



J2EE Evaluation Module Primer

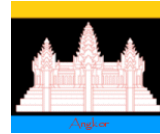
Introduction

Many organizations wish to port Lotus Notes applications to a J2EE server (such as Websphere). Regardless of the reason for such a port (open standards, scalability, technical skills, or political pressure), wholesale ports of Lotus Notes applications represent a major technical and managerial challenge.

While all porting tools on the market promise “automation”, their success is directly proportional to the simplicity of the Notes designs. The task remains mostly manual, requiring expertise in both Lotus Notes and J2EE. This explains why despite IBM’s commercial push for Websphere, so few Notes applications have been ported. The fully automated port from Notes to J2EE will remain a technical Holy Grail, with incremental gains expected from tool vendors, mostly in the area of automated migration of the data rather than the designs.

Given the lack of predictability and the importance of labor, how does a manager estimate the work necessary to port Lotus Notes applications? How does she evaluate the project and prioritize the applications to process?

The answer requires unbiased and factual analysis: collecting information about the designs, their inherent complexity and combining it with estimates of the challenge each represents for a programmer “skilled in the trade” to execute the port.



Methodology

Angkor is a proven analytical tool, capable of discovering, scanning and retrieving the design of Lotus Notes applications. Angkor is capable of processing a single or tens of thousands of designs of a Domino infrastructure.

The proprietary extraction engine leaves no design element behind, storing in a central RDBMS even the most remote properties. The open and extensible rule-based engine permits the wholesale yet rapid analysis of the designs to suit a variety of intents.

The new J2EE rule-set succeeds a long line of proven modules, starting in 1998 with Y2K, then R5, ND6 and today's R7 and UNIX incompatibilities modules. The J2EE module leverages several components of Angkor:

- Design complexity: the ability to list, weigh and compile in an index all the design elements that make a Notes application, such as views, forms, fields and server-based agents.
- Code complexity: the unique ability to look at every line of code, regardless of its programming language, and compute an index
- Rule-based identification of "Notes-isms" such as proprietary security restrictions (readers fields, encrypted sections, etc.), full-text index, design element inheritance and delivery platform (Notes client, browser, mobile).
- Customizable weights that permit the computation of the indices based on the intent of the module.
- Understanding of the inter-dependencies of the code (external references)
- Usage module with information about the volume of documents, their frequency of use and the application stakeholders "owners".

© Copyrights 2006-2007 Atlantic Decisions, Inc.



Overview of the J2EE Module

1. User interface

- Identification of a Notes client vs. Web-enabled Notes database. Web-enabled elements have unique properties that are hard to port to J2EE:
 - o Pass-thru HTML, which is directly written at the Notes form
 - o Special Notes field as \$\$HTMLHead, Query_String...
 - o Lotus Script or Java agent triggered by WebQueryOpen and WebQuerySave
 - o JavaScript, Stylesheet and image

Such information is not available through the Notes Synopsis and requires the use of complex C API queries.

- Identification of UI enhancement. This means keeping track not just of fields on a form, but also
 - o Fonts
 - o Colors
 - o Backgrounds and colors
 - o Layout
 - o Fields properties

2. Business Logic

Many organizations have a large number of simple applications who may not warrant a port to begin with. Those with a substantial value to the organization will include business logic. For instance:

- Identification and quantification of the use of Sub-form and Computed Sub-form
- Identification and analysis of the number of instances and complexity of LotusScript and Java Notes agents.

3. Database

J2EE uses a relational database as a repository. It is significantly different logically from Notes (relational vs. object-based). A port should not assume that creation of an identical schema is the most desirable implementation.

- Identification of the impact of a 1-to-1 mapping of the Form fields
- Identification of fields with security restrictions
- Identification of logical grouping of fields (for reasons of security, cosmetics or convenience)
- Identification of external data references (run-time)
- Identification of rich-text fields
- Analysis of the duplication of on-disk fields