Executable Domain Model

Bo Wang
Outline

• Domain Models
• Domain-driven and ICSM
• Current Progress & Infrastructures
• Future Directions
Domain Models

• Operational System Context

• **Application Domain Model**
  – General domain terminologies, basic notions of the application domain
Domain Models

- Domain Model Evolution

Development Lifecycles
Domain-driven and ICSM

• 3 phases focusing on domain modeling

<table>
<thead>
<tr>
<th>Phase</th>
<th>Inception</th>
<th>Foundations₁</th>
<th>Development₁ Foundations₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity</td>
<td>Initial scoping</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Concept definition</td>
<td>Operational concept</td>
<td></td>
<td></td>
</tr>
<tr>
<td>System architecture</td>
<td>Increment design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Increment foundation rebaseline</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task</th>
<th>Initial Domain model Identification</th>
<th>Domain model prototype &amp; analysis</th>
<th>Domain model update &amp; integration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Detailed design</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
• Code generation built upon

**Problem Domain Abstraction**

**Easy Access**

**Flexible Schema**

- Domain model identification
- Domain model prototype & analysis
- Domain model update & integration
- Detailed design

---

**Code generation built upon**

- Initial Domain model identification
- Domain model prototype & analysis
- Domain model update & integration
- Detailed design

---

**Problem Domain Abstraction**

- RESTful API
- Server Side
- Business Side
- API Server
- Model Controllers
- Adapter
- Code Patterns
- Extraction
- Generation
- Source Code
- CRUD functions
- DB schema
- NoSQL
- riak
- Cassandra
- redis
- CouchDB
- MongoDB

---

**Easy Access**

- CRUDBuilding RESTful API
- Extraction
- Generation
- Source Code

---

**Flexible Schema**

- Domain model identification
- Domain model prototype & analysis
- Domain model update & integration
- Detailed design

---

**Business Side**

- Model Controllers
- Adapter
- RESTful API
- CRUDBuilding RESTful API
- Extraction
- Generation
- Source Code

---

**Server Side**

- Domain model identification
- Domain model prototype & analysis
- Domain model update & integration
- Detailed design

---

**USCViterbi**

School of Engineering

2018 Spring
Overview of generated code and benefits

Auto-generated database functions

Java  C#  ...
Comparison of tickets distribution

Tickets related to domain modeling and server-side construction for service-oriented projects
• More effort spent on design update in early stages
• Less issues generated in later stages
Infrastructures

• Domain Modeling Tool
  — Enterprise Architect (EA) & Visual Paradigm (VP)

• Amazon EC2
  — Business Server (Python Flask)
  — RESTful API server (NodeJS), DB (MongoDB)
Enable per-use-case-basis sandbox development in parallel

- Initial Domain Model
  - Updated Domain Model
  - Updated Domain Model
  - Updated Domain Model

Integration

Detailed Domain Model
Feedback-driven on bug-free featured templates development

• Functionality
  — Large files, Email, Spatial querying, etc.

• Server-side Observability
  — Logging, Exception, Health

• Security
  — Encryption and Permission control

• Experience gained from Parallel Agile projects
  • original concept (LBA)
  • need for large file support (CarmaCam)
  • need for API documentation (TikiMan and CarmaCam)
  • need for database and API security (CarmaCam)
  • ...
• Extract potential entities from User Story
  — Use case specification
  — Win conditions
    - 39 projects, 865 win conditions
    - 80% **Functional requirements**
      As [a|an] ...[, ][I i] [can|should|shall]...
    - 12% non-functional requirements
      (The)?[Ss]ystem [shall|should|must]...
    - 8% Others

• TDs and POS-tags
  — Type Dependencies (TDs): Grammatical dependency between the words of a sentence.
  — Parts-of-Speech tags (POS-tags): words in a sentence tagged (or annotated) with parts of speech, such as noun, pronoun, verb, adjective, etc.
As a user, I can view other users' profiles so that I can see their information and points.

Figure out nouns based on POS-tags

Figure out Sentence Structure Rule based on TDs

<table>
<thead>
<tr>
<th>Users’ Profiles</th>
</tr>
</thead>
<tbody>
<tr>
<td>- information</td>
</tr>
<tr>
<td>- points</td>
</tr>
</tbody>
</table>
• Intersection
  “email” appears in both user case.

• Intersection*
  “username” and “useraccount” in different use case are talking about the same thing.

• Relative Complement
  “password” and “lastlogin” only appear in Developer#1’s use case while “nickname”, “fullname” and “age” only appear in Developer#2’s use case.

• There may be other cases (TBD)
  E.g. an attribute is removed or moved from User entity to another entity. (Out of the problem scope)
Solution

• Heuristic:
  — Infer from the DataType of attribute. It will partition potential attributes with same data type. Attributes with different data type cannot be Intersection*
  — Infer from the testing data about the value format of the attributes. Same data format (e.g. email format, phone # format, all digits, same length or same complex data type, etc.) may denote to Intersection*
  — Semantic Similarity could be cost function:
    – Pre-define a synonyms model, then figure out Cosine or Euclidean during the runtime.

• The result could be sent to human for final adjustment
Thank You

• Questions?