Symulacja w Badaniach i Rozwoju
Vol. 6, No. 1/2015

**Andrzej NAJGEBAUER, Michał DYK,
Dariusz PIERZCHAŁA**
Wojskowa Akademia Techniczna, Wydział Cybernetyki,
00-908 Warszawa, ul. Gen. Sylwestra Kaliskiego 2
E-mail: andrzej.najgebauer@wat.edu.pl, michal.dyk@wat.edu.pl,
dariusz.pierzchala@wat.edu.pl

Modelowanie i symulacja sieci IoT w symulatorze SenseSim

1 Wstęp

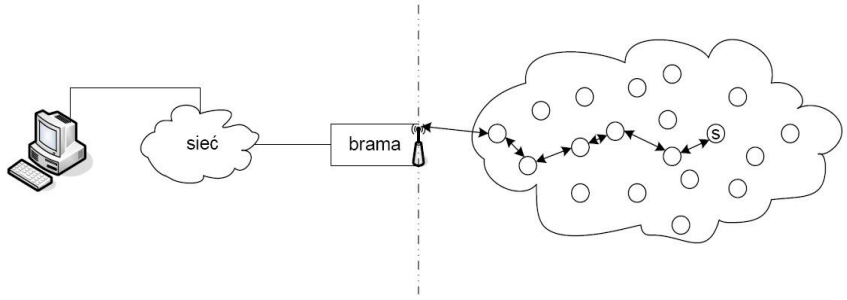
Dynamiczny rozwój szeroko pojętych urządzeń mobilnych, w szczególności bezprzewodowych, które stają się „wszechobecne", sprzyja rozwojowi dwu silnie związanych koncepcji: sieci sensorowych oraz Internetu Rzeczy (ang. IoT – Internet of Things). Sieć sensorową najogólniej można określić jako sieć złożoną z wielu - najczęściej więcej niż kilkunastu - urządzeń, które rozlokowane na ustalonym obszarze realizują wspólne zadanie [1]. Ich badania, w tym symulacyjne, prowadzone są od wielu już lat (m.in. [1, 2, 3, 4, 5, 6]). Dostęp do sieci Internet, a także do różnego rodzaju danych sensorycznych stał się łatwy i powszechny, zatem kolejnym krokiem w ewolucji sieci sensorycznych stał się Internet Rzeczy, gdzie głównym postulatem jest podłączenie jak największej liczby urządzeń elektronicznych do sieci Internet. W efekcie każde z nich otrzyma własny adres internetowy, za pośrednictwem którego możliwe będzie pozyskiwanie danych lub sterowanie nim. Daje to ogromne możliwości - począwszy od regulowania ogrzewaniem we własnym domu po zaawansowane monitorowanie i sterowanie procesem produkcyjnym za pośrednictwem jednego komputera, laptopa czy telefonu.

Wdrażanie w życie koncepcji Internetu Rzeczy (ang. IoT – Internet of Things), stwarza szereg potrzeb wobec stosowanych metod, algorytmów, protokołów, modeli i interfejsów danych etc. Najlepszą metodą ich weryfikacji i oceny byłoby środowisko typu „testbed", zawierające rzeczywiste urządzenia mobilne oraz sensory, co jest niestety bardzo kosztowne i czasochłonne w realizacji. Powszechnie stosowaną alternatywą są środowiska symulacyjne, odwzorowujące wszystkie elementy rzeczywiste, niezbędne z punktu widzenia celu badań. Niniejszy artykuł stanowi kontynuację prac [7], [8], [9], [10] w kierunku modelu symulacyjnego sieci sensorowej i Internetu Reczy dla oprogramowania symulatora SenseSim, prezentacji SenseSim oraz koncepcji rozwinięcia o metody modelowania i symulacji różnej rozdzielczości.

2 Sieci sensorowe i Internet Rzeczy

Podstawowym elementem sieci sensorowej jest sensor (stąd nazwa sieci), nazywany także węzłem. Każdy z nich ma takie same możliwości komunikacyjne i żaden nie jest wyróżniony w momencie tworzenia sieci. Przyjmuje się, że sieć sensorowa posiada „bramę" (nazywaną też „ujściem"), czyli element, do którego przesyłane są zgromadzone dane lub z którego wysyłane są zapytania do sieci. Ogólny schemat bezprzewodowej sieci sensorowej przedstawia rysunek 1. Przesyłanie danych pomiędzy

bramą a sensorami (lub w razie potrzeby tylko pomiędzy węzłami) odbywa się najczęściej metodą „skoków". Jeśli węzeł będący źródłem komunikatu nie ma bezpośredniego połączenia z bramą, przesyła dane do węzła sąsiedniego, który z kolei przekazuje komunikat następnym sensorom. Sieci sensorowe mają wiele zastosowań – wykorzystywane są między innymi do monitorowania zjawisk przyrodniczych, wczesnego ostrzegania przed lawinami lub pożarami (np. lasów). Oczywiście, mają także zastosowania militarne, np. w celu wykrywania lub śledzenia przeciwnika.



Rys. 1. Ogólny schemat sieci sensorowej. Źródło: Wikipedia
Fig. 1. General scheme of sensor networks. Source: Wikipedia

Sieci IoT, rozwijane na bazie sieci sensorowych, adekwatnie do rodzaju łączonych urządzeń, operują danymi o różnej gradacji, zatem stosowane dla nich modele powinny charakteryzować się zmienną rozdzielczością lub wielorozdzielczością (ang. MRM – *multi-resolution modelling*). Przyjmuje się, że MRM to różna skala, perspektywa widzenia i szczegółowość odwzorowania tego samego fragmentu rzeczywistości. W praktyce nie jest możliwe, aby jeden monolityczny (w sensie rozdzielczości) symulator zrealizował zróżnicowane potrzeby użytkowników, w szczególności spełnił oczekiwanie wielu rozdzielczości – stąd wynika potrzeba modelowania i symulacji typu MRM.

Dobór stopnia szczegółowości w modelowaniu jest zasadniczą trudnością, gdyż wpływa na wydajność systemu, ale także na jego przydatność. Ponadto większość znanych z literatury podejść do agregacji i deagregacji powoduje utratę informacji przy operacjach przejście/powrót pomiędzy różnymi poziomami rozdzielczości – tracone są wartości niektórych istotnych parametrów związanych z wyższym poziomem rozdzielczości. Kolejny powtarzalny problem to fakt, iż częste stosowanie transformacji między poziomami wpływa znacząco na pogorszenie efektywności symulacji (w sensie zwiększenia czasu realizacji eksperymentu) [12]. Rozwiązaniem może być metoda MRE, która wprowadza koncepcję obiektu wielorozdzielczego, opisanego wieloma modelami adekwatnymi dla przyjętych poziomów rozdzielczości. W proponowanym w artykule podejściu grunt dla hierarchii modeli oraz funkcji zależności i przejść między modelami stanowi ontologia oraz zapisany w niej metamodel, służący transformacji danych o różnych rozdzielczościach.

3 Model symulacyjny sieci sensorowej/IoT w symulatorze SenseSim

Założeniem symulatora SenseSim jest możliwość symulowania urządzeń połączonych zarówno w sieć sensorową, jak i Internet Rzeczy. Aspekty techniczne komunikacji bezprzewodowej modelowane są jednakże w uproszczeniu, stąd zastosowany model komunikacji jest idealistyczny i nie uwzględnia wielu niskopoziomowych aspektów. Wynika to z faktu, iż SenseSim został zaprojektowany jako narzędzie do symulacji sieci sensorowej rozumianej jako samoorganizujący się system osadzony w środowisku, w którym występują różne, zmieniające się zjawiska. Ponadto proponowany symulator umożliwia makroprogramowanie sieci, dzięki czemu jest ona programowana jako całość. Przygotowywany jest jeden wysokopoziomowy program, który następnie jest dystrybuowany pomiędzy urządzeniami, i to do nich należy dopasowanie swojego zachowania do programu.

Sieć bezprzewodowa w SenseSim modelowana jest jako graf typu Unit Disc Graph (UDG) [14], co jest popularnym sposobem na modelowanie bezprzewodowych sieci sensorowych (WSN). Definicja sieci jest następująca:

$$SN(t) = \langle S, E(t), b(t, s_1, s_2) \rangle,$$

gdzie *S* to zbiór urządzeń, a *E(t)* to zbiór krawędzi grafu. Para urządzeń, które w danej chwili czasu *t* znajdują się wzajemnie w swoim zasięgu radiowym, traktowana jest jako sąsiedzi, czemu odpowiada krawędź grafu. Funkcja *b* zdefiniowana jest na krawędziach i określa przepustowość łącza komunikacyjnego w chwili czasu *t*. Podstawowym zadaniem sensorów w SenseSim jest obserwacja zmieniającego się środowiska. Każdy z nich modelowany jest jako ósemka uporządkowana, której stan może się zmieniać w czasie:

$$s(t) = \langle L(t), V, r, \{O_i\}_{i=1..n}, \{A_i^i\}_{i=1..n}, P(t), R(t), SNT(t) \rangle.$$

L(t) jest geograficzną lokalizacją urządzenia w chwili czasu *t*; *V* reprezentuje prędkość; *r* jest zasięgiem radiowym; *A* jest zbiorem obserwacji zebranych przez urządzenie za pomocą każdej z jego zdolności percepcyjnych; *P(t)* jest zbiorem zawierającym programy wykonywane przez urządzenie w chwili *t*; *R(t)* to zbiór przechowujący dostępne w chwili *t* zasoby urządzenia (takie jak ilość wolnej pamięci RAM, stan baterii itp.); *SNT(t)* reprezentuje część sieci, o której dane urządzenie ma pełną wiedzę (w szczególności zna położenie innych węzłów sieci, co jest kluczowe dla trasowania). Istotną częścią modelu sensora jest zbiór *O*, który przechowuje definicje tzw. obserwatorów. Każdej zdolności percepcyjnej sensora odpowiada jeden obserwator, zgodny z formalną teorią percepcji. Takie podejście pozwala na elastyczne konfigurowanie symulowanych urządzeń. Jedno z nich może posiadać wiele zdolności do obserwacji świata, wśród których mogą występować również te, które aktualnie uznawane są za futurystyczne.

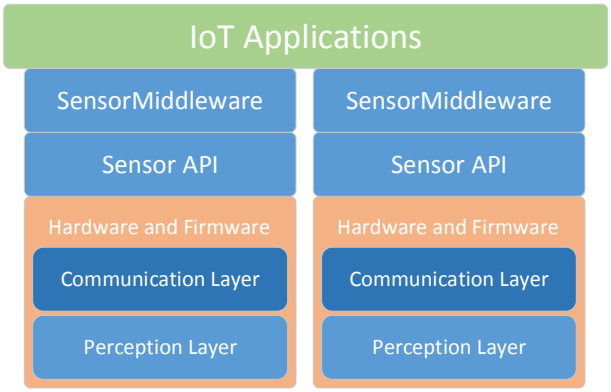
Źródłami obserwacji dla sensorów są zmienne zjawiska, które występują w otaczającym urządzenie środowisku. Każde zjawisko definiowane jest jako czwórka:

$$e(t) = \langle R(t), \{s_i(t)\}_{i=1..n}, t_b, t_e \rangle,$$

gdzie $R(t)$ to obszar, jaki obejmuje zjawisko; t_b, t_e są chwilami czasu określającymi jego początek i koniec. Zbiór $\{S_i(t)\}_{i=1..n}$ przechowuje funkcje (mogą być utożsamiane ze zmiennymi losowymi), które opisują rozwój zjawiska w czasie. Każda z tych funkcji odpowiada konkretnej zdolności percepcyjnej sensora.

Zasadniczą cechą symulatora SenseSim jest jego zdolność do modyfikowania zachowania symulowanych urządzeń. Realizowane jest to poprzez makroprogramowanie sieci, co oznacza, że użytkownik nie musi programować każdego urządzenia z osobna. W zamian budowany jest jeden wysokopoziomowy program, który każde z urządzeń potrafi zinterpretować. Obecnie dostępnych jest wiele języków makroprogramowania sieci, takich jak Korios, Regiment, TinyDB czy ATaG – jest natomiast istotne, aby mogły poprawnie działać w heterogenicznej sieci z urządzeniami o różnej konfiguracji sprzętowej oraz różnych zdolnościach percepcyjnych. Kluczowa dla zapewnienia takiej interoperacyjności urządzeń jest ich architektura.

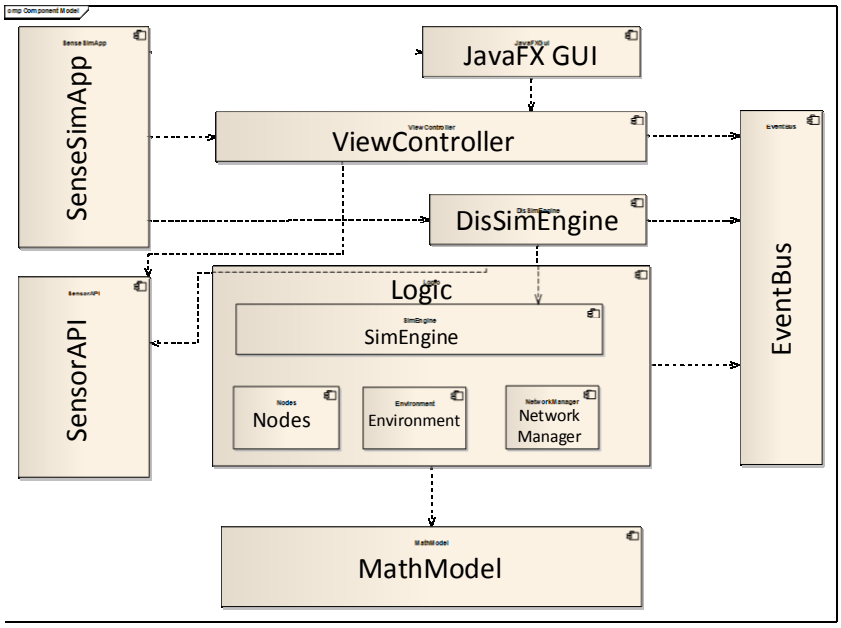
Istnieje wiele koncepcji architektury urządzeń włączonych w Internet Rzeczy i sieci sensorowe. Wiele z nich, np. [6], zakłada że ma ona charakter wielowarstwowy. Takie podejście zostało również zastosowane w SenseSim. Rysunek 2 przedstawia architekturę sensora w symulatorze. Wyróżnione zostały trzy główne warstwy: sprzętowa i wbudowanego oprogramowania (ang. *hardware and firmware*), API oraz *middleware*. W skład pierwszej z nich wchodzą dwie warstwy: warstwa percepcji, która odpowiada za obserwację przez urządzenie świata zewnętrznego, oraz warstwa komunikacji, która odpowiada za wymianę danych i informacji z sąsiednimi sensorami. Na bazie tych warstw zbudowana została warstwa API, która udostępnia mechanizmy kontrolowania urządzenia z zewnątrz. Warstwa ta jest podstawą dla opracowania warstwy middleware, którą uważamy za najbardziej istotną w SenseSim. To ona odpowiada za interoperacyjność heterogenicznych urządzeń i jest kluczowa w procesie makroprogramowania sieci. Jednym z głównych zadań stawianych przed SenseSim jest symulowanie sensorów z middleware, które posiada te same funkcje co middleware w rzeczywistych urządzeniach. Dzięki temu możliwe będzie uruchomienie tego samego programu w symulowanej, jak i rzeczywistej sieci.



Rys. 2. Architektura sensora w SenseSim. Źródło: Opracowanie własne
Fig. 2. SenseSims's sensors architecture. Source: own preparation

Kluczowym celem, który przyświecał projektowaniu SenseSim, była jego łatwa reakonfigurowalność i dalszy rozwój. Rysunek 3 przedstawia ogólną architekturę symulatora. Głównymi komponenetami są: *MathModel*, *Logic*, *DisSimEngine*, *ViewController* and *EventBus*. *MathModel* jest programową implementacją modelu matematycznego, w oparciu o który budowany był symulator. Zawiera wszelkie, czasowo niezależne, operacje związane z zarządzaniem siecią i jej węzłami, jak np. dodanie/usunięcie krawędzi. Z komponentu *MathModel* czerpie kluczowy komponent symulatora, czyli *Logic*. W jego skład wchodzą kolejne cztery składowe:

- *Nodes* - zawiera klasy, które odpowiadają za zarządzanie sensorami; w tym miejscu zdefiniowane są operacje związane na przykład z ruchem bądź obserwacją świata;
- *Environment* – zawiera klasy odpowiedzialne za zarządzanie obserwowanymi zjawiskami; w odróżnieniu od *Nodes* operacje tu są czasowo zależne;
- *NetworkManager* – odpowiedzialny za zarządzanie siecią bezprzewodową;
- *SimEngine* – główny komponent realizujący sterowanie eksperymentem symulacyjnym dla wybranego silnika symulacji zdarzeniowej.



Rys. 3. Architektura symulatora SenseSim. Źródło: Opracowanie własne
Fig. 3. SenseSim architecture overview. Source: own preparation

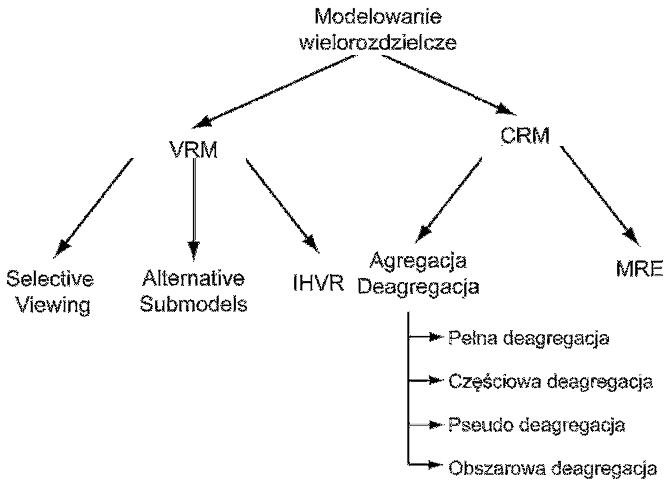
W dużym uogólnieniu praca urządzeń modelowana jest według następującego schematu. Każde urządzenie w symulatorze SenseSim posiada domyślne zachowanie, co rozumiemy jako operacje realizowane podczas symulacji bez dodatkowego makroprogramowania. W przypadku rzeczywistych urządzeń jest to definiowane poprzez natywny hardware i firmware. Po uruchomieniu symulacji węzeł rozpoczyna

wykrywanie oraz ruch (jeśli zdefiniowano marszrutę i zadano niezerową prędkość w scenariuszu początkowym) – są to stany *Sensing* i *Moving*. Jeśli nadejdzie wiadomość, sensor przejdzie do stanu *Receiving*, który jest współbieżny do obu powyższych. Po zainicjowaniu komunikacji sensor odbiera wiadomość (oddzieloną na bity), przy czym prędkość transferu zależy od przepustowości połączenia między nadawcą a sensorem. Wyznacza to funkcja *b*. Pomyślne przesłanie wiadomości (wszystkich bitów) kończy się jej zachowaniem w sensorze. Domyślnie stosowana jest komunikacja typu *hop by hop* – jeśli sensor nie jest domyślnym odbiorcą, przekazuje wiadomość dalej, przechodząc do stanu *Sending Message*. Urządzenie przeszukuje sąsiadów i uruchamia algorytm routingu, który oprócz domyślnego może być reprogramowany w eksperymencie. Równolegle do stanów *Sensing*, *Moving*, *Receiving Message* i *Sending Message* sensor może wykonywać programy instalowane w jego middleware, co prowadzi na przykład do nowych procedur zarządzania otrzymywanymi wiadomościami.

4 Koncepcja rozwinięcia SenseSim o modele wielorozdzielcze

W modelowaniu i symulacji sieci IoT podstawowym założeniem jest odwzorowanie danych o różnej gradacji oraz modelowania w różnej skali i szczegółowości odwzorowania tych samych rzeczywistych obiektów. Dla przykładu pojedynczy sensor, jako obiekt typu HRE (ang. *High-Resolution Entity*), generuje bardzo szczegółowe dane, podczas gdy podsieć kilku sensorów (np. na jednym obiekcie mobilnym) to LRE (ang. *Low-Resolution Entity*) może generować agregaty właściwe na jego poziomie jak i bardziej szczegółowe, aż do pojedynczego sensora włącznie. Zakłada się również konieczność jednoczesnego odwzorowania obiektów na różnych poziomach rozdzielczości z możliwością ich wzajemnych oddziaływań, dynamicznej (tzn. w trakcie trwającej symulacji) zmiany poziomu rozdzielczości obiektów (agregacja lub deagregacja) wyzwalanej przez użytkownika bądź zdarzenia w symulatorze, a także projekcji przebiegu zjawiska (w sensie przebiegu procesów symulacyjnych) na więcej niż jednym poziomie rozdzielczości.

W literaturze [11, 12] spotyka się wiele metod modelowania wielorozdzielczego, dla których uogólnieniem klasyfiacyjnym może być diagram z rysunku 4.



Rys. 4. Klasyfikacja metod modelowania wielorozdzielczego. Źródło: opracowanie własne

Fig. 4. The classification of multiresolution modelling methods: Source: own preparation

W opisywanym podejściu zaproponowana została ontologia dla realizacji hierarchii modeli oraz funkcji zależności i przejść między modelami. Metamodel, zapisany w ontologii, służy transformacji danych o różnych rozdzielczościach. Dzięki przystosowanemu do ontologii językowi OWL można ułożyć obiekty w hierarchii, zdefiniować relacje między atrybutami stanów, a także zastosować mechanizmy wnioskujące. Skutkiem ontologii jest możliwy kompletny oraz spójny opis struktur klas i ich relacji, realizowalna jest integracja modeli różnych dziedzin, stosowalny jest aparat wnioskowania regułowego z wykorzystaniem logik opisowych, istnieje automatyczna weryfikacja spójności i klasyfikacja, natomiast operacje na danych realizuje się z wykorzystaniem języków zapytań RDQL oraz RQL. Ponadto powszechnie stosowane podejście obiektowe jest uszczegółowione przez dodatkowe zależności wielorozdzielcze (kontekst wystąpienia danych i zależności między poziomami). Spotykane podejścia do agregacji i deagregacji stosują reguły transformacji implementowane na stałe w oprogramowaniu, razem z definicją metod (algorytmów). Brak możliwości ponownego użycia takich komponentów jest istotną wadą. W proponowanym podejściu zaletą ontologii jest oddzielenie reguł definiujących użycie metody od jej implementacji programowej. Reguły zdefiniowane zostały w ontologii, a implementacje metod znajdują się w wyróżnionym module programowym. Typy obiektów reprezentowane są w OWL Class Hierarchy, reguły wielorozdzielczości przez OWL Restrictions, wielorozdzielcza struktura przez OWL Object Property, wreszcie reguły i metody transformacji mają odwzorowanie w OWL Class Hierarchy/ OWL Class Restrictions.

5 Podsumowanie

W niniejszym opracowaniu przedstawiony został model symulacyjny sieci sensorowej i Internetu Reczy dla symulatora SenseSim oraz architektura oprogramowania SenseSim. Daje on możliwość symulowania urządzeń połączonych zarówno w sieć sensorową jak i Internet Rzeczy. SenseSim został zaprojektowany jako narzędzie do symulacji sieci sensorowej rozumianej jako samoorganizujący się system, a ponadto umożliwia makroprogramowanie sieci, dzięki czemu jest ona programowana jako całość.

Kolejnym aspektem jest modelowanie i symulacja wielorozdzielcza, upraszczające integrację wielu zróżnicowanych rozdzielczością modeli oraz interakcje pomiędzy poziomami, co pozwala symulować na przykład działanie sieci zróżnicowanych urządzeń i sensorów. Zaproponowane podejście oparte na ontologii upraszcza wydzielenie reguł nałożonych jako restrykcje na klasy oraz określających, kiedy i jak użyć metod, a ponadto uszczegóławia modele obiektowe o dodatkowe zależności wielorozdzielcze (kontekst wystąpienia danych i zależności między poziomami). Zwrócić tu jednakże należy uwagę na zidentyfikowane problemy, w tym czasochłonność procesu budowy modelu ontologicznego, znaczną złożoność pojęciową ontologii w porównaniu z podejściem obiektowym (hierarchie, rodzaje i właściwości, rodzaje restrykcji), wreszcie konieczność stosowania klasyfikatorów programowych (np. opartego na zbiorach przybliżonych).

Literatura

1. Najgebauer A., Dyk M.: Sieci sensorowe dla potrzeb pozyskiwania danych w symulacji wielorozdzielczej
2. Hall D. L., Llinas J.: *Handbook of Multisensor Fusion: Theory and Practice*. Second Edition, ed. Martin Liggins, CRC Press, Boca Raton, FL 2009
3. Šerić L., Stipaničev D., Štula M.: Agent Based Sensor and Data Fusion in Forest Fire Observer .In: *Sensor and Data Fusion*, Dr. ir. Nada Milisavljević (ed.), In-Teh, 2009, ISBN 978-3-902613-52-3
4. Schmid S., Wattenhofer R.: Modeling sensor networks. In: *Algorithms and Protocols for Wireless Sensor Networks*, Boukerche, A., John Wiley & Sons, 2008
5. Musznicki B., Zwierzykowski P.: Survey of Wireless Sensor Networks Simulation Tools for Demanding Applications. *International Journal of Grid and Distributed Computing*, Vol. 5, No. 3, September 2012
6. Yuqiang C., Jianlan G. and Xuanzi H.: The research of internet of things' supporting technologies which face the logistics industry, *Computational Intelligence and Security* (CIS), 2010 International Conference on, pp. 659-663
7. Dyk M., Najgebauer A., Pierzchała D.: Model symulacyjny sieci sensorowej na potrzeby systemów fuzji danych, *Symulacja w Badaniach i Rozwoju*, red. Leon Bobrowski, ISSN 2081-6154, vol. 4, no. 4/2013, pp. 202-212
8. Dyk M., Najgebauer A., Pierzchała D.: Augmented perception using Internet of Things, *Information Systems Architecture and Technology: Selected Aspectss of Communication and Computational Systems* (eds. A. Grzech, L. Borzemski, J. Świątek, Z. Wilamowska), pp. 109-118, Oficyna Wydawnicza PW, Wrocław, 2014

9. Dyk M., Najgebauer A., Pierzchała D.: Agent-based M&S of smart sensors for knowledge acquisition inside the Internet of Things and sensor networks, In: *Intelligent Information and Database Systems, Lecture Notes in Computer Science*, vol. 9012, Subseries: Lecture Notes in Artificial Intelligence, red. Nguyen, Ngoc Thanh, Trawiński, Bogdan, Kosala, Raymond, 2015, XXXVI, pp. 212-223
10. Dyk M., Najgebauer A., Pierzchała D.: SenseSim: An Agent-Based and Discrete Event Simulator for Wireless Sensor Networks and the Internet of Things, *Proceedings of The 2015 IEEE 2nd World Forum on Internet of Things*, pp. 14-16 December 2015, Milano
11. Pierzchała D., Szymański P.: Ontology-based adapter for data of multi-resolution battlefield simulation. *XVIII Workshop of the Society for Computer Simulation*, Poland, 2011
12. Natrajan A., Reynolds P.F., Srinivasan S.: *Guidelines for the Design of Multi-resolution Simulations*. US DoD DMSO, July 1997
13. Bennet B.M., Ho_man D.D., Prakash C.: *Observer Mechanics. A Formal Theory of Perception*, ISBN 0-12-088635-9, Academic Press, 1989
14. Clark B.N., Colbourn C.J., Johnson, D.S.: Unit disk graphs. In: *DiscreteMathematics* 86 (13): 165177, 1990

Streszczenie

W pracy przedstawiono model symulacyjny sieci sensorowej oraz IoT, będące podstawą dla budowy symulatora SenseSim. Pozwala on na zbudowanie heterogenicznej sieci (lub wielu sieci), której sensory obserwują zmieniające się w czasie zjawiska. Zaletami proponowanego modelu są: możliwość szerokiej konfiguracji węzłów, które posiadać mogą wiele zdolności do obserwacji środowiska, oraz łatwa konfigurowalność algorytmów trasowania i komunikacji bezprzewodowej. Symulator zbudowany w oparciu o przedstawiony model będzie również umożliwiał modyfikację zachowania sensorów poprzez ich makroprogramowanie. Dodatkowo proponowane jest rozszerzenie symulatora o modelowanie wielorozdzielcze.

Słowa kluczowe: sieci sensoryczne, IoT, symulacja dyskretna zdarzeniowa

Modeling and simulation IoT network in SenseSim simulator

Summary

The paper presents a simulation model of both sensor network and IoT as a base for the simulator SenseSim. SenseSim allows one to build a heterogeneous network (or networks), whose sensors observe the time-varying phenomena. The advantages of the proposed model are: the possibility of a wide configuration of nodes that can have a lot of capacity for environmental monitoring and easy configurability of both routing algorithms and wireless communications. The simulator built on the presented model will also allows one to modify the behavior of sensors through their macroprogramming. In addition, it is proposed to extend the simulator for multi-resolution modelling.

Keywords: sensor networks, IoT, discrete event simulation