Anna PIASKOWY, Roman ŻURKOWSKI
POLITECHNIKA ŚLĄSKA, INSTYTUT METROLOGII, ELEKTRONIKI I AUTOMATYKI

Internetowe metody wizualizacji wyników pomiarów

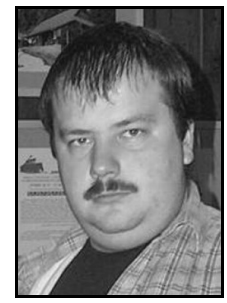
Mgr inż. Anna PIASKOWY

Absolwentka Wydziału Elektrycznego Politechniki Śląskiej w Gliwicach – specjalność Energoelektronika (2006). Obecnie doktorantka w Instytucie Metrologii, Elektroniki i Automatyki Politechniki Śląskiej. Zajmuje się komputerowymi systemami pomiarowymi i pomiarami dokładnymi.

e-mail: anna.piaskowy@polsl.pl

Mgr inż. Roman ŻURKOWSKI

Absolwent Wydziału Elektrycznego Politechniki Śląskiej w Gliwicach – specjalność Automatyka i metrologia elektryczna (1999). Zajmuje się zagadnieniami budowy i analizy metrologicznej komputerowych systemów pomiarowych.



e-mail: roman.zurkowski@polsl.pl

Streszczenie

Niektóre systemy pomiarowe dostarczają wyników, które muszą być prezentowane szerokiemu gronu odbiorców, na przykład przez Internet. W niniejszej pracy zaprezentowano metody stosowane do prezentacji wyników pomiarów w przeglądarkach internetowych. Omówione zostały dostępne metody pobierania wyników pomiarów z serwera, ich przetwarzania (parsowania), a także metody dynamicznego tworzenia grafiki. Zastosowanie metod dynamicznego tworzenia grafiki pozwala zmniejszyć obciążenie procesorów serwera, a także zmniejszyć ilość danych przesyłanych przez łącze internetowe.

Słowa kluczowe: przeglądarki internetowe, grafika dynamiczna, technologia AJAX.

Internet methods for measurement result presentation

Abstract

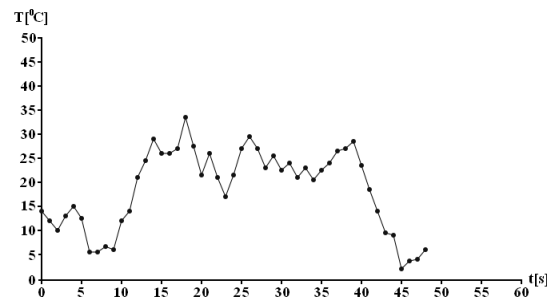
Some measurement systems deliver the results which are presented to many purchasers, for example by the Internet. Methods of measurement results presentation in the Web Browser are presented in this paper. The available methods for downloading the measurement results are discussed, also to parse and to create a dynamic graphics. If the dynamic methods are used to create graphics, then a bias of the server processors is decreased. Also the quantity of transmitted data is decreased.

Keywords: web browsers, dynamic graphics, AJAX technology.

1. Wstęp

W budowanych współcześnie komputerowych systemach pomiarowych coraz częściej zachodzi konieczność prezentacji wyników w nich uzyskanych szerokiej grupie odbiorców. Odbiory mogą używać komputerów pracujących pod kontrolą różnych systemów operacyjnych. Powoduje to konieczność umożliwienia prezentacji wyników pomiarów niezależnie od stosowanego przez odbiorcę systemu operacyjnego. Jednym z rozwiązań jest stworzenie programu do prezentacji wyników dla każdego systemu operacyjnego oddzielnie. Jest to jednak rozwiązanie bardzo pracochłonne i w wielu przypadkach niecelowe. Lepszym rozwiązaniem jest wykorzystanie do prezentacji wyników pomiarów przeglądarek internetowych. Dają one możliwość przedstawiania wyników zarówno w postaci tekstowej (tabele) jak i graficznej (wykresy). W celu prezentacji wyników w oknie przeglądarki internetowej, system pomiarowy uzupełniany jest o serwer WWW dostarczający opis wyglądu strony. Możliwe są dwa rozwiązania sposobu przygotowania takiej strony. W jednym z nich wszystkie dane potrzebne do wyświetlenia strony (teksty i grafikę) dostarcza serwer. Obciąża to w znaczącym stopniu procesor oraz ze względu na konieczność przesyłania dużych ilości danych także łącze internetowe. W drugim rozwiązaniu serwer dostarcza jedynie wyników pomiarów, które są przetwarzane w komputerze i wyświetlane na ekranie monitora odbiorcy.

W niniejszej pracy przedstawione zostały metody dynamicznej prezentacji wyników pomiarów pobieranych z serwera w formacie XML na przykładzie aplikacji do wizualizacji wyników pomiarów z bezprzewodowego systemu monitorowania rozkładu temperatury w pomieszczeniu [1]. Na podstawie danych odbieranych z serwera tworzony jest wykres zmian temperatury przedstawiony na rys. 1. W czasie testu aplikacji wartości temperatur dostarczanych przez serwer zmieniają się szybciej niż w rzeczywistym pomieszczeniu.



Rys. 1. Przykładowy wykres temperatury
Fig. 1. Exemplary temperature chart

2. Struktura danych XML

XML [2] (ang. Extensible Markup Language), to język opisujący dane, czyli metajęzyk. Język XML nie jest kolejnym językiem do przechowywania konkretnych danych. W uproszczeniu można powiedzieć, że XML służy do tworzenia języków do przekazywania informacji. „Rozszerzalność" daje możliwość swobodnego definiowania własnych znaczników organizujących przekazywane dane. Przykładową strukturę danych przekazywanych przez serwer w języku XML przedstawia rys. 2.

```
<?xml version="1.0" ?>
<pomiary>
 <wynik>21.5</wynik>
  .
  .
  .
 <wynik>21.4</wynik>
</pomiary>
```

Rys. 2. Przykładowa struktura danych XML
Fig. 2. Exemplary XML data structure

Pierwsza linia zawiera tzw. prolog. W omawianym przykładzie jest nim standardowa deklaracja XML, której jedynym wymaganym atrybutem jest „version", innym może być standard kodowania znaków („encoding"), jednak biorąc pod uwagę że przekazywane są dane liczbowe nie jest on wymagany. Druga linia zawiera deklarację elementu głównego („pomiary"). W kolejnych liniach serwer przekazuje pobrane z bazy danych wyniki umieszczając je w elementach „wynik".

Wykorzystanie do opisu danych języka XML, pozwala na zastosowanie dostępnych w niżej przedstawionych technologiach funkcji parsowania, co upraszcza proces tworzenia aplikacji.

3. Technologia AJAX

AJAX [3] (ang. Asynchronous JavaScript and XML) jest techniką tworzenia aplikacji internetowych, której dużą zaletą jest rozłożenie pracy po stronie zarówno serwera jak i klienta (przeglądarka internetowa). Proces tworzenia prezentacji wyników pomiarów z wykorzystaniem tej technologii można podzielić na trzy etapy:
1) wysłanie „pytania" i odebranie wyników z serwera,
2) przetworzenie (parsowanie) otrzymanych danych,
3) wyświetlenie wyników w oknie przeglądarki.

Do realizacji dwóch pierwszych zadań konieczne jest utworzenie obiektu XMLHttpRequest. W zależności od używanej przeglądarki internetowej obiekt ten tworzy się w różny sposób. Dla przeglądarek Opera i Mozilla stosuje się zapis:

```
xmlHttp = new XMLHttpRequest()
```

natomiast dla Internet Explorer'a:

```
xmlHttp = new ActiveXObject("Microsoft.XMLHTTP")
```

Po utworzeniu obiektu XMLHttpRequest() ustawiona zostaje właściwość

```
xmlHttp.onreadystatechange = handleRQ
```

co powoduje wykonanie funkcji „handleRQ" za każdym razem gdy obiekt zmieni swój stan (np.: odbierze dane z serwera). Wywołanie metody:

```
xmlHttp.open("GET", "baza.php", true)
```

rozpoczyna dostęp do serwera i wykonanie po jego stronie skryptu „baza.php", pobierającego dane z bazy i tłumaczącego na XML. Kolejnym krokiem jest wywołanie metody:

```
xmlHttp.send(null)
```

powodującej wysłanie zapytania do serwera i oczekiwanie na odpowiedź.

Po odebraniu danych z serwera, wywoływana jest funkcja „handleRQ()". Jeżeli właściwość `xmlHttp.readyState==4` oraz `xmlHttp.status==200` to oznacza, że nastąpiło poprawne odebranie danych z serwera i możliwe jest wykonanie parsowania. Dostęp do odebranych danych uzyskuje się przez wywołanie metody:

```
wyniki = xmlHttp.responseXML.documentElement.
        getElementsByTagName("wynik")
```

powodującej utworzenie tablicy obiektów zawierającej wyniki. Odczyt wartości realizowany jest przez odczytanie właściwości:

```
wynik = wyniki[i].firstChild.data ,
```

gdzie „i" jest indeksem obiektu w tablicy, a zmienna „wynik" przyjmuje wartość przekazaną przez serwer.

Funkcja „handleRQ()" jest wywoływana po każdej zmianie stanu obiektu XMLHttpRequest, w związku z czym realizuje ona również obsługę błędów, które mogą się pojawić w czasie próby odczytania danych z serwera. Funkcja ta możne również informować o aktualnym postępie transmisji danych z serwera.

Po odebraniu i wyodrębnieniu danych można przystąpić do wyświetlenia uzyskanych wyników pomiarów w postaci wykresu liniowego (rys. 1). W tym celu w zależności od używanej przez odbiorcę informacji przeglądarki internetowej konieczne jest zastosowanie różnych metod wyświetlania grafiki. Dla przeglądarek internetowych tj.: Opera czy Mozilla do dynamicznego wyświetlania grafiki wykorzystano obiekt CANVAS. Natomiast dla przeglądarki Internet Explorer wykorzystano VML.

Tworzenie grafiki z wykorzystaniem obiektu CANVAS rozpoczyna się od umieszczenia w opisie strony obiektu:

```
<canvas id="kanwa" width="szerokość"
   height="wysokość"></canvas> ,
```

a następnie uzyskuje się dostęp do tzw. kontekstu (obiektu)

```
ctx = canvas.getContext("2d") .
```

Cały proces tworzenia grafiki odbywa się przez ustawianie właściwości lub wywoływanie metod tego obiektu. Do podstawowych należą:

- ustawienie koloru wypełnienia
  `ctx.fillStyle = "rgb(255,255,255)";`
- rysowanie wypełnionego kwadratu
  `ctx.fillRect (x, y, szerokość, wysokość);`
- ustawienie koloru linii
  `ctx.strokeStyle="rgb(0,0,0)";`
- ustawienie grubości linii
  `ctx.lineWidth=2;`
- początek definicji ścieżki
  `ctx.beginPath();`
- przeniesienie punktu początkowego
  `ctx.moveTo(x0,y0);`
- rysowanie linii
  `ctx.lineTo(x1,y1);`
- wyświetlenie niewypełnionej ścieżki
  `ctx.stroke();`

Ponieważ wykres składa się z wielu linii, metoda ctx.lineTo(x1,y1) powtarzana jest aż do wyczerpania dostępnych wyników pomiarów zwróconych jako odpowiedz serwera.

Tworzenie grafiki z wykorzystaniem VML (ang. Vector Markup Language) rozpoczyna się od wstawienia w nagłówku strony:

```
<html xmlns:v ="urn:schemas-microsoft-com:vml">
```

a w treści strony obiektu:

```
<v:shape style='width:wys.; height:szer.'
stroke="true" strokecolor="black"
 strokeweight="2px" fill="false">
<v:path id=path v=" e "/>
</v:shape>
```

gdzie wszystkie atrybuty dotyczące rysowanego wykresu zdefiniowane są w obiekcie, a funkcja napisana w języku JavaScript modyfikować będzie jedynie właściwość path.value. Opis ścieżki w języku VML tworzy się z wykorzystaniem następujących „poleceń" podawanych jako ciąg znaków:

- m X Y – „move to" przesuwa punkt początkowy,
- nf – „no fill" nie wypełniaj,
- l X Y– „line to" rysuje linie do punktu,
- e – „end" koniec definicji ścieżki.

Polecenia „l" nie trzeba powtarzać, wystarczy wydać je raz, po czym podaje się tyle par współrzędnych X,Y z ilu prostych składa się wykres. Najwygodniejszą formą tworzenia ciągu znaków zawierającego opis ścieżki (wykresu) jest użycie pomocniczej zmiennej „tmp", a po zdefiniowaniu całego opisu ścieżki przepisanie tej zmiennej do obiektu v:path

```
path.value = tmp
```

Dla przykładu opis ścieżki dla kwadratu o boku 100 ma postać:

```
m 100 100 nf l 200 100 200 200 100 200 100 100 e
```

4. Adobe Flash – ActionScript

Kolejną metodą na dynamiczne tworzenie grafiki jest wykorzystanie technologii Flash, a konkretnie języka programowania ActionScript. Również i w tej metodzie podobnie jak w AJAX proces tworzenia graficznej reprezentacji wyników pomiarów jest trzyetapowy.

Do pobierania danych z serwera służy obiekt XML() w którym po zdefiniowaniu właściwości:

```
xmlobj.onLoad = wczytanyXML;
```

czyli ustawieniu funkcji, która ma być wykonana po załadowaniu danych wykonuje się metodę:

```
xmlobj.load("baza.php");
```

powodującą uruchomienie po stronie serwera skryptu baza.php pobierającego dane z bazy i wysyłającego odpowiedź w XML.

W funkcji wczytanyXML() następuje wyodrębnienie danych potrzebnych do wyświetlenia wykresu. W tym celu z odebranego obiektu XML odczytywana jest właściwość:

```
var wyniki = this.firstChild.childNodes;
```

co powoduje powstanie tablicy zawierającej wyniki pomiarów, które można odczytać z właściwości:

```
var wynik = wyniki[i].firstChild.nodeValue;
```

i wykorzystać do narysowania wykresu.

Aby w ActionScript rozpocząć rysowanie wykresów konieczne jest stworzenie obiektu MovieClip:

```
_root.createEmptyMovieClip ("wykres", 1);
```

następnie na obiekcie wykres wykonuje się metody:

- ustawienie parametrów linii
  `lineStyle (grubość, kolor, pokrywanie);`
- przeniesienie punktu początkowego linii
  `moveTo( X , Y );`
- rysowanie linii do punktu
  `lineTo( X, Y);`

Analogicznie jak w przypadku CANVAS również w tej metodzie funkcję lineTo() powtarza się, aż do wykreślenia wszystkich wymaganych prostych.

5. JAVA

Język Java należy do języków obiektowych. Programowanie sprowadza się do tworzenia obiektów i używania ich metod. Podczas korzystania z danej klasy należy dołączyć bibliotekę, w której jest zdefiniowana:

```
import java.awt.*;
```

Do pobrania danych z serwera wykorzystać można klasę URL (Uniform Resorce Lokator), która zawiera metody umożliwiające nawiązanie połączenia z hostem oraz wymianę danych. Należy zdefiniować obiekt klasy URL

```
URL url = new URL(„baza.php”);
```

następnie po stronie serwera wykonywany się skrypt, a dane wynikowe w postaci XML umieszczane są w strumieniu danych:

```
InputStrreamReader in = new InputStream-
        Reader(url.openStream());
```

Dane będą przetwarzane za pomocą metod klasy BuferReader:

```
BufferReader br = new BufferedReader(in);
```

Odczytane z bufora dane przetworzyć można przy pomocy metod klasy String:

```
String wyniki = br.readLine();
```

Alternatywnie dane z XML można odczytać korzystając z biblioteki JAXB (Java Architecture for XML)

```
JAXBContext jc = JAXBCon-
text.newInstance(„wyniki”);
Unmarshaller u = jc.createUnmarshaller();
Object wynik = u.unmarshal(is);
```

W Javie wykres można narysować korzystając z klasy Graphics. W tym celu należy zdefiniować obiekt

```
Graphics g;
```

a wtedy można używać metod takich jak:

- ustawienie koloru wypełnienia
  `setBackground(Color.WHITE);`
- ustawienie koloru linii
  `g.setColor(Color.BLACK);`
- ustawienie grubości linii
  `g.setStroke(2);`
- rysowanie linii od punktu (x_1, y_1) do punktu (x_2, y_2)
  `g.drawLine(x1, y1, x2, y2)`

Wszystkie zaprezentowane rozwiązania poszczególnych operacji opierają się jedynie na wykorzystaniu podstawowych bibliotek Javy.

6. Porównanie metod

Zastosowanie metod dynamicznego tworzenia grafiki ma na celu ograniczenie obciążenia serwera oraz zmniejszenie ilości transmitowanych danych. Plik graficzny zapisany w formacie GIF zawierający wykres jak na rys. 1. ma objętość ok. 10 KB, natomiast dane w formacie XML potrzebne do narysowania takiego wykresu po stronie klienta (przez przeglądarkę internetową) to ok. 1,2 KB, stąd uzyskuje się ok. 8 krotny spadek ilości przesyłanych danych. Zaproponowana na rys. 2. struktura dokumentu XML nie jest jednak optymalna. Po skróceniu nazw znaczników można uzyskać nawet 12 krotny spadek ilości przesyłanych danych.

W tab. 1 zestawiono obciążenia procesora realizacją wyświetlania otrzymywanych różnymi metodami wykresów. W celu zwiększenia obciążenia procesora pobieranie i wyświetlanie danych przyspieszono do 10 razy na sekundę.

Tab. 1. Porównanie obciążenia procesora
Tab. 1. Processor load comparison

Technologia	Obciążenie procesora P4 3,06 GHz	Obciążenie procesora Celeron 1,3 GHz
AJAX + CANVAS	25 %	95 %
AJAX + VML	13 %	88 %
Flash + Mozilla	10 %	43 %
Flash + Internet Explorer	25 %	42 %
JAVA + Mozilla	2 %	10 %
JAVA + Internet Explorer	2 %	8 %

7. Wnioski

Zastosowanie metod dynamicznego tworzenia grafiki pozwala w znaczącym stopniu zmniejszyć ilość danych przesyłanych przez łącza internetowe. Jednocześnie odciążany jest serwer, który nie musi się „zajmować” tworzeniem grafiki, a jedynie pobiera dane z bazy i przesyła je w formacie XML do odbiorcy. Przy zastosowaniu technologii AJAX konieczne jest tworzenie w różny sposób grafiki dla różnych przeglądarek internetowych. Zastosowanie technologii Flash (ActionScript) lub apletów JAVA pozwala lepiej uniezależnić się od przeglądarki używanej przez odbiorcę informacji, jednak wymaga zainstalowania w systemie dodatkowych programów, ale są one jednak powszechnie dostępne dla wielu systemów.

8. Literatura

[1] A. Piaskowy, R. Bogacz: Bezprzewodowy system monitorowania rozkładu temperatury w pomieszczeniu. Materiały konferencyjne PPM’2008, Wyd. Komisja Metrologii PAN Oddział Katowice, seria: Konferencje nr 12.
[2] E. Castro: Po prostu XML. Wydawnictwo Helion, Gliwice 2001.
[3] C. Darie, B. Brinzarea, F. Cherecheş-Toşa, M. Bucica: AJAX i PHP. Tworzenie interaktywnych aplikacji internetowych. Wydawnictwo Helion, Gliwice 2006.
[4] J. Lott: ActionScript. Receptury. Wydawnictwo Helion, Gliwice 2004.
[5] B. Eckel: Thinking in Java. Wydawnictwo Helion, Gliwice 2006.

Artykuł recenzowany