Piotr BOBIŃSKI, Wiesław WINIECKI
POLITECHNIKA WARSZAWSKA, INSTYTUT RADIOELEKTRONIKI

Środowisko do projektowania oprogramowania rozproszonych systemów pomiarowo-sterujących

Dr inż. Piotr BOBIŃSKI

Adiunkt na Wydziale Elektroniki i Technik Informacyjnych PW. Autor lub współautor ponad 20 publikacji naukowych. Aktualne obszary zainteresowań: nowoczesne technologie komunikacyjne i programowe w skupionych i rozproszonych systemach pomiarowo-kontrolnych, systemy multimedialne, cyfrowe przetwarzanie sygnałów audio.

e-mail:pbo@ire.pw.edu.pl

Dr hab. inż. Wiesław WINIECKI

Prof. nzw. na Wydziale Elektroniki i Technik Informacyjnych PW. Kierownik zespołu Komputerowej Techniki Pomiarowej. Autor lub współautor 4 książek i ponad 120 publikacji naukowych. Obszary zainteresowań: systemy pomiarowe, przyrządy wirtualne, nowoczesne technologie komunikacyjne i programowe w skupionych i rozproszonych systemach pomiarowo-kontrolnych. Prezes POLSPAR, członek Sekcji Systemów Pomiarowych w Komitecie Metrologii PAN, członek IEEE.

e-mail:W.Winiecki@ire.pw.edu.pl

Streszczenie

W referacie przedstawiono opis środowiska do projektowania oprogramowania rozproszonych systemów pomiarowych z uwzględnieniem wykorzystania terminali MDA jako mobilnych klientów systemu. Przedstawiono nowe elementy środowiska LabVIEW w wersji 8.0 istotne z punktu widzenia projektowania systemów rozproszonych oraz przedstawiono technologię serwisów sieciowych – neutralnej platformy komunikacyjnej, stanowiącej podstawę w warstwie komunikacji sieciowej projektowanych systemów rozproszonych. Dalej opisano metodykę projektowania poszczególnych elementów systemu.

Słowa kluczowe: rozproszone systemy pomiarowo-sterujące, serwisy sieciowe, LabVIEW, terminal MDA

Development Environment for Distributed Measurement and Control Systems

Abstract

The paper presents the concept of the software development environment for distributed measurement and control system including MDA terminals as mobile clients. The new features of LabVIEW 8.0, fundamental for development of distributed system are shown. The web services technology is presented as a neutral communication platform. The methodology for designing particular system's modules is shown.

Keywords: distributed measurement and control systems, web services, LabVIEW, MDA terminals

1. Wstęp

Najistotniejszą cechą systemów rozproszonych jest wykorzystanie wielu procesorów i odpowiedni podział zadań dla nich tak, aby spełnione zostały wszystkie założenia projektowe systemu, przy jednoczesnym optymalnym wykorzystaniu dostępnych zasobów. Ze względu na niewielki koszt i ogromny postęp w technologii półprzewodnikowej systemy wieloprocesorowe stają się z dnia na dzień coraz bardziej popularne, a olbrzymi postęp w dziedzinie telekomunikacji sprawia, że rozproszenie elementów systemów jest już niemalże standardem. Z tych samych powodów rośnie również popularność urządzeń typu PDA (*Personal Digital Assistant*). Ich integracja ze współczesnymi technologiami komunikacyjnymi stwarza jeszcze większe pole dla rozwoju nowoczesnych, bezprzewodowych rozproszonych systemów pomiarowych. Wykorzystanie nowoczesnych technologii komunikacji bezprzewodowej, zarówno telefonii komórkowej GSM czy UMTS, jak i technologii Bluetooth czy Wireless Ethernet (WLAN), umożliwiło realizację aparatury pomiarowo-sterującej sterowanej zdalnie, np. z telefonu komórkowego, przenośnego komputera, czy też urządzeń klasy MDA (*Mobile Digital Assistant*).

Opracowanie systemu rozproszonego wymaga szczególnego podejścia do programowania. Dla przykładu, w sieciach czujników, czujniki bezprzewodowe stanowią samodzielne jednostki, które łączą się z innymi znajdującymi się w pobliżu, aby stworzyć strukturę komunikacyjną. Twórcy tego typu technologii stawiają czoła całkowicie nowym wyzwaniom związanymi z programowaniem. Innym przykładem może być projektowanie elektronicznych systemów sterowania pojazdami, inteligentnych telefonów, systemów analizy obrazu, monitorowania procesów przemysłowych, a także tworzenie zautomatyzowanych urządzeń testujących.

Systemy rozproszone wykorzystywane są w wielu różnych branżach i w wielu rozmaitych produktach, lecz wymagania przed nimi stawiane zawsze są bardzo podobne:

- wykorzystanie wielu procesorów opartych na takiej samej lub mieszanej architekturze, m.in. układów FPGA lub procesorów sygnałowych,
- efektywne współdzielenie danych przez wiele procesorów, które są połączone bezpośrednio na jednej płytce drukowanej lub połączone w sieci komputerowej,
- synchronizacja wszystkich węzłów w jednym systemie,
- integracja różnego typu urządzeń np. analogowych i cyfrowych wejść/wyjść, sterowania napędu i analizy obrazu,
- obsługa danych współdzielonych przez rozproszony system, ich rejestracja, integracja z systemami zarządzania przedsiębiorstwa.

Zintegrowanie wielu urządzeń obliczeniowych w system testowania lub sterowania powoduje pojawienie się wielu całkowicie nowych wyzwań związanych z programowaniem, z którymi nie mogą poradzić sobie tradycyjne narzędzia. Dlatego podjęto prace mające na celu analizę rynku i zaproponowanie z jednej strony zintegrowanego, z drugiej jednak rozproszonego środowiska do projektowania i rozwoju oprogramowania rozproszonych systemów pomiarowo-sterujących. W początkowym etapie powstało wiele elementów koncepcji środowiska do zdalnego projektowania rozproszonych systemów pomiarowo-sterujących wykorzystującego technologię serwisów sieciowych (*Web Services*)[1, 2, 4, 5] oraz wykorzystujących przenośne terminale MDA [1, 3]. Pojawienie się wersji 8.0 środowiska LabVIEW spowodowało zmianę koncepcji. Prace poszły w kierunku wykorzystania tego środowiska jako głównego narzędzia do projektowania i zdefiniowania pewnej liczby dodatków (propozycji rozwiązań, zaleceń i modułów programowych) wspomagających wykorzystanie idei

uniwersalnych serwisów sieciowych i coraz bardziej popularnych urządzeń typu MDA. Na zmianę wpływ miało przede wszystkim wprowadzenie do nowej wersji LabVIEW elementów „rozproszonej inteligencji" (*distributed intelligence*).

2. Nowe elementy w LabVIEW 8.0

„Rozproszona inteligencja" w LabVIEW 8.0 to przede wszystkim zestaw wielu zupełnie nowych elementów programowych, zintegrowanych w jedno efektywne środowisko, umożliwiające projektowanie, zarządzanie, uruchamianie i synchronizowanie systemów rozproszonych. Poniżej przedstawiono najistotniejsze, nowe elementy LabVIEW 8.0.

- Programowanie rozproszonych jednostek obliczeniowych, dzięki narzędziom do programowania wielu platform sprzętowych z jednego środowiska. W LabVIEW 8 można napisać program dla komputerów stacjonarnych, urządzeń działających w systemie czasu rzeczywistego, a także układów FPGA. Dodatkowo środowisko zawiera funkcje wymagane do analizy obrazu, sterowania napędem oraz pomiarów, które zwykle wymagałyby użycia odrębnych środków.
- Ułatwiona komunikacja i przesyłanie danych pomiędzy rozproszonymi modułami systemu dzięki jednemu elastycznemu i otwartemu interfejsowi komunikacyjnemu – mechanizmowi zmiennych współdzielonych. Zapewnia on współdzielenie danych pomiędzy różnymi elementami systemu. Zmienne współdzielone mogą przechowywać złożone typy danych wymagane w zaawansowanych rozproszonych aplikacjach, a także wykorzystywać tak istotne funkcje jak skalowanie czy rejestracja i alarmowanie.
- Nawigacja, weryfikacja i wykonywanie programu w węzłach rozproszonych za pomocą okna projektu, które zapamiętuje kod źródłowy i ustawienia dla wszystkich elementów rozproszonego systemu, w tym komputerów osobistych, systemów czasu rzeczywistego, układów FPGA oraz komputerów kieszonkowych. Zawiera ono również wiele narzędzi, które mogą zostać wykorzystane przez zespół programistów, aby zarządzać dużą aplikacją, m.in. kontrolę nad kodem źródłowym,
- Synchronizacja procesorów i urządzeń w systemie poprzez nowe rozwiązanie deterministycznego Ethernetu, zapewniającego niezawodną synchronizację w systemie rozproszonym oraz wyzwalanie czasowe zmiennych współdzielonych.

3. Serwisy sieciowe

Idea Serwisów Sieciowych [6] jest prosta: używanie Internetu do publikowania i dostarczania oprogramowania, z którego może korzystać dowolna liczba zarejestrowanych użytkowników. Serwisy mogą działać w dowolnej dziedzinie - od przeliczania walut, sprawdzania wiarygodności kart kredytowych po zarządzanie produkcją. Aby mogły działać w praktyce, serwisy sieciowe muszą potrafić komunikować się ze sobą niezależnie od języka programu, czy systemu operacyjnego. Tylko wtedy będzie możliwe bezkonfliktowe połączenie wzajemne oraz z już istniejącymi systemami komputerowymi. W skład serwisu wchodzą różne standardy komunikacyjne, które mogą być realizowane za pomocą dowolnych narzędzi oraz przez różnych dostawców.

Trzy główne elementy koncepcji serwisów sieciowych to:

UDDI (*Universal Description, Discovery and Integration*)
Protokół opisu dostępnych składników serwisu sieciowego umożliwiający rejestrację, reklamowanie i automatyczne wyszukiwanie usług. Stanowi on niezależne od platformy, otwarte ramy opisu usług, przedsięwzięć, odkrywania tych opisów oraz integracji różnych przedsięwzięć w Internecie. Protokół jest otwarty, tzn. mimo zaleceń stosowania protokołu SOAP dla dostępu do zarejestrowanych serwisów, nie ma obowiązku jego używania. Dostawca usługi indywidualnie decyduje o rejestracji opisu swojego serwisu, który to serwis wystawia do automatycznego odkrywanie przez innych użytkowników, poszukujących partnerów. Rejestr UDDI jest logicznie zcentralizowany (fizycznie jest rozproszony w sieci) i działa w modelu podobnym do DNS (*Domain Name System*). Użytkownicy używają dowolnego węzła do komunikacji z rejestrem, informacje z rejestru węzłów są replikowane regularnie (np. codziennie), kompletna lista wpisów jest dostępna w każdym węźle.

SOAP (*Simple Object Access Protocol*)
Protokół inicjalizacji konwersacji pomiędzy modułami opisanymi przez UDDI [7]. Moduły (obiekty) rezydujące na zdalnych komputerach udostępniają pewne swoje metody (lub funkcje) do zdalnych wywołań, a aplikacja SOAP tworzy blok żądania dostępu w XML, dostarczając dane na potrzeby wywołania oraz identyfikując lokalizację modułu (obiektu). Jest to więc protokół wymiany informacji w zdecentralizowanym, rozproszonym środowisku informatycznym.
Wiadomości SOAP przesyłane są w tzw. kopertach SOAP, które stanowią pojedynczy dokument XML. Ciało wiadomości (obowiązkowe) zawiera informacje przeznaczone dla końcowego odbiorcy, nagłówek zaś (nieobowiązkowy) zawiera dodatkowe informacje, potrzebne modułom pośredniczącym w wymianie wiadomości (dodatkowa funkcjonalność).
Ze względu na stosowany model żądanie/odpowiedź, semantyka SOAP naturalnie mapuje się w semantykę HTTP.

WSDL (*Web Service Description Language*)
Standard opisu serwisu sieciowego oraz charakterystyki implementacji w języku typu IDL (*Interface Definition Language*) opakowanego w XML [8], czyli XML-owy format opisu serwisu sieciowego, jako zbioru punktów końcowych, operujących komunikatami i zawierającymi dane zorientowane na dokumenty lub procedury. Dokumenty i procedury są opisywane w sposób abstrakcyjny, następnie dowiązane do konkretnego protokołu komunikacyjnego i formatu komunikatów.

Powiązania pomiędzy elementami składowymi serwisów sieciowych można opisać w zwartej formie w następujący sposób. Wpisy w języku WSDL w rejestrach UDDI opisują komunikaty SOAP definiujące konkretny serwis sieciowy. Powiązania te można przedstawić opisując w kilku krokach przypadek użycia serwisu sieciowego:

- dostawca tworzy serwis,
- dostawca opisuje swój serwis w języku WSDL (na potrzeby rejestru UDDI),
- dostawca rejestruje swój serwis w UDDI,
- inny serwis lub użytkownik lokalizuje i żąda dostępu do serwisu przez zapytania do rejestru UDDI,
- inny serwis lub użytkownik tworzy aplikację korzystającą z odnalezionego wcześniej serwisu (komunikacja przez SOAP),
- dane i komunikaty wymieniane są w postaci XML najczęściej po protokole HTTP.

4. Środowisko - elementy i metodyka projektowania

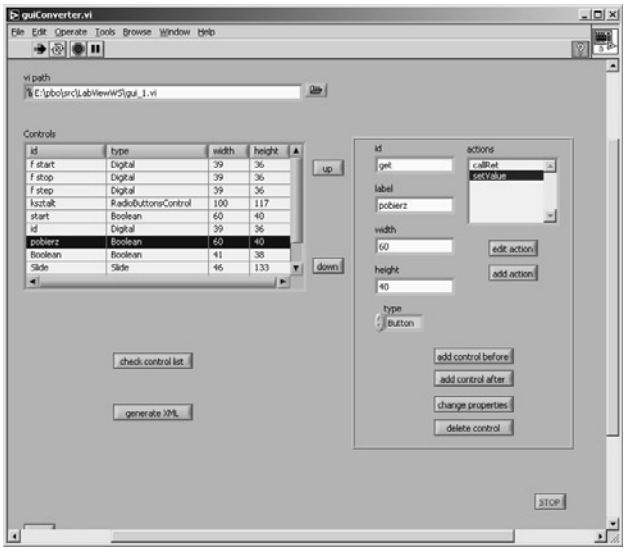
Proponowane środowisko, to środowisko powstałe przez zintegrowanie LabVIEW wykorzystanego do oprogramowania sprzętu z technologią serwisów sieciowych i narzędziami platformy J2ME. Serwer pomiarowo-sterujący sprzężony z silnikiem serwisów sieciowych, poprzez uniwersalne interfejsy usług sieciowych może udostępniać zarówno funkcjonalność pomiarową, obliczeniową, jak i wspomagać tworzenie graficznych interfejsów użytkownika po stronie urządzeń klientów. Odpowiednie oprogramowanie zainstalowane po stronie

mobilnego klienta potrafi na żądanie, automatycznie zbudować interfejs graficzny, zgodny ze specyfikacją otrzymaną od serwisu sieciowego. Podejście to pozwala na zaprojektowanie interfejsu użytkownika od razu w fazie projektowania jądra systemu pomiarowego, a jednoczenie nie ogranicza dostępu do systemu pomiarowego ze standardowych klientów Serwisów Sieciowych. Elementy proponowanego środowiska to:

- LabVIEW w wersji 8.0.
- Dowolne środowisko do projektowania, implementacji i uruchamiania serwisów sieciowych (np. Microsoft .NET wraz z serwerem aplikacyjnym IIS lub Sun Microsystems NetBeans IDE wraz z serwerem aplikacyjnym SAS).
- Platforma Java 2 Micro Edition wraz z pakietem do obsługi serwisów sieciowych (J2ME Web Services Specification)
- Silnik relacyjnej bazy danych (opcja dla systemów, dla których wykorzystanie bazy danych jest wymagane).
- Opracowane przez autorów narzędzia i biblioteki:
  - zestaw graficznych kontrolek interfejsu użytkownika,
  - narzędzia programowe wspomagające tworzenie graficznych interfejsów użytkownika dla przenośnych terminali MDA,
  - narzędzia programowe do automatycznej generacji interfejsu użytkownika po stronie mobilnego klienta,
  - konwerter projektu panelu czołowego z LabVIEW do wymaganego formatu definicji GUI lub dowolny edytor XML do definiowania struktury GUI.

Większość opracowanych narzędzi została przedstawiona w [1] i [3], na rysunku 1 przedstawiono panel czołowy konwertera GUI, prostego narzędzia wspomagającego tworzenie definicji GUI. Narzędzie wykorzystuje możliwości LabVIEW w dostępie do swoich własnych komponentów, a w szczególności możliwość ingerencji w strukturę panelu czołowego. Etapy tworzenia definicji GUI są następujące.

- Utworzenie panelu czołowego korzystając ze standardowego edytora paneli czołowych LabVIEW.
- Uruchomienie narzędzia konwertera z podaniem ścieżki do pliku, w którym zapisano stworzony w poprzednim punkcie panel czołowy.
- Odpowiednie operacje na liście kontrolek, które zmieniają ich kolejność i właściwości, a w szczególności dodają do wybranych kontrolek odpowiednie akcje.
- Wygenerowanie docelowego pliku z definicją.



Rys. 1. Panel czołowy konwertera GUI
Fig. 1. The GUI converter front panel

Poniżej przedstawiono fragment XML-owej definicji GUI, uzyskanej z konwertera, zawierający dwie kontrolki:

- przycisk wyzwalający pobranie wyników pomiaru,
- wykres, na którym wyniki będą prezentowane.

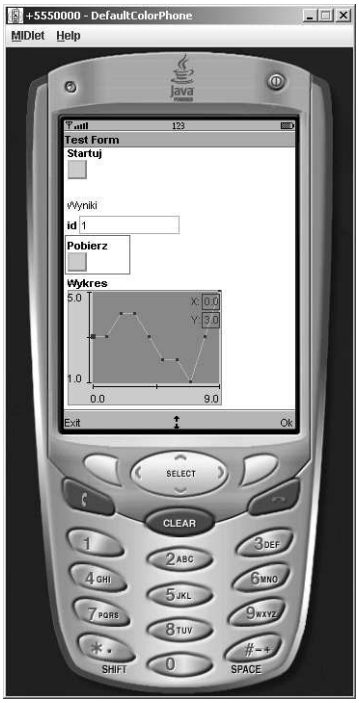
```
=== fragment definicji GUI ===
...
  <item id="get" type="Button" label="Pobierz"
  width="20" height="20">

    <action function="call" method="getResults"
    return="results">
      <param id="id"/>
    </action>

    <action function="setValue">
      <param id="chart"/>
      <param id="results"/>
    </action>
  </item>

  <item id="chart" type="Chart" label="Wykres"
  width="160" height="120"/>
...
=== koniec fragmentu ===
```

Na rysunku 2 przedstawiono przykładowy panel czołowy wygenerowany przez oprogramowanie terminala MDA, na podstawie definicji pobranej z serwisu sieciowego.



Rys. 2. Panel czołowy aplikacji mobilnego klienta (uruchomionej w symulatorze telefonu komórkowego)
Fig. 2. The mobile client application's front panel (running in mobile phone simulator)

Metodykę projektowania rozproszonego systemu pomiarowo kontrolnego można przedstawić w kilku krokach:

- Oprogramowanie wszystkich modułów sprzętowych z wykorzystaniem środowiska LabVIEW w wersji 8.0.
- Projekt graficznego interfejsu dla klientów systemu.
- Opracowanie XML'owej definicji GUI np. z wykorzystaniem opracowanego konwertera paneli czołowych LabVIEW.
- Zdefiniowanie usług sieciowych, które system ma udostępniać oraz implementacja odpowiedniego serwisu sieciowego i sprzężenie go ze środowiskiem LabVIEW, np. z wykorzystaniem jednej z proponowanych metod, przedstawionych dalej.
- Wygenerowanie na podstawie dokumentu WSDL opisującego zrealizowany serwis sieciowy części składowych oprogramowania dla mobilnych klientów, skompilowanie ich do postaci midletu (aplikacji dla przenośnych terminali) i jego instalacja na terminalu MDA.

Dla potrzeb sprzęgnięcia serwera pomiarowego z silnikiem serwisów sieciowych zaproponowano trzy klasy rozwiązań.

- Wykorzystanie zaawansowanych funkcji LabVIEW do implementacji serwisu sieciowego w tym środowisku.
- Wykorzystanie dostępnych narzędzi wspomagających tworzenie serwisów sieciowych w języku Java, np. środowiska NetBeans.
- Wykorzystanie środowiska .NET i wbudowanej do LabVIEW w wersji 8.0 obsługi modelu zdarzeń .NET.

Pierwsze rozważane rozwiązanie polega na wykorzystaniu zaawansowanych dodatków LabVIEW, dostępnych w dodatkowym pakiecie Enterprise Connectivity - Internet Toolkit. Dodatki te to przede wszystkim:
- serwer WWW - G Web Server,
- parser dokumentów XML pracujący w modelu DOM (*Document Object Model*) - pozwalający na tworzenie, czytanie i modyfikowanie dokumentów XML,
- zestaw funkcji CGI (*Common Gateway Interface*) pozwalających na tworzenie dynamicznych stron WWW.

Wykorzystując wyżej wymienione dodatki można stworzyć prosty serwis sieciowy, jednak nie jest to rozwiązanie uniwersalne i elastyczne. Implementacja w środowisku LabVIEW w pełni automatycznego środowiska do tworzenia serwisów sieciowych oraz silnika, który będzie je udostępniał, wydaje się przedsięwzięciem realnym, ale wymagającym ogromnego nakładu sił i środków, niewspółmiernego do efektu końcowego, zwłaszcza w obecności na rynku innych (także darmowych) środowisk specjalizowanych do tego typu zastosowań.

Druga propozycja to wykorzystanie środowiska NetBeans (Sun Microsystems), dostępnego na zasadach otwartej licencji, w wersji z wbudowanym serwerem aplikacyjnym SAS (Sun Application Server) oraz narzędziami wspomagającymi tworzenie serwisów sieciowych. Stworzenie serwisu sieciowego zajmuje w tym środowisku kilka minut, za pomocą dostępnych automatów (*wizards*) tworzy się szkielet serwisu, a następnie wypełnia właściwe ciała udostępnianych metod. Wszystkie pliki konfiguracyjne oraz dokument WSDL opisujący tworzony serwis generowane są automatycznie przez środowisko, a serwis po wdrożeniu go na wbudowany serwer aplikacyjny jest od razu gotowy do świadczenia usług. Do integracji serwisu sieciowego z serwerem pomiarowym można zastosować np. jeden z uniwersalnych protokołów komunikacyjnych lub wykorzystać bazę danych, która stanowić będzie pomost pomiędzy silnikiem usług a serwerem pomiarowym.

Trzeci sposób to wykorzystanie środowiska .NET (Microsoft) oraz wbudowanych do LabVIEW w wersji 8.0 elementów umożliwiających obsługę zdarzeń .NET i wołanie zdalnych metod zgodnie z technologią .NET Remoting umożliwiającą komunikację odpowiedniego modułu z obiektami .NET poprzez uniwersalny protokół (np. SOAP).

Podsumowując rozważania nad metodyką sprzęgania serwerów pomiarowych z silnikami usług sieciowych można dojść do wniosku, iż wybór konkretnego rozwiązania zależeć będzie w dużej mierze od upodobań i doświadczenia programisty implementującego ten element oraz od klasy złożoności tworzonego systemu rozproszonego. Wynika to głównie ze spełnionych w całości założeń technologii serwisów sieciowych dotyczących ich uniwersalności, elastyczności i kompatybilności, zarówno w warstwie platform sprzętowych, jak i programowych.

5. Podsumowanie

W referacie przedstawiono opis środowiska do projektowania rozproszonych systemów pomiarowych z uwzględnieniem wykorzystania terminali MDA jako mobilnych klientów systemu. Opisano metodykę projektowania poszczególnych elementów systemu oraz przedstawiono przykłady realizacji. Główna idea proponowanego środowiska to wykorzystanie środowiska LabVIEW w wersji 8.0 do oprogramowania rozproszonych modułów sprzętowych. W warstwie komunikacji sieciowej zdecydowano się na wykorzystanie neutralnej platformy komunikacyjnej - technologii Serwisów (Usług) Sieciowych. Połączenie tej technologii wraz z narzędziami platformy J2ME pozwoliło na zbudowanie mobilnego klienta systemu pomiarowego, który potrafi automatycznie zbudować swój interfejs graficzny na podstawie specyfikacji dostarczonej przez serwer. W pracy przedstawiono także opracowane narzędzie ułatwiające tworzenie definicji graficznego interfejsu użytkownika.

Praca naukowa finansowana ze środków Komitetu Badań Naukowych w latach 2004-2006 jako projekt badawczy.

6. Literatura

[1] W. Winiecki, P. Bobiński: Automatyczna generacja interfejsu użytkownika dla mobilnych klientów pomiarowych serwisów sieciowych, Pomiary Automatyka Kontrola PAK, no. 6, 2006
[2] P. Wagner, W. Winiecki : Wykorzystanie technologii web-serwisów w rozproszonych systemach pomiarowo-kontrolnych, Pomiary Automatyka Kontrola PAK, no. 9-bis 2006, pp. 169-171.
[3] W. Winiecki, P. Bobiński: Wykorzystanie terminali PDA w bezprzewodowych systemach pomiarowych, Mat. VII Szkoły-Konferencji: Metrologia Wspomagana Komputerowo: MWK 2005, Waplewo, 17-20 Maja 2005.
[4] T. Mielcarz, W. Winiecki: The Use of Web-services for Development of Distributed Measurement Systems, Proc. IEEE Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, (Sofia, Bulgaria, Sept. 5-7, 2005), pp. 320-325.
[5] P. Daponte, C. De Capua,A. Liccadro: A technique for remote management of instrumentation based on web services. Proceedings of the IMEKO-TC-4 13th Symposium., Athens, Greece, 2004, pp. 687-692.
[6] Web Services Architecture, W3C Working Group Note, February 2004.
[7] http://www.w3.org/TR/soap12 , SOAP Version 1.2 specification, W3C Recommendation 24 June 2003.
[8] http://www.w3.org/TR/wsdl , Web Services Description Language (WSDL) 1.1, W3C Note 15 March 2001.

Artykuł recenzowany