

TaskJuggler

Un Plan sencillo

Las empresas y los proveedores de servicios no son los únicos que confían en los ordenadores para planear tareas y proyectos. Los clubs y organizaciones sin ánimo de lucro pueden ahorrar mucho tiempo, dinero y dolores de cabeza con la herramienta adecuada. TaskJuggler le ayuda a organizar sus recursos y proporciona una descripción clara.

POR FREDERIK BULSMA

Y PATRICIA JUNG



Las tareas y los proyectos que necesitan de organización han acompañado a la humanidad durante siglos. Lamentablemente, las personas y más recientemente los programas que pueden asignar recursos y tareas de manera eficiente siempre han escaseado. *TaskJuggler* [1] por *Chris Schläger* y *Klaas Freitag* no sólo proporciona esta clase de funcionalidad sino que ha resultado ser una enorme ayuda en optimizar cualquier escenario definido en términos de proyectos y recursos.

El programa permite definir costes y utilidades sencillas de control, con turnos y prioridades. Acepta como entrada un fichero de texto con extensión *.tjp* que describe los objetos. La siguiente sintaxis indica al programa que analice un archivo

```
taskjuggler nombre_fichero.tjp
```

TaskJuggler produce informes en *HTML* o ficheros con formato *CSV* que son perfectos para publicaciones en Internet o para la manipulación de hojas de cálculos en *OpenOffice*. Como alternativa, el programa puede generar datos *XML* que

pueden convertirse a cualquier otro formato permitiendo una migración fácil a su sistema de gestión de contenidos.

Funcionalidad Avanzada

Echémosle un vistazo a la versión de desarrollo 1.9.2, que proporciona nuevas opciones tales como los Informes *CSV* mencionados anteriormente. Además de un compilador *C++*, como *g++* del "GNU Compiler Collection", *GCC*, también será necesario el paquete de desarrollo del sistema *X Windows*. La documentación se encuentra en el sitio web de Taskjuggler, <http://www.taskjuggler.org/>, también se necesitará al menos la versión 3.1 de la biblioteca *Qt*.

Para descomprimir el paquete TaskJuggler hay que introducir lo siguiente

```
tar xjf taskjuggler-1.9.2_2
unstable.tar.bz2
cd taskjuggler-1.9.2_unstable
```

Entonces lanzamos el proceso de compilación introduciendo

```
./configure --with-kde-support
make
```

Puede omitirse la opción *--with-kde-support* si no tiene *KDE*. La rutina de configuración se ha diseñado para proporcionar tanta funcionalidad como la máquina de desarrollo pueda soportar. Como ejemplo, esto regenera los ficheros de documentación si se puede localizar las librerías *Docbook*.

Si *Perl* y los módulos *XML::Parser*, *Postscript::Simple*, *Date::Calc*, *Class::Method Maker* y *Data::Dumper* *CPAN* están instalados, el proceso de compilación también genera un programa llamado *tjx2gantt* el cual crea *diagramas Gantt* como se muestra en la Figura 1.

Después de completar el proceso de *make*, hay que asumir privilegios de administrador antes de instalar TaskJuggler tecleando *make install*.

Proyectos, Turnos y Recursos

Tiene sentido crear un directorio para sus propios proyectos, dándole un lugar a TaskJuggler para almacenar sus definiciones e informes. La página web de TaskJuggler [2] proporciona algunos ejemplos para ayudar a explicar como utilizar la aplicación.

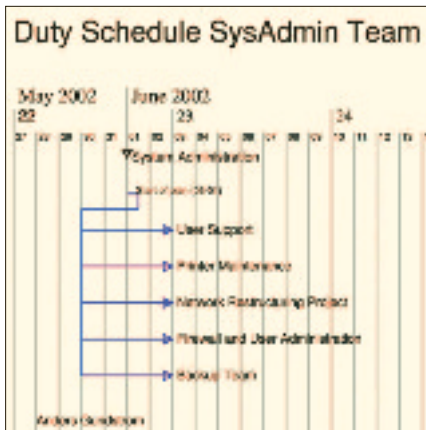


Figura 1: Un diagrama Gantt de un proyecto.

Podemos echarle un vistazo al directorio de los ejemplos en las fuentes que nos hemos bajado. El fichero *ShiftSchedule.tjp* detalla los cuatro componentes del proyecto: La definición, los niveles, los recursos disponibles y las tareas.

Echémosle un vistazo a la sección *project* de este primer ejemplo (Véase el Listado 1). *shifts* proporciona un identificador, que va seguido por el nombre del

1: Definiciones Shift

```
01 project shifts "Duty Schedule
02 SysAdmin Team" "$Id"
03 2002-06-01
04 2002-08-01 {
05     02     dailyworkinghours 8
06     yearlyworkingdays 256
07 }
08 flags hidden
09 shift phonesupport "Phone
10 support" {
11     workinghours mon 9:00
12     - 12:00
13     workinghours tue 9:00
14     - 12:00
15     workinghours wed off
16     workinghours thu 14:00
17     - 17:00
18     workinghours fri 9:00
19     - 12:00
20 }
21 }
22 shift studenthours "Student
23 Hours" {
24     workinghours mon 9:00
25     - 14:00
26 [...]
27     workinghours fri 9:00
28     - 14:00
29 }
30 }
```

proyecto. Este proyecto tiene que ver con los turnos de trabajo para el equipo formado por los administradores de sistemas.

Los últimos parámetros indican la línea temporal del proyecto en el formato AAA-MM-DD. A la hora de definir la fecha de conclusión del proyecto no debería ser muy estricto y permitir definirla con algo de margen, ya que las tareas fuera de la línea de tiempo del proyecto se ignoran.

El tercer parámetro que se necesita es el número de versión. Puede ser un simple "1.0"; si se utiliza un sistema de control de versiones para gestionar los ficheros de TaskJuggler, se puede especificar un comodín ("*\$Id*" para CVS).

Estos parámetros obligatorios pueden llevar detrás opciones detalladas entre paréntesis, como las horas diarias trabajadas (ocho horas en nuestro ejemplo) y el número de días trabajados por año (256 en el ejemplo).

Los objetos *shifts* se pueden usar para definir períodos de turnos de trabajo. Los turnos se usan para acumular la cantidad de tiempo de la gente que trabaja a la vez. El objeto *workinghours* define los períodos de trabajo diarios específicos. Los días se abrevian de la siguiente manera: *tue* para martes, *wed* para miércoles, etc. La palabra *off* indica un día sin horas de trabajo dentro del proyecto. Si se pierde un día, TaskJug-

Listado 2: Definiciones de Recursos

```
01 resource joe "Joe Bughunter"
02 {vacation 2002-06-10 -
03 2002-06-13
04 }
05 }
06 resource khaled "Khaled Safri"
07 {
08     shift studenthours
09 }
10 [...]
11 resource anders "Anders
12 Gundstrom" {
13     maxeffort 0.8
14 }
15 }
16 resource paul "Paul Gutier" {
17     vacation 2002-07-02 -
18 2002-07-08
19 }
20 }
```

ger supondrá las horas de trabajo por defecto del proyecto para este turno.

Cada turno está asignado a un ID (por ejemplo *phonesupport* para el tiempo que los miembros del equipo de administradores de sistemas pasarán dando soporte telefónico) y un título.

Los miembros de la plantilla afectados por el proyecto están indicados por el uso de la palabra *resource* como se muestra en el Listado 2. Además de un descriptor único (como *joe*), el nombre completo es útil como descripción. El atributo *vacation* puede usarse opcionalmente para definir un período vacacional; *maxeffort* define un factor que permite calcular el trabajo a media jornada. En vez de una semana de trabajo de cuarenta horas, Anders Gundstrom sólo trabaja $8 * 0.8 = 6.4$ horas. La palabra *shift* vincula a Khaled Safri con un horario de trabajo especial; en este caso, las horas definidos para los estudiantes en el Listado 1.

Finalmente, lo más importante: que se completen las tareas (ver Listado 3). Nuestro primer ejemplo define una

Listado 3: Definiciones de Tareas

```
01 task sysadmin "System
02 Administration" {
03     # Lo siguiente no es realmente
04     una tarea. Sólo
05     # define la fecha de comienzo
06     del proyecto.
07     task start "Start of plan" {
08         start 2002-06-01
09         milestone
10         flags hidden
11     }
12 }
13 }
14 task usersup "User Support" {
15     depends !start
16     duration 2m
17     shift
18     phonesupport
19     priority 900
20     allocate joe {
21         alternative
22     }
23     anders, khaled, sally select
24     minloaded }
25 }
26 [...]
27 } # Fin de Sysadmin Tasks
```

tarea, *sysadmin* que se divide en trabajos individuales entre llaves. El punto *start* define un hito, donde puede comenzar otra tarea. El soporte de los usuarios depende de la terminación de *start* y no puede suceder antes del 1 de Enero de 2002. El hito no aparecerá en los informes debido a que es una etiqueta oculta.

Ahora podemos asignar las tareas de soporte que van a ser planeadas para los próximos dos meses (*duration 2m*) al turno *phonesupport* con una prioridad de 900. 1 indica algo sin importancia, mientras que 1000 tiene la máxima prioridad. El atributo *allocate* asigna la plantilla a las tareas. Esto significa en este caso seleccionar aquellos miembros de la plantilla de un grupo formado por *joe*, *anders*, *khaled* y *sally* que tienen la carga de trabajo más ligera (*select minloaded*). La selección de criterio *maxloaded* es lo contrario a esto y selecciona los miembros de la plantilla con la carga más pesada. *Order* selecciona el primer miembro de la plantilla sin nada asignado y *random* simplemente coge uno cualquiera.

Planificación Simplificada

Tan pronto como el alcance del proyecto ha sido especificado, TaskJuggler puede proporcionar el planning: Añadiendo la siguiente línea al fichero *.tjp*

Listado 4: Programa Asignado a Sally

```
01 htmlweeklycalendar
   "Calendar-sally.html" {
02     headline "Assignment
   Schedule
03 for Sally"
04     columns schedule
05     hidetask 1
06     hideresource
07 ~isresource(sally)
08 }
```

Listado 5: Vistazo del número de horas de trabajo por pregunta y día

```
01 csvtaskreport "effort.csv" {
02     columns name, daily,
   effort
03     start 2002-06-01
04     end 2002-07-01
05     hidetask hidden
06     loadunit hours
07 }
```

Monday	Tuesday	Wednesday	Thursday
27 May 2002	28 May 2002	29 May 2002	30 May 2002
3 Jun 2002	4 Jun 2002	5 Jun 2002	6 Jun 2002
10 Jun 2002	11 Jun 2002	12 Jun 2002	13 Jun 2002
17 Jun 2002	18 Jun 2002	19 Jun 2002	20 Jun 2002

Figura 2: Calendario HTML de Sally.

A	B	C	D	E	F	G
1	2002-06-01	2002-06-02	2002-06-03	2002-06-04	2002-06-05	2002-06-06
2	System Admin	0	0	2	0	0
3	User Support	0	0	0	0	0
4	Printer Maintenance	0	0	0	0	0
5	Network Administration	0	0	0	0	0
6	Hardware Upgrade	0	0	0	0	0
7	Backup	0	0	0	0	0
8	Documentation	0	0	0	0	0
9	Training	0	0	0	0	0
10	Other	0	0	0	0	0

Figura 3: OpenOffice con los datos cargados desde TaskJuggler.

```
xmlreport "ShiftSchedule.tjx"
```

le dice a TaskJuggler que cree un informe XML y lo almacene en un fichero llamado *ShiftSchedule.tjx*. Por supuesto se puede ver este fichero con un editor XML o una herramienta similar, pero su potencia real queda reflejada cuando se necesita importar o exportar datos. El DTD apropiado está disponible en http://www.taskjuggler.org/show_dtd.php. Puede pasarse el fichero XML al script Perl mencionado antes, *tjx2gant* [3],

para crear un diagrama de Gantt.

Además TaskJuggler está perfectamente capacitado para generar informes HTML útiles sin usar ninguna herramienta adicional. Añadiendo el código del Listado 4 *htmlweeklycalendar* al fichero *.tjp* para el calendario de Sally de la Figura 2 se creará un planning semanal del período del proyecto en *Calendarsally.html*.

La función *isresource(sally)* filtra las asignaciones para el recurso *sally* y *hideresource* oculta cualquier recurso que no (~) coincida con este criterio. *columns schedule* saca un programa detallado usando estos datos. Dejando la línea *hidetask 1* inserta una línea para notas entre la fecha y la tarea en el calendario HTML.

El informe CSV también está definido en el fichero *.tjp*, como se muestra en el Listado 5. *csvtaskreport* proporciona los nombres individuales y *effort* en horas (*loadunit hours* para todas las tareas que no están

ocultas para el período entre 06.01.2002 y 07.01.2002 (ver Figura 3).

Un Plan Diferente

No importa si se necesita informes de quien trabajará, durante cuanto tiempo y en qué, o planes sencillos de asignación para su plantilla, TaskJuggler ofrece un abanico de funciones que podrían fácilmente rellenar un manual de referencia. El directorio *Examples* en el fichero fuente *taskjuggler* proporciona más ejemplos. Este directorio y los ficheros de ejemplo también ofrecen un número de pistas, por ejemplo, en el uso de scripts de macros y proyectos multi-partes.

RECURSOS

- [1] TaskJuggler: <http://www.penguin.org/handbook>
- [2] Proyecto TaskJuggler: <http://www.taskjuggler.org/example.php>
- [3] Script de Perl *tjx2gant*: Instalado por defecto con TaskJuggler