DB2 Next is now on its final rally towards GA and this presentation will give the audience a view into the amount of work being done in what is normally known as “regression testing”. This is the testing being done where we try to expose a new version to a direct repeatable workload without any changes to neither the applications nor the system. We will therefore NOT discuss new features in DB2 NEXT but only give a preview of how JN Data saw DB2 NEXT in aspect of performance and stability. At the same time we will look into the methods and the tools used to re-produce a given workload and repeating this. Finally we will talk about how performance figures were collected and compared. This walk-through can be used by other installations to do similar testing for instance when investigating impact of table changes etc…

The hope is that this presentation will :

1) Make the audience aware of the kind of testing the ESP customers are doing to ensure smooth migrations for others.

2) Give information about the way stress testing was done by us during the ESP. This can be inspiring for customers own stress testing procedures in other situations as well.

3) Give first indications on what this ESP customer experienced of performance impact and BIND issues in the new version.

4) Give early information of the migration process as experienced at this ESP customers installation

Agenda

- DB2 11 ESP – A new process ?
- What do we actually do in this process
- The importance of “regression testing”
- Tools used during “regression test”
- Specific results
- Conclusions

The agenda for one hour of smooth talking……

DB2 11 ESP – what is it ?

- ESP says Early Support Program
- DB2 10 was QPP (**Quality** Partnership Program)
- QPP meant assuring the quality of the product as it evolved.
  - Not necessary to end the program running DB2 10
  - DB2 10 evolved as the program progressed
- ESP means focusing towards actually running the code
  - We need to end the program on a ”production-system”
  - Code was in principle ready to run when program started
  - We were allowed to back out of DB2 11 when program ended

There has been a significant change in the way this early quality assurance has been done from IBM’s side. In DB2 11 they have shipped a ‘GA-ready’ product where the ESP customers will track errors and documentation incorrectness to be rectified by normal PMR-process. The customers are supposed to end up running the actual DB2 11 code in a “production-like” system. In DB2 10 QPP it was more like a version under development that was given to QPP-customers in bits and pieces.

The realty, however, was a little mix of ESP and QPP

Testing activities

- A lot of activities in migration and fallback testing
  - This is the users first view of a new version
- Looking at new features
  - ESP customers sign in to test the different issues
  - Strong commitment on some features (Extended RBA/LRSN, etc)
  - Normally it will take a while before installations get into new features
    - ESP customers will value these up towards their business….
- How will current workload behave (regression):
  - Performance – on general workload and on specific SQL
  - Errors
  - Results
  - Access Paths !!

One of the things that still (after all these years) seems to worry people are the first rebinds on a new version. Giving all the changes for access path stability and ways of influencing the chosen access path it still is an issue in many installations.

Another very important issue is the regression – will the new version carry on the V10 workload without any binds or changes to application at the same or (preferable) better performance ???

On day one (and for the first year) this is perhaps more important than new functionality as this will normally take a long (too long) time before people start using new stuff.

It is therefore very important that (at least some) ESP customers go deep into ensuring that the new version will execute the old versions workload without problems and overhead…… We tried (really !!!) to ensure this – with the problems this gave !!!

DB2 11 ESP regression testing– Building targets

- Defining a new Subsystem right away…
  - Enables DBA's to go straight into testing of new features in a sandbox
  - Requires CF structures, Disk space, VSAM cat, RACF defs, STC etc etc
  - Gives first shot at DSNTINST system
  - After initial definitions approx 8 hours before system was up and running
  - Experience said little initial usage as
    - No tools from vendors
    - Not incorporated in our own infra structure
    - Too many surprises (Different ZPARM setting, not resources enough)

For our DBA team is was necessarily that we had been through the migration process before they could be handed a system to look at workload and access paths. To try to kickstart the ability to look at new functionality we started by defining a complete new subsystem that could be scratched and re-defined at will. This is a very fruitful activity as a subsystem like this can be used for many purposes.

In this process it gave us an early experience on the DSNTINST system and on the Install/migration documentation. In our opinion this could need an overhaul as the doc and the jobs does not give any information as to what can be done BEFORE the system is shut down for migration; what NEED to be done while the system is inactive and what can be POSTPONED to after the workload has been re-instated. It seems that the doc and jobs are from before we ran 24*7 and we pinpointed this need already at QPP for DB2 10. But, patience as we are, we repeated this recommendation.

The new subsystem was however not that useful when started as no tooling was working. Therefore the DBA's only had SPUFI and DB2 commands. It was also needed to ensure more resources (like CPU,CF,DISK and RealStorage) was available. It was also a problem that the subsystem was not incorporated into the infrastructure for performance collecting etc etc

DB2 11 ESP regression testing– Building targets

- Cloning a customer subsystem
  - Known territory for DBA's
  - Cloned to isolated LPAR ** important **
  - Took QA environment to assure required privacy
  - Cloned VSAM, RACF, CF definitions
  - Took DFDSS dump of "almost" entire system after stopping
    - Around 1 Tb of data (4 hours)
    - We don't have disk flash technology licenses
    - Moving 1 TB from one LPAR to another
    - Restoring 1 TB of data (8-10 hours)
    - All changes to subsystem has to be saved to be redone if subsystem recreated
    - So same subsystems, same DS-grp, same dataset names, same RACF defs etc
    - All DS members have to be ready
    - Set ZPARM to scratch CF-structures (group restart)

Next was to clone actual production-like system. We started by taking a QA-system from one customer and simply did a DFDSS DUMP of most of the data (around 1 Tb) after the system was stopped. This took around 4 hours and was done as we did not have the licenses for flash copy API's.

We then ported the dumped datasets to an test-LPAR where we wanted to re-establish exactly the same subsystem on this LPAR, bit by bit. We had the luck that we did the same in the QPP for DB2 10 and we had preserved the list and most of the definitions for RACF, STC, CF-structures, VSAM and automation and much, much more. All the datasets were the restored and this took some 8-10 hours before the 1 Tb was restored. All that was needed was to set the Zparm to do a group restart to avoid having manually to scratch the structures before a repeated restore. However a need for different binds of tooling etc as the originating subsystem evolved has to be repeated at the clone. Save all this work for future reference….

DB2 11 ESP regression testing– Building targets

- Cloning another customer subsystem
  - Kept on same LPAR
    - Need to rename everything
  - Define VSAM, RACF, CF definitions
  - Took DFDSS dump of system after stopping
    - We don't have disk flash technology licenses
    - Renaming VCAT for all tablespaces
    - Renaming SSID's and DS group names
    - New CF names
    - Renaming BSDS names and contents
    - All changes has to be saved to be redone if subsystem recreated
    - So new subsystems, new DS-grp, new dataset names, new RACF defs etc
    - Convenient procedure for other activities

To do some additional testing on changed to facilities not used on the previous cloned subsystem (like the RACF-exit changes) we cloned another customers subsystem. This was a little more complicated as we could not go to another LPAR and retain all names. We did therefore use quite some time in renaming a complete subsystem with its subsystem-id's, DS-grp names, STC-names, CF structures, VSAM names etc etc etc.

Eventually this was done successfully and are now part of a procedure that enable us to do this at will.

DB2 11 ESP regression testing– Access path analyze

- Used the cloned – non-renamed subsystem
  - Started it on DB2 10 with fallback/toleration fixes applied
  - Did a REBIND(*) to get DB2 10's view on current resources (CP's ,storage)
  - Saved all access paths into user tables
  - Migrated subsystem to DB2 11
  - Did a REBIND(*) on all (63300) packages and saved the access paths again
  - Compared using own method :
    - 99% not changed
    - <1% (470 packages=1100 statements) changed – but of these
      - 7% judged as harmless
      - 45% judged as potential improvement
      - 12% judged as uncertain
      - 36% judged as a potential problem
    - the last 48% still represents > 500 statements !!!!!!!
  - Redid scenario using APCOMPARE on DB2 11 at a higher fix-level.

Gives different result. Hopefully more details at the conference

Before you do a migration to DB2 11 it is important that all members have been active with the toleration/fallback SPE applied. This was like it was in DB2 10. However the SPE was not an official fix but a USERMOD which is a problem to get rid of again.

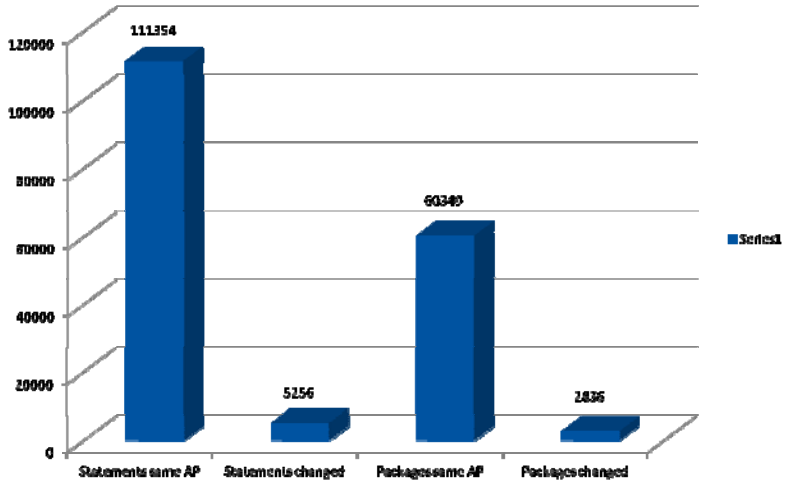
But to be able to compare access paths it is vital to do a REBIND(*) on the clone on the old version to cope with DB2 10 changes, CPU sizes etc etc

After the migration we redid this REBIND(*) and did a compare on the access paths. We did NOT use APCOMPARE etc etc but relied on a own REXX-system….

The comparison showed a lot of differences…..

DB2 11 ESP regression testing– Access path analyze

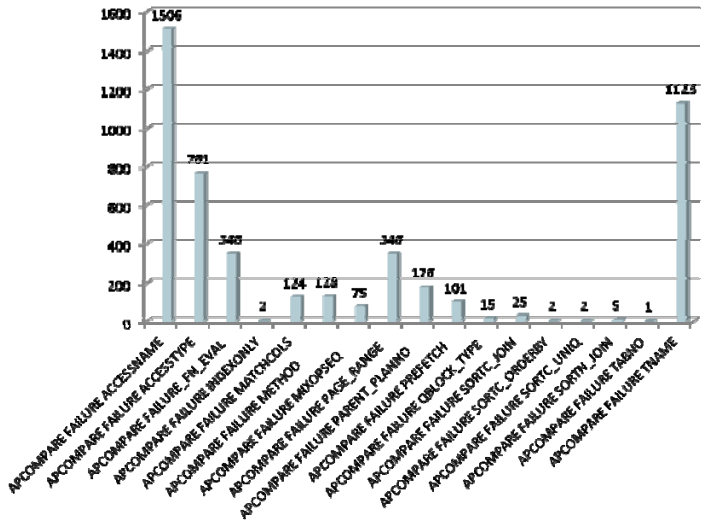
- The 60000 packages contains 111000 statements.
  - 5200 statements changed access path



REBIND APCOMPARE(ERROR) at a later point of time gave a different result. The comparison was done at a very late DB2 11 fix level.

DB2 11 ESP regression testing– Access path analyze

- Using APCOMPARE(ERROR) EXPLAIN(ONLY) shows "WHY"
  - But not better/worse
  - Nor makes analysis a lot easier – but still a good overview :



The column REMARKS in PLAN_TABLE contains details of why APCOMPARE(ERROR) failed. Not there is still a long way to go to be able to understand what is a problem and what is an improvement…

DB2 11 ESP regression testing– Performance analyze

- For verifying performance we need the usual tools :
  - Monitors
  - Administration tools
  - Change tools
  - Accounting/performance reporting
  - Accounting/performance comparisons
  - Workload simulator/canons
- As always :
  - New versions of all tooling needed (additional Beta-programs)
  - These versions not available at start of ESP
  - Need install process as we receive working versions
  - Takes a LOT of work

When we started considering performance considerations we looked at how we did it in DB2 10 where we had HUGE problems generating a complex repeatable workload. From our professional point of view what we did in DB2 10 was good (better than nothing) but inadequate. So we wanted to do a better job this time and we spend some time looking for ways of producing a repeatable workload of a suitable complexness.

At the same time it was needed that tooling for administrating and performance were able to cope with the new version.

Doing a scenario like this takes a LOT of time !!!

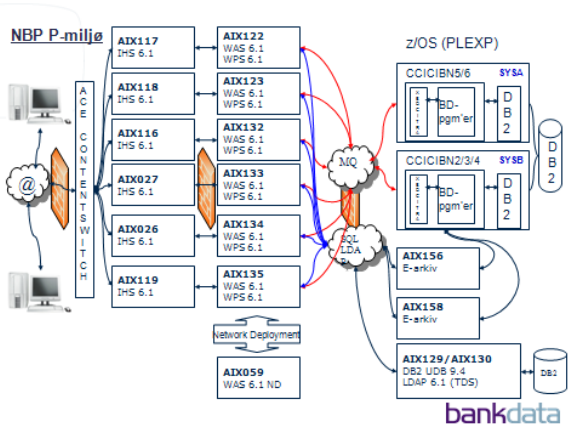
Workload simulator/canons

- For testing regression and migration a workload is needed
  - That can be repeated, measured and compared
  - This is a need that we have already today :
    - DBA needs to verify new options
    - SYSPROG wants to check a new ZParm or bufferpool
    - Capacity people need to verify zIIP setting or # of processors
  - Most installation have not recognized the need ?
    - Too complicated
    - Too expensive
    - Unknown reference installations
    - Many different needs = many different approaches
  - A working procedure/tooling in this area will
    - lift quality
    - reduce outages

The process of executing a repeatable workload again and again and collect documentation of the performance is really not a DB2 11 ESP issue. One can argue that this is an everyday need in most installations as we could use this every time we want to change a setting. For instance if a DBA could change a tablespace from index partitioned to a UTS PBR and issue a full days workload on a clone this could give invaluable knowledge of the effect of a change. Few installations have a significant workload in testing that can prove that a change will not give problems in production. And what if a new storage subsystem is deployed and you could fire in 4 hours of production workload on a clone using the new box ?? There are so many possibilities if a method like this could be possible.

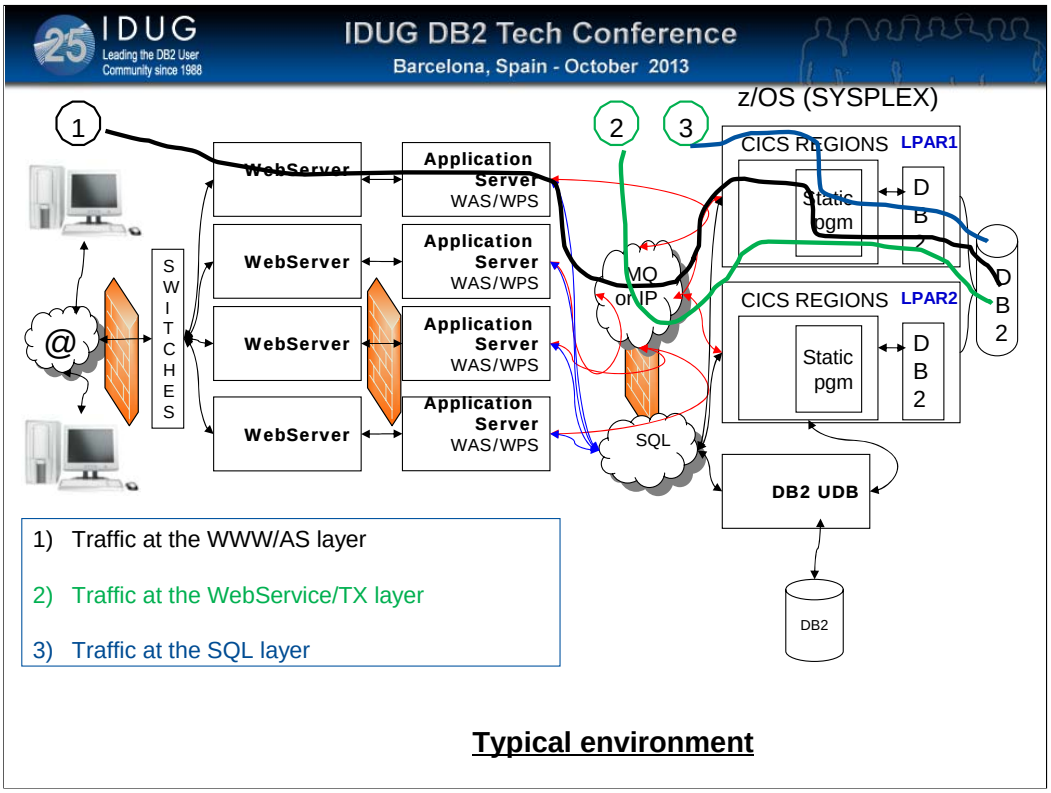
Workload simulator/canons

- Let's try to draw it :



Let's look at a "normal" environment these days. A number of web-servers receiving traffic going into application serves on AIX, Windows or Unix will send traffic to CICS using WebService-calls or MQ and finally go to DB2 on z/OS.

Besides this there can be other distributed servers or external partners.



Typical environment

If we take that same application (it COULD be a vital Internet banking application) there are 3 different levels where we could fire in traffic using some sort of artificial workload generator.

We could fire it in at the WWW layer using one of these WEB traffic generators that are available. In this way we will verify all components all the way through but it should be obvious that the setup and the cost of software and hardware for all these levels of components are very high !!

Another alternative could be to somehow collect the webservice or MQ calls and redo these calls. In this way we would in this application test all components from MQ/CICS and all the way through. However the cost will still be high as all SQL calls have to execute successfully. So all tables used have to exist and to be reset before a rerun.

The last way is to only fore in the SQL. Then we could accept that tables were missing and we would only verify DB2 and the objects !

Workload simulator/canons

- Different approaches :
  - Shooting a repeatable workload can be done at 3 levels :
    - From the WebServer (for application Servers)
    - From the transaction level (CICS TX, WebServices, MQ messages)
    - From the DB2 point of view (single SQL call, dynamic or static)
  - All these levels have each their pros and cons in conjunction with:
    - Different needs
    - Costs
    - Replicating/simulating workloads
    - HW and SW demands
    - # of people involved

It is so important that you know what you objectives are for the different level of testing using these methods. They aim towards very different needs and have big differences in expenses for using. However in my opinion if you need to test all the way from webservers through CICS down to DB2 but feel this is too expensive it is of no use to pick the alternatives. Careful considerations are essential (like it is in life in general ☺)

Scope	Purpose	Pros	Cons	Comment
Webserver Layer	To validate end-to-end changes in an environment	•Will validate every components •It is normally possible to script transactions for new values of keys etc •Can cope for any change in SW or HW or application	•Expensive and time consuming •Difficult to understand differences. Will require a lot of monitors and tools. •Need to be maintained at all levels •Security can be very complicated •Parallelism is very difficult to control	These tools do exist and they can normally script on the input. They tend however to be used very little because of price and amount of work.
Simulation at WebService or MQ-layer	To validate the environment from the WebService or application on the in-ward side	•Will validate all central components •Scripting can be done if the layout for the message is known (XML ?) •Can cope with changes to components on the inward side.	•Expensive and time consuming. •Complicated to target to another environment (other MQ queues) •Has to be managed on the 'transaction' level. •Parallelism is difficult to control	I am unsure if tool exists at this level but it would be fairly easy to build a sort of logging into applications at the MQ or WebService layer and in this way spool the calls onto the queue again.
Simulation at database call layer	To validate application performance and behavior at the database call level.	•Will look isolated at database calls meaning no application code executing •Excellent for investigating performance and locking problems. •Can look at a subset of tables	•Easy and inexpensive to use •Database calls has to be run in sequence – no SQL call can overtake the other. This is both an advantage as a disadvantage as it means that thins can be compared but makes it difficult to do real stress testing. •Will never find bottlenecks in other things than the database.	Static SQL support is essential to compare performance . If static SQL is being used understand impact on new version of application. If dynamic : can SQL be modified or scripted ?

Here I have tried to set all these different ways up in a comparison table. Even though there are many words I think it is well worth reading if you consider doing some sort of workload testing of your system. The short story is that the more components you want to test the more expensive it will get. And I don't think the expense curve is linear. Doing a test from the www-layer is extremely expensive and work-intensive while doing a test on the SQL-level can be done at a lower price. However the number of components that is being verified is far, far less.

There is so big a difference so I might even say that comparison is unfair. The testing is done for different purposes. But remember to judge if a test-method might be so work-intensive that it is not used. On the other hand if it only tests so few components that you miss your objective is it also a waste of money and effort.

Regression testing – workload simulation method

- If workload simulation can not be used it is IMPOSSIBLE to compare performance
- In DB2 10 QPP we did not have any so we used some (few) batch jobs and a isolated CICS-workload
  - Not representative
- In DB2 11 ESP IBM offered beta process on some tooling :
  - OPTIM Query Capture and Replay
  - Can capture at the SQL level (remember the ugly slide 14) on one/more DB2's
  - Can replay 'static as static' and 'dynamic as dynamic' at same or other DB2
- Several similar tools exist from different vendors but we could try this for free
- Other tools were not considered or validated !!!!!!!!!

When we did the DB2 10 QPP we had huge difficulties in doing comparable runs as a given application tend to use more or less all tables in the system. And applications might not be able to execute on another schema again and again because of date problems or some common subroutine that can not handle to run in the past. So we ended by having very few and very simple scenarios in DB2 10.

As IBM now offered OQCR for DB2 11 ESP customers we participated in the Beta for this to gain some early information. There are several tools on the market in this area and we have not evaluated any of them and we only plan to use this for this ESP process. However it was important to pick a tool that could replay static SQL as static SQL and dynamic SQL as dynamic SQL. In this way we could get all accounting reports comparable (to some degree) and we could evaluate rebinds etc etc.

Regression testing – Optim QCR

- The product consists of several components :
  - 1 (preferable physical) X86 based server from a positive list (the “G-machine”)
    - Will run the Guardium product
    - Linux black box system with only limited access (NO root !!!!)
    - Installs directly from a DVD
    - Difficult to comprehend at the beginning
  - Several Started tasks on z/OS
    - 1 per LPAR where captured/replayed subsystems reside
      - Controlling other STC’s
    - One additional STC per subsystem
      - Running USS processes
      - USS/UNIX installation required (knowledge)
      - Is doing the captures
      - communicating with G-machine
      - Started/stopped automatically
    - One additional STC per LPAR
      - Handled entirely by the product

If we look at the architecture for OQCR it is fairly complex. It uses almost every know component on z/OS and a Linux server for the Guardium appliance.

Regression testing – Optim QCR

- Observations :
  - Firewall definitions needed (=knowledge)
  - LINUX knowledge not directly needed – but useful !
  - USS knowledge VERY needed !!
  - z/OS knowledge (STC, troubleshooting etc) VERY needed !
  - A good sense of analytics and error-isolations will save days !!!!
    - For the end user to do capture/replay a WEB browser is THE tool
      - Don't think that this is enough
      - Put together a team of the skills above !!
  - The capture is started on a time basics and run for a time period
    - Lot of filtering possible
  - Product is hooking into DB2 and SQL is shipped over IP to the G-machine
    - SQL can NOT be reviewed after capture

In my opinion and experience it takes skills inside everyone of those components to ensure a successful implementation. Once the tool is running it is driven from a browser driven frontend... And it intended to be operated by a DBA or a Systems Programmer … (Or perhaps a developer ???)

The capture is stated using filters and for a defined amount of time. Remember to synchronize the data somehow so you can re-establish the same data for the replay somewhere……

The product get a hook into DB2 and will collect the SQL to its own STC where it will be buffered and uploaded to the G-machine over the IP network.

When the capture is completed you have the SQL frozen in the G-machine. But you can not review it nor change it for now !!!!

Regression testing – Optim QCR

- Observations :
  - After successful capture the SQL is “transformed”
  - Optional to change SCHEMA (only for dynamic SQL)
  - Optional to change collections (for static SQL)
  - Optional to change USERID for CONNECT statements
  - Transform process also ask for target subsystem…
  - Transform process normally not expensive
  - After transforming the workload is now ready to be replayed again and again
  - Before replay :
    - Target tables have to be reset to start position (tools or backups)
    - Remember your politics concerning data privacy
    - Check the WLM priority for the USS processes – it can consume LOTs of CPU

Now the SQL have to be made ready for the repeated replaying. You target for another (Or the same) DB2 subsystem and you can transform the Schema and userid for the connect. Static SQL is static SQL and can not be changed !!!! It would have been nice if you could change to a different collection, though……

Regression testing – Optim QCR

- Replay – needed knowledge
  - The Replay will embed a new capture (might change) !!!!!!!!
  - The Replay will replay everything over a number distributed threads
    - But dynamic as dynamic and static as static
  - The Replay can be influenced on speed (partly)
    - Only on removing/reducing pauses – SQL can NEVER overtake other SQL
    - No more/less contention than on original workload – unless something changes
  - IF replay fails – you have wasted a replay :
    - Data has to be reset as replay can not restart !!!!
    - Can be time-consuming and frustrating
    - DB2 know how far we got (backout/COMMIT). OQCR should know it as well ☹
  - Replay can be re-done…..
    - On new version of DB2, DFSMS, archive strategy of LOGs, DISK setup
    - On new "DDL" – convert to UTS, PAGESIZE, LOCKSIZE, ZPARMs, REBINDs
    - Not for new static applications !!!!! (-805)

And now we replay on the target subsystem. When the replay is taking place OQCR will do an internal capture on all workload on the target subsystem. This might not be what you want but we used a cloned subsystem as target using DFDSS dumps taken just before the capture.

All SQL replayed have to be replayed in sequence. One piece of SQL can not overtake another one which is fail but it would be nice if you could overrule this valid rule. The result would be unpredictable but you might be able to provoke some locking issues. You can however ask the replay to remove pauses between SQL – "white space".

Be sure you understand that the static SQL is fired in through the real packages but all SQL does not have to run successful. You can very well only have a subset of tables restored and let the rest of the SQL for that package give -904 on every replay and still compare the successful calls from replay to replay….

If a replay fails you have to restart by restoring the data and redoing the capture. You can not restart it from where it left….

Tools in this area could give a very valid workbench when comparing different DDL and Zparms and version. Or disk subsystems… OR ??

Regression testing – Optim QCR

- Report – needed knowledge
  - After the first replay you can start doing comparisons
    - You can use business as usual : SMF, RMF, Monitors, Performance DataWarehouse
    - Or use the built-in reporting facility
      - Will look at elapsed time only
      - Draw charts etc
      - Have some (limited) filtering
      - Will tell overall outcome (SQL with same result, SQL with errors etc)
    - For ESP we used SMF, RMF, Monitors, Performance DataWarehouse
      - To ship to IBM
      - CPU time and different wait times essential to view outcome in detail
      - Omegamon delivered XLS-Sheet to uniform STVL reporting (courtesy Norbert Jenninger)
    - In my opinion the report is good for an overview
      - Not for detailed study of applications
      - If working scientifically save report as PDS under the project (“Larger pagesize”)

OQCR has some reporting facilities but these are very much based on SQLCODEs and elapsed time. For the ESP process we look more for CPU time than elapsed time. So for this reporting we used SMF and RMF to report back into the IBM organization.

The OQCR reporting is more to get an overview and if the product is used for more DBA-oriented stuff this should be saved for future reference.

Regression testing – Omegamon spreadsheet

- Omegamon delivered spreadsheet for customers to use
  - Ensures performance observations reported in standardized way
    - Omegamon job to extract needed data into CSV-datasets
    - Executed on before and after SMF
    - Uploaded to PC and imported into spreadsheet
  - In DB2 10 QPP each customer had their own method/way
    - Difficult and time consuming for IBM performance experts (yes, they DO look !)
    - Difficult for QPP customers to compare experience
    - So time consuming that I believe a number QPP customers gave up
  - The Omegamon product was delivered for all customers in ESP process
    - Gave a very neat and easy way of reporting
    - Saved a LOT of time
    - Lifted the quality of the Quality Assurance Process
    - Worked like a charm !!!

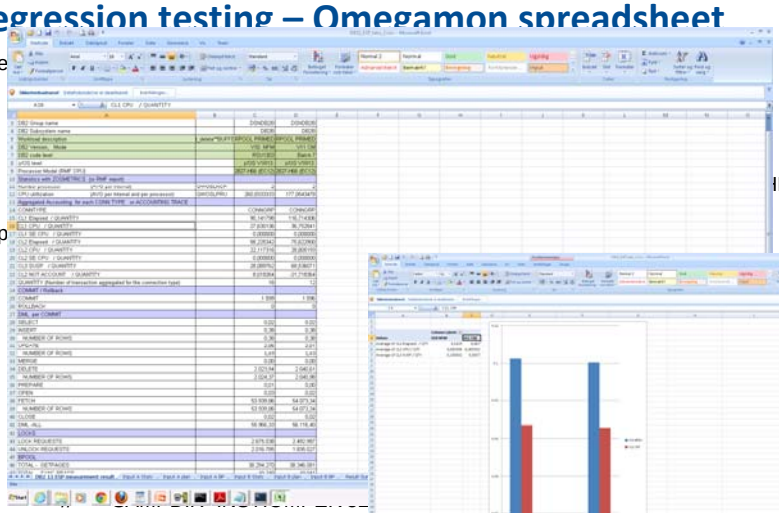
We wanted to do it better for DB2 11 ESP !!!! At the evaluation of DB2 10 QPP we challenged IBM for a way of comparing workloads. IBM's performance experts gave us a spreadsheet to which Omegamon (delivered for the ESP customers) could produce CSV-files to direct import into the sheet.

These spreadsheets ware used by IBM's DB2 11 performance team as entrance for the evaluation of the results….

Compared to DB2 10 this saved us weeks of work and produced more uniform data from the variety of customers….

Regression testing – Omegamon spreadsheet

• One



HMM) END(HHMM)

• step


```
//      FLDSELDS=INST.OMPE.V5
//      INPUTDS=UBIFAP.##JOBID
//      OUTPUTDS=UBIFAP.##JOB
//      OUTDISP=OLD
```

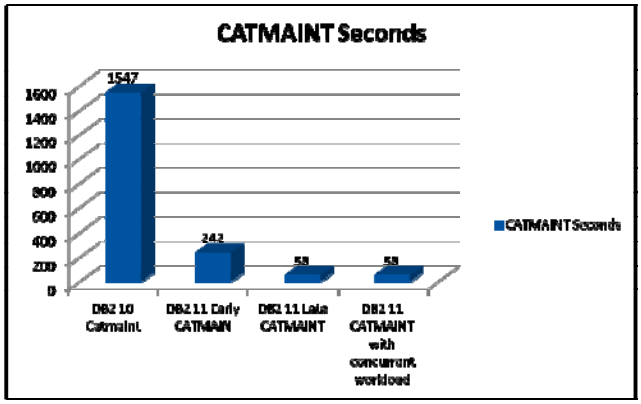
The spreadsheet and the JCL to produce the CSV-files. Very neat indeed !!!

Regression testing – CATMAINT

- CATMAINT is normally one of the more feared disciplines
- When you sit alone at migration night and watch CATMAINT throw the catalog up in the air :
  - You fear whether it will fall down in the right places
  - You fear if you can run workload on other members
  - You fear the execution time
  - In short you suffer from CATMAINT-PHOBIA
    - Recommendation :
      - Rehearse on a clone like earlier described
      - Will find old PTF-actions never performed in a safe way
      - Will remove the phobia
      - Do the catalog reorgs in advance
- Questions :
  - Will the catalog restructure from V10 benefit DB2 11 CATMAINT
    - On elapsed time
    - On the ability to run workload while you 'catmaint'
      - of course on other members !!

Regression testing – CATMAINT

- Migrating DB2 V9 to DB2 10
  - 26 minutes
- Migrating DB2 10 to DB2 11 first code release
  - 4 minutes
- Migrating DB2 10 to DB2 11 final code rally
  - < 1 minute !!!!!!!!!!!!!!!!!!



"Regression testing" – DSNTIJEN

- The DB2 10 DSNTIJEN converted these spaces :

SPT01	DBD01	SYSDBASE
SYSDBAUT	SYSGROUP	SYSOBJ
SYSPKAGE	SYSPLAN	SYSVIEWS

- The DB2 11 DSNTIJEN converts these spaces :

SYSLGRNX	SYSCOPY	SYSRTSTS
SYSTSIXS	SYSTSTAB	SYSSTR

- DB2 11 Elapsed time total job : 5 minutes (longest step 150 seconds)
- DB2 11 CPU time total job : 30 seconds
- Planning needed :
  - Still not able to run while dynamic SQL, Monitors are active :

```
DSNU778I  -DB2B …DSNUEXDL - ERROR PROCESSING SQL STATEMENT -
                    SQL CODE IS: -913
        SQLERRP: DSNXICTS
        SQLERRD: 0000005F 00000000 00000000 FFFFFFFF 00000000 00000000
        SQL MESSAGE TEXT: UNSUCCESSFUL EXECUTION CAUSED BY DEADLOCK OR
        TIMEOUT. REASON CODE 00C9008E……..RESOURCE NAME DSNDB06
            SQL STATEMENT:
    CREATE TABLESPACE SYSTSCPY IN DSNDB06 CCSID EBCDIC USING STOGROUP
```

- Cancel thread for the holder and rerun job. Nothing will be out of service after the abend !!!

Regression testing – Actual approach

- Run 1 -
  - Restore subsystem from DFDSS dumps
  - start replay
- Run 2 -
  - Restore subsystem from DFDSS dumps
  - start replay
  - Extract SMF
- Run 3 –
  - Restore subsystem from DFDSS dumps
  - Migrate subsystem to DB2 11
  - start replay
- Run 4 –
  - Restore subsystem from DFDSS dumps
  - migrate subsystem to DB2 11
  - start replay
  - extract SMF
  - Run omegamon extract to CSV
  - Import CSV into IBM spreadsheet
- 10M sql
- 1000 different packages

The steps involved in doing a regression test

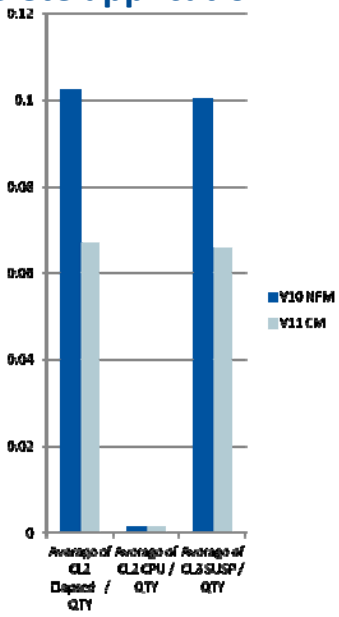
Regression testing – Actual approach

- Important note !!
  - In DB2 pre 10 bufferpools were allocated in memory when first object in pool opened
  - In DB2 10 this changed so bufferpools :
    - Were allocated at a minimal size when first object in pool opened
    - Expanded until defined size reached as usage increase
  - In DB2 11 we are back to pre 10 behavior :
    - Bufferpools are now again allocated in one go at their defined size
  - This has an impact on performance comparisons :
    - Performance will reflect this fact until bufferpool is "stabilized" (in DB2 10)
    - In comparisons like this it is impossible to prime bufferpools in a decent way
    - Therefore a longer run is needed – but still no guaranty !
  - General observation is that the real memory consumption in V11=V10

The steps involved in doing a regression test

Regression testing – Huge cursor delete application

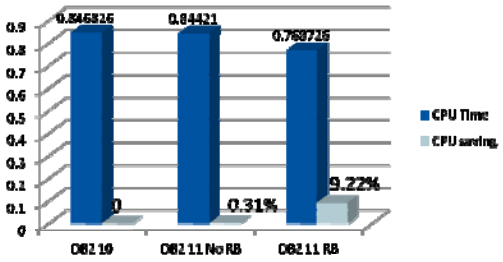
	Column Labels	
Values	V10 NFM	V11 CM
Average of CL2 Elapsed / QTY	0,1025	0,067
Average of CL2 CPU / QTY	0,001598	0,001532
Average of CL3 SUSP / QTY	0,100502	0,0657



•In general :
- •In runs like this - forget about elapsed
- •CPU difference seems to be some 4%

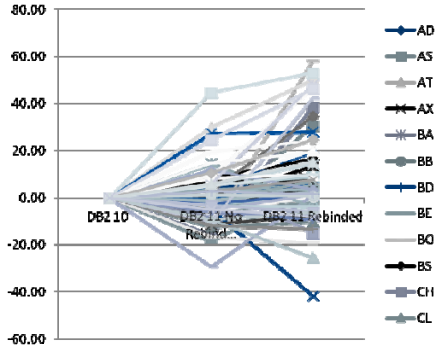
Regression testing – OQCR Replayed workload

- Workload consumption summarized on package level
  - DB2 11 without rebind = status quo
  - DB2 11 after rebind shows saving of 9%
  - But ……..



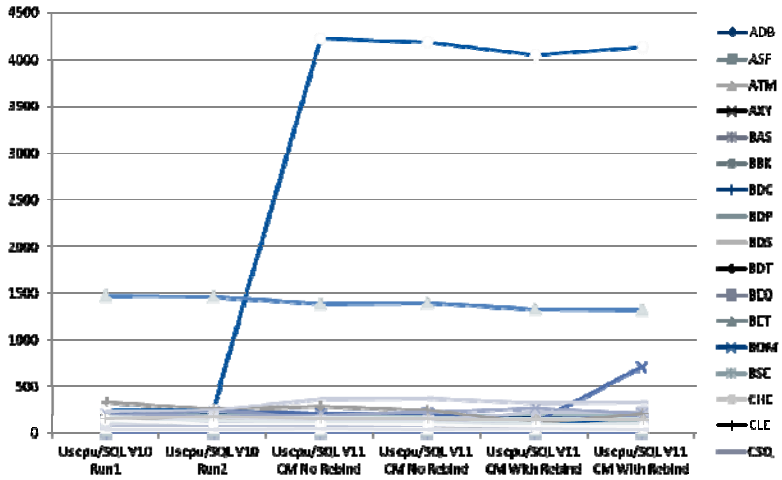
Regression testing – OQCR Replayed workload

- Workload consumption summarized on package level
  - plotting the saving in % compared gives other picture
  - the inventor of “the theory of chaos” has not lived in vain !
  - a clear picture is impossible to get
  - But ……..



Regression testing – Actual approach

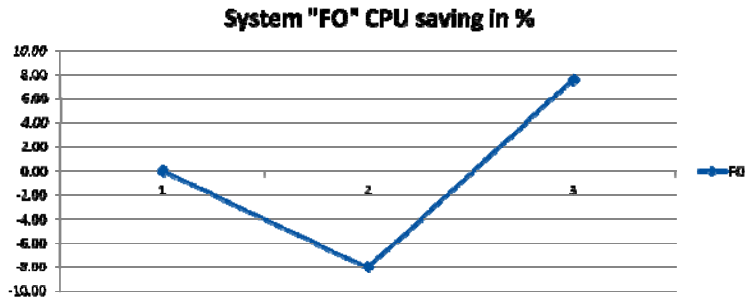
- The picture is however much more complex :
  - Some packages seems to go berserk
  - Currently being investigated



Regression testing – Actual approach

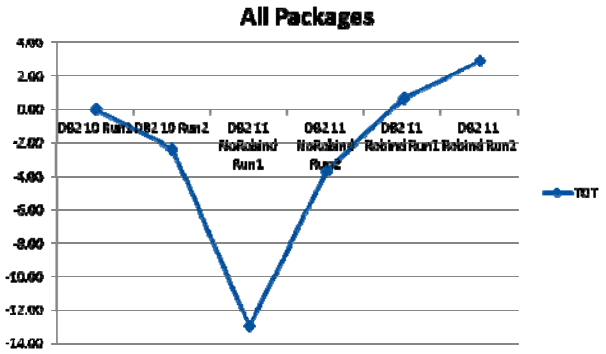
•The picture is however much more complex :

- Other packages follow the pattern we saw from DB2 V9 to 10
- first plot is baseline DB2 10, then DB2 11 before and after reb
- Seen is that a rebind is common sense to pick up the new SP



Regression testing – Actual approach

- The picture is however much more complex :
  - Seen here is all packages on average CPU per statement
  - our conclusion is the same as for DB2 10 :
    - we did not hit the “best” sweet spots for CPU saving
    - probably because most static, over years highly optimized SQL
    - YOU might see a complete picture.
    - DON’T sell (or buy ?) the saving before you have seen it !!!



Regression testing – More results

- More results and/or explanations might be available at the time of presenting !!!

Regression testing – Conclusions

- It has been very interesting to performance/regression test DB2 11
  - As always extremely dedicated people from IBM
  - And not least from the fellow ESP-participants
  - almost makes one cry when understanding how much work done 
    - For the sake of DB2
    - For the sake of the community
    - For the sake of our mutual interest in evolving effectiveness
    - For the sake of our mutual interest in reducing costs
  - Sophisticated tooling required to really measure performance

Frank Petersen
JN Data
fap@jndata.dk

Session A4

DB2 NEXT ESP – The process of verifying the performance and access path selection in a new DB2 version

INTERNATIONAL DB2 USERS GROUP
Founded 1988