Krzysztof M. BRZEZIŃSKI
INSTYTUT TELEKOMUNIKACJI POLITECHNIKI WARSZAWSKIEJ,
ul. Nowowiejska 15/19, 00-665 Warszawa

O wbudowanych testerach biernych

Dr inż. Krzysztof M. BRZEZIŃSKI

Dr Brzeziński jest adiunktem w Instytucie Telekomunikacji P.W. Specjalizuje się w teorii i metodyce wykorzystywania standardów telekomunikacyjnych oraz formalnych aspektach projektowania systemów, a zwłaszcza weryfikacji i walidacji metodami testowymi. Jest autorem trzech książek (z dziedziny sieci LAN i ISDN) oraz autorem lub współautorem kilkudziesięciu prac publikowanych (rozdziałów, artykułów i referatów) i wielu raportów technicznych.

e-mail: kb@tele.pw.edu.pl

Streszczenie

Rozważa się modele architektury rozproszonego, reaktywnego systemu zdarzeń dyskretnych, do którego zostaje dołączony bądź którego częścią staje się tester bierny. Modele adekwatne dla testowania aktywnego okazują się niespójne w kontekście testowania biernego. Identyfikuje się tu naturę tej niespójności i proponuje się nowy, bardzo prosty model, dzięki któremu tester bierny może zostać potraktowany jak każdy inny komponent systemu, co pozwala na naturalne modelowanie także systemów, w które tester bierny został wbudowany (zagnieżdżony).

Słowa kluczowe: system, architektura, model, poprawność, testowanie.

On embeddable passive testing

Abstract

In this paper a controversial concept of passive testing [2] is discussed, focusing on representation of a passive tester within the architecture of a distributed, reactive Discrete Event System. It is argued that well known architectural models used for paradigmatic active testing (Fig. 1) become deficient when applied to structures with a passive tester (Fig. 2), which undermines the basis for the formal treatment of passive testing. It is shown that such direct re-use either leads to intrinsically inconsistent and unimplementable interpretations (Fig. 2b and e), or requires additional mechanisms (co-location of a tester, Fig. 2c; instrumentation of a tested entity, Fig. 2d) that are restrictive and inconsistent with the specific use patterns of passive testing. A very simple, minimal architectural model is proposed (Fig. 3), in which each system entity is equipped with a pair of input and output ports, and communication channels are members of the Cartesian product of port sets. This model does not allow the inconsistent elements of previous models to be directly expressed, while making explicit their metaphorical interpretation (Fig. 4). It can deal with both external and embedded passive testers (Fig. 5), as they are treated exactly as any other system entity.

Keywords: system, architecture, model, correctness, testing.

1. Wprowadzenie

Zachowanie logiczne systemów dyskretnych DES (*Discrete Event Systems*) można oceniać metodami ujmowanymi w pojęcie formalnego *testowania* [1]. Najogólniej, metody testowania dzieli się na *aktywne* i *bierne*, choć "testowanie bierne" bywa wciąż traktowane jako pewna metafora, przez co pozostaje jedynie niszowym przedmiotem badań [2], stosunkowo słabo poznanym.

W rozważaniach nad testowaniem istotną rolę pełni *model architektury* systemu. Takie modele, opracowane wcześniej dla testowania aktywnego (i uwzględniające jego szczególne cechy) adaptowano następnie do potrzeb testowania biernego, poprzez uzupełnianie ich *ad hoc* o konstrukcje nie-ogólne, niezdefiniowane bądź uznane za zrozumiałe same przez się, jak np. *podsłuchiwanie kanału*. Zostanie tu zaproponowany bardzo prosty i *uniwersalny* model architektoniczny, który jest pozbawiony zidentyfikowanych niespójności i pozwala skutecznie opisywać i analizować struktury, w których występuje tester bierny.

Model powinien być adekwatny do celu jego stosowania i istoty przedmiotu modelowania. Ten przedmiot (to jest testowanie aktywne i bierne, wraz z okolicznościami ich prowadzenia) zostanie tu na wstępie zdefiniowany i pokrótce omówiony.

Testowanie aktywne jest rozumiane jako łączne działanie dwóch różnych systemów: *Systemu testowego* **T**, w uproszczeniu nazywanego *testerem*, oraz *Systemu testowanego* (**Sut** – *System under test*), którego częścią jest obiekt testowany, określany jako *Implementacja testowana* (**Iut** – *Implementation under test*). W tym procesie tester: *(a)* generuje i wysyła do **Sut** pobudzenia testowe; *(b)* obserwuje odpowiedzi; *(c)* analizuje podobieństwa pomiędzy odpowiedziami wynikającymi z treści ustalonego wcześniej wzorca **Ref** a rzeczywiście uzyskanymi; *(d)* wydaje *werdykt* co do behawioralnej poprawności **Iut** względem **Ref**.

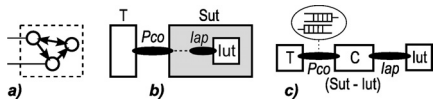
Operacyjnie, testowanie aktywne jest procesem skończonym, realizowanym w ramach kampanii testowej (*test campaign*) i ulokowanym w określonym "punkcie" na trajektorii cyklu życia testowanego systemu. Można przewidywać wiele takich "punktów", ale każdy z nich ma charakter *fazy*, z jej początkiem i końcem [3]. Prowadzenie testów aktywnych systemu już eksploatowanego wymaga odpowiedniego przygotowania go do testów, w tym – *dołączenia* testera. Po zakończeniu kampanii testowej tester jest *odłączany*, a **Sut** działa dalej samodzielnie. W klasycznej telekomunikacyjnej metodyce testowania [4] nie dąży się do wbudowania testera (aktywnego) w system testowany. Przeciwnie, ma on stanowić odrębną, *zewnętrzną* jednostkę i "ma prawo" uzyskiwać dostęp do **Sut** jedynie w przekroju jego roboczych, zewnętrznych interfejsów. W konsekwencji, mechanizmy modelowania i implementowania każdego z tych dwóch odrębnych systemów (**Sut** i **T**) rozwijały się niezależnie.

Testowanie bierne (w którym tester jedynie *rozpoznaje* okoliczności pozwalające na wydanie werdyktu) jest natomiast "bezkampanijne". Tester może towarzyszyć systemowi testowanemu przez cały czas realizacji jego misji użytkowej, więc moment dołączenia (jako wyróżnialna akcja) i odłączenia testera może nigdy nie nastąpić. Werdykty wydawane przez tester bierny w takiej ciągłej misji mogą służyć do nadążnego wykrywania usterek zachowania (*failure*) – wówczas są one przetwarzane *zewnętrznie* (np. przez operatora ludzkiego), bądź jako sygnał sprzężenia zwrotnego w *wewnętrznej* pętli sterowania, podtrzymującej poprawną pracę systemu. Pierwsze z tych zastosowań jest koncepcyjnie zbliżone do problemu znanego jako *DES Diagnosis* [5], a drugie – do *Supervisory Control* [6]. Odpowiednio, tester bierny może występować jako jednostka *zewnętrzna* bądź *wbudowana*.

Przyjmujemy tu następujące rozumienie pojęć "systemu" i "wbudowania". *System* jest, ze swej istoty, obdarzony granicą (*boundary*), którą w określonych punktach (*gate*, *port*, *interface*) mogą przekraczać pewne "interakcje" (wiadomości, sygnały). Ponadto system składa się z rozróżnialnych jednostek (części), będąc w tym zakresie obiektem badań mereologii (dziedziny traktującej o relacjach część-całość). *Jednostka wbudowana* jest po prostu częścią właściwą systemu. *System wbudowany* jest natomiast pojęciem pragmatycznym. Jest to jednostka wbudowana, która posiada własną, wyróżnialną (zwykle dodaje się – ustaloną) funkcjonalność i jest zdolna realizować ją *jako system* także, gdyby została "wydobyta" z miejsca swego wbudowania.

2. Niedostatki dotychczasowych modeli

Na rys. 1(a) przedstawiono podstawowy model systemu rozproszonego, z węzłami odpowiadającymi aktywnym komponentom (jednostkom, *entity*) systemu i krawędziami reprezentującymi możliwość bezpośredniego komunikowania się tych jednostek. Etykiety krawędzi mogą wskazywać nie-topologiczne własności komunikacji, jak zachowywanie kolejności wiadomości.



Rys. 1. Tester aktywny w modelach architektury systemu
Fig. 1. Active tester in architectural models

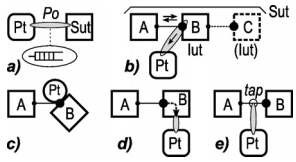
Tester aktywny wprowadza się w dziedzinę modelu, dołączając go do **Sut** w *Punktach sterowania i obserwacji* **Pco** (*Point of control and observation*), definiowanych jako "takie punkty w środowisku testowym, w których zdarzenia testowe mają być powodowane i obserwowane" [7]. W telekomunikacyjnej metodyce testowania [4], powiązanej z technologią języka testów TTCN (*Testing and Test Control Notation*), **Pco** ma własności dwóch nieskończonych, doskonałych, opóźniających kolejek FIFO, po jednej dla pobudzeń testowych i odpowiedzi.

Iut stanowi jedną z mereologicznych części **Sut** i wchodzi w interakcje z innymi częściami w punktach **Iap** (*Implementation access point*), w braku innych ustaleń - także mających własności FIFO (rys. 1b). Oceniane zachowanie zachodzi właśnie w punktach **Iap**, które mogą nie być *bezpośrednio* dostępne dla testera, choć w szczególności **Iap** i **Pco** mogą się zbiegać. Testowanie w sytuacji, gdy między testerem a **Iut** występują pewne struktury pośredniczące, jest określane jako *testing in context* [8] bądź *embedded testing* (co może rodzić nieporozumienia, bo chodzi tu nie o wbudowanie testera, lecz o zagnieżdżenie **Iut** w **Sut**).

Na rys. 1(c) pokazano transformację rysunku (b), uwypuklającą rolę kontekstu. W wyniku jego obecności tester (aktywny) ma jedynie pośredni wpływ na **Iut**, a obserwacja zachowania **Iut** jest zniekształcana: wiadomości mogą zostać zmienione, utracone, odwzorowane w zupełnie inny zbiór, a ich kolejność może nie być zachowywana [2]. Obiekt testowany "w całości" (gdy **Iut** = **Sut**) ma "prawie pusty" kontekst, na który składają się jedynie kolejki FIFO punktu **Pco**, co jest określane jako *queue context* [9].

Pco jest definiowany jako "*punkt* …", a następnie ten punkt jest reprezentowany przez złożoną strukturę, która nie ma punktowej natury – ma bowiem dwa "końce" i "wnętrze". Nie jest wcale oczywiste gdzie, w tej reprezentacji, mają zachodzić zdarzenia. Zdarzenia są elementarnymi składnikami zachowania jednostek; można powiedzieć, że każde zdarzenie ma swego *aktora*. Jednostki umieszczają (*put*) wiadomości w kolejce oraz odczytują (*get*, *read*, *receive*) wiadomości z kolejki, co odpowiada dwóm podstawowym typom zdarzeń: "wysłanie" i "odebranie". Natomiast "przebywanie w kolejce" czy "dotarcie do określonej pozycji kolejki" *nie są* zdarzeniami (co miałoby być ich aktorem?). Zachowanie jednostki składa się wyłącznie ze zdarzeń występujących w jej *portach* tożsamych z końcami kolejek FIFO.

Komunikacja pomiędzy testerem biernym **Pt** a **Sut** jest jednokierunkowa (rys. 2). *Punkty obserwacji* **Po** (*Point of observation*), przez analogię, mogą być przedstawiane jako *pojedyncze* kolejki FIFO, prowadzące do testera. Tak uzyskany przez **Pt** obraz zachowania **Sut** może być zniekształcony przez kontekst, co jest nazywane *niewiernością obserwacji* (*observational infidelity* [2]).



Rys. 2. Tester bierny w modelach architektury systemu
Fig. 2. Passive tester in architectural models

Kwestią zasadniczą jest ustalenie, *w jakim zakresie* i *w jaki sposób* tester może obserwować oceniane zachowanie (inne niż swoje własne). W popularnym w środowisku informatyki teoretycznej modelu komunikacji *synchronicznej*, formalizowanym z użyciem algebr procesów, *obserwacja* jest równoznaczna z *uczestniczeniem* w zdarzeniu. W tym celu tester musi zostać zestawiony (*composed*) z jednostką testowaną, przez co staje się (współ)aktorem zdarzeń. W domenie implementacyjnej, taka kompozycja odpowiada konstrukcyjnemu zintegrowaniu testera z **Sut** (rys. 2c). Tester staje się wówczas wprawdzie jednym z elementów systemu, ale zarazem nie może już zostać z tego systemu "wyjęty" i ponownie użyty jako jednostka *zewnętrzna*. Poszukujemy takich modeli, które są pozbawione tego ograniczenia.

Tester mógłby obserwować zdarzenia zachodzące w portach innej jednostki także bez konieczności poddania się kompozycji z tą jednostką. Tester może być połączony (poprzez **Po**) z portem, do którego jest dołączona także jakaś *inna* jednostka systemu (patrz rys. 2b). "Punkt" **Po** pomiędzy *B* i **Pt** należy do architektury testowej, zaś połączenie pomiędzy *A* i *B* należy do podstawowej architektury systemu, w której obecność testera mogła nie być uwzględniona. Koegzystencja tych dwóch mechanizmów jest wątpliwa. Dążymy do umożliwienia testerowi biernemu "zobaczenia" zarówno zdarzeń odbioru, jak i zdarzeń wysłania, zachodzących w portach jednostki testowanej *B*. Zdarzenie odbioru zachodzi, gdy odpowiedni sygnał jest przez *B* konsumowany – usuwany z końca kolejki *A*-*B*. Nie jest jasne, jak "zniknięcie" sygnału z jednej kolejki miałoby zostać powiązane z automatycznym "pojawieniem się" sygnału w innej kolejce, prowadzącej do **Pt**. Konfiguracja przedstawiona na rys. 2(b), pozornie oczywista, wymaga zastosowania mechanizmów o niejasnym statusie.

Można sprawić, by jednostka **Iut**, w wyniku odpowiedniej *instrumentacji*, raportowała do **Pt**, poprzez odrębny port, zdarzenia zachodzące na jej *innym* porcie (rys. 2d), co zostało określone w [10] jako *communicated observability*. Jednakże przywoływana już metodyka telekomunikacyjna nie zezwala na niezbędną do tego ingerencję w system testowany.

Wreszcie, można próbować zastąpić pojęcie o*bserwowania portów* testowanej jednostki pojęciem *podsłuchu kanału komunikacyjnego* w punkcie, w którym zainstalowano *odczep* (*tap* na rys. 2e). Taki odczep miałby być powiązany z pewną "środkową komórką" kolejki kanału. Jednakże kolejka kanału nie przenosi *zdarzeń*, więc informacja, którą **Pt** mógłby w ten sposób uzyskać, jest typu innego niż oczekiwany. „Odczep" jest ponadto pojęciem obcym dla podstawowego modelu systemu.

Podnoszonym tu wątpliwościom można zarzucić, że są jedynie nominalne (lingwistyczno-terminologiczne), zaś w rzeczywistości "wiadomo", o co chodzi. Jednakże takie domniemane rozumienie jest niewystarczające jako podstawa dla *formalnego* testowania.

3. Model proponowany

Rozproszony system reaktywny *S* definiujemy jako strukturę:

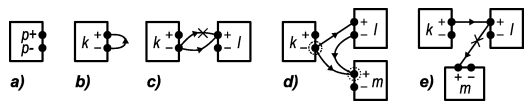
$$S = \left\langle \left(E_i\right)_{i\in\{1,\ldots,N\}}, \varphi \right\rangle, \text{ gdzie } E_k = \left\langle B_k, p_k^-, p_k^+ \right\rangle,$$

złożoną z N *jednostek E* oraz ze zbioru *φ kanałów* komunikacyjnych *c*. Każda jednostka E_i jest wyposażona w dokładnie dwa *porty*: jeden port *wyjściowy* p_i^- i jeden port *wejściowy* p_i^+, przy czym $p_i^- \neq p_i^+$, oraz jest charakteryzowana przez swe zachowanie zewnętrzne (*behaviour*) B_i. Elementarnymi składnikami tego zachowania są *zdarzenia* (*events*), zachodzące w portach tej jednostki: zdarzenia *wysyłania* sygnałów do portu wyjściowego p^- i *odbierania* sygnałów z portu wejściowego p^+. Sygnały odpowiadają *wiadomościom* (*messages*) w domenie systemu modelowanego. Ograniczając liczbę portów do dwóch, kierujemy się zasadami ekonomii (*parsimony*) – dwa przeciwstawne porty są niezbędne i wystarczające do wyrażenia reaktywnego charakteru jednostki.

Dla zbioru P^+ wszystkich portów wejściowych w systemie i zbioru P^- wszystkich portów wyjściowych, kanały komunikacyjne są definiowane jako elementy relacji $\varphi \subseteq P^- \times P^+$. Żąda się przy tym, by $P^- \cap P^+ = \{\}$. Każdy kanał $c = < p_k^-, p_l^+ >$, $c \in \varphi$ łączy port wyjściowy z portem wejściowym pewnych, niekoniecznie różnych jednostek. Port p_k^- jest jego *początkiem*, a p_l^+ – jego *końcem*.

System może być postrzegany jako pozbawiony części właściwych (rys. 3a) – jest wówczas traktowany jako "czarna skrzynka" (*black box*). Nawet wtedy odpowiednie porty są obecne i *mogłyby* zostać połączone (b). Na mocy własności przysługujących zbio-

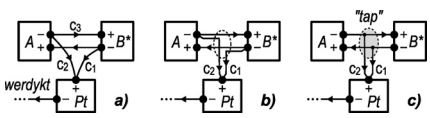
rom (a więc także relacjom), nie istnieją kanały równoległe, łączące tę samą parę portów w tym samym kierunku (c). Może natomiast występować wiele kanałów wychodzących z tego samego portu wyjściowego bądź prowadzących do tego samego portu wejściowego (d). W tym drugim przypadku, sygnały przenoszone przez zbiegające się kanały przeplatają się w końcowym porcie wejściowym – tworzą porządek częściowy, w którym porządek liniowy sygnałów przenoszonych przez każdy kanał *z osobna* jest zachowany (co jest istotne dla, nie rozważanego tu, zachowania B_k poszczególnych jednostek E_k).



Rys. 3. Własności proponowanego modelu
Fig. 3. Properties of the proposed model

Port nie może być użyty równocześnie jako początek i koniec kanałów (e), gdyż z $c_a = <p_1,p_2>$ i $c_b = <p_2,p_3>$ wynika, że $p_2 0 P^+$ i $p_2 0 P^-$, co zaprzecza postulatowi $P^- \cap P^+ = \{\}$. Zarazem uzasadnia to odrzucenie mechanizmu obserwacji pokazanego na rys. 2(b). Taki mechanizm można próbować przedstawiać jak na rys. 3(e), gdzie obiekt $<p_l^+, p_m^+>$ miałby reprezentować punkt **Po** umożliwiający testerowi biernemu E_m obserwowanie zdarzeń odbioru w porcie *wejściowym* testowanej jednostki E_l. Jednakże, jak już stwierdzono, w omawianym tu modelu taka struktura nie istnieje.

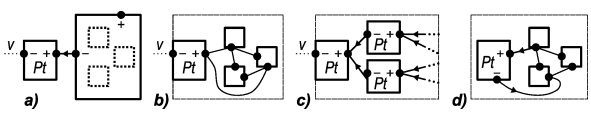
Pokażemy teraz, jak proponowany model wspiera konceptualizację systemu z testerem biernym. Załóżmy, że tester bierny ma testować jednostkę *B*, jak na rys. 4(a). Port wyjściowy testera biernego to miejsce zachodzenia zdarzeń *wydawania werdyktów*. Aby zostały wykorzystane, te werdykty muszą zostać odebrane przez pewną (nie pokazaną tu) "jednostkę postrzegającą" – operatora ludzkiego bądź tester wyższego rzędu.



Rys. 4. Użycie modelu - transformacje
Fig. 4. Use of the model - transformations

Port wejściowy testera biernego służy do odbierania sygnałów wysyłanych przez inne wybrane jednostki systemu w ich portach *wyjściowych*. Odbiór tych sygnałów przez tester jest równoznaczny z "zobaczeniem" zdarzeń ich wysłania (umieszczenia na czele kolejki). Tu, system początkowo zawiera tylko dwie jednostki: *B* pełniącą rolę **Iut** oraz *A* – jednostkę partnerską (*peer*) w stosunku do *B* (mógłby nią też być tester aktywny). Sygnały wysyłane przez *B* do *A* są *także* wysyłane do testera poprzez kanał c_1, dodany w celu "zainstalowania" testera. Z własności modelu wynika, że nie istnieje analogiczny kanał mający swój początek w porcie *odbiorczym* jednostki *B*. Dlatego **Pt** odbiera, poprzez nowy kanał c_2, sygnał wysłany przez *A* do *B*, i na tej podstawie *wnioskuje* o pojawieniu się tego sygnału w porcie wejściowym jednostki *B* (która go, względem chwili wnioskowania, już odebrała bądź odbierze).

Na rys. 4(b) i (c) pokazano kolejne fazy przekształcania konfiguracji (a) do postaci opisującej "podsłuchiwanie" kanału komunikacyjnego. W przypadku (b) wprowadzono czysto geometryczne zniekształcenie. W (c) w końcu pojawia się odczep (*tap*), który jednak nie jest *bona fide* obiektem modelu, lecz pewną metaforyczną reprezentacją przypadku (b). Można te kroki odwrócić, otrzymując formalną interpretację koncepcji odczepu.



Rys. 5. Wbudowywanie testera biernego
Fig. 5. Embedding a passive tester

W omówionym modelu każda jednostka jest traktowana tak samo, więc testery bierne mogą być teraz swobodnie (bez koncepcyjnych przeszkód) wbudowywane w system, jako jego części. Na rys. 5(a) przedstawiono tester bierny *zewnętrzny* w stosunku do systemu (nie-wbudowany). Dla takiego testera, system jawi się *jako całość* – pojedyncza jednostka, która (z definicji) jest wyposażona w jeden port wejściowy i jeden port wyjściowy. Wewnętrzne komponenty systemu są narysowane linią przerywaną: wiadomo, że tam są, lecz ta wiedza jest intencjonalnie porzucana.

W przypadku (b), umyślona granica systemu zostaje przesunięta tak, że tester staje się jego wbudowaną częścią. System może teraz informować środowisko o swym "stanie poprawności". W system może zostać wbudowana cała *sieć* połączonych testerów biernych (c), także zorganizowanych hierarchicznie. Wreszcie, w przypadku (d), sygnały werdyktów wysyłane przez wbudowany tester mogą być używane przez inne jednostki, np. do nadążnego nadzorowania pracy systemu (co jest istotą wspomnianego na wstępie podejścia określanego jako *Supervisory Control*).

4. Podsumowanie

Można by się spodziewać, że studia nad testowaniem biernym, prowadzone od dawna, są ufundowane na mocnej podstawie koncepcyjnej, której istotną częścią jest spójny model architektury systemu. Pokazaliśmy, że tak nie jest – stosowane modele okazują się niedoskonałe, podczas gdy analogiczne modele dla testowania aktywnego nie budzą wątpliwości i nawet zostały poddane standaryzacji [4, 7]. Jako wkład w dążenie do *jednolitego* traktowania testowania aktywnego i pasywnego, zaproponowano tu prosty i implementowalny model architektoniczny, w którym można bezpośrednio i z równą łatwością wyrażać struktury systemów zarówno bez testera, jak i z testerem biernym zewnętrznym bądź wbudowanym. Dla testowania biernego niezbędne są także mechanizmy *dekodowania* oraz opisywania *zachowania* jednostek systemu (w tym – zachowania testera, określanego odpowiednim algorytmem testowania biernego). Zaproponowany model, wraz z odpowiednimi algorytmami testowania i technologią dekodowania [11] tworzą warsztat dla skutecznego konstruowania systemów z testerami biernymi – struktur o dużym potencjale zastosowań w systemach teleinformatycznych i sterujących.

Praca była finansowana, w ramach Projektu Badawczego Zamawianego MNiSW, ze środków na naukę w latach 2007-2010.

5. Literatura

[1] Broy M., Jonsson B. et. al., (ed.): Model-Based Testing of Reactive Systems. LNCS 3472, Springer, 2005.
[2] Brzeziński K. M.: Towards the Methodological Harmonization of Passive Testing Across ICT Communities. W: Soomro S. (ed.): Engineering the Computer Science and IT, rozdział 9, In-Tech, 2009.
[3] Brzeziński K. M.: Testowanie w cyklu życia systemu: nieregularności meta-standaryzacji. W: Krajowe Sympozjum Telekomunikacji i Teleinformatyki, Warszawa, 2009.
[4] ISO/IEC 9646. Conformance testing methodology and framework.
[5] Sampath M., Sengupta R. et. al.: Failure Diagnosis Using Discrete-Event Models. IEEE Trans. Control Systems Technology, 4(2): 105–124, Mar. 1996.
[6] Charbonnier F., Alla H., David R.: The Supervised Control of Discrete-Event Systems. IEEE Trans. Control Systems Technology, 7(2):175–187, Mar. 1999.
[7] ITU-T Z500. Framework on formal methods in conformance testing.
[8] Heerink L., Brinksma E.: Validation in Context. W: Protocol Specification, Testing and Verification XV, Chapman & Hall, 1995.
[9] Verhaard L., Tretmans J. et. al.: On Asynchronous Testing. W: Protocol Test Systems V, North-Holland, 1993.
[10] Brzeziński K. M.: On Common Meta-Linguistic Aspects of Intrusion Detection and Testing. Int. J. Information Assurance and Security (JIAS), 2(3):167–178, 2007.
[11] Brzeziński, K. M.: Towards Practical Passive Testing. W: Parallel and Distributed Computing and Networks, Innsbruck, 2005.

otrzymano / received: 15.10.2010
przyjęto do druku / accepted: 01.12.2010 *artykuł recenzowany*