No More AIxcuses

PAUL KOUFALIS
PRESIDENT
PROGRESSWIZ CONSULTING

Progresswiz Consulting

- Based in Montréal, Québec, Canada
- Providing technical consulting in Progress®, UNIX, Windows, MFG/PRO and more
- Specialized in
  - Security of Progress-based systems
  - Performance tuning
  - System availability
  - Business continuity planning

Progresswiz

Agenda

- Introduction
- After-Imaging (AI) Basics
- Pre-OpenEdge Implementation Challenges
- The AI File Management Daemon
- Step-by-Step Implementation
- Understanding the Results
- A Few Final Steps
- Questions

Introduction

- What is After-Imaging?
  - Quick Definition:

    A logging system that stores all information required to reapply all changes made to a database between two points in time

Introduction

OpenEdge AI Gremlins Hard at Work!

Introduction

- Recipe for Recovery
  - One backup
  - All AI files generated since that backup

- Result
  - One fully recovered database with near-zero data loss

Introduction

- Many businesses only backup once per day
  - Without AI, any changes made since the last backup are lost
- Best of all, it's 100% free
  - Both Workgroup and Enterprise
  - And Personal DB too!

Progresswiz

After-Imaging (AI) Basics

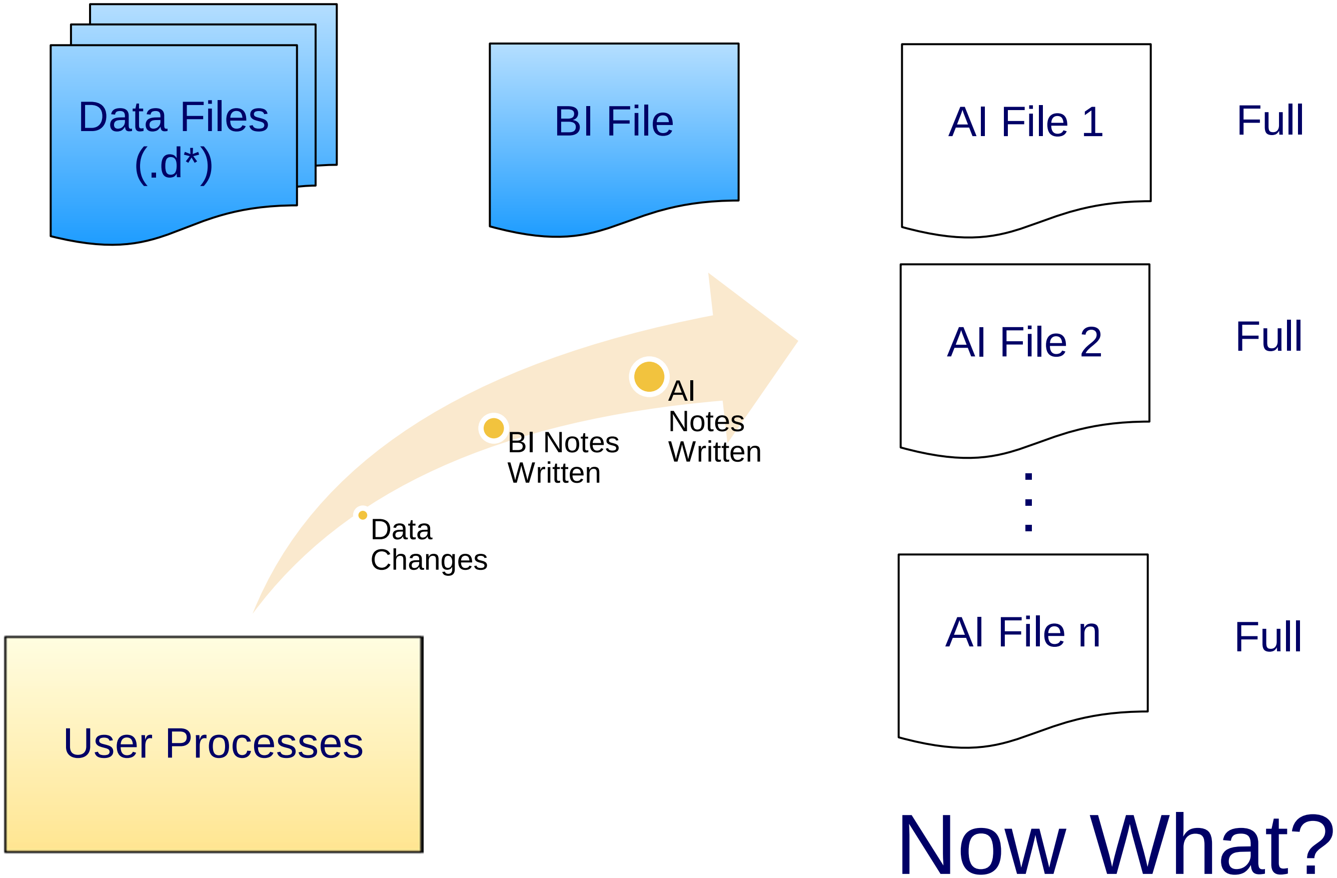
- Basic transaction process:
  1. Begin Transaction
     a) Make some change
     b) Before and after value written to BI
     c) Before and after value written to AI
     d) Repeat…
  2. End Transaction

- Note that I didn’t mention writes to the data files
  - They will happen eventually

After-Imaging (AI) Basics

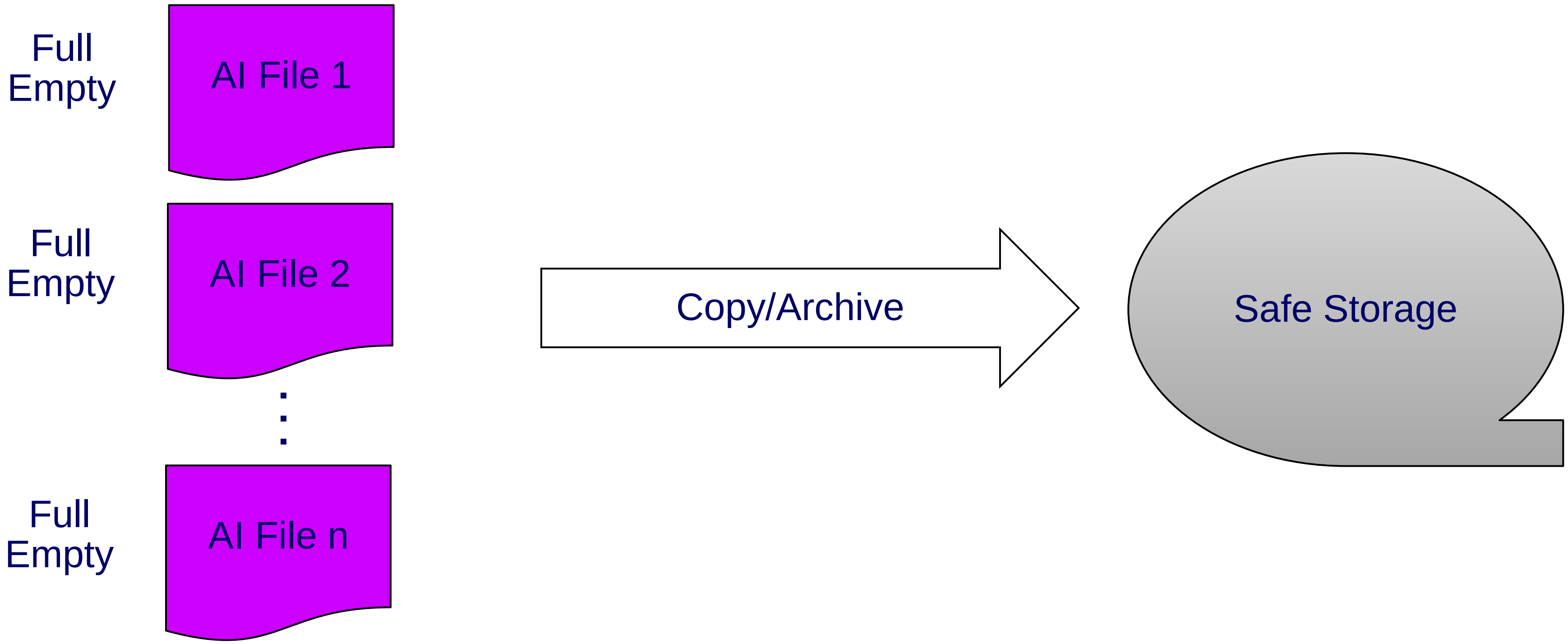
- What happens to all these BI/AI notes?
  - OpenEdge automatically manages the reuse of BI file space
  - AI files must be archived before allowing OpenEdge to reuse them

After-Imaging (AI) Basics



Now What?

After-Imaging (AI) Basics



After-Imaging (AI) Basics

- Sounds simple enough:
  - When full, copy/archive AI files to a safe location
  - Mark them as empty so OpenEdge can reuse the space in the files
- What's the problem?

Progresswiz

Pre-OpenEdge Implementation Challenges

- Problem #1: Scripting

- Commands are simple enough:

```
rfutil sports –C aimage extent full
rfutil sports -C aimage query extnum by name sports.a4
cp sports.a4 /archive/sports.${extnum}
rfutil <dbname> -C aimage empty
```

- But adding the logic around them requires work

Pre-OpenEdge Implementation Challenges

- Scripts to automate AI file management can be as simple or complex as you like
  - Depends how “smart” you want them to be

- Not that difficult to write “dumb” scripts
  - Plenty of example scripts available online
  - Search the peg (www.peg.com)

Pre-OpenEdge Implementation Challenges

- Problem #2: Performance Impact

- Ye Olden Days of Yore…yes
- Today: Please…
  - The effect is minimal unless you have really old junky hardware
  - Of course, there are exceptions
    - But if you're that busy you *really* need AI!

Pre-OpenEdge Implementation Challenges

- Resist the “If it ain’t broke…” mentality
  - If you are not running AI your system **IS** broken

- With modern hardware the performance issue is moot
- And now with the AI File Management Daemon, so is the scripting issue

The New AI File Management Daemon

- Available as of 10.1A

- Automates:
  - Rotation and reuse of AI files
  - Archiving of full extents
  - The <ahem> *intelligent* <ahem> naming of archived AI extents
    - More on this later

The New AI File Management Daemon

- Can be activated in one of three modes:
  - Time interval: AI file rotation occurs every x seconds
    - Min: 2 minutes
    - Max: 24 hours
  - Full AI File: Rotation occurs when the currently active AI file is filled
    - Used with fixed length AI extents
  - Both
    - This is the preferred method

The New AI File Management Daemon

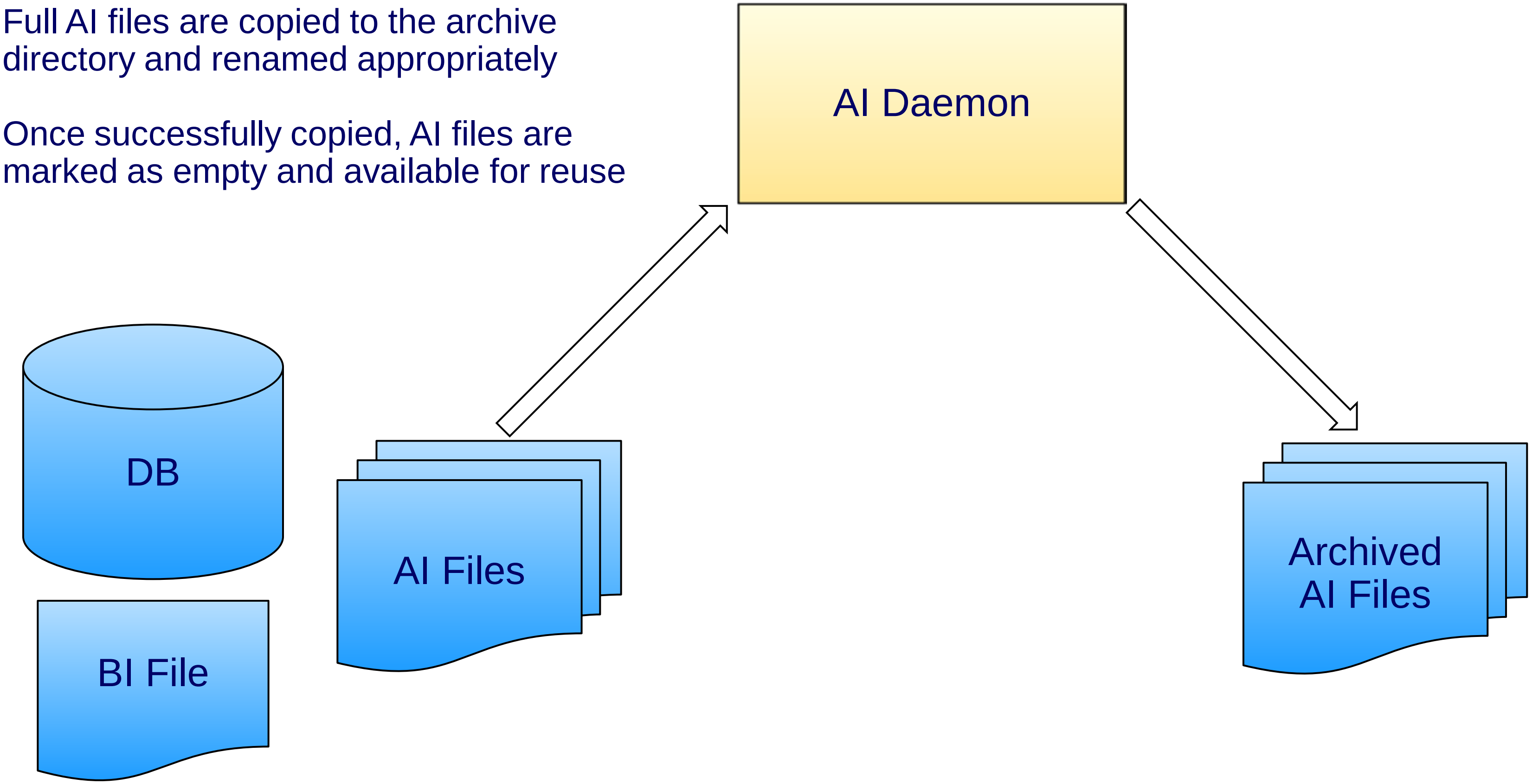
- Two configurable parameters:
  - -aiarcdir: Directory where full AI extents will be copied
    - Can specify multiple directories in case writes to the first fail
  - -aiarcinterval: Time between AI file rotations
    - If not specified, rotation occurs when extent fills

How it Works

1. Daemon wakes up and checks for full AI files or elapsed aiarcinterval

2. Full AI files are copied to the archive directory and renamed appropriately

3. Once successfully copied, AI files are marked as empty and available for reuse



Progresswiz

Step-by-Step Implementation

- Four ways to activate AI and AI File Mgmt:

After-Imaging	AI File Management
OFFLINE	OFFLINE
ONLINE	OFFLINE
OFFLINE	ONLINE
ONLINE	ONLINE

- Note: Online activation as of 10.1B

Progresswiz

Add AI Extents

- Can be done offline or online as of 10.1A
- Create add.st file

```
# Three variable length AI files in /db/prod
a /db/prod
a /db/prod
a /db/prod
```

- Apply to database

```
$ prostrct add sports add.st
$ prostrct addonline sports add.st
```

- Addonline could be tough in a secure, shared memory environment

Progresswiz

Enable AI & AI Mgmt Daemon Offline

- Take a full backup of your database

```
$ probkup sports /backup/sports.probkp
```

- Enable after-imaging

```
$ rfutil sports –C aimage begin
```

- Enable AI Management Daemon

```
$ rfutil sports –C aiarchiver enable
```

Progresswiz

Enable AI & AI Mgmt Daemon Offline

- Start your database
  - Be sure to include the –aiarcdir and, if required, -aiarcinteval startup parameters

```
$ proserve sports –pf sports.pf
                  –aiarcdir /aiarch,/aiarch2
                  -aiarcinterval 900
```

Enable AI & AI Mgmt Daemon Online

- Enabling AI & the AI File Management Daemon *offline* required three distinct steps:
  - Backup
  - Enable AI
  - Enable AI File Management
- As of 10.1B, do it *online* all in one command

Enable AI & AI Mgmt Daemon Online

```
$ probkup online sports/backup/sports.probkp
                 enableai
                 enableaiarchiver
                 -aiarcdir /aiarch,/aiarch2
                 -aiarcinterval 900
```

Mixing Online and Offline

- Parameters *enableai* and *enableaiarchiver* are optional
- Can use one or other in probkup online cmd
  - Of course, AI must be enabled in order to enable AI Archiver

Mixing Online and Offline

- Example:
  - Backup offline

    ```
    $ probkup sports /backup/sports.probkp
    ```

  - Enable AI offline

    ```
    $ rfutil sports –C aimage begin
    ```

  - Start DB

    ```
    $ proserve sports –pf sports.pf
    ```

  - Enable AI Archiver online

    ```
    $ probkup online sports/backup/sports.probkp
                  enableaiarchiver
                  -aiarcdir /aiarch,/aiarch2
                  -aiarcinterval 900
    ```

Progresswiz

Stopping the AI Archiver Daemon

- To permanently disable the AI Archiver

```
$ rfutil sports –C aiarchiver disable
```

- To stop the daemon temporarily without disabling the functionality
  - I.e. for maintenance

```
$ rfutil sports –C aiarchiver end
```

Starting the AI Archiver Daemon

- To restart the AI Archiver

```
$ _mprshut sports -C aimgt –aiarcdir <dir>
                           -aiarcinterval <n sec>
```

  - There is no “rfutil db –C aiarchiver begin”

- Was not documented until 10.2B docs

Starting the AI Archiver Daemon

- Note the similarity to other helper processes
  - APW, AIW, BIW and WDOG

```
root 2906 1  0 00:40 pts/1 00:00:00 _mprshut atm -C apw
root 2907 1  0 00:40 pts/1 00:00:00 _mprshut atm -C biw
root 2910 1  0 00:40 pts/1 00:00:00 _mprshut -db atm -C aimgt
root 2929 1  0 00:40 pts/0 00:00:00 _mprshut atm -C aiw
```

AI Archiver Status

- Activated?
  - Use “proutil <dbname> -C describe

```
Database Features

   ID   Feature                            Active  Details
  ----  ---------------------------------  ------  -------
     8  After Image Mangement/Archiver     Yes
     9  64 Bit DBKEYS                      Yes
    10  Large Keys                         Yes
    11  64 Bit Sequences                   Yes
```

AI Archiver Status

- Running?
  - Look in promon – R&D – 1 – 4 – 1

```
10/22/08        Status: All Processes
7:17:24

 Usr Name     Type   Wait  Trans id       Login time

   0 Paul Kou BROK   --          0    0  10/22/08 17:09
   5          AIMD   --          0    0  10/22/08 17:09
   6 Paul Kou MON    --          0    0  10/22/08 17:16
```

Changing Parameters Online

- Both parameters (-aiarcdir and –aiarcinterval) can be changed online

```
$ rfutil sports –C aiarchiver setdir /aiarchnew

$ rfutil sports –C aiarchiver setinterval 1800
```

- The second option is particularly useful
  - Decrease interval during high activity
  - Increase interval during quiet times

Understanding the Results

- Enabling AI Archiver affects three output streams:
  - DB Log File
  - AI Archive output directory (-aiarcdir)
  - AI Archiver log
    - New

Database Log File

- AI Archiver entries show up as “AIMGT”
- Example:

```
AIMGT  5: (13213) A new archive interval 120 has been set.
```

Archived AI Files

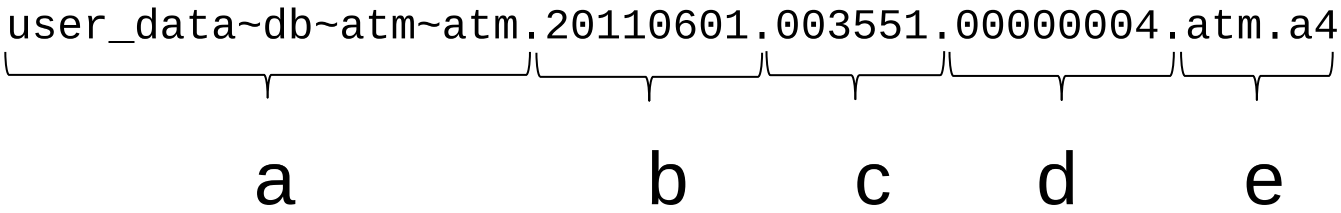
- The good:
  - All the information you need is stored in the file name
- The bad:
  - All the information you **don’t** need is stored in the file name
- The ugly:

```
user_data~db~atm~atm.20110601.003551.00000004.atm.a4
```

  - Now that’s a file name!

Progresswiz

Archived AI Files



```
user_data~db~atm~atm.20110601.003551.00000004.atm.a4
```

a) The full file path of the source DB
   - /user_data/db/atm/atm.db

b) The date of the “aimage begin” backup

c) The time of the “aimage begin” backup

d) The AI sequence number

e) The original AI file name

Archived AI Files

- Careful with the file naming convention
  - It has changed from 10.1 to 10.2
- Old version had the date and time of the ***last*** backup in the filename
  - Was confusing

AI Archiver Log

- AI Archiver creates it’s own log:
  - <DBNAME>.archival.log

- Whoa!!
  - A little bit challenging to read

AI Archiver Log

- First field is entry type:
  - 0001 = Automatic AI file archive
  - 0003 = Manual AI file archive
  - 0032 = Full backup
  - 0033 = Incremental backup

AI Archiver Log

```
0001,/user_data/db/atm/atm,20110601,011325,20110601,010645,1,/user_data
   /db/atm/atm.a1,/user_data/ai_arch/user_data~db~atm~atm.20110601.0106
   45.00000001.atm.a1
0001,/user_data/db/atm/atm,20110601,011830,20110601,010645,2,/user_data
   /db/atm/atm.a2,/user_data/ai_arch/user_data~db~atm~atm.20110601.0106
   45.00000002.atm.a2
0032,/user_data/db/atm/atm,20110601,011952,1,5,20110601,011321,3,BACKUP
   _SET_20110601_011321,1,/dev/null
0001,/user_data/db/atm/atm,20110601,012111,20110601,010645,3,/user_data
   /db/atm/atm.a3,/user_data/ai_arch/user_data~db~atm~atm.20110601.0106
   45.00000003.atm.a3
0001,/user_data/db/atm/atm,20110601,012616,20110601,010645,4,/user_data
   /db/atm/atm.a4,/user_data/ai_arch/user_data~db~atm~atm.20110601.0106
   45.00000004.atm.a4
```

DB Log

- Compare to DB log file:

```
BACKUP105: (1362)  Full backup started.

BACKUP105: (3777)  Switched to ai extent /user_data/db/atm/atm.a2.

BACKUP105: (3778)  This is after-image file number 2 since the last
                   AIMAGE BEGIN

AIMGT 104: (3777)  Switched to ai extent /user_data/db/atm/atm.a3.

AIMGT 104: (3778)  This is after-image file number 3 since the last
                   AIMAGE BEGIN
```

Careful…

- BACKUP105 switched to AI #2

```
BACKUP105: (3777)  Switched to ai extent /user_data/db/atm/atm.a2.
```

- #2 shows up ***before*** the backup in the archival log
- But it contains data from ***after*** the backup snapshot

Careful…continued…

- UNC pathnames in -aiarcdir
  - I.e “-aiarcdir \\bkpserver\aidir”
  - There was a bug on Windows until 10.1C
  - Not sure if fixed yet

A Few Final Steps

- AI Archive Directory
  - Ideally, NFS mount from another server
    - NOT in the same building as production
  - Delete files older than x days
    - 3 is good starting number
    - You will have to script this yourself

A Few Final Steps

- Monitoring
  - Periodically check that the AIFMD is up:

    $ ps –ef | grep aimgt

  - Check for various errors in the log file
  - Recently lived through issue with NFS mount
    - AIFMD could not write to mount (perms)
      - See ppt notes for error text
    - Failed rather than writing to secondary arc dir

A Few Final Steps

- Log File Maintenance
  - <Dbname>.archival.log
    - Archive and truncate periodically

Restore and Roll Forward

- Restore backup

```
$ prorest <DB> <bkp file>
```

- Copy AI files to roll forward to a temporary directory

- Use db log or archival log to determine first AI file to apply
  - Or just try one at random – the error msg will tell you the correct sequence to apply first

Restore and Roll Forward

- Roll forward AI files

```
for file in $(ls $TMP_AI_DIR)
do rfutil $DB –C roll forward –a $file
done
```

- File names ensure correct order:

```
 $ ls
 user_data~db~atm~atm.20110601.003551.00000003.atm.a3
 user_data~db~atm~atm.20110601.003551.00000004.atm.a4
 user_data~db~atm~atm.20110601.003551.00000005.atm.a1
 user_data~db~atm~atm.20110601.003551.00000006.atm.a2
 user_data~db~atm~atm.20110601.003551.00000007.atm.a3
 user_data~db~atm~atm.20110601.003551.00000008.atm.a4
```

That’s It!

- You’re done! Congratulations!
- Restores and roll-forwards are easy
  - AI Files are semi-intelligently named
  - AI Archiver log file shows which files go with which backup
- Just remember to cleanup AI Archive directory

Progresswiz

Questions

Questions?

Progresswiz

Progresswiz Consulting

- Questions or comments? Send me an email:
  pk@progresswiz.com

Progresswiz