

## .NET Framework Developer Center

Procurar no .NET Framework com o Bin


Brasil (Português) Sign in

[Home](#) [Biblioteca](#) [Aprender](#) [Downloads](#) [Fóruns](#)



.NET Framework Developer Center > Fóruns do .NET Development > Windows Communication Foundation - WCF > Melhores Praticas - WebService

## Fazer uma Pergunta

Pesquisar Fóruns:  

## Melhores Praticas - WebService



**Bruno Velaz - MCTS, MCITP, CanalSharePoint**

quarta-feira, 24 de janeiro de 2007 12:49

0

[Entrar para Votar](#)

Pessoal tenho o seguinte cenário.

Fiz uma aplicação distribuida.

Sendo que Todas as regras de negocio estao em projetos BS(business) e DS(data source) estao em outro projeto.

Entao criei uma aplicação, WebService, para disponibilizar estas informacoes.

USei como parametros de entrada e saida, DataSetTipados.

O Caso que tem alguns métodos que retornam um volume muito grande de dados.

Nelas contém dados Imagens e muitos dados tb, tendo tamanho em torno de 500K as vezes.

Infelizmente não tem o que fazer pois só as imagens são em torno de 150K cada uma.

A pergunt5a é seguinte.

Gostaria de saber o que posso fazer para diminuir o tráfego da rede, pq quando a referencia é na mesma rede fica bom.

Porém quando alguém está usando ADSL para usar os métodos entao gera ai o problema !!!

Demora muito para baixar.

Ve artigos para Fazer compressao dos dados

[http://www.linhadecodigo.com.br/artigos.asp?id\\_ac=700](http://www.linhadecodigo.com.br/artigos.asp?id_ac=700)

Porém quero antes de implementar sugestões e discutir esta ou outras práticas.

Obrigado

Abraços

Movido [Caio Proiete \[MVP\]](#) MVP , Moderador *domingo, 21 de fevereiro de 2010 14:19*  
Movido para o f&#243;rum apropriado &#40;De&#58;.NET Development - Geral&#41;

[Responder](#)

[Citação](#)

## Precisa de Ajuda com Fóruns? (FAQ)

### Meus Links para Fóruns

[Entrar para Fóruns](#)

[Página Inicial dos Fóruns](#)

[Procurar Usuários de Fóruns](#)

### Tópicos relacionados

= Não Respondido      = Respondido

[Quais as melhores práticas para um WebServices extremamente ...](#)  
[WebService](#)

[Menu do portal Sharepoint em forma de Treeview](#)

[Logar usuario no sharepoint via codigo](#)

[Win Workfow Foundation nas Organizações - Aplicações Reais.](#)

[Importando dados de um TXT para uma lista](#)

[Dicas no desenvolvimento de grandes Sistemas](#)

### Estatísticas

Iniciado em: 24/01/2007

Última Resposta: 22/02/2010

Votos Úteis: 0

Respostas: 4

Exibições: 1.478

## Respostas



**Bruno Velaz - MCTS, MCITP, CanalSharePoint**

quinta-feira, 25 de janeiro de 2007 12:13

0

[Entrar para Votar](#)

Galera depois de algumas conversas adotei a compressao de dados.

Quando enviar dados a para o webservice irei receber ja do formato comprimido, e dai eu irei descomprimir ele e trabalhar normalmente.. e quando irei enviar para o requisitor, irei enviar tb comprimido.

Portanto como ficou padrao comprimido, o cliente usa o descompressor e compressor.

Veja o código.

Detalhe de 500k foi para 40k... a consulta que demorava num adsl 300, uns 4 minutos, agora demora em torno de 3,5 segundos..

```
class Zip
{
    #region "Compactacoes via strings"
    /// <summary>
    /// Compacta a string enviada
    /// Anderson Miranda
    /// </summary>
    /// <param name="dados">valor da string</param>
    /// <returns>byte[]</returns>
    ///
    public static byte[] CompactarString(string dados)
    {
        Encoding encoding = Encoding.UTF8;
        byte[] dad_b = (byte[])encoding.GetBytes(dados);
        return CompressToByte(dad_b);
    }
    /// <summary>
    /// Descompacta os bytes enviados
    /// Anderson Miranda
    /// </summary>
    /// <param name="dados_compactados">dados compactados</param>
    /// <returns>string</returns>
    public static string DescompactaToString(byte[] dados_compactados)
    {
        Encoding encoding = Encoding.UTF8;
        byte[] res = DecompressToByte(dados_compactados);
        string ret = encoding.GetString(res);
        return ret;
    }
    /// <summary>
    /// Faz a compressão de bytes
    /// </summary>
    /// <param name="data">dados</param>
    /// <returns>byte[]</returns>
    public static byte[] CompressToByte(byte[] data)
    {
        MemoryStream output = new MemoryStream();
        GZipStream gzip = new GZipStream(output, CompressionMode.Compress, true);
        gzip.Write(data, 0, data.Length);
        gzip.Close();
        return output.ToArray();
    }
    /// <summary>
    /// Descompressao de Bytes
    /// Anderson Miranda
    /// </summary>
    /// <param name="data">dados</param>
    /// <returns>byte[]</returns>
    public static byte[] DecompressToByte(byte[] data)
```

©

```
{
    //Armazena dados compactados no memorystream
    MemoryStream input = new MemoryStream();
    input.Write(data, 0, data.Length);
    input.Position = 0;
    //Descompacta o Stream
    GZipStream gzip = new GZipStream(input, CompressionMode.Decompress, true);
    //Lê stream compactado e armazena em um novo MemoryStream
    MemoryStream output = new MemoryStream();
    byte[] buff = new byte[64];
    int read = -1;
    read = gzip.Read(buff, 0, buff.Length);
    while (read > 0)
    {
        output.Write(buff, 0, read);
        read = gzip.Read(buff, 0, buff.Length);
    }
    gzip.Close();
    return output.ToArray();
}

#endregion
#region "Compactacoes via arquivos"
/// <summary>
/// Compression/zipa um arquivo
/// Bruno Velaz
/// </summary>
/// <param name="sourceFile">arquivo de origem</param>
/// <param name="destinationFile">arquivo de destino</param>
///
public void CompressFile(string sourceFile, string destinationFile)
{
    // make sure the source file is there
    if (File.Exists(sourceFile) == false)
        throw new FileNotFoundException();
    // Create the streams and byte arrays needed
    byte[] buffer = null;
    FileStream sourceStream = null;
    FileStream destinationStream = null;
    GZipStream compressedStream = null;
    try
    {
        // Read the bytes from the source file into a byte array
        sourceStream = new FileStream(sourceFile, FileMode.Open, FileAccess.Read,
            FileShare.Read);
        // Read the source stream values into the buffer
        buffer = new byte[sourceStream.Length];
        int checkCounter = sourceStream.Read(buffer, 0, buffer.Length);
        if (checkCounter != buffer.Length)
        {
            throw new ApplicationException();
        }
    }
}
```

```
}

// Open the FileStream to write to
destinationStream = new FileStream(destinationFile, FileMode.OpenOrCreate,
FileAccess.Write);

// Create a compression stream pointing to the destination stream
compressedStream = new GZipStream(destinationStream, CompressionMode.Compress,
true);

// Now write the compressed data to the destination file
compressedStream.Write(buffer, 0, buffer.Length);

}

catch (ApplicationException ex)
{
    throw new Exception("An Error occurred during compression:" + ex.Message.ToString());
}

finally
{
    // Make sure we always close all streams
    if (sourceStream != null)
        sourceStream.Close();
    if (compressedStream != null)
        compressedStream.Close();
    if (destinationStream != null)
        destinationStream.Close();
}
}

/// <summary>
/// Faz Descompressao de um arquivo.
/// Bruno Velaz
/// </summary>
/// <param name="sourceFile">arquivo de origem</param>
/// <param name="destinationFile">arquivo de destino</param>
///
public void DecompressFile(string sourceFile, string destinationFile)
{
    // make sure the source file is there
    if (File.Exists(sourceFile) == false)
        throw new FileNotFoundException();

    // Create the streams and byte arrays needed
    FileStream sourceStream = null;
    FileStream destinationStream = null;
    GZipStream decompressedStream = null;
    byte[] quartetBuffer = null;

    try
    {
        // Read in the compressed source stream
        sourceStream = new FileStream(sourceFile, FileMode.Open);

        // Create a compression stream pointing to the destination stream
        decompressedStream = new GZipStream(sourceStream,
```

```
CompressionMode.Decompress, true);

// Read the footer to determine the length of the destination file
quartetBuffer = new byte[4];
int position = (int)sourceStream.Length - 4;
sourceStream.Position = position;
sourceStream.Read(quartetBuffer, 0, 4);
sourceStream.Position = 0;
int checkLength = BitConverter.ToInt32(quartetBuffer, 0);
byte[] buffer = new byte[checkLength + 100];
int offset = 0;
int total = 0;

// Read the compressed data into the buffer
while (true)
{
    int bytesRead = decompressedStream.Read(buffer, offset, 100);
    if (bytesRead == 0)
        break;
    offset += bytesRead;
    total += bytesRead;
}

// Now write everything to the destination file
destinationStream = new FileStream(destinationFile, FileMode.Create);
destinationStream.Write(buffer, 0, total);

// and flush everything to clean out the buffer
destinationStream.Flush();
}

catch (ApplicationException ex)
{
    throw new Exception("An Error occurred during compression:" + ex.Message.ToString());
}

finally
{
    // Make sure we always close all streams
    if (sourceStream != null)
        sourceStream.Close();
    if (decompressedStream != null)
        decompressedStream.Close();
    if (destinationStream != null)
        destinationStream.Close();
}

}

#endregion
}
```

Marcado como Resposta [AndreAlvesLima](#) MVP , Moderador  
sábado, 13 de março de 2010 18:37

[Responder](#)   [Citação](#)

## Todas as Respostas



Carlos dos Santos

MVP

quarta-feira, 24 de janeiro de 2007 22:25

[Entrar para Votar](#)

Bruno,

O ideal é você não usar Dataset neste caso, pois ele são "pesados". Tente substituir o dataset por uma classe e neste caso você passa um array de objetos pelo WS, por exemplo:

```
public class Registro
{
    public int codigo;
    public int nome;
}
```

No webservice:

```
public Registro[] Metodo()
{
    Registro[] reg = new Registro[qtd_itens]

    // processamento

    return reg;
}

[]s,
Carlos.
```

[Responder](#)[Citação](#)

Bruno Velaz - MCTS, MCITP, CanalSharePoint

quinta-feira, 25 de janeiro de 2007 12:13

[Entrar para Votar](#)

Galera depois de algumas conversas adotei a compressao de dados.

Quando enviar dados apara o webservice irei receber ja do formato comprimado, e dai eu irei descomprimir ele e trabalhar normalmente.. e quando irei enviar para o requisitor, irei enviar tb comprimado.

Portanto como ficou padrao comprimado, o cliente usa o descomprisor e compressor.

Veja o código.

Detalhe de 500k foi para 40k... a consulta que demorava num adsl 300, uns 4 minutos, agora demora em torno de 3,5 segundos..

```
class Zip
```

```
{
```

```
#region "Campactacoes via strings"
```

```
/// <summary>
```

```
/// Compacta a string enviada
```

```
/// Anderson Miranda
```

```
/// </summary>
```

```
/// <param name="dados">valor da string</param>
```

```
/// <returns>byte[]</returns>
```

```
///
```

```
public static byte[] CompactarString(string dados)
```

```
{
```

```
Encoding encoding = Encoding.UTF8;
```

```
byte[] dad_b = (byte[])encoding.GetBytes(dados);
```

```
return CompressToByte (dad_b);
```

```
}
```

```
/// <summary>
/// Descompacta os bytes enviados
/// Anderson Miranda
/// </summary>
/// <param name="dados_compactados">dados compactados</param>
/// <returns>string</returns>
public static string DescompactaToString(byte[] dados_compactados)
{
    Encoding encoding = Encoding.UTF8;
    byte[] res = DecompressToByte(dados_compactados);
    string ret = encoding.GetString(res);
    return ret;
}

/// <summary>
/// Faz a compressão de bytes
/// </summary>
/// <param name="data">dados</param>
/// <returns>byte[]</returns>
public static byte[] CompressToByte(byte[] data)
{
    MemoryStream output = new MemoryStream();
    GZipStream gzip = new GZipStream(output, CompressionMode.Compress, true);
    gzip.Write(data, 0, data.Length);
    gzip.Close();
    return output.ToArray();
}

/// <summary>
/// Descompressao de Bytes
/// Anderson Miranda
/// </summary>
/// <param name="data">dados</param>
/// <returns>byte[]</returns>
public static byte[] DecompressToByte(byte[] data)
{
    //Armazena dados compactados no memorystream
    MemoryStream input = new MemoryStream();
    input.Write(data, 0, data.Length);
    input.Position = 0;

    //Descompacta o Stream
    GZipStream gzip = new GZipStream(input, CompressionMode.Decompress, true);

    //Lê stream compactado e armazena em um novo MemoryStream
    MemoryStream output = new MemoryStream();
    byte[] buff = new byte[64];
    int read = -1;
    read = gzip.Read(buff, 0, buff.Length);
    while (read > 0)
    {
        output.Write(buff, 0, read);
        read = gzip.Read(buff, 0, buff.Length);
    }
}
```



```
}

gzip.Close();
return output.ToArray();
}

#endregion

#region "Compactacoes via arquivos"

/// <summary>
/// Compressiona/zipa um arquivo
/// Bruno Velaz
/// </summary>
/// <param name="sourceFile">arquivo de origem</param>
/// <param name="destinationFile">arquivo de destino</param>
///
public void CompressFile(string sourceFile, string destinationFile)
{
    // make sure the source file is there
    if (File.Exists(sourceFile) == false)
        throw new FileNotFoundException();

    // Create the streams and byte arrays needed
    byte[] buffer = null;
    FileStream sourceStream = null;
    FileStream destinationStream = null;
    GZipStream compressedStream = null;

    try
    {
        // Read the bytes from the source file into a byte array
        sourceStream = new FileStream(sourceFile, FileMode.Open, FileAccess.Read,
            FileShare.Read);

        // Read the source stream values into the buffer
        buffer = new byte[sourceStream.Length];
        int checkCounter = sourceStream.Read(buffer, 0, buffer.Length);
        if (checkCounter != buffer.Length)
        {
            throw new ApplicationException();
        }

        // Open the FileStream to write to
        destinationStream = new FileStream(destinationFile, FileMode.OpenOrCreate,
            FileAccess.Write);

        // Create a compression stream pointing to the destiantion stream
        compressedStream = new GZipStream(destinationStream, CompressionMode.Compress,
            true);

        // Now write the compressed data to the destination file
        compressedStream.Write(buffer, 0, buffer.Length);

    }

    catch (ApplicationException ex)
    {
        throw new Exception("An Error ocured during compression:" + ex.Message.ToString());
    }

    finally

```

```
{

    // Make sure we allways close all streams
    if (sourceStream != null)
        sourceStream.Close();
    if (compressedStream != null)
        compressedStream.Close();
    if (destinationStream != null)
        destinationStream.Close();
}

}

/// <summary>
/// Faz Descompressao de um arquivo.
/// Bruno Velaz
/// </summary>
/// <param name="sourceFile">arquivo de origem</param>
/// <param name="destinationFile">arquivo de destino</param>
///
public void DecompressFile(string sourceFile, string destinationFile)
{
    // make sure the source file is there
    if (File.Exists(sourceFile) == false)
        throw new FileNotFoundException();
    // Create the streams and byte arrays needed
    FileStream sourceStream = null;
    FileStream destinationStream = null;
    GZipStream decompressedStream = null;
    byte[] quartetBuffer = null;
    try
    {
        // Read in the compressed source stream
        sourceStream = new FileStream(sourceFile, FileMode.Open);

        // Create a compression stream pointing to the destiantion stream
        decompressedStream = new GZipStream(sourceStream,
            CompressionMode.Decompress, true);

        // Read the footer to determine the length of the destiantion file
        quartetBuffer = new byte[4];
        int position = (int)sourceStream.Length - 4;
        sourceStream.Position = position;
        sourceStream.Read(quartetBuffer, 0, 4);
        sourceStream.Position = 0;
        int checkLength = BitConverter.ToInt32(quartetBuffer, 0);
        byte[] buffer = new byte[checkLength + 100];
        int offset = 0;
        int total = 0;

        // Read the compressed data into the buffer
        while (true)
        {
            int bytesRead = decompressedStream.Read(buffer, offset, 100);
            if (bytesRead == 0)
```

```
break;

offset += bytesRead;
total += bytesRead;
}

// Now write everything to the destination file
destinationStream = new FileStream(destinationFile, FileMode.Create);
destinationStream.Write(buffer, 0, total);
// and flush everythng to clean out the buffer
destinationStream.Flush();
}

catch (ApplicationException ex)
{
    throw new Exception("An Error ocured during compression:" + ex.Message.ToString());
}

finally
{
    // Make sure we allways close all streams
    if (sourceStream != null)
        sourceStream.Close();
    if (decompressedStream != null)
        decompressedStream.Close();
    if (destinationStream != null)
        destinationStream.Close();
}

}

#endregion
}
```

Marcado como Resposta **AndreAlvesLima** MVP , Moderador  
sábado, 13 de março de 2010 18:37

[Responder](#)[Citação](#)**Chico Drummer**

sábado, 20 de fevereiro de 2010 20:02

0

Boa Tarde

Como vc usa essa compressao em seu WebService.

[Entrar  
para  
Votar](#)

EX: no retorno do seu método é um DataSet.. certo?? como comprimir essa data set e enviar para o cliente...

pode me enviar um exemplo..

obrigado!!

---

<http://projetofinal.wordpress.com/>[Responder](#)[Citação](#)**Israel Aece**

segunda-feira, 22 de fevereiro de 2010 10:45

MVP, Moderador

0

[Entrar  
para  
Votar](#)

Boas Bruno,

Se estiver falando de WCF, então tem aqui duas alternativas, que "plugam" um código customizado nas extensões fornecidas pelo WCF:

<http://www.codeplex.com/wcfextensions><http://msdn.microsoft.com/en-us/library/ms751458.aspx><http://www.israelaece.com>[Responder](#)[Citação](#)

2011 Microsoft. Todos os direitos reservados. [Termos de Uso](#) | [Marcas Comerciais](#) | [Política de Privacidade](#) | [Fale Conosco](#)