
GeneXus™
by Globant

DEPLOYMENT

Nicolas Adrién

GeneXus

Neste vídeo falaremos sobre a passagem entre ambientes e upgrades, problemas comuns com isto e aprenderemos a usar a ferramenta GAM Deploy Tool.

Em certas ocasiões, pode acontecer que um desenvolvedor não tenha acesso à base de dados e deseje realizar uma passagem de GAM entre ambientes.
Aqui podemos nos perguntar como fazer para realizar as reorganizações e impactos de metadados.

Para resolver este problema, GAM nos oferece diferentes scripts que já estão incluídos na pasta de instalação do GeneXus.
Dentro da pasta Platforms, você deve encontrar os ambientes que definimos em nossas KBs.
Lá, encontraremos os seguintes arquivos.

Com o primeiro arquivo poderemos criar toda a base de dados do GAM.
Os restantes consistem em migrar entre as diferentes versões do GAM, como indicam seus nomes. Algo importante com isto é que no caso de utilizar estas migrações, elas devem ser executadas em ordem para que tudo funcione corretamente.
Isso faz parte da estrutura da base de dados, onde nestes arquivos teremos todas as instruções SQL correspondentes que devem ser executadas em um cliente da base de dados do GAM.

GeneXus

GAM Deploy Tool

GAM Deploy Tool é uma ferramenta que tem como objetivo nos auxiliar nas implantações de aplicações utilizando GAM, pois possui o conhecimento da estrutura e relações da base de dados do GAM, proporcionando a possibilidade de importar dados de forma gradual mantendo a consistência do modelo de dados.
A ferramenta é voltada para levar à produção a informação que precisa ser atualizada na Base de Dados GAM e está disponível a partir de GeneXus 16 U5.

Vamos nos concentrar na versão por linha de comandos.

GeneXus

GAM Deploy Tool command line

JAVA: /tomcatproduction/webapps/XXX/WEB-INF/classes/com/kbname#

java -cp "../../../lib/*" genexus.security.api.agamdeploytool "<Action> <Corresponding Flags>"

.NET Framework:

agamdeploytool.exe "<Action> <Corresponding Flags>"

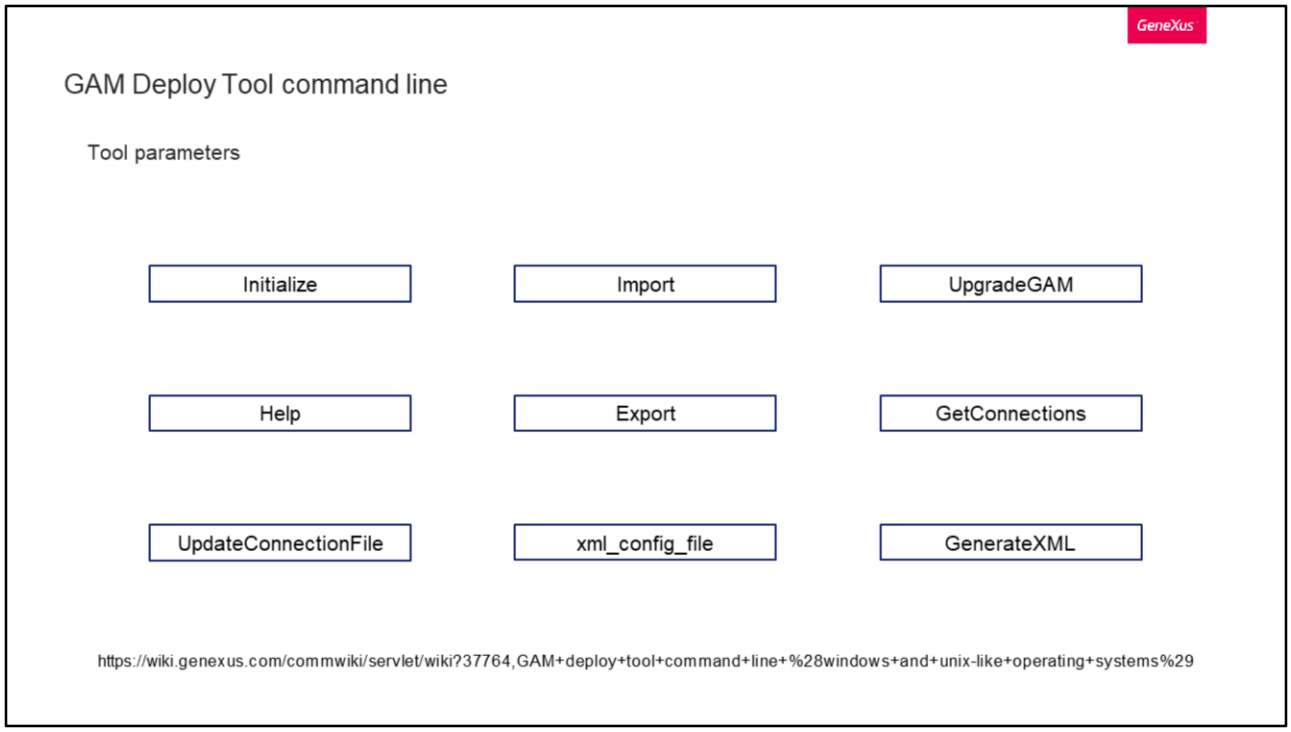
.NET:

dotnet agamdeploytool.dll "<Action> <Corresponding Flags>"

As chamadas à ferramenta devem ter o seguinte formato, de acordo com o ambiente em que se encontram.

O ideal é executar a ferramenta posicionados abaixo do ambiente de implementação. No caso de .NET, no diretório virtual \bin. No caso de Java, abaixo de seu webapp como vemos no comando.

A ferramenta não solicitará a configuração de conexão da base de dados GAM, como o servidor, porta, usuário ou senha, pois essas informações são obtidas dos arquivos de configuração como o client.cfg no caso de Java e web.config no caso de .NET.



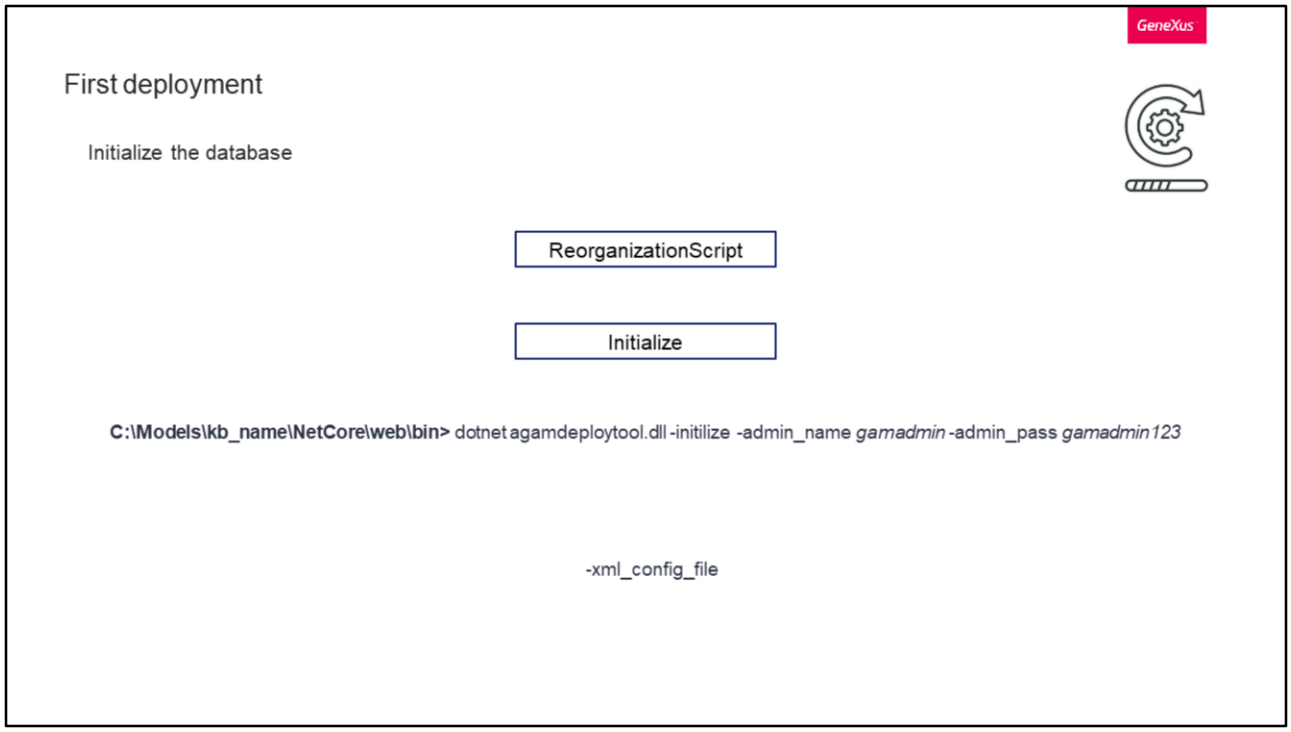
Esta ferramenta nos oferece diferentes ações que podemos fazer com ela. Vamos ver detalhadamente quais são.

- Inicializar: Nos permite inicializar a base de dados GAM com seus metadados (somente repositório 1)
- Importar: Importa um pacote com suas configurações e dados
- Atualizar GAM: Atualiza a versão da base de dados GAM. Caso seja necessário fazer uma reorganização, ela deve ser realizada previamente e depois executar esta ação da ferramenta
- Ajuda: Como todo comando, com isto veremos as ações disponíveis que a ferramenta possui
- Exportar: Exporta os dados de uma base de dados GAM e os armazena em um pacote .gpkg.
- Obter Conexões: Obtém as conexões agrupadas por Repositório. Esta função é útil para obtermos os GUID de nossos repositórios e os nomes de conexão que são enviados como parâmetros na opção UpdateConnectionFile
- Esta última atualiza ou cria o arquivo connection.gam com os dados de conexão que obtém
- xml_config_file: Esta é uma flag especial que apenas recebe um arquivo XML no qual são carregados todos os parâmetros a serem inseridos na ferramenta, incluindo a ação que queremos

Finalmente, temos a Gerar XML, que gera um XML de exemplo e o exibe na saída padrão. Este XML pode ser usado como entrada para a ferramenta, alterando os

valores de tag correspondentes.

Dentro de cada ação, é claro, temos variadas flags para incluir. O detalhe destas podem ser encontrados na Wiki de GeneXus com toda a documentação necessária.



Agora vamos ver como deveríamos fazer para implantar o GAM e seus dados pela primeira vez e para posteriores implantações.

Vamos começar com a primeira implantação.
Obviamente, utilizaremos a ferramenta GAM Deploy Tool.

No início do vídeo vimos que o GAM nos fornecia diferentes scripts na pasta de instalação do GeneXus. Como em nosso caso queremos inicializar a base de dados, vamos executar o ReorganizationScript. Com ele criaremos a base de dados do GAM vazia.
Por sua vez, também vimos as diferentes opções que nos oferece a ferramenta que estamos vendo, onde uma delas é Initialize. Com ela, inicializamos a base de dados do GAM da seguinte maneira.
Na execução devemos indicar o nome de usuário e senha do GAM.

Um parênteses a mencionar é que este e todos os outros exemplos são baseados em .NET, que é o que utilizarão no prático.

Por sua vez, temos a possibilidade de usar a seguinte flag, onde através dela podemos indicar o caminho para um arquivo XML que tenha todos os parâmetros configurados. Se definirmos esta flag, todos as outras são ignoradas automaticamente e apenas são levados em consideração os parâmetros do arquivo. Um exemplo de

arquivo para esta flag pode ser o seguinte.

GeneXus

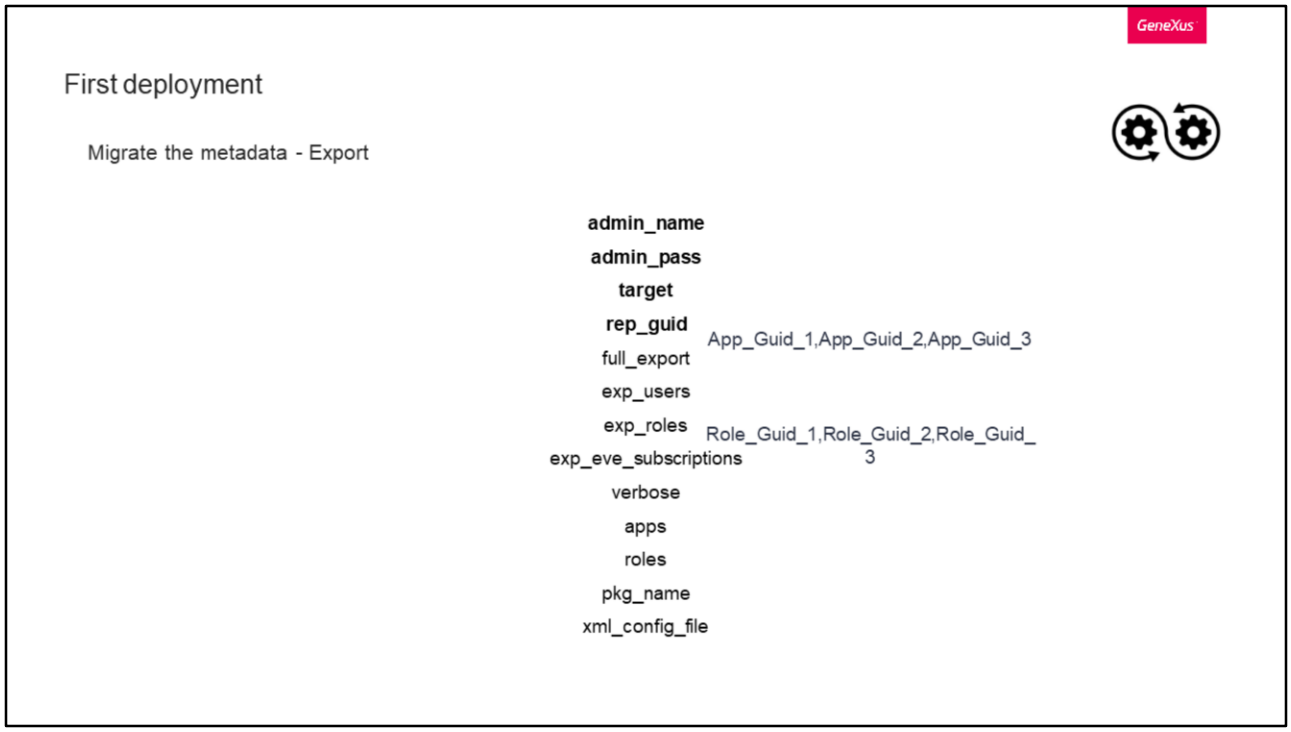
```
<?xml version="1.0" encoding="utf-8"?>
<GamDeployTool>
    <GeneralSettings>
        <ProcessType>Import</ProcessType>
        <GamAdminName>gamadmin</GamAdminName>
        <GamAdminPass>gamadmin123</GamAdminPass>
        <Verbose>true</Verbose>
    </GeneralSettings>

    <DbmsSettings>
        <DbmsCode>18</DbmsCode>
    </DbmsSettings>

    <ImportSettings>
        <PackageFilePath>C:\Temp\fullgx\GAM_package_GAMPackage403.gpkg</PackageFilePath>
        <ImportType>Full</ImportType>
        <GenerateDefaultConnection>false</GenerateDefaultConnection>
        <ImportRepositoryAction>update</ImportRepositoryAction>
        <RepositoryGuid>0d777d89-790e-4a19-84c5-e92f5e9e5920</RepositoryGuid>
    </ImportSettings>

</GamDeployTool>
```

DB2 for iSeries => 9
DB2 Universal Database => 5
Informix => 11
MySQL => 18
Oracle => 7
PostgreSQL => 15
SQL Server => 12

Na tag ProcessType temos os valores: Import, Export, GenerateConnectionFile, RestartGamDb e UpdateSchema.

Dentro das configurações de conexão para a base de dados, temos o código do DBMS. Ali, temos os seguintes valores conforme o sistema de base de dados que estamos utilizando.

Finalmente temos a configuração para executar um Import. Obviamente, isto só é levado em consideração se ProcessType tiver o valor Import.
Aqui dentro primeiro temos o caminho de onde carregar o pacote de deploy, depois o tipo de importação (se é Full ou Custom), um booleano para indicar se deve ser gerada uma conexão padrão para o repositório, uma flag que pode indicar se é desejado atualizar um repositório existente (como está agora com o valor update, ou se queremos criar um novo repositório que nesse caso deveríamos colocar o valor CreateNew) e, finalmente, temos um identificador de repositório que se aplicaria ao repositório que queremos atualizar se na flag anterior inserimos o valor update.



Uma vez executado o Initialize, teremos o repositório padrão criado. Mas agora temos que passar nossos próprios repositórios, junto com usuários, roles, etc.
Para fazer isto, precisamos primeiro exportar os dados e depois importá-los no ambiente de Produção ou Pré-Produção para o qual queremos migrar os dados.

Além do usuário e senha, em target especificaremos o diretório de destino da exportação e em rep_guid introduziremos o identificador do repositório a exportar.
Vejamos os seguintes campos:

- full_export: Booleano que no caso de ser verdadeiro, estamos indicando que a exportação seja completa, incluindo usuários, roles e aplicações.
- exp_users, roles e eve_subscriptions: indicam se queremos exportar usuários, roles e as subscrições a eventos, respectivamente. Esta flag só é levada em consideração se full_export tiver valor Falso.
- apps e roles: conterão a lista de aplicações e roles, respectivamente, que queremos que sejam exportadas. Estes devem ter o seguinte formato.
- pkg_name: é o nome do pacote que estamos exportando
- e xml_config_file que como já dissemos é um xml com toda esta configuração de parâmetros.

Algo a se destacar é que as flags em negrito são obrigatórias para fazer o export.

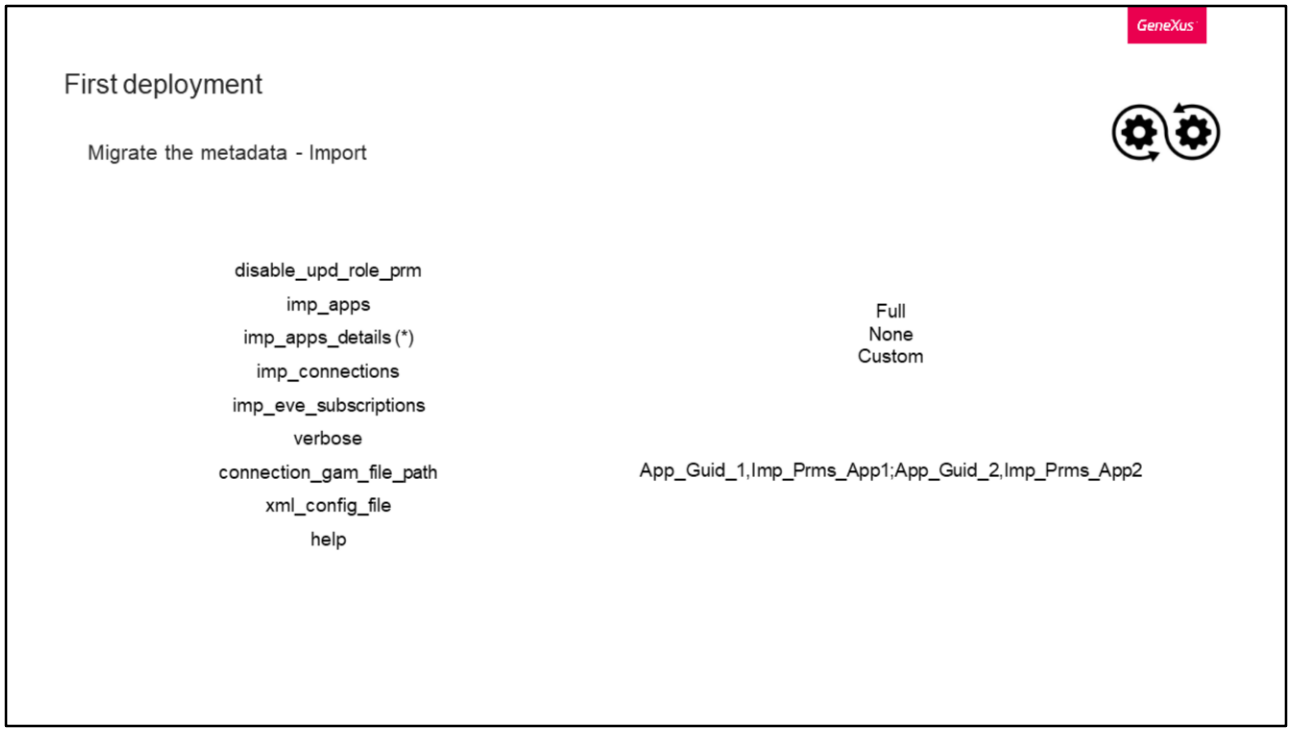
Depois de ter feito o Export, temos que fazer o Import. Vamos ver as flags disponíveis para isto.

Em file_path_package devemos inserir o caminho do nosso pacote exportado anteriormente.
Em seguida, temos os dados do administrador, que neste caso, além do nome e senha, podemos inserir o nome de usuário e o identificador da role do referido administrador.
upd_rep é um booleano para indicar se vamos fazer uma atualização do repositório existente, se sim, na seguinte flag indicamos seu identificador. Caso não, temos as seguintes flags:

- new_rep_create , que é um booleano que indica se será criado um novo repositório. Em seguida, temos o nome, namespace, identificador, nome e senha do novo administrador do repositório e nome de usuário e senha da conexão do repositório.

Depois temos todas aquelas que indicam o que queremos importar, que correspondem aos tipos de autenticação, políticas de segurança, usuários e roles. Todas booleanas. Caso queiramos importar tudo, podemos usar a flag seguinte que é imp_full. (Esta só é válida a partir de GeneXus 16 U6).

Assim como no painel anterior, as flags em negrito são as obrigatórias.
Aquelas com um asterisco também são, mas apenas quando a flag new_rep_create tiver o valor True.



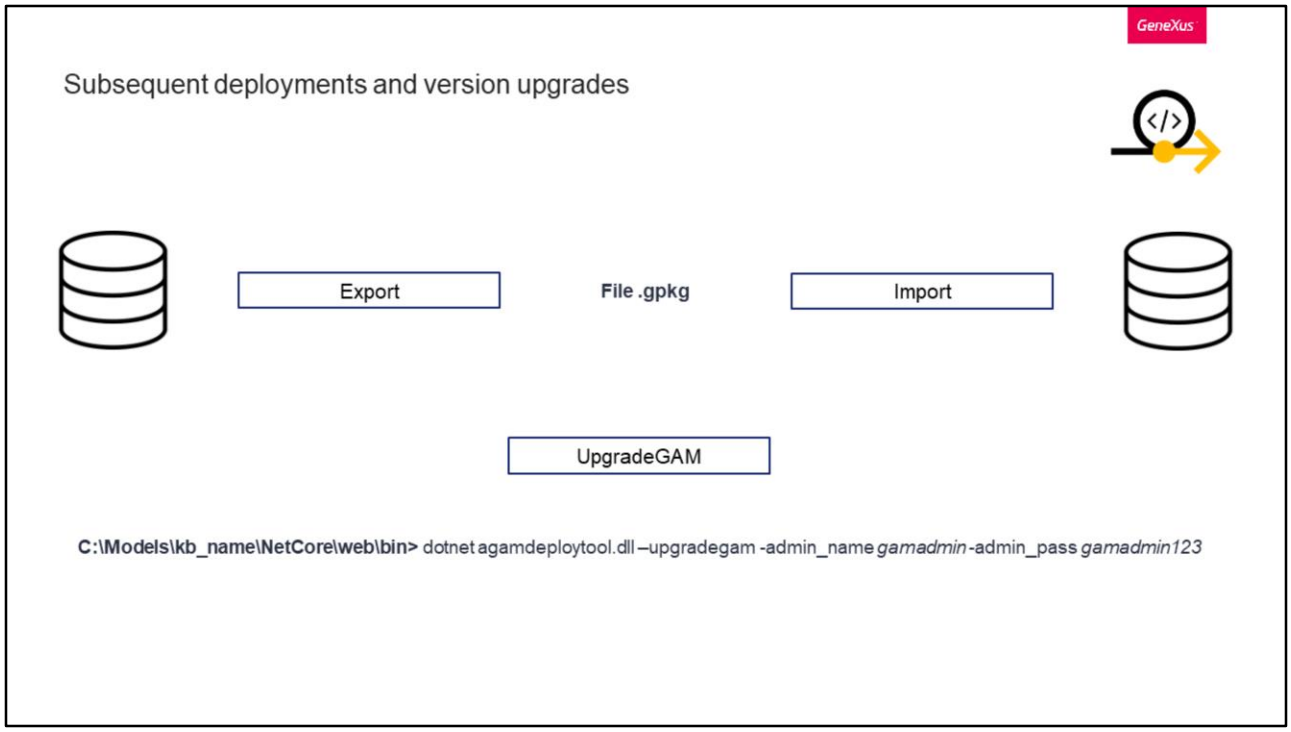
Continuando com as flags, com disable_upd_role_prm indicamos se devem ser importadas as permissões das roles que já existem na Base de Dados, por padrão está em False.
imp_apps é o nível com o qual são importadas as aplicações, e para isto temos os seguintes valores:

- Completo, onde todas as aplicações são importadas com todas as permissões
- Nenhum, que indica que nada é importado em relação às aplicações
- Ou custom, onde são configuradas de acordo com a flag imp_apps_details

Esta última flag representará então a lista de pares Identificador de aplicação e booleano indicando se são importadas as permissões dessa aplicação.
Então temos imp_connections que importa as conexões do pacote. Para os novos repositórios, sempre é criada uma nova conexão independentemente do valor desta flag; mas em caso de atualização, as conexões não são atualizadas, a menos que tenham o mesmo nome de usuário de conexão de uma conexão já existente.
Para fechar a parte de importação, temos a imp_eve_subscriptions, que vai importar as subscrições a eventos.

Do restante de flags, podemos destacar connection_gam_file_path, com a qual indicaremos o destino de onde será gerado o arquivo connection.gam.

A flag imp_apps_details tem um asterisco, pois é obrigatória somente quando a flag imp_apps tem o valor Custom.



Uma vez que já fizemos a primeira implantação, vamos ver como são feitas as próximas, ou como migrar de versões de GAM.

Caso apenas tenhamos que atualizar nosso próprio metadado, devemos realizar o mesmo processo anterior de primeiro Exportar, o qual trará da base de dados o que for necessário, gerando um arquivo .gpkg, e depois Importar, selecionando é claro o que queremos migrar através das diferentes opções que nos oferece a funcionalidade.

Agora, se apenas quisermos atualizar a versão do GAM, basta executar a ação de UpgradeGAM, onde ela atualizará o GAM juntamente com seus metadados. Sua execução é análoga à de inicializar, onde são solicitados o nome de usuário e senha do administrador.

GeneXus™
by Globant

training.genexus.com
wiki.genexus.com