arquitectónicas tomadas cuando se construyó la aplicación heredada. El propósito es extender los patrones a las particularidades de cada cliente para ampliar la cobertura de descubrimiento de código automático.

- Gran parte del código PL/SQL implementa lógica de interfaz de usuario que es muy particular del ambiente gráfico de Oracle Forms (por ejemplo, navegar los registros de la base de datos con accesos directos de teclado) y, por lo tanto, es casi imposible migrarla a otras tecnologías modernas como Web.

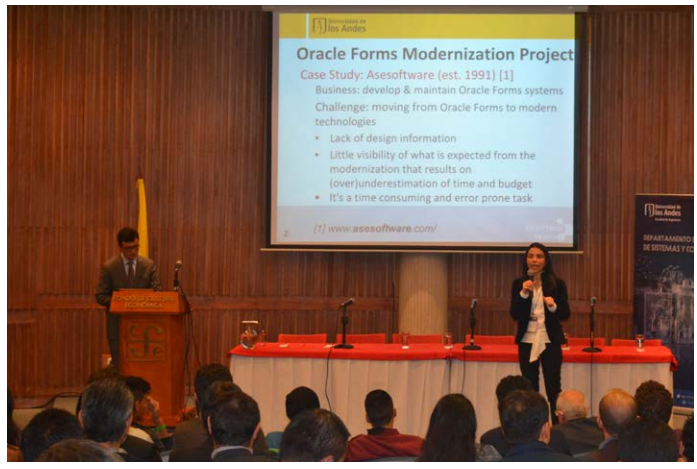


Foto: Natalia Fernanda Madrid Vidales

Kelly Garcés, profesora del Departamento de Ingeniería de Sistemas y Computación, y Cristo Rodríguez, durante su presentación del trabajo que el DISC desarrolló en alianza con Asesoftware.

Dos conclusiones

El modelo intermedio impide los altos costos de retrabajo. No solo se transforman las funcionalidades Crud, sino también la lógica escrita en PL/SQL. El código destino es claro y entendible, lo cual facilitará su mantenimiento. Adicionalmente, en PL/SQL la traducción del código no es, por ejemplo, de instrucción a instrucción, sino que cada patrón catalogado tiene un propósito funcional y es implementado siguiendo las mejores prácticas en la arquitectura destino.

La segunda conclusión es que, a través de modelos, se facilita independizar la solución de las tecnologías origen y destino. Por ejemplo, si Asesoftware decide migrar Oracle Forms hacia .NET, el proceso de modernización es el mismo, sólo bastaría construir una nueva transformación para generar el código en esa tecnología específica. ■

Calidad y eficiencia, beneficios de los generadores

Con el fin desarrollar proyectos de *software* con alto nivel de calidad y de productividad, Heinsohn escogió una solución basada en MDE (*Model Driven Engineering*). Por cada 4,6 líneas de código desarrolladas a mano, se generan automáticamente hasta 8794.

La reutilización de componentes existentes al momento de construir nuevas aplicaciones se ha constituido en una ventaja competitiva para Heinsohn, empresa de desarrollo de *software*. Así lo afirmó Catalina Acero, socia y directora del área de Soluciones de Ingeniería, quien presentó la experiencia, los resultados y los desafíos del uso de MDE en Heinsohn, durante su conferencia "Ingeniería dirigida por modelos en el campo de batalla".

Relató cómo enfrentaron el reto de semiautomatizar la construcción de componentes particulares de un negocio que se integran con los componentes transversales, tales como módulo de seguridad, notificaciones de correo, manejo de históricos. Además, ¿cómo minimizar el tiempo que los ingenieros invierten en entender los microdetalles de las nuevas tecnologías y en solucionar errores de integración, sean de tecnología o por incomprensión del negocio?

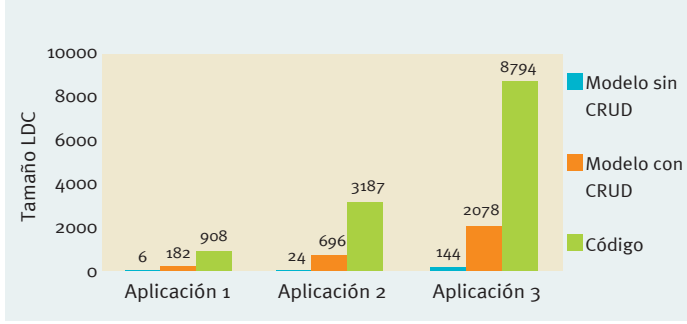
Foto: Natalia Fernanda Madrid Vidales

Catalina Acero es directora del área de Soluciones de Ingeniería de Heinsohn Business Technology, una compañía con 700 colaboradores y presencia en Colombia, Estados Unidos, El Salvador y Ecuador.

Para aumentar la productividad, Heinsohn desarrolló un proyecto en el que se aplicaron los principios de MDE. En particular, se propuso un lenguaje de modelaje llamado ISML (*Interface Specification Meta-Language*), que aporta facilidad de abstracción, es comprensible, permite la automatización, es flexible, soporta los elementos adecuados para modelar proyectos de *software*, controla versiones y es fácil de adoptar. Además, se puede integrar con *frameworks* y componentes existentes.

Luego se escogió una arquitectura por capas. En la de presentación, las páginas invocan los controladores, que a su vez acceden a la capa de servicios y lógica. Por su parte, la lógica utiliza funcionalidades de persistencia, para guardar, consultar, actualizar o remover un objeto. Para generar los artefactos de las diferentes capas, el ingeniero solo debe especificar las entidades de negocio y las páginas usando ISML. Éste ofrece constructores en lenguaje natural, independientes de las tecnologías específicas abordadas en la arquitectura de referencia.

En la segunda etapa se implementaron la gramática de ISML usando XText y el generador de las diferentes capas para tecnología Java. Las tecnologías usadas en cada capa siguen: presentación en JSF (*Java Server Faces*) y JavaFX, servicios en REST, lógica y persistencia en JEE. En fases próximas se diseñarán nuevos generadores del *stack* completo para .NET y capa de presentación en Angular. "Esto no es muy difícil, ya que la semántica de ISML es clara y se tienen los generadores base. Por ejemplo, el de JavaFX lo hizo un solo ingeniero en mes y medio", explicó Catalina Acero.



Fuente: Heinsohn

El gráfico muestra la cantidad de líneas de código escritas trabajando con y sin modelos en distintas aplicaciones.

Este proyecto ha sido probado con pilotos en aplicaciones muy pequeñas. De acuerdo con las estadísticas de Heinsohn, para desarrollar el Crud de una entidad de negocio en las tecnologías descritas, se tienen que escribir 8794 líneas de código. En contraste, trabajando con modelos, solo se tienen que escribir 4,6 líneas. El código generado es de alta calidad, ya que la arquitectura de referencia aplica buenas prácticas como, por ejemplo, asegurar un correcto uso del manejo de las transacciones, uno de los errores más habituales. Además, en presentación no se verán los llamados 'errores groseros', porque se surten las validaciones básicas. "Por supuesto es inevitable la participación de los ingenieros, posterior a la generación, ya que ellos son los encargados de la implementación de lo grueso del negocio, de la lógica, pues, por muy bueno que sea el lenguaje de modelado, ese aspecto no se refleja. El reto es cómo madurar esta situación, aunque no sea 100 % automática", señaló.

Todo esto, afirmó Catalina Acero, impacta el paradigma de los roles y actividades de los ingenieros. "Si pudiéramos masificar el uso de esta aproximación, podrían centrarse más en el modelo de negocio porque los detalles específicos de la tecnología se manejan desde los generadores. También podrían percatarse de inconsistencias en los requerimientos y aportarle más al cliente".

Los desafíos para implantar MDE incluyen la curva de aprendizaje, así como pensar diferente, en otro nivel de abstracción. También el cambio en el paradigma de programación, no solo entre los ingenieros sino también en la industria. Un reto más se presenta por el hecho de que los generadores evolucionan rápidamente, tal como la tecnología y las arquitecturas. Es decir, se deben seguir manteniendo. ■

Modelaje, eficiencia para un **equipo de desarrollo pequeño**

Crear aplicaciones en una compañía ajena a la tecnología fue el reto de este proyecto que ha encontrado en MDE una herramienta para mejorar la productividad. Aunque el proceso continúa, ya dio origen a una *startup* emergida del Grupo GHL.

Versatilidad en la generación de código, facilidad en la corrección de errores y problemas en la capa de presentación son los principales resultados que mostró Andrés Yie, director de tecnología del GHL. Hace tres años emprendió este proyecto para producir