Dump & Load: More than just a few Proutil Commands
Paul Koufalis, White Star Software

Experience is a hard teacher because she gives the test first, the lesson afterward. - Vernon Law

Presentations on dump&load typically revolve around cool methods to get the data exported and imported in a ridiculously short period of time. The suspense! The excitement! Can he do it!?! The reality is that the dump&load process is far more than just dump+load+idxbuild. There are a hundred little bits and pieces and if you mess up even one of them, Monday morning is going to be one of those experience-building opportunities.

This session will present a full D&L methodology: analysis, documentation, scripting and automation, verification and testing. With the information from this presentation, you can spend Monday morning accepting thank yous and congratulations rather than...gaining experience.
Dump & Load
More Than Just a Few Proutil Commands

Paul Koufalis, White Star Software
pk@wss.com
Why this D&L Presentation?

- Good Fast Cheap: pick two
- DBA projects fail because people choose fast and cheap
- This session is about *GOOD*: do it right, the first time

- Memorize this:

  *Go-live is not a time to think. It’s a time to do. All your thinking should have been done beforehand.*
Before we start

• A little context...
Who is White Star Software?

- The oldest and most respected independent DBA consulting firm in the world
- Five of the world’s top OpenEdge DBAs
- Author of ProTop, the #1 FREE OpenEdge Database Monitoring Tool
  - [http://protop.wss.com](http://protop.wss.com)
Paul Koufalis

• Progress DBA and UNIX admin since 1994
• Expert OpenEdge technical consulting
• Wide range of experience
• Small 10 person offices to 3500+ concurrent users
• AIX, HPUX, Linux, Windows...if Progress runs on it, I’ve worked on it
• Father to these two monkeys

• pk@wss.com
Think

- Gather data
- Analyse
- Document
- Test
- Execute
- Verify, verify and verify again
Step 0: Rules of Engagement

- Server change?
- Opsys change? New version? New Opsys?
- OpenEdge change?
- Codepage change (ex.: iso8859-1 to utf-8)?
- Rough available downtime: A few hours or long weekend?
Sanity Check

• Check that all the pieces play nice
• New OE version vs old opsys version (happened to me last week)
• 32 vs 64 bit
• Available disk space (at least 4X DB size)
• Downtime: the more you have, the less you need to test/prepare
Step 1: Collect Data

- ServerName, DBName, Full Path, DB Size on Disk, File Owner, Broker Process Owner, codepage
- If probkup –com, note backup size
- Permissions: check all DB files as often there are inconsistencies
- Physical parameters
- Start-up parameters
Physical DB Data

- DB, AI, BI Block size
- BI Cluster Size
- Enabled features (proutil db –C describe)
- Create/toss limits
- Generate full DB analysis (proutil db –C dbanalys)
  - More on this later
- Note anything unusual
  - Auditing, TDE, table partitioning...
Parameter Data

- All start-up parameters for all brokers for all databases
- All the *stuff* in Admin – Dump
  - User tables, security domains...
  - Note existence
- SQL data: views, permissions, stored procedures...
  - Note existence
Table & Index Statistics

• Start gathering detailed CRUD data
  – You may need to adjust –tablerangesize and –indexrangesize first
• Hourly stats
• Included with ProTop ($$)
Step 2: Analyse

- **Security**: What are the implications if you change something?
  - Are you still using root everywhere?
  - Are all the DB files rw-rw-rw?
- **Codepage**: are there characters that won’t convert?
- **Physical DB**:
  - 1 KB DB block size to 8 KB
  - BI Cluster size change?
Analyse – DB Analysis

• Check for long RM chains
  – May imply adjusting create/toss limits

• Check DB size as reported by DB Analysis vs physical DB size
  – You think you have to D&L 1 TB but really only 300 GB

• Check Type 1/2 storage area structure
  – If type 1, migrate to type 2 as part of D&L
  – If already type 2, can you improve?
Analyse – DB Analysis

• New type 2 storage area structure
  – A presentation of it’s own (sorry!)
  – Many ways to do this well
• Generally:
  – segregate by object type (table, index, LOB)
  – “Large” tables get their own storage areas
  – Every table area has a corresponding index area
Type 2 SA

- We have automated most of the analysis and execution
  - ST file creation, DF conversion, etc.
Analyse – CRUD Statistics

• There are usually a handful of tables/indexes that dominate CRUD activity
• Consider dumping using most-used index
  – Dump may be slower, but post-D&L performance will be better

• Some examples...
Examples – CRUD Statistics
Examples – CRUD Statistics

- Purge index
Examples – CRUD Statistics

- Table with 300 records, 100K-200K table scans per SECOND all day
- Fix code, of course, and move to -B2
## Analysis – Start-up Parameters

### Health Check Findings Summary

<table>
<thead>
<tr>
<th>Severity</th>
<th>Category</th>
<th>Issue</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>Opsys</td>
<td>syncIO &gt;= 9</td>
</tr>
<tr>
<td>Medium</td>
<td>Monitoring</td>
<td>excessidx &lt;= 10</td>
</tr>
<tr>
<td>Medium</td>
<td>Monitoring</td>
<td>excessTbl &lt;= 10</td>
</tr>
<tr>
<td>Medium</td>
<td>Performance</td>
<td>dbSpin &gt; 20000</td>
</tr>
<tr>
<td>Medium</td>
<td>Performance</td>
<td>IruSkips = 0</td>
</tr>
<tr>
<td>Medium</td>
<td>Performance</td>
<td>pfDelay &lt;&gt; Enabled</td>
</tr>
<tr>
<td>Medium</td>
<td>Performance</td>
<td>pfNumReqs &lt; 100</td>
</tr>
<tr>
<td>Medium</td>
<td>DB Analysis</td>
<td>dbaAge = 0</td>
</tr>
<tr>
<td>Medium</td>
<td>Opsys</td>
<td>loResponse &gt;= 10</td>
</tr>
<tr>
<td>Low</td>
<td>Recovery</td>
<td>oeReplication &lt;&gt; Enabled</td>
</tr>
<tr>
<td>Low</td>
<td>Monitoring</td>
<td>idxRange = 50</td>
</tr>
<tr>
<td>Low</td>
<td>Monitoring</td>
<td>tblRange = 50</td>
</tr>
<tr>
<td>Low</td>
<td>Performance</td>
<td>aBuffers &lt; 50</td>
</tr>
<tr>
<td>Low</td>
<td>Performance</td>
<td>bBuffers &lt; 50</td>
</tr>
<tr>
<td>Low</td>
<td>Performance</td>
<td>pica &lt; 98</td>
</tr>
<tr>
<td>Low</td>
<td>DB Analysis</td>
<td>fragPct = 1%</td>
</tr>
<tr>
<td>Low</td>
<td>DB Analysis</td>
<td>ixUtil &lt; 70%</td>
</tr>
<tr>
<td>Low</td>
<td>DB Analysis</td>
<td>rmChain &gt; 1000</td>
</tr>
<tr>
<td>Low</td>
<td>Best Practices</td>
<td>con_APW = 0</td>
</tr>
<tr>
<td>Low</td>
<td>Best Practices</td>
<td>con_BIW = 0</td>
</tr>
<tr>
<td>Low</td>
<td>Best Practices</td>
<td>con_WDOG = 0</td>
</tr>
<tr>
<td>Info</td>
<td>Recovery</td>
<td>bThreshold &gt;= 0</td>
</tr>
<tr>
<td>Info</td>
<td>Other</td>
<td>con_total &gt; 0</td>
</tr>
</tbody>
</table>
Documentation Check

• At this point, you should have numerous new documents
1. List of databases, sizes, etc.
2. Start-up parameters
3. Physical parameters
4. New type 2 structure (st and df)
5. CRUD remarks
• You have done 75% of the work without dumping a single record
Step 3: Test

- Three phases: dump, load, index rebuild
- The next few slides are how I D&L 99% of the time
Dump

• Simple to automate
• Generally

for each _file no-lock where _tbl-type = "t":
    put unformatted "proutil sports -C dump " +
        _file-name + " " + vDirName skip.
end.
Example Auto-Generated Dump Script

```bash
export DLC=/opt/dlc
export DUMPPDIR=/flashmount/dumpload/dump/dbuat_bc
export LOGDIR=/flashmount/dumpload/log/dbuat_bc
export DB=/dbuat/bc/bc
export PATH=$DLC:$DLC/bin:$PATH
mkdir -p $DUMPPDIR > /dev/null 2>&1
mkdir -p $LOGDIR > /dev/null 2>&1
cd $DUMPPDIR

_proutil $DB -C holder > /dev/null 2>&1
if [[ $? = 16 ]]; then BG='&';fi
#ECHO=echo

####### LARGE TABLES BACKGROUND DUMP #######
$ECHO eval _proutil $DB -C dump aaa $DUMPPDIR -thread 1 -threadnum 8 > $LOGDIR/aaa.bindump.out 2>&1 $BG
$ECHO eval _proutil $DB -C dump bbb $DUMPPDIR -thread 1 -threadnum 8 > $LOGDIR/bbb.bindump.out 2>&1 $BG
$ECHO eval _proutil $DB -C dump ccc $DUMPPDIR -thread 1 -threadnum 8 > $LOGDIR/ccc.bindump.out 2>&1 $BG

####### SMALL TABLES SERIAL DUMP #######
$ECHO eval _proutil $DB -C dump ddd $DUMPPDIR -thread 0 > $LOGDIR/ddd.bindump.out 2>&1
$ECHO eval _proutil $DB -C dump eee $DUMPPDIR -thread 0 > $LOGDIR/eee.bindump.out 2>&1
$ECHO eval _proutil $DB -C dump fff $DUMPPDIR -thread 0 > $LOGDIR/fff.bindump.out 2>&1
```
Testing Your Dump

• I like to use “ECHO=echo” in my scripts
  – $ECHO _proutil DB –C dump ...
  – Script only echoes command until I comment out ECHO=echo

• Each command outputs to its own log file

• Typically after 10-20 minutes, there are only 2-3 dumps left

• If parallel “big” dumps finish before series dumps, add more tables to parallel section
Testing Your Dump

• Start the DB with a large –B
  – Enough to hold all the large indexes being used plus more
• If dumping from type 1 SA, use –B = DB size if you can

• Type 1 SA dump recent example:
  – 35 GB DB
  – -B = 8 GB RAM: 3 hr to dump
  – -B = 40 GB RAM: 1 hr to dump
Testing Your Dump

• -n: big enough for all parallel dumps
  – Watch out for multi-thread
• -lru:skips 1000: Haven’t benchmarked, but lru chain maintenance probably more harm than good during dump
  – At least 200

• Try and write to non-DB disks
  – Why compete on read AND write?
Testing Your Dump

• Keep a detailed log of each test
• How long before only 3-4 dumps left?
• How long to dump the last 3-4 tables?
## Example:

<table>
<thead>
<tr>
<th>Test #</th>
<th>Server</th>
<th>Env</th>
<th>DB</th>
<th>Size (GB)</th>
<th>Dump Time (min)</th>
<th>Notes</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>volta</td>
<td>/dbuat</td>
<td>bc</td>
<td>45</td>
<td></td>
<td>12:52 - Started</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>cc</td>
<td>18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>cf</td>
<td>19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>co</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>dr</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>gl</td>
<td>32</td>
<td></td>
<td>Started gl with -B 1M (8 GB)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>gs</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>st</td>
<td>100</td>
<td></td>
<td>Started st with -B 6M (48 GB)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>tmq-fxe</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>histo-edi</td>
<td>35</td>
<td></td>
<td>Started histo-edi with -B 5M (40 GB)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DUMPING FROM SCHEMA AREA !!</td>
<td></td>
</tr>
</tbody>
</table>

13:39: Only 3 tbls left

Start time

10 DBs to dump
Testing Load

• First: create your “empty” databases
  – New structure, new df, correct physical parameters
• Backup to dbname.DFONLY.bkp
  – You will use it numerous times

• Ignore the first load test
  – Will waste time expanding the variable-length extents
Testing Load

- 99.99% of the time, I do single-threaded, -i load with the DB down
- No APW/BIW/nothing
- I don’t even bother comparing multi-user load anymore
- Occasionally there is a special case, but very rare

`proutil DB -C load table.bd -i`
Index Rebuild

- Highly dependant on availability of –TF parameter
  - 10.2B04+, 11.x
- Without –TF, need to resort to RAM drive to avoid disk sort
- With –TF, should be zero disk writes for sorting if enough RAM
- Example command (*threads params CPU dependent):

  _proutil DB -C idxbuild all -B 512 -TB 64 -TM 32 -TMB 1024 -TF 80 -SG 64 -thread 1 -threadnum 8 -z -rusage -mergethreads 4 -datascanthreads 8
Index Rebuild - Example

• For the 10 databases in the previous slides, full idxbuild of all 10 in less than 2h
  – Largest DB 100 GB
• Total D&L time: less than 5h
  – Without really trying very hard
Maintenance Documentation

• Remember: *Go-live is not a time to think. It’s a time to do. All your thinking should have been done beforehand.*

• I use two tools:
  – Detailed maintenance plan: Word document format
  – Task Checklist: Online Google Sheet for collaboration
Maintenance Plan

• Four sections:
  – Pre-Maintenance
  – Maintenance
  – Post-Maintenance
  – Rollback
Pre-Maintenance

- All the things you need to do in the days leading up to go-live
- Dictionary freeze date
- Last minute compare of DF
  - You may need to generate new “empty” databases and dump scripts
- Mount temporary resources
  - Fast SSD disks
  - Extra RAM/CPU
Maintenance

• All the actual D&L steps, in detail
• Include example command lines
• This is the “Don’t think. Do.” section
Checklist

- We have about 30 items (columns) on our checklist
- One line per DB
- Simplifies working with a colleague
- Each cell is “Not started”, “On hold”, “In Progress”, “Completed” or “ERROR”
Post-Maintenance

• All the verifications – make sure you didn’t miss anything
  – More later
• GO/NO-GO user testing
• Turn on all the other *stuff*
  – Cron jobs, batch processing, EDI...
• Release to production
Rollback Procedure

• Easiest: new databases on new server or at least new disks
• Rollback = swap old and new

• Almost always as part of GO/NO-GO
  – Very difficult to rollback after prod has started working
  – I’ve done it. Wasn’t pretty.
Step 4: Execute

- This part is easy because you’re ready
- Print your maintenance plan and mark it “Execution Copy”
  - Take notes on the physical copy
- Use your checklist
Step 5: Verify, verify, verify

- Check log files for errors
  - Automate
- Check before-and-after record counts
  - Table by table
- Check DB Analysis for stupid errors
  - Objects in schema area
  - Tables in index areas or vice-versa
  - LOB placement
Post Maintenance Verification

• Double-check AI/BI block size and cluster size
  – Other params, startup PF files, etc.
• Double-check checklist
  – After 8-10 hours, easy to miss something
• Double-check permissions
• Something always goes wrong during the maintenance
  – Go back and double-check everything related to what went wrong
  – One time we had a SAN crash during idxbuilds and we ended up missing ONE index rebuild on ONE small (but important) table
Post-Maintenance Monitoring

• Collect trend data to compare before and after
Bask in the Light of Your Awesomeness

We are the best at what we do.

White Star Software
Questions?
#1 OpenEdge Database Monitoring Tool
http://protop.wss.com