MOVE-10: Darwin or Big Bang?

A Darwinian Case Study of a WebSpeed® Application's Evolution

SMAT-Team Tobago

We Are The Temporary Extension of Your Development Department

Thomas Hutegger & Pheona Job

tmh@theNatureIsle.com  ptj@smat-consulting.com
Agenda

• **Overview – Modernization Questions**
  – WebSpeed

• Version 0 (Prototype, Legacy Code)
  – WebSpeed

• Version 1 (some UI/BL separation)

• Version 2 (Business Logic GUI ready)
  – OpenEdge® Reference Architecture
  – ProDataSets

• Wrap-Up
Goals For This Session

“Application Modernization is a fact of life, and a never ending battle - but also a never ending opportunity to make it better…”

- Show real-life example of step-by-step modernization
- Provide seeds for thought
- Encourage you to try it too
Architecture Overview – Layers

Page 1 | Page 2 | Page 3
---|---|---
| BE.p | DA.p |

DB

Version 0 | Version 1 | Version 2
Three Modernization Main Questions

• How to reuse as much as possible of the existing application?
• How to integrate programs of the old and the new architecture within one application, so they can run in parallel?
• How to design the new architecture, so future modernization efforts can be implemented easily, and with maximum code-reuse?
Agenda

✔ Overview – Modernization Questions

→ Version 0 (Prototype, Legacy Code)
  – WebSpeed

• Version 1 (some UI/BL separation)

• Version 2 (Business Logic GUI ready)
  – OpenEdge Reference Architecture
  – ProDataSets

• Wrap-Up
WebSpeed Architecture

Client

Browser (Opera)

Intranet

Internet

Server

Web Server (Apache)

Agent

Progress

DB
Seed of Thought

• 3 Event-types submit a page:
  – View (first request [menu], refresh, change in filter values, request details...)
  – Print (providing what and how to print)
  – Save (providing data to save, Ids to delete)

• Posting page can be
  – Same as next page (next instance, save&refresh,...)
  – Different from next page (browser->viewer,...)
WebSpeed Round Trip

Agent

Web-Server
- Name-Value Pairs
- HTML

DB

- Read Data from Page
- Save Data to DB
- Create Next HTML Page
- Get Data from DB

MOVE-10: Darwin or Big Bang?
Architecture Overview – Version 0

Web-Server

Agent

PPO.p

Save Event
RetrievePageData

Framework

PPO.p

saveData

PreparePage

GetData:
Block1
... 
Blockn

DB
Main-Issues – Version 0

• Duplication of code
  – Drop-downs, pages with similar data, ...
• Fields as parameters
• No specialization is possible
• GUI module would require reprogramming and further code-duplication
• QA: several steps testable only in one block
Architecture Overview – Version 0

Agent

Web-Server

Framework

PPO.p

Save Event

RetrievePageData

saveData

PreparePage

GetData:
Block1
...
Blockn

HTML saved in plain OS text file

Simulating Test Program

DB

Provide Name-Value Pairs Temp-Table
Seed of Thought

• Testing step by step makes life much easier
  – Prototyping and development
  – Debugging
  – Maintenance
Agenda

✔ Overview – Modernization Questions
✔ Version 0 (Prototype, Legacy Code)
  ✔ WebSpeed
→ Version 1 (some UI/BL separation)
  • Version 2 (Business Logic GUI ready)
    – OpenEdge Reference Architecture
    – ProDataSets
  • Wrap-Up
Main-Adjustments – Version 1

→ Data layer

→ Minor UI enhancement
Version 0 – Version 1: Data-Layer

Web-Server

Agent

Framework

PPO.p

Save Event
RetrievePageData

PPO.p

GetData:
Block1
... 
Blockn

PreparePage

saveData

DB
Architecture Overview – Version 1

Simulating Test Program

Web-Server

Framework

Agent

- Provide Name-Value Pairs Temp-Table
- HTML saved in plain OS text file

PPO.p
- Save Event
- RetrievePageData

PPO.p
- PreparePage

BL.p
- saveData

GetData:
- Block1
- ... Blockn

DB

HTML saved in plain OS text file

X

X
Improvements

• Separated DB access from Front-End specifics
  – Allow programmer's specialization ("backend" - “frontend”)
  – Programs less cluttered → easier debug-able and maintainable

• Test DB-Access separately

• Use Temp-Tables, instead of single parameters
  – adding/deleting fields impacts end-points only, but not the plumbing
Seed of Thought

- Specialization allows average programmer to become an Expert
Main Issues – Version 1

• Back-end not GUI ready

• Performance
  – Drop-Downs
  – Extensive use of static temp-table parameters
  – Replace function of editor EXTREMELY slow

• Front-End
  – UI data structure <> DB data structure
  – Component-ize pages (code reuse)
Agenda

✔ Overview – Modernization Questions
✔ Version 0 (Prototype, Legacy Code)
  ✔ WebSpeed
✔ Version 1 (some UI/BL separation)
  ➔ Version 2 (Business Logic GUI ready)
    ➔ OpenEdge Reference Architecture
    ➔ ProDataSets
• Wrap-Up
Main Adjustments – Version 2

- Open Edge Reference Architecture
- ProDataSets
- Card-oriented UI (vs. page-oriented before)
OpenEdge Reference Architecture

- Separated presentation and integration layers
- Common business logic with advanced models
- Data access abstracted from storage
WebSpeed Implementing The OpenEdge Reference Architecture

Web-Server

Agent

- Read Data From Page
- Create Next HTML Page

Business Entity

Data Access Object

DB

Presentation

Enterprise Services

Business Services

Data Access

Data Sources

Common Infrastructure
Architecture Overview – Version 2

Web-Server

Agent

Framework

PPO.p

PreparePage

GetData:

Block1

... Blockn

BusinessRules

BE.p

saveData

DA.p

GetData:

DB

MOVE-10: Darwin or Big Bang?

25
Issues With WebSpeed

• Did record change since page was prepared?
  – Additionally sending complete record to page
    • So it can be retrieved as “before image” record
  – Storing before-images in context DB
  – Using “version” field

• Total disconnect between preparing page and next post
  – ProDataSet content lost
  – No before-image temp-table possible
  – Standard ProDataSet update feature can't be used
Seed of Thought

- **WebSpeed is “Batch-Client” - WebService too**
  - Plumbing and best practices for WebSpeed are useful for WebServices too

- **Context Management**
  - Sophisticated system -> John Sadd's white paper on PSDN
ProDataSet Related Adjustments

• **PageProcessObject**
  – Plumbing changed to use ProDataSet
  – Use of data the same as before!

• **DataAccessObject**
  – SaveData procedures the same
  – GetData procedures:
    • used as call-back procedures for TT
    • Parameter handling different; rest the same!
Architecture Overview – Version 2

Web-Server

Agent

Framework

PrepPage

PPO.p Save Event

Save Events

CPO.p PrepCard

Simple.p PrepCard

CPO.p

Prepare Card

BusinessRules

BE.p

GetData: Block1 ...

Blockn

saveData

DA.p

DB
Card-Oriented System

- Single program for all simple cards
- Use replace on string instead of editor
- Cards generic enough to be used in multiple pages
Improvements

• Back-end ready for GUI
• Improved overall performance
  – Codes and other semi-static data cached
  – Javascript for Drop-Downs and Menues
  – BY-REFERENCE for data-structure parameters
• Moved to cards as process-unit (instead of whole page)
Agenda

✔ Overview – Modernization Questions
✔ Version 0 (Prototype, Legacy Code)
  ✔ WebSpeed
✔ Version 1 (some UI/BL separation)
✔ Version 2 (Business Logic GUI ready)
  ✔ OpenEdge Reference Architecture
  ✔ ProDataSets
→ Wrap-Up
Outlook – Version 3

• Improving Standards – the “How”
  – Performance
    • More “Profiler” work
  – Productivity
    • More code-generation

• Expanding Functionality – the “What”
  – Add GUI for certain functionalities
  – Move larger existing parts to the new architecture
Goals For This Session

“Application Modernization is a fact of life, and a never ending battle - but also a never ending opportunity to make it better...”

• Show real-life example of step-by-step modernization
• Provide seeds for thought
• Encourage you to try it too
Questions?

Thomas Hutegger & Pheona Job
tmh@theNatureIsle.com  ptj@smat-consulting.com
www.smat-consulting.com
Thank you

www.smat-consulting.com

tmh@theNatureIsle.com  ptj@smat-consulting.com