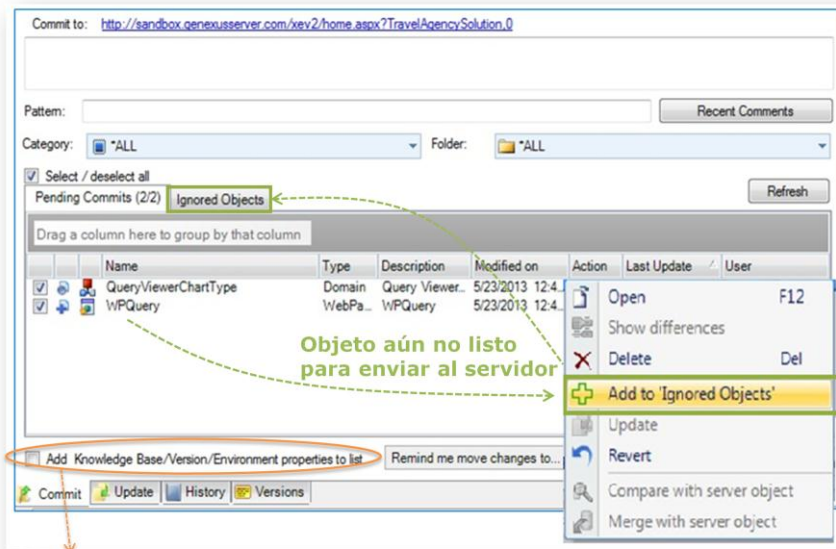


## MÁS SOBRE GXSERVER

GeneXus training  
[training.genexus.com](http://training.genexus.com)

## Commit: envío de modificaciones..



Enviar al server las propiedades que sufrieron cambios

### Ignored Objects

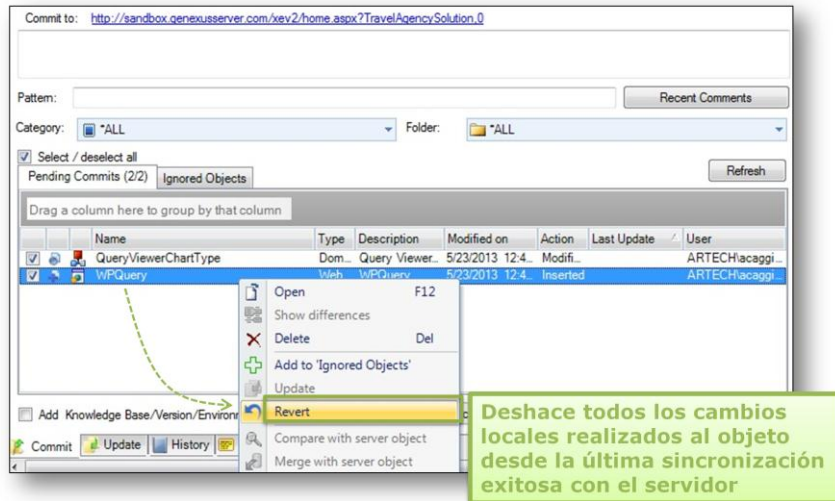
Muchas veces sucede que un desarrollador tiene en su KB local objetos de prueba o que aún no han sido finalizados y testeados. En estos casos no es deseable que dichos objetos se visualicen dentro de la lista de objetos a ser enviados al server (commit) y por lo tanto se pueden enviar a la lista de objetos a ser ignorados en las operaciones de commit.

Para enviar un objeto a la lista de objetos ignorados, alcanza con hacer clic con el botón derecho del mouse sobre el objeto (desde la solapa Ready for Commit), y seleccionar "Add to ignore on Commit list".

Una vez que el objeto está terminado y listo para ser enviado al server (commit), se debe eliminar de la lista de objetos ignorados. Para esto, alcanza con hacer clic con el botón derecho del mouse sobre el objeto (en la solapa Ignored Objects) y seleccionar "Remove from Ignore on Commit list".

# Commit: envío de modificaciones..

## Revert de objetos Ready for Commit:

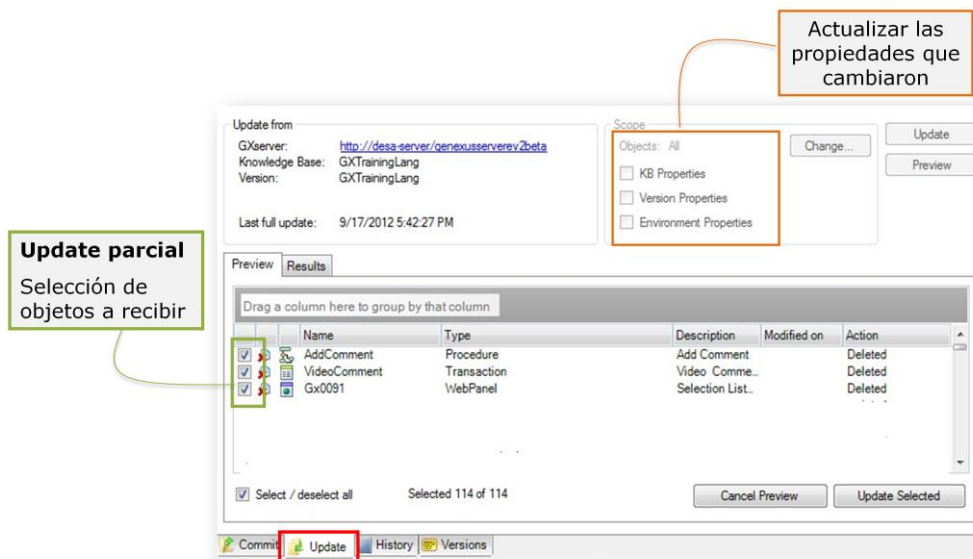


La operación **Revert**, permite deshacer todos los cambios realizados sobre un objeto desde la última sincronización exitosa con el server. En definitiva, es hacer que un objeto que aparece en la lista Ready for Commit, deje de estarlo.

Esta opción aparece disponible haciendo clic con el botón derecho del mouse sobre un objeto desde la lista Ready for Commit.

Se setea como activa (SetAsActive) la versión correspondiente, sin que se pierda nada de la historia del objeto.

## Update: recepción de modificaciones..



La operación de **Update** es de algún modo la contraparte de la operación de **Commit**. Aplica a la KB personal del desarrollador que la está ejecutando, todos los cambios realizados por otros desarrolladores en la KB del servidor, dándole feedback al desarrollador del resultado de dicha operación.

No es obligatorio recibir todas las modificaciones juntas ya que el desarrollador puede seleccionar los cambios a recibir (Update parcial).

## Integración de cambios...

**Merge automático:** Dos desarrolladores modifican la estructura de la misma trn.

**Peter**

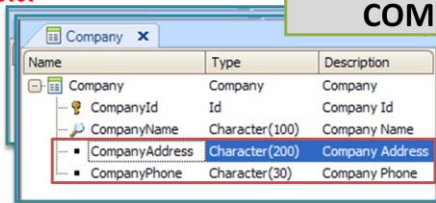


Diagram illustrating Peter's TRN structure. A table shows the structure of the 'Company' TRN. The table has columns: Name, Type, and Description. The structure includes: Company (Type: Company, Description: Company), CompanyId (Type: Id, Description: Company Id), CompanyName (Type: Character(100), Description: Company Name), CompanyAddress (Type: Character(200), Description: Company Address), and CompanyPhone (Type: Character(30), Description: Company Phone). A COMMIT arrow points from this structure to the server.

Name	Type	Description
Company	Company	Company
CompanyId	Id	Company Id
CompanyName	Character(100)	Company Name
CompanyAddress	Character(200)	Company Address
CompanyPhone	Character(30)	Company Phone

COMMIT

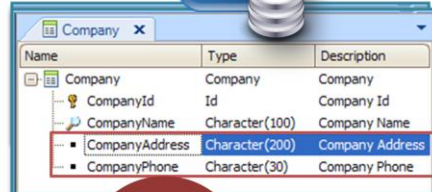


Diagram illustrating the server's TRN structure after Peter's commit. The table shows the structure of the 'Company' TRN. The table has columns: Name, Type, and Description. The structure includes: Company (Type: Company, Description: Company), CompanyId (Type: Id, Description: Company Id), CompanyName (Type: Character(100), Description: Company Name), CompanyAddress (Type: Character(200), Description: Company Address), and CompanyPhone (Type: Character(30), Description: Company Phone). A red circle with a diagonal line through it is placed over the table, indicating a warning or error.

Name	Type	Description
Company	Company	Company
CompanyId	Id	Company Id
CompanyName	Character(100)	Company Name
CompanyAddress	Character(200)	Company Address
CompanyPhone	Character(30)	Company Phone

**Mike**

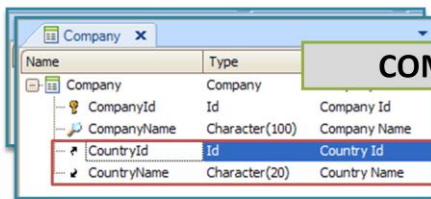


Diagram illustrating Mike's TRN structure. A table shows the structure of the 'Company' TRN. The table has columns: Name, Type, and Description. The structure includes: Company (Type: Company, Description: Company), CompanyId (Type: Id, Description: Company Id), CompanyName (Type: Character(100), Description: Company Name), CountryId (Type: Id, Description: Country Id), and CountryName (Type: Character(20), Description: Country Name). A COMMIT arrow points from this structure to the server.

Name	Type	Description
Company	Company	Company
CompanyId	Id	Company Id
CompanyName	Character(100)	Company Name
CountryId	Id	Country Id
CountryName	Character(20)	Country Name

COMMIT



Veamos cuál es el comportamiento cuando dos usuarios modifican el mismo objeto.

### Ejemplo:

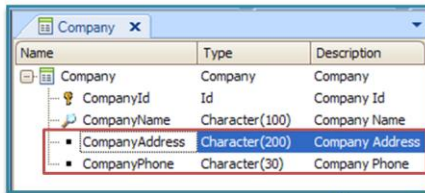
Ambos usuarios parten de la misma TRN Company:

- Peter le agrega 2 atributos: CompanyAddress y CompanyPhone.
- Mike le agrega CountryId y CountryName.
- Peter hace commit y se envían sus cambios al server.
- Mike hace commit y le aparece una advertencia porque no tiene la última versión de la TRN (ya que Peter la modificó), así que debe sincronizarse primero (ejecutar la operación Update para recibir las nuevas definiciones).

## Integración de cambios...

**Merge automático:** Dos desarrolladores modifican la estructura de la misma trn.

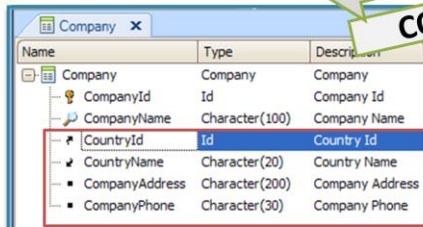
**Peter**



A screenshot of a TRN structure window for 'Company'. It shows a tree view with 'Company' as the root, containing 'CompanyId', 'CompanyName', 'CompanyAddress', and 'CompanyPhone'. The 'CompanyAddress' and 'CompanyPhone' nodes are highlighted with a red box.

Name	Type	Description
Company	Company	Company
CompanyId	Id	Company Id
CompanyName	Character(100)	Company Name
CompanyAddress	Character(200)	Company Address
CompanyPhone	Character(30)	Company Phone

**Mike**



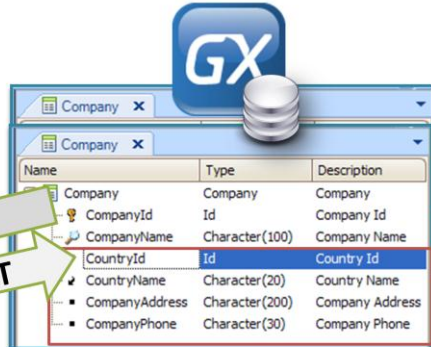
A screenshot of a TRN structure window for 'Company'. It shows a tree view with 'Company' as the root, containing 'CompanyId', 'CompanyName', 'CountryId', 'CountryName', 'CompanyAddress', and 'CompanyPhone'. The 'CountryId' and 'CountryName' nodes are highlighted with a red box.

Name	Type	Description
Company	Company	Company
CompanyId	Id	Company Id
CompanyName	Character(100)	Company Name
CountryId	Id	Country Id
CountryName	Character(20)	Country Name
CompanyAddress	Character(200)	Company Address
CompanyPhone	Character(30)	Company Phone

UPDATE

COMMIT

Merge



A screenshot of the merged TRN structure window for 'Company'. It shows a tree view with 'Company' as the root, containing 'CompanyId', 'CompanyName', 'CountryId', 'CountryName', 'CompanyAddress', and 'CompanyPhone'. The 'CountryId' and 'CountryName' nodes are highlighted with a red box. A 'GX' logo and a stack of coins are visible above the window.

Name	Type	Description
Company	Company	Company
CompanyId	Id	Company Id
CompanyName	Character(100)	Company Name
CountryId	Id	Country Id
CountryName	Character(20)	Country Name
CompanyAddress	Character(200)	Company Address
CompanyPhone	Character(30)	Company Phone

- Cuando Mike ejecuta el Update, automáticamente se realiza un Merge con los cambios que vienen del server y los cambios que él había hecho en la TRN. Por lo tanto obtiene una versión con todos los cambios integrados.

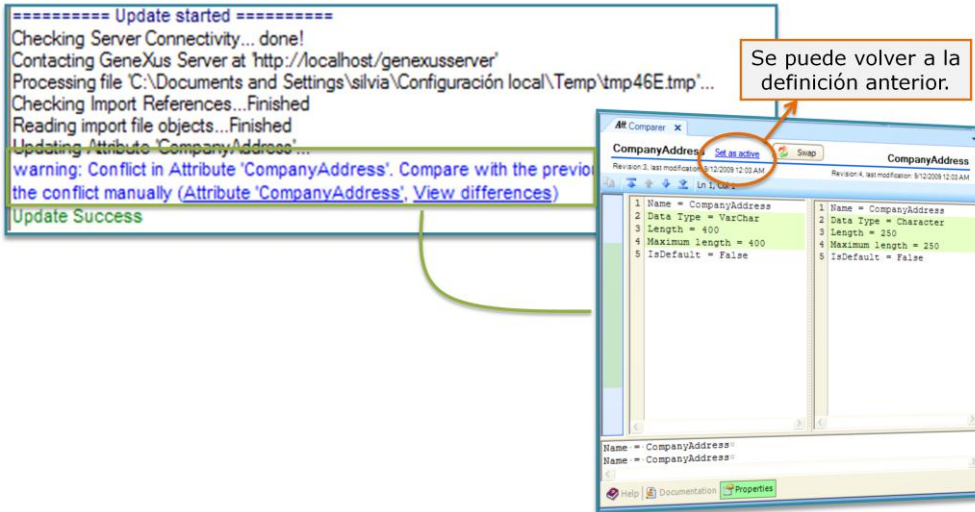
- Ahora sí puede enviarlos al server para integrar todos los cambios en la KB.

En este ejemplo cada usuario modificó la misma TRN, pero diferentes atributos.

¿Cuál sería el comportamiento si modificaron el mismo atributo?

# Integración de cambios...

**Conflicto:** Dos desarrolladores modificaron la definición del mismo atributo.



¿Cuál sería el comportamiento si modificaron el mismo atributo?

Al momento de realizar la operación de update, el desarrollador recibirá un mensaje indicando que dicho atributo fue modificado y será sobrescrito por el que se encuentra en el server.

Podrá ver con detalle cuál fue el cambio que se realizó en el atributo, y en caso de decidir mantener sus propios cambios podrá consultar las revisiones del atributo y volver a la versión que tenía su propio cambio.

Ningún cambio se pierde y en todo momento se puede decidir cuál es la versión correcta de los objetos y qué es lo que se quiere enviar al server.



# Visualizador de KBs...



Curso GeneXus | MÁS SOBRE GXSERVER

GeneXus training

Desde este Visualizador se puede, entre otras cosas:

- Tener información general y gráfica de la KB (cantidad de objetos, objetos no referenciados, etc.).
- Tener una visión total de la KB: Consultar todos los objetos (estructura, forms, reglas, eventos, variables), propiedades, etc.
- Editar la documentación (Main Document): Cuando posteriormente el desarrollador realice una operación de update, tendrá dicha documentación actualizada en la copia local de su KB.
- Tener un listado con las revisiones.
- Consultar configuraciones: Extensiones, User Controls (inclusive agregar user controls), Patterns, Indexer Monitor.
- Consultar el status de la licencia de GeneXus Server: Cantidad de días restantes, ingreso de autorización, etc.
- Eliminar una KB.



## Ciclo de vida de la KB...



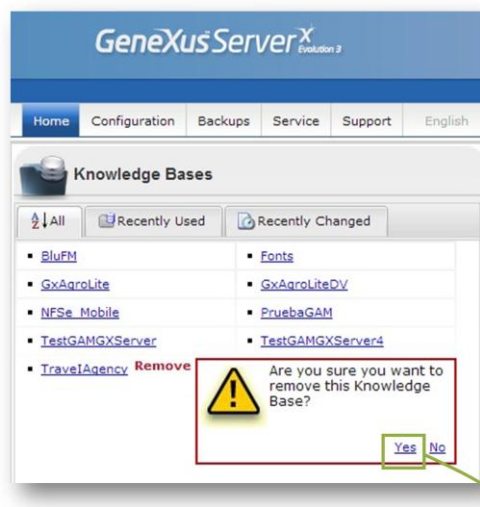
Otra cosa que permite el visualizador de KBs es llevar el ciclo de vida de la KB, definiendo allí las versiones que se van necesitando.

En esta pantalla se ve la línea principal de desarrollo. Las versiones congeladas representan liberaciones del producto, y a su vez líneas paralelas de desarrollo (por ejemplo para corregir errores).

De esta forma, al realizar la operación Create KB from server, se van a ver todas las versiones que hay en ella, y se podrá elegir sobre cuál hacer dicho create.

Cada KB de GeneXus quedará conectada a una versión del server.

## Eliminación de una KB...

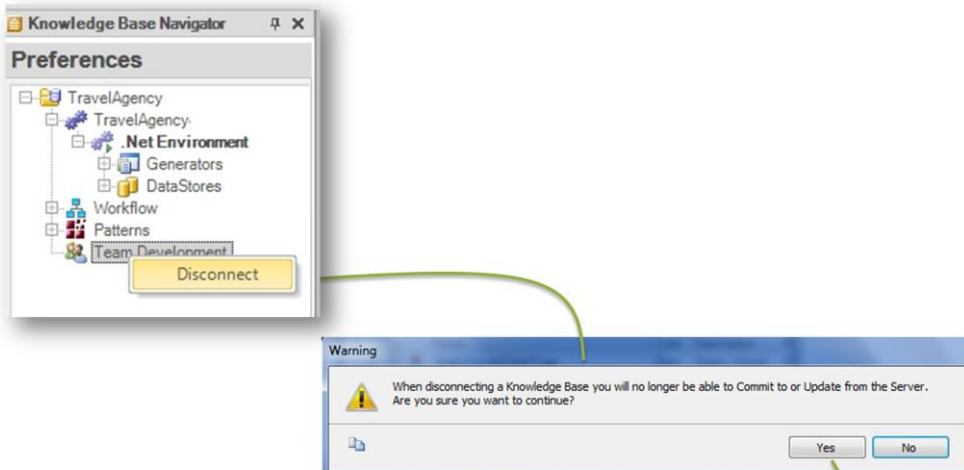


Se elimina la KB del server.

Para eliminar una KB del server bastará con deslizar el mouse sobre su nombre y aparecerá la opción Remove.

Esta operación no elimina físicamente la KB sino que solamente la borra del catálogo de KBs publicadas en el server.

## Disconnect from Server...



Nuevamente queda habilitada la operación **Send Knowledge Base to Server** en el menú File.

Para desconectar una KB local de un server, se deberá ejecutar la operación Disconnect from Server.

Desde la vista Preferences, bastará con presionar el botón derecho del mouse sobre el sector **Team Development** y seleccionar **Disconnect**. Esto hace que dicho usuario desconecte su KB local del server y pueda publicarla en otro server.

Al confirmarse el mensaje, automáticamente vuelve a quedar habilitada la operación **Send Knowledge Base to Server** desde el menú **File**. La KB que se desconecta NO se puede volver a reconectar, pero sí se puede enviar nuevamente al server.