ASPOSE
Your File Format APIs

CASE STUDY

Aspose.Tasks for .NET Case Study

XenTek Limited
www.xentek.co.uk
www.planrunner.co.uk

Using Aspose.Tasks to Import and export Microsoft Project files

Author: Jonathan Tooth, Managing Director, XenTek Ltd, 22nd August 2014

About XenTek

XenTek is a small software house developing primarily a product called Plan Runner – a tool to aid Project Managers. There is just one employee, Jonathan Tooth who is an accomplished software developer, analyst & consultant with over 30 years of commercial experience ranging from Z80 machine code to C#, VB.Net.

Problem

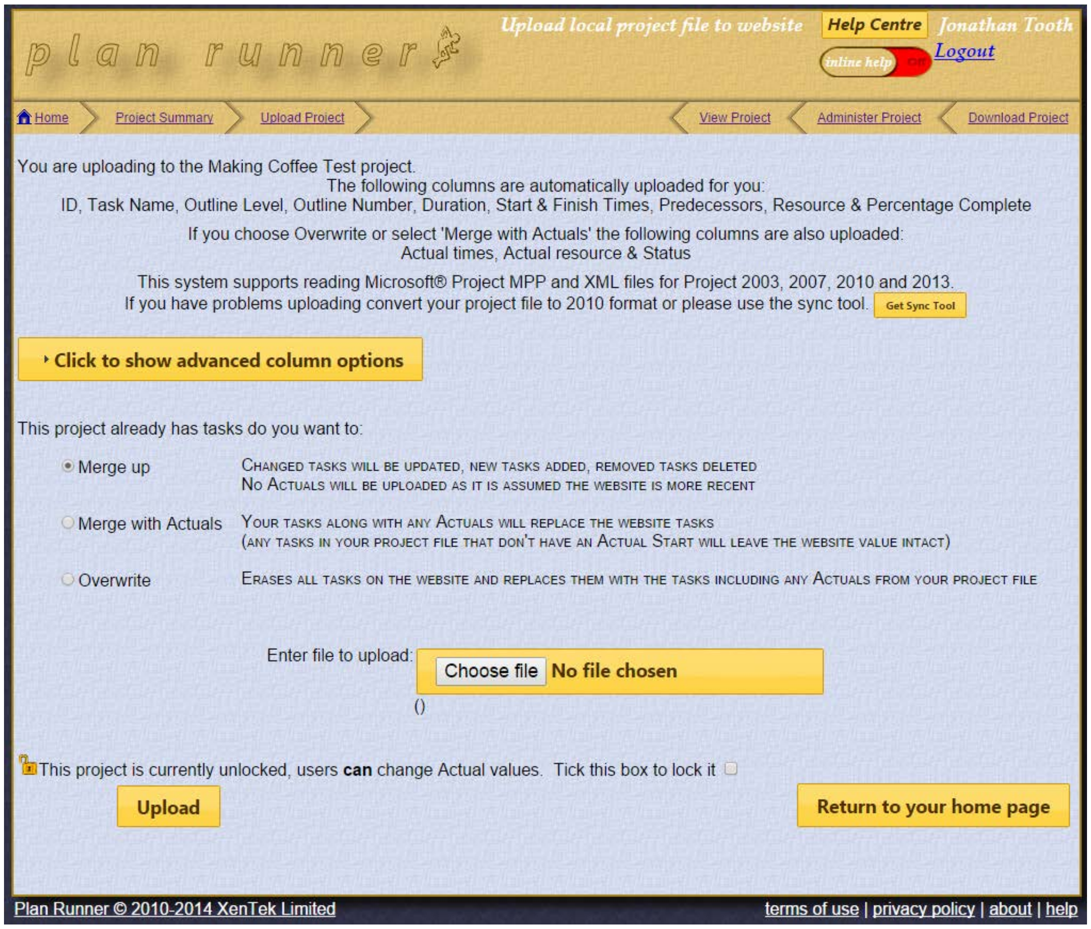
The tool (www.planrunner.co.uk) is a complex piece of software that provides a level of control and visibility of 'running' projects previously unreachable by Project Managers. Vital to this is being able to accept the assets the Project Manager has created. To this end the tool provides two mechanisms to input existing project plans. A sync tool that sits inside Microsoft Project and provides the easiest method to the user and, an import/export tool built into the website for Project Managers who are unwilling or unable to download and install programs from the internet. It is this import/export tool that requires a mechanism to read and write Microsoft project files.

Solution

Three solutions have been trialled over the course of developing PlanRunner, the first two (Microsoft Interop and MPXJ) have been discounted for reasons discussed below. The third solution trialled is that of Aspose.Tasks for .NET. Whilst cost is an issue for a 'Start-up' company; having a commercially developed tool does ensure that it is kept up to date and faults are dealt with and fixed in a timely manner.

PlanRunner users do not directly experience the tool directly; they are presented with an interface as shown below:

Figure 1: Project file upload page



You are uploading to the Making Coffee Test project.
The following columns are automatically uploaded for you:
ID, Task Name, Outline Level, Outline Number, Duration, Start & Finish Times, Predecessors, Resource & Percentage Complete

If you choose Overwrite or select 'Merge with Actuals' the following columns are also uploaded:
Actual times, Actual resource & Status

This system supports reading Microsoft® Project MPP and XML files for Project 2003, 2007, 2010 and 2013.
If you have problems uploading convert your project file to 2010 format or please use the sync tool. Get Sync Tool

Click to show advanced column options

This project already has tasks do you want to:

(● Merge up)	CHANGED TASKS WILL BE UPDATED, NEW TASKS ADDED, REMOVED TASKS DELETED NO ACTUALS WILL BE UPLOADED AS IT IS ASSUMED THE WEBSITE IS MORE RECENT
Merge with Actuals	YOUR TASKS ALONG WITH ANY ACTUALS WILL REPLACE THE WEBSITE TASKS (ANY TASKS IN YOUR PROJECT FILE THAT DON'T HAVE AN ACTUAL START WILL LEAVE THE WEBSITE VALUE INTACT)
Overwrite	ERASES ALL TASKS ON THE WEBSITE AND REPLACES THEM WITH THE TASKS INCLUDING ANY ACTUALS FROM YOUR PROJECT FILE

Enter file to upload: Choose file No file chosen
()

This project is currently unlocked, users **can** change Actual values. Tick this box to lock it ☐

Upload Return to your home page

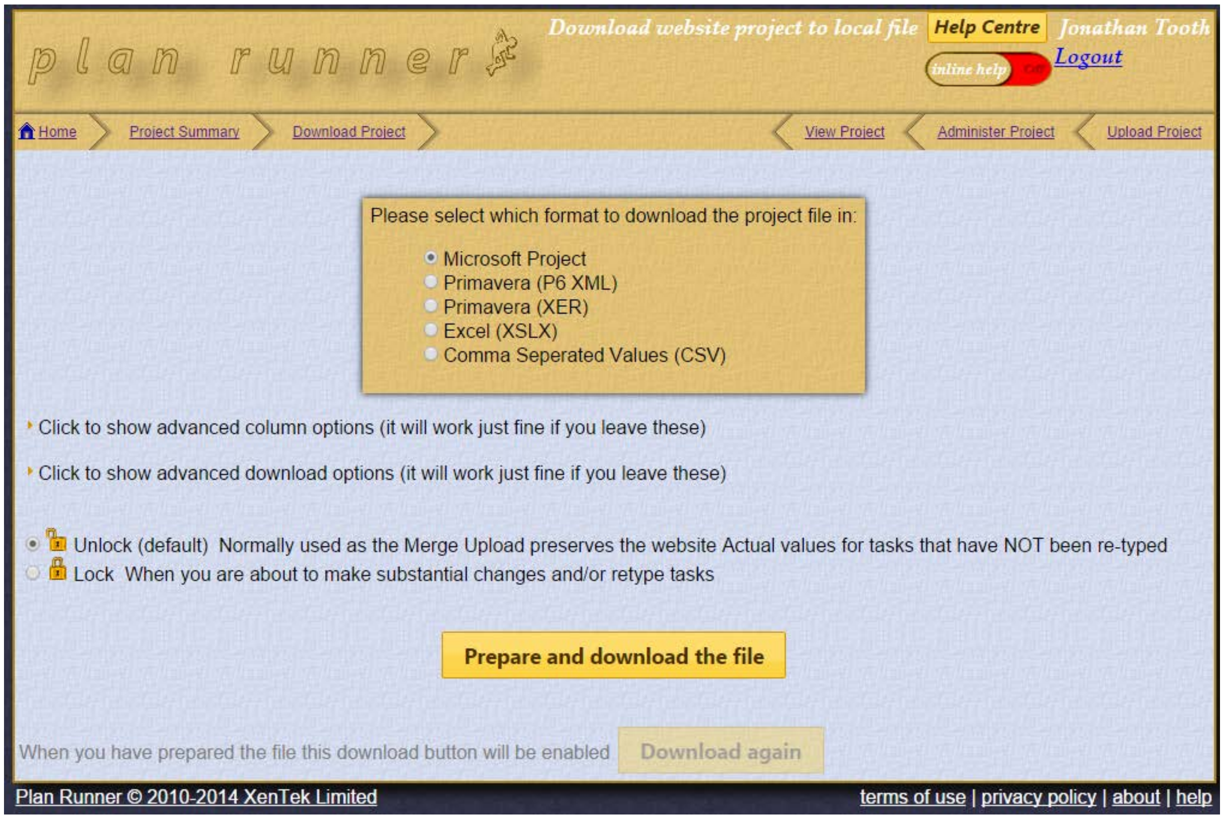
Plan Runner © 2010-2014 XenTek Limited terms of use | privacy policy | about | help

Upload

This facilitates the user to get their project plan into Plan Runner. By simply selecting their project file using the 'Choose file' button and clicking on the 'Upload' button the website uploads and reads their project file, storing selected contents inside a SQL Server database.

The user can control whether the data is merged into the website or erase and overwrite it.

Figure 2: Project file download page


plan runner
Download website project to local file
Help Centre
Jonathan Tooth
inline help
Logout
Home
Project Summary
Download Project
View Project
Administer Project
Upload Project
Prepare and download the file
Download again
Plan Runner © 2010-2014 XenTek Limited
terms of use | privacy policy | about | help

Please select which format to download the project file in:

- [✓ Microsoft Project]
- Primavera (P6 XML)
- Primavera (XER)
- Excel (XSLX)
- Comma Seperated Values (CSV)

Click to show advanced column options (it will work just fine if you leave these)

Click to show advanced download options (it will work just fine if you leave these)

[✓ Unlock (default)] Normally used as the Merge Upload preserves the website Actual values for tasks that have NOT been re-typed

Lock When you are about to make substantial changes and/or retype tasks

When you have prepared the file this download button will be enabled

Download

This facilitates the additional information stored by the website to be inserted into the project file the user originally uploaded. The simplest use is to click the ‘Prepare and download the file’ button which updates the originally uploaded project file and downloads it to the user.

Experience

Finding a solution

Originally the reading and writing of Microsoft project files was performed by Microsoft's own Interop assemblies. This is fine for organisations that are hosting the website internally; however, Microsoft licensing prevents this from being used on public facing websites.

A decision was made to use MPXJ, an open source library which enables reading of Microsoft Project files. Unfortunately this tool would sometimes read Microsoft Project 2013 files incorrectly; this has affected our first customer extensively and is not an acceptable solution.

Aspose.Tasks was trialled to see if it could read and write all of the project files currently available. A free trial reduced financial risk enabling a full POC to be performed. The following criteria were measured to determine suitability of the product:

- Ease of integration
- Reliability of product
- Performance of product
- Quality of help
- Availability of support

Implementation

Effort

Integration of Aspose.Tasks into the Plan Runner import/export mechanism took:

- Coding and unit testing Upload mechanism – 7 hours
- System Testing including bug fixing – 7 hours
- Coding and unit testing Download mechanism – 5 hours
- System Testing including bug fixing – 7 hours
- User Acceptance Testing (of both mechanisms) – 6.5 hours

This results in a total of 32.5 hours.

A rough comparison to the integration of MPXJ is possible. It is favourable to Aspose.Tasks as MPXJ coding was quicker at ~10 hours however testing and debugging took over 40 hours.

Technical Solution

Implementation was made easier by having had two previous mechanisms. While integrating MPXJ the decision was made to isolate all project file reading and writing. A shared library that provided the business logic and connectivity to SQL Server was created along with a ‘Neutral Task’ class that hid tool specific code.

Figure 3: Sample implementation to read Task Name

```
Public ReadOnly Property Name() As String
  Get
    Select Case cGlobals.ActiveImportExportMechanism
      Case cGlobals.ImportExportMechanism.UseMPXJ
        Return _TaskMPXJ.Name
      Case cGlobals.ImportExportMechanism.UseMicrosoft_OfficeInterop
        Return _TaskMSP.Name
      Case cGlobals.ImportExportMechanism.UseAsposeTasks
        Return _TaskAspose.Name
      Case Else
        Return String.Empty
    End Select
  End Get
End Property
```

However some task elements did present in significantly different ways, Predecessors and Duration fields required the most work to convert them into the visual format users expect.

Setting fields is also hidden from the Plan Runner specific code, the three interface methods:

```
Public Function SetFieldValue(ByVal task As Aspose.Tasks.Task, ByVal eFieldName As
AsposeFieldNames, ByVal Value As String) As Boolean

Public Function SetFieldValue(ByVal task As Aspose.Tasks.Task, ByVal eFieldName As
AsposeFieldNames, ByVal Value As DateTime) As Boolean

Public Function SetFieldValue(ByVal task As Aspose.Tasks.Task, ByVal eFieldName As
AsposeFieldNames, ByVal Value As Integer) As Boolean
```

Allow Plan Runner to set any field in a task without needing to understand how it is done.

Note that Plan Runner does not use calendars, resource scheduling or project information so these have not been used or evaluated.

Challenges

Initial provision of the import and export mechanisms was relatively straight forward; the example code supplied on the ExamplesDashboard and online help provided enough help.

A later addition to create a ***new*** project file on download proved more problematical. The forums proved very useful here providing advice and example code the online help didn't contain, for example using a blank project file. Many problems were experienced with Start and Finish times being set incorrectly, Appendix 1 provides some guidance on this.

Outcome

For the most part, a successful outcome was achieved. Importing and exporting to MPP files is fully operational and tested successfully with over 20 different project files from various versions of Microsoft Project. Export to CSV has been rewritten to generate directly from the PlanRunner SQL Server database as going via Microsoft Project gave no real benefit. Export to Excel proved to be problematic, the files generated could not be opened with Excel 2010. It was deemed prudent at this time to disable the facility and revisit when the impact on delaying other development is reduced.

Next Steps

At present the facility is working, over the coming weeks it is expected to have to tweak the mechanism to allow for edge cases and bugs being found. At some point further refactoring of the code is required including the removal of Interop code and integration of MPXJ to cater for formats not present in Aspose.Tasks.

Summary

Overall, the implementation went reasonably smoothly. The online help could be improved with examples that showed the whole process. The product appears robust, performs well and once coded was stable. It coped well with Project files from 2007 to 2013, extensive testing was performed to ensure that the task fields were being read and written correctly. Once users have put a few of their own project files through this new mechanism we expect to be able to fully sign off this part of the system.

I certainly would recommend Aspose.Tasks to anyone requiring read and write access to Microsoft Project files.

Appendix 1 – Recommendations on creating a new task

When creating a new task you may experience problems with the Start and Finish times (and to a lesser extent dates) changing or being reset. The following rules should provide some help in this endeavour.

- Set `Duration` first
- Then set `Work`, `RemainingWork`, `RegularWork` and `Work` again to the duration
- Then set `DurationFormat`
- Then set the `Start`, `ManualStart`, `Finish` and `ManualFinish`

If you assign resources to the task use

```
resource = mppFile.AddResource(resourceName)
```

Don't use

```
Resource = New Aspose.Tasks.Resource
mppFile.Resources.Add(resource)
```

As is discussed in one of the forum entries, Microsoft Project automatically assigns the times and dates to the resource assignment when you link them using Microsoft's interface. When using Aspose.Tasks this needs to be done manually.

Figure 4: How to assign a resource

```
Dim ra As New ResourceAssignment(Me.neutralTask, resource)
ra.Start = Me.neutralTask.Start
ra.Finish = Me.neutralTask.Finish
ra.Task.Work = ra.Task.Duration
ra.RemainingWork = ra.Task.Work
ra.RegularWork = ra.RemainingWork
ra.Work = ra.RegularWork
ra.Finish = Me.neutralTask.Finish
mppFile.ResourceAssignments.Add(ra)
```

Similarly we found that using `AddTaskLink` in preference to `TaskLinks.Add` seemed to create a more stable result.