



Tutorial de Preparação de um Instalador do CSGrid

Tecgraf/PUC-Rio
csgrid@tecgraf.puc-rio.br

Resumo

Esse documento é destinado a equipe de desenvolvimento e explica como gerar um instalador do CSGrid para ser distribuído. Assume-se que, para gerar o instalador, o desenvolvedor está trabalhando em uma máquina Linux do ambiente Tecgraf.

Sumário

1	Diretório de trabalho	4
1.1	Criar o diretório	4
1.2	Fazer check-out do Subversion	5
1.3	Fazer o build	5
2	Criação de um brach	7
2.1	Ir para o diretório do cmake	7
2.2	Executar o comando de criação de branch	7
3	Criação de um tag	8
3.1	Ir para o diretório do cmake	8
3.2	Executar o comando de criação de tag	8
4	Instalador do CSGrid	10
4.1	Identificar plataformas de instalação	10
4.2	Compilar os módulos dependentes de plataforma	10
4.3	Criar o diretório do Tomcat	11
4.4	Gerar uma chave digital para assinar o CSGrid	12
4.5	Fazer o deploy do cliente	14
4.6	Gerar o executável de instalação	14

5	Empacotamento para Distribuição	15
5.1	Incluir o arquivo bin	15
5.2	Criar o arquivo de validação	15
5.3	Criar o arquivo de configuração de SGAs	16
5.4	Criar o arquivo Readme	16
5.5	Criar o arquivo de distribuição	17
6	Criação de patch	18
6.1	Anotar todos os arquivos alterados	18
6.2	Fazer o checkout do branch do svn	19
6.3	Gerar uma tag	19
6.4	Fazer o check-out da tag do svn	20
6.5	Gerar o arquivo com os nomes dos arquivos do patch	20
6.6	Gerar o patch	21
6.7	Renomear o arquivo patch.changes para o nome do patch	22
6.8	Criação do arquivo de instruções	22

1 Diretório de trabalho

Essa etapa é básica para qualquer uma das tarefas envolvidas na preparação de uma instalação. Supõe-se que o desenvolvedor crie um diretório na sua área de trabalho para efetuar o check-out dos fontes do SVN, alterar o necessário e fazer commit das suas mudanças. Esse diretório de trabalho sempre reflete um determinado **ramo** do subversion.

Especificamente na preparação de uma instalação a ser feita, temos várias áreas de trabalho envolvidas, cada uma referente a uma etapa diferente do processo. Normalmente, são essas as áreas de trabalho que mencionamos no decorrer desse documento.

- **trunk_dir**: é o diretório que possui os fontes que serão usados para criar um branch (conforme Seção 2) de versão de instalação. Normalmente essa área possui os fontes provenientes do trunk.
- **branch_dir**: é o diretório que possui os fontes que serão usados para criar um tag (conforme Seção 3) de versão de instalação. Normalmente essa área possui os fontes provenientes do branch correspondente a versão.
- **tag_dir**: é o diretório que possui os fontes que serão usados para geração do instalador (conforme Seção 4.6). Normalmente essa área possui os fontes provenientes do tag correspondente a versão a ser instalada. Não se deve fazer alterações nessa área de trabalho, já que representa uma versão “congelada”. Correções devem ser feitas sempre no **branch_dir** (com posterior merge para o trunk) ou no **trunk_dir**.

Repare que os nomes **trunk_dir**, **branch_dir** e **tag_dir** são apenas exemplos. Cada desenvolvedor deve criar os diretórios com os nomes que melhor refletem o seu propósito.

Para criar um diretório de trabalho a partir do subversion, você deve seguir os passos seguintes.

1.1 Criar o diretório

Para fazer o check-out do CSGrid, o primeiro passo é definir um diretório de trabalho onde os módulos do CSGrid serão colocados e compilados. Usamos o nome **WORKING_DIR** como exemplo. Esse nome, obviamente, deve ser substituído pelo nome do diretório que o desenvolvedor está criando.

```
[sepetiba:~/] mkdir WORKING_DIR
```

onde **WORKING_DIR** é o nome do diretório.

1.2 Fazer check-out do Subversion

No diretório de trabalho que você criou na etapa anterior, faça check-out do ramo do SVN dos fontes e recursos do CSGrid. Repare que, apesar do CSGrid ser parte do CSBase (é um subdiretório do CSBase no SVN), o check-out é feito apenas do CSGrid. Por exemplo, para o caso de um check-out a partir do trunk, você deve fazer:

```
[sepetiba:~/] cd WORKING_DIR  
[sepetiba:~/WORKING_DIR] svn co https://subversion.  
tecgraf.puc-rio.br/engdist/cibase/trunk/csgrid csgrid
```

onde <https://subversion.tecgraf.puc-rio.br/engdist/cibase/trunk/csgrid> é a URL no SVN.

1.3 Fazer o build

Como o check-out do passo anterior foi feito apenas sobre o CSGrid, é necessário recuperar os outros módulos do CSBase para que o processo de build possa ser feito. O CSGrid usa o **csmake** para recuperar os módulos que o compõe. O arquivo `csmake.cfg` define quais módulos fazem parte do CSGrid, ou seja, aquilo que é recuperado do CSBase e o que é parte do próprio CSGrid.

Atualmente o CSGrid é composto dos seguintes módulos, de acordo com a árvore definida no `csmake.cfg`:

```
installTree {  
    { rep = repositories.csbase, module = "csbase" },  
    { rep = repositories.csbase, module = "csgrid", main = true },  
    { rep = repositories.csbase, module = "install" },  
    { rep = repositories.csbase, module = "init" },  
    { rep = repositories.csbase, module = "sgad" },
```

```
{ rep = repositories.csbase, module = "sgab" },  
  { rep = repositories.csbase, module = "validations" },  
  { rep = repositories.csbase, module = "validations_sys" },  
}
```

A execução do `csmake` é sempre feita a partir do diretório `csgrid`:

```
[sepetiba:~/WORKING_DIR] cd csgrid  
[sepetiba:~/WORKING_DIR/csgrid] csmake rebuild
```

Você pode confirmar todas as opções requisitadas, teclando ENTER. Esse passo pode demorar um pouco, em função do tamanho do CSBase.

2 Criação de um brach

Essa é a primeira etapa na preparação de uma versão do CSGrid para instalação. Ao criar um branch, você assegura que uma “cópia” do código fonte envolvido na versão que será gerada, isolando esse novo ramo no ciclo de desenvolvimento. Normalmente um branch é criado a partir do trunk.

Para criar um branch do CSGrid, você já deve estar com uma área de trabalho preparada conforme o descrito na Seção 1. Normalmente, essa área de trabalho contém o código proveniente do trunk.

Para criar um branch, você deve seguir os passos seguintes.

2.1 Ir para o diretório do cmake

```
[sepetiba:~/WORKING_DIR] cd csgrid
```

2.2 Executar o comando de criação de branch

```
[sepetiba:~/WORKING_DIR/csgrid] cmake branch
```

O script solicita o nome do branch. Esse nome é criado no SVN dentro do diretório **branches**. Você pode confirmar o nome sugerido, digitando ENTER, ou pode entrar com um novo nome. O trecho a seguir é um exemplo do que aparece na console como resultado da execução do comando.

```
Entre com o identificador do branch [CG-v1_05_01-2010_11_08]
```

```
[INFO ] aplicando branch CG-v1_05_01-2010_11_08 em >>> csbase
```

```
Committed revision 97937.
```

Após criado, testes e correções de bugs podem ser feitos nesse branch, dando continuidade assim aos preparativos para fechamento de versão. Lembre-se que, mudanças feitas no branch devem ser posteriormente levadas (**merge**) para o trunk.

3 Criação de um tag

Após criado um branch e estabilizada a versão para instalação, deve ser criada uma cópia “congelada” desse branch. Essa cópia, no Subversion, não deve sofrer modificações. Deve-se evitar ao máximo fazer *commits* em uma ramo de tag.

Essa é a primeira etapa na preparação de uma versão do CSGrid para instalação. Ao criar um branch, você assegura que uma “cópia” do código fonte envolvido na versão que será gerada, isolando esse novo ramo no ciclo de desenvolvimento. Normalmente um branch é criado a partir do trunk.

Para criar um tag do CSGrid, você já deve estar com uma área de trabalho preparada conforme o descrito em 1. Essa área de trabalho normalmente contém o código proveniente de um branch.

Para criar um tag, você deve seguir os passos seguintes.

3.1 Ir para o diretório do csmake

```
[sepetiba:~/WORKING_DIR] cd csgrid
```

3.2 Executar o comando de criação de tag

```
[sepetiba:~/WORKING_DIR/csgrid] cmake tag
```

O script solicita o nome do tag. Esse nome é criado no SVN dentro do diretório **tags**. Costumamos adotar o seguinte padrão de nome de tag: CG-v1_05_01-2010_11_08, onde:

- **CG**: abreviação de 2 letras para o CSGrid
- **v1_05_01**: código da versão no formato **major**, **minor** e **patch**.
- **2010_11_08**: data de criação do tag, no formato **ano**, **mês** e **dia**.

Você pode confirmar o nome sugerido, digitando ENTER, ou pode entrar com um novo nome. O trecho a seguir é um exemplo do que aparece na console como resultado da execução do comando.



Entre com o identificador da tag [CG-v1_05_01-2010_11_08]

[INFO] criando tag CG-v1_05_01-2010_11_08 em >>> csbase

Committed revision 97962.

A resources/version.properties

Checked out revision 97962.

Sending version.properties

Transmitting file data .

Committed revision 97963.

A resources/version.properties

Checked out revision 97963.

Sending version.properties

Transmitting file data .

Committed revision 97964.

Após criado, esse código copiado para o ramo da tag, será usado para gerar o instalador do CSGrid.

4 Instalador do CSGrid

Essa etapa se refere a geração do binário de instalação do CSGrid, compatível com a versão que está sendo criada. Normalmente, essa etapa é feita usando um diretório de trabalho construído a partir de um tag previamente criado. Consulte a Seção 3 sobre como criar um tag de uma versão e a Seção 1 sobre como preparar uma área de trabalho a partir de um tag do CSGrid.

Note que em um ambiente típico de trabalho, você precisa fazer o check-out do tag para gerar o instalador. Seu diretório de trabalho, nesse ponto, estará apontando para branch.

Para gerar o instalador do CSGrid você deve seguir os passos seguintes.

4.1 Identificar plataformas de instalação

Antes de iniciar a geração do instalador, é fundamental que você identifique em quais plataformas serão instalados o servidor CSGrid e os servidores de execução de algoritmos (SGA). A geração do binário de instalação do CSGrid depende da versão do sistema, da versão do compilador gcc instalado nas máquinas do ambiente e de sua arquitetura.

Uma forma mais rápida de obter essa informação é fornecendo o script da Figura 4.1¹ e o valor da variável de ambiente `TEC_UNAME` seja informado à equipe de desenvolvimento.

Na Seção 2 do Manual de Administração do CSGrid você encontra mais informações sobre os requisitos de hw/sw e configuração necessária no ambiente de execução do CSGrid.

4.2 Compilar os módulos dependentes de plataforma

Compilar módulos SGAs específicos de cada plataforma. Esses módulos devem ser compilados na plataforma correta. No Tecgraf, existem máquinas com os nomes das principais plataformas: `linux26g4`, `linux26g4_64` e outras. Você pode usar essas máquinas para compilar os módulos dos SGAs. Por exemplo, para compilar um SGA para `Linux26g4_64`, você deve fazer o seguinte:

- Fazer login na máquina com a plataforma (por exemplo, `linux26g4`).

¹Código desenvolvido para o shell `csh`.

```
setenv TEC_SYSNAME `uname -s`
setenv TEC_SYSVERSION `uname -r|cut -f1 -d.`
setenv TEC_SYSMINOR `uname -r|cut -f2 -d.`
setenv TEC_UNAME "${TEC_SYSNAME}${TEC_SYSVERSION}${TEC_SYSMINOR}"

set GCCVER=`/usr/bin/gcc -dumpversion|cut -f1 -d.`
if ($GCCVER == '4') then
    setenv TEC_UNAME      $TEC_UNAME'g4'
endif
unset GCCVER

setenv TEC_SYSARCH `uname -m`
if ($TEC_SYSARCH == 'x86_64') then
    setenv TEC_UNAME      $TEC_UNAME'_64'
endif
echo $TEC_UNAME
```

Figura 1: Definição da variável de ambiente TEC_UNAME.

```
[sepetiba:~/WORKING_DIR] ssh linux26g4_64
```

- Executar o comando de compilação do binário do SGA

```
[linux26g4-64:~/WORKING_DIR/csgrid] ./csmake build sgad
```

Para gerar o executável do SGA para Windows, ir uma uma máquina Windows no Tecgraf (tipicamente a Pero), ir para o diretório `sgad/src` que possui os fontes do SGA e rodar o script `build.bat`.

O diretório `sgad/bin` passa a ter um sub-diretório de cada plataforma para a qual um SGA foi compilado. Por exemplo:

```
[sepetiba:~/WORKING_DIR/sgad/bin] ls
Linux26g4/  Linux26g4_64/  sga-daemon@ sgad-cnfvavc.lua@  sgad-cnfv.lua@
```

4.3 Criar o diretório do Tomcat

Colocar o tomcat no diretório de trabalho. Normalmente basta expandir o arquivo mais atual que fica em `/softwares/linux/bin/tomcat`. Em seguida, fazer um link com o nome

tomcat para o diretório criado.

```
[sepetiba:~/WORKING_DIR/csgrid] cd ..  
[sepetiba:~/WORKING_DIR] tar -xvzf /softwares/linux/  
bin/tomcat/apache-tomcat-6.0.20.tar.gz  
[sepetiba:~/WORKING_DIR] ln -s apache-tomcat-6.0.20/ tomcat
```

4.4 Gerar uma chave digital para assinar o CSGrid

Antes de executar o **deploy** do CSgrid, é necessário criar uma chave digital para fazer a assinatura dos arquivos jar. Isso porque o CSGrid usa alguns recursos da máquina cliente (p.ex. impressão, acesso ao filesystem local) que exigem que os JARs sejam assinados digitalmente para serem “confiáveis”.

A rigor esta assinatura deveria ser obtida por um fornecedor oficial como VeriSign² ou Thawte³. Por enquanto, alguns sistemas serão assinados com um certificado "auto-assinado", i.e. gerado pelo próprio Tecgraf. Este certificado tecnicamente não oferece garantia de segurança ao usuário, e tem validade de apenas 6 meses, mas é a única alternativa no momento.

O arquivo gerado com a chave, no CSGrid, possui o nome **CSGKeystore** e deve estar no diretório **install**. Para se gerar o arquivo com a chave digital para o CSGrid deve-se executar o seguinte comando:

```
[sepetiba:~/WORKING_DIR] cd install  
[sepetiba:~/WORKING_DIR/install] keytool -genkey  
-keystore CSGKeystore -alias CSGrid
```

O resultado a seguir é exibido na console. O texto marcado com vermelho deve ser digitado por você.

```
Enter keystore password: tecgraf  
Re-enter new password: tecgraf  
What is your first and last name?
```

²<http://www.verisign.com>

³<http://www.thawte.com>



```
[Unknown]: CSGrid
What is the name of your organizational unit?
[Unknown]: Tecgraf
What is the name of your organization?
[Unknown]: PUC-Rio
What is the name of your City or Locality?
[Unknown]: Rio de Janeiro
What is the name of your State or Province?
[Unknown]: Rio de Janeiro
What is the two-letter country code for this unit?
[Unknown]: BR
Is CN=CSGrid, OU=Tecgraf, O=PUC-Rio, L=Rio de Janeiro,
ST=Rio de Janeiro, C=BR correct?
[no]: yes

Enter key password for <csgrid>
(RETURN if same as keystore password): <ENTER>
Re-enter new password:
```

Após a geração da chave, o certificado auto-assinado é gerado via comando:

```
[sepetiba:~/WORKING_DIR/install] keytool -selfcert
-alias CSGrid -keystore CSGKeystore
```

O resultado a seguir é exibido na console. O texto marcado com vermelho deve ser digitado por você.

```
Enter keystore password: tecgraf
```

Repare que a senha “tecgraf” deve ser a mesma usada no script **install** que executa o comando de assinatura do jar. No caso de querer usar outra senha, não se esqueça de alterar o argumento **storepass** do seguinte comando desse script:

```
jarsigner -storepass tecgraf -keystore CSGKeystore FILE csgrid
```

Também é importante lembrar que certificados auto-assinados têm validade de apenas 6 meses.

4.5 Fazer o deploy do cliente

Nessa etapa, os arquivo jar e outros *resources* (por exemplo, imagens, etc) necessários para execução do cliente, são copiados para a pasta de aplicações do Tomcat. Nesse momento, a chave digital criada na Seção 4.4 é usada para fazer a assinatura dos jars e, portanto, já deve ter sido criado.

O comando que faz o deploy é executado pelo **csmake** e, assim como os outros, deve ser executado no diretório **csgrid**.

```
[sepetiba:~/WORKING_DIR/csgrid] csmake webdeploy
```

4.6 Gerar o executável de instalação

Essa etapa se refere a geração do instalador, ou seja, do binário executável de instalação. Após executar o comando a seguir, um arquivo com extensão .bin será criado no diretório corrente.

```
[sepetiba:~/WORKING_DIR/csgrid] csmake pack
```

5 Empacotamento para Distribuição

Após ter feito a validação da instalação, deve ser criado um arquivo **tar** com o binário do instalador e outros arquivos necessários. No CSGrid, costumamos chamar o arquivo de distribuição de **csgridInstall.tgz** e ele é composto de:

- o binário do instalador gerado na Seção 4.6,
- o arquivo **Readme**, com informações sobre os procedimentos básicos de instalação,
- o arquivo **sgad-cnflua**, com um exemplo de configuração de SGAs para as plataformas disponibilizadas no binário do instalador
- o arquivo **Validations.properties** com o arquivo de propriedades usado pelo binário de instalação.

Para criar o arquivo **csgridInstall.tgz**, primeiro crie um diretório **install**. Nos exemplos que seguem, vamos supor que esse diretório foi criado dentro do diretório **csgrid**:

```
[sepetiba:~/WORKING_DIR/csgrid] mkdir install
```

5.1 Incluir o arquivo bin

Mova o arquivo binário do instalador (gerado conforme descrito na Seção 4.6) para o diretório **install**:

```
[sepetiba:~/WORKING_DIR/csgrid] mv CG-v1_05_01-2010_11_08.bin install
```

Lembrando que o nome **CG-v1_05_01-2010_11_08.bin** varia em função da versão a ser instalada.

5.2 Criar o arquivo de validação

Esse arquivo de propriedades é passado como parâmetro para o binário de instalação e usado para verificar condições no ambiente onde o CSGrid está sendo instalado como, por exemplo,

no caso de já haver uma outra instalação anterior cujos dados precisem ser migrados para a nova instalação.

O arquivo de validação é uma cópia do arquivo `validations_sys/System.properties.default`. Para criá-lo basta executar o seguinte comando de cópia:

```
[sepetiba:~/WORKING_DIR/csgrid] cp ../validations_sys/System.  
properties.default install/Validations.properties
```

5.3 Criar o arquivo de configuração de SGAs

Para facilitar, costumamos enviar no arquivo de distribuição, a configuração dos SGAs que serão instalados. Você pode usar como base o arquivo `sgad-cnf.lua` que está disponível em `sgad/src` e alterá-lo de acordo com as plataformas nas quais o SGA será instalado.

```
[sepetiba:~/WORKING_DIR/csgrid] cp ../sgad/src/sgad-cnf.lua install
```

Um exemplo de um nó para a execução do SGA em Linux26g4 é a seguinte:

```
Node{  
    name = "Linux26g4",  
    num_processors = 2,  
    platform_id = "Linux26g4",  
    byte_order = LITTLE_ENDIAN,  
}
```

5.4 Criar o arquivo Readme

O arquivo Readme deve conter orientações básicas que não façam parte do Manual de Administração e sejam específicas do ambiente onde o CSGrid será instalado.

Segue um exemplo de um exemplo de Readme que usamos como base para a distribuição de uma versão.



Versão do CSGrid: CG-v1_05_01-2010_11_08

Conteúdo do arquivo:

- . Readme
- . CG-v1_05_01-2010_11_08.bin: instalador
- . Validations.properties: parâmetro a ser passado para o instalador
- . sgad-cnf.lua: exemplo de configuração de SGAs para o ambiente a ser instalado

Esse instalador foi preparado para fazer a instalação do CSGrid e o SGA em uma máquina Linux26g4.

O manual de instalação do CSGrid é o ManualAdminCSGrid.pdf. Nesse manual vocês encontram as orientações com relação aos requisitos do ambiente, instalação e execução do servidor CSGrid.

5.5 Criar o arquivo de distribuição

Crie um arquivo compactado contendo todos os arquivos que fazem parte da distribuição do CSGrid.

```
[sepetiba:~/WORKING_DIR/csgrid] tar -czf csgridInstall.tgz install
```

6 Criação de patch

Um patch no CSGrid é um arquivo tar.gz que possui um conjunto de arquivos que devem ser adicionados ou substituídos em uma instalação do CSGrid. Para criar um patch, é necessário primeiro gerar um tag no svn com as mudanças referentes a esse patch e depois gerar os arquivos necessários para aplicar o patch na instalação.

Note que nem sempre é indicado gerar um patch. Se as mudanças forem muitas, o melhor é gerar um novo instalador para uma nova versão. O patch é mais indicado quando as mudanças são poucas e pontuais.

Para criar um patch, você deve seguir os passos seguintes.

6.1 Anotar todos os arquivos alterados

Você precisará da relação dos arquivos alterados mais a frente, para indicar ao comando patch do csmake quais arquivos farão parte do patch. Para descobrir os arquivos alterados, você pode usar o subversion. Basta fazer um `svn diff` entre o branch em que as alterações foram feitas e o último tag do svn. Por exemplo, suponha que o último tag é 1.5.2. Fazendo um `svn diff` entre o branch da versão 1.5 e o tag 1.5.2, o subversion exibirá os arquivos que estão alterados no branch e que, portanto, devem fazer parte do novo patch 1.5.3. A opção `summarize` do comando `svn diff` exibe apenas os nomes dos arquivos modificados.

```
[ubu:~] svn diff --summarize http://.../tags/CG_v1_05_01_2010_11_11  
http://.../branches/CG_v1_05_2010_10_19
```

```
M https://.../tags/.../csfs/src/csfs/impl/RemoteFileDefaultServantImpl.java  
M https://.../tags/.../csfs/src/csfs/impl/util/NativeFileSystem.java  
M https://.../tags/.../csbase/libs/csfs/csfs.jar  
M https://.../tags/.../sgad/csfs/libs/csfs/csfs.jar
```

A relação acima indica que os seguintes arquivos foram alterados:

- `csfs/src/csfs/impl/RemoteFileDefaultServantImpl.java`
- `csfs/src/csfs/impl/util/NativeFileSystem.java`

- `csbase/libs/csfs/csfs.jar`
- `sgad/csfs/libs/csfs/csfs.jar`

Note que, no exemplo acima, os arquivos `.class` referentes a `RemoteFileDefaultServantImpl` e `NativeFileSystem` não precisam ser enviados no patch pois já estão empacotados no `csfs.jar`.

6.2 Fazer o checkout do branch do svn

Como um patch é normalmente feito a partir de um branch, essa etapa se refere a trazer para a área de trabalho esse branch do svn. Sobre ele, as modificações e os testes podem ser feitos, antes de gerar uma nova tag e o patch com as alterações a serem aplicadas.

Para fazer check-out de um branch do svn, basta seguir as instruções desse manual, descritas na seção 1.

```
[ubu:~/csgrid/WORKING_DIR] svn co https://.../branches/.../csgrid
csgrid
[ubu:~/csgrid/WORKING_DIR] cd csgrid/
[ubu:~/csgrid/WORKING_DIR/csgrid] csmake rebuild
```

6.3 Gerar uma tag

Essa etapa se refere a criar, no svn, a tag correspondente as modificações feitas no branch e que serão parte do patch.

```
[ubu:~/csgrid/WORKING_DIR/csgrid] csmake tag
Entre com o identificador da tag [CG_v1_05_03_2011_05_04]
```

```
[INFO ] criando tag CG_v1_05_03_2011_05_04 em >>> csbase
```

```
Committed revision 117715.
```

```
A      resources/version.properties
Checked out revision 117715.
```

```
Sending          version.properties
Transmitting file data .
Committed revision 117716.
A      resources/version.properties
Checked out revision 117716.
Sending          version.properties
Transmitting file data .
Committed revision 117717.
```

6.4 Fazer o check-out da tag do svn

Essa etapa se refere a trazer para o diretório local a tag recém-criada e que possui as modificações que serão aplicadas no patch. Isso é necessário para eventuais testes e para gerar o patch. Você pode usar o mesmo diretório de trabalho, depois de assegurar-se que ele está vazio.

```
[ubu:~/csgrid/WORKING_DIR] cd ..
[ubu:~/csgrid] rm -rf WORKING_DIR
[ubu:~/csgrid] mkdir WORKING_DIR
[ubu:~/csgrid] cd WORKING_DIR
[ubu:~/csgrid/WORKING_DIR] svn co https://.../tags/.../csgrid
csgrid
[ubu:~/csgrid/WORKING_DIR] cd csgrid/
[ubu:~/csgrid/WORKING_DIR/csgrid] csmake rebuild
```

6.5 Gerar o arquivo com os nomes dos arquivos do patch

Para gerar o patch, é necessário relacionar todos os arquivos que serão incluídos no patch. Note que essa relação de arquivos pode ser diferente da relação que você identificou na primeira etapa dessa seção. Isso porque:

- Não devemos incluir arquivos .java no patch, apenas .class. Assim, para cada arquivo java, entre com o .class correspondente.
- Incluir todos os .class das Inner Classes dos arquivos alterados. Você deve verificar se os arquivos java possuem inner classes e, caso afirmativo, relacionar os nomes dos arquivos .class dessas inner classes.

- Lembrar também de incluir os .jars de libs que devem ser substituídas.

Por exemplo, um arquivo **patch.changes** com a lista dos arquivos a serem incluídos no patch pode ser criado da seguinte forma:

```
[ubu:~/csgrid/WORKING_DIR] vi patch.changes
csbase/libs/csfs/csfs.jar
sgad/csfs/libs/csfs/csfs.jar
csbase/server/services/csfsservice/CSFSService$1.class
csbase/server/services/csfsservice/CSFSService.class
csgrid/properties/CSFSService.properties
sgad/src/sga-daemon.lua
sgad/src/sga-command.lua
sgad/src/sga-services.lua
```

6.6 Gerar o patch

O comando `csmake patch` cria um arquivo `tar.gz` com todos os arquivos que devem ser incluídos ou substituídos na instalação do CSGrid. O arquivo do patch deve ter o mesmo prefixo da versão e um incremento no número usado para patch. Por exemplo, versão = `IG_v1_05_00`, então o patch será `IG_v1_05_01`.

Esse comando pedirá que você informe cada um dos arquivos anotados no passo anterior e salvos em **patch.changes**.

```
[ubu:~/csgrid/WORKING_DIR/csgrid] csmake patch
Entre com o numero sequencial do patch [1] 3
[INFO ] gerando patch CG-v1_05_03-patch_03-2011_05_04.tar.gz
Confirma? (s/n) [n] s
```

```
[INFO ] Entre com os nomes dos arquivos, um por linha.
Digite apenas ENTER para encerrar a lista.
```

IMPORTANTE: os paths devem ser relativos ao diretorio-raiz do projeto
(i.e. `init/...` `src/...` `sgad/...` etc.)

```
csbase/libs/csfs/csfs.jar
sgad/csfs/libs/csfs/csfs.jar
```

```
csbase/server/services/csfsservice/CSFSService$1.class
csbase/server/services/csfsservice/CSFSService.class
csgrid/properties/CSFSService.properties
sgad/src/sga-daemon.lua
sgad/src/sga-command.lua
sgad/src/sga-services.lua
```

[INFO] verificando arquivos

Para verificar se os arquivos foram incluídos corretamente no tar.gz do patch.

```
[ubu:~/csgrid/WORKING_DIR/csgrid] cd ..
[ubu:~/csgrid/WORKING_DIR] tar -tvzf CG-v1_05_03-patch_03-2011_05_04.tar.gz
```

6.7 Renomear o arquivo patch.changes para o nome do patch

```
[ubu:~/csgrid/WORKING_DIR] mv patch.changes
CG-v1_05_03-patch_03-2011_05_04.changes
```

6.8 Criação do arquivo de instruções

Um arquivo txt deve ser criado com as instruções que o administrador do CSGrid deve seguir para fazer a aplicação de um patch em um ambiente onde o CSGrid está instalado.

Em geral, as seguintes orientações devem ser fornecidas nesse arquivo de instrução:

1. Interromper o servidor e o cliente.
2. Colocar os arquivos CG-v1_05_03-patch_03-2011_05_04.changes e CG-v1_05_03-patch_03-2011_05_04.tar.gz no diretório install da sua instalação.
3. Fazer um backup dos arquivos que serão alterados.

```
cd <instalação>/src
tar -czvf ../install/backupPatch1.5.3.tgz
`cat ../install/CG-v1_05_03-patch_03-2011_05_04.changes`
```



4. Depois disso, a partir do diretório <instalação>/src, você deve aplicar o patch fazendo um

```
tar -xzf ../install/CG-v1_05_03-patch_03-2011_05_04.tar.gz
```

5. Feito isso, pode reiniciar o servidor.