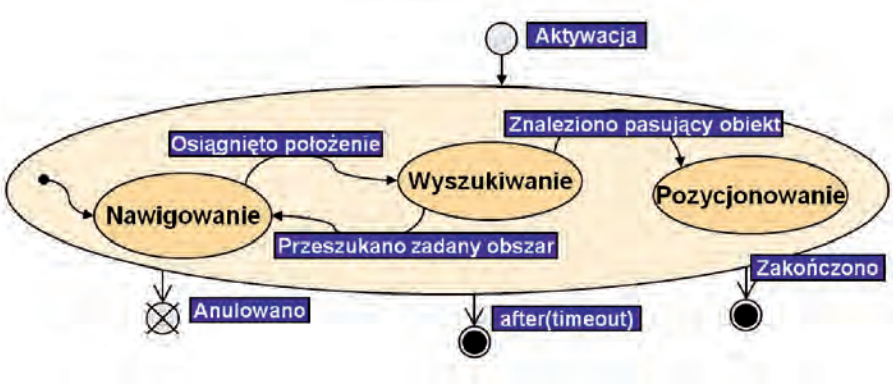
Po tych operacjach robot jako zasób jest zwalniany a MU może przekazać kolejne zadanie do realizacji do odpowiedniego kontrolera.

6.2. Wykonanie usługi *search*

Robot, realizując tę usługę, przeszukuje wskazany w zadaniu obszar lub zbiór obszarów w celu odnalezienia obiektu na podstawie jego ***nazwy***, ***wartości atrybutów*** lub na podstawie ***relacji***, w jakiej może on być z innym obiektem (lub obiektami).

Realizacja zadania (rys. 4) kończy się w chwili odnalezienia wyszukiwanego przedmiotu lub przeszukania zadanego obszaru. Po zakończeniu wykonania usługi *search*, system robota wysyła do *Agenta* (zlecającego zadanie) wiadomość zawierającą rezultat wyszukiwania.

Ze względu na zakres przeszukiwań można podzielić zadania realizowane przez tę usługę na dwie kategorie: *przeszukiwa-*



Rys. 4. Schemat przedstawiający wykonanie usługi search
Fig. 4. Search service realization schema

mapy obiektowej znajduje się poszukiwany obiekt. Pojedyncze pomieszczenie jest przeszukiwane w oparciu o mapę w postaci siatki zajętości. System robota wykonuje zdjęcia wszystkim przeszkodom znajdującym się na tej mapie w celu odnalezienia zadanego obiektu. Po bezowocnym przeszukaniu pomieszczenia, robot nawiguje do kolejnego pomieszczenia, będącego następnym elementem wcześniej utworzonej listy.

nie obszaru wokół obiektu oraz *przeszukiwanie całych lokalizacji* (pomieszczeń).

Przeszukiwanie obszaru wokół obiektu jest najmniej eksploracyjnym typem przeszukiwania. Zadania tego typu dotyczą przeważnie sprawdzenia tego, czy zadany obiekt znajduje się w bezpośrednim sąsiedztwie jakiegoś innego obiektu, np. *czy pudło znajduje się przy szafie*. W takim przykładowym zadaniu z mapy obiektowej odczytywane jest położenie wspomnianej szafy, a następnie rozpoczyna się proces **nawigowania** w kierunku tego położenia. Po osiągnięciu odpowiedniej odległości od szafy, system robota może (np. na podstawie kształtu szafy) zaplanować **wyszukiwanie** zadanego obiektu. W ten sposób może zostać wyznaczonych kilka podobszarów do przeszukania (np. jeden na każdą z widocznych ścian szafy). Po odnalezieniu zadanego obiektu, robot względem niego się **pozycjonuje**, wykorzystując te same funkcje, co w procesie lokalizowania się (podrozdział 5.1). Pozycjonowanie jest uzasadnione tym, że duża odległość i ostry kąt osi kamery w stosunku do znacznika, mogą powodować znaczące błędy w określeniu prawidłowego położenia znacznika (a zatem i obiektu) w globalnym układzie współrzędnych robota, i co za tym idzie, nieprawidłowe określenie relacji, w jakich znajduje się wyszukany obiekt z innymi obiektami. Prawidłowe określenie relacji jest kluczowe dla wykonania usługi **search**.

Przeszukiwanie całych lokalizacji jest bardzo czasochłonnym przedsięwzięciem. Może się wiązać z przeszukaniem zarówno pojedynczego, jak również zbioru pomieszczeń (np. należących do całego piętra). Robot wykonując tego typu zadanie tworzy plan składający się z listy pomieszczeń do przeszukania. Problem tworzenia takiej listy sprowadza się do klasycznego zagadnienia komiwojażera [2], które doczekało się wielu sposobów rozwiązania. Dobrym pomysłem wydaje się to, aby pierwszym elementem tej listy było pomieszczenie, wewnątrz którego wg

6.3. Wykonanie usługi *update Map*

Realizacja usługi **updateMap** (rys. 5.) polega na aktualizacji wartości dynamicznych atrybutów obiektów (takich jak ich położenie, relacje) znajdujących się we wskazanej w zadaniu lokalizacji lub zbiorze lokalizacji.

Usługę tę można postrzegać w pewnym sensie jako rozszerzenie usługi **search**. Również i w tym przypadku chodzi o przeszukanie pewnego obszaru lub zbioru obszarów. Różnica w działaniu obu typów usług jest jednak zasadnicza. Usługa **search** kończy swoje działanie, gdy opisany w zadaniu obiekt zostanie odnaleziony, niezależnie od tego, czy cały teren został dokładnie przeszukany. Natomiast usługa **updateMap** kończy swoje działanie dopiero wtedy, gdy zostanie przeszukany cały zadany obszar/zbiór obszarów. W trakcie wykonywania usługi **updateMap** są wyszukiwane wszystkie obiekty a następnie są sprawdzane relacje, w których występują te obiekty z innymi obiektami. W prototypowym systemie zostały zdefiniowane następujące kryteria określania relacji obiektów przez system robota:

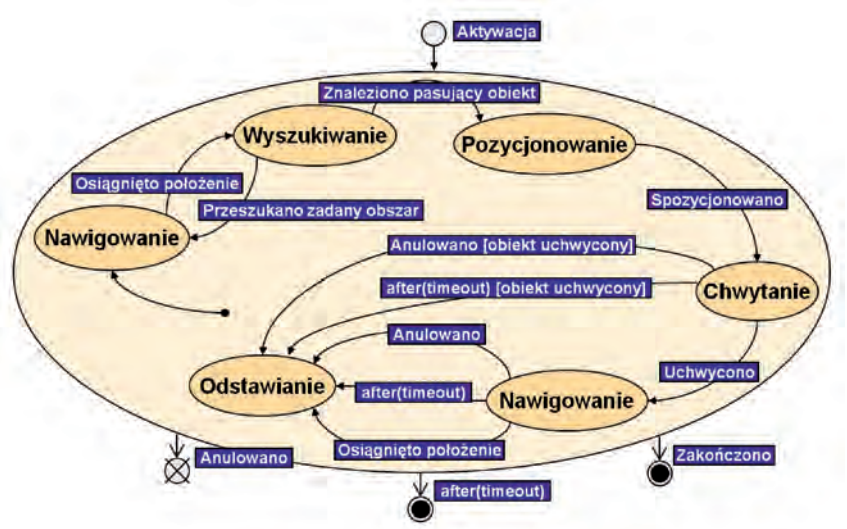
- **isAdiacentTo** – dwa obiekty są w relacji *isAdiacentTo* (przyleganie), jeśli odległość między tymi obiektami jest mniejsza od umownej wartości oraz obiekt będący pierwszym parametrem relacji jest dynamiczny,
- **isGluedWith** – dwa obiekty są w relacji *isGluedWith* (sklejenie), jeśli odległość między tymi obiektami jest mniejsza od umownej wartości oraz obydwa parametry tej relacji są statyczne,

Rys. 5. Schemat przedstawiający wykonanie usługi update-Map
Fig. 5. UpdateMap service realization schema

– **isOn** – dwa obiekty są w relacji *isOn* (ułożenie), jeśli jeden leży na drugim, np. pudło leży na podłodze lub komputer na biurku,
– **isIn** – robot stwierdza wystąpienie tej relacji między dwoma obiektami, jeśli sam jest w tej relacji z obiektem będącym jej drugim argumentem. Inaczej, robot jest w relacji *isIn* (należenie) z lokalizacją, w której aktualnie się znajduje. Po wyszukaniu obiektu w tej lokalizacji robot wnioskuje, że taka relacja zachodzi też między wyszukanym obiektem i daną lokalizacją.

Zbiór postrzeganych relacji jest ograniczony możliwościami związanymi z rozpoznawaniem obiektów i ich atrybutów oraz z możliwościami sprzętowymi robotów wykorzystanych w eksperymentach.

Wyniki realizacji usługi **updateMap** ostatecznie trafiają do komponentu *Repozytorium*.



Rys. 6. Schemat przedstawiający wykonanie usługi move
Fig. 6. Move service realization schema

6.4. Wykonanie usługi *move*

Najciekawszą usługą zaimplementowaną w prototypowym systemie jest usługa transportowa **move**. Sposób jej wykonania został przedstawiony na rys. 6.

Pierwszym etapem fazy wykonania usługi *move* jest **nawigowanie** do bieżącego położenia obiektu, którego dotyczy zadanie. Położenie to jest znane z mapy obiektowej. Po znalezieniu się robota w sąsiedztwie obiektu do przemieszczenia, następuje etap jego **poszukiwania**. Przeszukiwane jest tylko i wyłącznie sąsiedztwo położenia znanego z mapy obiektowej i nie musi zakończyć się sukcesem.

Należy zauważyć, że obiekt do przemieszczenia jest obiektem dynamicznym, tak więc jego położenie, które jest znane z mapy obiektowej nie musi być aktualne. W takim przypadku usługa *move* zakończy się niepowodzeniem, a system SOMRS będzie musiał zaaranżować i uruchomić usługę (lub usługi) poszukujące obiekt. Gdy to "dodatkowe" poszukiwanie zakończy się sukcesem, SOMRS powtórnie zaaranżuje i rozpocznie wykonie usługi move. Usługa wyszukiwania może zostać zrealizowana przez to samo urządzenie, które zajmuje się transportem obiektów, ale system robota nie podejmuje tej decyzji samodzielnie. Takie kompetencje ma tylko system SOMRS.

Po wyszukaniu obiektu do przemieszczenia, robot dokładnie **pozycjonuje** się względem niego. Po tym etapie następuje **chwycenie** obiektu i jego transport (**nawigowanie**) do położenia docelowego, gdzie następuje jego **odstawienie**.

Po odstawieniu obiektu kontroler wykonujący usługę przemieszczenia powiadamia o tym warstwę wyższą, a ona powiadamia system SOMRS. Sytuacje awaryjne związane z wykonywaniem usług również są zgłaszane do SOMRS.

7. Warstwa zarządzania usługami

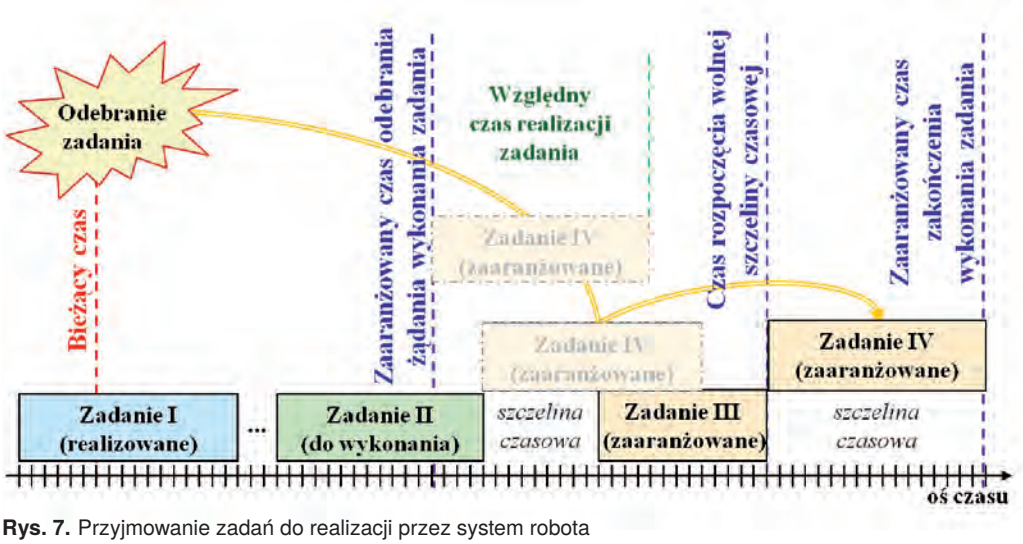
Warstwa ta odpowiedzialna jest za zarządzanie usługami świadczonymi przez system robota. Ponadto zapewnia ona komunikację systemu robota z pozostałymi komponentami systemu SOMRS. To tutaj są odbierane wszystkie wiadomości wysyłane przez komponenty SOMRS do systemu robota. Komponentem implementującym funkcjonalność tej warstwy jest Menadżer Usług (MU). Reaguje on na zapytania systemu SOMRS związane z **fazą aranżacji** oraz **fazą wykonania** poszczególnych zadań. Zarządzanie usługami w przypadku robotów mobilnych, działających w dynamicznych środowiskach rzeczywistych, jest wyzwaniem niebanalnym. W prototypowym systemie zostało to zrealizowane w sposób opisany poniżej i jest to przykładowe rozwiązanie tego problemu.

7.1. Rejestracja usług

Aby dana usługa była widoczna na zewnątrz, system robota musi ją zarejestrować w specjalnym komponencie systemu SOMRS o nazwie Rejestr Usług. Wpis rejestru zawiera **adres usługi** (adres IP robota i port, na którym nasłuchuje MU w komunikacji z resztą systemu), **zasięg** (obszar świadczenia usługi przez urządzenie), **nazwę akcji** (typ wykonywanej operacji) oraz jej **ograniczenia** (np. wymiary lub masa obiektów w przypadku usługi **move**). Taki wpis do rejestru jest jednocześnie definicją interfejsu usługi i jest dostarczany przez kontroler wykonania tej usługi.

7.2. Faza aranżacji – przyjmowanie zadań do realizacji

Zanim dojdzie do fizycznej realizacji jakiegokolwiek zadania (czyli świadczenia usługi) przez system robota musi nastąpić aranżacja tego zadania. Faza aranżacji jest etapem negocjacyjnym, w czasie którego ustalane są warunki wykonania zadania (m.in. końcowy czas jego realizacji). W tej fazie zadanie wysłane przez *Agenta* jest odbierane



Rys. 7. Przyjmowanie zadań do realizacji przez system robota
Fig. 7. Task scheduling in the robot control system

przez MU, a następnie przekazywane dalej do odpowiedniego kontrolera wykonania usługi. Wybór odpowiedniego kontrolera następuje na podstawie **nazwy akcji**, która jest parametrem zadania. Kontroler ten (niezależnie od tego, czy system robota świadczy w tym czasie jakąś usługę) jest odpowiedzialny za ocenę wykonalności zadania oraz za określenie przybliżonego **względnego czasu**, którego system robota będzie potrzebował na jego realizację. Każdy kontroler wykonania danej usługi ma własne algorytmy szacujące czas wykonania swojej usługi na podstawie parametrów otrzymanego zadania oraz ograniczeń środowiska. Odpowiedź kontrolera zawiera potwierdzenie wykonalności zadania oraz względny czas jego realizacji lub odmowę wykonania tego zadania.

Gdy kontroler odpowie pomyślnie, MU wyszukuje w utrzymywanej przez siebie kolejce (chronologicznie posortowanych pod względem wykonania) zadań (rys. 7), szczeliny czasowej, w której „zmieści się" **względny czas realizacji zadania** wyliczony przez kontroler. Czas rozpoczęcia szczeliny czasowej nie może być jednak wcześniejszy niż **zaaranżowany** (spodziewany) **czas odebrania żądania wykonania zadania**, będący parametrem tego zadania. Nowemu dodanemu do kolejki zadaniu nadawany jest status *zaaranżowane*. Spodziewany czas rozpoczęcia wykonania zadania jest to zatem maksimum z czasów: **zaaranżowany** (spodziewany) **czas odebrania żądania wykonania zadania** *i* **czasu rozpoczęcia** wyszukanej **wolnej szczeliny czasowej**. **Zaaranżowany** (spodziewany) **czas zakończenia realizacji wykonania zadania** jest sumą *spodziewanego czasu rozpoczęcia* wykonania zadania i względnego czasu realizacji zadania.

Odpowiedź wysyłana do *Agenta* zawiera informację, czy zadanie może być wykonane przez usługę na robocie i jeśli tak, to jaką wartość ma parametr **spodziewany czas zakończenia wykonania** tego **zadania**.

MU może utrzymywać w kolejce wiele zadań, których realizacja czasowo nie nakłada się na siebie. Brak możliwości równoległego wykonania różnych zadań jest spowodowany tym, że wszystkie zaimplementowane usługi pracują na tym samym zasobie, którym jest fizyczny robot. Nie można dopuścić więc do sytuacji, w której robot wykonuje kilka usług jednocześnie, gdyż kontrolery ich wykonania sterują m.in. ruchem platformy robota. Oczywisty jest fakt, ze robot fizycznie nie może znajdować się w dwóch miejscach jednocześnie, realizując dwa różne zadania.

Pomyślna odpowiedź kontrolera usługi w fazie aranżacji zadania nie oznacza jeszcze, że będzie ono wykonane. Usługa zlecona robotowi musi zostać wybrana przez *Agenta*. W prototypowym systemie SOMRS wybór następuje na podstawie **spodziewanego czasu zakończenia realizacji zadania** przez usługę zwróconego w czasie fazy aranżacji zadania. W przypadku wybrania przez *Agenta* usługi świadczonej przez dany system robota (o wyborze lub rezygnacji system robota jest informowany), następuje oczekiwanie na wiadomość rozpoczynającą fazę wykonania zadania lub jego odwołania.

7.3. Faza wykonania – uruchamianie usług

Po odebraniu przez MU wiadomości dotyczącej wykonania przyjętego na etapie aranżacji zadania, może nastąpić fizyczna jego realizacja, czyli wykonanie usługi. Bezpośrednio po odebraniu tej wiadomości przez MU, skojarzone z nią zadanie zyskuje nowy status – do wykonania (rys. 8). Samo wykonanie tego zadania może, ale nie musi, rozpocząć się natychmiast.

Żądanie wykonania zadania może zostać odebrane od *Agenta* przed czasem ustalonym w fazie aranżacji. Ta sytuacja może być spowodowana np. tym, że dany robot może wykonywać tylko część (pewne podzadanie) jakiegoś większego (złożonego) zadania podzielonego na etapy (podzadania) przez Menadżera Zadań (specjalny komponent systemu SOMRS). Etapy poprzedzające wykonanie danego zadania przez opisywany system robota mogą zakończyć się wcześniej niż to zostało ustalone na etapie aranżacji. Stąd *Agent* mógł wysłać żądanie wykonania podzadania do kolejnego systemu robota, ale nie może spodziewać się natychmiastowej jego reakcji. *Agent* musi wysłać do systemu robota żądanie wykonania danego zadania najpóźniej w czasie uzgod-

nionym w fazie aranżacji (w przeciwnym razie zadanie jest usuwane z kolejki przez MU).

Najpóźniej realizacja zadania (ze statusem *do wykonania*) rozpocznie się w czasie ustalonym na etapie aranżacji, czyli w ***spodziewanym czasie rozpoczęcia wykonania zadania***, ale może to nastąpić wcześniej (nawet natychmiast po otrzymaniu żądania wykonania zadania) – muszą być spełnione odpowiednie warunki. Wykonanie zadania o statusie *do wykonania* zostanie rozpoczęte tylko wtedy, gdy MU stwierdzi, że jego realizacja nie spowoduje kolizji z żadnym innym zadaniem znajdującym się w kolejce zadań – czyli zadaniem ze statusem *realizowane*, *do wykonania*, lub *zaaranżowane*. MU rozpatruje możliwość realizacji zadania z kolejki zadań bezpośrednio po trzech zdarzeniach:

1. Po odebraniu żądania wykonania zadania od Agenta i nadaniu mu statusu do wykonania. Wówczas (jeśli robot w danej chwili nie realizuje żadnej usługi) to zadanie jest brane pod uwagę przez MU jako kandydat do natychmiastowego wykonania.
2. Po odwołaniu zaaranżowanego przez Agenta zadania, znajdującego się na początku kolejki, lub przekroczeniu czasu oczekiwania na żądanie jego wykonania. Może to spowodować powstanie szczeliny czasowej, umożliwiającej wcześniejszą realizację innego zadania, oczekującego na wykonanie (tj. ze statusem *do wykonania*). W tym przypadku MU przeszukuje całą kolejkę zadań w celu wyszukania takiego zadania.
3. Po zakończeniu realizacji bieżącego zadania przez kontroler usługi (czyli zwolnieniu zasobu, którym jest robot). W tej sytuacji MU, podobnie jak w drugim przypadku, wyszukuje zadanie, które może zostać w danej chwili wykonane.

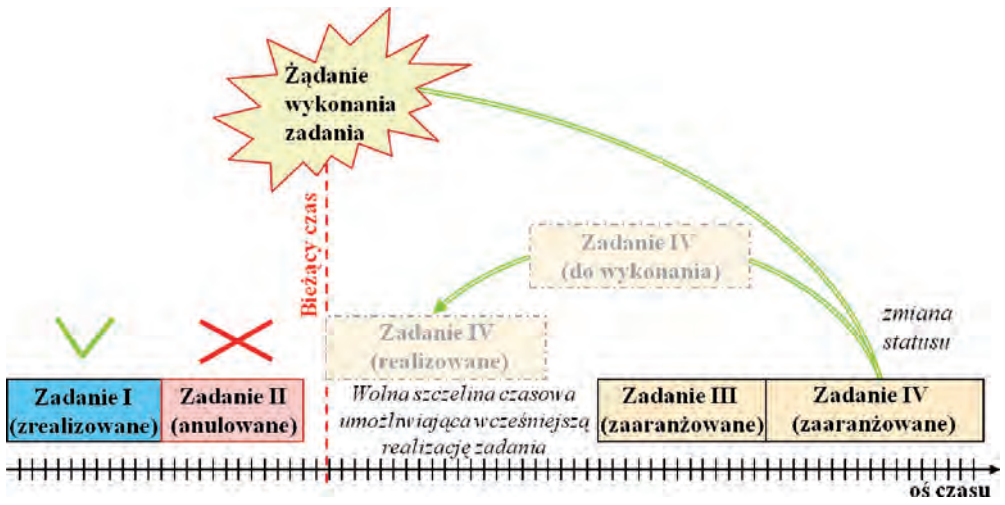
W prototypowym systemie została przyjęta zasada, że robot rozpocznie wykonanie danego zadania (ze statusem *do wykonania*), jeśli tylko pojawi się odpowiednia szczelina czasowa, która to umożliwi. Na rys. 8 został zilustrowany przypadek, w którym po odebraniu zadania wykonania *Zadania IV*, jest ono natychmiast realizowane. Dzieje się tak dlatego, że w danej chwili nie jest świadczona żadna usługa (robot jest bezczynny). Należy również zauważyć, że *Zadanie II* zostało wcześniej anulowane, co mogło przyczynić się do rozpoczęcia realizacji *Zadania III*, gdyby miało status *do wykonania*.

Faza wykonania zadania kończy się jego wykonaniem (pomyślnym lub nie) przez system robota lub w ostateczności odwołaniem tego wykonania przez Agenta.

8. Podsumowanie

W artykule przedstawiono propozycję nowej architektury układu sterującego pojedynczym robotem mobilnym. Zadaniem tego robota jest świadczenie usług w wielorobotowym systemie SOMRS. Sama architektura wchodzi w skład technologii opracowanych na potrzeby projektu Robo-enT (będącego realizacją SOMRS) i zakłada podział systemu robota na cztery warstwy programowe (pkt 3 i rys. 1). Zwłaszcza funkcjonalności zlokalizowane w dwóch najwyższych warstwach, tj. w *warstwie zarządzania usługami* oraz w *warstwie logiki wykonania usług*, należy uznać za szczególne rozwiązania przeznaczone do budowy układu sterującego robotem mobilnym, będącym częścią systemu SOMRS.

Na podstawie opracowanej architektury powstał prototypowy system robota, który został zainstalowany na dwóch fizycznych robotach typu Pioneer P3-DX. Z udziałem tych robotów zostały wykonane eksperymenty polegające na wyszukiwaniu w środowisku obiektu, którym było prostopadłościenne pudło, i jego transporcie we wskazane w zadaniu miejsce. Pudło było rozpoznawane dzięki przyklejonym do jego ścian znacznikom. Transport odbywał się między dwoma pomieszczeniami. Każde pomieszczenie było indywidualnym terenem świadczenia usług przez danego robota mobilnego. Innymi słowy żaden robot nie świadczył swoich usług w obu lokalizacjach (pomieszczeniach). W takich warunkach system SOMRS musiał dokonać aranżacji współpracy między wspomnianymi robotami mobilnymi. W tym celu zadanie transportu zostało podzielone na dwa podzadania. Jako punkt spotkania obu robotów i przekazania sobie pudła zostały wyznaczone drzwi, będące obiektem łączącym (wspólnym) obu pomieszczeń. Podobne eksperymenty wykonane w różnych pomieszczeniach kończyły się powodzeniem



Rys. 8. Odebranie żądania wykonania zadania i jego wcześniejsza realizacja
Fig. 8. Conditions of earlier task execution

w ok. 70 % przypadków, co należy uznać za znaczący sukces. Główną przyczyną niepowodzeń były nieprawidłowe zachowania prototypowego systemu nawigacyjnego, który wymaga dalszych usprawnień.

Zarówno opisana w tym artykule architektura układu sterującego pojedynczym robotem mobilnym, jak również cały system SOMRS są obecnie adoptowane i rozwijane w ramach projektu NCBR Nr PBS1/A3/8/2012 pt. *RobREx – Autonomia dla robotów ratowniczo-eksploracyjnych*. Niniejszy artykuł również jest efektem prac wykonanych nad tym projektem.

Bibliografia

1. Ambroszkiewicz S., Bartyna W., Terlikowski G., Faderewski M., *Service Oriented MultiRobot System*, Proc. The 3rd Israeli Conference on Robotics, 10–11 November, 2010, Herzlia, Israel.
2. Applegate D.L., Bixby R.E., Chvtal V., Cook W.J., *The Traveling Salesman Problem: A Computational Study*, Princeton University Press, 2009.
3. Arkin R.C., *Towards the Unification of Navigational Planning and Reactive Control*, AAAI Spring Symposium on Robot Navigation Working Notes, 1989, 1–5.
4. Bailey T., Nieto J., Nebot E., *Consistency of the EKF-SLAM algorithm*, Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems, Beijing, 2006, 3562–3567.
5. Bartyna W., *Protokoły realizujące współdziałanie inteligentnych kognitywnych urządzeń w otwartych i heterogenicznych systemach opartych na SOA* [w:] Inteligencja wokół nas: „Współdziałanie w systemach agentów softwareowych, mobilnych robotów oraz inteligentnych urządzeń", Akademicka Oficyna Wydawnicza EXIT, 2010, 87–132.
6. de Deugd S., Carroll R., Kelly K., Millett B., Ricker J., *SODA: Service Oriented Device Architecture*, IEEE Pervasive Computing, vol. 5, no. 3, 94–96, July-Sept. 2006, DOI:10.1109/MPRV.2006.59.
7. Faderewski M., *Reprezentacja środowiska robota* [w:] Inteligencja wokół nas: „Współdziałanie w systemach agentów softwareowych, mobilnych robotów oraz inteligentnych urządzeń", 13–43, Akademicka Oficyna Wydawnicza EXIT, 2010.
8. Jarvis R.A., *Distance Transform Based Collision-Free Path Planning for Robot*, Advanced Mobile Robots, World Scientific Publishing, 3-31, 1994.
9. Moritz G., Zeeb E., Pruter S., Golatowski F., Timmermann D., Stoll R., *Devices Profile for Web Services and the REST*, 584–591, 13–16 July 2010 Osaka.
10. Payton D., *An Architecture for Reflexive Autonomous Vehicle Control*, Proceedings of the International Conference on Robotics and Automation (ICRA), 1986. AAAI Spring Symposium on Robot Navigation Working Notes, 1–5.
11. Volkov S.N., Kaul B.V., Shelefontuk D.I., *Optimal Method of Linear Regression in Laser Remote Sensing* Appl. Opt. 41, 2002, 5078–5083.

Architecture of a control system for mobile robots in Service Oriented MultiRobot System

Abstract: A new architecture of a control system of a mobile robot is proposed. It is based on the SOA paradigm (Service Oriented Architecture), in which the robot is seen as a set of services it provides. In Computer Science, the SOA paradigm is a valid and often used approach when designing distributed systems. A multi-robot system is an example of such a system. The proposed architecture of a mobile robot control system consists of four software layers. The lowest layer, the *device control and data aggregation layer*, is responsible for the control of devices (sensors, effectors, etc.), with which the robot is equipped, and for aggregation , processing and fusion of data gathered by these devices. Functions of the next layer, the *navigation layer*, are usually implemented by a robot navigation system which enables efficient determination of routes and robot movement. Service execution controllers reside in the *service execution logic layer* and are responsible for the realization of various services provided by the robot. The top layer, the *service management layer*, consists of Services Manager which is responsible for the communication between the robot control system and the other components of the SOMRS system as well as for the management of service realization. A prototype robot system was developed based on the proposed architecture. It was installed on two Pioneer P3–DX mobile robots. Experiments involving these robots allowed us to verify the usefulness of the developed architecture in practical applications.

Keywords: SOA, architecture, service, service manager, marker, robot control system, object map

Artykuł recenzowany, nadesłany 21.11.2013 r., przyjęty do druku 20.12.2013 r.

dr Grzegorz Terlikowski

Otrzymał stopień doktora nauk technicznych w zakresie Informatyki w 2012 r. w Instytucie Podstawi Informatyki PAN. Obecnie jest pracownikiem Instytutu Informatyki Uniwersytetu Przyrodniczo-Humanistycznego (znanego wcześniej jako Akademia Podlaska) w Siedlcach, z którym jest związany od 2006 r. Zainteresowania: systemy rozproszone, urządzenia mobilne.

e-mail: g.terlikowski@gmail.com

mgr Waldemar Bartyna

Jest asystentem w Instytucie Podstaw Informatyki PAN oraz w Instytucie Informatyki UPH w Siedlcach. Jego główną dziedziną zainteresowań badawczych jest współdziałanie w ramach systemów wielorobotowych, a w szczególności architektura takich systemów i protokoły komunikacyjne.

e-mail: wbartyna@gmail.com