Artículo
Robert Cemper · 1 feb, 2022 Lectura de 4 min

El futuro de ObjectScript

Si desarrollas en IRIS, te enfrentas a dos fenómenos principales:

- un motor de almacenamiento de datos increíblemente rápido y con un excelente diseño
- un lenguaje para trabajar en este motor de almacenamiento, llamado ObjectScript

Motor de almacenamiento

Se trata de un concepto de almacenamiento organizado jerárquicamente y se consideró "anticuado" cuando "relacional" era la palabra de moda del momento.
Sin embargo, si usas sus fortalezas, con su sencilla y eficiente construcción, supera a todos sus competidores "gigantes".
Y esto sigue siendo cierto desde el día 1. Me sorprende cómo esos competidores copiaron a lo largo del tiempo varias funciones de este motor de almacenamiento, confirmando indirectamente la calidad del concepto básico.

ObjectScript

Mirando el lenguaje, es bastante fácil separar el subconjunto perfectamente delimitado de elementos del lenguaje que manipulan y navegan por el motor de almacenamiento, que ha visto solo algunas extensiones y ajustes a lo largo de su vida. Yo llamo a esto el NÚCLEO.
Sus compañeros son la **Navegación** (en su mayoría similar a otros lenguajes) y la **Estética** (todo para manipular el contenido de los datos).

La navegación es un requisito estructural inevitable, pero ha tenido varias mejoras, principalmente para comodidad de los programadores. No es obligatorio, pero es similar a los lenguajes más nuevos.

La estética es el campo de más rápido crecimiento y hasta hoy sus propuestas me sorprenden de vez en cuando. Su existencia viene de cuando el motor de almacenamiento también formaba parte de un sistema operativo (y nada más) sobre el HW.

Historia

Desde el pasado, los desarrolladores solían escribir aplicaciones con ObjectScript. Incluso InterSystems hizo esto a gran escala (Interoperability, es decir, Ensemble), Analytics (es decir, DeepSee, Health*...). ObjectScript era/es el "puede con todo, hace de todo".

Pero ObjectScript era más bien un "llanero solitario". El intento de tener algo más popular en el mercado fue BASIC (ya era un "caballo muerto" en ese momento). La oportunidad de saltar a la ola de JavaScript se perdió hace unos 15 años.

La disponibilidad de una amplia gama de adaptadores de lenguajes pudo ocultar la dimensión del problema durante algún tiempo, pero nunca alcanzó la fuerza que el principal competidor mostró cuando PL/SQL fue congelado y reemplazado por Java.

Hoy en día

Veo que el número de desarrolladores formados en ObjectScript se reduce con el tiempo por puros efectos

demoscópicos. Encontrar y educar desarrolladores que quieran escribir en ObjectScript es un ejercicio duro, que experimenté personalmente varias veces. E incluso si son brillantes y muy hábiles, no es garantía de que se queden con uno. La demanda sigue siendo espectacular.

Delante de este trasfondo, veo la llegada de Python embebido como lenguaje completo al mismo nivel que ObjectScript como un gran paso adelante con IRIS.
La navegación y la estética están cubiertas y añadir las funcionalidades NUCLEARES no requiere un aprendizaje dramático. De modo que vemos un mercado mucho más grande de recursos de desarrollo especializados que acaban con la limitación que en ese aspecto teníamos con ObjectScript .

Futuro

- Con Python, veo mucha energía nueva entrando en el mundo de IRIS. Después de todo, es un paso que algunos competidores dieron hace bastante tiempo y el inconveniente de "nunca visto en ningún otro lugar" queda superado.
- El paradigma: "Lo programo porque es posible"; se romperá. Nos libera de una serie de ruedas re-inventadas.
- ObjectScript seguirá funcionando en paralelo y tendrá un papel importante en todas las actividades cercanas al NÚCLEO de almacenamiento. Nadie (seguramente) lo usará para lógica empresarial o (espero que no) para replicar interfaces que ya están disponibles fuera a mayor escala.
- ObjectScript se reducirá a una dimensión que vemos hoy para C, C ++ o para código en cualquier lenguaje ensamblador. Las aplicaciones existentes no se verán afectadas. La forma en que se integró Python parece ser una garantía para una migración sin problemas y sin presión de tiempo.
- Los desarrolladores de ObjectScript de hoy no perderán sus trabajos. Hay suficientes tareas complejas en torno al diseño y mantenimiento de bases de datos. Pero podrían deshacerse de tareas poco atractivas (en mi opinión): estilos CSS, colorear páginas web,. . . .
- Sin olvidar el exigente desafío de leer y comprender el código existente e interpretar y comunicar su contenido y lógica. Los "Golf Code" (1) pueden ser divertidos, pero se trata de una lógica de negocio muy seria en muchas aplicaciones escritas tradicionales antiguas e incluso en algunas utilidades en "%SYS".
- Veo a los desarrolladores de hoy como los sacerdotes de ObjectScript, del mismo modo que los sacerdotes egipcios que comprenden los jeroglíficos y leen "El libro de los muertos" y celebran sus milagros.

15 de julio de 2021

#Concurso #Early Access Program (EAP) #Lenguajes #ObjectScript #Python #InterSystems IRIS

URL de fuente:https://es.community.intersystems.com/post/el-futuro-de-objectscript