Tecniche di programmazione e il mondo Travian

di Giancarlo Trevisan
27/11/2006

Introduzione

Inizio 2006 ho scoperto uno dei tanti MMOG (Massive Multiplayer Online Game) in rete, www.travian.it; un gioco di strategia dove: "…combatterai con migliaia di altri utenti. Tutti avranno un unico e solo obiettivo, sviluppare il loro piccolo villaggio e portarlo alla massima Gloria."[1] Non intendo descrivervi il gioco nella sua complessità ma solo una parte che ci porterà all'argomento di questo articolo. Il piccolo villaggio che vi viene affidato lo fate crescere lentamente aumentando la produzione di campi di grano, argilla, legno e ferro, costruendo le strutture del villaggio, dal magazzino, all'officina, dal castello, alle mura e molte altre, e addestrando truppe per la difesa, l'attacco e la conquista. Il mondo in cui cresce è vasto, una griglia definita dal rettangolo (-250, -250)—(250, 250) di cui il tuo villaggio occupa un punto—per attraversarla occorrono giorni. Sebbene alcuni giocatori preferiscono una crescita solitaria, eremitica, la maggior parte entra in alleanze. Le alleanze sono comunità di giocatori uniti per la Gloria, sono strutturate, c'è chi comanda e chi è commandato da sorti democratiche o dittatoriali. Per raggiungere la Gloria è indispensabile una strategia! Eccoci qua, per metterla in atto gli strateghi delle alleanze hanno bisogno di I-N-F-O-R-M-A-Z-I-O-N-I, ad esempio, posizione, capacità dei villaggi nonché il numero di truppe disponibili. Quanto segue descrive come raccogliere, inviare, archiviare e trasformare i dati per renderli informazione utile, vedremo in questo cammino diverse tecnologie tra cui JavaScript, DOM, XMLHTTP, servizi web, ASPX, XML, XSL.

Questo progetto è stato sviluppato con Visual Studio 2005 in VB.Net e lo trovate all'indirizzo http://labs.keyvisions.it/travian.

La raccolta dati dei singoli villaggi

Il primo passo è necessariamente l'estrazione dei dati utili agli strateghi dalle pagine Travian, ed il conseguente invio ad un server per l'archiviazione. Ho pensato di ridurre al minimo lo sforzo necessario da parte dei singoli giocatori per inviare i propri dati, limitandoli ad attivare il menu di contesto di Internet Explorer per poi selezionare una voce piuttosto descrittiva, Invia Report Travian. L'impatto e minimo anche a livello di installazione nel client.

DOM (Document Object Model)

I dati che fanno al nostro caso si trovano nelle due pagine riportate in Figura 1, un esame attento del HTML dietro le quinte, abbinato all'utilizzo furbo del DOM[2] da JavaScript, ci consente di estrarre i dati desiderati. Il file `http://labs.keyvisions.it/travian/filereport.htm` contiene il codice che confeziona un distillato di dati per un POST HTTP di tipo `application/x-www-form-urlencoded`. La sintassi del formato POST è la seguente:

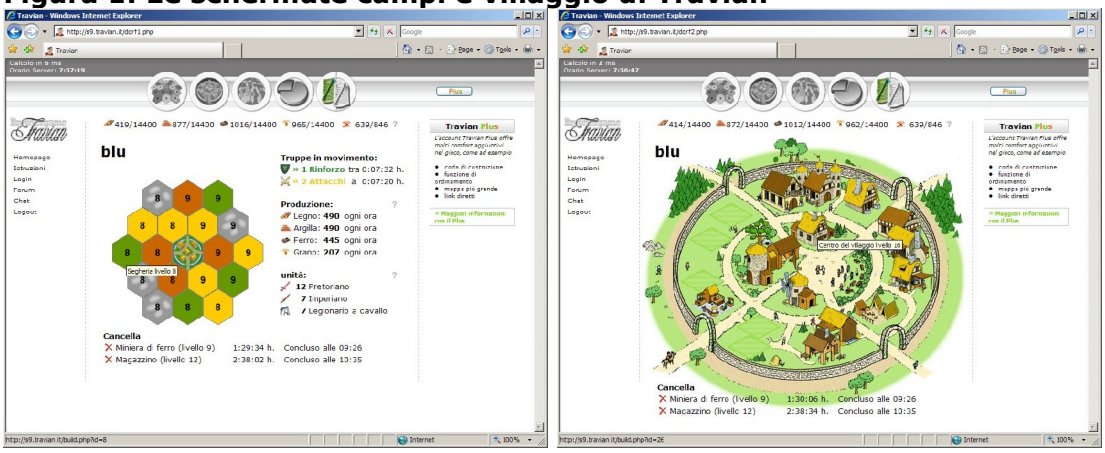
```
[<nome variabile>=<valore>[&<nome variabile>=<valore>]*]
```

Dove <valore> è necessariamente URL encoded come specificato dal tipo di POST (per questo usiamo la funzione JavaScript `escape()`). Il codice in `filereport.htm` è un parser che raccoglie nella stringa `txt` i dati desiderati nel formato seguente:

```
troops=<valore>&buildings=<valore>&village=<valore>&resources=<valore>
```

laddove <valore> è un array, lo codifico come elenco separato da punti e virgola. Il codice completo è nell'appendice A. L'estrazione di dati da una pagina HTML usando DOM è lineare, la furbizia stà nel minimizzare il codice e questo è possibile conoscendo i tag HTML e facendo buon uso dei metodi DOM getElementsByTagName e getElementById.

Figura 1: Le schermate campi e villaggio di Travian



[1] Tratto dalla home page di Travian.it

[2] Document Object Model (DOM) è una forma di rappresentazione dei documenti strutturati, ad esempio HTML, come modello orientato agli oggetti. (wikipedia)

XMLHTTP

Dopo la raccolta, i dati vengono inviati ad un web service usando XMLHTTP[3]. L'implemenatazione è piuttosto semplice e consiste nell'istanziazione dell'oggetto XMLHTTP, quindi l'apertura del canale di comunicazione, l'impostazione del tipo di contenuto (rigorosamente `application/x-www-form-urlencoded`), l'invio del messaggio ed infine la verifica dell'esito.

```
var xmlHttp = (win.XMLHttpRequest) ? new XMLHttpRequest() : new ActiveXObject('Microsoft.XMLHTTP');
xmlHttp.open('POST', 'http://labs.keyvisions.it/travian/service.asmx/FileReport', false, null, null);
xmlHttp.setRequestHeader('Content-Type','application/x-www-form-urlencoded');
xmlHttp.send('url=' + escape(win.location.href) + txt);

if (xmlHttp.status == 200) // Risposta 200 OK
   alert('Rapporto Travian inviato correttamente');
else
   alert('Rapporto Travian non inviato, riprova');
```

Gli ultimi tre parametri del metodo `open` corrispondono a: modalità invio asincrona, utente e password, nel nostro caso prediligo la modalità sincrona con utente e password inesistenti—sarebbe utile parlare di sicurezza. XMLHttpRequest è integrato nei browser di ultima generazione, se il client non dispone di tale browser nota l'istanziazione da ActiveX.

Voce nel menu di contesto di IE

Il meccanismo indolore per avviare la raccolta dati è stato di aggiungere una voce al menù di contesto di Internet Explorer. Niente di più facile, basta aggiungere la chiave seguente al registro di sistema:

```
[HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer\MenuExt\<titolo voce>]
@="<URI> "
"Contexts"=hex:01
```

Nel nostro caso sostituiamo <titolo voce> con Invia Report Travian e <URI> con `http://labs.keyvisions.it/travian/filereport.htm`. Bello, il codice risiede nel server e ciò semplifica gli aggiornamenti.

L'archiviazione dati

I dati inviati via HTTP POST dai singoli giocatori vengono archiviati in un database SQL Server composto da due tabelle senza sfronzoli: tReport e tWorld. tReport raccoglie i dati inviati mentre tWorld raccoglie altri dati piuttosto interessanti messi a disposizione da Travian, ma di questo ne parleremo in seguito.

Il web service: FileReport

Il metodo FileReport del web service ha il compito di importare i dati, inviati tramite HTTP POST dal codice in `filereport.htm`, nella tabella tReport. Non c'è molto da dire a riguardo eccetto un commento sulla tecnica di composizione del comando di inserimento/aggiornamento inviato a SQL Server. Intanto ricordiamoci la forma del POST di tipo form URL encoded:

```
troops=<valore>&buildings=<valore>&village=<valore>&resources=<valore>
```

Il codice sottostante assembla l'istruzione SQL facendo furbo uso del metodo `String.Format`. `For Each` considera una per una le variabili *postate* che guardacaso sono state rigorosamente chiamate come i campi della tabella tReport, omessa la f iniziale[4], notate la sostituzione dei pattern {0} e {1} con stringhe che li contengo e la creazione dei parametri, lo `String.Format` finale sostituisce i pattern rimanenti.

```
.CommandText = "INSERT INTO tReport (f{0}) VALUES (@{1})"
For Each s As String In Context.Request.Form
   .CommandText = String.Format(.CommandText, s + ",f{0}", s + ",@{1}")
   .Parameters.Add(s, Data.SqlDbType.NVarChar).Value = _
      HandleValue(s, Context.Request.Form(s).ToString.Trim())
Next
.CommandText = String.Format(.CommandText, "user", "user")
```

La Mappa di un Server Travian

Leggendo con dedizione il forum Travian, verrete a conoscenza di un file, aggiornato settimanalmente, che contiene l'archivio dei giocatori, a quale alleanza appartengono, la tribu, i villaggi le loro coordinate x e y, e popolazioni. Questo file, `map.sql`, è composto da righe SQL INSERT, una per ogni villaggio, di cui trascrivo un esempio:

```
INSERT INTO `x_world` VALUES (76517,-28,107,1,10425,'blu',10376,'camaleonte',200,'©KOSAKK1',682);
```

[3] XMLHTTP è un set di API che possono essere usate da JavaScript, JScript, VBScript e altri linguaggi di scripting dei browser per trasferire XML o altri dati da e a un web server tramite HTTP. (wikipedia)
[4] Ho preso l'abitudine, ormai da molti anni, di nominare i campi di tabelle con un f iniziale

il codice sottostante mostra come caricare map.sql programmaticamente usando il meccanismo di HTTP request e poi inviare a SQL Server gli INSERT uno per uno. Vengono praticate delle modifiche agli INSERT prima di essere eseguiti, sostituzione del nome della tabella di destinazione, tWorld, e l'aggiunta della colonna nome server.

```
With WebRequest.Create(server & "map.sql")
   .Method = "POST"
   .ContentType = "application/x-www-form-urlencoded"
   .ContentLength = 0

   Dim buffer As New StreamReader(.GetResponse.GetResponseStream, System.Text.Encoding.Default)
   While Not buffer.EndOfStream()
      cmd.CommandText = Replace(buffer.ReadLine().Replace("`x_world`", "tWorld"), ");", ",@server)")
      cmd.ExecuteNonQuery()
   End While
End With
```

La consultazione dati

A questo punto abbiamo dei dati archiviati in un database relazionale, adesso presentiamoli! Siamo ad un crocevia, potremmo creare una pagina ASPX che interroga direttamente le tabelle e genera HTML, oppure, che richiede tramite un web service o direttamente a SQL Server i dati in formato XML per poi trasformarli in HTML con XSL, naturalmente, per un tocco raffinato, coinvolgiamo il CSS.

Trasformazioni XSL lato server

L'XSL trasforma un file XML in un altro file, il .Net Framework 2.0 include la classe `XslCompiledTransform` che esegue questa trasformazione. Il codice sottostante è il nocciolo di `report.aspx`, notiamo la richiesta dei dati in formato XML al web service[5], il caricamento del XSL, quindi la trasformazione inviata a `Response.OutputStream`.

```
Dim xmlDoc As XmlDocument
Dim service As Travian = New Travian() 'Vedi http://labs.keyvisions.it/travian/service.asmx
Dim xslt As New XslCompiledTransform()

xmlDoc = service.XMLReports(Session("server").ToString, Session("ally").ToString)
xslt.Load(Request.PhysicalApplicationPath + "situation.xsl")
xslt.Transform(xmlDoc, Nothing, Response.OutputStream)
```

Scusatemi la precisazione, la trasformazione avviene lato server, non viene inviato un file XML che fa riferimento ad un XSL al client! Sì, certamente, il server in questo caso viene oberato della trasformazione[6], potremmo pertanto caricare il documento XML quindi modificarlo aggiungendo un link allo stylesheet così:

```
xmlDoc = service.XMLReports(Session("server").ToString, Session("ally").ToString)
xmlDoc.InsertBefore(xmldoc.CreateProcessingInstruction("xml-stylesheet",
   "type='text/xsl' href='situation.xsl'"), xmlDoc.FirstChild)
Response.Write(xmlDoc.InnerXml)
```

In questo caso però è opportuno ripensare `report.aspx` e impostare `Response.ContentType = "text/xml"`.

Gli XSL sono molto potenti, questa tecnica viene utilizzata anche per creare una mappa HTML di un server: l'XML è generato dal web method `XMLMap` e XSL è `map.xsl`.

Creazione di un'immagine GIF

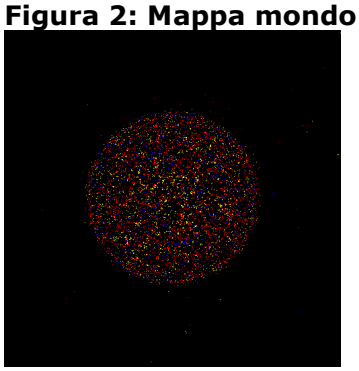
Vediamo infine come creare una semplice immagine GIF programmaticamente. Il codice sottostante interroga la tabella tWorld, colora, a seconda della tribù, un pixel per ogni villaggio e salva la bitmap in un file GIF. Questo codice viene chiamato successivamente all'aggiornamento della mappa.

```
Dim bitmap As New System.Drawing.Bitmap(501, 501)
cmd.CommandText = "SELECT fPositionX, fPositionY, fTribe FROM tWorld WHERE fServer=@server"
With cmd.ExecuteReader()
   While .Read()
      bitmap.SetPixel(250 + .Item("fPositionX") , 250 - .Item("fPositionY"),
         Choose(.Item("fTribe"), Color.Red, Color.Yellow, Color.Blue))
   End While
End With
bitmap.Save(Context.Request.PhysicalApplicationPath + "images\" + Replace(Mid(server, 8), "/", "") +
   ".gif", ImageFormat.Gif)
```

Attenzione alla dimensione della bitmap in pixels, se provate a impostare un pixel al di là dei confini verrà evocata un'eccezione tanto quanto se cercherete di salvare la bitmap in una cartella a cui non siete autorizzati. La figura sottostante mostra una tipica mappa di un mondo di Travian.

[5] I server Travian sono molti, esistono server in molti paesi ed ogni paese ha più di un server: in Italia 11 ad oggi. Il web method XMLReports richiede il nome di un server e un alleanza.
[6] In compenso possiamo supportare browser che non gestiscono XSL

Figura 2: Mappa mondo travian



Conclusione

Ho voluto scrivere questo articolo par dare un esempio. Parlando con laureandi informatici mi sono reso conto della limitata conoscenza pratica in programmazione, nonché della scarsa immaginazione nel coltivarla; per coloro che si accingono ad entrare nel mondo del lavoro, ingegnatevi, aggiungente alla vostra rastrelliera nuove armi e affilate quelle esistenti, mettendole alla prova anche per gioco, vedrete, sarete sempre pronti ad attaccare progetti reali nel mondo che ci circonda.

Appendice

A. http://labs.keyvisions.it/travian/filereport.htm

File di estrazione ed invio dati dalla pagina dei campi e del villaggio.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head><title>File Report</title></head>
<body>
<script type="text/javascript">
// Visita http://support.microsoft.com/kb/177241 per come aggiungere opzioni al menu di contesto
var win = external.menuArguments, numbers = '0123456789';

// Vincola l'invio solo a determinate pagine
if (win.location.href.indexOf('s9.travian.it/dorf1.php') != -1 ||
      win.location.href.indexOf('s9.travian.it/dorf2.php') != -1) {
   var xmlHttp = (win.XMLHttpRequest) ? new XMLHttpRequest() : new ActiveXObject('Microsoft.XMLHTTP');

   if (xmlHttp != null) {
      var usr = null, pwd = null;

      xmlHttp.open('POST', 'http://labs.keyvisions.it/travian/service.asmx/FileReport', false, usr, pwd);
      xmlHttp.setRequestHeader('Content-Type','application/x-www-form-urlencoded');

      // Preleva dati da pagina HTML con DOM
      var txt = '&', tmp, i, doc = win.document;

      for (i=0; i < doc.getElementsByTagName('div').length; ++i) {
         tmp = doc.getElementsByTagName('div')[i].innerText;
         if (tmp.indexOf('unit') != -1) {
            txt += 'resources=';
            tmp = tmp.substr(tmp.indexOf('Prod') + 8) + '; ';
            for (i=0; i < tmp.indexOf('unit'); ++i) {
               if (numbers.indexOf(tmp.charAt(i)) != -1) {
                  txt += tmp.charAt(i);
                  if (numbers.indexOf(tmp.charAt(i+1)) == -1) txt += ';';
               }
            }

            txt += '&troops=';
            tmp = tmp.substr(i + 8) + '; ';
            for (i=0; i < tmp.length; ++i) {
               txt += escape(tmp.charAt(i));
               if (numbers.indexOf(tmp.charAt(i)) != -1 && numbers.indexOf(tmp.charAt(i+1)) == -1)
                  txt += '+';
            }
            txt += '&';
            break;
         }
      }

      var murata = false, flag = false;
      tmp = '';
      for (i=0; i < doc.getElementsByTagName('area').length; ++i)
         if (doc.getElementsByTagName('area')[i].title.indexOf('livello') != -1) {
            murata = (doc.getElementsByTagName('area')[i].title.indexOf('Murata') != -1 ||
                      doc.getElementsByTagName('area')[i].title.indexOf('Palizzata') != -1);
            if (!(murata && flag)) // La murata appare diverse volte, considera solo la prima
               tmp += doc.getElementsByTagName('area')[i].title.replace('livello', '') + ';';
            flag |= murata;
         }
      if (tmp != '') txt += 'buildings=' + escape(tmp) + '&';

      txt += 'village=' + escape(doc.getElementsByTagName('h1')[0].innerText);
      xmlHttp.send('url=' + escape(win.location.href) + txt);

      if (xmlHttp.status == 200)
         alert('Rapporto Travian inviato correttamente');
      else
         alert('Rapporto Travian non inviato, riprova');
   }
}
</script>
</body>
</html>
```

B. Metodo web service FileReport

Questo codice riceve, via HTTP POST, i dati raccolti nelle schermate campi e villaggo e li salva nella tabella tReport in un unico record giornaliero.

```
<WebMethod(Description:="Importa nella tabella tReport i dati inviati da un giocatore")> _
Public Function FileReport() As String
   Dim cmd As New SqlCommand, fId As Object

   Try
      With cmd
         .Connection = New SqlConnection(WebConfigurationManager.ConnectionStrings("DB").ToString)
         .Connection.Open()

         .CommandText = "SELECT fId FROM tReport" & _
            " WHERE fVillage=@fVillage AND (fDateTime >= @fToday AND fDateTime < @fToday+1)"
         .Parameters.Add("fVillage", Data.SqlDbType.NVarChar).Value = _
            Me.Context.Request.Form("village").ToString
         .Parameters.Add("fToday", Data.SqlDbType.DateTime).Value = Now().Date
         fId = .ExecuteScalar()
         .Parameters.Clear()

         'Fields: tReport (fURL, fPage, fUser, fVillage, fResources, fBuildings, fTroops, fActivities)
         If Not IsNumeric(fId) Then
            .CommandText = "INSERT INTO tReport (f{0}) VALUES (@{1})"

            For Each s As String In Context.Request.Form
               .CommandText = String.Format(.CommandText, s + ",f{0}", s + ",@{1}")
               .Parameters.Add(s, Data.SqlDbType.NVarChar).Value = _
                  HandleValue(s, Context.Request.Form(s).ToString.Trim())
            Next

            .CommandText = String.Format(.CommandText, "user", "user")
            .Parameters.Add("fUser", Data.SqlDbType.NVarChar).Value = _
               Me.Context.Request.LogonUserIdentity.Name
         Else
            .Parameters.Add("fId", Data.SqlDbType.BigInt).Value = DirectCast(fId, Integer)

            .CommandText = "UPDATE tReport SET f{0}=@{1} WHERE fId=@fId"

            For Each s As String In Context.Request.Form
               .CommandText = String.Format(.CommandText, s, s + ",f{0}=@{1}")
               .Parameters.Add(s, Data.SqlDbType.NVarChar).Value = _
                  HandleValue(s, Me.Context.Request.Form(s).ToString.Trim())
            Next

            .CommandText = String.Format(.CommandText, "DateTime", "DateTime")
            .Parameters.Add("fDateTime", Data.SqlDbType.DateTime).Value = Now()
         End If

         .ExecuteNonQuery()
         .Connection.Close()
      End With

   Catch ex As Exception
      Return ex.Message
   End Try

   Return "200 OK"
End Function
```