Symulacja w Badaniach i Rozwoju
Vol. 3, No. 4/2012

Dariusz PIERZCHAŁA, Rafał SYLWESTRZUK, Roman WANTOCH-REKOWSKI
Wojskowa Akademia Techniczna, Wydział Cybernetyki
ul. Gen. Sylwestra Kaliskiego 2, 00-908 Warszawa
E-mail: dariusz.pierzchala@wat.edu.pl, rrekowski@wat.edu.pl

Federacyjny adapter systemu GIS do standardu oprogramowania HLA

1 Wprowadzenie

Współczesne operacje wojskowe coraz częściej prowadzone są w formie działań połączonych, w których pod jednolitym dowództwem realizują przydzielone zadania elementy wszystkich rodzajów sił zbrojnych. Działania połączone przyjmują różną formę – od klasycznych po niekonwencjonalne, zmienne w różnych fazach operacji. Doktryna ta jest źródłem nowych wymagań dla systemów modelowania i symulacji pola walki, skutkiem czego autonomiczne symulatory wybranych działań lub rodzajów sił zbrojnych nie są już wystarczające. Przygotowanie symulatorów wszystkich rodzajów sił zbrojnych dla prowadzenia symulacji wszystkich działań, w tym połączonych, na wszystkich poziomach szczegółowości – od niemalże pojedynczego żołnierza aż do zgrupowania międzynarodowego dowolnie wysokiego szczebla – implikuje wiele zadań. Wśród nich jest jednolite zobrazowanie (ang. *Common Operational Picture*) oraz integracja programowa, niezbędne dla efektywnego przygotowania sztabów i dowódców do współdziałania, w tym także z międzynarodowymi siłami sojuszniczymi.

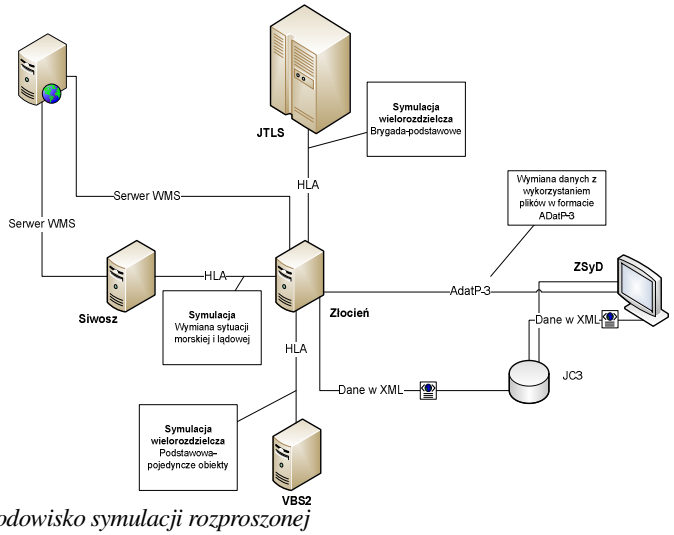
Wielorozdzielczość symulacji dotyczy zagadnień różnej skali i złożoności, na wielu poziomach szczegółowości odwzorowania. Bezpośrednim powodem jej stosowania są potrzeby informacyjne o zróżnicowanej szczegółowości na różnych poziomach decyzyjnych. Stosowane modele powinny charakteryzować się zmienną rozdzielczością w zależności od rodzaju działań i szczebla wojsk. Dobór stopnia szczegółowości w modelowaniu jest zasadniczą trudnością, gdyż wpływa na wydajność systemu, ale także na przydatność do spełnienia stawianych przed nim zadań. Ponadto należy zapewnić możliwość jednoczesnego uczestnictwa w symulacji dowódców różnych szczebli, dla których dostępne są różne zestawy zmiennych decyzyjnych, inna perspektywa widzenia, ale spójny powinien pozostać obraz symulowanego pola walki.

Podstawowym protokołem umożliwiającym integrację i wymianę danych pomiędzy geograficznie rozproszonymi systemami jest federacyjny standard High Level Architecture (w wersji wojskowej DoD HLA 1.3, w cywilnej IEEE 1516) [2,3,6]. Środowiska symulacyjne, czyli Federacje, złożone są z federatów wymieniających pomiędzy sobą dane: interakcje lub uaktualnienia atrybutów obiektów. Warunkiem koniecznym w procesie wymiany danych jest utworzenie dla nich specyficznych deklaracji w dedykowanych obiektowych modelach federacji (ang. FOM – *Federation Object Model*). Przyjmują one postać uaktualnień wartości atrybutów obiektów lub interakcji. Centralną i jednocześnie wspólną częścią HLA jest RTI (ang. *Runtime*

Infrastructure), które nie ulega zmianom podczas rekonfiguracji środowiska. Wypełnia funkcje usługowe zarządzania czasem, wspólnymi danymi, i komunikacją wobec integrowanych federatów.

2 Koncepcja adaptera systemu GIS

Przeprowadzone badania były częścią projektu [5], którego celem było opracowanie heterogenicznego rozproszonego środowiska symulacyjnego, łączącego funkcjonujące w Siłach Zbrojnych RP autonomiczne symulatory konstruktywne różnego szczebla dowodzenia. Rysunek 1 prezentuje w sposób uproszczony przyjętą architekturę.



Rys. 1. Środowisko symulacji rozproszonej
Fig. 1. Distributed simulation environmen

Zasadnicze modele symulacyjne zrealizowano w systemie *Złocień*, dlatego też badania adaptera wykonano we współdziałaniu z tym symulatorem. Natomiast zadania udostępniania jednolitego obrazu pola walki zrealizowano w zmodyfikowanym środowisku programowym *OpenMap*.

Oprogramowanie *OpenMap* jest to narzędzie zbudowane w technologii *Java Beans*, będące zbiorem komponentów Swing. Wspomagają one wyświetlanie danych na mapie oraz przechwytywanie zdarzeń podczas ich modyfikacji. *OpenMap* umożliwia rozwój aplikacji poprzez tworzenie własnych modułów oraz komponentów. Moduły mogą być dowolną aplikacją działającą na obiektach będących źródłem informacji geograficznych. Natomiast komponentem może być warstwa lub plug-in, pełniący zadanie prezentowania danych. Nanoszenie nowych elementów oraz manipulowanie nimi dostępne jest poprzez interfejs użytkownika, którym także można zarządzać oraz tworzyć nowe elementy. *OpenMap* umożliwia przetwarzanie danych różnego typu i specyfikacji. Takimi danymi są CADRG, CIB, DCW, ESRI shapefiles, RPF, VMAP, VPF.

W uproszczonym ujęciu dominującym zadaniem systemu OpenMap jest prezentacja struktury jednostek i dynamiki zmian stanów, zadań oraz obiektów pola walki. Wszystkie dane do zobrazowania (z wyłączeniem wolnozmiennych danych opisujących teren) generowane są w procesach symulacyjnych realizowanych w symulatorze *Złocień*. Komunikacja i przesłanie danych do zobrazowania w *OpenMap* wymagała adaptera przystosowującego ten system do roli federata standardu HLA oraz konwertującego odbierane z RTI komunikaty o strukturze Federation Object Model (FOM) na kolekcje danych akceptowane w OpenMap. Założono zatem, iż:

- adapter-federat odbiera dane z systemu Złocień wyłącznie za pośrednictwem RTI;
- federat jest dostosowywalny do różnych modeli obiektowych FOM;
- federat przesyła odbierane dane do struktur danych obsługiwanych przez *OpenMap*;
- możliwa powinna być praca adaptera w przeciwnym kierunku.

W rozwiązaniu zaproponowano następujące składowe programowe: *federat*, *Annotation writer*, *Cache*, *Cache manager* oraz *Klient* aplikacji *OpenMap*. *Klient* odpowiedzialny jest za przekazania do *OpenMap* informacji do zobrazowania. *Federat* jest natomiast łącznikiem z RTI, odpowiedzialnym za odbieranie komunikatów oraz ich rejestrację w składnicy danych *Cache*. Sama składnica *Cache* przechowuje wymieniane dane obiektowe (opisane modelem FOM). Zarządzaniem danymi w *Cache* zajmuje się *Cache* manager, do którego *Klient* powinien móc się zarejestrować/ wyrejestrować. *Cache* manager odpowiedzialny jest również za przetworzenie obiektów modelu.

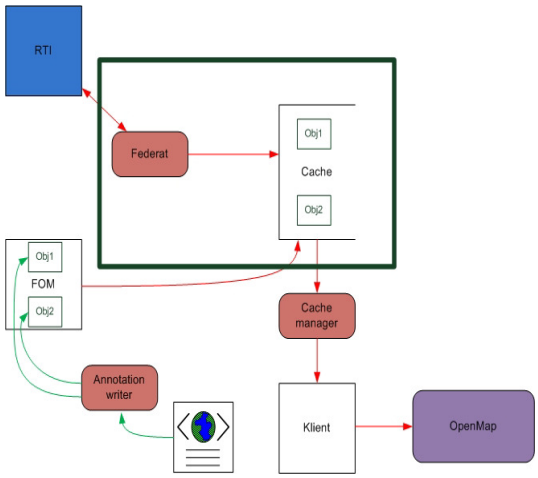
W celu zapewnienia możliwości wykorzystania różnych modeli FOM, a tym samym otwarcia się na niemalże dowolnych federatów, zastosowano generator klas FOM oraz wprowadzono model adnotacji, przedstawiający sposób mapowania atrybutów obiektów FOM na atrybuty obiektów *Klienta* systemu *OpenMap*. Do wytworzenia klas programowych obsługujących każdy model FOM zaprojektowano parser plików FOM, współpracujący z generatorem klas źródłowych. Docelowo zarówno *federat*, jak i klasy obsługujące FOM uzupełniane są o model adnotacji zapisany w schemacie pliku XSD. *Federat* wymaga między innymi modyfikacji metody subskrybującej wskazane klasy obiektów komunikatów, zaś klasy dla FOM są rozszerzane o wpisy adnotacji (m.in. typ danych oraz nazwa mapowanego atrybutu). Mapowanie z wykorzystaniem adnotacji wykonuje się przed wysłaniem danych do *Klienta*, dzięki czemu realizuje się to w jednej procedurze, jednakowej dla różnych klas FOM:

```
1. Pobierz dane z pliku XML
2. Pobierz listę elementów annotations
3. Odczytaj element z listy annotations
       4.  Odczytaj element classname
       5.  Sprawdź, czy istnieje plik klasy
              6. Jeśli tak, to przejdź do 7 – w pp. idź do 3
       7.  Odczytaj zawartość pliku
       8.  Dodaj definicje adnotacji do listy importów w klasie
       9.  Pobierz listę atrybutów props-list
       10. Odczytaj element listy props-list
              11. Wstaw adnotację nad atrybutem klasy
              12. Sprawdź, czy koniec listy props-list
                     13. Jeśli tak, to przejdź do 14 – w pp. idź do 12
       14. Zapisz plik klasy
```

```
        15. Sprawdź, czy koniec listy annotations
              16. Jeśli tak, to przejdź do 17 – w pp. idź do 3
17. Koniec
```

Uproszczony schemat działania przedstawia się następująco (rys. 2):

1. Federat odbiera dane z RTI;
2. Dane są przekazane do Cache;
3. Cache manager dokonuje mapowania danych;
4. Dane są przesyłane do repozytorium Klienta.



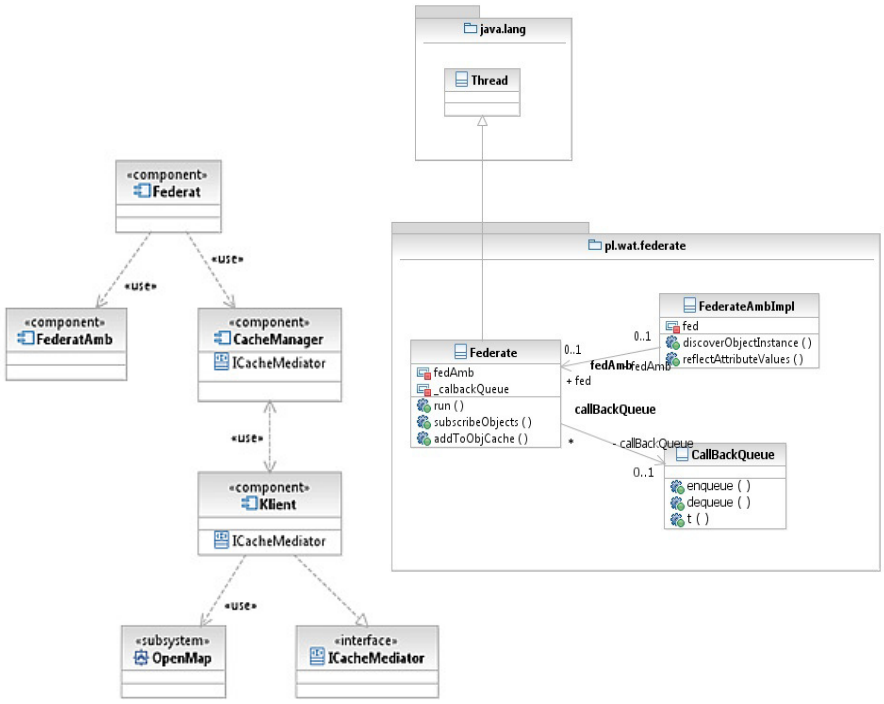
Rys.2. Schemat koncepcyjny adaptera
Fig. 2. Concept diagram of adapter

3 Składowe projektu adaptera

Kluczową składową adaptera jest *federat*, którego główną klasą (będącą jednocześnie wątkiem) jest *Federate*. Posiada referencję na ambasadora, repozytorium obiektów w aplikacji OpenMap oraz obiekt klasy CallbackQueue dla potrzeb zwrotnego połączenia z RTI. Głównymi zadaniami klasy tej jest rejestracja instancji federata w federacji, określenie zarządzania czasem, subskrypcja odpowiednich obiektów FOM oraz umożliwianie odbierania informacji z RTI. *FederateAmbImpl* jest implementacją klasy *hla.rti.FederateAmbassador*, która jest wysyłana do RTI, gdy federat włącza się do federacji. Wywołania pochodzące z RTI uruchamiane są w obiekcie klasy *FederateAmbImpl*, która używając referencji do klasy *Federate*, wywołuje jej odpowiednie metody. Drugą składową adaptera jest *CacheManager* - jego istotne klasy to *Cache, CacheMediator, CacheChecker* oraz *AnnotProp*. *CacheMediator* umożliwia klientom rejestrowanie/wyrejestrowanie się w celu odbierania danych z *Cache*. Referencje klientów są przechowywane w *registeredList*. *ICacheMediator* jest interfejsem, który musi być implementowany przez każdego klienta chcącego odbierać dane z *Cache* (metoda *insertObject*). *AnnotProp* to interfejs tworzący adnotację, wykorzystywany we wszystkich klasach w zasięgu projektu. Metody użyte w *AnnotProp* to: *typeInTab* (wskazująca na typ przechowywany w obiekcie klienta)

oraz *mappedProp* (mapująca atrybut w pierwotnym obiekcie). *CacheChecker* odpowiada za nadzorowanie stanu ilościowego obiektów zaktualizowanych i dodanych do *Cache*, a także za przetworzenie każdego z tych obiektów do obiektu akceptowanego przez klienta i wysłanie go poprzez interfejs. Interfejsem jest metoda klasy *ICacheMediator*, którą klient powinien posiadać.

Poszczególne składowe adaptera wiążą się zgodnie ze schematem opisanym diagramem na rysunku 3. *Federat* jest źródłem danych dla reszty klas, a dane które otrzymuje poprzez komponent *FederatAmb*, przekazuje do komponentu *CacheManager*. Ten natomiast przechowuje dane, przekształca i przesyła do *Klienta*, który przygotowuje postać danych na potrzeby systemu *OpenMap*, gdzie zostaną zobrazowane.



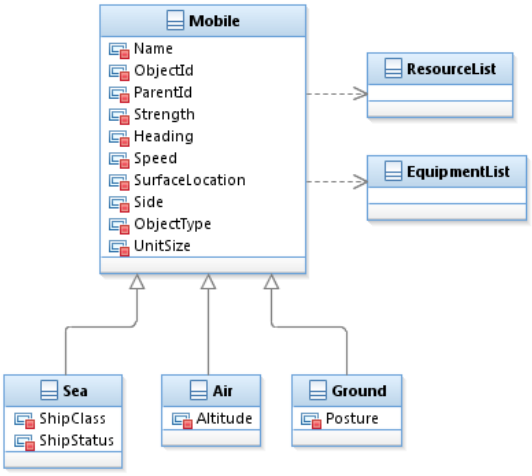
Rys.3. Diagramy wybranych komponentów i klas
Fig. 3. Selected component and class diagrams

4 Model danych federacyjnych

W federacyjnych środowiskach symulacyjnych zgodnych ze standardem HLA rozproszone systemy (federaci) wymieniają pomiędzy sobą dane poprzez komunikaty typu interakcje lub uaktualnienia atrybutów obiektów. Warunkiem koniecznym dla wymiany danych jest utworzenie obiektowych opisów w dedykowanych modelach SOM/FOM. Dla klas obiektów określa się ich jednoznaczną dla całej federacji nazwę oraz atrybuty. Dla klas interakcji określa się analogicznie jednoznaczną nazwę

oraz zestaw parametrów. Zasadniczą różnicą między oboma typami komunikatów jest to, że obiekty są trwałe, czyli istnieją przez pewien czas eksperymentu symulacyjnego, natomiast interakcje zostają powołane przez nadawcę, a usuwane natychmiast po ich odebraniu przez odbiorców (nie są zatem trwałe). Obiekty powinno stosować się, gdy wymieniane dane można traktować jako byty, które mają stany zmieniające się w czasie. Jeśli natomiast dane wymieniane mają charakter zdarzeń, powinno stosować się interakcje.

Model interfejsu danych systemu Złocień oparty jest na modelu FOM opracowanym w projekcie *DiMuNDS2000* [7]. Wykonane rozszerzenia mają na celu odwzorowanie algorytmów działań specyficznych dla systemu Złocień. Na diagramie klas z rysunku 4 przedstawiono elementy modelu opisujące obiekty mobilne (np. pododdział).



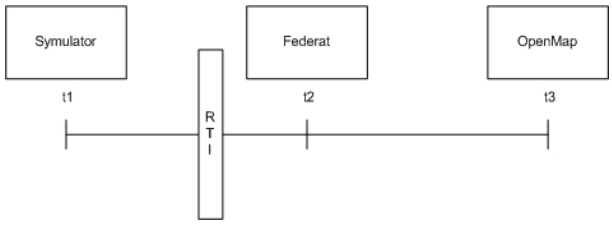
Rys. 4. Klasy obiektów mobilnych w FOM
Fig. 4. Mobile object classes inside FOM

5 Eksperymenty wydajnościowe

Wprowadzenie pomiędzy RTI i *OpenMap* dodatkowych składowych programowych, realizujących zadania mapowania, konwersji i buforowania danych, wiąże się z dodatkowym opóźnieniem w udostępnianiu użytkownikowi obrazu symulowanego pola walki. W celu zbadania wielkości opóźnienia i jego wpływu na proces decyzyjny przeprowadzono eksperymenty w następującej konfiguracji:

Sprzęt i system	Parametry
Procesor	Intel Core 2 CPU T5500 1,66GHz
Pamięć	2,00 GB DDR2-667
System operacyjny	Microsoft Windows 7 x64
Serwer RTI	MÄK RTI v3.3.2

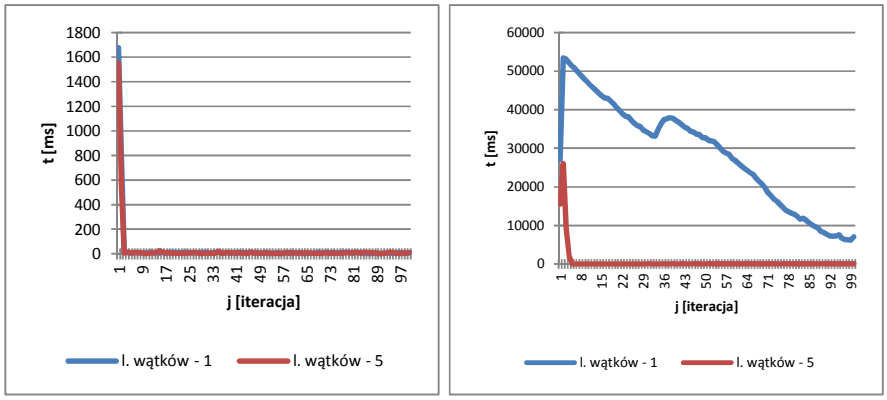
Podstawowe szacowane charakterystyki to czasy przesyłania danych z symulatora *Złocień* do adaptera, a następnie z adaptera do systemu OpenMap. Procedura pomiaru zaprezentowana jest schematycznie na rysunku 5. Pomiar czasu t_1 odbywa się po wysłaniu modyfikacji każdego obiektu do RTI. Czas t_2 to moment odebrania obiektu z RTI. Natomiast czas t_3 rejestrowany jest po zobrazowaniu przez *OpenMap*.



Rys. 5. Koncepcja testowania
Fig. 5. Testing idea

W planie testów uwzględniono po sto iteracji, a w każdej z nich modyfikację określonej liczby obiektów. Wysyłane obiekty łączono w różne paczki (1, 10 lub 100) i wysyłano ze stałym interwałem jednej sekundy. W badaniach zmieniano liczbę wątków (1 lub 5), które jednocześnie wykonywały zadania w *CacheChecker*.

Na kolejnych wykresach przedstawiono zależności łącznego czasu przesyłania od liczby wątków dla paczek: jedno- (z lewej) i 100- obiektowych (z prawej).



Rys. 6. Zależności czasowe – paczki jedno- i 100-tu obiektowe
Fig. 6. Time dependency – packages of one and 100 objects

Zysk czasowy przy obsłudze paczek jednoelementowych przez pięć wątków jest niezauważalny. W przypadku paczek 10-obiektowych pojawia się już nieznaczne przyspieszenie: średni czas obsługi zmniejsza się o 88,7 [ms]. Zdecydowaną poprawę uzyskano natomiast dla paczek 100-obiektowych – średni czas przy obsłudze przez jeden i pięć wątków, wynosi odpowiednio: 28460 [ms] oraz 578 [ms].

5 Podsumowanie

W ramach prac powstał programowy adapter integrujący w środowisku federacyjnym symulatory i systemy zobrazowania ujednoliconej sytuacji taktycznej. Rozwiązanie to umożliwia wykorzystanie systemów nowo projektowanych oraz już istniejących, gdyż uwzględnia heterogeniczność sprzętowo-programową i rozproszenie federatów. Uzyskano możliwość zobrazowania informacji pochodzących z wielu różnych źródeł, a jednocześnie udostępniania informacji w wielu instancjach OpenMap jednocześnie.

Istotną zaletą oprogramowania jest zapewnienie dostosowania OpenMap do różnych modeli obiektowych FOM. Adaptowalność tę uzyskuje się dzięki komponentom interpretującym zapisy w FOM oraz tworzącym klasy uzupełnione o model mapowania atrybutów. Dodatkowy nakład pracy na budowę warstwy integrującej zrekompensowany może być dopiero wówczas, gdy integracji poddanych jest wiele systemów.

Literatura

1. Pierzchała D.:, Integracja rozproszonych systemów symulacji oraz wspomagania decyzji. *Przegląd Telekomunikacyjny i Wiadomości Telekomunikacyjnych*, Zeszyt 8-9/2009, CD appendix, 2009
2. Pierzchała D.: Standardy symulacji rozproszonej. *Materiały konferencyjne XV Warsztatów Naukowych PTSK*, pp. 322–333, Warszawa 2009
3. Pierzchała D.: Designing and testing method of distributed interactive simulators. *Proceedings of the 15th ICSS*, Paper 2004-90, Wrocław 2004
4. Antkiewicz R., Kulas W., Najgebauer A., Pierzchala D., Rulka J., Tarapata Z., Wantoch-Rekowski R.: Selected Problems of Designing and Using Deterministic and Stochastic Simulators for Military Trainings. *43rd Hawaii International*

Conference on System Sciences, IEEE Computer Society, Kauai, Hawaii (USA), 2010

5. Antkiewicz R., Kulas W., Najgebauer A., Pierzchala D., Rulka J., Tarapata Z., Wantoch-Rekowski R.: Modelling and simulation of C2 processes based on cases in the operational simulation system for CAXes, *Bulletin of Military University of Technology*, 4 (652), Vol. LVII, pp. 9-24, 2008
6. Kuhl F., Weatherly R., Dahmann J.: Creating Computer Simulation Systems. *An introduction to the High Level Architecture*, ISBN-10: 0130225118, 1999
7. Briggs R.A.: Experiences in the NATO pre-pathfinder DiMuNDS 2000 federation, *Simulation Conference Proceedings*, ISBN 0-7803-5780-9, pp. 1039 – 1043, 1999

Streszczenie

W artykule przedstawione zostały wybrane problemy projektowania i integracji komputerowych rozproszonych symulatorów działań bojowych dla potrzeb wspomagania dowodzenia oraz szkoleń dowódczo-sztabowych. Omówione zostały w szczególności zagadnienia wielorozdzielczego zobrazowania mapowego i udostępniania jednolitego obrazu w spójnej przestrzeni modelowanego obszaru działań. Programową platformą integracji oraz wymiany danych jest Runtime Infrastructure standardu High Level Architecture. Podstawowym systemem symulacyjnym jest Złocień, natomiast funkcje graficzne wypełnia zmodyfikowana wersja oprogramowania OpenMap. Rozważania kończy krótki opis eksperymentu, którego celem było badanie wydajności rozwiązania.

Słowa kluczowe: federacja HLA, symulacja rozproszona, systemy GIS

The federated adapter of a GIS system to HLA standard

Summary

The paper presents some aspects of modelling, building and integrating computer distributed combat simulators for Computer Assisted Exercises and decision support. Multiresolution Common Operational Picture is discussed. As an integration platform the HLA Runtime Infrastructure has been chosen. A basic combat simulator called Zlocien plays the main role during exercises. Graphical tasks are accomplished with open-source software OpenMap. The conducted work is concluded with short description of the experiment aimed to quantitative assessment of the proposed solution.

Keywords: HLA federation, distributed simulation, GIS systems