
sfafinity

Technology Overview

M. A. Sridhar

sridhar@sfafinity.com

Platform: Pure Java

- Runs on virtually any hardware
- No JNI or other platform dependencies
- Servlets only (no EJBs or other app server components)
 - In fact, just *one* servlet for the whole app
- Compact, lightweight, high-performance runtime engine
 - *Entire* application package is shipped in 7 MB

Database interface

- JDBC only – no O/R mappers or other third-party components
- Uses industry-standard SQL92 structures
- Does not use triggers, stored procedures or other vendor dependencies
- App can be re-targeted from one JDBC-compliant database to another almost instantaneously
- Connection pooling, caching and prepared statements enhances performance

Business logic

- Complete separation of logic from rendition
 - Same code base used to provide multiple rendition modes (HTML, XML, CSV, etc)
- *Data model* created from metadata describing each data source
 - Easy to aggregate data from multiple sources (databases, email, web services) *within the same data model*
- Application code incorporates *no knowledge* of database schemata, rather is driven entirely by data models

Data models

- The core of the system: “user data model”
 - Data aggregates mapped into a tree
 - Strongly typed: Good error checking
 - Represents the dynamic data elements needed in a transaction (e.g., a web page)
 - Incorporates data from multiple data sources, as well as their relationships
 - Declarative, thus easy to understand
 - Evolves easily over time, as functionality evolves
 - No need to change the application code
 - Data source access via Java “data port” interface

Database-specific data models

- Strongly adhere to data integrity constraints (foreign key, uniqueness)
 - Data update operations *automatically* happen in correct foreign-key dependency order
- Modeling depends on the input schema, not the actual database schema
 - Makes it easy to offer app-level data integrity on databases such as MySQL or SQLite, which don't use foreign keys

Data models

- Data models can be automatically inferred from database schemas
- Custom data models easily created
- Core technology described in <http://www2002.org/CDROM/alternate/698/index.html>
- Mature, proven technology base – has been in use for several years, in many applications

Rendition layer

- “Meta-templates” simplify creation of common renditions
 - HTML meta-template features automatic generation of Javascript validation code, sorting, paging and other common housekeeping tasks
 - Other renditions (CSV, XML etc) are easy
- Templates are used extensively, e.g., querying web services (so we don’t need Apache Axis or other such extensions)
- Uses a customized version of freemarker (now <http://fm-classic.sourceforge.net> and <http://freemarker.org>)

User interfaces

- Web UI: lightweight pure HTML and Javascript
 - No ActiveX or other proprietary technologies
 - Supports all DOM-compliant browsers
- Easy to create multiple rendition modes, because of reliance on data models
 - E.g., Outlook Sync, data export and partner data exchange work as if they are just another user interface

Domain-specific technologies

- *Data import engine*, for uses such as importing CSV or XML data
 - Maps business logic rules for import to database schema
 - Used in multiple contexts, e.g., in-place editing
- *Filtering and reporting engine*, for database-independent multi-entity reporting
 - Can filter across multiple entities
 - Generates efficient SQL for reporting across arbitrary numbers of database entities and attributes

Web service interaction

- Each web service is modeled as a collection of data aggregates
 - Integrates smoothly into the data model
 - Provides strong typing
- Interaction occurs via data port and template expansion – thus no additional components needed

Thank you!

M. A. Sridhar

sridhar@adinxight.com

<http://www.adinxight.com>