Creating an Application Framework to Update a Legacy ERP

Apero’s Journey to Modernization

Oscar Perez & Sophy Nathanail
Apero Solutions Inc.
Modernization – A Simple Matter of Survival

Without starting from scratch on new development, modernizing legacy software allows organizations to maximize the use of their existing application assets and investments. It’s a simple matter of survival.
Agenda

- Story of Apero’s Progress ERP Solution
  - Where We Came From
- New Emerald GUI Application Framework Strengths
- Challenges and Things to Watch Out For
- Emerald Framework in Action
  - Emerald Architecture
  - Future Roadmap
- Recommendations
Apero Solutions Inc.

- Based in Edmonton, Alberta, Canada
  - Employee group in Edmonton, Calgary, Vancouver and Toronto
  - Offshore partner development team

- Legacy ERP Solution written in Progress
  - Roots in the early 1980s Progress development platform
  - Small Solution provider to mid / large size hard goods distributors
  - Global customer base
New Company, Apero Solutions, formed 2011

Problems facing the new company

- Old Legacy ERP
- Migration path for Existing Customers

Goals

- New Rich User Interface BUT support CHUI architecture
- Future proofing our modernization
- Leveraging years of business logic development
Apero’s Solution

- **Options:**
  - Start New Development
  - Modernize our existing application

- **Decision:**
  - Move core business logic from the traditional Progress legacy code to service-orientation
  - Same services to be called from multiple UI platforms
  - Use the standard Progress Application Server Technology
  - Develop a new rich UI based on .net and Browser delivery
ERP Trax Product was Developed on an Application Framework “Workbook”

- Traditional “culture” for development based on toolkit “workbook”
- Allowed for consistent character user interface
- Organized Application Structure
- Generally Kept re-usable business logic “free” of UI code
- Code Generation for basic CRUD operations and master table list and printing
- Generic pass of dynamic parameters among shared application code
Elements & Goals for the New Application Framework “Emerald”

Application framework was crucial

“Emerald” application framework designed to support:

- Browser based graphical user interface (GUI)
- Progress Web Services and REST services calls
- More flexible and configurable data inquiries
- More solid application-security control
- Transaction control and record locking for data modifications
New Emerald GUI Application Framework Strengths

Consistent UI

- Standard side bar menus, search options, lookup of tables, display format of individual records, insert of new records, edit of existing records, etc.
- UI controlled by Emerald and not individual programmer
New Emerald GUI Application Framework Strengths

Consistent UI

Component-Type Definition

SEARCH

SO

AR

HOME

CUSTOMER LOOKUP

SELECTING A COMPONENT TYPE
Each type of component offers specific functionality

Viewport

LOOKUP

SETUP

LITERAL

New Emerald GUI Application Framework Strengths

Component-Type Definition

SEARCH

SO

AR

HOME

CUSTOMER LOOKUP

SELECTING A COMPONENT TYPE
Each type of component offers specific functionality

Viewport

LOOKUP

SETUP

LITERAL
New Emerald GUI Application Framework Strengths

Consistent UI

Component-Type Definition

SELECTING A COMPONENT TYPE
Each type of component offers specific functionality
New Emerald GUI Application Framework Strengths

Consistent UI

Component-Type Definition

SELECTING A COMPONENT TYPE
Each type of component offers specific functionality
New Emerald GUI Application Framework Strengths

Component-Type Definition

SELECTING A COMPONENT TYPE
Each type of component offers specific functionality

Consistent UI
New Emerald GUI Application Framework Strengths

Consistent UI

UI based on XML Definition
New Emerald GUI Application Framework Strengths

```xml
<?xml version="1.0" encoding="utf-8"?>
<Application>
  <Component name="Account.Lookup" refId="Account" bllProcess="Gl_Account" closable="yes" dataModelName="Gl_Account" requestBy="Query" maxRecords="100" pageSize="10">
    <Description>Account</Description>
    <Help>Account Inquiry</Help>
    <Models metaModel="Gl_Account" fieldLists=""/>
  </Component>
</Application>
```

Diagram showing the interaction between UI Controller (RenderLookup.asc) and Data Controller (QueryProvider.asc) with HTML and JSON as data formats.
New Emerald GUI Application Framework Strengths

Organized Application Structure

<table>
<thead>
<tr>
<th>JAVASCRIPT LIBRARY</th>
<th>Presentation Layer</th>
<th>DOT.NET MVC FRAMEWORK</th>
<th>BUSINESS SERVICING LAYER</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOT.NET UI RENDERING ENGINE</td>
<td></td>
<td>GENERIC QUERIES</td>
<td></td>
</tr>
<tr>
<td>RPC</td>
<td></td>
<td>PROGRESS OE WEB AND REST SERVICES</td>
<td></td>
</tr>
<tr>
<td>PROGRESS OE APPLICATION SERVERS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROGRESS OPENEDGE DATABASE</td>
<td></td>
<td></td>
<td>DATA ACCESS LAYER</td>
</tr>
</tbody>
</table>
New Emerald GUI Application Framework Strengths

Reuse Our Legacy Business Rules Code

- **Character Code**: so/GetOrd.p, so/AddLine.p
- **Common Code**: So/GetOrd.p, So/AddLine.p
- **Rest Services**: ws/GerOrd.p, ws/AddLine.p
- **Dot.Net / JS**: Rest/GerOrd.p, rest/AddLine.p
- **OEMobile/ JS**: Rest Services
- **Mobile App**: 3rd Party Applications
- **Rich Web App**
New Emerald GUI Application Framework Strengths

Leverage Our Existing Form Printing and Report Engine

- Existing Reports and forms automatically displayed
- Character report run through a screen scrapper to auto generate create Postscript to PDF file

Database report option records automatically render in browser UI
Challenges Apero Faced in Development

Development Issues

- Extracting embedded Character UI not an easy task
  - Sometimes it is better to start with a fresh design

Customer Credit Limit Calculation

Include File: ar/cstclklc.i
Challenges Apero Faced in Development

Development Issues

- Supporting both types of application UI, Character and Graphical/web, adds complexity to:
  - Implementation of transaction-control scope
  - Implementation of record locking control to support both environments
Challenges Apero Faced in Development

Development Issues

Flexible XML definitions for UI can be cumbersome & required good documentation
Challenges Apero Faced in Development

Development Issues

- Reusing legacy code that uses include files (.i) with shared variables
- Performance for data retrieval through web calls & also Rich UI screen rendering
- Creating Open Edge generic and dynamic code to receive CRUD-type call from clients
Challenges Apero Faced in Development

Use of Dynamic Temp-Tables and Queries

```
hBuffer = phDataSet:GET-BUFFER-HANDLE(i).
ValidCS = false.
LatTable = substring(hBuffer:name,3).

create buffer hBuffer for table LatTable.

/* Creating record output */
create temp-table MTable.
MTable:ADD-NEW-FIELD ("CheckSum", "character","","").
MTable:ADD-NEW-FIELD ("RowId", "character","","").
MTable:ADD-NEW-FIELD ("ParentRowId", "character","","").
MTable:ADD-NEW-FIELD ("Action", "character","","").
MTable:TEMP-TABLE-PREPARE(T " + hBuffer:name).

CREATE QUERY qh.
qh:SET-BUFFERS(hBuffer).
qh:QUERY-PREPARE(substitute("for each &1 no-lock", hBuffer:name)).
qh:QUERY-OPEN().
qh:GET-FIRST().
```

Event Driving Calls to Business Rules (Implemented using OOP)

```
writeEvent = new WriteEvents (input LatTable, input pUserID).
/* Business rule — Init Context */
writeEvent:InitContextWriteRel(abb).
repeat transaction on error undo, leave:
  case Action:
    when ADD_ACTION then
do:
  hBuffer:BUFFER-CREATE ()..
/* Copy all the buffer fields manually (not use buffer-copy ) to allow
of automatic data-type casting */
writeEvent:BeforeInsert(abb).
```
Future Features for our Application Framework

Emerald Roadmap

- Graphical screen design / mapping tool for the developer
- More templates for screens
- User customized screens
  - Re-arrange columns and fields and save the “view”
  - Hide / secure tabs
  - Hide / secure at the field level
Modernization – Why It’s Worth the Effort

- Richer User Interface / Experience
- Utilizing New Technology
- Opening the re-usable business logic layer to multiple UI platforms (mobility)
- Increased compatibility with 3rd party products

Progress Application is Easier:
- Market
- Demo
- Sell
- Support
Why Devote Time and Money to an Application Framework Design?

- Consistent User Interface
- Empowers Developers
- Faster & Lower Cost
- Organized Application Structure
Visit the Resource Portal

- Get **session details & presentation downloads**
- Complete a **survey**
- Access the latest Progress **product literature**

www.progress.com/exchange2014