

IDMS

connections

THE publication of the IDMS User Association

The CA-IDMS Database and Applications User Association

<http://www.iuassn.com>

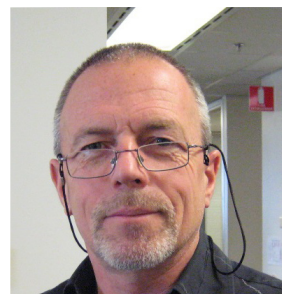
March 2011, Number 74

INSIDE THIS ISSUE

Letter from the Editor...	1
Message from International Chair ...	2
Open Letter to the IDMS Community...	3
IDMS Business Value Story...	4
Database Performance Analysis...	5
FOS/WEB Services at State of Iowa...	11
If...Then...	12
IDMS-L Rule Change...	12
How many ways to do things right...	13
Fingerprinting Load Modules...	14
The Backpage Homepage...	16

LETTER FROM THE EDITOR

At the moment I am splitting my time between editorial duties for “IDMS Connections”, and general “Communications” dogsbody for the IUA/EIUA Global User Community. Both roles have their challenges, and their rewards.



One sign of the health of any User “Community” is the level of participation by the Community Members. I see the role of “Community Communications” as being to encourage and engage the Members in matters of Community interest, and to raise participation rates.

In the case of IUA/EIUA the primary interest is of course in sharing information, experience, historical, current and future information about IDMS and the developer tool kit (ADSO, OLQ, Culprit, DML programming in Assembler and Cobol, IDMS/SQL and so forth). Of secondary concern is how we pursue our primary objective in a cost effective, entertaining, and easy to use manner.

At the moment there is some lively discussion on IDMS-L which is being cross-posted to our Community web presence at <https://communities.ca.com/web/ca-idms-iua-eiua-global-user-community>. This discussion is not being cross-posted to <http://iuassn.org> because there is no facility to do so. I would encourage all members who have an interest in how your IUA/EIUA User Community is represented in the nether world of the internet to participate in this discussion.

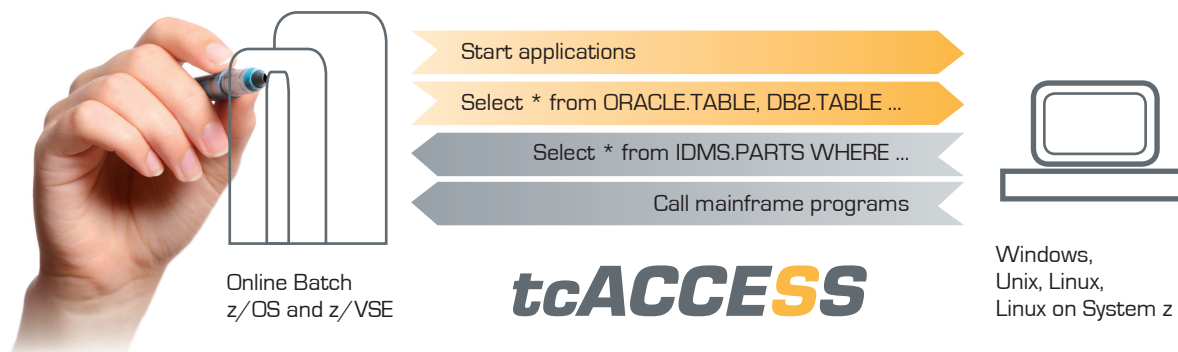
It does not matter if you use IDMS-L or the Community web site – your posts will be read and cross-posted. I can not manually cross-post every comment, and I will not cross-post any comments that do not adhere to IDMS-L rules, which are off topic, or which reflect an individual’s

(continued on page 3)

Bi-directional Integration and Data Synchronization for IDMS and Open System RDBMS.

Bi-directional Integration

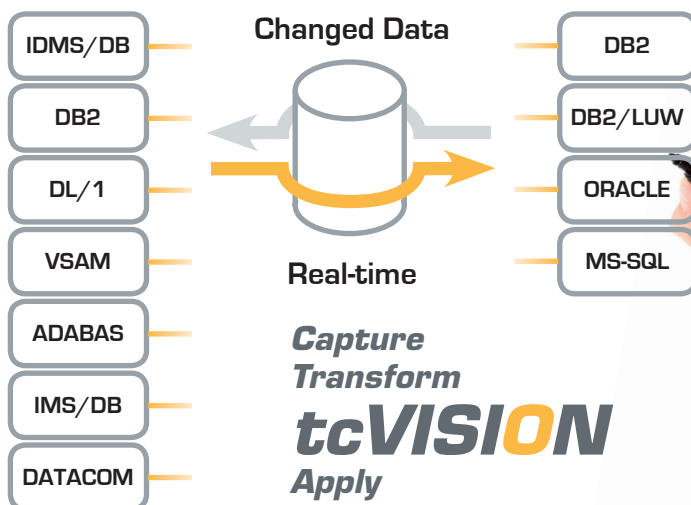
Direct SQL based access to IDMS/DB from JAVA, .NET, MS-WINDOWS, UNIX/LINUX ...



Data Synchronization

IDMS/DB data replication, synchronization, exchange in real-time, event driven and scheduled.

IBM Mainframe



B.O.S. Software offers solutions for IBM-Mainframe integration and data synchronization:

- NO interface programming
- NO performance impact
- MORE transparency
- Platform and system independent usage of data and programs
- Reusability of information
- Low maintenance efforts
- More flexible IT structures
- Reduced costs

tcVISION automatically captures and propagates changes applied to IDMS/DB, DB2, VSAM, IMS/DB, DL1, ADABAS, DATACOM/DB, DB2 LUW, ORACLE and SQL Server.

B.O.S. offers tcVISION: the unique solution for a platform and system independent data exchange in real-time between relational and non-relational data sources.

For more detailed information contact



Software
Service und
Vertrieb GmbH

Münchner Strasse 17
D-85540 Haar, Germany

Phone: +49 / 89/4619930
Fax: +49 / 89/468715

Mail: sales@bossoftware.com
Internet: www.bossoftware.com



MESSAGE FROM INTERNATIONAL CHAIR

By Linda Casey

Two years ago, I found myself in the brave new world of graduate school. After a lengthy absence from higher education, I'd hit a road block in a professional endeavor. I wanted to explore moving my classes online, but most of the online learning I'd experienced in the corporate world left me flat and bored. I knew there had to be a better way. I found and enrolled in a certificate program at Cal State East Bay to learn more about online teaching and learning. After four classes and a certificate, I wanted more and returned for a final six classes and the achievement of a MSED. This unique program not only taught me the pedagogy of online teaching and learning, but we literally "ate our own dog food"; that is, all classes were taught online. My cohort was comprised of students from not only California, but also from across the U.S. and around the world.

Through this experience, Web 2.0 became more than words on a page, but rather part of my everyday life. Wiki's, blogs, and discussion boards were an integral part of learning. This is where we exchanged ideas. This replacement for the face to face learning world I'd been so comfortable in for years was confusing and frustrating at first. I was an old dog learning new tricks. Eventually the new world became my world.

One important lesson I walked away with was the importance of community for this type of learning to take place; the respectful exchange of ideas, the elevation of a thought from knowing to doing, and the use of tools as a means to get there. Not technology driven education, but education enabled by technology.

But technology is tricky, it can move faster than we're comfortable with. Many of us in the IDMS community, are survivors of first-generation corporate IT; using punch cards, unit record equipment, moving into machine language, assembler, COBOL, IDMS and on. The next generations of IT professionals stand on our shoulders and are where they are because of where we've been. But they have never known a world without cell phones, they text at the speed of light, and find email passé.

Our challenge as charter members in the world of technology is to remain curious; to give freely of our wisdom and to listen and learn. The power of IDMS as a technologically superior data store need not die with us. To fully engage in communication with younger IT professionals, we need to embrace their communication modalities... we need to meet them where they gather. As a vital community freely exchanging ideas, we can continue to learn from one another, blending old and new in exciting ways.

Editorial.. cont'd from page 1

opinion to flatly refuse to explore new ways of interacting with other Community members. This is not constructive and does not further the discussion.

Having said all of that – I am pleased to be able to present for your enjoyment and information - another issue of "IDMS Connections". Many thanks to Richard Weiland from CA, and Kay Rozeboom from the User Community for their contributions.

That's all there is – because there is no more!

Gary Cherlet

<mailto:gary.cherlet@sa.gov.au>

Justice Technology Services (Department of Justice - South Australia)

IUA Board Member Responsible for IDMS Connections

President Australian IDMS Database User Group (OZIUA)

"IF IT AIN'T BROKE, YOU BETTER WORRY ABOUT FIXING IT": ONE GRUMP'S OPEN LETTER TO THE IDMS COMMUNITY.

By Don Casey

"If it ain't broke, don't fix it". This phrase came up in recent discussions about IDMS-L. This letter is **not** about IDMS-L (maybe NEXT Connections), but that phrase started me thinking about the shrinking status of IDMS in today's IT landscape. What follows shouldn't be shocking to anybody, but we don't talk about this much so let's be frank and honest with each other. I expect some of you will disagree with some or all of this.

One could argue that in the mid-1980s, both Cullinet and the IDMS user community as a whole felt that IDMS was a technically superior product, especially in comparison to the not-fully-baked relational offerings from Oracle and IBM (among others). IDMS didn't need fixing. While correct at that moment, that opinion missed the point that relational systems were going to improve and gain mind share among the decision-makers in IT. Cullinet also didn't foresee the rise of distributed platforms, and/or didn't focus on the right ones. Cullinet did not react to these sea changes in the market in time, nor correctly. Fully relational IDMS: in the pipeline, but by the time CA delivered it that ship had sailed. IDMS VAX? Right direction, wrong platform. UNIX and/or Windows IDMS versions would have been nice; but those didn't happen and here we are.

The decision makers in the user community were NOT the DBAs; they were IT management, application developers, and IT architects. These were the people who embraced the new platforms and relational technology, and these remain the decision makers in most organizations today.

(continued on page 4)

Even today one can argue that IDMS has superior operational characteristics, including performance, stability, and low resource consumption, than many/most relational systems. The problem is, those characteristics rank so far down the list of needs of those developing new systems as to require a microscope to see. The metrics an application project team is most interested in are ease of development, flexibility and time-to-market. As long as a DBMS is stable enough and can be made to perform acceptably, the ability to quickly produce a new application will trump operational considerations every time. We have lost the 'Technically Superior' discussion. Our only option now is to not lose the whole war.

This community understands (or should) how IDMS can be made to exist in a modern, distributed application, by serving as the back-end datastore and/or repository of business rules. Distributed access, SOA, function calls, etc. Until/unless this community takes and sells that message to their respective architects and application teams, with an emphasis on the value to the enterprise of leveraging the investment in current data, the erosion of the IDMS installed base will continue.

You are the people who are in a position to 'save' IDMS in your shops. CA can't do it, at least without your engagement. On the other hand, you can't do it without CA helping you promote this (CA please take note).

Only by engagement with your application leaders can this community build a vision among the decision-makers in your organizations that includes IDMS as part of the technical future. Only by helping them understand how use of existing data can actually shorten time-to-market for new applications, while minimizing risk and data migration/conversion costs, can they make informed choices. Nobody else can or will do this for you.

Legacy modernization projects are going to continue to fix what ain't broken, your choice is whether you want to attempt to influence the nature of the modernization that takes place in your environment.

I realize many of you have, are, and continue to fight these battles, I wish you well. For those who haven't yet; you should weigh your options. Either you take action, or you can let events inevitably unfold. If you choose to attempt to elevate the mind-share of IDMS among your decision makers, CA should be able help you with material, and can likely make presentations to the right people in your organization. You will have to identify those decision makers, work with CA on how to position an IDMS future that makes sense for your shop, and assist them in gaining access to those 'right people'.

Don Casey; Run Right, LLC

IDMS BUSINESS VALUE CORNER MARCH 2011

By Greg Beedy

Business applications are the precious cargo of database engines. The strong and powerful database engine speeds the application data to its business process destination reliably and securely. Apps are like the paying customers on planes or trains – generating revenue, driving the business. Without application and data assets, the database engine is out of work.

Knowing how important the relationship between our applications and IDMS is, in August 2010 we asked the IDMS-L list to tell us about their IDMS applications. The questions were:

We are interested in the longevity of IDMS applications ... when did your current IDMS apps go into production? Are they old enough to drive? To vote? To go to a 20th reunion? To remember the 80s? What do they do? And are you/they planning application retirement?

Thanks to all of you that responded to IDMS-L and under the radar. There were common threads, notably that many of your IDMS apps still running were first implemented in the 1980s and they still power your organization. Many of you also said while there is always talk of retiring apps or replacing them, typically you've been hearing that for years and not many – if any – have been replaced. Applications are always changing, and some have been retired – often replaced by packages – but in some cases the IDMS application portfolio is expanding.

A sampling of the stories we received follows.

- Del Bartlett, DBA at the State of New Hampshire reports: "The Department of Safety for the State of New Hampshire first used IDMS in 1984. I still have the original research documentation that was used to start the business process. It has been added to, in both good and bad ways. It is a mission critical system that now has Oracle and SQL Server interfaces, and all bi-directional. It was originally 14 subsystems. In 11 years they have migrated 4 of the subsystems to Oracle. Now there are 13 subsystems left. Yes, it grew."
- John Collins from SAIC in San Diego reports: "The IDMS applications here are financial applications, and the first one went into production in approximately 1984. For over 17 years, they've been saying the applications would be retired in two years! It is getting closer but I have my doubts."
- "They started doing IDMS apps in 1980 and still do today. They have been migrating off IDMS with a target of five years or so... since 1991. They have dropped a few IDMS systems over the years ... and added three for every one dropped!"
- CA IDMS world headquarters is in Framingham Massachusetts, so all us here use the Massachusetts

(continued on page 5)

Registry of Motor Vehicles every year when we register our cars, renew our licenses, etc. The RMV's ALARS system started in 1984. Major revisions were done in about 1998. The system is continually being enhanced and holds 750 million records. Dick Pierce reports that "they" say they will be converting off, but they've been saying that for years and nothing is designed yet."

- Are they old enough to drive? "Yes, and drive the business they do."
- "Our apps all date from the 80s and are old enough to drink. That may come in handy as the mainframe is scheduled to be unplugged next year."
- Martin Wieland from Neckermann in Holland reports: "We have applications that date back to the late 1970s. Most of them are critical, most online apps are coded in ADS, with batch in COBOL and mostly developed in-house. We are the #2 company in our space and the applications vary from our customer information system, to purchasing systems, project management and many others."
- There are no immediate plans to retire any legacy applications. Our biggest application, customer information system is being modernized now using Java and WebSphere. IDMS is still our major, mission critical database platform. We have some satellite systems that run on other platforms, but our operational data still resides in network IDMS databases. We are still building new databases and CV's.
- I think we are lucky: most of our IT staff, including management, grew up with mainframes and IDMS. We know the value of legacy systems and we have seen so many conversion attempts around us go terribly wrong."

Under the radar, edited out of respect to the authors and their organization:

- "Our first IDMS systems went live in the late 1980s. Many of them were home-grown and they run pretty much most of our business. In 1998 we converted our customer information and billing system to IDMS."
- I work as a DBA for two large IDMS systems that went into production at the late 1980's and early 1990s. Both have been given a face-lift with custom-made browser GUIs. There are no plans for retirement and the systems are being constantly developed.

We'd like to tell your story and get more management attention on the heavy lifting that IDMS business applications perform in your organization. Please contact me greg.beedy@ca.com with your story.

DATABASE PERFORMANCE ANALYSIS

By Dick Weiland

Database tuning in a run-time environment usually focuses on the definition of the I/O subsystem environment. Unfortunately little attention is given to the database structure once the initial design is complete. This paper will examine the statistics that are available and how they correlate run-unit performance to the database design.

Establishing the test environment

The purpose of run-time analysis is to evaluate the level of database activity generated by application logic given the physical structure of the database. For the analysis to be meaningful there are four major items that must be considered.

The evaluation should be focused on the most critical transactions.

During the online day the most critical transactions will typically be the highest volume transactions. It is certainly more important to achieve sub-second response times for transactions that will run millions of times during the online window than for a transaction that will typically be executed 100 times. In the batch environment long running jobs that are critical to the completion of the batch window should be the focus of the analysis as opposed to small or infrequently run batch jobs. Hopefully the physical database design was created with these same critical transactions in mind.

Results must be compared against expectations.

These standards may be developed by accumulating the desired statistics before a planned change such as a database design or application modification, type of DASD, CA IDMS Release level, or operating system level. For new applications it may be a matter of manually walking through an application's logic and estimating the number of records accessed, reads/writes, and other statistics that may be relevant based on the portions of the database that will be processed. This will require reasonably accurate estimates of the number of record occurrences within the database, the expected lengths of sets, and the anticipated page sizes to determine cluster overflow. For example if a program will walk a single occurrence of a VIA set owned by a CALC record and you expect each occurrence of this type of set to contain 10 members you might expect a transaction to access 11 records and generate one read if you anticipate all 10 member records to be able to fit on a single page. However if the same set was a non-VIA chained set you would estimate that the program would access 11 records and generate 11 pages read. Based on the run-time statistics you will be able to determine whether your expectations are being met, or not. This discussion will help you to identify changes so that your expectations can be met.

(continued on page 6)

Test environments must simulate true production environments.

Although it may not be practical for a test database to be as large as a really huge production version the general characteristics of the databases should be the same. If your test environment is only 10% of production, reduce your test data to 10% of the expected production volume. However if you expect the average length of a set to be 100 members in production then occurrences of the same set in the test environment should also contain an average of 100 members. Page sizes should also be the same between the two environments so that clustering in test will accurately mimic production and the percentage of space used in the test database should be similar to that in production. By keeping these two environments similar you will generate statistics that will be accurate on a transaction basis and can be reasonably extrapolated to estimate overall run-times for things like long running batch jobs.

Do not allow buffering to mask potential problems.

Physical I/O is probably the most costly resource that one can consume. If you create buffer pools in your test environment that are too large then you have the potential to load more pages into buffers than would be expected in a production environment. Doing something like walking long non-VIA sets, which would be expected to generate large numbers of reads, may be masked by the database becoming resident within the buffer pool.

Statistics overview

The following list contains the standard database or run-unit statistics that CA IDMS accumulates for every active run-unit. Those statistics marked with an '*' are worth considering for run-time analysis.

01 DB-STATISTICS.			
03	DATE-TODAY	PIC	X(8).
03	TIME-TODAY	PIC	X(8).
* 03	PAGES-READ	PIC	S9(8) COMP.
* 03	PAGES-WRITTEN	PIC	S9(8) COMP.
* 03	PAGES-REQUESTED	PIC	S9(8) COMP.
* 03	CALC-TARGET	PIC	S9(8) COMP.
* 03	CALC-OVERFLOW	PIC	S9(8) COMP.
* 03	VIA-TARGET	PIC	S9(8) COMP.
* 03	VIA-OVERFLOW	PIC	S9(8) COMP.
* 03	RECS-REQUESTED	PIC	S9(8) COMP.
* 03	RECS-CURRENT	PIC	S9(8) COMP.
03	CALLS-TO-IDMS	PIC	S9(8) COMP.
* 03	FRAGMENTS-STORED	PIC	S9(8) COMP.
* 03	RECS-RELOCATED	PIC	S9(8) COMP.
03	LOCKS-REQUESTED	PIC	S9(8) COMP.
03	SEL-LOCKS-HELD	PIC	S9(8) COMP.
03	UPD-LOCKS-HELD	PIC	S9(8) COMP.
03	RUN-UNIT-ID	PIC	S9(8) COMP.
03	TASK-ID	PIC	S9(8) COMP.
03	LOCAL-ID	PIC	X(8).
03	FILLER	PIC	X(8).

Figure 1

DB statistics are accumulated and reported at the system, task, and run-unit levels and it is important to understand what is being counted at each level. The highest level of statistic is at the system level which is accumulated for an entire Central Version (CV). System level statistics can be displayed online using the 'DCMT DISPLAY STATISTICS SYSTEM' command or generated by running batch report SREPORT 03. These statistics

are a total of all of the database activity that has occurred during the life of the CV and are relatively worthless when attempting to analyze the performance characteristics of a single task or program.

Task level reports provide access to the second level of DB statistics. Installations with CA IDMS Performance Monitor installed have the ability to look at these numbers online using the Application Monitor (PMAM) or in batch by running reports PMARPT17 and 18. Sites without Performance Monitor would typically use batch reports SREPORT 5, 6, 7, 8, or 9. These reports can take their input from the CA IDMS DCLOG for the CV on which the testing occurs when 'STATISTICS TASK WRITE' is specified in the sysgen. With 'STATISTICS TASK COLLECT' specified, and with the "write to SMF" option specified in #PMOPT for Performance Monitor, the reports can be run from SMF. Task level DB statistics contain the accumulation of all of the run-units that have been executed during the life of the reported task. This will include all run-units for which the application program(s) issued a BIND RUN-UNIT as well as any run-unit activity performed for queue, program load, and message activity. Task level statistics will provide a good representation of the overall database work required to complete the entire task, but the inclusion of non-application generated database activity still makes it difficult to analyze the performance of the application's database design.

For the purposes of analyzing the performance of an application against its database structure the lowest level of statistic provides the most meaningful values. Run-unit statistics will reflect the database activity for a single run-unit since the run-unit's BIND RUN-UNIT command until the time at which the statistics were requested. Run-unit statistics can be requested by the application program by issuing the ACCEPT FROM IDMS-STATISTICS command but it is then the program's responsibility to output them in a usable manner. This can be useful when trying to isolate the database activity generated within a particular section of a program's code. Run-unit statistics are also written to journal checkpoint records and can be accessed from archive journal files using journal reports JREPORT 2, 3, and 4. Many sites suppress the journaling of retrieval run-units to reduce the amount of data generated so it may be necessary to temporarily change the CV's sysgen to specify JOURNAL RETRIEVAL on its SYSTEM statement so that critical retrieval programs can be reviewed.

Individual statistics

Since physical I/O is the most expensive resource used by a task the two most important statistical fields provided within the DB statistics block are **PAGES-READ** and **PAGES-WRITTEN**. Even if a site does not have specific guidelines as to how many physical I/Os a task is permitted, especially in an online environment, it is necessary for a database designer to have an idea of how many I/Os are to be expected per transaction and how many are really being generated by an application. **PAGES-READ** identifies the number of times a physical

(continued on page 7)

Database Performance Analysis... cont'd from page 6

I/O operation was required to bring a page into the buffer pool. This is a number that can easily be skewed when testing is performed with a buffer pool that is too large for the test database size and task volume relative to those components in the production environment. PAGES-WRITTEN reflects the number of times that an update of a page results in the page's 'must-write' switch being turned on. The actual physical I/O may be experienced by this task, but could actually be forced by another task. This would be the case if the second task has a need to reuse the space in the buffer before the updating task has had an opportunity to issue the physical I/O. However, the estimated value that will provide a reasonably accurate estimate of the number of physical write I/Os that a particular program may require.

The two major causes of excessive reads or writes are navigating non-VIA or CALC-to-CALC sets. When these structures are involved one must frequently make the decision to reduce the I/O requirements of the affected critical transactions and possibly increase the needed I/O for those transactions deemed less critical.

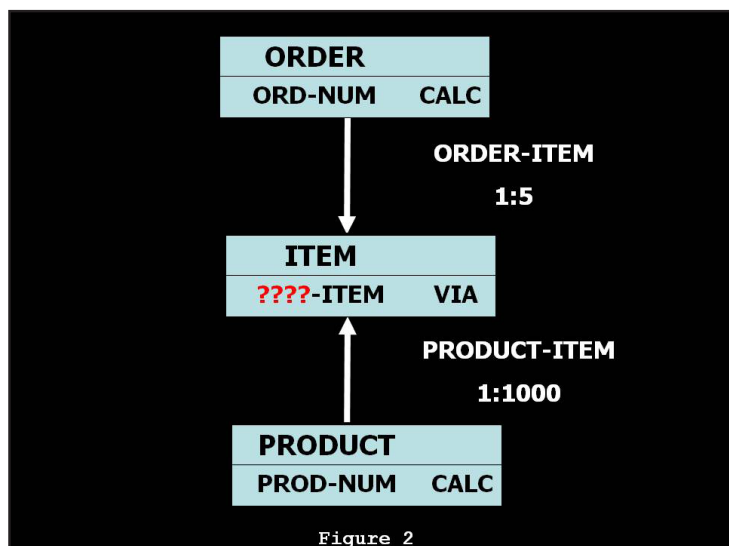
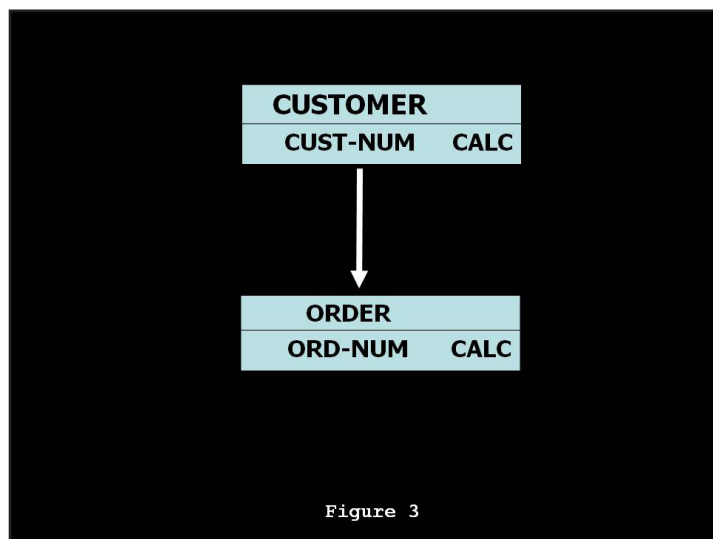


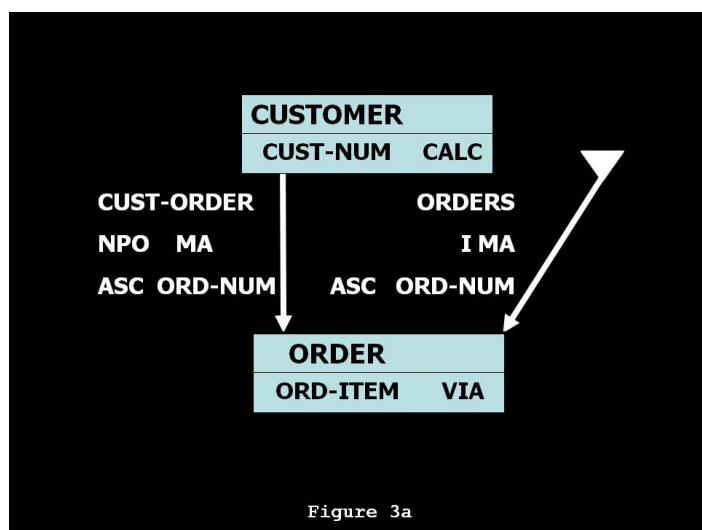
Figure 2 represents a database where the ITEM record is a member that participates in both the ORDER-ITEM and PRODUCT-ITEM sets. Specification of which set will be used as the VIA set will have a significant impact on the number of physical I/Os needed to process related transactions. If the transactions that navigate the ORDER-ITEM set are executed frequently and those that walk the PRODUCT-ITEM set are rare it is desirable to select the ORDER-ITEM set as the VIA set. This would allow the tasks walking the ORDER-ITEM set to get all of their required data in an average of 1 I/O if the projected number of ITEMS within the set of 5 is accurate. Tasks needing to walk the entire PRODUCT-ITEM set could invoke up to 1000 I/Os. If the amount of I/O needed to walk the PRODUCT-ITEM set is unacceptable then further design work on the physical implementation of the ITEM records would be in order.

Sometimes the logical design of the database and the need to directly reference two record types directly by a key value leads to two CALC records being related through a set relationship. Since CALC records are randomized

across a database area the navigation of this set will typically result in one physical I/O for every member record accessed. Figure 3 provides an example of this type of relationship.



Sometimes there may be multiple critical transactions that use the records involved in this type of structure and it may be necessary to inflict a small overhead on one type of transaction to create a larger benefit for another transaction. If transaction 'A' needs to access a CUSTOMER and all of its associated ORDER record occurrences that task will incur one physical I/O for each ORDER in the set. However transaction 'B' needs to directly access an ORDER based on that ORDER's calckey. Both are considered critical transactions but testing of transaction 'A' shows that it is generating too many physical reads. A solution may be to redefine the ORDER record as a VIA record and cluster it around its owning CUSTOMER record occurrence. The direct access through a key value for transaction 'B' can be supported by adding an index to the ORDER records. Although this will add some I/O to transaction B it should be a manageable amount that may still allow those transactions to be processed in the allowed amount of time (see Figure 3a).



(continued on page 8)

PAGES-REQUESTED is a statistic that identifies the number of times that the I/O sub-system is called to access a database page. If the page must be physically read the I/O sub-system will increment the value of **PAGES-READ**. As a result **PAGES-REQUESTED** and **PAGES-READ** have a direct correlation to each other. This relationship will be discussed when various performance related ratios are examined. However it is important to note that the execution of the exact same test in a CV environment will result in a larger value for **PAGES-REQUESTED** than will be generated in a local mode test. This is due to the CV releasing internal buffer locks at the completion of each DML command to provide for better throughput of run-units. This is not a consideration in local mode since there is only one active run-unit in the CA IDMS environment.

The **OVERFLOW** and **NO-OVERFLOW** values report on the number of times that records could not be stored on the page to which they were initially targeted. These statistics are kept separately for **CALC** and **VIA** record types. To understand what these numbers mean one has to understand how CA IDMS determines a record's target page. For a **CALC** record the value of the record occurrence's calckey is hashed to calculate the target page. For **VIA** records the target page is typically that of the record occurrence that is current of the record type's **VIA** set. For example, if a program accesses the owner of a set and then stores the member record the member's target page will be the page on which the owner resides. When a second member is then stored into the database, the page on which the previous record was stored will be the target page for the second occurrence. If each of these records is successfully stored on their target page the **VIA NO-OVEFLOW** field would be incremented for each store operation. However if the first record occurrence did not fit on its target page the **VIA OVERFLOW** field is incremented and the page on which the record was finally stored will be the target page of the second record. Assuming there is space on this page for the second record the **VIA NO-OVERFLOW** field is incremented.

The overflow statistics represent the number of overflowed record occurrences generated by the run-unit being measured and do not represent the total overflowed occurrences within the entire database area. The **IDMSDBAN** utility can be used to give you a better idea of the overall condition of the database. However consistent numbers of overflowed record occurrences over what your pre-test estimates might suggest as expected should be examined. High values for **CALC** overflows might suggest hotspots within the database or that the database is starting to run short on available space. Hot spots may be eliminated by changing the page range of the **CALC** record's area slightly which can alter the distribution of the **CALC** records within the area.

Higher than expected numbers for **VIA** overflows usually indicates some type of problem with decisions that were made regarding record clustering. Clustering is the practice of storing occurrences of one record type in close physical proximity to occurrences of another record type to reduce I/O requirements. Perhaps the greatest impact

on clustering efficiency is the selection of an area's page size. If your initial analysis of the database calls for a cluster size of 10,000 bytes the selection of a 4K page size should be expected to result in overflowed record occurrences across multiple pages. You might consider selecting a page size closer to 12K to reduce the number of overflows encountered.

The most common problem encountered with **VIA** clustering is that the clusters for owner records are just too large regardless of the page size selected. This may be due to the fact that there are just too many member occurrences of a single member record type per owner to fit on a page or that there are too many **VIA** record types related to the owner. In these cases you might consider moving some of the member record types to a different area than the owner. This provides the ability to define a member area with a much larger page size than you might want to define for the owner area. Many sites will move the less frequently used record types into the new areas and retain the frequently used record types in the same area as the owner record.

When calculating your expected cluster size for an owner record type, do not forget to include the amount of space needed to maintain any user-owned indexes. The **SR8** records making up the index structure for a user-owned index are clustered with the occurrences of the set's owner record. Large index occurrences can have a devastating impact on the efficiency of an area's data clusters. When large user-owned indexes exist for a record type, you may wish to convert the indexes into a single system-owned index which can be moved to its own area. This may require the inclusion of the owner's key within each member record occurrence to provide the ability to differentiate between the member records for each owner occurrence. Figure 4 gives an example of such a modification.

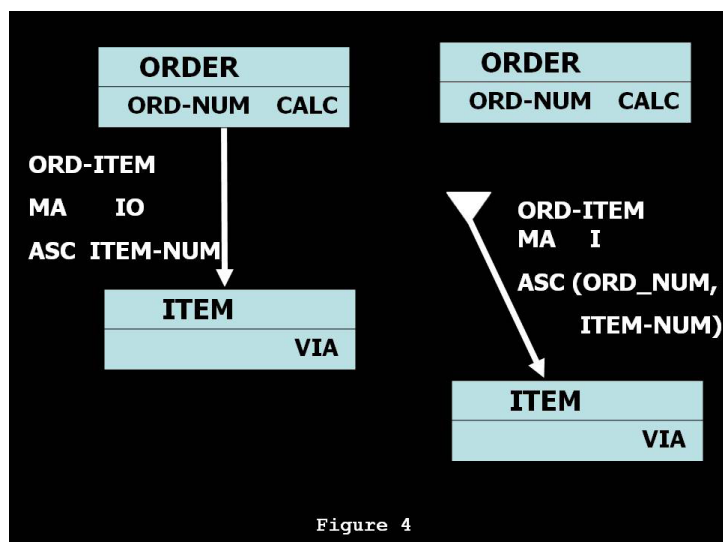


Figure 4

RECORDS-CURRENT and **RECORDS-REQUESTED** provide an insight into how much work the DBMS must do to service a DML request once the required data has been brought into the CA IDMS buffer pools. **RECORDS-CURRENT** roughly equates to the number of functional DML commands issued by the run-unit. For example if the run-unit issued 21 **OBTAIN**

(continued on page 9)

commands between the BIND RUN-UNIT and FINISH commands and all of the OBTAINs successfully returned a record to the program one would expect RECORDS-CURRENT to be reported as 21. RECORDS-REQUESTED reports on the number of record occurrences the run-unit had to access to accomplish the functional request of these DML commands.

To properly use these statistics the person analyzing the tests should be familiar with the type of DML commands being issued by the application program and the structures being navigated. For example a program that does a single OBTAIN CALC would be expected to ideally have a RECORDS-CURRENT value of 1 and a RECORDS-REQUESTED value of 2. The 2 for RECORDS-REQUESTED could be attributed to one access of the SRI record on the data record's target page and one access for the actual data record. Higher values for RECORDS-REQUESTED would be caused by multiple data records with a location mode of CALC targeting to the same database page. Very high values could indicate a hot spot on the database that might be alleviated by a change in the number of pages in the area which would change the CALC record distribution.

The processing of long set relationships will frequently cause inflated numbers for both the RECORDS-CURRENT and RECORDS-REQUESTED statistics. When only a subset of the member records of a set relationship are needed it may be helpful to add an additional set relationship between the two record types to reduce the number of records that must be accessed by higher priority transactions. Figure 5 shows the addition of an OPEN-ORDER set between the CUSTOMER and ORDER records to reduce the number of records that must be accessed by those transactions only interested in orders that have not yet been filled for a particular customer. This should reduce both the RECORDS-CURRENT and RECORDS-REQUESTED counts for those transactions.

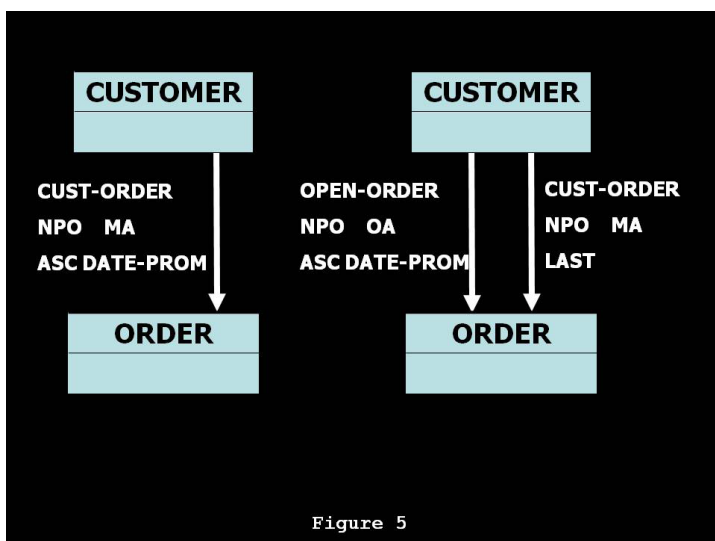


Figure 5

Long sorted chain sets can often result in large numbers of RECORDS-REQUESTED while the value of RECORDS-CURRENT is relatively small when random reads are made against the set using the set's sort key. With a sorted chain set CA IDMS must walk, on average, half the set's members to locate the desired record. Each

record checked results in the RECORDS-REQUESTED field being incremented to reflect the amount of work performed by the DBMS. If the bulk of the DML requests will be expected to be satisfied by records at the end of the set, consider reversing the sequence of the set so that the desired records will be found at the set's beginning. However, if the records are expected to be found throughout the set, consider changing the set from a chain set to a user-owned index set. Though this means that you must always access an index structure to locate the desired record occurrence, this frequently results in fewer RECORDS-REQUESTED than having to walk each data record preceding the desired occurrence.

Finally, a high RECORDS-REQUESTED value can result from the lack of proper PRIOR or OWNER pointers on a set relationship. Missing PRIOR pointers can result in extra record occurrences being accessed for DISCONNECT and ERASE verbs or MODIFY verbs where the sort key of a set is being changed. Execution of FIND/OBTAIN OWNER verbs can also cause extra record occurrences to be accessed if the set involved has not been defined with OWNER pointers. If any of these verbs are used by the program being analyzed, then the missing pointer types should be added to the involved sets.

FRAGMENTS-STORED reports the number of variable length record fragments that were stored by the monitored run-unit. A variable length record is any record that is defined with data compression or whose element structure contains an OCCURS DEPENDING ON clause. As variable length records are modified such that their length increases it can be expected that some fragments will be generated. However, if the run-unit only stores new occurrences of the variable length records, or if the value of FRAGMENTS-STORED seems to be excessive, there may be a problem with the database.

As the record's database area fills the number of fragments that are generated can be expected to increase. If the area is greater than 70% utilized the solution may be to enlarge the record's area. However the problem is frequently in the record's schema definition. To minimize fragmentation every variable length record's schema definition should contain the clauses MINIMUM ROOT IS RECORD LENGTH and MINIMUM FRAGMENT IS RECORD LENGTH. In addition each area containing variable length records should specify a PAGE RESERVE to reserve space on each database page for the expansion of any variable length records residing on the page. A PAGE RESERVE will not eliminate fragmentation but will delay its onset.

You might also review the variable length record to see if defining it as variable length is warranted. If the record is initially stored using only a small number of the repeating group occurrences but will typically grow to use close to the maximum number of repeating occurrences you may be better off defining the record as a fixed length record. In a similar manner a record that is defined with data compression may be better defined as uncompressed if it initially contains long strings of repeating characters

(continued on page 10)

but over the life of the record occurrence will always be replaced by other non-repeating values.

The meaning of the **RECS-RELOCATED** statistic is dependent on whether the database is a non-SQL database or an SQL defined database. When monitoring a non-SQL database a non-zero value indicates that relocated records or variable length fragments are being brought back to their associated record's original target page. Relocated records in a non-SQL database can only be created as a result of running the RESTRUCTURE SEGMENT utility to expand the length of a record.

Although a positive indicator that relocated records or variable length fragments are being cleaned up, it is also an indicator that they exist and are causing additional processing overhead for the run-unit. An UNLOAD/RELOAD or REORG execution may be in order to remove these types of records.

When non-zero values for RECS-RELOCATED are found while monitoring processing against an SQL database it means that record occurrences are being relocated away from their original target pages. When the number of columns in an SQL table's definition is increased the actual size of the rows within the database are not increased until the next time that row is updated. If there is insufficient space on the database page for the increased space requirement the database record is relocated. Seeing consistent non-zero values for RECS-RELOCATED may indicate space shortages and an UNLOAD/RELOAD or REORG execution may be advisable to reduce the overhead associated with relocated records.

Ratios

Many of the database statistics that have been discussed are used together as ratios to aid in their interpretation and the analysis of the run-unit environment.

To this point the subject of buffer pools has been avoided but the first ratio to be considered provides a measure of the effectiveness of the buffer utilization. **PAGES-REQUESTED/PAGE-READ** should be used along with an understanding of the type of processing being performed by the run-unit. A run-unit that does primarily CALC retrieval is likely to see a 1:1 ratio of pages requested to the number of pages read due to the random nature of the process. However if the application is expected to read a CALC record and walk a VIA set that on average has 5 members, a value of 6:1 would be expected. In general large values are desired. A decrease in this ratio may indicate the introduction of some form of new buffer contention such as a new application being added to the environment or an increase in overflow conditions within one of the areas involved with the run-unit.

RECORDS-REQUESTED/PAGES-READ should be reviewed for those run-units that walk VIA sets and is a measure of the clustering efficiency of the database. The values of this ratio should be as large as possible but you must still view them with a knowledge of the expected VIA clusters. If the number of records per cluster is expected to be 20 and the member records are small you

might expect them to fit on a single page. A value of 20:1 would be anticipated. However if the member records are very large the initial design may have expected the set to span two database pages. In that case the expected value of the ratio would be 10:1. Decreases in the value over time would be an indication that overflow conditions may be increasing.

The amount of work performed by the DBMS on a per DML basis can be estimated using the ratio of **RECORDS-REQUESTED/RECORDS-CURRENT**. This ratio should be as low as possible. Run-units that perform key -valued retrievals against sorted chain or index sets, store records into these types of sets, or modify the fields used as the sort keys should be expected to have larger ratio values than those processes that walk entire sets using FIND/OBTAIN NEXT commands. Increasing values for the ratio could be an indication that these sets are becoming longer than anticipated and alternate structures may eventually be warranted. Increases for run-units that primarily perform CALC could be an indication that hot spots are developing in the associated areas.

The ratio of **NO-OVERFLOW/(OVERFLOW + NO-OVERFLOW)** should be reviewed for those run-units that store record occurrences into the database. A value of 1 for this ratio is an indication no overflow has occurred while values less than 1 indicate some level of overflow. As stated earlier some overflow is to be expected. If ratio values become smaller over a period of time it may be an indication that space utilization problems are developing. The overflow statistics are maintained separately for CALC and VIA overflow conditions. Pay particular attention to the ratio for the CALC overflows as they can have a greater impact on database performance as a CALC overflow may impact direct retrieval of all CALC records within the area as opposed to a single cluster of data.

Summary

The database's physical structure has the greatest influence on the number of I/O operations and the amount of CPU required by the DBMS to process an application's requests. Unfortunately it is usually the aspect of performance that is the least evaluated. Database statistics provide a means with which to measure the interaction of application programs and the database structure. However to do this successfully requires a knowledge of the application's critical transactions so the numbers can be properly interpreted. There should also be a set of expectations established by application designers so that over a period of time you are able to spot statistical trends that may be a signal of developing performance problems. If you do not have such documented expectations, establish a "baseline" of current values and monitor trends away from the baseline.

Dick Weiland is a Senior Systems Engineer for CA IDMS Level II Support working out of the CA Lisle office. He began working with CA IDMS in 1977 as a DBA and joined Cullinet in 1981 in the Field Support organization. He moved to Level II Support in 1988 and is responsible for the DBMS engine and various utilities.

FOS/WEB SERVICES AT THE STATE OF IOWA

By Kay Rozeboom

The State of Iowa uses a product from Formula OpenSoft called “fos/Web Services for IDMS” to both provide and consume SOAP web services. It has two components:

- The web service requester, which allows an ADS or DC-COBOL program to call an external web service.
- The web service provider, which allows an external program to call a web service in IDMS.

(Disclaimer: The software described in this article was originally developed at the State of Iowa. It is now owned and licensed by Formula OpenSoft.)

Web service requester

Figure 1 shows the flow of a web service call from IDMS. The green arrows depict the request, and the orange arrows depict the reply.

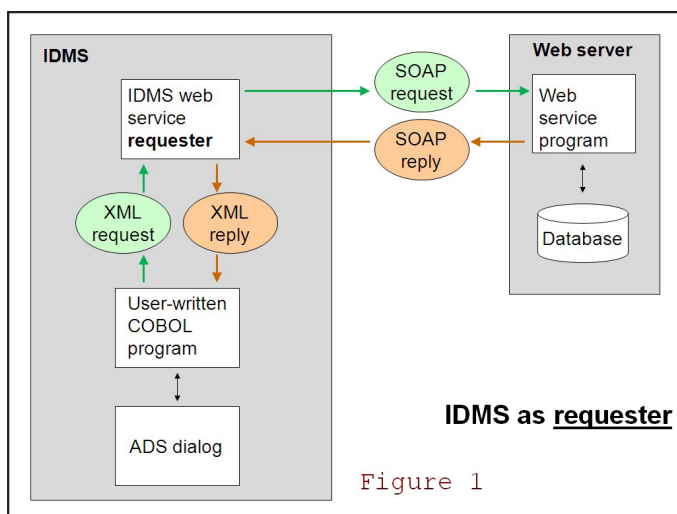


Figure 1

In this example, an ADS dialog links to a DC-COBOL program. The COBOL program formats the XML request and places it in a work record, along with some additional information such as the URL of the web service. It then links to the IDMS Web Service Requester, passing the work record.

The Requester packages the request into a SOAP message. It also performs some other functions, such as translating the message from EBCDIC to ASCII. It then sends the message to the web service, and waits for a reply.

After receiving the reply, the Requester translates it from ASCII to EBCDIC and extracts the XML reply from the SOAP message. It then sends the XML reply back to the calling program via the same work record that was used to make the request.

Web service provider

The web service provider follows a similar process, only in reverse. Figure 2 shows the flow of a web service call to IDMS. The green arrows depict the request, and the orange arrows depict the reply.

In this example, a web application sends a SOAP request message to the IDMS Web Service Provider. The Provider determines which web service has been requested, checks security, and extracts the XML request from the SOAP message, placing it in a work record. It then links

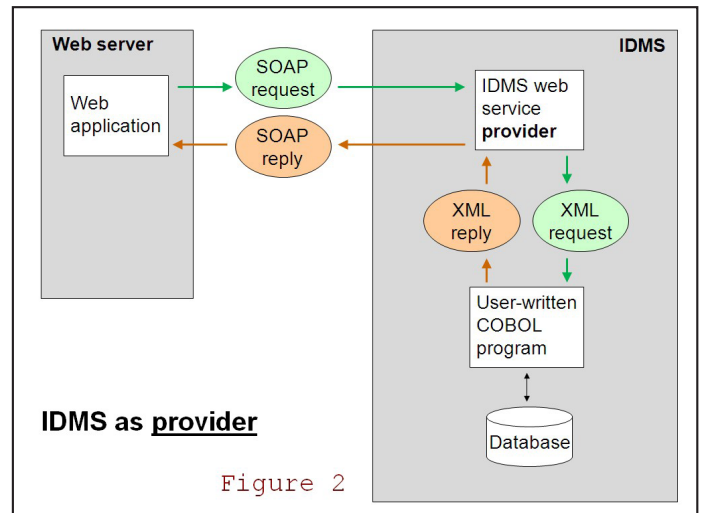


Figure 2

to the requested web service (which is actually a user-written DC-COBOL program), passing the work record.

The user-written COBOL program parses the XML request, and carries out the requested actions – usually reading and/or updating an IDMS database. It then creates an XML reply, which is sent back to the Provider via the same work record that was used to pass the request.

The Provider packages the reply into a SOAP message, then sends the message to the requesting program.

Additional points

IDMS Web Services uses the IDMS TCP/IP sockets feature. Therefore it requires an IDMS maintenance level of R16.2 or above.

With this product, your COBOL code must read and write XML messages. This can be much simplified by using the COBOL XML parser and generator, which are included with COBOL 3.1 (parser) and COBOL 3.3 (generator) and above. Please see my previous two articles for coding examples.

IDMS Web Services runs entirely within the IDMS address space. There are no workstation or server components to install.

This software does not create WSDL files. At our site, we use the CICS program DFHLS2WS to create the WSDL files for provider programs. There are also a number of workstation tools that you can use.

If you have the IDMS SQL Option, this product can be used to call external web services from batch and CICS programs as well.

IDMS Web Services in production at the State of Iowa:

External web services are called from IDMS via the Requester approximately 170,000 times per day. Response time is dependent on the speed of the external

(continued on page 12)

web service.

IDMS web services are called via the Provider 400-3000 times per day, depending on the time of year. (Most of our web access to IDMS uses IDMS Server, which has been available longer. IDMS Server is an ODBC driver for IDMS which requires the SQL option.) CPU and wait time within IDMS are less than for comparable ADS dialogs, because the mapping functions are being done elsewhere.

Here are descriptions of web services currently in production at the State of Iowa:

Requester:

- An ADS application reads and updates data in a SQL Server database, in addition to IDMS. The users do not know or care which parts of the data are stored in IDMS, and which parts are stored in SQL Server.
- A batch program loads data from a VSAM file into a non-mainframe DB2 database. (requires IDMS SQL option)
- A CICS program reads and updates data in a SQL Server database, in addition to VSAM. (requires SQL option)
- A web application which does not support web services accesses an email validation web service by making an SQL query to IDMS. IDMS makes the call to the external web service, then returns the response to the web application as an SQL answer set. (requires SQL option)

Provider:

- A public-facing web application reads data in an IDMS network database.
- An internal cross-department web application reads and updates data in an IDMS SQL-defined database.

"Kay Rozeboom is a DBA/Systems Programmer with the State of Iowa. She has 18 years of IDMS experience. Her special interest is in integrating mainframe data and applications with other platforms."

SUBJECT: IF – THEN

A wife asks her husband, a software developer, "Could you please go shopping for me and buy one carton of milk, and if they have eggs get 6."

A short time later the husband comes back with 6 cartons of milk.

The wife asks him, "Why did you buy 6 cartons of milk?"

He replied, "They had eggs."

IDMS-L RULE CHANGE

The IUA board has unanimously passed a motion changing the IDMS-L rules. The amended rules in their entirety are below and the addition to the rules is as follows:

"By participating in the IDMS-L or the IDMSVENDOR-L, you grant permission to the IUA, IDMS User Association, to use, publish, and distribute your posting for any purpose."

This change will allow us to use IDMS-L posts in Connections and to cross post to the causergroups forum. By continuing to subscribe to the IDMS-L you signal your acceptance of this rule change.

Terry Schwartz

IDMS-L Admin

To subscribe or unsubscribe

- If you become a subscriber, you agree to accept the LISTSERV's policies in this document
- To subscribe or unsubscribe, send an e-mail with the following:

Address: LISTSERV@LISTSERV.IUASSN.COM<<mailto:LISTSERV@LISTSERV.IUASSN.COM>>

Subject: Will be ignored so it isn't needed

Message: Action Listname Yourname

Action: Either SUBSCRIBE or UNSUBSCRIBE

Listname: Either IDMS-L or IDMSVENDOR-L, depending on which you are addressing

Yourname: Enter your first and last name if you want it to be available to other subscribers (this is optional)

Complete IDMS-L Rules

The IUA is happy to provide services to help IDMS professionals communicate.

Two LISTSERVs are supported:

- **IDMS-L** Intended for all postings related to just IDMS
- **IDMSVENDOR-L** Intended for conversations about third-party software.

If you are interested in just IDMS, you should only join IDMS-L. If you are interested in hearing how third-party software products and services might help you and your organization you should also join IDMSVENDOR-L

As the list owner the IUA reserves the right to post information relevant to IUA business and that supports CA IDMS and related products.

By participating in the IDMS-L or the IDMSVENDOR-L, you grant permission to the IUA, IDMS User Association, to use, publish, and distribute your posting for any purpose.

(continued on page 13)

How do you send postings to the IDMS-L?

First become a subscriber and then send e-mails to

- IDMS-L@LISTSERV.IUASSN.COM<<mailto:IDMS-L@LISTSERV.IUASSN.COM>>

How do you send postings to the IDMSVENDOR-L?

First become a subscriber and then send e-mails to

- IDMSVENDOR-L@LISTSERV.IUASSN.COM<<mailto:IDMSVENDOR-L@LISTSERV.IUASSN.COM>>

All IDMS-L and IDMSVENDOR-L Subscribers are expected to:

- Communicate professionally and courteously
- Although opinions are allowed and encouraged, all comments should be kept professional and respectful of all parties.
- Respond with only enough of the original message to convey original question
- Respond with messages that add value; sending "Me too" or "I agree" replies are unnecessary
- Stop auto replies - either unsubscribe, set yourself to Digest Mode or set auto-replies to not reply to listserv messages
- The List Server does not allow attachments. Attachments should be sent directly between users. If you believe this file would be helpful to a number of IDMS-L users, ask the IDMS-L Admin to post the file on the IUA Website.

All IDMS-L subscribers are expected to:

- Limit discussions to IDMS and the IDMS Tools products.

All IDMSVENDOR-L subscribers are expected to reply directly to sender for inquiries about

- Purchasing/Selling 3rd Party software
- Any offers of corporate/personal technical services
- Submit all discussions concerning third-party software products via IDMSVENDOR-L.

If mail is non-deliverable for 5 consecutive days or 20 attempts, the subscriber is automatically removed from both IDMS-L and IDMSVENDOR-L

Inappropriate Activity

Inappropriate activity by subscribers can cause the subscriber to be removed from the LISTSERV. The inappropriate activity includes but is not limited to these actions below.

IDMS-L does not allow

- Posts that compare products or discuss third-party software. Discussions of third-party software products should be confined to IDMSVENDOR-L.

IDMS-L and IDMSVENDOR-L do not allow

- Commercial advertising or promotion of service offerings
- Inappropriate language in any part of the posting including user names and addresses
- Comments about the administration of the LISTSERV (any comments should go to IDMS-L-Owner@IUASSN.ORG<<mailto:IDMS-L-Owner@IUASSN.ORG>>)
- Issue resolution that should be with the vendor
- Discussions concerning database conversions off of IDMS or database migrations to other platforms/databases.
- Disparaging comments against other list members, ethnic groups, companies, or types of companies.
- Job Postings

Dealing with Inappropriate Activity

- Depending on the severity or frequency of the problem, the subscriber's postings will either be banned from posting or monitored for a time period
- The time the subscriber will be banned from posting is up to the LISTSERV administrator and the IUA Board.

Appeals can be made to IDMS-L-Owner@IUASSN.ORG<<mailto:IDMS-L-Owner@IUASSN.ORG>> if desired

HOW MANY WAYS ARE THERE TO SAY YOU SHOULD DO THINGS RIGHT THE FIRST TIME?

There's a project management saying that "...if you have the time to do the job twice ... then you have the time to do it right the first time!".

Gary Cherlet on IDMS-L

In my 32+ years in IT, it's always been "there's never enough time or money to do it right, but there's always plenty of both to do it over".

Tim Kupin on IDMS-L

I thought it was "if you don't have time to do it right When do you have time to do it over" - at least that was what was written on my old boss Bonnie Macklin's white board in her office at DOL back in the early 1990's!

Jerry Roberson on IDMS-L

"FINGERPRINTING" LOAD MODULES TO IDENTIFY WHICH VERSION IS BEING RUN

Here is the "fingerprint" of an IDMS-DC Cobol program – using the LOOK task to see the program load module in XA re-entrant Program Pool:

look PROGRAM=DBUT051P

IDMSLOOK - OPSYS=z/OS Release 17.0 Service pack 1 tape GJH01B
PROGRAM=DBUT051P

was loaded From CDMSLIB DSN --> JIS.DEV.UTILONL.LOADLIB
Entry Point Offset +0 - Reentrant - AMODE 31 - RMODE ANY
27,920 Bytes in Load Module DBUT051P loaded at 1A368E00

	Module	Offset	Date	Time
	IDMS	+58A0	080805	1142
1A368E00 +0	47F0F028 00C3C5C5	00000220	00000014	.00..CEE.....
1A368E10 +10	47F0F001 98CEAC00	1A368EB6	00000000	.00.q.....
1A368E20 +20	00000000 00000000	90ECD00C	4110F038}...0.
1A368E30 +30	98EFF04C 07FF0000	1A368E00	00000000	q.0<.....
1A368E40 +40	1A36E368 1A368EAE	1A368E00	1A369AF8	..T.....8
1A368E50 +50	1A36EFD0 1A368ECA	00104001	00000008	...}.....
1A368E60 +60	C4C2E4E3 F0F5F1D7	F2F0F1F1	F0F2F1F4	DBUT051P20110214
1A368E70 +70	F1F1F5F1 F1F0F0F3	F0F4F0F1	04740000	115110030401....
1A368E80 +80	1000076C E088EC4C	00002000	50084B09	...%\h.<...&...
1A368E90 +90	01000040 08000000	00D80000	000000E4Q.....U

Breaking it down we get Program Name:

DBUT051P Date and Time down to hours, minutes and seconds of the compile: **2011-02-14-11.51.10** but we were left wondering what the rest of the "time" value was – this part: **030401 !**

We got the answer by looking at the Cobol Compile listing – here's the heading line from the compiler output:

PP 5655-G53 IBM Enterprise COBOL for z/OS 3.4.1

So it looks to us as though what we thought was part of the "time" (but which oddly enough was the same in every load module) was actually the Version of the Cobol Compiler!

Another mystery solved. At JTS we use the **#JISTAMP** macro to put a date:time "fingerprint" into the load modules. The result "looks" like this when using the LOOK task code:

look PROGRAM=DBUT052P
IDMSLOOK - OPSYS=z/OS Release 17.0 Service pack 1 tape GJH01B
PROGRAM=DBUT052P

was loaded From CDMSLIB DSN --> JIS.DEV.UTILONL.LOADLIB
Entry Point Offset +0 - Reentrant - AMODE 31 - RMODE ANY
2,832 Bytes in Load Module DBUT052P loaded at 1A410600

	Module	Offset	Date	Time
1A410600 +0	90ECD00C 18CF50D0	B05C47F0	C03EC1A2	..}...&}.*.0{. As
1A410610 +10	A2859482 93858440	96957A40	F2F0F1F1	sembled on: 2011
1A410620 +20	F0F2F1F4 6B4081A3	40F1F14B	F5F06B40	0214, at 11.50,
1A410630 +30	82A840D1 C4C1C7D7	C35BF200	00005860	by JDAGPC\$2....-
1A410640 +40	A3101266 4780C62C	58606058	12664780	t.....F.--.....
1A410650 +50	C63CD503 6000CAA8	4770C75C	417060DC	F.N.-..y..G*...-

This "fingerprint" even tells us who did the Assembly – in case there is a need to discuss anything with the developer 8-)

Cheers – Gary



International Chair

Company: Managing Member
Run Right, LLC
Email: lindajcasey@runrightllc.com



Secretary/Treasurer

Email Coordinator
Bob Wiklund
Company: Tiburon Technologies
Address: 17101 W. Gable End Lane,
Surprise, AZ 85387
Phone: 623 594-6022
Email: bob_wiklund@tiburontech.com



International Vice Chair

Contributed Software Librarian
Laura Rochon
Company: Ajilon Professional Services
Address: 22 Jolliet, St-Bruno,
Quebec J3V 4Z1 Canada
Phone: 514-943-8290
Fax: 450 441-6880
Email: l.rochon@videotron.ca



European IUA Representative

Steve Rundle
Company: British Telecom BT Group plc.
Address: PP2B33 Angel Centre,
403 St. John Street, London
EC1V 4PL UK
Phone: +44 (0)20 7777 6920
Fax: +44 (0)20 7777 6921
Email: steve.rundle@bt.com



Board Member

Terry Schwartz
Company: Perot Systems
Address: PO Box 269005
Phone: 972 577-3722
Email: terry.Schwartz@ps.net

Board Member

Craig McGregor
Axiom
craig.mcgregor@axiom.com

Board Member

Diane Montstream
Allen Systems Group
diane.montstream@asg.com

Board Member

Jan Rabaut
jan.rabaut@dexia.be

Editor

Gary Cherlet
Justice Technology Services
South Australian Department of Justice
gary.cherlet@sa.gov.au

Desktop Publishing

Rebecca Shaw 404 377-6982
shawrh@bellsouth.net

IDMS Connections is a bi-annual publication of the CA-IDMS Database and Applications User Association (IUA). It is designed to promote its members' objectives. *IDMS Connections* is not responsible for the opinions expressed by its writers and editors.

Information User Association

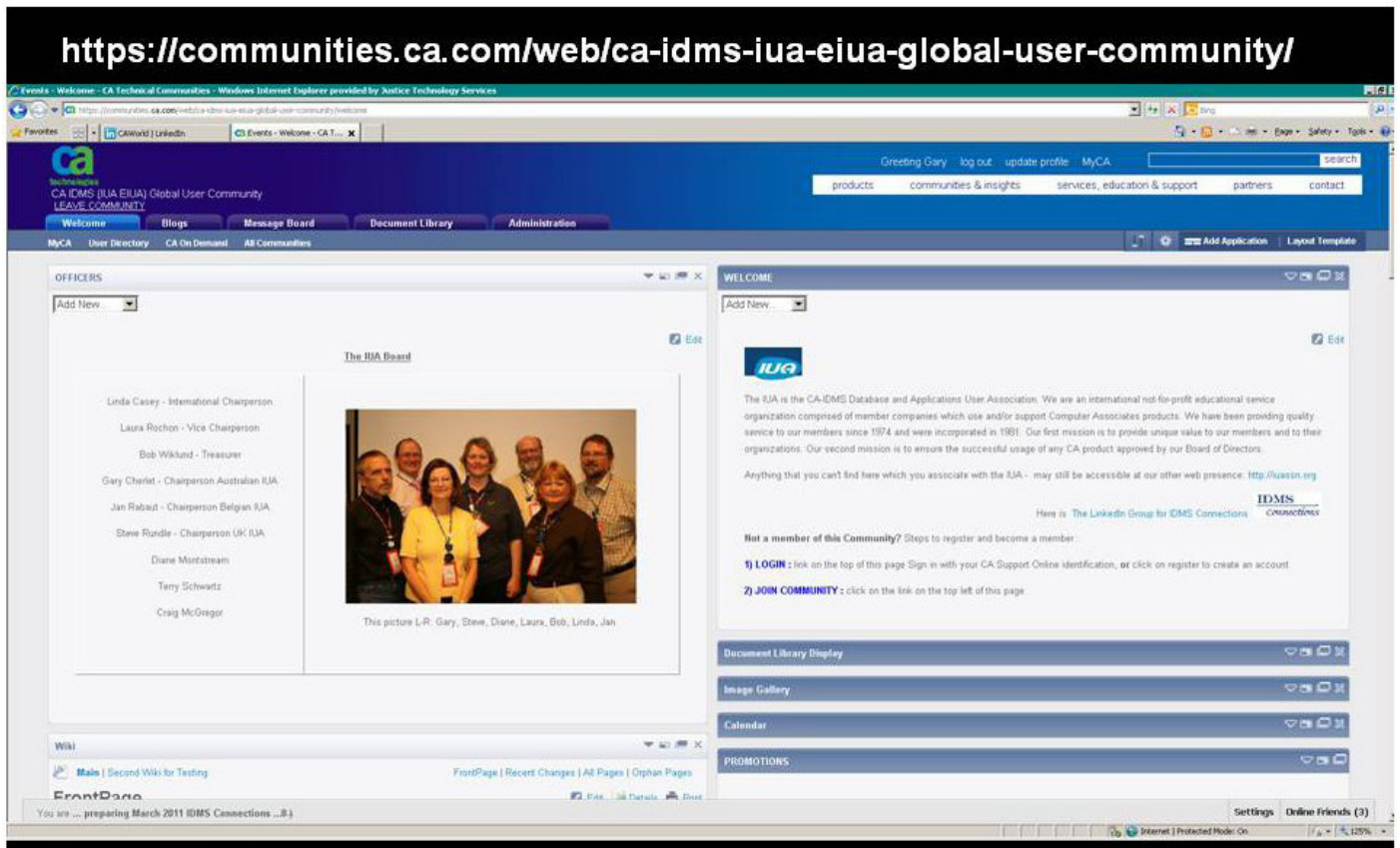
401 N. Michigan Ave.
Chicago, IL 60611-4267
Phone: 312/321-6827
Fax: 312/245-1081

Internet: iua@iuassn.org

Web: <http://iuassn.org>

IUA/EIUA GLOBAL USER COMMUNITY

As referred to in the editorial <https://communities.ca.com>



**DON'T FORGET
TO FORWARD
CONNECTIONS
TO OTHERS IN YOUR
ORGANISATION**

**SEND ARTICLE
CONTRIBUTIONS
TO GARY CHERLET AT:
GARY.CHERLET@SA.GOV.AU**

CONTRIBUTED SOFTWARE LIBRARY

Save time
and use the
experience of
others to resolve
problems.