Evolución de los objetivos de Genexus
hacia la Cuarta Dimensión

Preparado por: Breogán Gonda y Juan Nicolás Jodal [1]
Artech, febrero de 2009

¿Qué podemos decir, 15 años después, de los objetivos de Genexus? [2]: ¿Cómo han evolucionado?, ¿cómo ha sido su cumplimiento?

Los objetivos de Genexus han evolucionado constantemente y siempre hacia otros más exigentes en la medida que nuestras investigaciones, el desarrollo tecnológico general y un mejor conocimiento de la realidad lo han ido aconsejando.

Los hechos muestran un muy buen grado de cumplimiento de los objetivos.

Las Cuatro Dimensiones

Se ha conseguido una buena manera de visualizar los objetivos y evaluar su cumplimiento introduciendo cuatro dimensiones:

1. **Completitud** (Será de un 100% si tanto la base de datos como todos los programas necesarios son generados por Genexus)
2. **Productividad** (Es el aumento potencial de productividad obtenible con una buena utilización de Genexus respecto a la que obtendrían buenos desarrolladores utilizando manualmente lenguajes de programación comunes (como COBOL o RPG originalmente, Java o C# hoy)
3. **Universalidad** (Será el 100% si pueden generarse aplicaciones para cualquier plataforma viva)
4. **Usabilidad** (Cuantifica la facilidad de uso: será 100% si cualquier persona puede utilizarlo sin mayor capacitación específica)

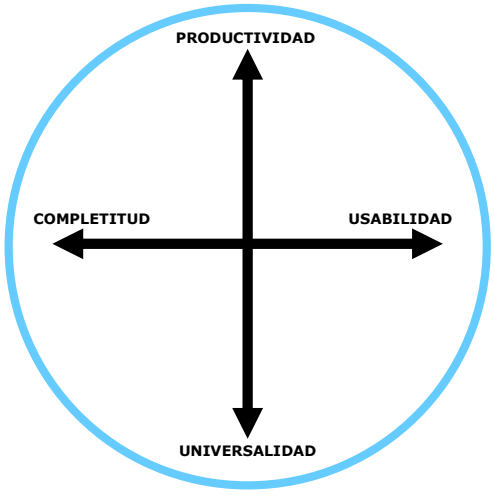
Es interesante ver como han evolucionado los objetivos y los logros desde la primer versión liberada de Genexus a fines de 1989.

Siempre es difícil representar cuatro dimensiones sobre un plano. La siguiente es una representación razonable e ilustrativa (aunque no rigurosa).

Representamos cada dimensión por una flecha según el diagrama siguiente y, a cada una de estas flechas, le atribuimos el valor que le asigna la evaluación de Genexus según esa dimensión.

Arbitrariamente colocamos las dimensiones de la siguiente manera: **Completitud** apuntando hacia la izquierda, **Productividad** hacia arriba, **Universalidad** hacia abajo y **Usabilidad** hacia la derecha, de acuerdo al siguiente gráfico.

LAS CUATRO DIMENSIONES



Objetivos en el momento de la liberación de la primera versión de Genexus

Cuando la primera versión de Genexus fue liberada ya tenía importantes logros en relación a la Completitud, Productividad y Usabilidad.

Completitud:

En la primera versión se tuvo como objetivo generar el 70% de los programas. Para cada uno de ellos, se generaba el 100% del código. De esta manera se evitaba la necesidad de modificar manualmente los programas generados.

Esta característica fue siempre considerada esencial: dado que el sistema tiene pleno conocimiento para generar tanto la Base de Datos como aquellos programas (los que caen dentro del 70% que Genexus era capaz de generar), también tiene conocimiento para efectuar el mantenimiento automático de todo lo que genera (Base de Datos -estructura y contenido - y programas).

Era claro, sin embargo, que lo ideal sería generar el 100% de los programas. Si ello fuera conseguido, Genexus sería capaz de efectuar automáticamente el mantenimiento de todo el sistema, con una disminución dramática de costos (tiempo y dinero). La tecnología de que se disponía en ese momento no lo permitía.

Productividad

La Productividad aumentaba de manera muy importante dado que el desarrollador no necesitaba dedicar su atención a un conjunto de tareas tradicionales: análisis de datos, proyecto de la base de datos, proyecto y escritura de los programas.

Las pruebas básicas de los sistemas consisten en verificar la corrección de las especificaciones por la vía de prototipos vivos, completos, muy fáciles de usar y oportunos.

El objetivo desde el primer momento fue un aumento potencial de productividad del 500% sobre programar manualmente con los lenguajes disponibles (en el caso RPG y COBOL).

Usabilidad

La Usabilidad aumentaba comparándola con la de programar manualmente (en aquella primera versión sobre el sistema operativo IBM OS/400, el Sistema de Gerencia de Base de Datos nativo del computador IBM AS/400, su lenguaje de comandos y un lenguaje de programación que podía ser RPG o COBOL).

¿Por qué aumentaba la usabilidad? Porque el desarrollador no necesitaba conocimiento detallado alguno de dichos elementos. Permitía entonces que el desarrollador se independizara de estos elementos de bajo nivel y pudiera dedicar su atención a entender y resolver conceptualmente el problema del cliente: dedicarse a resolver los problemas reales y no los que le presentan las limitaciones de la tecnología que utiliza.

No era fácil cuantificar este aumento de la usabilidad pero era importante, porque permitía tanto a viejos usuarios, acostumbrados a tecnología obsoleta, como a nuevos usuarios con poca o ninguna experiencia, usar la nueva tecnología de inmediato.

Los objetivos se cumplieron. ¿cuál fue la reacción de los clientes?, ¿por qué contrataron Genexus?, ¿qué características valoraron más?

Los principales elementos que llevaron a la contratación a los primeros clientes fueron:

- Genexus viabilizaba el desarrollo de aplicaciones para el nuevo computador IBM AS/400 a técnicos con muy poco conocimiento de ese nuevo computador y su tecnología.
- Genexus permitía un aumento muy grande en la productividad de los desarrolladores.
- **En particular no fue considerada como una ventaja esencial por parte de los potenciales clientes el hecho de que Genexus prometiera el mantenimiento automático de todo lo que generaba.** ¡Realmente nadie evaluaba seriamente esta ventaja porque nadie creía que el mantenimiento automático fuera posible!

A poco de usar Genexus, los clientes mantuvieron la evaluación positiva anterior: técnicos con muy poco conocimiento del AS/400 desarrollaban sin dificultades y con gran productividad su trabajo.

Sin embargo el mayor cambio fue que los clientes pasaron a evaluar como una ventaja fundamental que Genexus mantenía automáticamente todo lo que generaba.

Al mismo tiempo, Genexus sólo pretendía generar el 70% de los programas necesarios. Este hecho había sido acogido como razonable en el momento de la contratación pero luego pasaba a ser considerado una gran limitación para los clientes. ¿Por qué? Porque ahora les quedaban claras varias ventajas de Genexus:

- Fuerte aumento de la productividad de la programación automática sobre la manual.
- Enorme ventaja del mantenimiento automático sobre el mantenimiento manual.
- Sólo lo generado automáticamente podría ser mantenido también automáticamente.

Como consecuencia de todo esto, los usuarios quisieron evitar el desarrollo manual del restante 30% de los programas. Pero, más importante, la necesidad de mantener manualmente para siempre todos los programas desarrollados manualmente.

Como resultado de lo anterior se dieron dos cosas importantes:

- Todos comprendieron muy bien que en ningún caso debían modificar manualmente los programas generados, para preservar la capacidad de mantenimiento automático.
- Los clientes ejercieron sobre Artech una fuerte presión para lograr que Genexus generara el 100% de los programas.

Generar el 100% de los programas era un muy buen objetivo, por múltiples razones. Algunas de estas razones eran muy claras:

Mantenimiento automático del 100% de la aplicación. Aumento de productividad consiguiente.

Otras sólo quedaron claras años después:

Portabilidad de las aplicaciones de unas plataformas a otras.

Comercialización de Bases de Conocimiento, etc.

Pero el objetivo de generar y mantener automáticamente toda la aplicación no era un objetivo fácil de lograr (ninguna otra herramienta lo hacía - ni lo hace hoy - en todo el mundo) ¿sería posible?

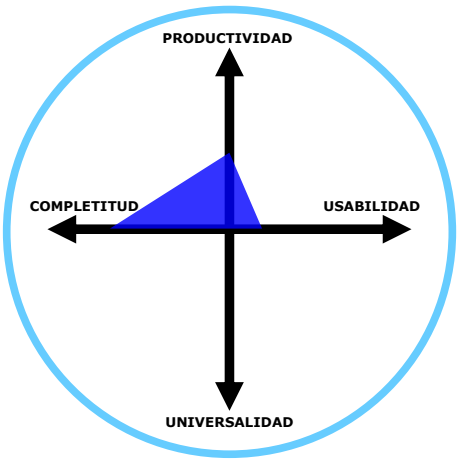
Para generar el 100% de una aplicación, primero es necesario poder describirla total y fielmente. En aquel momento, Genexus no tenía el suficiente poder expresivo para ello.

¿Cómo se podría adicionar rápidamente poder expresivo a Genexus? Genexus era 100% declarativo, si se le agregaba un lenguaje de programación – por ejemplo un lenguaje de 4ª Generación - se ganaría poder expresivo. Pero, al mismo tiempo, se perdería la capacidad de mantener automáticamente todo lo que se generaba, porque en los lenguajes procedurales **[3]** conocidos, los programas fuente no se mantenían válidos ante modificaciones de la Base de Datos.

Lo que se necesitaba era un lenguaje procedural cuyos programas fuente se mantengan válidos ante modificaciones en la base de datos (y que sea del mayor nivel posible).

No se consiguió ninguna solución inmediata y se colocó el asunto en primer lugar en la lista de investigación.

OBJETIVOS EN EL MOMENTO DE LA PRIMERA VERSIÓN DE GENEXUS (1990)

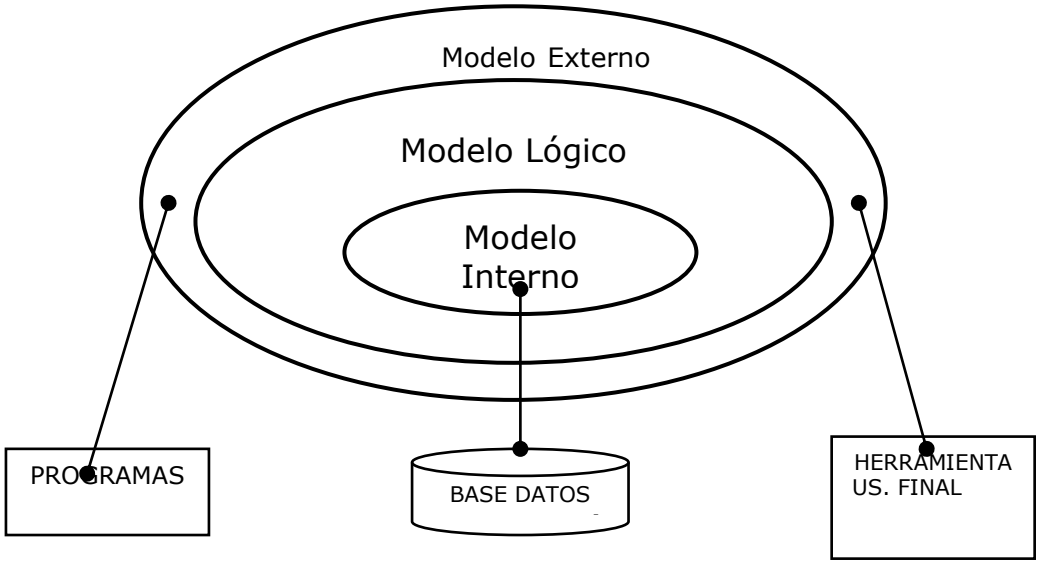


Objetivos de 1992: 100% de Completitud

Luego de considerar y descartar muchas alternativas se llegó a un planteamiento simple del problema:

Solución a través del Sistema de Gerencia de Base de Datos. La solución más razonable (e independiente de Genexus) era la implementación por parte de los fabricantes de Sistemas de Gerencia de Base de Datos de la recomendación ANSI Sparc del Esquema de tres modelos (Externo, Lógico, Interno). Todas las aplicaciones interactuarían con el Modelo Externo que no se altera ante modificaciones en la Base de Datos

Recomendación ANSI Sparc



Las tendencias que se percibían en la época hacían muy poco probable esta solución: el mercado de los Sistemas de Gerencia de Base de Datos, donde habían actuado múltiples empresas con diferentes productos, compitiendo permanentemente en un clima de innovación y entusiasmo, adoptaba como estándar el SQL en una versión nada innovadora.

El estándar implicó una quasi congelación de las funcionalidades y la supervivencia de unos pocos de todos aquellos fabricantes.

Conclusión: no había (ni hay) un camino por aquí.

"Bases de Datos Estables". Muchos teóricos postularon las "bases de datos estables, bien diseñadas a priori" y mucho se habló de ello (Obviamente si la base de datos es estable el problema que estamos tratando de resolver no existe: como no hay modificaciones estructurales en la base de datos, no hay repercusiones de ellas sobre los programas).

Sin embargo estos conceptos no se compadecen con la realidad: sólo pueden existir Bases de Datos Estables en empresas u organizaciones decadentes, que han perdido toda capacidad de innovación.

Conclusión: se trata de un abordaje que no tiene ningún contacto con la realidad práctica.

Lenguaje procedural cuyos programas sean independientes de la estructura de la Base de Datos. La tercera opción era diseñar y desarrollar un lenguaje procedural tal que la validez de sus programas fuente no fuera afectada por los cambios en la base de datos.

Adoptamos este abordaje y, algo que es muy difícil o imposible tomado aisladamente, resultó totalmente posible en un ambiente basado en conocimiento como Genexus: ¿por qué colocar manualmente en los programas elementos (nombres de tablas, archivos y similares) que pueden ser inferidos automáticamente (en el momento oportuno, que en este caso es el de la generación de los programas)?

El lenguaje procedural de Genexus actúa sólo sobre el Modelo Externo (cuyos elementos no son afectados por los cambios en la Base de Datos) y no utiliza elementos físicos de bajo nivel como tablas, archivos, etc. y sus programas fuente son inmunes a las modificaciones estructurales de la Base de Datos.

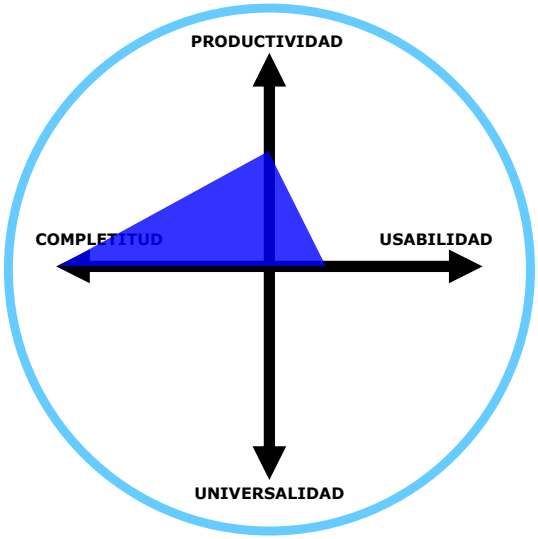
La inclusión de este lenguaje procedural de alto nivel permitió resolver con Genexus el 100% del problema.

Ahora Genexus era capaz de generar y mantener automáticamente la Base de Datos (estructura y contenidos) y el 100% de los programas.

De esta manera se lograba el 100% de Completitud y se asumía el compromiso de mantenerlo siempre en el futuro. Esto aumentaba sustancialmente las ventajas de usar Genexus. Los usuarios así lo entendieron.

De todas formas, Genexus era una herramienta que generaba básicamente aplicaciones para una sola plataforma: el computador IBM AS/400. Sin embargo, no había restricción teórica alguna para generalizarlo a otras plataformas.

OBJETIVOS DE 1992: 100% DE COMPLETITUD



Objetivos de 1995: soporte de la arquitectura Cliente / Servidor

En 1995 ocurrió algo largamente esperado y postergado: una fuerte expansión de la arquitectura Cliente / Servidor.

Artech lanzó generadores Cliente / Servidor para los Sistemas de Gerencia de Base de Datos más importantes de la época: IBM DB2, IBM DB2 para AS/400, Informix, Microsoft SQL Server y Oracle.

Estas nuevas capacidades tuvieron muy buena acogida y el segmento de clientes usuarios de la arquitectura Cliente / Servidor pasó a ser rápidamente el de mayor crecimiento.

Paralelamente, de pronto y sin que muchos lo esperaran, la liberación para fines comerciales de Internet constituyó un tremendo éxito y la informática toda comenzó a transformarse de una manera vertiginosa.

Hasta entonces los sistemas eran previsibles y estructurados, para algunos pocos millones de usuarios en todo el mundo. Estos usuarios los utilizaban sin grado de libertad alguno y luego de ser especialmente entrenados para ello.

Los desarrolladores – relativamente pocos en todo el mundo – mantenían para si todas las decisiones y los grados de libertad.

Frente a esa realidad tradicional, aparecía Internet, accesible desde el comienzo por mucho más gente de cualquier parte del mundo, que no podía ser entrenada y con un nivel de libertad mucho más grande. Genexus muy rápidamente lanzó su primer Generador para Web.

¿Cuál era la situación a fines de 1995?

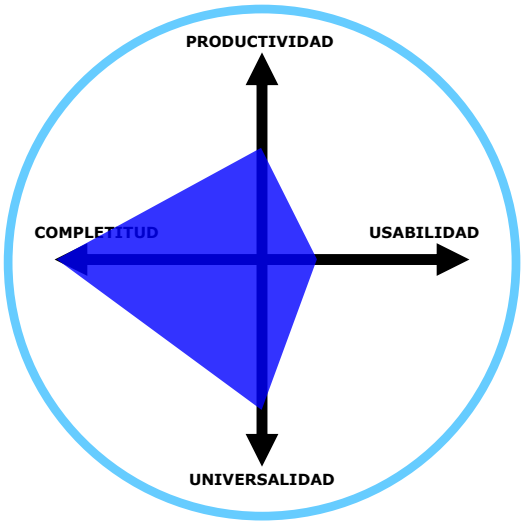
Genexus mantenía su **Completitud**: el 100% de los programas eran generados y mantenidos automáticamente.

La **Productividad** se mantenía muy alta (potencialmente un 500% de la que se podría llegar a obtener programando manualmente).

La **Universalidad** se había incrementado sustancialmente al incorporar las arquitecturas Cliente / Servidor y Web. ¿Qué quedaba fuera del alcance de Genexus? Básicamente aplicaciones para mainframes (pero ya era notorio que no existía una tendencia a desarrollar nuevas aplicaciones para ellos).

La **Usabilidad** se mantenía en los niveles anteriores.

OBJETIVOS DE 1995 SOPORTE DE LA ARQUITECTURA CLIENTE / SERVIDOR



Objetivos 2001: aumento radical de la Universalidad

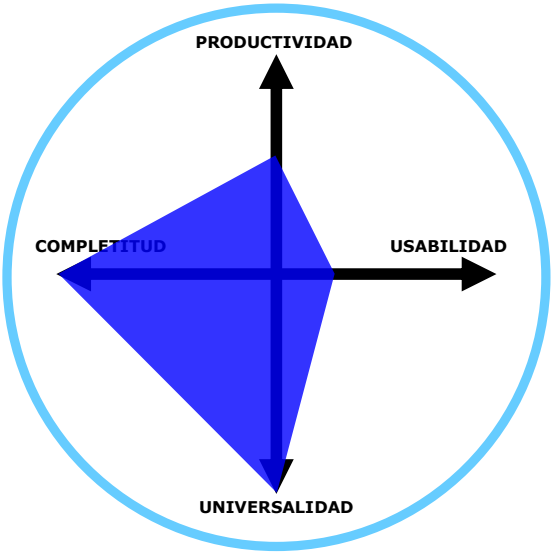
En la segunda mitad de la década de los 90 apareció el concepto de "plataformas" de ejecución de aplicaciones multi capa, orientadas a la red y que ofrecían algo que los usuarios pedían: actualización automática de las versiones de los programas en los PCs clientes.

Las dos plataformas rivales Java y .NET rápidamente se dividieron una buena parte del mercado (qué con su competencia permanente hicieron crecer). Java y .NET compartían el mercado fundamentalmente con la arquitectura Cliente / Servidor, mientras comenzaba el crecimiento de la arquitectura Web y disminuía sustancialmente la participación de los mainframes.

Genexus lanzó muy oportunamente sus generadores para Java y .NET y paralelamente perfeccionaba fuertemente sus generadores para arquitectura Web.

Con esto mantuvo sus buenos indicadores en lo que atañe a **Completitud**, **Productividad** y **Usabilidad** y aumentó sustancialmente su **Universalidad**: ahora Genexus generaba para todas las plataformas para las que, realmente, se estaban desarrollando aplicaciones nuevas.

OBJETIVOS 2001 AUMENTO RADICAL DE LA UNIVERSALIDAD



Objetivos de 2004: aumento radical de la Productividad

La informática ha cambiado mucho desde la liberalización de Internet: hay muchos más usuarios (muchos cientos de millones más). **Estos usuarios, en general, no son entrenables**.

Las Bases de Datos ya no son sólo las Bases de Datos físicas que tenemos dentro de la empresa sino Bases de Datos Extendidas que involucran a clientes, proveedores, servicios Web públicos o privados, etc. Simultáneamente, nuestras Bases de Datos son accedidas por otros, convenientemente autorizados, pero de las formas más diversas.

Los dispositivos que actúan como terminales se han diversificado añadiendo Hand Helds, Palms, Pocket PCs, teléfonos celulares, etc.

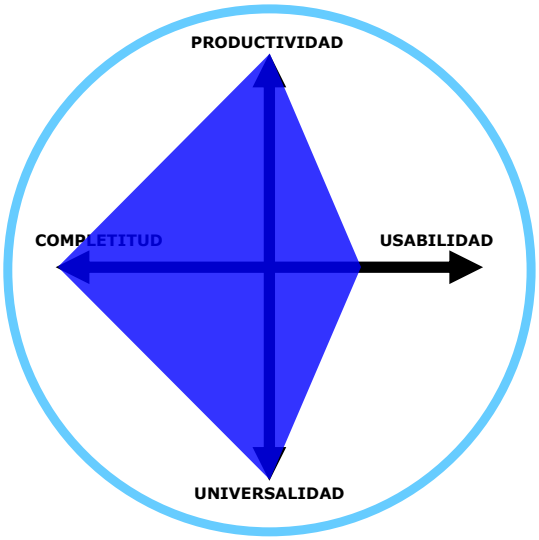
Las redes se han generalizado incluyendo ahora velocidades mucho mayores, conexiones inalámbricas de corta distancia (Wi Fi), conexiones inalámbricas de distancia media (Wi Max), conexiones telefónicas de alta velocidad, asociadas a las redes de telefonía celular (GPRS, EDGE).

Pero lo fundamental es que, ahora, nuestras aplicaciones deben servir a muchos millones de usuarios potenciales, que tienen un grado de libertad mucho mayor del acostumbrado y que no son entrenables. Como consecuencia las aplicaciones son mucho más complejas, necesitan ser mucho más estudiadas y elaboradas, de manera de esconder toda esa complejidad y presentarse al usuario final con interfases muy simples, muy intuitivas, muy fáciles de usar (por usuarios generalmente no catalogados y, por ello, no entrenables).

Ante esto Artech concluyó que un aumento potencial de productividad del 500% sobre la productividad de desarrollar a mano en un lenguaje de bajo nivel común (como Java o C#, por ejemplo) no será más suficiente en un futuro próximo. Entonces se asumió como objetivo aumentar la productividad sustituyendo el tradicional 500% por un 2000%, el que se logra con dos aumentos de productividad del 100% cada uno sobre la de la versión inmediatamente anterior, el primero en la versión liberada a fines de 2005 y el segundo en la versión a liberar a mediados del 2007.

Paralelamente se siguieron cuidando las demás dimensiones y, en particular la Universalidad, agregando oportunamente soporte para diferentes plataformas o elementos de plataformas que fueron surgiendo y pasaron a ser importantes en el mercado, como, por ejemplo, PostgreSQL, MySQL, AJAX.

OBJETIVOS DE 2004: AUMENTO RADICAL DE LA PRODUCTIVIDAD



El futuro: aumento radical de la Usabilidad

¿Y ahora?, ¿cuál es la próxima innovación?, ¿qué podemos esperar de Genexus?:

Genexus ha resuelto muy bien sus tres primeras dimensiones:

La **Completitud** y **Universalidad** han llegado al óptimo y habrá que trabajar permanentemente para mantenerlas ante las innovaciones tecnológicas que se aproximan.

Productividad: el aumento de productividad que se está adicionando es tal que permitirá desarrollar con Genexus - en tiempo hábil y con costos razonables - las aplicaciones cada vez más complejas que los diferentes usuarios requerirán en los próximos años. Esto será muy importante porque no será más posible hacerlo a mano con lenguajes de programación comunes como Java y C#.

¿Cuál es, en resumen, la situación respecto a la Cuarta Dimensión (la usabilidad)?

La **Usabilidad** es buena, pero no puede compararse su nivel con el de las otras tres dimensiones: Genexus es una herramienta para desarrolladores con una buena formación algorítmica, por ejemplo, porque su componente procedural se mantiene importante. Para aumentar en forma sustancial su Usabilidad, para que una persona cualquiera, con buena formación general, pueda usarlo en beneficio de sus tareas normales sin una costosa capacitación específica, debe facilitarse mucho más aún su uso.

Pero ¿cuál es la situación actual? La Usabilidad de Genexus ha sido importante desde el principio, y hoy lo es como queda claro en lo siguiente:

No es imprescindible conocer, cuando se comienza un desarrollo, cuál será la plataforma de ejecución (Hardware, Sistema Operativo, Sistema de Gerencia de Base de Datos, Arquitectura, Lenguaje de Programación a utilizar).

Los desarrolladores no necesitan nunca un conocimiento detallado de la plataforma de ejecución. En particular, esta característica, facilita el reciclaje de viejos desarrolladores acostumbrados a tecnologías obsoletas y la incorporación de nuevos desarrolladores sin experiencia.

El Cliente gana mucho en libertad porque, en cualquier momento puede resolver el cambio de la plataforma de ejecución y transformar con Genexus sus aplicaciones a la nueva.

Nuevas tecnologías pueden aparecer en el medio de un gran proyecto y utilizárselas de inmediato sin traumas.

Genexus integra automáticamente los diferentes elementos de una aplicación, asegura su permanente consistencia y mantiene una documentación completa, activa y siempre actualizada.

La corrección de los sistemas se verifica probando las especificaciones vía prototipación viva, completa, oportuna.

Todo esto es muy importante y, la mayor parte de estas características son únicas. ¿Dónde está, entonces, el problema? No existe ningún problema pero existe sí una restricción: Genexus debe ser utilizado siempre por desarrolladores profesionales.

¿Es defendible en el medio plazo la idea de que todas las aplicaciones deban ser construidas por desarrolladores profesionales?

Nuestra opinión es que no: ¡la necesidad de utilizar únicamente desarrolladores profesionales es consecuencia de las limitaciones de la tecnología!

Levantando esas limitaciones podemos pensar en un aporte mucho mayor de usuarios de la información, de usuarios que conocen el negocio – o el asunto que sea - y que no conocen (ni pretenden conocer) los detalles de bajo nivel que hoy son necesarios para construir un sistema:

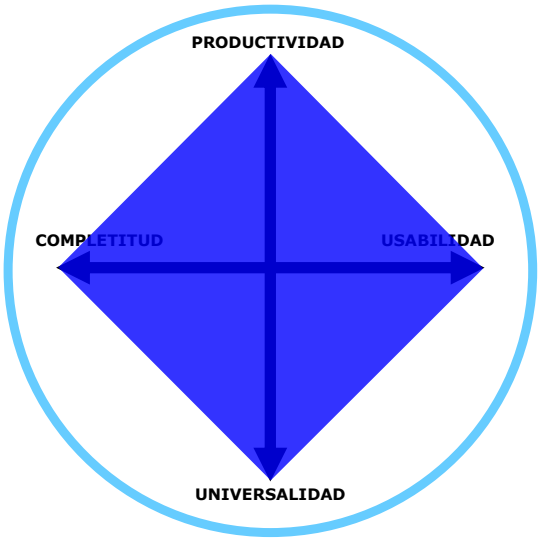
Cada vez será más importante el conocimiento que se tenga sobre el problema que se quiere resolver y menos el de la tecnología necesaria para resolverlo. Cada vez los sistemas son más diversificados: en poco tiempo los sistemas comerciales basados en contabilidad, compras, ventas, nómina, stock o bien ERPs, CRMs, etc. constituirán una proporción menor de las aplicaciones que los usuarios (los muchos cientos de millones de usuarios) necesitan ¿Dónde está el conocimiento para construir las nuevas aplicaciones?: pues mucho más en esos usuarios que en los desarrolladores profesionales.

Para ello Genexus deberá evolucionar mucho haciéndose más "usable": mucho más fácil de utilizar por usuarios que necesitan resolver un problema (que conocen bien) y no son desarrolladores profesionales.

En particular, es necesario hacer a Genexus más amigable, disminuyendo todo lo posible el nivel de abstracción necesario para utilizarlo, ocultando la complejidad y, en particular, aumentando su componente declarativa y adicionando, probablemente, componentes gráficas intuitivas y fáciles de usar, de manera de minimizar la necesidad de utilizar su componente procedural.

¡Ésta es la gran tarea de los próximos años!

EL FUTURO: AUMENTO RADICAL DE LA USABILIDAD



[1] **Breogán Gonda y Juan Nicolás Jodal** son Ingenieros en Computación, formados por la Facultad de Ingeniería de la Universidad de la República, Uruguay.

Han desarrollado una amplia actividad internacional de consultoría, docencia e investigación.

Sus áreas principales de investigación son Bases de Datos Relacionales, Inteligencia Artificial, tratamiento automático del conocimiento y desarrollo automático de Aplicaciones.

Han recibido el Premio Nacional de Ingeniería 1995, otorgado por la Academia Nacional de Ingeniería del Uruguay.

Son socios fundadores y, respectivamente, Presidente y Vicepresidente de Artech, empresa que desarrolla y comercializa en todo el mundo el producto Genexus.

---

[2] **Genexus es un sistema basado en conocimiento** y, en particular, es un sistema que se basa en una muy buena capacidad de tratamiento automático del conocimiento de los sistemas de negocios.

Con una rigurosa fundación matemática, Genexus, partiendo de visiones de usuarios, captura todo el conocimiento contenido en ellas y lo sistematiza en una Base de Conocimiento.

La base de conocimiento de Genexus tiene una gran capacidad de inferencia lógica: en cualquier momento es capaz de proporcionar cualquier conocimiento que se ha almacenado en ella o que puede inferirse lógicamente de aquellos almacenados en ella. Basado en esta capacidad de inferencia es capaz de proyectar, generar y mantener, en forma 100% automática, la base de datos y los programas necesarios para satisfacer todas las visiones de usuarios conocidas en un determinado momento.

---

[3] **Procedural / Procedimental**: La palabra española correcta es "Procedimental", sin embargo en la informática no se la usa sino que se la sustituye por Procedural, que es la palabra inglesa que se utiliza en todo el mundo. En este trabajo utilizaremos siempre "Procedural"