Programación generativa

Ulises Juárez Martínez
ujuarez@itorizaba.edu.mx

Instituto Tecnológico de Orizaba

15 de octubre de 2010

Agenda

¿De qué se trata?

La idea central

- Principios y técnicas de ingeniería de software
- Basada en la configuración automática
- Basada en fragmentos
- Todo con modelos de componentes

El problema

Tecnologías de componentes y objetos

- Poco soporte para el análisis y diseño con miras a la reutilización
- Técnicas no efectivas para las variaciones de componentes
- Pérdidas del conocimiento del diseño debido a la diferencia semántica entre anstracciones del dominio y los lenguajes de programación
- Fallas de desempeño a tiempo de ejecución para un diseño flexible y claro

El propósito

- Desarrollar software para producir:
  - Alta reutilización
  - Alta adaptación
  - Alta intención: solo el QUÉ, no el CÓMO
- Sin comprometer el desempeño a tiempo de ejecución
- Sin comprometer los recursos computacionales del software que se produce

Áreas que integran la PG

- Ingeniería de dominios (familias de sistemas)
- Análisis y diseño OO y OA (sistemas únicos)
- Modelación de características (reutilización en forma abstracta, sin un modelo de implementación)
- Generadores de configuraciones basadas en mixins sin configuracion automática
- Repositorios de configuración y plantillas de metaprogramación
- Programación intencional

Los 4 grandes problemas

Áreas para:

- Reutilización
- Adaptación
- Administración de complejidad
- Desempeño (en espacio y tiempo de ejecución)

Necesidades específicas

- Modelos reutilizables para clases de sistemas
- Desarrollo sistemático (no ad-hoc)
- Mecanismos para proveer implementaciones alternativas (solución parcial con GoF)

Paradigmas relacionados

- Programación genérica: reutilización mediante parametrización
- Lenguajes de dominio específico: incrementan el nivel de abstracción de un problema
- Programación orientada a aspectos

Paradigmas relacionados

- Programación genérica: reutilización mediante parametrización
- Lenguajes de dominio específico: incrementan el nivel de abstracción de un problema
- Programación orientada a aspectos

PG

- Toma ideas de estos paradigmas
- Agrega configuración automática, técnicas genéricas y mayor parametrización

Propósito

- Toman un rol activo en la generación de código
- Proveen abstracciones que son auto-optimizables
- Las abstracciones generan componentes, algoritmos especializados, código optimizado, autoconfigurables y autoafinables para una máquina particular y verifican código fiente para correctitud
- Son agentes que al interactuar con otros producen componentes concretos

Compilación extensible y sistemas de procesamiento de meta-nivel

- Manipulación directa de las construcciones del lenguaje
- Transformaciones de AST
- Generación de código fuente a tiempo de compilación

Especialización de programas

- Evaluación parcial: generación de extensiones
- Generadores de componentes
- Variación a tiempo de ejecución

Lenguajes muti-nivel

- Manejan código estático: evaluado a tiempo de compilación
- Manejan código dinámico: compilado y ejecutado posteriormente a tiempo de ejecución
- Permiten la escritura de generadores de programas

Lenguajes muti-nivel

- Manejan código estático: evaluado a tiempo de compilación
- Manejan código dinámico: compilado y ejecutado posteriormente a tiempo de ejecución
- Permiten la escritura de generadores de programas

Técnica de la plantilla de meta-programas

- Explota el mecanismo de plantilla para realizar cálculos arbitrarios a tiempo de compilación
- Permite la generación de código en forma selectiva al ejecutarse el meta-programa

Generación de código a tiempo de ejecución

- Bibliotecas que generan código optimizado a tiempo de ejecución
- Se desarrollan optimizaciones que dependen de la información disponible a tiempo de ejecución
- También se utiliza la reflexión dinámica

Antecedentes

Comunidades que han estudiado generadores

- Ingeniería de software basada en conocimiento o síntesis de programas
- Reutilización de software
- Especificación formal

Anteccedentes

Capacidades de los generadores

- Separan el espacio de modelación del espacio de implementación
- Los espacios pueden tener estructuras diferentes
- Las implementaciones se calculan a tiempo de implementación por optimizaciones del dominio específico y substituyendo, mezclando, agregando y removiendo componentes

Sobre los generadores

Objetivo principal

- Producir sistemas de software desde especificaciones de alto nivel.

Técnicas utilizadas durante la generación

- Composición: generadores por composición (n componentes)
- Transformación: sistemas de transformación (modificaciones)
- Combinación de las dos anteriores

Componente generativo

- Acepta una descripción abstracta de una instancia concreta
- Genera la instancia de acuerdo a la descripción
- Acepta parámetros abstractos
- Ejemplo de parámetro abstracto: una especificación de desempeño

Transformación

El concepto

- Modificación automatizada y sintácticamente correcta de un programa
- En otras palabras, describe una modificación genérica de un programa
- Es una instancia específica de la transformación
- Categoría más conocida de transformación: reglas de re-escritura
- Ejemplo de regla de re-escritura: substitución de la división por cero

Tipos de transformaciones

Refinamiento

- Agrega detalle a la implementación
- Ejemplos: descomposición, opción de representación, opción de algoritmo, especializacion (evaluación parcial), concretización

Tipos de transformaciones

Refinamiento

- Agrega detalle a la implementación
- Ejemplos: descomposición, opción de representación, opción de algoritmo, especializacion (evaluación parcial), concretización

Optimizaciones

- Mejoran las características de desempeño de un programa mediante cambios estructurales en el código (ejecución, rapidez, tiempo de respuesta, consumo de memoria, etc.)
- Ejemplos: evaluación parcial, diferencia finita, inlining, fusión de ciclos, eliminación de código muerto

Tipos de transformaciones

Recomposición

- Reorganizan código al nivel de diseño
- Ejemplos: Generalización, simplificación, introducción de nuevos puntos de variación

Tipos de transformaciones

Recomposición

- Reorganizan código al nivel de diseño
- Ejemplos: Generalización, simplificación, introducción de nuevos puntos de variación

Otras transformaciones

- Por calendarización (ubicaciones específicas en AST)

Enfoques generativos

Draco

- Ingeniería de dominios basada en lenguajes de dominio específico y tecnología de transformación
- Organiza conocimiento de costrucción de software dentro de un número de dominios relacionados

Enfoques generativos

Draco

- Ingeniería de dominios basada en lenguajes de dominio específico y tecnología de transformación
- Organiza conocimiento de costrucción de software dentro de un número de dominios relacionados

GenVoca

- Construcción de generadores de sistemas de software basadas en capas de abstracción OO
- Propaga tipos hacia capas superios e inferiores
- Algunos conceptos de GenVoca aparecen en ActiceX y JavaBeans

Enfoques generativos

Programación intencional

- Lenguaje de programación extendible (Microsoft)
- Se enfoca al desarrollo de lenguajes de dominio específico y generadores para cualquier arquitectura
- Una intención es una abstracción o característica del lenguaje

Enfoques generativos

Programación intencional

- Lenguaje de programación extendible (Microsoft)
- Se enfoca al desarrollo de lenguajes de dominio específico y generadores para cualquier arquitectura
- Una intención es una abstracción o característica del lenguaje

Programación naturalística

- Análisis de AOP vs. lenguajes naturales
- Los humanos piensan y describen en términos crosscutting

Conclusiones

Arquitectura de software

- Ingeniería de dominios
- LPS

Conclusiones

Arquitectura de software

- Ingeniería de dominios
- LPS

Componentes

- Adaptación en tiempo de ejecución
- Descripciones abstractas - AOP
- Arquitecturas reconfigurables

Conclusiones

Arquitectura de software

- Ingeniería de dominios
- LPS

Componentes

- Adaptación en tiempo de ejecución
- Descripciones abstractas - AOP
- Arquitecturas reconfigurables

Programación generativa

- Componentes altamente configurables
- Generación de código a tiempo de ejecución

¿Preguntas?

ujuarez71@gmail.com
http://computacion.cs.cinvestav.mx/~ujuarez/