Domain-Specific Declarative Languages

Johannes Gehrke
Cornell University
(Currently on sabbatical at FAST, A Microsoft Subsidiary)

http://www.cs.cornell.edu/johannes
Declarative Domain-Specific Languages

- **Declarative**
  - Nice algebraic properties for ease of reasoning ...
    - for the user / query writer
    - for system designer / implementer
  - (Note that the language could *look* imperative.)

- **Domain-specific**
  - Language should be appropriate for domain
  - Can’t have “one size fits all”
Computer Games

- Unique challenges
  - Virtual environments
  - High degree of interactivity

- $17B in sales in 2007
  - Rivals movie industry
Data-Driven Game Design

- Game design brings together many disciplines
  - Art, music, computer science, etc...

- Today’s games are designed *data-driven*
  - Game content is separated from game code

- Examples:
  - Art and music are kept in separate, industry-standard file formats
  - Character data is kept in XML
  - Character behavior is specified through scripts
Advantages of Data-Driven Design

- Engine is reusable.
  - Able to recoup R&D costs over several games.
  - Possible to license engine to other companies.
- Can extend the life span of the game
  - Adjust game balance
- Players can change the character of the game
  - Modder communities develop around the game.
    - Half-life → Counter Strike
  - (Air Force uses modded StarCraft in officer training.)
Modding: Simulation Games

- Non-Player Characters (NPCs): Characters not directly controlled by the player.
  - Controlled either by built-in or by player-designed scripts.

- "Doll House" games (e.g., The Sims)
  - NPCs have needs and desires.
  - Objects can satisfy needs and desires.
  - Player controls the game via object placement.

- Real-Time Strategy games
  - Troops move and fight in real time.
  - Player controls the game via a limited number of commands.
  - Player multitasks between large number of units.
Simulation Games: NPCs

- Games need complex NPC behavior.
  - Example:
    - Units are afraid of skeletons
    - Likelihood of staying inverse proportional to number of skeletons around

- Simple example: morale
  - Units afraid of skeletons
  - Morale proportional to number of skeletons seen
  - Processing cost:
    - \(O(n)\) to count skeletons
    - \(O(n^2)\) to process all units

Time per tick:
- 3 units
- 2 units
- 1 unit
Expressiveness vs. Performance

- **Expressiveness**: the range of behavior scriptable by modders

- As # of NPCs increases, expressiveness decreases
  - *Neverwinter Nights 2*
    - Each NPC fully scriptable
  - *WarCraft III*
    - Script armies, not NPCs
    - Little NPC coordination
  - *Midieval: Total War*
    - No individual scripting at all
Expressiveness vs. Performance

- **Expressiveness:**
  the range of behavior scriptable by modders

- As # of NPCs increases, expressiveness decreases
  - *Neverwinter Nights 2*
    - Each NPC fully scriptable
  - *WarCraft III*
    - Script armies, not NPCs
    - Little NPC coordination
  - *Midieval: Total War*
    - No individual scripting at all
Expressiveness vs. Performance

- **Expressiveness**: the range of behavior scriptable by modders

- As # of NPCs increases, expressiveness decreases
  - *Neverwinter Nights 2*
    - Each NPC fully scriptable
  - *WarCraft III*
    - Script armies, not NPCs
    - Little NPC coordination
  - *Midieval: Total War*
    - No individual scripting at all
What We Have Done

- Developed domain-specific **imperative** language
  - Game designers find SQL difficult
  - Cannot program at the individual level

```plaintext
main : Unit {
  // Compute # skeletons, group center
  accum c with sum, sx with sum,
  sy with sum over u from UNIT {
    if (isEnemySkeleton(u) &&
      dist(me, u) < range) {
      c <- 1; sx <- u.x; sy <- u.y;
    }
  }
  in {
    // If too many skeletons
    if (c > morale) {
      let (norm=(x-sx/c)*(x-sx/c) +
        (y-sy/c)*(y-sy/c)) in {
        // Run in opposite direction
        vx <- (x-sx/c)/norm;
        vy <- (y-sy/c)/norm;
      }
    }
  }
  ...
```
What We Have Done

- Developed domain-specific *imperative* language
- Compiles down to *set-at-a-time* Bag Algebra

```haskell
main : Unit {
    // Compute # skeletons, group center
    accum c with sum, sx with sum,
    sy with sum over u from UNIT {
        if (isEnemySkeleton(u) &&
            dist(me, u) < range) {
            c <- 1; sx <- u.x; sy <- u.y;
        }
    } in {
        // If too many skeletons
        if (c > morale) {
            let (norm = (x-sx/c)*(x-sx/c) +
                (y-sy/c)*(y-sy/c)) in {
                // Run in opposite direction
                vx <- (x-sx/c)/norm;
                vy <- (y-sy/c)/norm;
            }
        }
    }...
```
What We Have Done

- Developed domain-specific *imperative* language
- Compiles down to *set-at-a-time Bag Algebra*
- Built prototype of a game engine
- Integrated crowd simulations into SGL
- Developed novel transactional models for virtual worlds

- Opened up many more problems
  - Main-memory data management with decision-support-style *update* workloads
    - Query optimization, query processing, indexing
  - Steering
  - Collaborative motion planning
Recipe Applied to Other Domains

- Complex Event Processing
- Personalization of three-tier applications
  - Build full CMT in 15 minutes
  - Personalize the system to add double-blind reviewing for your conference in 5 minutes
- Expressive ad auctions
  - Submit bidding programs to Google according to an ROI strategy
Declarative Domain-Specific Languages

- **Declarative**
  - Nice algebraic properties for ease of reasoning ...
    - for the user / query writer
    - for system designer / implementer
  - (Note that the language could *look* imperative.)

- **Domain-specific**
  - Language should be appropriate for domain
  - Can’t have “one size fits all”
Summary and Discussion

- Declarative, domain-specific languages

There is much more:

- Multi-core
- Creativity
  - Attractor for the next generation of students?
  - Gate to a new way of enabling and understanding creativity?
Let’s Play!

http://www.cs.cornell.edu/johannes

Thank you: National Science Foundation, Air Force Office of Scientific Research, Microsoft, Yahoo!

DB Summit, 5/29/2008