Technical Challenges:
TIFF Image Scanning and Retrieval using Forms

Coleman Leviter
NYOUG
June 6, 2006
Presentation Goals

- Review TIFF Project
- Overcoming Obstacles outside Oracle
- Overcoming Obstacles within Oracle
Project History

- Digital Designer - Norden Systems
- Software Engineer - Grumman F-14D Tomcat
- RWR Software Engineer - Litton Systems
- VAX Programmer - Financial District
- WMS Development - Arrow Electronics
Age of the hardware may not be an indicator of its functionality
Digital Alarm Clock - circa 1974
Still Works
Fujitsu Scanner M3096E - Circa 1988
Still Works
Lexmark T630 - Circa 2002
Has Bugs
Moral of System Integration

- It is not always software causing a problem
- Sometimes, it is indeed the hardware
- Break the system down into its constituent parts and test
Image Scanning and Retrieval

PC Print Request       Oracle Server (AIX)       Scanner

Printer                      Scanner

PC Scan Request
Background

• Why TIFF Files?
  - Certificate of Compliance (paperwork) ships with special products
  - Images stored on host system, users do not have to retrieve hard copy
  - TIFF files print whenever product is shipped from warehouse
Typical TIFF Image
Previous System

- ALPHA (DIGITAL/COMPAQ) hosted images on file system -
  - Images scanned using Fujitsu Scanner M3096E with Kofax Board (use on new system)
- Images stored with VMS file system - DECFORMS
- Indexed file contained image details
  - number of images, archived, tape information
New Host System

- Forms 6.0 (Form Compiler) Version 6.0.8.25.2 (Production) Patch 16
- Patch 16 Required for reading TIFF files using Forms 6i - (Originally using Patch 10)
- AIX 5L Version 5.2
- Oracle9i Enterprise Edition Release 9.2.0.5.0 - 64bit Production
**Migration Path**

- Transfer almost 877,000 images from ALPHA to AIX
- File sizes range from 50k to 150k
  - \(877k \times 75k = 65.8 \text{ GB disk required}\)
- Avg one sec file store time into ORACLE
- Retrieve images from 9 track tapes
  - 10 tapes, 3 hours average, 14,000 images
TIFF File Format - Background

- Motorola Format (4D4D) identification in first two bytes signify Big Endian
- Intel Format (4949) identification in first two bytes signify Little Endian
- All files on previous host system (ALPHA) stored as Intel Format
- Preserve format in ORACLE
TIFF File Storage on AIX

• Use ORACLE for TIFF File storage vs. AIX file system
  - Reliable and Proven
  - Backups reliable
  - Critical Application

• Remote Oracle Table
  - Isolate image backups and restoration from disaster recovery
Forms 6i Implementation

- Forms 6i on AIX implementation character mode
  - existing user base consisting of WYSE VT 420 (Dumb Terminals)
  - maintain existing form, fit and function with user community
- Existing Application
  - scan images with product, print images out
WYSE VT-420 Dumb Terminal
Forms Built Ins

- TIFF File management plan with Forms
  - Use built in read_image_file to store images to Oracle
  - Use built in write_image_file to retrieve images to AIX file system and queue to T630 Lexmark printer with ImageQuick Card
  - Goal - Rapid Application Development
LOBs and Remote Tables

- Reading LOBs from Oracle using write_image_file

- Error Message
  - ORA-22992 cannot use LOB locators selected from remote tables
  - Cause: A remote LOB column cannot be referenced.
  - Action: Remove references to LOBs in remote tables.

- Solution
  - Global Temporary Tables
• Writing LOBs to Oracle using read_image_file

• Remember TIFF File Format? MOTO vs. INTEL?

• Oracle’s Built-in read_image_file converts 4949 to 4D4D
  – 4D4D002A0000DC MOTO
  – 49492A0008000000 INTEL
MOTO Problems in Forms

- Some images stored in Oracle using MOTO format caused Forms to crash when using write_image_file. (Segmentation Fault)

- INTEL Format using utility TIFFDUMP
  - Magic: 0x4949 <little-endian> Version: 0x2a
  - Directory 0: offset 8 (0x8) next 0 (0)
  - SubFileType (254) LONG (4) 1<0>
  - ImageWidth (256) LONG (4) 1<2544>
  - ImageLength (257) LONG (4) 1<3300>
  - BitsPerSample (258) SHORT (3) 1<1>
  - Compression (259) SHORT (3) 1<4>
  - Photometric (262) SHORT (3) 1<0>, )
  - Software (305) ASCII (2) 56<Kofax standard Multi-Pag . NORMAL>
MOTO Problems in Forms

- MOTO Format using utility TIFFDUMP
  - Magic: 0x4d4d <big-endian> Version: 0x2a
  - Directory 0: offset 8 (0x8) next 0 (0)
  - SubFileType (254) LONG (4) 1<0>
  - ImageWidth (256) LONG (4) 1<2544>
  - ImageLength (257) LONG (4) 1<3300>
  - BitsPerSample (258) SHORT (3) 1<1>
  - Compression (259) SHORT (3) 1<5>
  - Photometric (262) SHORT (3) 1<1>

- Additionally: Software (305) ASCII (2) 30<Oracle Multimedia Toolki ...> CHANGED BY read_image_file
• Syntax
  - PROCEDURE READ_IMAGE_FILE (file_name VARCHAR2,
    - file_type VARCHAR2,
    - item_id ITEM);
• Problem: modifies TIFF file when storing to Oracle
• Solution: ???
Create Stored Procedure `read_image_file_f`

- **Solution**: create stored procedure that reads images and does not modify them
  - directly uses global temporary table

- **Create Or Replace Function `read_blob_file_f`**
  - `INSERT INTO IMAGE_DETAIL_LOCAL_READ_GT (stoc_num, vndr_pord_num, vers_num, date_code, lot_code, page_num, img_data)
    VALUES (p_stoc_num, p_vndr_pord_num, p_vers_num, p_date_code, p_lot_code, p_page_num, EMPTY_BLOB())
    RETURNING img_data INTO dest_loc;
  - `INSERT INTO image_detail SELECT * FROM image_detail_local_read_gt;
  - `DELETE FROM image_detail_local_read_gt;
  - `RETURN lcl_success;`
Using Built In - TIFF Storage Into Oracle

- Read TIFF file from host system
  - Store Tiff file in Forms Block
    - read_image_file ( lineread, 'TIFF', 'image_detail_local_read.IMG_DATA');
    - COMMIT_FORM - stores TIFF file into global temporary table (image_detail_local_read), same name block.item
Using Built In - TIFF Storage Into Oracle

- Store TIFF file into Oracle
  - `ddl_text := 'INSERT INTO image_detail SELECT * FROM image_detail_local_read';`
  - `FORMS_DDL (ddl_text);`
- `image_detail` table contains all TIFF files
- `CREATE SYNONYM IMAGE_DETAIL FOR IMAGE_DETAIL@tiff_1`
TIFF Storage - Data Flow Without Stored Procedure

1. TIFF on AIX
   - Read_Image_File

2. TIFF in Block
   - Commit

3. TIFF in Global Temp Table

4. TIFF in Remote Table

Procedure:
1. Read_Image_File
2. TIFF in Block
3. TIFF in Global Temp Table
4. TIFF in Remote Table
   - Commit

Data Flow Without Stored Procedure
TIFF Storage - Data Flow Using Stored Procedure

1. TIFF on AIX → TIFF in Global Temp Table

2. NOTE: READ_BLOB_FILE_F bypasses Block

3. Commit

TIFF in Remote Table
TIFF Retrieval - Print Image

- Use built-in Write_Image_File to retrieve files from ORACLE
- FORMSDDL
  - retrieve tiff files from remote database
  - store files into global temporary table
- Go_Block
  - transfers TIFF files from global temporary table to Forms block
TIFF Retrieval - Print Image

- Built-in Write_Image_File
  - Transfers TIFF file from Forms block onto host file system
- Print File
  - use TIFF file name on local host system
TIFF Retrieval - Data Flow

1. TIFF in Remote Table
   - FORMS_DDL

2. TIFF in Global Temp Table

3. TIFF in Block
   - GO_BLOCK
   - WRITE_IMAGE_FILE

4. TIFF on AIX

Flow:
- 3: TIFF in Block
- 1: TIFF in Remote Table
- 4: TIFF on AIX
- 2: TIFF in Global Temp Table
Print TIFF Files from Forms 6i

- host ('enq -P <queue-name> ' || lcl_image_file, no_prompt );
Example T630 Formats

- T630 Formats with ImageQuick Card (Lexmark fixed a bug)
  - TIFF grayscale images (monochrome)
  - TIFF LZW (Lempel-Ziv-Welch)
  - TIFF CCITT Group 4
  - TIFF CCITT Group 3 2D
  - TIFF CCITT Group 3 ID
  - TIFF Type 2
  - TIFF Packbits
  - TIFF uncompressed
LEXMARK T630 Bug & Fix

• Send image file to printer
• Printer engine activated
• After one minute, printer engine stopped, no printout
• Image readable using Photo Editor
• LEXMARK supplied a patch that fixed the problem
Resolving Errors

• SEGMENTATION FAULT using Print Image from Forms

• Traced error to TIFF file in MOTO format
  – 4D4D002A0000DC    MOTO
  – 49492A0008000000    INTEL

• Solution - convert MOTO (4D4D) to INTEL (4949) format

• EASY 😞
Convert MOTO to INTEL

• Write program swapping bytes from 3rd byte to end of file

• Little Endian vs. Big Endian
  - "Little Endian" means that the low-order byte of the number is stored in memory at the lowest address, and the high-order byte at the highest address. (The little end comes first.) For example, a 4 byte LongInt
    - Byte3 Byte2 Byte1 Byte0
    - will be arranged in memory as follows:
      - Base Address+0 Byte0
      - Base Address+1 Byte1
      - Base Address+2 Byte2
      - Base Address+3 Byte3

• Intel processors (those used in PC's) use "Little Endian" byte order.
Convert MOTO to INTEL

• "Big Endian" means that the high-order byte of the number is stored in memory at the lowest address, and the low-order byte at the highest address. (The big end comes first.) Our LongInt, would then be stored as:

- Base Address+0  Byte3
- Base Address+1  Byte2
- Base Address+2  Byte1
- Base Address+3  Byte0

• Motorola processors (those used in Mac's) use "Big Endian" byte order.
Problem Swapping Bytes

- Keep track of check sums
- Modify first two bytes i.e. 4D4D to 4949
- Fixing bugs, etc...
- Any freeware?
- Search TIFF on GOOGLE
TIFFCP to the Rescue

- TIFFCP(1) USER COMMANDS TIFFCP(1)
- NAME tiffcp - copy (and possibly convert) a TIFF file
- SYNOPSIS tiffcp [ options ] src1.tif … srcN.tif dst.tif
- DESCRIPTION tiffcp combines one or more files created according to the Tag Image File Format, Revision 6.0 into a single TIFF file. Because the output file may be compressed using a different algorithm than the input files, tiffcp is most often used to convert between different compression schemes. By default, tiffcp will copy all the understood tags in a TIFF directory of an input file to the associated directory in the output file. tiffcp can be used to reorganize the storage characteristics of data in a file, but it is explicitly intended to not alter or convert the image data content in any way.

- Possibilities 😞 😊
Try TIFFCP on MOTO TIFF

- `tiffcp -c g3 "$tiff_old" "$tiff_new"

**MOTO Tiff File Format**

- TIFF Directory at offset 0x8
- Subfile Type: (0 = 0x0)
- Image Width: 2544 Image Length: 3264
- Resolution: 300, 300 pixels/inch
- Bits/Sample: 1
- Compression Scheme: LZW - Lempel-Ziv & Welch compression
- Photometric Interpretation: min-is-black
- Software: "Oracle Multimedia Toolkit, 6.0"
- Image Description: ""
- Samples/Pixel: 1
- Rows/Strip: 103
- Planar Configuration: single image plane
After TIFFCP

- **ERRORS**
  - Read error on strip 30; got 318 bytes, expected 634.
  - Read error on strip 31; got 3365 bytes, expected 3375.
  - But the TIFF file is readable
  - TIFF Directory at offset 0x1fb48
    - Subfile Type: (0 = 0x0)
    - Image Width: 2544 Image Length: 3300
    - Resolution: 300, 300 pixels/inch
    - Compression Scheme: CCITT Group 3
    - Photometric Interpretation: min-is-black
    - Software: "Oracle Multimedia Toolkit, 6.0"
Try ENQing TIFF Files

• Now, the modified TIFF file can be printed to a T630 LEXMARK.
• FORMS 6i does not crash
• Simple repair, run TIFFCP on over 48,000 MOTO format file @ ~150k/file
Repairing TIFF Files

• Approach
  - Extract from ORACLE MOTO TIFFs onto the production file system - AIX
  - ftp files to non-production environment
  - TIFFCP all ~48,000 files
  - Store all ~48,000 files back into ORACLE
**Oracle Environment**

- Directory entry in `SYS.ALL_DIRECTORIES` for file path

```
select * from sys.all_directories
```

<table>
<thead>
<tr>
<th>OWNER</th>
<th>DIR_NAME</th>
<th>DIR_PATH</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS</td>
<td>OUT_DIR</td>
<td>/u01/app/</td>
</tr>
<tr>
<td>SYS</td>
<td>REBUILD_DIR</td>
<td>/u02/oradata/</td>
</tr>
</tbody>
</table>

- 2 rows selected
PL/SQL

• Write BLOB contents to a file

v_out_file := UTL_FILE.FOPEN(
  location => 'REBUILD_DIR',
  filename =>
    'cal_' || '&1' || '_' || '&2' || '_' || '&3' || '_' || '
    &4' || '_' || '&5' || '_' || '&6' || '_' || '&7' || 
    '_tif_prod',
  open_mode => 'w',
  max_linesize => 32767);
Shell Script Insert TIFF

• Slash fails with ROWID
• File system considers directory
• Use keys

```bash
sqlplus -s scott/tiger@tiff_1 @cal_tiff_insert_slash_1.sql "002207238" "9780MB7226403" "FUB7861" "9743A" "1" "1" "AAAG71AAO"
```

• AAA+CqAA/" "107589"
Shell Script Extract TIFF

• Slash fails with ROWID

• File system considers directory

• Use keys

  • sqlplus -s scott/tiger@tiff_1
    @cal_tiff_extract_slash_1.sql "002207238"
    "9780MB7226403" "FUB786I" "9743A" "1" "1"
    "AAAG71AAOAAA+CqAA/" "107589"

• Note "/" in rowid interpreted as directory
• Store BLOB back into Oracle

• `src_file FILE := BFILENAME('REBUILD_DIR','cal_'||'&1'||'_'||'&2'||'_'||'&3'||'_'||'&4'||'_'||'&5'||'_'||'&6'||'_'||'&7'||'_tif_prod_new');`
• Store TIFF file to Oracle

• `DBMS_LOB.LOADFROMFILE(    --Sets the destination
   file with the source file value
   dest_lob => dst_file,
   src_lob  => src_file,
   amount   => DBMS_LOB.getLength(src_file));`

• `UPDATE image_detail d   --Updates the table (You
   can use insert...)
   SET img_data = dst_file
   WHERE   ROWIDTOCHAR(d.rowid) = '&7';`
Outstanding Issues

- Manipulate BLOBs using a remote database - ORA-22992 error
- Printing images to Lexmark T630 - no image, engine activates
- Forms crashes on some images - Segmentation Fault message on form
Resolving Issues

• Manipulate BLOBs using a remote database - use global temporary table
• Printing images to Lexmark T630 - Lexmark supplied firmware patch
• Forms crashes with some images - fix MOTO format
Things to Keep in Mind

- Metalink still logs issues with READ_IMAGE_FILE
- Latest entry Oracle Forms WebUtil : Technical FAQ  Doc ID: Note:270940.1 Type: FAQ  Last Revision Date: 23-MAR-2006
- Gotcha - ROWID is not always reliable – contains “/” – can be construed as a directory delimiter
Questions
Final Notes

- Feel free to contact me with any issues
- Email cleviter@ieee.org