

Se o seu problema é gerenciar projetos de maneira profissional no Linux, então o Taskjuggler é a alternativa correta. Com essa flexível ferramenta, até mesmo os mais ambiciosos desejos estão à mão. Nosso artigo mostra qual metodologia de gerenciamento o leva mais rapidamente ao seu objetivo.

POR UWE IRMER

Gerenciamento de projetos com o Taskjuggler

Malabarista de Projetos

Em projetos complexos, aqueles dentre nós que não possuem memória de elefante podem perder facilmente a visibilidade. Anotar e distribuir tarefas pode ser um primeiro passo, mas adotar um aplicativo adequado para gerenciamento de projetos pode ajudar muito e descomplicar o trabalho. Com um exemplo simples, vamos mostrar neste artigo como usar a poderosa ferramenta para gerenciamento de projetos *Taskjuggler* [1], um programa desenvolvido por Chris Schläger e Klaas Freitag, que é livre e está sob a Licença Pública Geral do Projeto GNU (GNU-GPL).

Imagine que chegou a hora de finalmente iniciar aquela – por várias vezes adiada – reforma do banheiro. Para que a obra não termine resultando em um caos total, ela exige algum planejamento, certo? Como mestre de obras desse projeto – um acontecimento com um início e um fim definidos

– deve lhe interessar saber quando quais tarefas vão começar, quando determinados serviços deverão ser contratados e, óbvio, quando será a primeira vez que você poderá entrar naquela banheira de hidromassagem de dois lugares com aquela companhia especial que você levou tanto tempo para conquistar (o duplo sentido da frase é proposital).

O plano

Após anos a fio de utilização no gerenciamento de projetos, o *Diagrama de Gantt* – assim batizado em honra ao seu criador, o engenheiro mecânico norte-americano Henry Laurence Gantt (1861 – 1919) – se tornou um padrão nesses casos. Nele são representadas graficamente todas as tarefas de um projeto em forma de barras horizontais, indicando o tempo que elas vão levar, definido com uma data inicial e uma data final. Dependências entre tarefas também são ilustradas nesse tipo de diagrama. O chamado “fluxo crítico” de tarefas, ou seja, aquele usado para determinar se as fases de um projeto serão executadas a tempo ou se haverá atrasos, é realçado graficamente.

O diagrama de Gantt fornece uma visualização gráfica do escopo geral de

tarefas a serem executadas, permitindo ao gerente, a qualquer momento, verificar a situação atual do projeto e reconhecer eventuais desvios em comparação com o planejado. Essa visualização facilita o reconhecimento das dependências entre as etapas do trabalho, tornando mais fácil a paralelização de processos, i.e., iniciá-los simultaneamente. Da mesma forma, também é possível planejar determinada fase do projeto de modo que ela termine ao mesmo tempo que outra. Essa paralelização tem a vantagem de encurtar a duração do projeto planejado, uma vez que várias de suas fases podem ser executadas ao mesmo tempo. Além disso, ela permite também um melhor aproveitamento de mão-de-obra e material – no jargão do gerenciamento de projetos, são conhecidos como “recursos”. No caso do nosso projeto de reforma do banheiro, recursos seriam os prestadores de serviço especializado (encanadores, pedreiros, pintores etc.), ferramentas, máquinas e, talvez, o espaço físico necessário. Se for possível separar os recursos para um projeto em cada uma de suas fases, torna-se fácil reconhecer, por exemplo, quando será necessário contratar mão-de-obra especializada. Passos intermediários

Quadro 1: Tarefas principais

Reforma do banheiro:

- ➔ 1. Banheiro do quarto de hóspedes em ordem
- ➔ 2. Demolição do banheiro antigo
- ➔ 3. Construção do banheiro novo
- ➔ 4. Acabamento do banheiro novo

Quadro 2: Detalhamento das tarefas

Deixar banheiro do quarto de hóspedes em ordem:

- 1. Testar instalações hidráulicas
- 2. Testar aquecimento de água
- 3. Efetuar os consertos necessários
- 4. Faxina no banheiro

Demolição do banheiro antigo:

- 1. Desligar água, aquecimento e parte elétrica
- 2. Desmontar os móveis do banheiro
- 3. Retirar a banheira velha
- 4. Retirar o chuveiro velho
- 5. Retirar a louça velha (vaso, pia e bidê)
- 6. Retirada dos azulejos e argamassa

[...]

importantes no escopo do projeto até o seu final são assinalados no Diagrama de Gantt como “marcos” (*milestones*), que são as datas em que certas tarefas **precisam** ter sido encerradas. Nessa seqüência, é importante verificar quais são as fases necessárias para o trabalho de reforma de um banheiro. Vamos fazer um quadro sinótico dessas atividades no **quadro 1**. A seguir, vamos detalhar as tarefas indicadas isoladamente, dividindo-as em subtarefas no **quadro 2**.

A divisão de tarefas

Optamos por ainda não fixar nesse início de projeto a seqüência que as subtarefas deverão seguir isoladamente. Por enquanto elas só estão arroladas abaixo das etapas principais do projeto, o que pode ser feito com maior ou menor riqueza de detalhes. Um esquema como o indicado nos quadros mencionados acima deveria ser, via de regra, suficiente para começar.

A próxima fase do nosso planejamento consiste em considerar quem vai ser responsável por que etapa do projeto. No nosso exemplo, está previsto que o próprio mestre de obras (que também é o gerente do projeto) será o responsável pela limpeza do local. A entrega do material de construção será feita pelo serviço de entregas da loja especializada. Há dois

prestadores de serviço à disposição: o Sr. Amassa Okano para serviços hidráulicos e de alvenaria e o Sr. Tomás Edilson para as instalações elétricas. O serviço de recolhimento de entulhos da prefeitura ficará a cargo do recolhimento do lixo produzido. A atribuição das tarefas para cada pessoa, bem como a disposição de todas as fases do projeto na linha do tempo, deve ser feita, finalmente, no aplicativo gráfico adequado.

O aplicativo

O Taskjuggler é, basicamente, uma coleção de bibliotecas e aplicativos de linha de comando. Felizmente, há também uma interface gráfica, baseada nas bibliotecas de desenvolvimento do KDE. O programa planeja sozinho as tarefas independentes e resolve conflitos quando, por exemplo, há um desencontro entre as datas de fim de uma tarefa e o início de uma outra, dependente da primeira. O Taskjuggler é flexível quando se trata de fixar as horas de trabalho padrão, bem como as folgas e o tempo livre, que são todos representados no Diagrama de Gantt. Os recursos para o projeto, para fins administrativos, podem ser classificados em grupos – um recurso (com perdão do trocadilho) do programa que não iremos utilizar neste artigo.

O Taskjuggler permite alocar custos específicos para cada um dos recursos, bem como obter os custos iniciais e finais do projeto. Com isso pode-se manter o orçamento para a reforma do banheiro sob controle e – caso o projeto se estenda por mais tempo – determinar o impacto dos atrasos no custo total da reforma.

O aplicativo é capaz de fornecer relatórios detalhados, por exemplo, organizados por tarefa, por progresso realizado em cada uma das fases e por custos. O Diagrama de Gantt reflete o progresso realizado

e mostra a correlação entre recursos e tarefas específicas. O programa dá visibilidade total ao gerente do projeto sobre o planejamento do pessoal, fornecendo dados como disponibilidade, utilização de mão-de-obra e relação de custos. Sua agenda de recursos mostra ainda quais estão disponíveis, em que período de tempo e em que quantidade.

Resumindo: o Taskjuggler é realmente uma ferramenta excepcional, adequada para o uso profissional. Todos os componentes para gerenciamento de tempo, recursos e custos estão disponíveis e os dados podem ser inseridos no programa de maneira simples e de fácil gerenciamento. Para melhorar, o aplicativo é dotado de recursos para geração de relatórios que ilustram claramente a situação atual do projeto, discriminando parâmetros como tempo, custos e uso de recursos.

Nosso próximo passo é começar a inserir as tarefas elencadas parcialmente no **quadro 2** no sistema. Para isso, o Taskjuggler usa um editor específico (ver **figura 1**).

Para criar um novo projeto, em primeiro lugar devem ser inseridos os dados do projeto usando as seguintes opções: *ID* do projeto, com uma descrição deste, informando também o período em que ele deve ser realizado, data atual, formato da data e moeda a ser utilizada nos cálculos de custo. Esses dados, para o nosso projeto, poderiam ficar da seguinte maneira: ➔

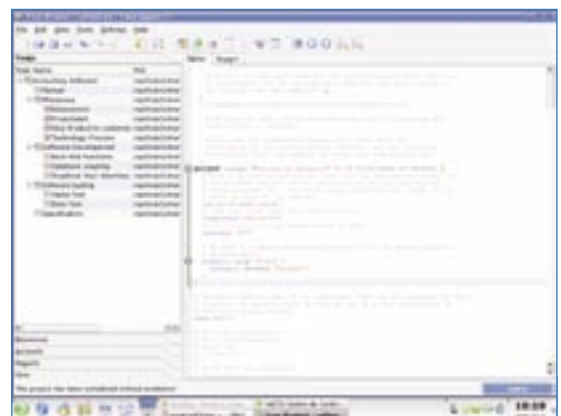


Figura 1: O editor do Taskjuggler é a ferramenta para entrada dos dados do projeto.

Quadro 3: Instalação

O código fonte do *Taskjuggler* pode ser encontrado na área de *download* da página web do projeto [1]. Para compilá-lo são necessárias as bibliotecas de desenvolvimento do KDE (*kdelibs-dev*, *kdelibs-devel* ou similar, dependendo da sua distribuição Linux). O pacote com o código fonte está comprimido com o *bzip2*, sendo necessário o uso da opção *-j* durante a descompactação com o comando *tar*:

```
tar -jxvf taskjuggler-2.1.tar.bz2
```

Entre no diretório criado pelo comando acima (*taskjuggler-2.1/*) e crie um *Makefile* adequado para a compilação do programa no seu sistema com o comando *./configure*. A compilação e a instalação do aplicativo serão realizadas simplesmente usando os comandos *make* e *su -c 'make install'*.

```
project refban "Reforma do Banheiro" "1.0" 01-07-2005 30-08-2005 {
now 11-07-2005
timeformat "%d-%m-%Y"
currency "R$"
scenario plan "Plan" {
scenario delayed "Delayed"
}
}
```

Os custos

Uma das grandes vantagens de se usar o *Taskjuggler* é a sua capacidade de contabilizar e calcular os custos do projeto. Assim, os gastos com a reforma do banheiro ficam o tempo todo sob controle e podem ser calculados a cada modificação do projeto inicial. Uma outra variável para o cálculo de custo do projeto é o chamado “fator de custo”, no nosso projeto com o valor de *rate 120.0*. Essa variável define o custo diário (ou de jornada) do prestador de serviços mais caro do projeto. A partilha dos fatores de custo é realizada de modo elegante através de macros, de modo que eles podem ser associados mais tarde no escopo de cada subtarefa àqueles que os geram. Com isso, poupa-se trabalho desnecessário durante a entrada dos dados do projeto, que são assim representados de uma maneira mais inteligível.

A macro para o nosso projeto exemplo fica, então:

```
macro alocar_mao-de-obra [
allocate mo1
allocate mo2 { load 0.5 }
allocate mo3
]
```

A mão-de-obra – *mo1* a *mo3* – é alocada de uma vez dentro da macro. O parâmetro *load 0.5* reduz o custo diário para a mão-de-obra 2, alocada pela entrada *mo2*, para a metade (ou seja, o

fator de custo é 0,5). Nas subtarefas essa macro é referenciada através do identificador `${alocar_mao-de-obra}`.

Nosso próximo passo é definir os recursos do projeto:

```
flags equipe
resource mo "mao-de-obra" {
resource mo1 "Okano"
resource mo2 "eu"
resource mo3 "Edilson"
flags equipe
}
```

Assim, a mão-de-obra necessária ao projeto está alocada adequadamente. As referências entre as pessoas envolvidas no projeto e às variáveis *mo1* a *mo3* também foram estabelecidas pelo bloco de texto acima. Detalhes adicionais sobre os prestadores de serviço podem ser inseridos na linha de cada um deles, respectivamente. Por exemplo, o Sr. Edilson vai estar de férias entre 1/8/2005 e 10/8/2005:

```
resource mo3 "Edilson" { vacation 2005-08-01 - 2005-08-10 }
```

Da próxima vez que o seu encanador fizer um orçamento constando um custo de jornada mais alto, você deve fornecê-lo individualmente, conforme segue:

```
resource mo1 "Okano" { rate 100.0 }
```

Inserindo dados

Após tudo isso, chegou a hora de inserir as tarefas no sistema:

```
task refban "Reforma do banheiro" {
task banhosp "Preparar banheiro do quarto de hóspedes"
task demolicao "Demolição do banheiro antigo"
task construcao "Construção do banheiro novo"
task fim "Encerramento"
}
```

Listagem 1: Subtarefas no editor do Taskjuggler

```
01 task refban "Reforma do banheiro" {
02 task banhosp "Preparar banheiro do quarto de hóspedes" {
03 task testar_hidraulica "Testar instalações hidráulicas"
04 task testar_eletrica "Testar aquecimento de água"
05 task reparos "Efetuar os consertos necessários"
06 task limpeza "Faxina no banheiro"
07 }
08 task demolicao "Demolição do banheiro antigo"
09 task construcao "Construção do banheiro novo"
10 task fim "Encerramento"
11 }
```

Listagem 2: Detalhamento adicional de tarefas

```

01 task banhosp "Preparar banheiro do quarto de hóspedes" {
02 task testar_hidraulica "Testar instalações hidráulicas" {
03 effort 1d
04 allocate mo2
05 }
06 task testar_eletrica "Testar aquecimento de água" {
07 effort 1d
08 allocate mo2
09 }
10 task reparos "Efetuar os consertos necessários" {
11 effort 2d
12 allocate mo1, mo3
13 }
14 task limpeza "Faxina no banheiro" {
15 effort 1d
16 allocate mo2
17 }
18 }

```

Na linguagem do editor do Taskjuggler uma tarefa é definida pelo parâmetro `task`. Cada tarefa consiste de um identificador, por exemplo `refban`, e de uma descrição, como no nosso caso *Reforma do banheiro*. Subtarefas ou detalhes de uma tarefa devem ser colocados entre chaves. De acordo com o nosso planejamento inicial, vamos alocar no Taskjuggler as subtarefas *Preparar banheiro do quarto de hóspedes*, *Demolição do banheiro antigo*, *Construção do banheiro novo* e *Encerramento*, sob a tarefa principal *Reforma do banheiro*. Essas subtarefas devem ainda ser subdivididas em subtarefas, como por exemplo no caso da subtarefa *Preparar banheiro do quarto de hóspedes*, que ilustramos na [listagem 1](#).

Proceda da mesma forma para todas as outras subtarefas.

Ainda faltam algumas informações para completarmos o nosso planejamento, como quanto tempo dura cada uma das tarefas, quem vai conduzi-las, como elas são executadas ao longo do tempo etc. Esses dados são importantes, afinal precisamos saber quando o banheiro novo vai estar pronto, quando cada um dos prestadores de serviço vai estar trabalhando e quanto tudo isso vai custar.

Os prazos

Vamos começar com a primeira questão: qual a duração de cada tarefa? Há um parâmetro no editor do Taskjuggler com o qual podemos definir isso, o chamado `effort`, calculado em “homem-dia” e que deve ser fixado para cada prestador de serviço (recurso de mão-de-obra) e tarefa. A opção `length` define a duração do projeto em dias úteis e a `duration` em dias corridos. Assim, precisamos anotar esses dados para cada recurso de mão-de-obra e estimar o esforço de cada um deles, para alimentar o Taskjuggler corretamente. Para a tarefa *Preparar banheiro do quarto de hóspedes*, esses valores poderiam ser mais ou menos os mostrados na [listagem 2](#).

O leitor ainda se lembra da relação entre identificador e recurso de mão-de-obra? Vamos recapitular: a variável `mo1` identifica o Sr. Okano, responsável pelos serviços de alvenaria e hidráulicos, `mo2` seria o próprio gerente do projeto, que, inevitavelmente, também vai ter que pôr a mão na massa, e `mo3` é o eletricista, o Sr. Edilson. A tarefa *Testar instalações hidráulicas*, que será executada pelo responsável pelo projeto, precisa de um dia para ser finalizada. O mesmo vale para a tarefa *Testar aquecimento de água*. Os reparos, executados pelo encanador e o eletricista, levam ao todo dois dias. A última tarefa nessa fase inicial – *Faxina no banheiro* – será feita novamente por nós e precisa de um dia para ser consumada. Da mesma forma devemos proceder com a entrada de dados de todas as tarefas restantes.

A palavra-chave `depends` fixa a seqüência das tarefas e serve para indicar as dependências entre elas. Como o Taskjuggler precisa calcular “fluxo de trabalho” (*workflow*), é importante trabalhar com informações relativas, quando se trata de tempo, do tipo “a tarefa 2 pode começar quando a 1 estiver terminada”. Tomando novamente como o exemplo a tarefa *Preparar banheiro do quarto de hóspedes*, fica claro o que queremos dizer: a subtarefa *Testar instalações hidráulicas* indica o início do projeto. Na seqüência, a subtarefa *Testar aquecimento de água* pode ser iniciada. Os reparos poderão começar tão logo essas duas tarefas anteriores tiverem sido realizadas e a *Faxina no banheiro* só será iniciada quando todos os consertos estiverem terminados. A [listagem 3](#) mostra como traduzir tudo isso na linguagem editor do Taskjuggler. ➡

Listagem 3: Definindo dependências

```

01 task banhosp "Preparar banheiro do quarto de hóspedes" {
02 task testar_hidraulica "Testar instalações hidráulicas" {
03 effort 1d
04 allocate mo2
05 depends refban.start
06 }
07 task testar_eletrica "Testar aquecimento de água" {
08 effort 1d
09 allocate mo2
10 depends !testar_hidraulica
11 }
12 task reparos "Efetuar os consertos necessários" {
13 effort 2d
14 allocate mo1, mo3
15 depends !testar_hidraulica, !testar_eletrica
16 }
17 task limpeza "Faxina no banheiro" {
18 effort 1d
19 allocate mo2
20 depends !reparos
21 }
22 }

```

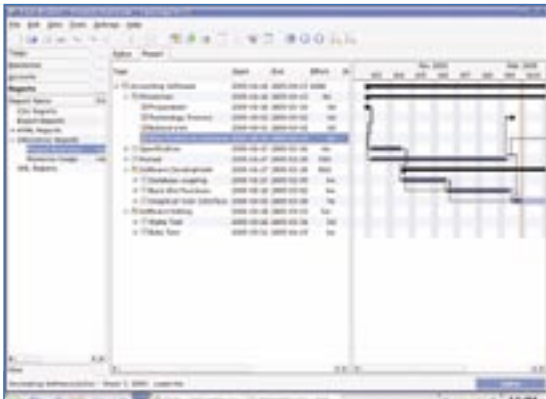


Figura 2: O Diagrama de Gantt do projeto exemplo mostra a seqüência de execução das tarefas.

O início do projeto

`refban.start` marca o início do projeto. O parâmetro `.start` foi criado automaticamente pelo Taskjuggler. As opções restantes descrevem momentos pontuais relativos na linha de execução do projeto, também calculados pelo aplicativo. Por exemplo, a variável `!testar_hidraulica` indica a data final para realização da tarefa *Testar instalações hidráulicas*. De acordo com a nomenclatura do editor do Taskjuggler, o sinal de exclamação indica uma data relativa dentro de uma tarefa em uma posição superior – no caso, *Preparar banheiro do quarto de hóspedes*. Dois sinais de exclamação indicariam uma data relativa dentro do projeto todo, ou seja, uma das tarefas principais. Seguindo essas definições, devemos entrar com os dados de dependências das tarefas e subtarefas.

A entrada de dados pode ser encerrada teclando-se `[F9]`. O programa vai testar os dados inseridos quanto à sintaxe e calcular o projeto. Pode-se ver então na representação gráfica mostrada pelo Diagrama de Gantt como as tarefas se relacionam, quando começam e terminam e quando o projeto vai ser concluído (**figura 2**).

Os recursos de relatório do Taskjuggler calculam os custos do projeto – lembre-se que alocamos recursos de

mão-de-obra para cada tarefa e que a esses recursos estavam alocados fatores de custo. Com base nessas informações, o Taskjuggler emite então o relatório apropriado ou, conforme mostra a **figura 3**, o uso de recursos de mão-de-obra no escopo do projeto.

O problema

O melhor dos planejamentos não serve para nada se não levar em conta alterações de acordo com as situações que aparecem no decorrer do projeto. Um exemplo de alteração muito comum em projetos são gargalos imprevisíveis no uso de recursos de mão-de-obra. Por exemplo, pode ocorrer que o electricista fique doente e não apareça para trabalhar. Ou, ao contrário, pode ocorrer que uma determinada tarefa seja executada em um tempo bem menor do que o planejado – no nosso exemplo, pode ser que não haja nada para consertar no banheiro do quarto de hóspedes.

Para atualizar o planejamento de acordo com as situações da vida real, usa-se novamente o editor do Taskjuggler. No caso da doença do electricista, modificamos as informações de tempo das tarefas em que ele está arrolado. Para tarefas que não precisam mais ser executadas, basta eliminar a entrada correspondente do planejamento. Para cada alteração, vamos simplesmente retificando os valores para as variáveis que foram modificadas (recursos de mão-de-obra, tempo e custo) e deixamos que o Taskjuggler se vire para calcular tudo de novo. Como trabalhamos com informações de tempo relativas no decorrer do projeto, o aplicativo vai calcular um novo

planejamento para ele. Somente no caso de tarefas que foram eliminadas é necessário fazer as alterações correspondentes no editor de maneira manual, caso haja referências às tarefas eliminadas nas tarefas ainda existentes.

Conclusão

O editor do Taskjuggler, cuja sintaxe é muito fácil de aprender, possibilita uma descrição detalhada de todas as partes de um projeto. O aplicativo se ocupa de colocar os dados inseridos através do editor em um Diagrama de Gantt. Além disso, é capaz de gerar uma série de relatórios sobre custos e planejamento de mão-de-obra. Nosso pequeno exemplo de reforma de banheiro contém praticamente todos os elementos básicos de um projeto e demonstra os recursos disponíveis no Taskjuggler para projetos de todas as dimensões.

Se o fato de ter que trabalhar com o editor do Taskjuggler não lhe agrada, vale a pena dar uma olhada no aplicativo *Planner* [2], que procura se espelhar no modo de funcionamento do Microsoft Project®, mas não dispõe de todos os recursos do Taskjuggler. ■

INFORMAÇÕES

[1] Taskjuggler: www.taskjuggler.org

[2] Planner: www.planner.org

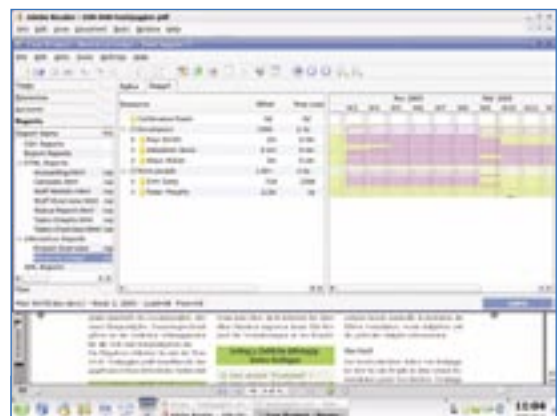


Figura 3: Planejamento de recursos de mão-de-obra no nosso projeto-exemplo.