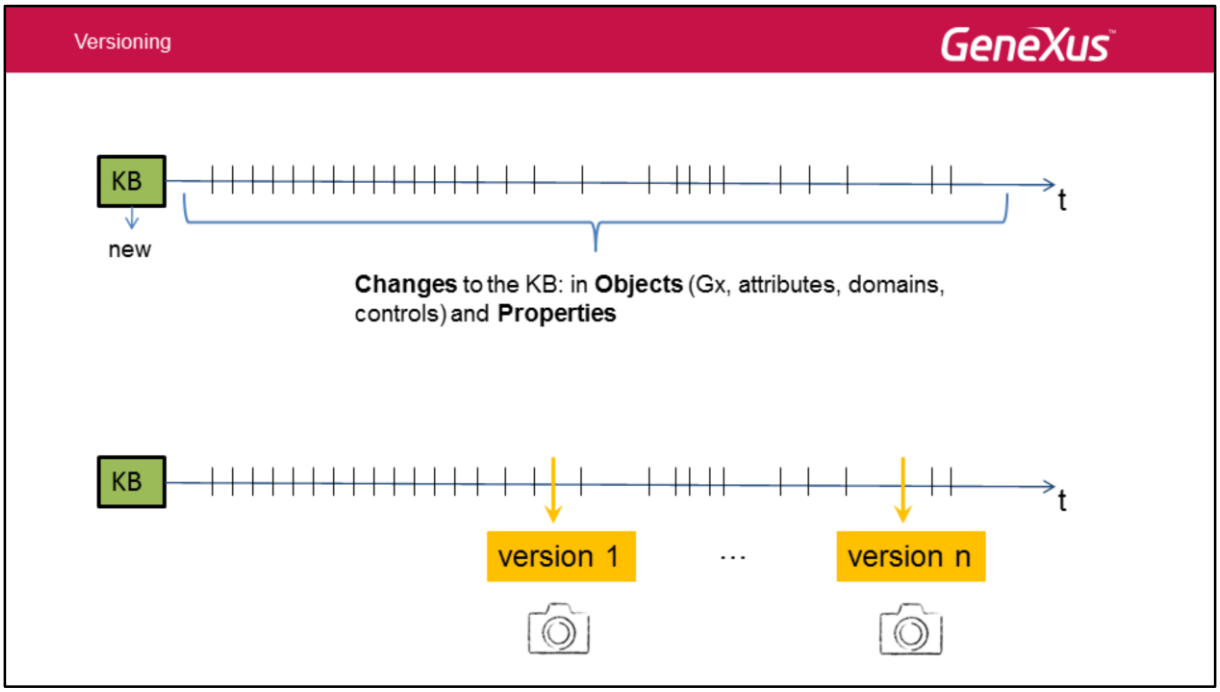
GeneXus X Evolution 3

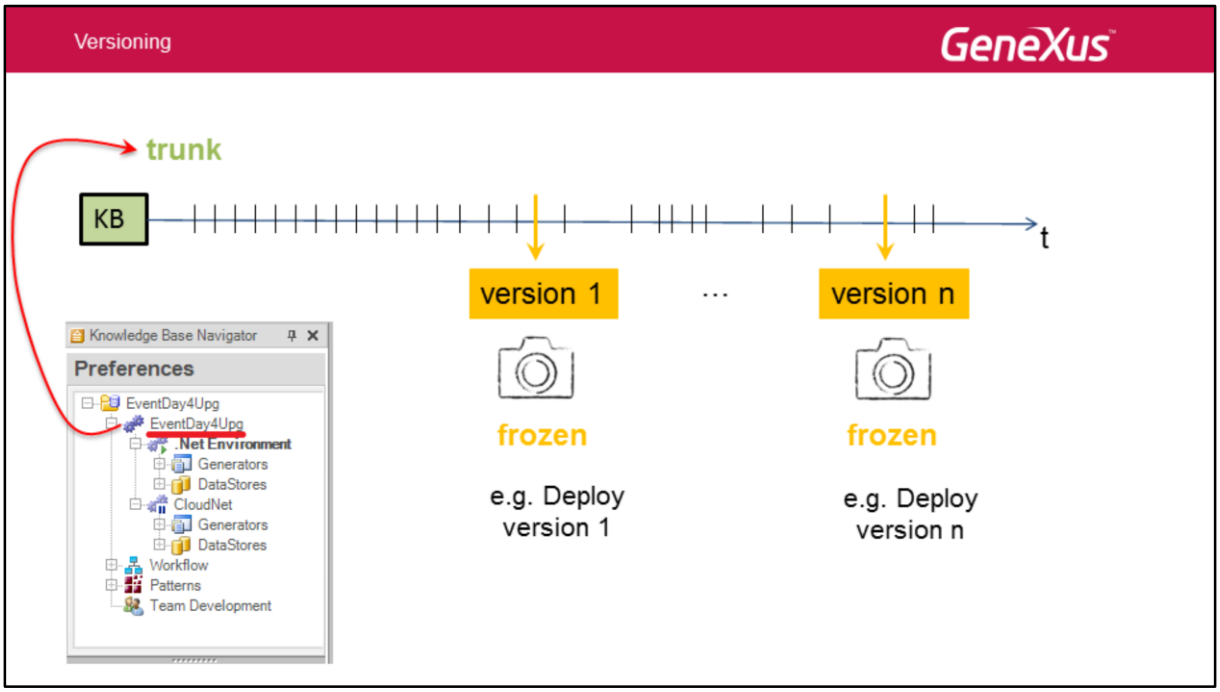
GeneXus

Knowledge Base Versioning



La vida de una KB se desarrolla en el tiempo, y va cambiando. A lo largo de esa vida se producen hitos —estados significativos de la KB— que es preciso aislar. Por ejemplo, cuando se pone en producción la versión 1 de la aplicación y en cada liberación subsecuente.

¿Cómo establecemos y aislamos esos hitos? Sacando una foto de la KB en un momento dado.



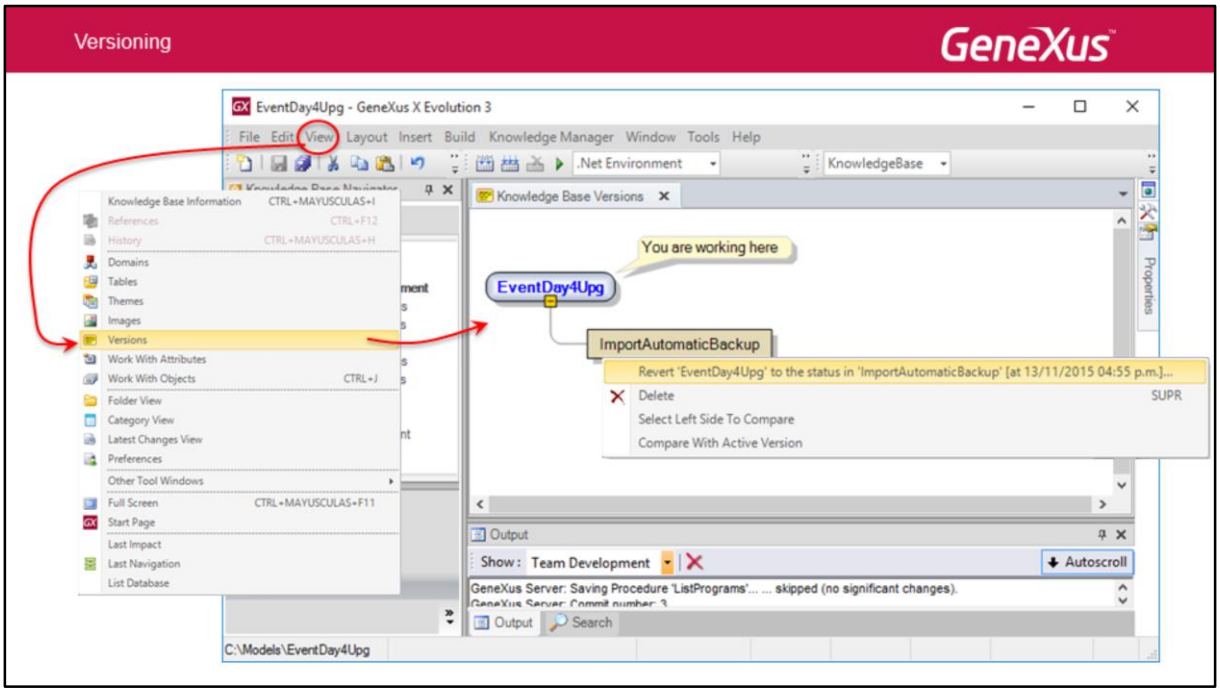
Cuando se crea una KB, se crea automáticamente una versión de desarrollo que denominamos “trunk” (tronco). Los objetos que se van creando, se hacen dentro de esta versión.

En la solapa Preferences del Knwoledge Base Navigator bajo el nodo raíz de la KB vemos en todo momento la versión activa. Hasta no crear otra, la única versión que existe es la trunk que asume el mismo nombre que la KB.

Observe que cada versión tiene sus environments.

El desarrollador comienza su desarrollo. Existen hitos a lo largo de la vida de esa KB, estados significativos que es preciso aislar, como cuando se pone en producción la aplicación.

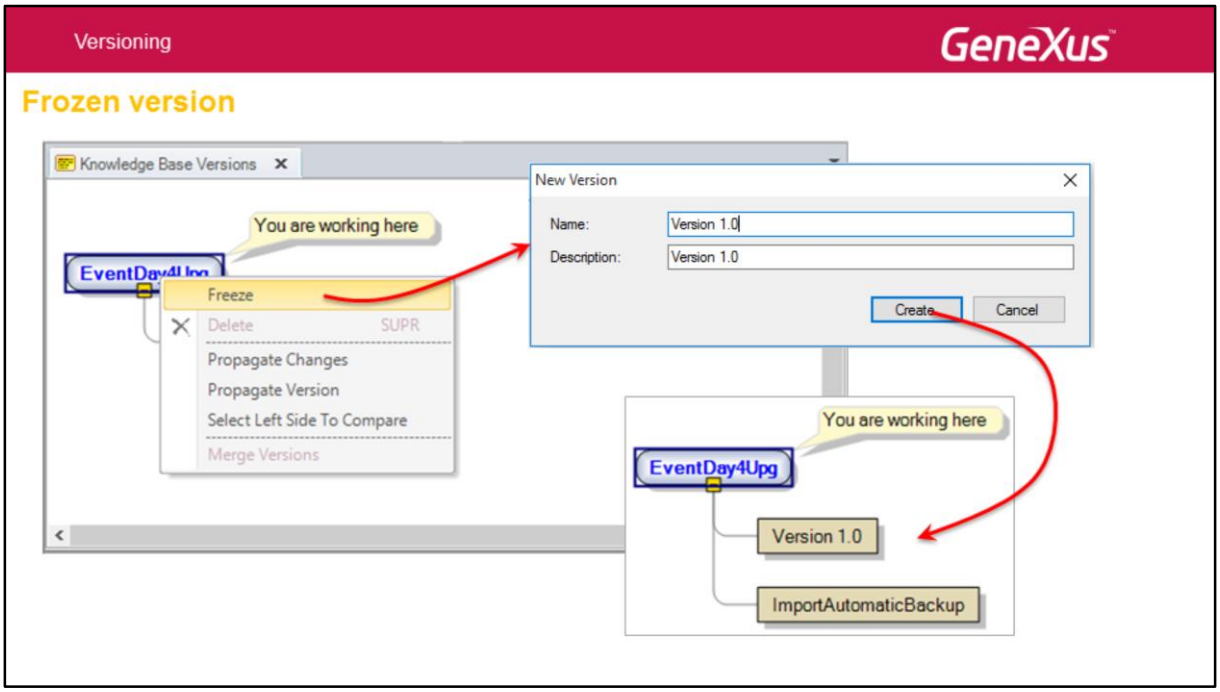
En esos momentos podemos congelar la versión, creando una Frozen version, que es como una foto de la versión en ese momento, una copia read only.



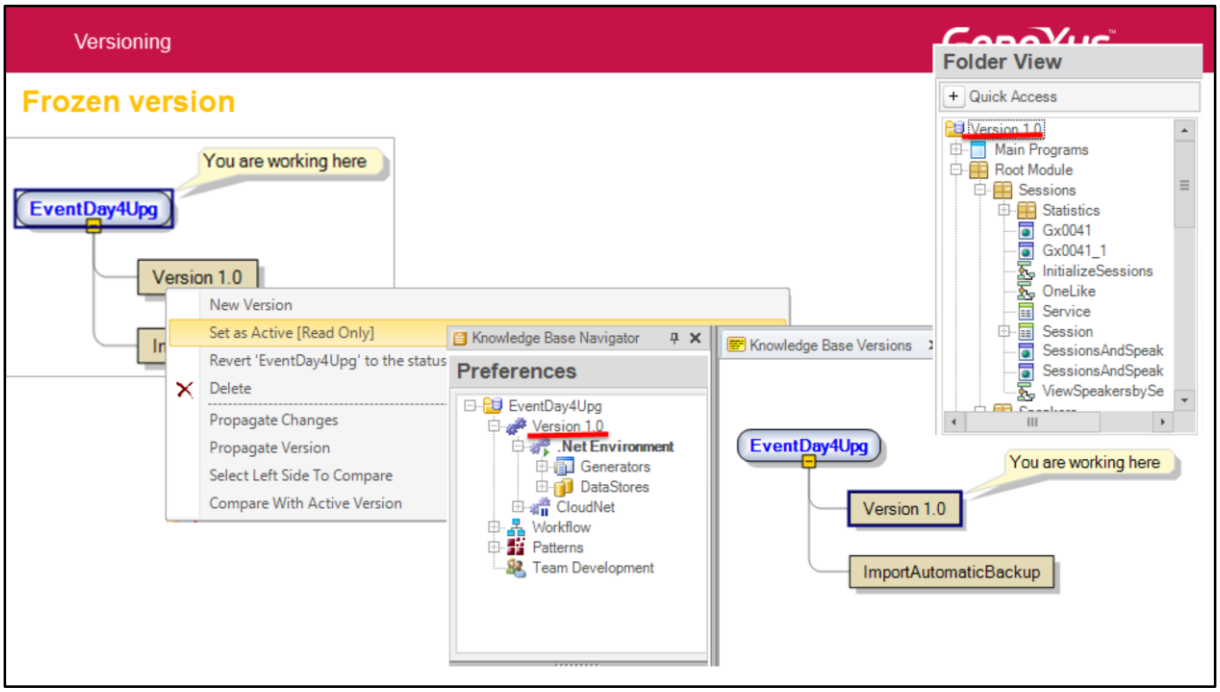
Para ello, contamos con la ventana **Knowledge Base Versions**, que se obtiene a través del menú View.

En este caso tenemos únicamente la versión Trunk, sobre la que estamos trabajando, y ya hay una versión congelada, de nombre “ImportAutomaticBackup”. Cada vez que se va a importar un xpz en la KB, antes de hacerlo se congela la versión, por si hay que deshacer los cambios luego de importar.

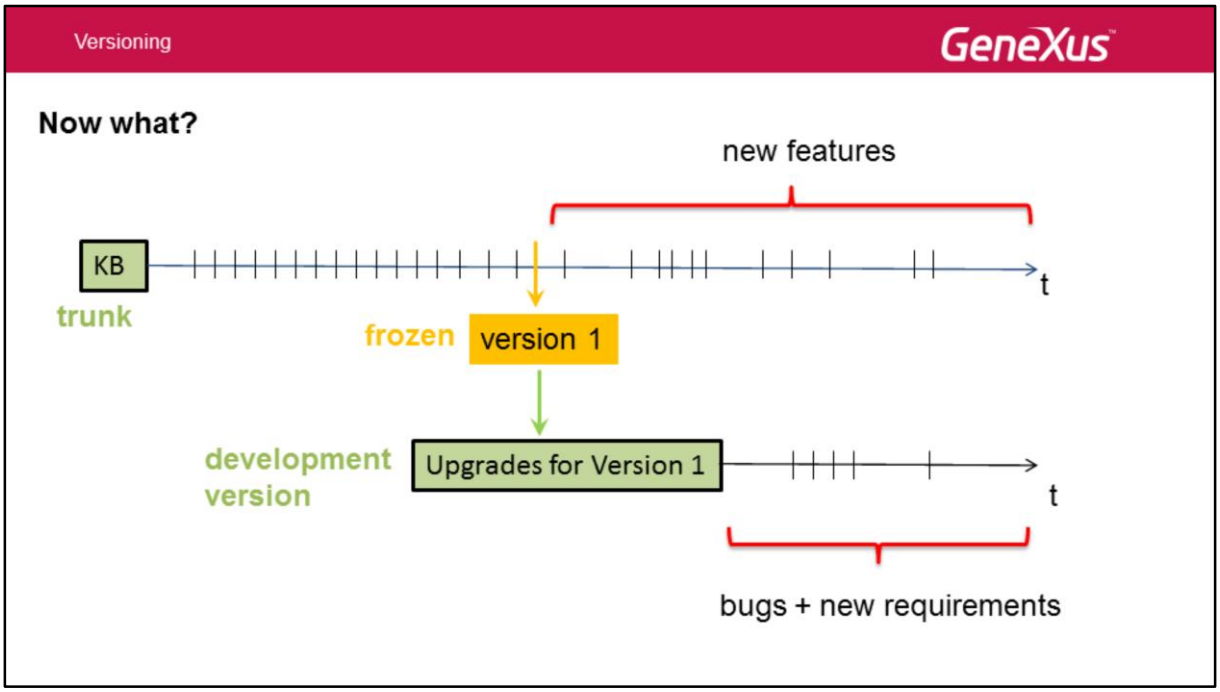
Supongamos que ya estamos listos para llevar a producción nuestra KB. Entonces creamos una Frozen version para tener una copia fiel del estado de la Kb en este momento, antes de seguirla modificando.



De esta manera creamos una Frozen version de la versión trunk de desarrollo. Observe cómo queda inmediatamente colgando de la versión de desarrollo de la que se tomó.



Con botón derecho sobre la versión congelada, vemos que podemos setearla como versión activa. De esta manera, en el árbol de las **Preferences** ahora veremos esta versión. Los objetos que aparecerán en el Folder View también corresponderán a los de esta versión. Como es una Frozen version, todo será Read Only.

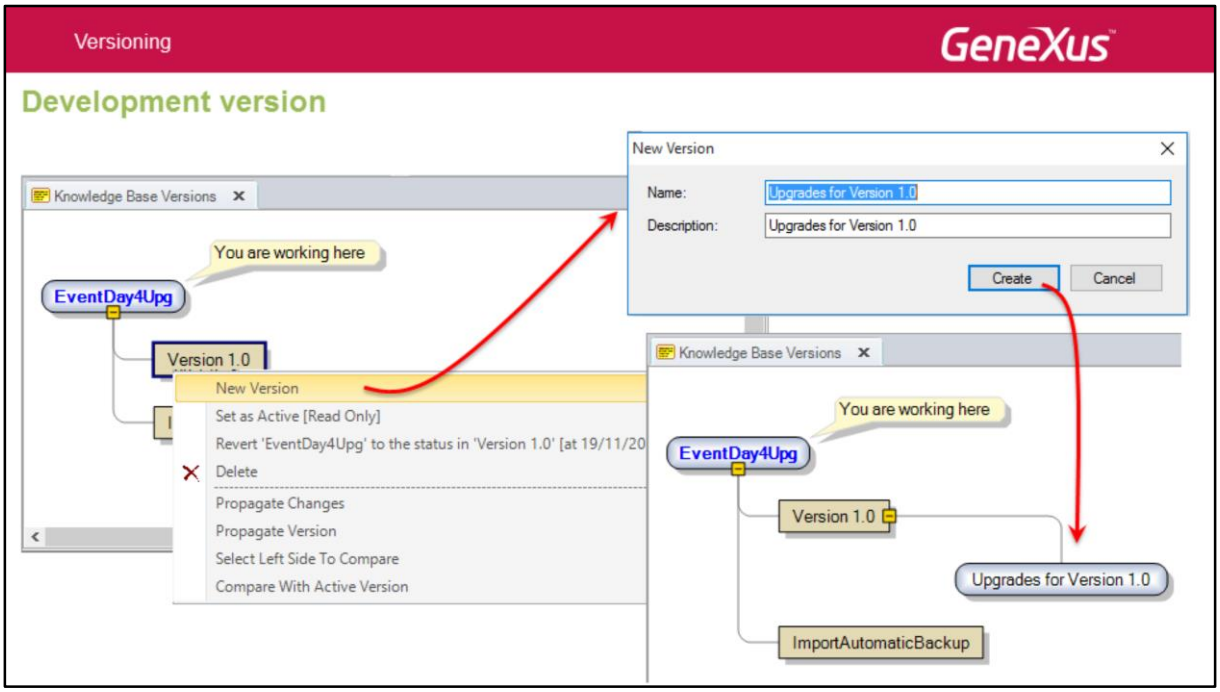


¿Qué vida sigue la aplicación después de la puesta en producción de la primera versión?

Normalmente el desarrollo sigue dos vertientes:

- corregir errores detectados por los usuarios e implementar cambios que éstos nos piden, e
- incorporar nuevas funcionalidades

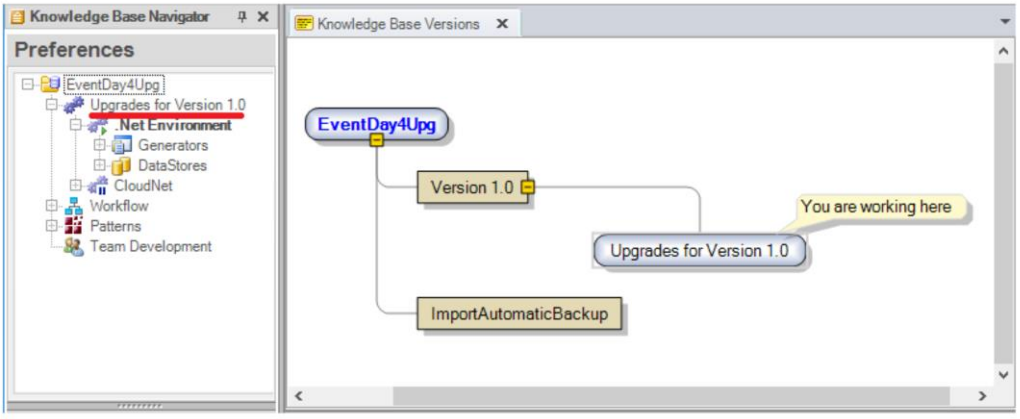
Para ello, podemos separar los desarrollos de estas dos vertientes: las nuevas funcionalidades se van incorporando en la versión trunk, y la corrección de bugs y la implementación de nuevas funcionalidades se realiza en una nueva versión de desarrollo, creada a partir de la versión congelada. De esta manera los cambios se paralelizan, sin afectarse mutuamente.



La versión de desarrollo se crea siempre a partir de una Frozen version. A diferencia de la frozen, esta no es readonly.

Se activa igual que una frozen.

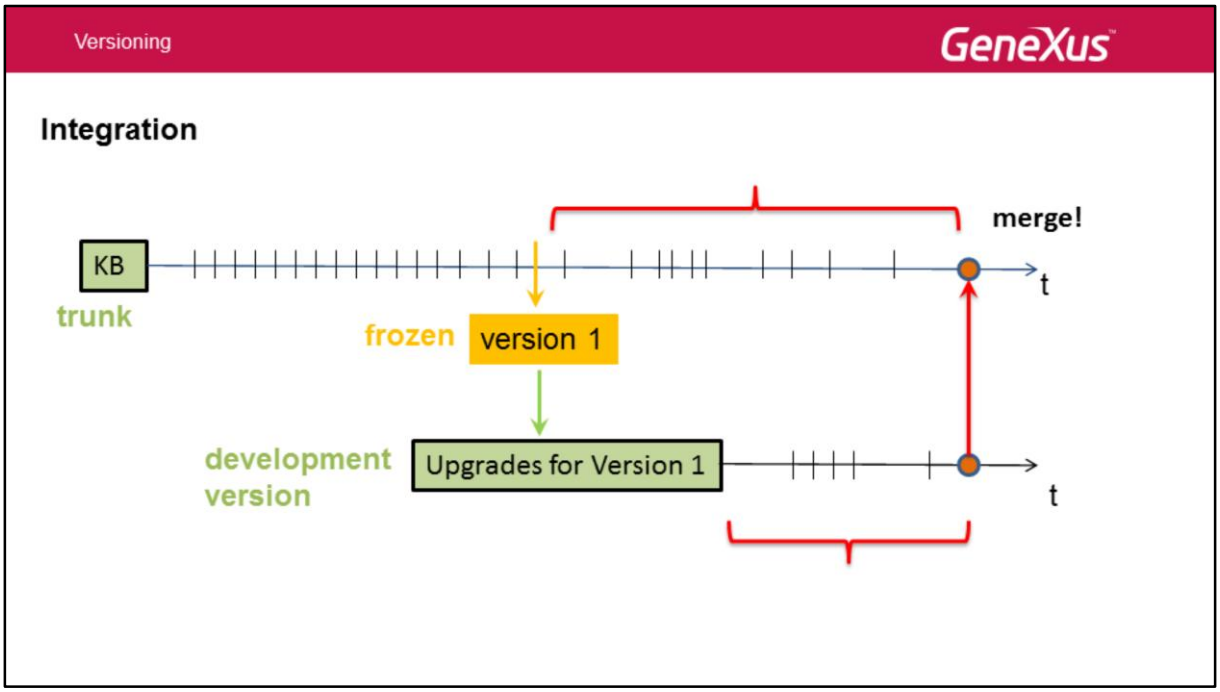
Development version



Hereda los mismos environments, y algunas de sus propiedades; no todas: no las propiedades de ejecución. Por ejemplo, en el environment .Net que prototipaba localmente el nombre de la base de datos aparece vacío.

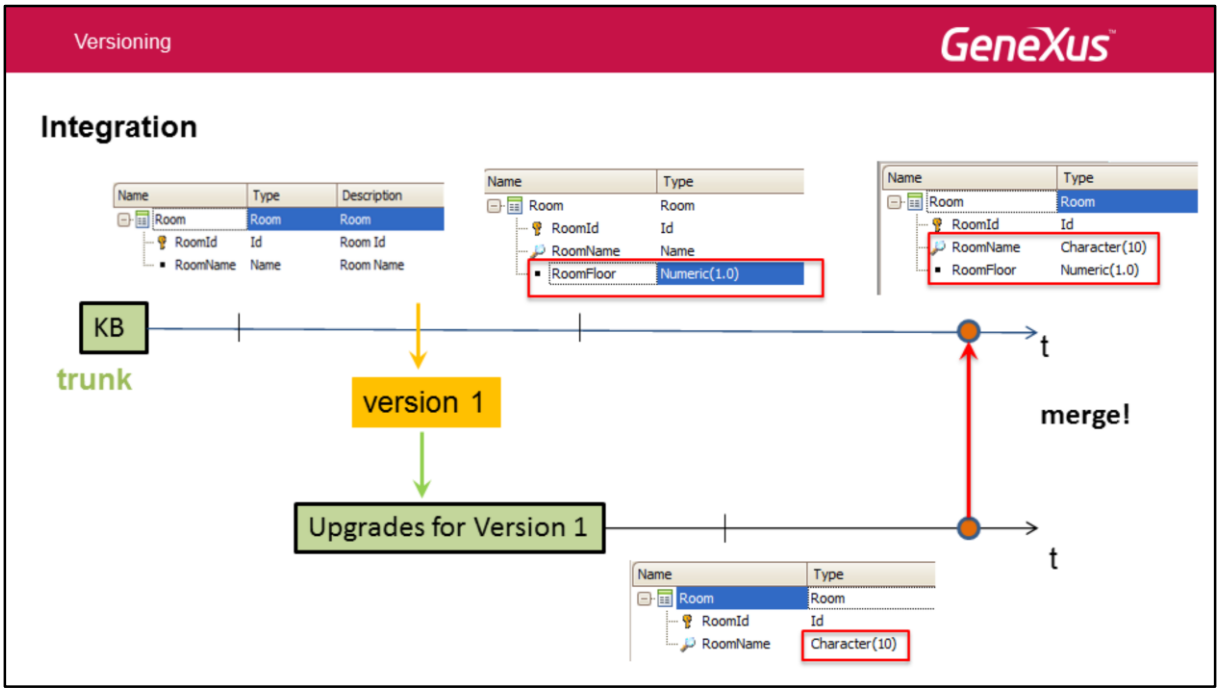
En esta versión se trabajará entonces corrigiendo los errores encontrados en la aplicación en producción e incorporando los cambios pedidos por el usuario.

Mientras en la versión trunk se sigue desarrollando la aplicación, para llevarla al siguiente estado (más evolucionado).

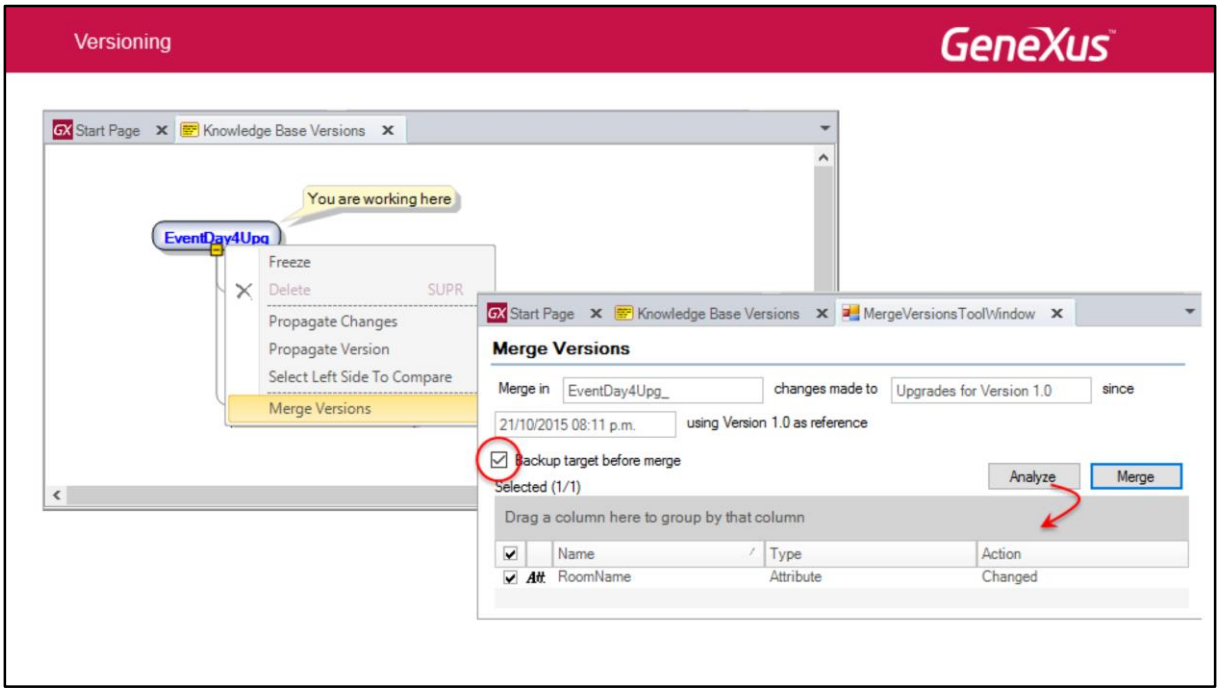


Llega un momento en que hay que unir los cambios de ambas versiones de desarrollo, para poder obtener una nueva versión liberable de la aplicación.

¿Cómo se hace el merge y cómo se trabaja con los posibles conflictos que surgieran a raíz del trabajo paralelo?



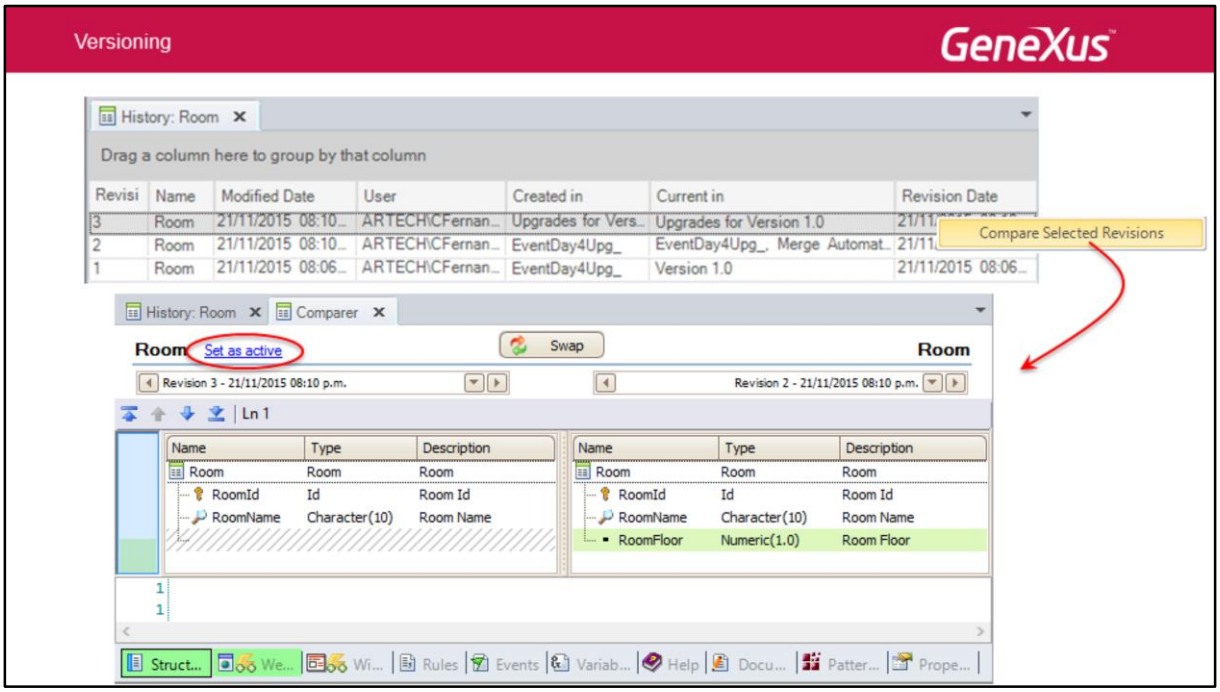
En este ejemplo, en el trunk se agrega a la transacción Room el atributo RoomFloor, y en la versión de desarrollo paralela se le modifica al atributo RoomName su tipo de datos. No hay conflicto entre ambos cambios, por lo que cuando se pida hacer el merge de la versión paralela en la versión trunk, se incorporarán ambos cambios a la vez.



Posicionado sobre el trunk, con botón derecho encontrará la opción Merge Versions, que le abre un diálogo que permite indicar que se quiere integrar en esta versión los cambios efectuados en la otra versión de desarrollo, usando como referencia la versión frozen Version 1.0.

Presionando Analyze le mostrará los cambios, para que usted pueda decidir qué quiere integrar. Presionamos Merge para realizar la integración.

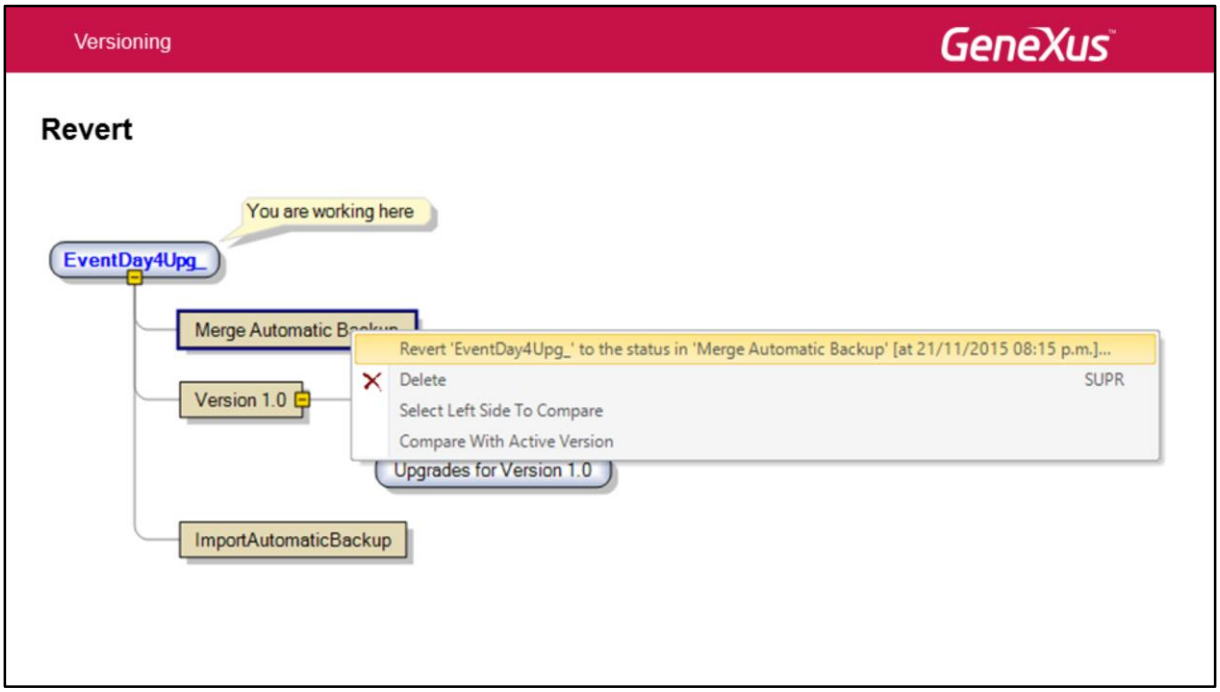
Observe que un check box le ofrece la posibilidad de armar una frozen version como backup de la versión sobre la que se hará el merge, antes de realizarlo.



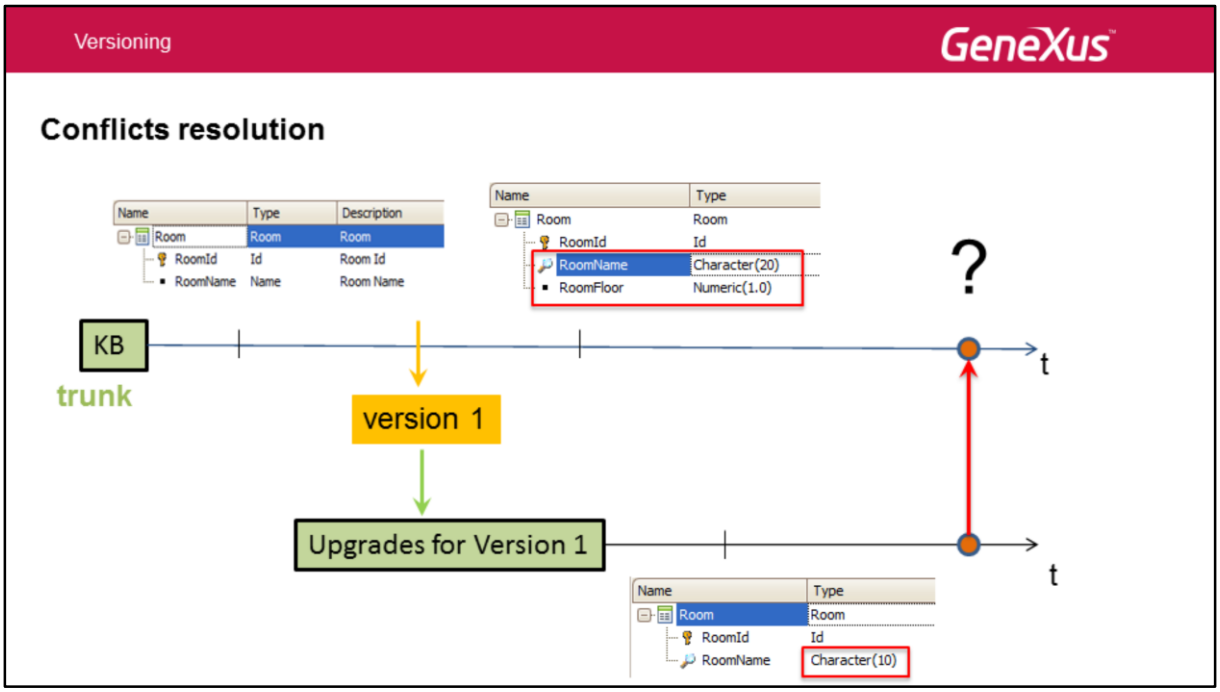
Accediendo al History de la transacción puede ver las distintas revisiones, en qué versión fueron creadas y en qué versiones son la revisión actual. En el ejemplo se ve que la última revisión, la 3, es la que corresponde al cambio que se realizó en la versión de desarrollo Upgrades for Version 1.0, y que es la versión actual allí. Mientras que la revisión 2 es la mergeada, versión actual en el trunk.

Se pueden comparar dos revisiones, seleccionándolas de la ventana de History y con botón derecho “Compare Selected Revisions”.

Observar que se puede setear como revisión activa en la versión en la que nos encontramos trabajando una que no es la activa allí.

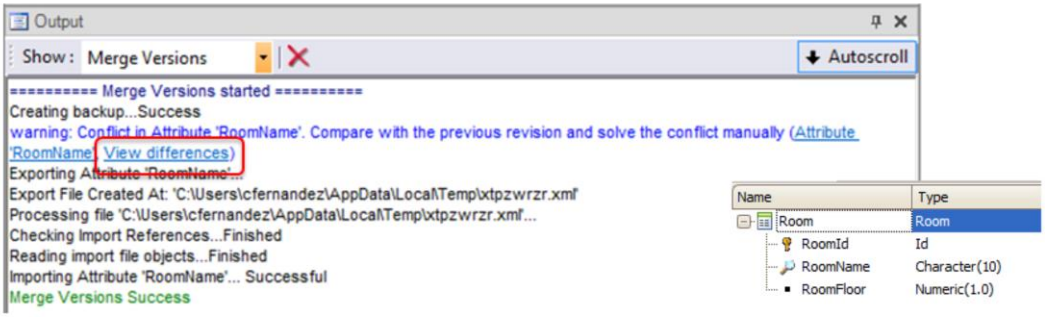


Si usted dejó el check box marcado cuando hizo el merge, se creó la frozen version correspondiente que le permite revertir el merge realizado.



Pero, ¿qué pasa cuando en ambas versiones de desarrollo se hicieron cambios sobre lo mismo? Por ejemplo, si en la versión trunk, además de agregarse el atributo RoomFloor se modificó el tipo de datos de RoomName, y paralelamente, en la versión Upgrades for Version 1 también se modificó ese tipo de datos.

Conflicts resolution

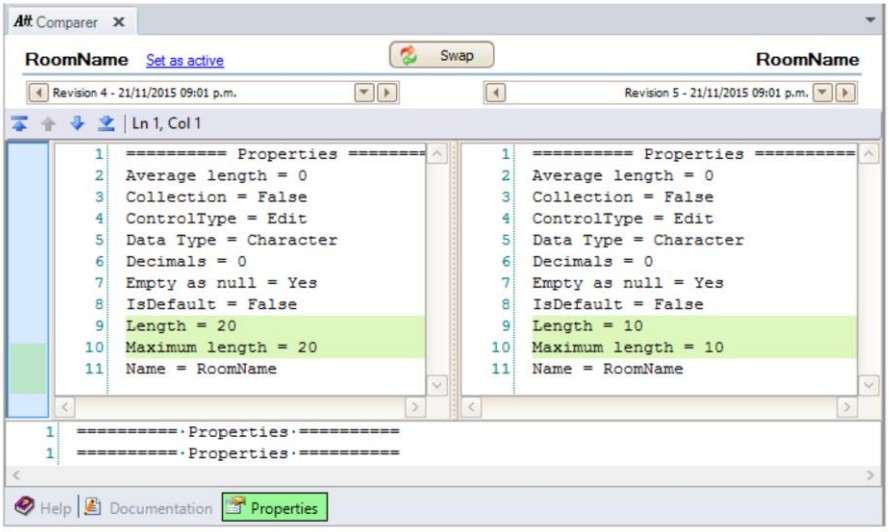


When conflict: Object is overwritten with the source version object revision

El merge de las versiones no falla, pero se le advierte al desarrollador que se encontró un conflicto en el atributo RoomName. Como consecuencia, el objeto es sobreescrito en la versión destino del merge de acuerdo al estado del objeto en la versión origen.

En la ventana de output se le ofrece visualizar las diferencias.

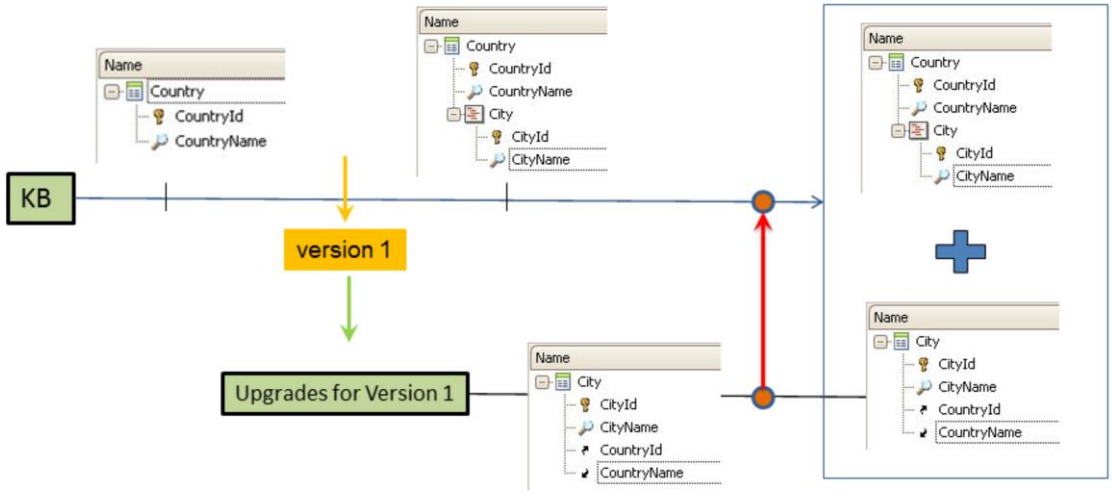
Conflicts resolution



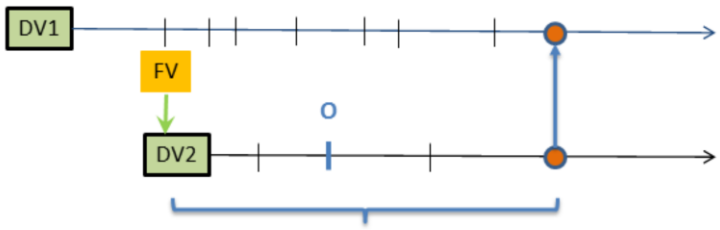
Aquí se ven las diferencias entre la revisión resultante del merge en el trunk (que es la que correspondía al objeto en la versión Updgrades for Version 1), y la versión que existía antes del merge en el trunk. Vea que puede activar la revisión anterior.

GeneXus

Integration by object (not by relationship)



Integration: all-or-nothing



O: GeneXus object, attribute, domain, control

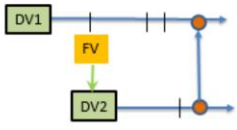
Merge is an **all-or-nothing** operation per object!
If there is a conflict in one part (i.e. rules) the other ones are not integrated, and the entire object is overwritten.

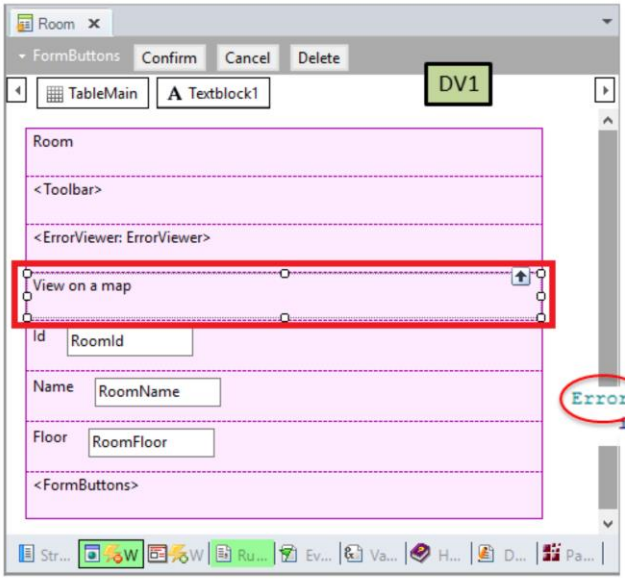
En el merge:

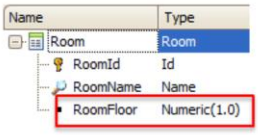
- Si O es nuevo en la development version origen (DV2), entonces se copia a la development version destino (DV1)
- Si O se eliminó en DV2 y en DV1 no se modificó desde que se congeló la versión de la que surge la DV2 (FV), ni es invocado por nadie en DV1, se elimina. De lo contrario se deja como está en DV1.
- Si O se modificó en DV2 y:
    - En DV1 no de modificó desde FV → se copia en DV1 sobreescribiendo
    - Se modificó también en DV1 → se intenta el merge (integrar las partes modificadas en DV1 y DV2), en caso de no haber conflicto (no haber modificado ambas versiones la misma parte del objeto). Si hay conflicto, se sobreescribe el objeto en DV1, con el objeto en DV2. Queda en manos del usuario resolver el conflicto, por ejemplo yendo al History donde tiene todas las revisiones del objeto.

GeneXus

Integration example

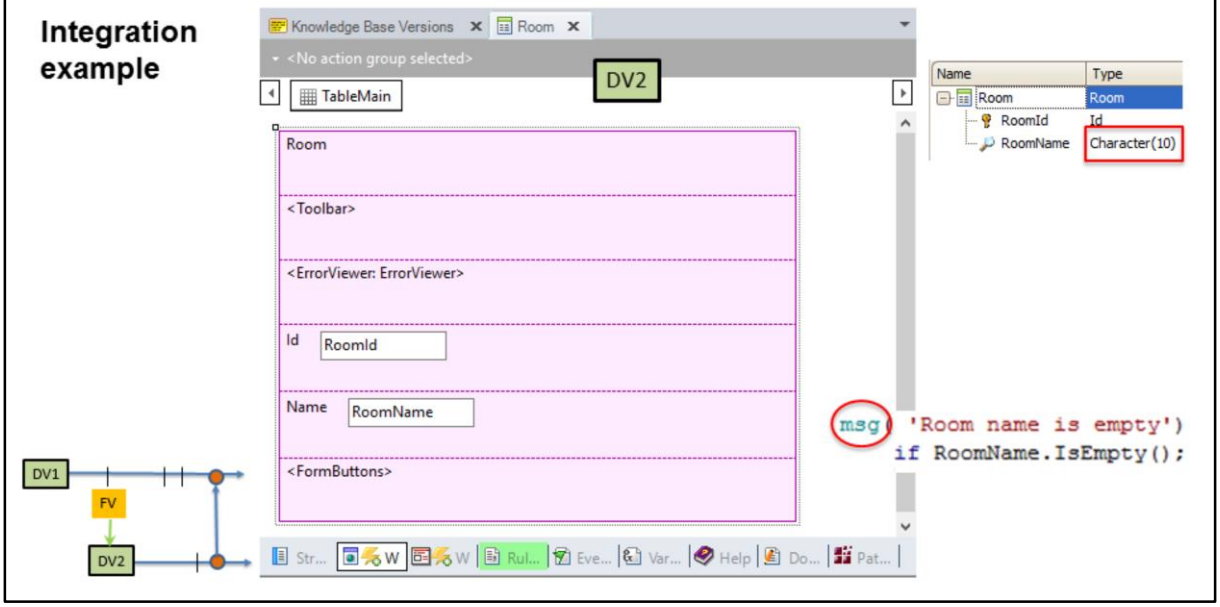


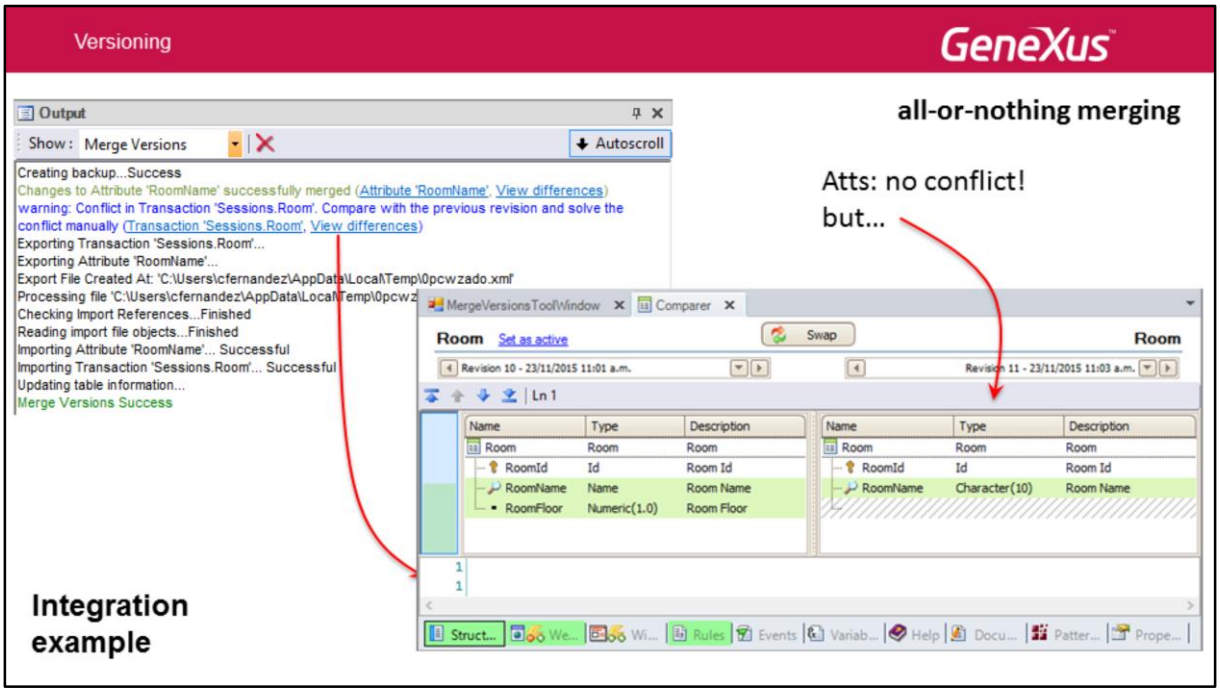




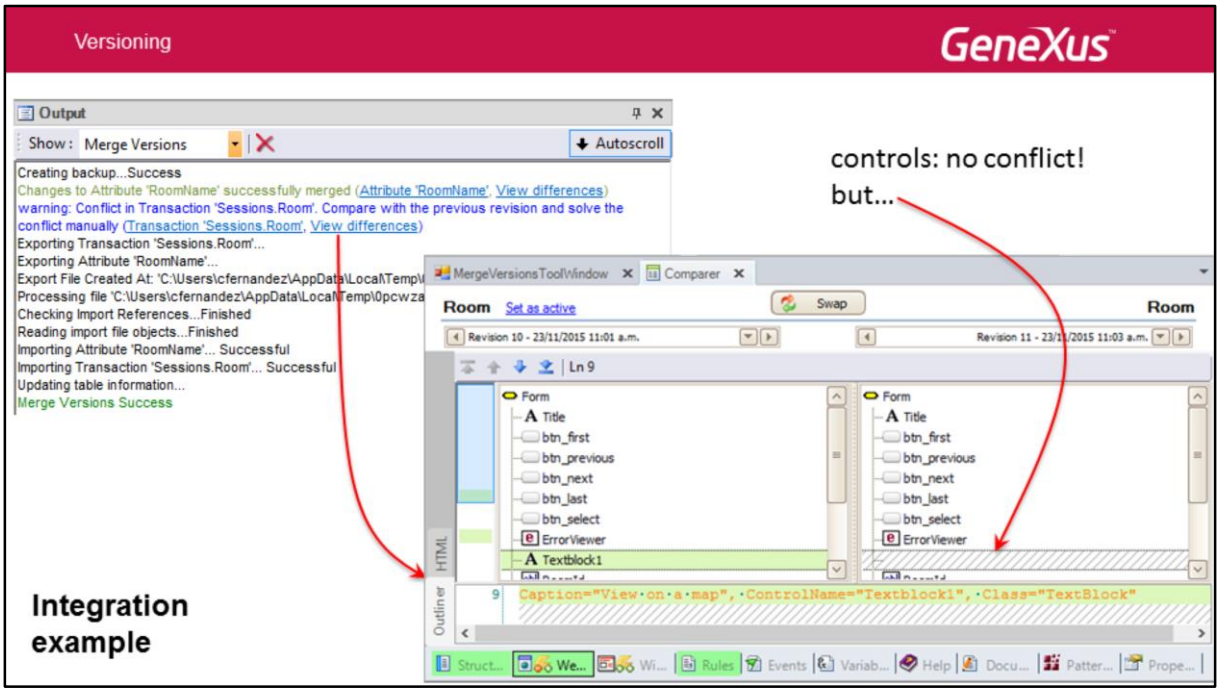
GeneXus

Integration example

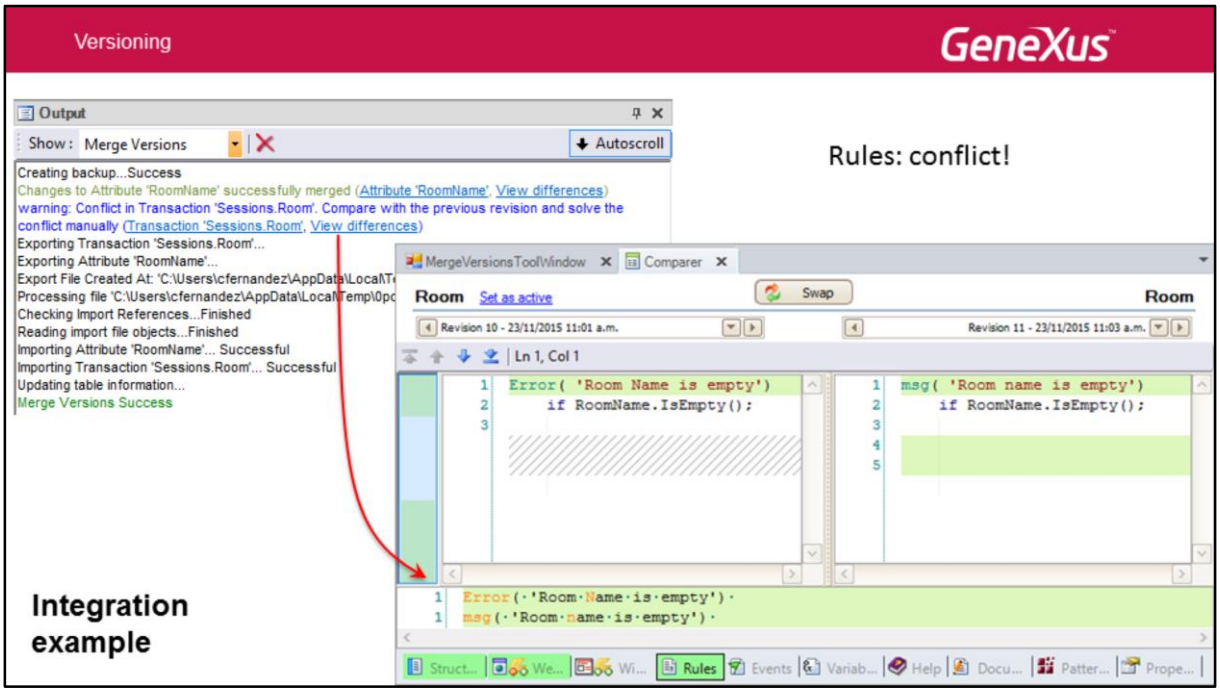




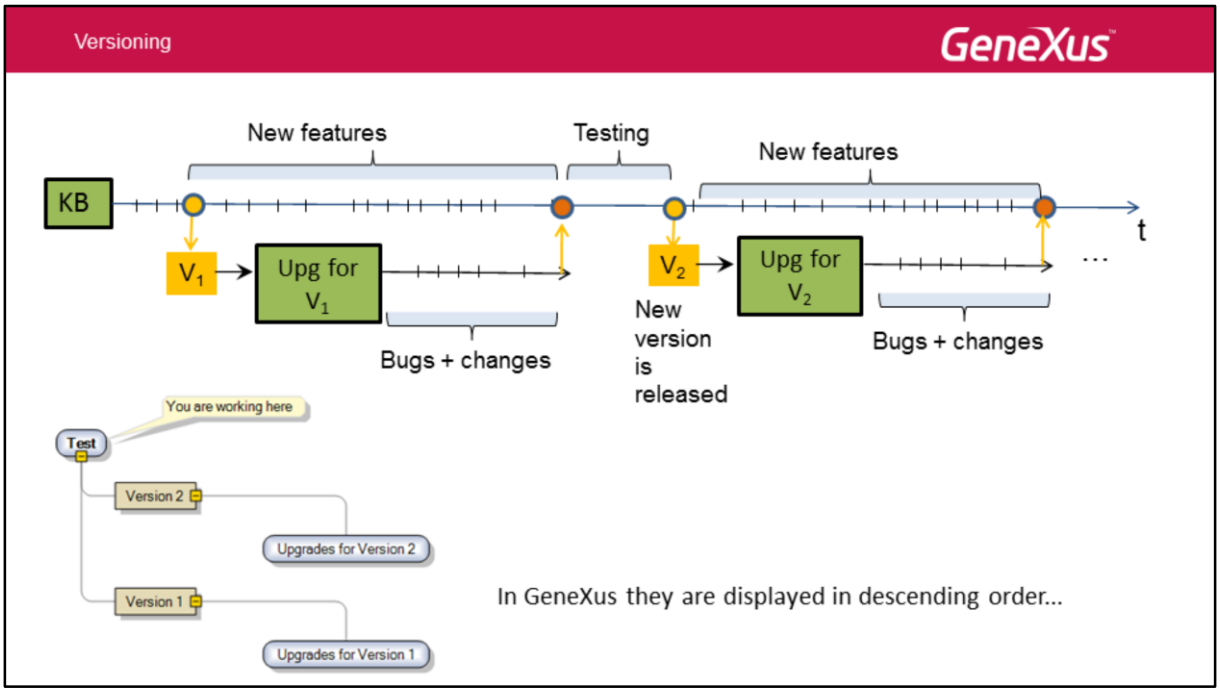
En cuanto a los atributos no hay conflicto, sin embargo veamos que la estructura final no quedó “mergeada”, sino quedó como en el objeto en la versión DV2. Esto es por el “todo o nada” del que hablábamos. En alguna parte del objeto se encontró un conflicto.



Aquí vemos lo mismo: no hay conflicto a nivel de los controles (en la versión DV1 se había agregado un text block que en la DV2 no se agregó). ¿Por qué en la versión resultante no está el text block? Evidentemente se encontró un conflicto que hizo que se sobreescribiera el objeto de acuerdo a la revisión correspondiente a DV2.



Aquí vemos el conflicto, y este conflicto es el que hace que el objeto entero sea sobreescrito.



Se pueden crear tantas versiones frozen y de desarrollo como se desee, de acuerdo a sus necesidades, estableciendo todo un árbol de versiones.

Some notes:

- GeneXusServer manges the same versioning mechanism
- ChangeDefender feature was implemented in order to manage changes between different KBs


GeneXus