INSTITUTO KHIPU

Semestre III

Sesión 21

PROGRAMA DE ESTUDIOS

DESARROLLO DE SISTEMAS DE INFORMACIÓN

ANÁLISIS Y DISEÑO DE SISTEMAS

Tema:

PUDS - GENERACIÓN DE CÓDIGO A PARTIR DE DIAGRAMAS DE SECUENCIA.

PUDS - GENERACIÓN DE CÓDIGO A PARTIR DE DIAGRAMAS DE SECUENCIA.

La **generación de código fuente con herramientas CASE** (Computer-Aided Software Engineering) es un enfoque que permite automatizar parte del desarrollo de software al convertir diagramas y modelos UML en código fuente. Estas herramientas son ampliamente utilizadas en proyectos grandes y complejos, ya que mejoran la productividad y reducen errores humanos.

Flujo General de la Generación de Código con Herramientas CASE

1. **Diseño UML**
   - Se crean diagramas UML, como el diagrama de clases, diagramas de secuencia o diagramas de estado.
   - Estos diagramas actúan como base para definir la arquitectura y comportamiento del sistema.
2. **Configuración de la Herramienta CASE**
   - La herramienta debe configurarse para el lenguaje de programación deseado (Java, Python, C#, etc.).
   - Se establecen preferencias para el formato del código, convenciones de nomenclatura y visibilidad de atributos/métodos.
3. **Generación Automática del Código**
   - A partir de los modelos UML, la herramienta genera código fuente básico con estructuras como clases, atributos, métodos y relaciones.
   - En algunos casos, puede incluir comentarios generados automáticamente que describen cada elemento del modelo.
4. **Personalización y Desarrollo Adicional**
   - El desarrollador ajusta el código generado para implementar la lógica de negocio y completar detalles específicos.

Herramientas CASE para Generación de Código

1. **Enterprise Architect**
   - Generación de código desde diagramas UML, como clases y estado.
   - Soporte para múltiples lenguajes (Java, C#, C++, Python, PHP).
   - Admite ingeniería inversa (creación de diagramas desde código existente).
2. **Visual Paradigm**
   - Generación de código fuente para lenguajes como Java, C#, PHP, Python.
   - Compatible con bases de datos, permitiendo generar scripts SQL desde modelos ERD (Diagrama Entidad-Relación).
3. **StarUML**

- Herramienta ligera para diseño UML con soporte de generación de código.
- Compatible con extensiones para Java, Python, Ruby, y otros lenguajes.

4. **IBM Rational Rose**
   - Soporte completo para generación de código y sincronización con el diseño UML.
   - Ideal para proyectos en C++, Java y Visual Basic.
5. **ArgoUML**
   - Open-source con funcionalidad básica de generación de código para Java.
   - Útil para proyectos educativos y pequeños.
6. **Astah Professional**
   - Permite modelado UML avanzado y generación de código en varios lenguajes.
   - Soporta exportación para IDEs como Eclipse y NetBeans.

Ejemplo Práctico de Generación de Código con Enterprise Architect

Diagrama UML de Clase

Clase **Producto**

Atributos - nombre: String

- precio: Double

Métodos + calcularImpuesto(): Double

Código Generado (Java)

java
Copiar código

```
public class Producto {
    private String nombre;
    private Double precio;

    public Double calcularImpuesto() {
        // Implementar cálculo de impuesto
        return null;
    }
}
```

Ventajas de la Generación de Código con Herramientas CASE

1. **Ahorro de Tiempo:** Automatiza la creación de estructuras repetitivas.
2. **Estandarización:** Asegura que el código siga las especificaciones del diseño UML.
3. **Consistencia:** Mantiene sincronía entre el diseño y el código fuente.
4. **Ingeniería Inversa:** Permite documentar proyectos existentes al crear diagramas desde el código.

Limitaciones

1. **Código Base:** El código generado suele ser un punto de partida; aún requiere ajustes manuales para agregar lógica específica.
2. **Complejidad Inicial:** La configuración y aprendizaje de las herramientas puede ser complejo para equipos nuevos.
3. **Dependencia de Herramientas:** No todas las herramientas admiten lenguajes de programación o arquitecturas modernas.

Recomendaciones para Usar Herramientas CASE

1. **Inicia con el Diseño:** Modela cuidadosamente tus diagramas UML antes de generar código.
2. **Selecciona una Herramienta Apropiada:** Elige una herramienta compatible con tu lenguaje y entorno de desarrollo.
3. **Sincronización Bidireccional:** Usa herramientas que admitan ingeniería directa e inversa para mantener diseño y código sincronizados.
4. **Capacitación del Equipo:** Asegúrate de que el equipo esté capacitado en el uso de la herramienta CASE elegida.

FUENTE:

- https://www.uml.org/
- https://www.codingdojo.la/2023/06/16/guia-del-ciclo-de-vida-del-desarrollo-de-software/
- https://aws.amazon.com/es/what-is/sdlc/
- BURCH, John; GRUDNISKY, Gary. "Diseño de Sistemas de Información", Grupo Noriega editores.
- SENN, James A. "Análisis y diseño de sistemas de información", 2da. ed., McGraw-Hill.


INSTITUTO
KHIPU