Week 2: Domain-Specific Languages

Nate Bachmeier

TIM-7241:Constructive Research Design

May 30, 2021

Northcentral University

# Domain-Specific Languages

Domain-Specific Languages (DSL) enable business professions to describe their rules and policies succinctly. This capability reduces the barrier to automation, promotes cross-disciplinary collaboration, and modulizes complex systems. For instance, financial trading applications contain an execution engine and various investment strategies. These different components require unique professional expertise, making it challenging to have one person build both. Instead, the engineering team can standardize the problem’s shared vocabulary into external files that drive the trading engine. Afterward, domain experts can contribute to those files (scripts) and audit the accuracy of the trading algorithm.

Countless scenarios can leverage DSL to automate specific business domains. For example, Cacciagrano & Culmone (2020) built IRON to simplify programming embedded systems through an Event-Condition-Action (ECA) model.

|  |  |  |
| --- | --- | --- |
|  | IRON (2020) | Air Traffic (2021) |
| Business Problem | Integrating event-driven embedded system | Validating air traffic control protocols |
| Why DSL | Need abstraction layer for heterogeneous topologies | Test cases have 235 parameters on average (e.g., 4-D space) |
| Artifacts | * LUA Interpreter * Verifiable call graph * ECA design pattern | * Compiles to AIS/ADS-B * Adapter for FDI-Test Framework |
| Measuring Effectiveness | * Abstract analysis (set theory) | * 11196 unique test cases |
| Language Extensions | * ECA is standard within cloud computing | * Naval control protocols |
| Measuring Extensions | * Usefulness wrt, e.g., Lambda | * See: Kontopoulos (2018) * TIM 7140 / W. 6 |