

PERFORMANCE INVESTIGATION TOOLS & TECHNIQUES

7C

Matthew Morris

Desynit



ADC AUSTIN

MKS

WEBSYDIAN™

cm FIRST



Desynit

- > Founded in 2001
- > Based in Bristol, U.K
- > Customers worldwide
- > Technology Mix
 - 2E/Plex
 - Java & .Net
 - Web & mobile applications
- > Contact
 - Matthew.morris@desynit.com

The business need for performance

- > Do more, with less
 - Need to make sure strategic applications can deliver
- > “Our core system must continue to support the business for at least another 5 years”
- > How can we make sure the system does not limit business growth?
- > IT must seek to:
 - Shorten our delivery times
 - **Improve quality of application code & structure**
 - Be able to integrate and adopt other technologies

2E Batch Application

IBM I SERVER

Support Business Change

- > Our business is now global
 - The end of day process takes 6 hours to complete
- > Our new Australian office can access the system for just 2 hours in their working day
- > How can we give them more time?
- > Review the EOD process:
 - Remove redundant processes
 - Reduce the system downtime
 - Minimise risk : DO NOT BREAK THE EOD PROCESS!

History

- > 2E generated application on IBM i
- > ~15 years of development
- > EOD process grown organically
- > Many, many programs and reports called
- > Hardware capacity not the limiting factor

Application needs to be more efficient

- > Long EOD is obstacle to business growth
- > Every hour saved in the EOD means more work can be done in other time zones
- > Improvements needed A.S.A.P
- > Solution needs to be cost effective

Where to start?

> Break down the process

- Too large and complex to start with the code
- Use runtime program analysis tools
- Diagnose problem & make changes

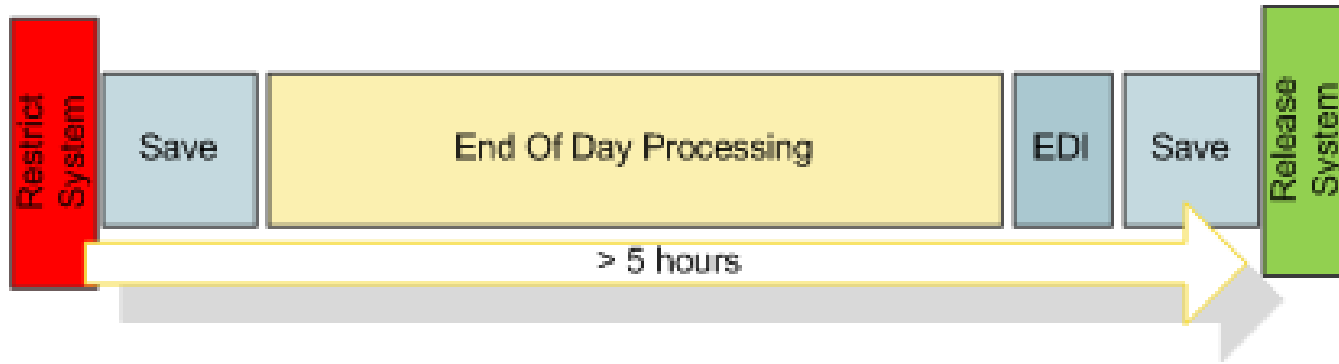
> Find the problem QUICKLY

- ROI based on turning things around quickly
- Find the easy wins first

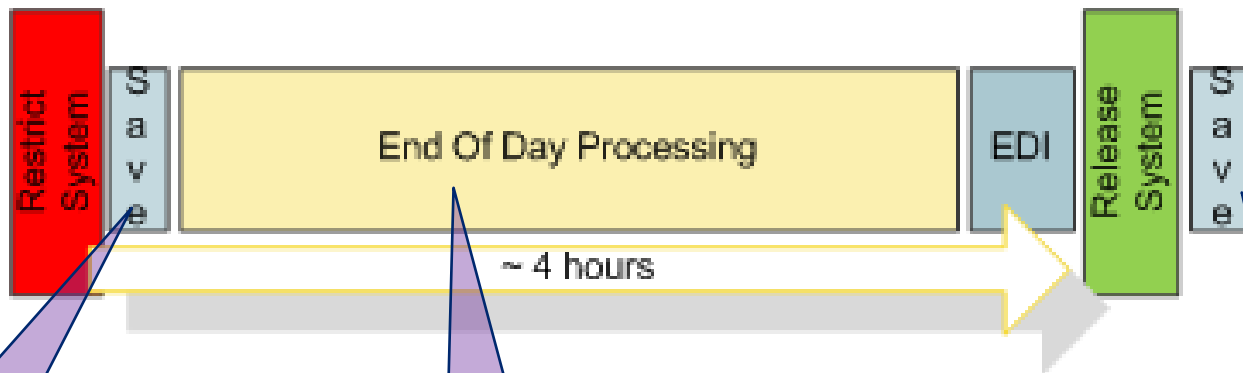
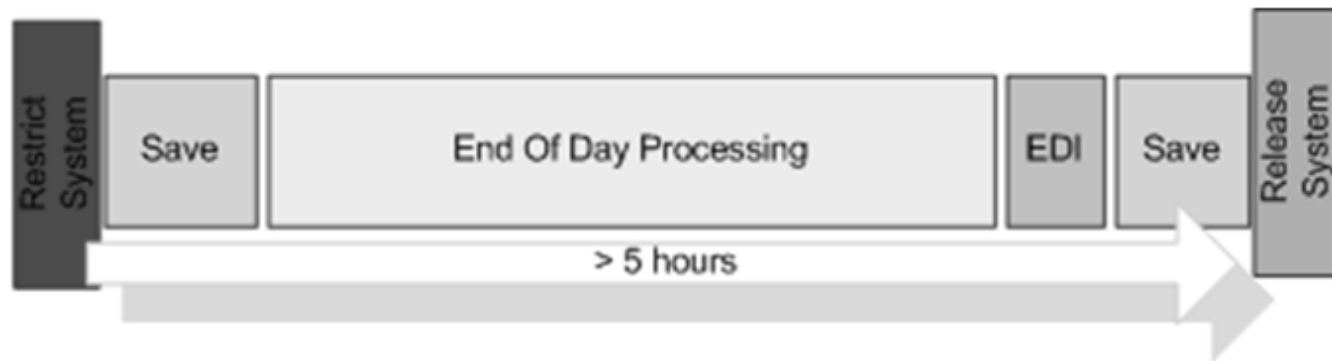
> Need to make changes with confidence

- Test that results are identical

"The System" - Today



Operational Improvements



DISK
REPLACES
TAPE

?

More information needed

SAVE
ACTIVE

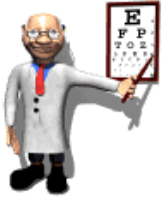
More Information needed...

> Need to find:

- What are the long processes ?
- Ratio of database READS vs. UPDATES ?
- How much time: Processing vs. Reports ?
- Central files ?
- Database locks or resource contention ?
- Artificial waits ?

IBM Tools – IBM iDoctor

“Collect, investigate and analyze performance data” - IBM



- > Profile system activity and resource use
- > Record program CPU & I/O
- > “Drill down” analysis tools
- > Results for our system...



System efficiency



Program efficiency

How to examine program efficiency?

- > Identify long running jobs
- > Build a timeline of key transaction files
- > Select programs for further investigation
- > Tag programs no longer needed

> Plan

- Map program & file dependencies
- Subset and prepare test data
- Isolated environment with copy of updated files

> Run

- Snapshot & rollback environment
- Save all results!

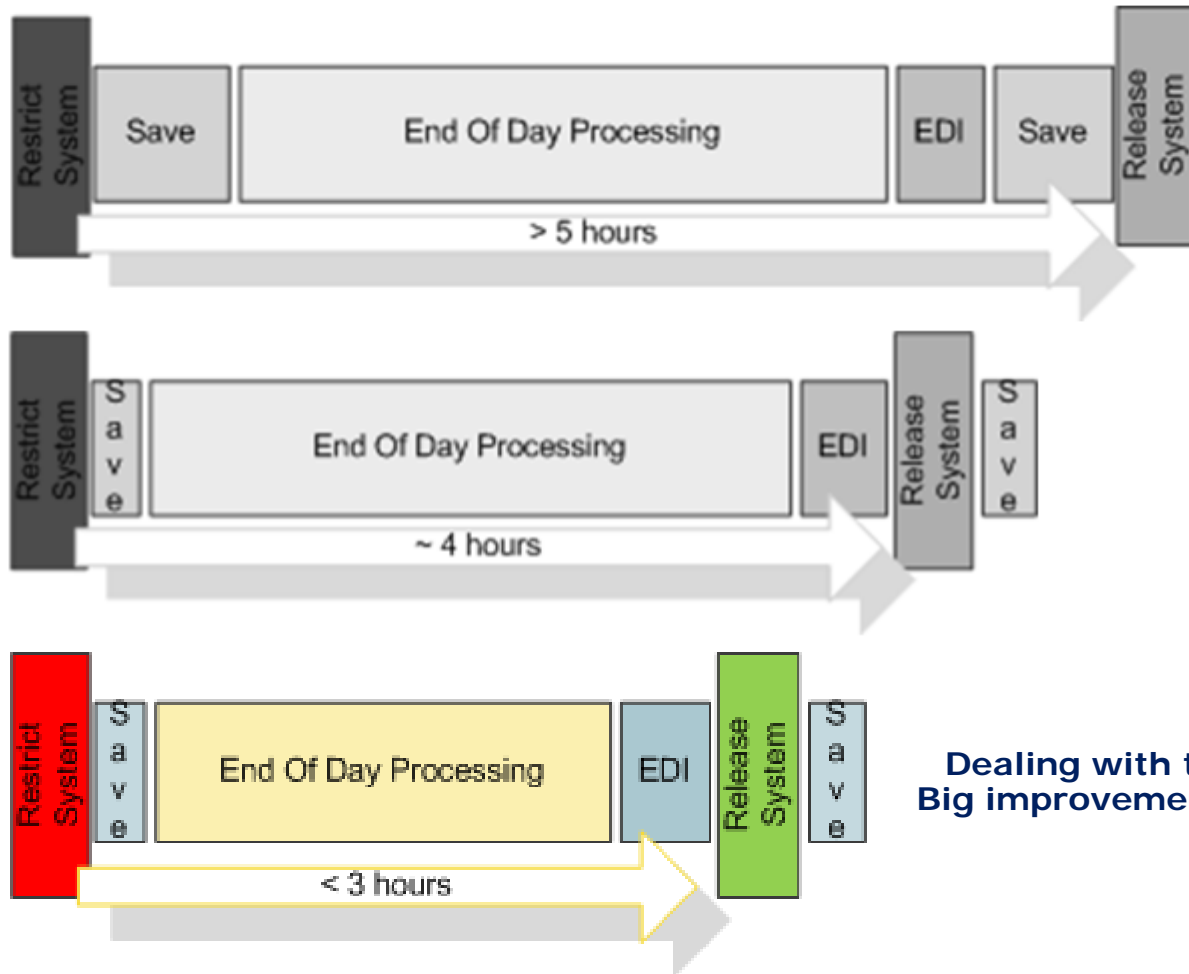
> Compare

- With baseline, or with other run
- Detect field level differences, changes in I/O

Solutions – Low impact – High return

- > Fix the key issue
- > Restructure can wait
- > Top offenders
 - Row by row record filtering – Replace Select/Omit
 - Unused virtual fields – Dropped from AP
 - Bad OPNQRYPIL – Selective use of SQL

Initial Results



**Dealing with top offenders:
Big improvement in efficiency**

ROI

- > Return on effort... 12 weeks effort = ~30% reduction in time
- > Business advantages
 - System available for longer
 - IT properly equipped to change core systems
- > Next Target = 2 hours or less

Confidence to make change

- > Like for like comparison
- > Experience and confidence in tools
- > Reengaging with older parts of the system

Performance was the beginning

> Database Modernisation

- To SQL Schema

> Application Modernisation

- Separate processes to run concurrent jobs
- Separate updates from report code
- Consolidate & rewrite key processes (in SQL?)

> Ongoing & incremental improvements

- Externalize “services” for business use
- Software change management
- Automated testing

Plex C/S Application

WINDOWS CLIENT

Client Performance Tuning

> Application

- Assist in development optimization (our use)
- Improve existing applications

> Locate bottlenecks

- Client side execution (VBScript, repeated external calls)
- Duplicated server calls
- Poor performing server processes

Capture Execution times

> C++ Source

```
API Call Source code: DSYBASE/DsyModel.GetTimeWithMillSeconds.GetTimeWithmillseconds
{
  /**
  #include <sys/timex.h>
  {
    struct _timeb timebuffer;
    char *timeline;
    _ftime(&timebuffer);
    timeline = ctime(&(timebuffer.time));
    char Buffer[50];
    char milliRuff[3];
```

> Minor code changes in function to trace

- Mini Pattern – “Go Sub LogTrace”
- Supports multiple traces

> Configurable

- “+ If Debug”

> It's fast/flexible & free

In use examples

> Time grid load

- Trace events row-by-row
- Added artificial waits and calls for demo

> Sample output

Common Problems & solutions

> Changes which make a difference

- Fragmented VBScript – consolidate

> Repeated Client server calls

- Not obvious due to Meta code?
- Fields in result set variable, are all used?

> Poor database access performance

- Replace Position statement with Select Where?
- Manage the size of the result set
- Move logic to database functions?

The business need for performance

- > Do more, with less
 - Need to make sure strategic applications can deliver
- > “Our core system must continue to support the business for at least another 5 years”
- > How can we make sure the system does not limit business growth?
- > IT must seek to:
 - Shorten our delivery times
 - **Improve quality of application code & structure**
 - Be able to integrate and adopt other technologies