Analysis Problems in ABL and How To Solve Them

Dr. Thomas Mercer-Hursh
VP Technology
Computing Integrity, Inc.

Let me begin by introducing myself. I began working with Progress in 1984 and I have been a Progress Application Partner since 1986. For many years I was the architect and chief developer for our ERP application. In recent years, I have refocused on the problems of transforming and modernizing legacy ABL applications. To transform an application, one must first understand the application. That, and a long history of modifying existing systems is the background for today’s presentation.

We are going to cover a lot of ground today. This means that I am going to point to a lot of possibilities and try to give you an idea of how it might be useful to you, but I’m not going to go into detail on any one tool. Hopefully, I can make you aware of what is possible and you can apply this to your own work.
So, here’s our agenda for today. First we are going to talk a bit about the problem and why it is important and then we are going to go through some broad categories of analysis problems and what tools there are to solve them.
First, let’s talk a little about why analysis is important.
Knowing the desired behavior for a program modification is only part of the problem. One also needs to know where to change the behavior and the impact of making that change.

In mature systems, it is common for a “simple” change to cause unexpected consequences and bugs which take more work to fix than the original change … not to mention other consequences.
Good analysis is good risk management.

Good analysis is understanding before doing.

Good analysis is like a surgeon knowing where to cut and knowing where not to cut.

Not doing thorough analysis means unexpected results, inevitably causing longer cycles to complete changes.

Good analysis is good risk management.

Good analysis is like a surgeon knowing where to cut and knowing where not to cut.

Good analysis is understanding before doing.

Not doing thorough analysis can mean unexpected results and much longer cycles to complete changes.
The simplest analysis problem in ABL is finding out basic information about a specific compile unit which it the current focus of attention.
COMPILE LIST

- Built-in to ABL compiler.
- Illustrates what’s in the program with includes in-line.
- Resolves pre-processor code.
- Shows scope of transactions and buffers.

<table>
<thead>
<tr>
<th>File Name</th>
<th>Line</th>
<th>Blk. Type</th>
<th>Tran</th>
<th>Blk. Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>...amples\src\test.p</td>
<td>0</td>
<td>Procedure</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>...amples\src\test.p</td>
<td>20</td>
<td>Do</td>
<td>No</td>
<td>OUTSIDE-BLOCK</td>
</tr>
<tr>
<td>...amples\src\test.p</td>
<td>22</td>
<td>For</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Buffers: Sports.bCustomer</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

COMPILE LIST

Built-in to ABL compiler
Illustrates what’s in the program with includes in-line
Resolves pre-processor code (i.e. code which is evaluated before compile, e.g. operating system dependent code)
Shows scope of transactions and buffers
COMPILE XREF

- Built-in to ABL compiler.
- Shows index usage of each line which references an index.
- Shows table and field access by line.
- Shows string references and other less frequently useful information.

```
...src\test.p ...src\test.p 21 REFERENCE Sports.Customer
...src\test.p ...src\test.p 22 ACCESS Sports.Customer Name
...src\test.p ...src\test.p 22 SEARCH Sports.Customer Name WHOLE-INDEX
...src\test.p ...src\test.p 23 ACCESS Sports.Customer Name
...src\test.p ...src\test.p 23 ACCESS Sports.Customer Name
...src\test.p ...src\test.p 23 UPDATE Sports.Customer Name
```

COMPILE XREF
Built-in to ABL compiler.
Shows index usage of each line which references an index.
Shows table and field access by line.
Shows string references and other less frequently useful information.
XML XREF

- Built-in to OpenEdge Architect.
- Shows index usage, table and field access by line (same as COMPILE XREF) but in XML form:
  - Easier to parse into database.
  - Harder for human to read.

XML XREF in OpenEdge Architect

Shows index usage, table and field access by line (same as COMPILE XREF) but in XML form:

- Easier to parse into database
- Harder for human to read
In addition to wanting to know about a single compile unit, one often wants to know about bigger issues like where a particular table or routine is accessed throughout the code base, e.g., when one needs to make a change and wants to see the impact.
Database of XREF Data

- XREF data of each compile unit is loaded into a database which allows querying, e.g. of all places where a table or index is used.
- Many people have built XREF databases.
- XREF databases exist in several frameworks.
- No standard implementation.

I am currently working on an open source offering in this area.
The closest thing to a “standard” implementation which includes an XREF database of this type is the Roundtable TSMS Software Configuration Management system. It is a commercial product and does a lot more than just provide the XREF data. They also store searchable information like annotations, properties, methods, internal procedures, shared variables, etc.
Another common analysis task is to find references to a table, procedure, variable, etc. in the code. A related task is to compare two or more sets of code to determine what the differences are between them.
OpenEdge Architect

- Ctrl-F: Search and replace in current file.

OpenEdge Architect
Search and replace can be performed in current file, workspace or selected resources (directory, files, CTRL + click, etc.).
Has problem shared by non-ABL tools where search is performed on string patterns, not ABL syntax (e.g. search for string will find in comment or variable name).

Ctrl-F and Ctrl-H in OEA
OpenEdge Architect

- Ctrl-H: Search and replace on workspace or selected resources (directory, files, CTRL + click, etc.).

OpenEdge Architect

Search and replace can be performed in current file, workspace or selected resources (directory, files, CTRL + click, etc.).
OpenEdge Architect

- Has problem shared by non-ABL tools where search is performed on string patterns, not ABL syntax (e.g. search for string will find in comment or variable name).

E.g., consider a search and replace for Customer with bfCustomer:

```plaintext
/* Total all orders for the Customer */
```

OpenEdge Architect

Search and replace can be performed in current file, workspace or selected resources (directory, files, CTRL + click, etc.).

Has problem shared by non-ABL tools where search is performed on string patterns, not ABL syntax (e.g. search for string will find in comment or variable name).

Ctrl-F and Ctrl-H in OEA
Non-ABL-specific Search Tools

- Many tools are available.
- Share problem of being unaware of ABL syntax.
- Can be used with Version 9 and earlier code (i.e., no need for OpenEdge Architect).
- Able to search a body of code without creating a project in OpenEdge Architect (e.g. search entire codebase).

Other tools nice interface, but still not ABL specific
File and Directory Comparison Tools

- Many tools available which compare 2 different versions of programs.
- Preferred are tools which can compare 3 items (programs or directories). Handy when base program has been modified by 2 programmers and need to figure out how to bring them together again.

(no experience with this one yet)
See slide at end. Very quick illustration of finding differences in two trees.

Demo was performed from own laptop compare IS/rc/rcal with /tmp and show program
Sometimes, the best way to do analysis is to record design decisions when the code is created in the first place because then you can just read how something works without having to figure it out from scratch. While comments provide some assistance here, one can do much better.
Commercial product “formerly” available from Joanju:

Parses code and annotations

Handles classes and procedures

Does not handle anything beyond 10.2B
• Product to be available from Consultingwerk and Riverside Software.
• Based on two new tasks in PCT.
• Parses code and annotations.
• Produces HTML documentation.
• Applicable only to OO code.

Exchange Session: Automated Class Reference Generation by Mike Fechner
Need some samples and more information from Mike.

Demo results from
So far, we have been looking only at static analysis, i.e., the code sitting in a file on a disk. Some problems, though, are only easily identified when the code is running. Other than putting in message or log statements, there are several tools which can help us understand running code.
When It Runs, What Happens?

Debugger

Allows controlled execution and monitoring of a running program instance:
- Breakpoints – fixed and computed.
- Stepped execution.
- Variable values.
- Object status.

Standard part of both Windows and Unix distributions.
When It Runs, What Happens?

Debugger

This breakpoint:

```plaintext
chLogName = "CollectionPerformance_Log.txt".
output to value(chLogName);
output close.
```

Allows seeing current variable values:
The profiler is built into standard ABL, but is not documented or officially supported. There is a document at the URL shown which briefly documents a GUI tool for using the profiler. See also $DLC/src/samples/profiler depending on the installation choices you have made. There is a Knowledgebase entry documenting the options for the profiler object at http://knowledgebase.progress.com/articles/Article/19495?q=profiler+handle &l=en_US&fs=Search&pn=1
To use:

• -profile run time parameter. Profiles whole session, but does not require modifying code.

• A GUI tool which can be found for running selected procedures (http://communities.progress.com/pcom/docs/DOC-2808 )

• The built-in profiler object which allows selective execution on selected blocks of code, but does require modifying the code. (http://knowledgebase.progress.com/articles/Article/19495?q=profiler+handle&l=en_US&fs=Search&pn=1)
Here is a very simple example using the built-in Profiler object.
LOG-MANAGER System Handle

Provides a reliable method to collect data during execution without interrupting the execution process with MESSAGE statements and with modes for systematic collection of many events and transitions without coding.

To help you understand the output, see

- [http://www.oehive.org/project/Profiler](http://www.oehive.org/project/Profiler)
- [http://www.oehive.org/project/log-manager-reporter](http://www.oehive.org/project/log-manager-reporter)
ProTop

Provides real time monitoring of many aspects of database performance, thus creating insight into possible coding problems that may not be apparent from examining the code itself.

Obtain from http://dbappraise.com/protop.html
Improving code quality isn’t analysis, per se, but can make it less likely to encounter problems in production and make it easier to understand code during maintenance. Moreover, some of the tools used for code quality improvement can also be used for certain analytical issues. There are many behavioral approaches to code quality, but today I am focused specifically on tools so I am going to look at tools which can read and understand the syntactic structure of the code.
What Is Parsing?

Parsers analyze code in the same way a compiler does, recognizing and resolving:
- tokens
- keywords
- table names
- field names
- etc.
Proparse

- Parses code and creates abstract symbol tree in memory.
- Open source product created and updated by John Green (Joanju).
- Available at http://www.joanju.com/proparse/index.php
- Is no longer maintained and has no support for any syntax past 10.2B.
How Do I Get Better Code Quality?

ProLint

- Uses Proparse to apply wide variety of code quality tests which help avoid errors and impose shop standards.
- Open source product created by Jurjen Dijkstra with multiple contributors.
- Rules are written in ABL, so easily modified.
- Has the same version limitation as Proparse.

**ProLint**

- Uses Proparse to apply wide variety of code quality tests which help:
  - Avoid errors
  - Impose shop standards
- Open source product created by Jurjen Dijkstra with many contributors.
- Available at [http://www.oehive.org/prolint/download](http://www.oehive.org/prolint/download)
Proparse Scripting

- Uses Proparse.
- ABL code can be written which uses Proparse to perform special tasks outside of Proparse’s purview, e.g. find all input and output statements in a set of code (finds all places where code interacts with external files).
- Searches are ABL syntax aware.
- Write your own scripts to suit your individual need.
- Has the same version limitation as Proparse.
Joanju Analyst

- Based on Proparse, reads all code, builds database.
- Same version issues.
• Data base includes:
  • Connections between run statements, methods, procedures.
  • Connections to database tables (to field level).
  • Dynamic call resolution (uses combination of automatic analysis and hints when unable to resolve calls).

• Data base includes:
  • Connections between run statements, methods, procedures.
  • Connections to database tables (to field level).
  • Dynamic call resolution (uses combination of automatic analysis and hints when unable to resolve calls).
Joanju Analyst (con’t)

- Html browser allows one to follow links:
  - From the run statement to the code that is run.
  - From an internal procedure or method to all the places which call it (where used).
- Very flexible, ABL syntax aware search tool.
- Produces Bill of Materials output (XML) which drives ABL2UML.
- Used for productivity, impact and flow analysis, debugging and re-engineering.

Analyst is no longer marketed, but can be available by special arrangement. See me if interested.

Watch videos?
Tools discussed so far provides a utilities to help explore the code, allow tracing some feature of the code, or examines how the code behaves. But, what if one is looking for the big picture, how it all fits together, the structure of the interaction of the code. For this, we want *pictures*!
Among those emphasizing analysis and design, the strongly predominant way of expressing that design is UML (Unified Modeling Language).

UML was created in the mid-1990s to unify a diverse set of modeling languages which had grown up, primarily for OO development. A standards body, the OMG or Object Management Group was created to oversee this and other standards and UML has undergone considerable expansion and development since the original version.

Different people use UML in different ways. Some use it simply as a sketching tool, something to put on a white board or in a document to facilitate discussion. Some will use it more completely to do a detailed analysis of a system and then write code from that design. They may or may not keep the design in sync with the code as the system evolves, although it is usually regrettable if they don’t. And, there are those … which is what interests us today … who actually generate the working code directly from the model.
Enterprise Architect

- Most favored in ABL community because of vendor support for using OpenEdge database as repository and multiple supporting tools.
- OE datatypes available:
  - Dr. Thomas Mercer-Hursh [http://www.oehive.org/node/1073](http://www.oehive.org/node/1073)
ABL2UML

- Open source tool created in 2007 by Dr. Thomas Mercer-Hursch [http://www.oehive.org/ABL2UML](http://www.oehive.org/ABL2UML)
- Takes schema from database and Bill of Materials from Analyst and builds UML Component diagram with code units, links and connections, database tables and fields.
- Contains diagram builder to automatically and flexibly build UML diagrams at any level of detail starting with any compile unit, table, field.
ABL2UML

• Open source tool created in 2007 by Dr. Thomas Mercer-Hursh
• Takes schema from database and Bill of Materials from Analyst and builds UML component diagram which has
  • all code units down to internal procedures, function, method level (detail level)
  • all links to detail level
  • summary of compile unit connections
  • database tables and fields
  • connections between code units (including tables and fields)
  • how/when table/field is read, modified or written
  • all where clauses which connect code
• Contains diagram builder to automatically and flexibly build UML diagrams at any level of detail starting with any compile unit, table, field.
ABL2UML Status

Revision of tool currently under way.

Goals:
- Move to OO
- Incremental builds
- Use pieces separately, e.g., schema only
- Support non-OpenEdge databases
- Support .df alternative to direct schema

And more!
Agenda

- Introduction: What Is the Problem?
- What’s Going On in This Compile Unit?
- What’s Going On in This Body of Code?
- Where Is That? or What’s Different?
- How Does That Work Anyway?
- When It Runs, What Happens?
- How Do I Get Better Code Quality?
- Understanding The Big Picture

- Looking Forward
- Summary
PCT and Jenkins

At Exchange: Beginners Guide to Continuous Integration by Gilles Querret
Sonic Source

At Exchange: Buttonhole Gilles Querret at previously mentioned talk.
Agenda

- Introduction: What Is the Problem?
- What's Going On in This Compile Unit?
- What's Going On in This Body of Code?
- Where Is That? or What's Different?
- How Does That Work Anyway?
- When It Runs, What Happens?
- How Do I Get Better Code Quality?
- Understanding The Big Picture
- Looking Forward
- Summary
Summary

There are lots of tools available. Some built-in, some free open source, some commercial – most are inexpensive. Not knowing is far more expensive.

A small investment in tool building, purchase, and/or learning will pay dividends year after year.

Would you perform surgery with a blindfold?
Thank you.
Questions

For more information:
http://www.cintegrity.com
thomas@cintegrity.com
510-233-5400

And now for questions.
Slide Equivalents for Demos

Use of slides vs demos to be determined by timing later
Slide equivalent of demo.
Need some samples and more information from Mike.
Demo results from
Demo ProLint on laptop
## ProLint Results

<table>
<thead>
<tr>
<th>Issue</th>
<th>Description</th>
<th>Severity</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>Privacy group ASSIGN with line 102</td>
<td>error</td>
<td>102</td>
</tr>
<tr>
<td>76</td>
<td>K002 node RECT, Width was entered that if on line 7. Expected to be 4. is 2.</td>
<td>warning</td>
<td>76</td>
</tr>
<tr>
<td>103</td>
<td>Complete with fall on lines. Use only lower-case identifiers: supported/provided</td>
<td>warning</td>
<td>103</td>
</tr>
<tr>
<td>150</td>
<td>Complete with fall on lines. Use only lower-case identifiers: supported/provided</td>
<td>warning</td>
<td>150</td>
</tr>
<tr>
<td>27</td>
<td>Complete with fall on lines. Use only lower-case identifiers: supported/provided</td>
<td>warning</td>
<td>27</td>
</tr>
<tr>
<td>61</td>
<td>INPUT PARAMETER (CHARACTER scope LOCAL) getboolean should start with &quot;v&quot;</td>
<td>warning</td>
<td>61</td>
</tr>
<tr>
<td>36</td>
<td>CLOSE only is CLOSE is a Progress retained</td>
<td>warning</td>
<td>36</td>
</tr>
<tr>
<td>39</td>
<td>DN CLOSING scope LOCAL should start with &quot;v&quot;</td>
<td>warning</td>
<td>39</td>
</tr>
<tr>
<td>11</td>
<td>INPUT PARAMETER (CHARACTER scope LOCAL) getbooleanProTrans should start with &quot;v&quot;</td>
<td>warning</td>
<td>11</td>
</tr>
<tr>
<td>146</td>
<td>INPUT PARAMETER (charScope LOCAL) getbooleanProTrans should start with &quot;v&quot;</td>
<td>warning</td>
<td>146</td>
</tr>
<tr>
<td>106</td>
<td>INPUT PARAMETER (charScope LOCAL) getboolean should start with &quot;v&quot;</td>
<td>warning</td>
<td>106</td>
</tr>
<tr>
<td>20</td>
<td>fWriter scope MIN should start with &quot;v&quot;</td>
<td>warning</td>
<td>20</td>
</tr>
<tr>
<td>137</td>
<td>PROCEDURE fDateCreateProTrans scope LOCAL fDateCreateProTrans should start with &quot;v&quot;</td>
<td>warning</td>
<td>137</td>
</tr>
<tr>
<td>134</td>
<td>PROCEDURE fDateCreateProTrans scope LOCAL fDateCreateProTrans should start with &quot;v&quot;</td>
<td>warning</td>
<td>134</td>
</tr>
<tr>
<td>102</td>
<td>PROCEDURE fDateValidation scope LOCAL fDateValidation should start with &quot;v&quot;</td>
<td>warning</td>
<td>102</td>
</tr>
<tr>
<td>20</td>
<td>PROCEDURE fDateWrite scope LOCAL fDateWrite should start with &quot;v&quot;</td>
<td>warning</td>
<td>20</td>
</tr>
<tr>
<td>52</td>
<td>FUNCTION fGetAccessorsProcess scope LOCAL fGetAccessorsProcess should start with &quot;v&quot;</td>
<td>warning</td>
<td>52</td>
</tr>
<tr>
<td>57</td>
<td>PROCEDURE fGetAccessors scope LOCAL fGetAccessors should start with &quot;v&quot;</td>
<td>warning</td>
<td>57</td>
</tr>
<tr>
<td>51</td>
<td>PROCEDURE getEnabled scope LOCAL getEnabled should start with &quot;v&quot;</td>
<td>warning</td>
<td>51</td>
</tr>
<tr>
<td>53</td>
<td>PROCEDURE fGetAccessibility scope LOCAL fGetAccessibility should start with &quot;v&quot;</td>
<td>warning</td>
<td>53</td>
</tr>
<tr>
<td>107</td>
<td>SUBMAIN # AUTOBOOL (CHARACTER scope LOCAL) fShouldStartWith &quot;v&quot;</td>
<td>warning</td>
<td>107</td>
</tr>
</tbody>
</table>
Proparse Scripting

Sample calling program is simply:

using proparse-scripts.*.

def var finder as class FindIO no-undo.
run proparse/setup/setup.p.
finder = new FindIO("autoedge").
finder:ParseAllFiles().

FindIO is a simple 60 line ABL which checks for INPUT and OUTPUT verbs and creates report.

Demo scripting locally
How Do I Get Better Code Quality?

Proparse Scripting

Searches 209 programs under “autoedge” in about a minute and produces output like:

C:\Work\tmp\PPScript\autoedge\web\escript\src\web\objects\psagent.p
Line: 120
OUTPUT STREAM WebStream TO "WEB";U.

C:\Work\tmp\PPScript\autoedge\web\escript\src\web\objects\psagent.p
Line: 193
OUTPUT STREAM WebStream TO "WEB";U CONVERT TARGET WEB-CONTEXT:HTML-CHARSET.

C:\Work\tmp\PPScript\autoedge\web\escript\src\web\objects\spooler.p
Line: 111
/* Open the file for reading */
input from value (file-info:full-pathname) binary no-map no-convert.

Demo scripting locally
How Do I Get Better Code Quality?

Joanju Analyst (con’t)

Powerful query capability

**Joanju.Analyst**

**Search**

**Browse a compile unit**

Enter the name of a .p, .w, or .cls file on the propath.

Submit Query

**Server**

Browse a specific compile unit ...
How Do I Get Better Code Quality?

Joanju Analyst (con’t)

Source is fully pre-processed and “pretty printed” with collapsible includes, links to original source, and many other features. One can click on a run

do: output stream WebStream to "WEB";U.
   run_init-cgi in web-utilities-hdl.
   run_init-request in web-utilities-hdl.

   init-request in stateware.p
   init-request in devpath.p
   init-request in runlog.p
   init-request in web-util.p
   init-request in session.p

   "debug":U then "webutil/debug.p":U else ( if AppProgram eq
   puing "U then "webutil/ping.p":U else ( if AppProgram eq "reset":U then "webutil/reset.p":U else
   AppProgram )))

and see all possible destinations based on the value of web-utilities-hdl and can go to that source.
Joanju Analyst (con’t)

One can also click on the definition of an internal procedure or function and see a list of all places where that code is referenced and link to that source.

```plaintext
function devCheck returns logical ( );

devpath.p
psagent.p
```
Joanju Analyst (con’t)

One can also query on includes, table and field references, calls, and ABL syntax.

**Joanju.Analyst**

**Search**

Tables: customer - (fields: "customer.customerID")

Submit Query

**Browse a compile unit**

srcwebobjects.upsagent.p

Enter the name of a .p, .x, or .class file on the path.

Submit Query

**Server**

Build Manager | Administrative Page | Log file
ABL2UML
Sample showing links between IPs, functions, and compile units.
ABL2UML
Sample showing links between IPs, functions, and compile units to a table.
Consider whether to update the following slides and add after Summary or just to go with the per slide links.
Here are some links for more information. Generally, look at OpenEdge Hive, Joanju.com, and my own website, Cintegrity.com.
For More Information, go to…

**Where Is That? or What’s Different?**

- **OpenEdge Architect Search** – Built-in to OpenEdge Architect
- **Non-ABL Specific Search and Index Tools** – many available
- **substitute** – scripts available on request

**How Does That Work Anyway?**

Here are some links for more information.