Advanced Metering Infrastructure Performance

TABLE OF CONTENT

EXECUTIVE SUMMARY

In many countries, the utilities industry is experiencing massive transformation, not only due to deregulation but also as a result of a shift towards new technologies such as smart metering. Whilst deregulation results in incumbent utilities companies becoming separate companies covering only parts of the total value chain from power generation through distribution to end-user sales and service, smart metering has significant impact on the entire value creation chain both in terms of handling business processes and also on data quantity and quality. Whilst a single monthly measurement value was previously sufficient to bill a customer, in future some countries will require 15 or 30 minute interval data to be stored and billed, even for residential customers.

SAP and Fujitsu have demonstrated in a stress test how well they can cope with the future data volumes to be handled using SAP's established SAP for Utilities solution. Running on a Fujitsu PRIMERGY RX600 S5 server, which hosts the Oracle Database 11g Release 2 database system and the SAP instance in a two-tier configuration, up to 30,000,000 daily time series can be uploaded and 1,500,000 business partners billed in under 3 hours. This figure depends on the scenario, of which three have been tested, each of them suitable for a different type of Utilities Company according to their market role.

RESULTS AT A GLANCE

Three different scenarios for uploading and billing (aggregated) interval metered data in SAP for Utilities have been tested for performance in an environment consisting of a Fujitsu PRIMERGY server and an Oracle Database 11g Release 2 database system. This chapter presents a brief summary of the main results. More details can be found in the sections below.

Introducing advanced meters for the residential customer market challenges utilities companies to upload millions of single values into their energy management and billing system and aggregate them for billing. Two different test scenarios have been performed to demonstrate SAP's ability to handle this large volume in a timely manner using moderate hardware. A third tested scenario uploads only aggregated time-of-use blocks from a Meter Data Unification and Synchronization System (MDUS System) and bills them.

The first scenario uses the classic EDM for C&I Customers solution for this task. Single values are uploaded and checked for consistency; replacement values are provided for any missing values. Billing uses the Real-Time-Pricing (RTP) interface to calculate four time-of-use (TOU) blocks for billing. Throughout the document this is referred to as "TOUB (EDM-RTP) scenario" or "scenario 1".

The second scenario uses the specially developed time-of-use functionality provided with ECC 6.0 EhP5 and is intended to speed up processing in cases where the full EDM functionality is not required (for example validation and replacement values) and where billing does not require the functions provided in the RTP interface. In this scenario, values are uploaded to EDM and accepted without modification, billing uses the new time-of-use interface for aggregation and billing residential customers in SAP for Utilities. In this document it is referred to as the TOUB (EDM-TOU) scenario or "scenario 2".

The third scenario uses another recently developed functionality that does not require interval data to be loaded into the SAP for Utilities System. In this scenario, single interval values remain in the MDUS System, from which the SAP Systems request only aggregated values according to pre-defined time-of-use blocks. Once these TOU blocks are received, billing uses again the new time-of-use interface. "TOUB (MDUS-TOU)" or "scenario 3" are used in this document to refer to this scenario.

The reason for the two first tests is to provide figures for a specific scenario, but also to provide a range in which most customer installations will be found. The first very heavy scenario and the second very lean scenario are not usually appropriate here. The third scenario could be an alternative for those customers who do not want to duplicate large amounts of data.

Measurements have been performed for different numbers of objects to demonstrate the scalability of the environment. The basic assumption of uploading daily values of *N* profiles and billing 1/20 of the corresponding installations has been retained. Measurement points were:

Total number of business partners (= # of profiles)	Number of business partners to be billed daily	Number of profile values loaded
20,000	1,000	1,920,000
40,000	2,000	3,840,000
100,000	5,000	9,600,000
200,000	10,000	19,200,000
400,000	20,000	38,400,000
1,000,000	50,000	96,000,000
2,000,000	100,000	192,000,000
4,000,000	200,000	384,000,000
10,000,000	500,000	960,000,000

The series covers everything from a small municipal utility to all but the very largest utility companies. For the last measurement corresponding to 10.000.000 profiles (or business partners because in this test the relationship between the two entities is 1:1) the following best results were obtained:

Scenario	Total runtime [sec]	Total runtime [hh:min]
TOUB (EDM-RTP)	7836	2:11
TOUB (EDM-RTP) including 3% replacement values	10341	2:53
TOUB (EDM-TOU)	3358	00:56
TOUB (MDUS-TOU) excluding aggregation in MDUS system	1093	00:18

The following graphs show there is a linear scalability of these times, so if the system only contains a quarter of the number of business partners, the time required to process them is also reduced to 25%.

TOUB (EDM-RTP) Scenario

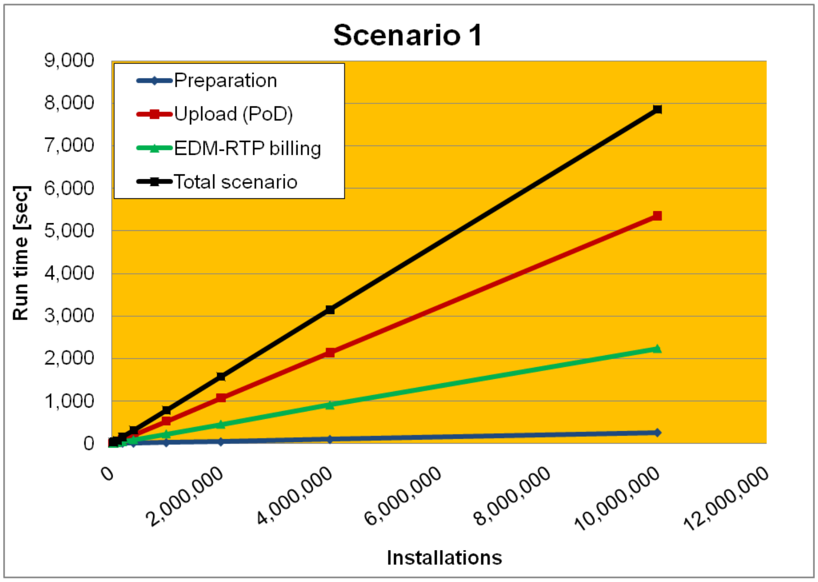


Figure 1: Scalability of TOUB (EDM-RTP) scenario steps in terms of installations processed

TOUB (EDM-TOU) Scenario

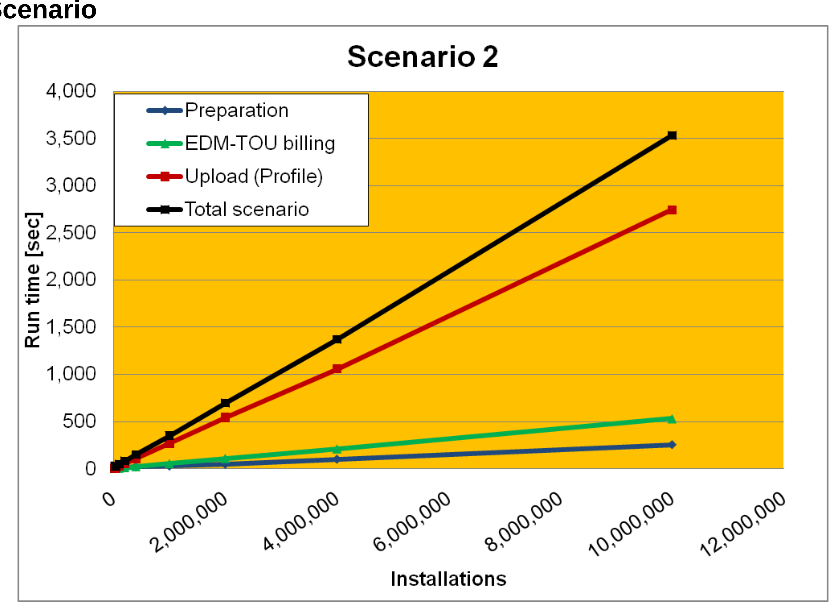


Figure 2: Scalability of TOUB (EDM-TOU) scenario steps in terms of installations processed

TOUB (MDUS-TOU) Scenario

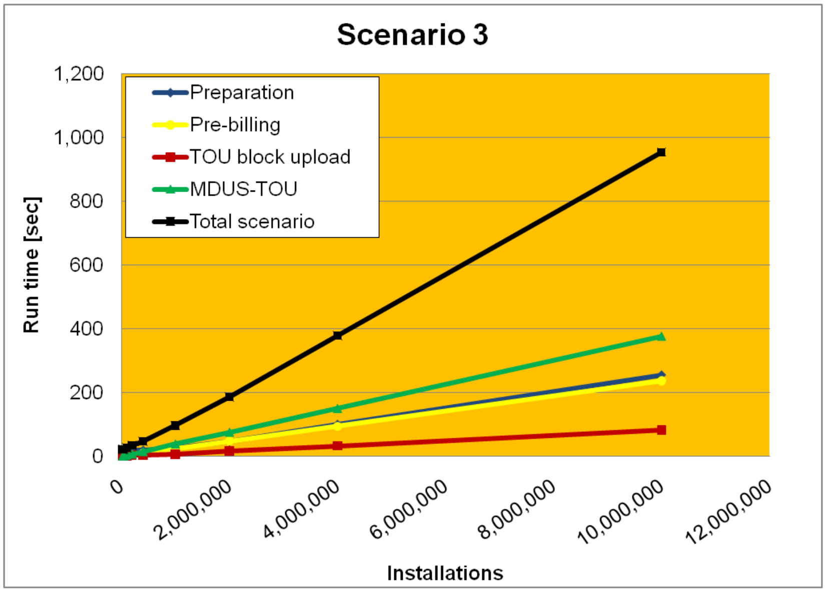


Figure 3: Scalability of TOUB (MDUS-TOU) scenario steps in terms of installations processed

DETAILED TEST DESCRIPTION

The aim of this section is to provide more details about the test for functional and technical experts.

Test Scenario Description

In this test, three scenarios have been chosen and evaluated. The first one is a scenario based on the Real-Time-Pricing (RTP) interface, which was first shipped with IS-U version 4.72 in 2005. This scenario provides all the capabilities of SAP's powerful and flexible utilities solution for billing major industrial customers based on time series, but there is a price to pay in terms of performance. Since RTP is not required in many cases where time-of-use billing is sufficient, a much faster solution based on billing variants has been developed recently; however, it does not provide all functionalities of the RTP interface.

All test scenarios are based on 10,000,000 business partners with monthly billing. The (fictitious) utilities company bills its business partners using a round robin method based on 20 working days per month, which gives the company 2-3 non-billing working days to deal with any problems. This means that 500,000 business partners have to be billed every day. To produce the graphs shown in the chapter above, only subsets of the business partners were handled, but the relation between uploaded values and billed business partners was retained.

Billing is based on time series of 96 individual values per customer and per day, which corresponds to one measurement every 15 minutes.

Single values are aggregated for billing in four time slices according to the following time zones:

Time zone night (00:00 - 05:59)	Time zone off-peak (06:00 - 08:59)	Time zone business hours (09:00 - 11:59)	Time zone peak (12:00 - 13:59)	Time zone business hours (13:00 - 16:59)	Time zone off-peak (17:00 - 21:59)	Time zone night (22:00 - 23:59)

No differentiation is made between weekdays and weekends.

The general assumption is that 3 hours are available each day of the week to upload the daily 10,000,000 time series that each contain 96 values. From Monday through Friday, billing must be performed within these 3 hours. Following billing, additional process steps are necessary before an invoice can be sent to the business partner. These steps are not covered here, since they are part of the SAP Utilities benchmark and are the same as those for a traditional metering scenario. Adding these steps would require approximately three to four additional hours in the same environment depending on the chosen configuration. This means that six to seven hours would be enough for the principal parts of the metering-to-cash value chain. This time-frame corresponds is in line with the requirements of most utilities companies.

TOUB (EDM-RTP) Scenario

This is the most flexible scenario both in terms of value handling and billing options. During the upload, the (externally known) point-of-delivery (PoD), register code and the internal profile ID are mapped to one another, values are validated and replacement values are generated for any missing values. This is done using a simple procedure: The system compares the last measured value before the interval of missing values and the first measured value after and uses the higher of the two to replace the missing ones. Many other procedures are used by utilities companies, so there is no standard available here.

Billing is performed using the RTP interface; the aggregation of values in four time slices is retained.

One additional step is technically required in this scenario; the preparation of billing orders, which also takes place within the timeframe specified.

Originally this scenario was designed to be used for industrial customers and will only be used to bill residential customers in special cases, for example if the functions provided by the TOUB (EDM-TOU) scenario are not sufficient.

TOUB (EDM-TOU) Scenario

The scenario above is not required in many cases. A pure retail company in a deregulated mass market usually already receives validated metering results from the distributor or the metering company. Therefore, all metered data can be uploaded to the SAP System without additional validation or generation of replacement values. This means that many checks in the coding are not required, which reduces the runtime.

The same arguments apply to the billing section. Most retail companies operating in the mass market do not require much more than relatively simple rates based on time of use. In this case, the price of electricity or gas depends on the time it is consumed. Complex rates, such as those based on power consumption that used for industrial customers are not required here. This means the RTP interface can be avoided in favor of the recently developed time-of-use interface in the SAP billing engine.

The preparation of billing orders is also part of this scenario due to its technical requirement.

These two scenarios have been chosen, because they largely represent the runtime limits for the upload and billing process. The upload process consists of four time-consuming blocks that allow the following paths:

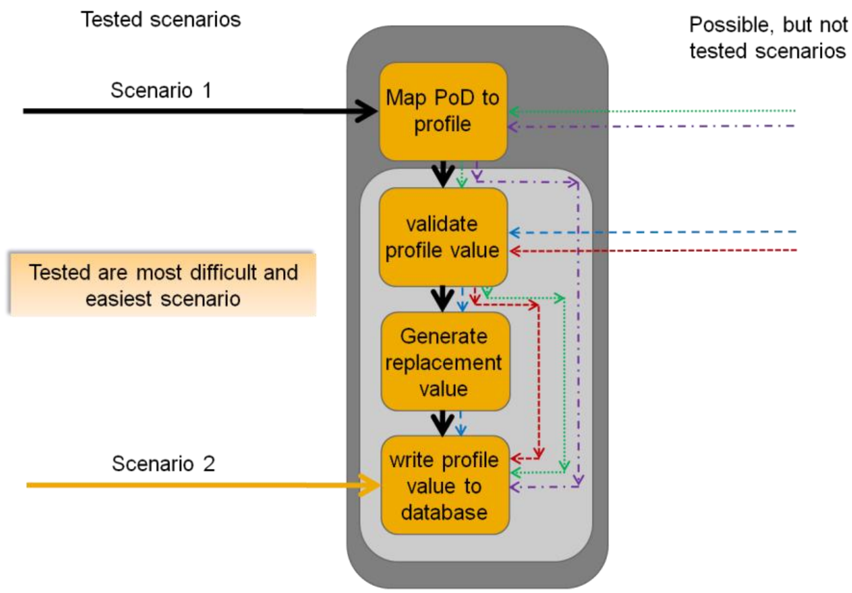


Figure 4: Possible and tested upload scenarios. Each type of arrow distinguished by colour and type denotes a possible way through the upload module.

As shown in figure 4, the first scenario runs through all blocks while the second one ‘simply’ writes the profile values to the database.

The upload scenario is only loosely linked with billing, meaning that every possible upload method above can be combined with billing by the RTP interface or the TOU interface. The test combined powerful but slower billing by the RTP interface with the most time consuming upload type to produce the scenario requiring the most resources. Faster billing by TOU interface using variants was combined with the fast upload method. Customers should therefore expect an overall throughput which is between the two described measurements.

It might not be easy to achieve the throughputs described in the second case because this requires some degree of system tuning. Having less high throughputs in the first scenario is unfortunately much easier since the flexibility of both steps, upload and billing, means that they can be enhanced with own routines that may introduce performance issues. SAP recommends that each customer should conduct their own performance tests prior to go live to ensure good performance.

TOUB (MDUS-TOU) Scenario

Companies that make extensive usage of an MDUS System (Meter Data Unification and Synchronization System) may not want to upload all profile values to their SAP for Utilities System. Another solution has been built for these customers. In this solution, the system only transfers the aggregated time-of-use blocks and not the single profile values to the SAP System, which requires certain changes to the process compared to the two earlier scenarios.

Billing order preparation is still required, but the second step is now a pre-billing run that collects information about the time slices required for billing in the SAP System and then requests them from the MDUS System. Once the time slices have been generated, the MDUS System has to send the aggregated time slices to the SAP System. Billing cannot be performed until these time slices have been received.

Compared to the TOUB (EDM-TOU) scenario, this scenario has some drawbacks, since certain business processes that rely on the status of single profile values require close integration between the SAP System and the MDUS System which cannot be guaranteed.

Upload Activities

In the interests of simplicity, no external tools were used during the test to generate values. A lean program in the system has generated these values and called the BAPI. This increases the system load accordingly, but in terms of this test has the advantage that runtimes for upload activities are slightly overestimated. This has been analyzed and the following overheads were identified:

Scenario	Overhead
TOUB (EDM-RTP)	4,4 %
TOUB (EDM-TOU)	5 %
TOUB (MDUS-TOU)	20 %

PRODUCT DESCRIPTION

SAP

SAP for Utilities solutions help improve operational efficiency, mitigate risk and increase profitability, helping clients gain enterprise-wide visibility for better decision making and improved responsiveness in mission-critical areas. SAP for Utilities employs more than 30 years of industry experience to deliver true end-to-end process support. Leveraging enterprise SOA and the SAP NetWeaver platform, SAP for Utilities drives business and market standardization with more flexibility for regulatory changes.

SAP for Utilities has worked closely with our customers and partners, via the AMI Lighthouse Council and the ACCU AMI working group to introduce AMI integration and innovative business processes to help improve sales and customer service performance. SAP® software can support your AMI initiative so that you can make customer service more cost-effective, automate the exchange of data within your business networks, and ultimately optimize your revenues and market demand. Using open standards, your organization will be able to collaborate with its customers in support of AMI-related processes and deliver the real-time information that today's market and regulatory mandates demand. SAP will continue as a thought leader in AMI and Smart Grid through its active participation in various AMI-related boards such as European Smart Grid Association, ESMIG (EMEA Smart Meter Industry Group), GridWise and Smart Grid Australia.

More than 1,600 utilities, in over 70 over countries, in power generation, transmission, distribution, retail, gas, water, waste and recycling run SAP Utilities software focusing on improving energy efficiencies and sustainability.

Visit SAP online at: http://www.sap.com/industries/utilities/index.epx

Oracle

Oracle Database 11g Release 2 Enterprise Edition delivers industry leading performance, scalability, security and reliability on a choice of clustered or single-servers running Windows, Linux, and UNIX. It provides comprehensive features to easily manage demanding transaction processing, business intelligence, and content management applications. Oracle Database 11g Release 2 Enterprise Edition comes with a wide range of options to extend the database to help grow your business and meet performance, security and availability service level expectations.

Advanced Compression is an option introduced in Oracle Database 11g and includes a comprehensive set of compression capabilities to help customers maximize resource utilization and reduce costs. It allows IT administrators to significantly reduce their overall database storage footprint by enabling compression for all types of data – be it relational (OLTP table compression, as used in this project), unstructured (SecureFiles deduplication and compression), backups (RMAN and Data Pump) and for Data Guard Redo Log network transport.

Although storage cost savings are often seen as the most tangible benefit of compression, innovative technologies included in the Advanced Compression Option are designed to reduce resource requirements and technology costs for all components of an IT infrastructure, including memory and network bandwidth.

Oracle's OLTP Table Compression uses a unique compression algorithm specifically designed to work with OLTP applications. The compression ratio achieved in a given environment depends on the nature of the data being compressed; specifically the cardinality of the data. In general, customers can expect to reduce their storage space consumption by a factor of 2 to 4x by using the OLTP Table Compression feature. With SecureFiles compression, typical files such as documents or XML files, experience a reduction of 2 to 3x in size. In addition to compressing data stored inside the database, Oracle Advanced Compression also includes the capability to compress backed up data. Recovery Manager (RMAN) and Data Pump are the two most commonly used tools to backup the data stored inside an Oracle Database. Data Guard Redo Transport Services are used to transfer redo data to the standby site(s). With Advanced Compression, redo data may be transmitted in a compressed format to reduce network bandwidth consumption and in some cases reduce transmission time of redo data.

Fujitsu

PRIMERGY RX600 S5

The scalable PRIMERGY RX600 S5 Quad socket server supports the latest Intel Xeon 7500 processor series. It is in every respect a reliable server for critical company SAP scenarios. The ideal interaction of integrated redundancy functions with server management components results in high-level availability and constantly efficient IT production as a character feature of this server platform.

The processors with up to 8 cores enable a unique performance boost paired with other features to provide optimal dividends: a high extendable main memory capacity with up to 64 DIMMs and a very high number of PCI Express channels enable a balanced system composition and scalability so as to meet increasing requirements and excellent best-in-class efficiency.

Balanced scalability means that by adding up processors, main memory or I/O cards to the PRIMERGY RX600 S5 system, the compound resulting performance gain will increase in an almost linear way. This in turn requires that the basic system architecture must be designed to support linear efficient scalability.

The RX600 S5 has all infrastructure components built into the 4U chassis. Thus it scales up inside its given infrastructure when adding up to the four Intel Xeon processors , the directly linked eight memory boards and the PCIe extension cards, and it does so without provoking changes to the rack infrastructure setup.

The "glue-less" design assures that no additional hardware is necessary to run all 4 CPUs, all memory boards and both I/O hubs seamlessly in concert. The new Intel Quickpath QPI link architecture, that replaces the old front-side bus system architecture, is directly implemented on the RX600 S5 server motherboard. This avoids any custom chipsets or external node controllers and cabling schemes to make up for a 4 socket server. Not using any "external glue" eliminates additional points of failure that can result from glue components. As a result of its "glue-less" architecture, the shortest routes between processors, memory and I/O are achieved, enabling for lowest possible latency times when passing data between memory and processor sections. These design principles are paying back into excellent performance results with the Fujitsu PRIMERGY RX600 S5, providing customers in the upper-mid-market as well as large enterprises a powerful and reliable platform for their critical SAP environments.

Test Environment

The system used consists of one Fujitsu PRIMERGY RX600 S5 server with 4 processors, 32 cores and 64 threads. Oracle Database 11g Release 2 Enterprise Edition with Partitioning and Advanced Compression was chosen because it is widely used by large enterprises.

The system environment uses a two-tier configuration where the database and SAP instance share the available resources. This facilitates automatic load balancing between them and minimizes idle resources such as CPU which is usually either required by the work process of the application server or the database process. This environment is also an ideal starting point for higher scalability when required. Using Oracle Real Application Clusters technology, it would be possible to place similar nodes in the system without having to consider how many resources the database would need in comparison to the SAP System.

Since part of the test involved uploading 960,000,000 values in a short timeframe, high I/O rates could be expected from the outset. Therefore, the database files were distributed between 36 disks in an EMC storage system to avoid hotspots on a single disk that could severely impact performance as showed by experience from some live installations.

Software layer

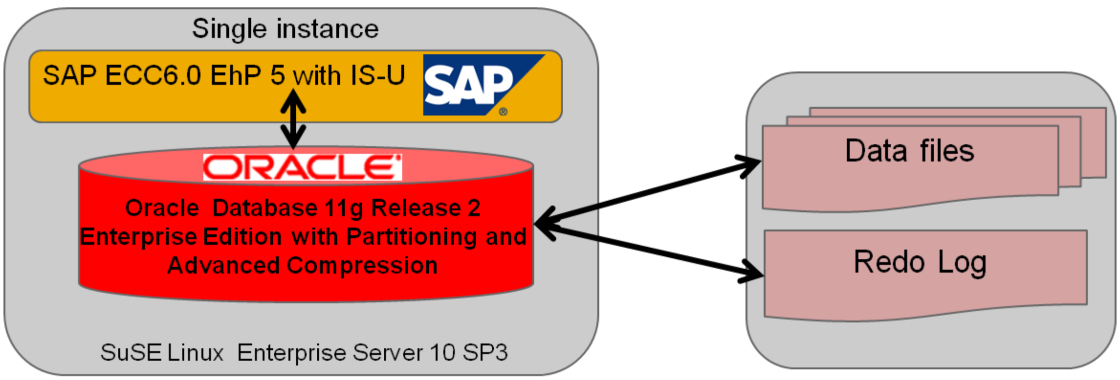
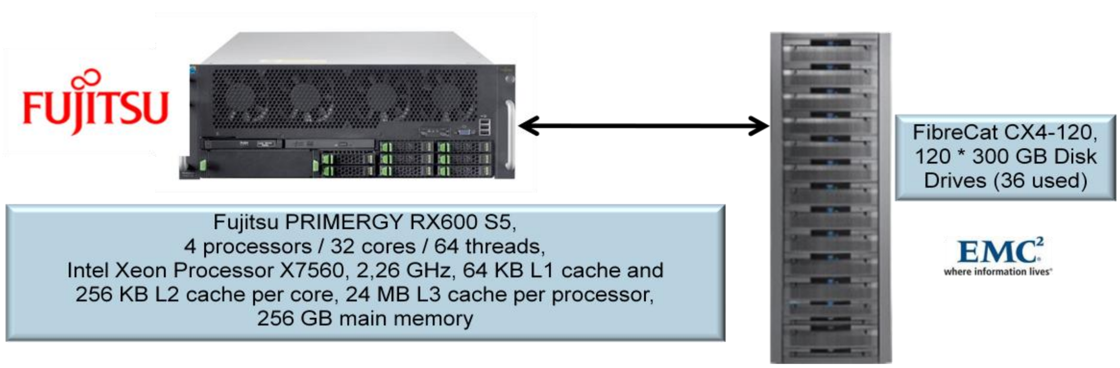


Figure 5: Software set-up of test environment

Hardware layer

Figure 6: Technical set-up of test environment



RESULTS IN DETAIL

This chapter presents the results of the tests. It is most important to prove the scalability of the solution in terms of parallelization since this is the key to higher throughput in cases where it is needed. These results are initially presented on a scenario basis.

A special section concerning the different upload activities follows that covers the features that are unique to these activities.

The remainder of the chapter discusses some topics that are common to all steps.

The number of jobs used to obtain the best result is significant here. Scaling figures are shown for up to 32 concurrent jobs, which correspond to the number of processor cores. However, a higher number is shown in the tables below where figures for the jobs are given that achieved the highest throughput. This is because Hyper-Threading, which is explained later in the “General Findings” section under “Processor”, was used. The difference between the throughput with 32 concurrent jobs and the best result is shown as “Gain by Hyper-Threading”

TOUB (EDM-RTP) Scenario

This scenario consists of

- Billing order preparation
- Upload of profile values identified by point-of-delivery
- Time-of-use billing by RTP interface

The curves obtained by increasing the number of concurrent batch jobs are shown below. These all show excellent scaling behavior:

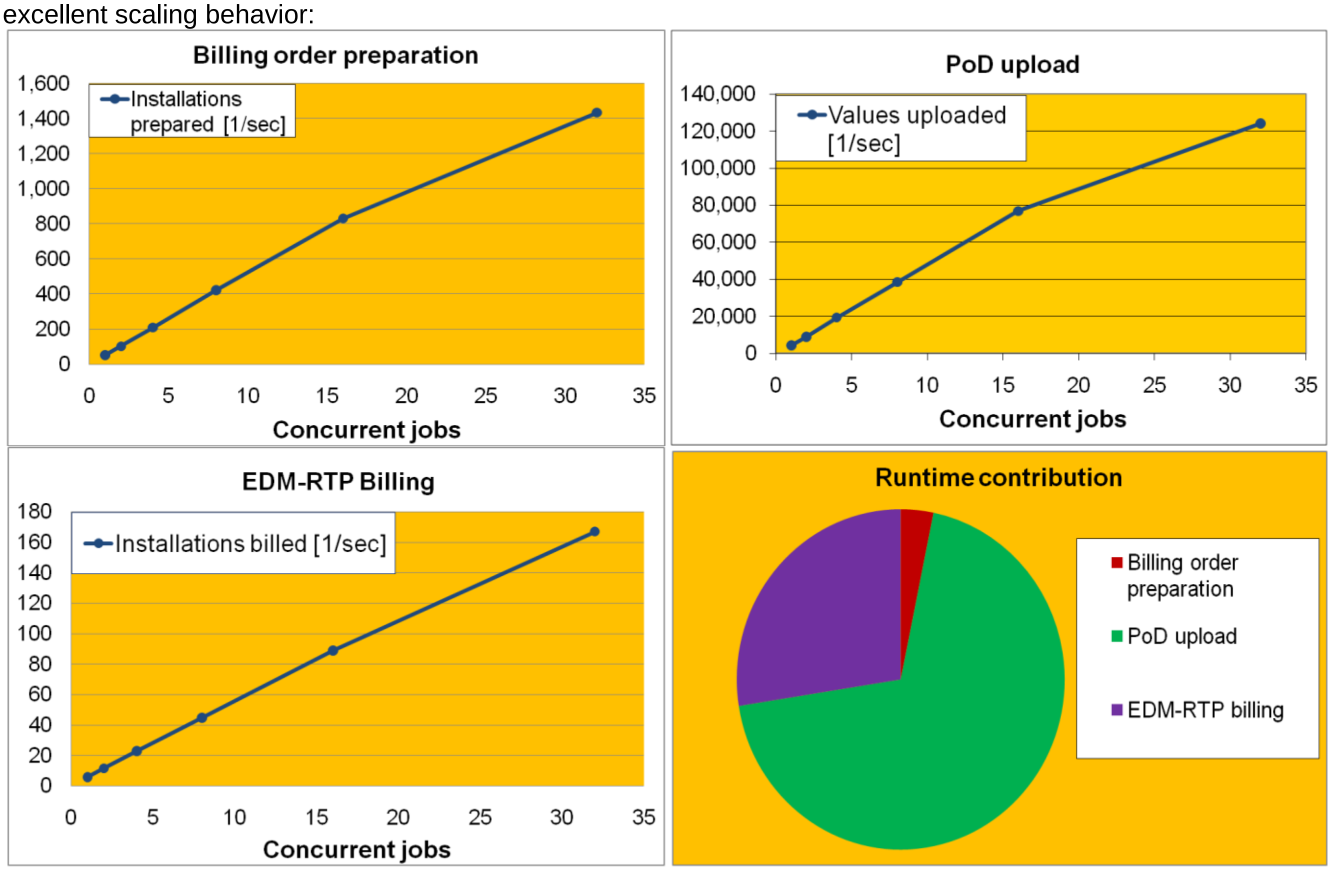


Figure 7: Throughput of steps in TOUB (EDM-RTP) scenario in terms of concurrent processes

The last chart shows the relative contribution of each step to the total runtime. Significantly, most time is spent uploading profile values. In these graphs, the upload is measured without generation of replacement values. The way in which they would influence the upload is discussed in another paragraph below. Some of these paragraphs cover the dependency on entities that do not apply to billing order preparation and billing, but some of the results are already listed in the following table:

Measurements performed scenario 1:

Test series scope	Billing order creation	PoD upload	TOUB (EDM-RTP) billing
Scalability by concurrent jobs (see figure 7)	✓	✓	✓
Scalability by number of objects (see figure 1)	✓	✓	✓
Dependency on block size	n/a	Best throughput achieved with > 500 values per call	n/a
Dependency on days uploaded	n/a	Improvement of factor two versus single upload of multiple days	n/a
Dependency on replacement values	n/a	Throughput depends on number of replacement values	n/a

Best Results Achieved:

	Billing order creation	PoD upload	TOUB (EDM-RTP) billing
Quantity measured	500,000 installations	960,000,000 profile values	500,000 installations
Run time [sec]	253	5276	2307 seconds
Concurrent jobs	75	75	64
Gain by Hyper-Threading	31.1%	31.8%	24.8%
Memory used [MB]	2110	1502	3591
Physical writes per second	154	2431	599
Physical reads per second	371	1298	619
Redo writes per second	824	737	213
Redo writes [MB per second]	5.7	20.5	4.8
Buffer hit ratio	99.87%	99.78%	98.71%

Replacement Values:

The influence of the generation of replacement values on the runtime of the upload can be seen in the graph below. A high number of replacement values leads to a substantial increase, but usually such high percentages are not expected. Missing values are usually retrieved some hours or days later. Nevertheless, it is demonstrated that a small amount of replacement values does not affect the throughput sigmificantly.

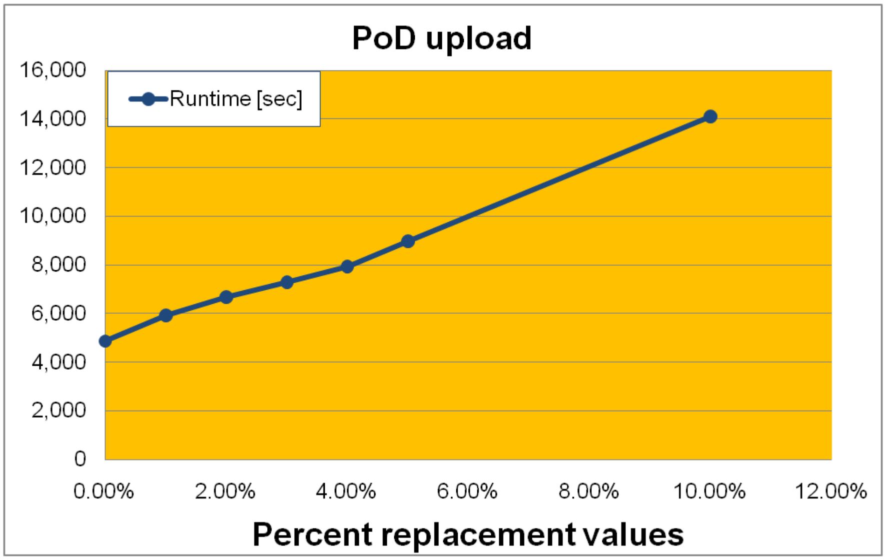


Figure 8: Run time dependency on number of replacement values

If for example 3% of replacement values require generation, the runtime of the full scenario would increase by 42 minutes, but still remain below 3 hours.

TOUB (EDM-TOU) Scenario

This scenario consists of

- Billing order preparation
- Upload of profile values identified by profile
- Time-of-use billing using the new time-of-use interface with profile values from EDM

In comparison with the first scenario described above, some differences have a positive impact on the throughput:

1. It is assumed the profile number is known to the system from which the values originate, this means that mapping between the point-of-delivery identifier and the profile number is not required.
2. It is assumed the profile values are copies of values from the original system that have already been validated and checked so this action no longer has to be executed in the SAP System.
3. Billing uses the time-of-use interface instead of the RTP interface. This restricts the billing options but still permits time-of-use billing. The restrictions should only affect commercial customers with complicated rates in almost all cases.

Billing order preparation is dependent on the scenario, but the differences are less than 1% and can therefore be ignored.

As shown below, excellent scaling behavior is given for all steps of the scenario:

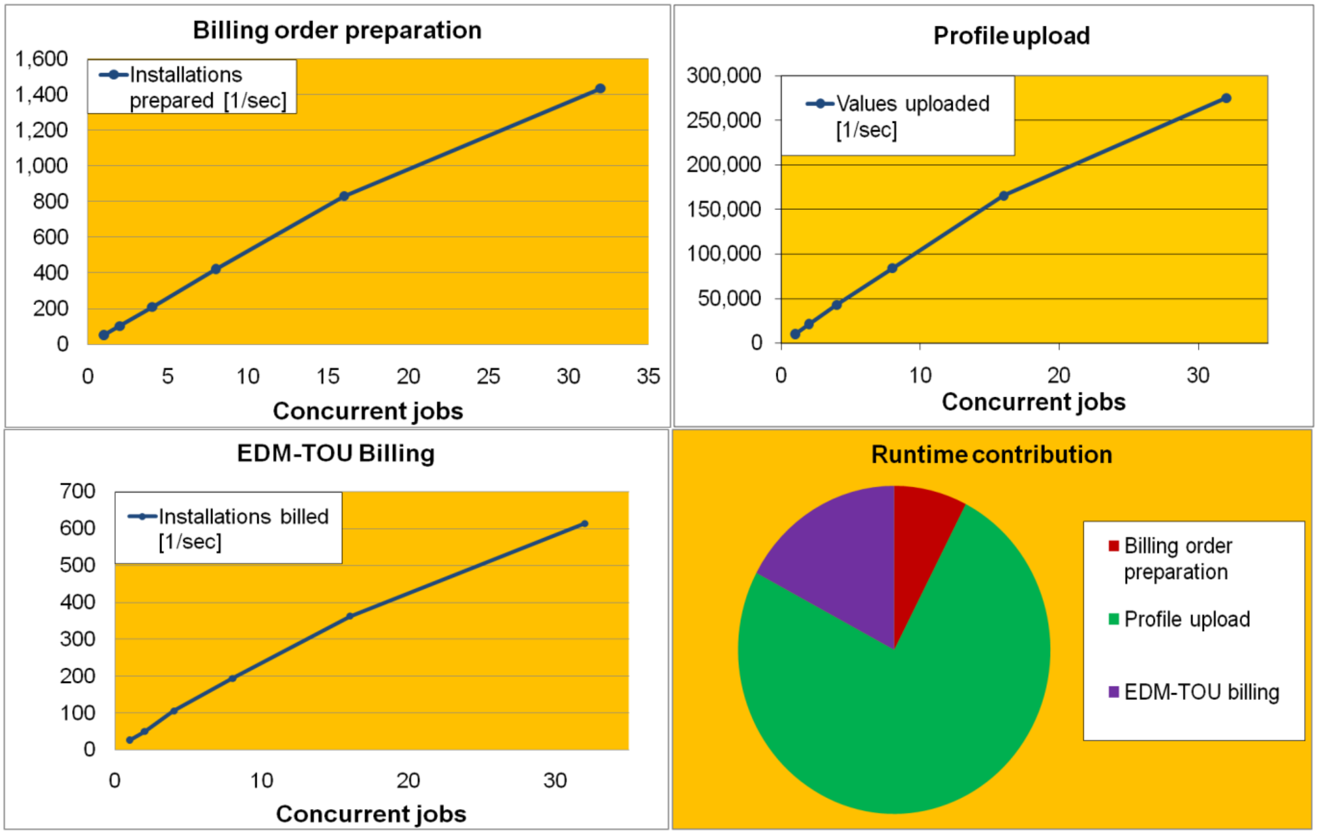


Figure 9: Throughput of steps in TOUB (EDM-TOU) scenario in terms of concurrent processes

Again it can be seen that uploading profile values requires most of the scenario runtime. As already discussed in the section about scenario 1, the upload has additional dimensions that are discussed in a later paragraph.

Measurements performed scenario 2:

Test series scope	Billing order preparation	Profile upload	TOUB (EDM-TOU) billing
Scalability with concurrent jobs (see figure 9)	✓	✓	✓
Scalability by number of objects (see figure 2)	✓	✓	✓
Dependency on block size (see figure 12)	n/a	Best throughput achieved with > 500 values per call	n/a
Dependency on days uploaded (see graph)	n/a	Improvement of factor two versus single upload of multiple days	n/a

Best Results Achieved:

	Billing order preparation	Profile upload	TOUB (EDM-TOU) billing
Quantity measured	500,000 installations	960,000,000 profile values	500,000 installations
Run time [sec]	253	2577	525
Concurrent jobs	75	64	85
Gain by Hyper-Threading	31.1%	26.2%	37.4%
Memory used [MB]	2110	1289	2392
Physical writes per second	154	2149	1390
Physical reads per second	371	457	2521
Redo writes per second	824	937	265
Redo writes [MB per second]	5.7	17.7	16.3
Buffer hit ratio	99.87%	99.92%	98.50%

Scenario 2 is roughly 2.5 times faster than scenario 1 overall.

TOUB (MDUS-TOU) Scenario

This scenario consists of

- Billing order preparation
- Pre-billing run to request TOU blocks from the MDUS System
- Retrieval of TOU blocks from the MDUS System
- Time-of-use billing using pre-calculated time-of-use blocks

The throughput is the highest of the three scenarios compared in this paper. However, it must be emphasized that this test does not include the value aggregation in the MDUS System. Therefore, the end-to-end runtime must be increased by the time the MDUS System needs to perform this task and the time required for the communication between the two systems. Hence the throughput provided here is definitely too high.

Different MDUS Systems are available on the market and they can be expected to have different response times leading to different throughputs when it comes to the end-to-end scenario.

Nevertheless, this scenario is different to the others even within SAP and therefore some performance gains can be expected:

1. Instead of loading one time series per profile each day, it is only necessary to load four time-of-use blocks when billing is due. This reduces the number of values to be loaded from 960 million to 2 million and the number of objects handled from 10 million profiles down to 500,000
2. Billing does not have to generate the TOU blocks, they simply have to be read from the database

However, an additional step must be executed that demands some runtime. Since the MDUS System does not recognize the billing rules, the definitions of the required TOU blocks must be transferred from SAP to the MDUS System on an installation basis. Once this is done, it can perform the aggregation and then send the consumptions for each TOU block back to the SAP System.

As shown in figure 10, excellent scalability is provided for all steps in this scenario.

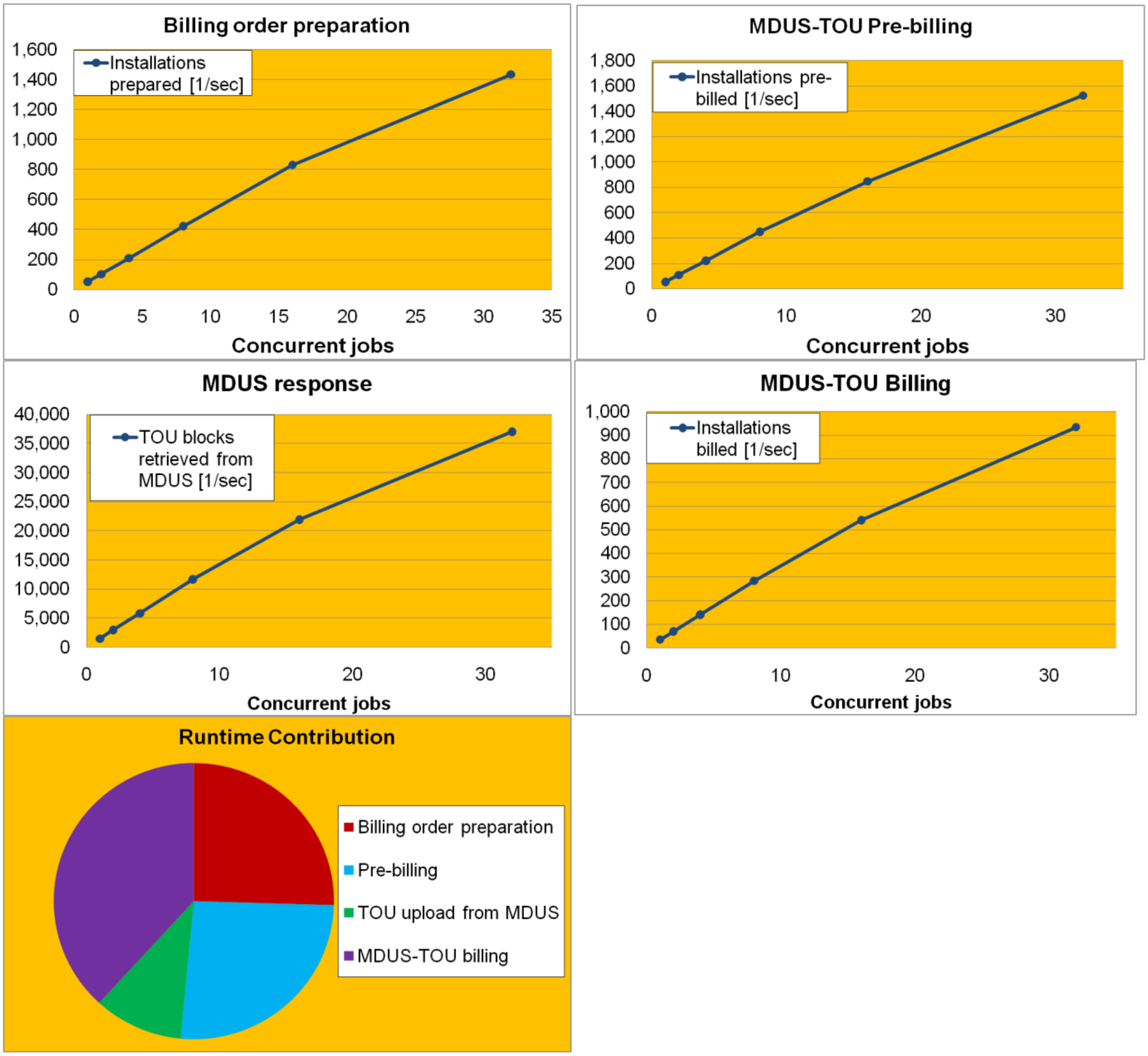


Figure 10: Throughput of steps in TOUB (MDUS-TOU) scenario in terms of concurrent processes

Measurements performed:

Test series scope	Billing order preparation	Pre-billing	Time-of-use block from MDUS system	TOUB (MDUS-TOU) billing
Scalability with concurrent jobs (see figure 10)	✓	✓	✓	✓
Scalability by number of objects (see figure 3)	✓	✓	✓	✓
Dependency on block size	n/a	n/a	100 installations per call gives best throughput	n/a

Best Results Achieved (500,000 installations):

	Billing order preparation	Pre-billing	Time-of-use block from MDUS system	TOUB (MDUS-TOU) billing
Quantity measured	500,000 installations	500,000 installations	500,000 installations	500,000 installations
Run time [sec]	253	266	91	387
Concurrent jobs	75	64	75	75
Gain by Hyper-Threading	31.1%	22.1%	33.8%	29,9%
Memory used [MB]	2110	1801	8403	2110
Physical writes per second	154	5826	3900	1125
Physical reads per second	371	265	2263	1102
Redo writes per second	824	48	31	711
Redo writes [MB per second]	5.7	37.0	52.0	12.9
Buffer hit ratio	99.87%	99.87%	99.51%	99.55%

It is challenging to compare scenario 3 with the others when no data are available for the time required by the MDUS System. Working on the reasonable assumption that the aggregation in the MDUS System should not take longer than the TOUB (EDM-TOU) billing, in which aggregation is also required, scenario 3 is roughly twice as fast as scenario 2 and five times faster than scenario 1.

In this test, a report that runs in the SAP System was used to simulate the response from the MDUS System. This report called the SAP inbound service so that the final runtime measured is likely to be over-estimated.

Upload Activities

All upload activities share certain behaviors that are not seen in the other activities. The latter ones are always handled on a strict single object basis. This means for example that a commit work is executed after each installation is billed and there is no bulk processing.

This is different for the upload activities and is therefore handled separately in this section.

The most important parameter is the block size. This is the number of values of a given profile in a single BAPI call. The dependency has been measured starting from a transmission of only one single value per BAPI call up to all values for 100 profiles per BAPI call.

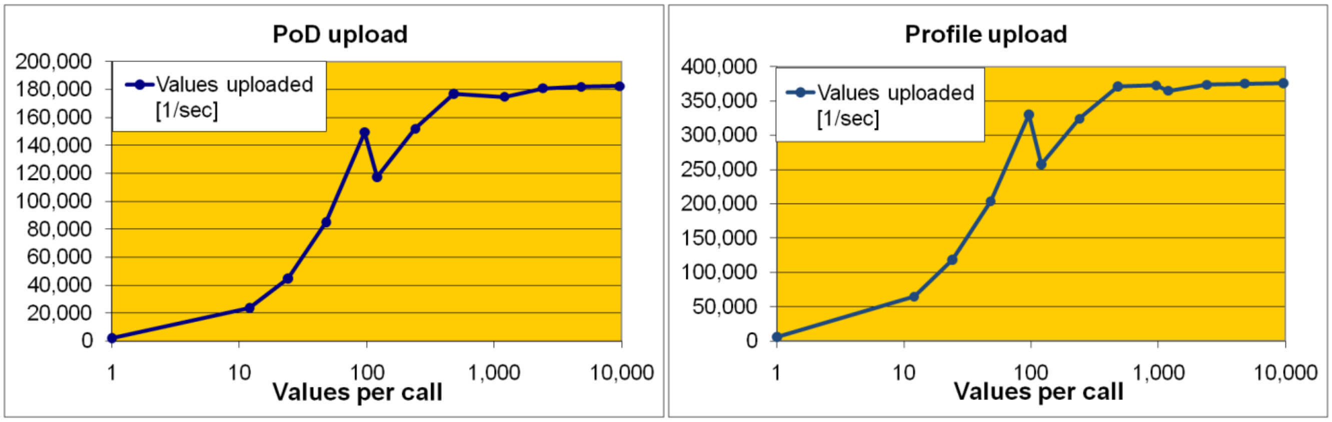


Figure 11: Run time dependency on number of values per BAPI call

Both graphs in figure 11 clearly show a dramatic decrease in throughput towards low values. This is because the checks of the master data have to be performed for every profile per BAPI call. If each call only handles one value, every profile has to be checked 96 times to see if it exists and is consistent with the send value(s).

It is also clear that sending more than 1000 values per BAPI call does not increase the throughput, so it is recommended to submit between 1000 and 10000 values per BAPI call, with the restriction that all values of a given profile should be sent with the same call.

The peaks in the curves have logical reasons; it appears for a block size of exactly 96 values per BAPI call. In this situation, a call is made for exactly one profile. With 120 values per call, many profiles have values in two subsequent calls of the BAPI, thus increasing the number of required checks and hence the reduction in terms of throughput.

In both cases, the results were obtained with 64 concurrent jobs.

The MDUS System response shows similar results, but as is always the case, no peak can be identified when uploading all TOU blocks for an installation. Again, maximum throughput is reached at around 500 installations per service call and no further improvement was seen when using higher numbers.

Sometimes it is not possible to perform a daily upload of profile values, for example if the system is down for a maintenance weekend. A naïve expectation would be *N* times the runtime of one day's value upload, where *N* is the number of days to be uploaded. As shown in figure 12, when done correctly, this is not the case:

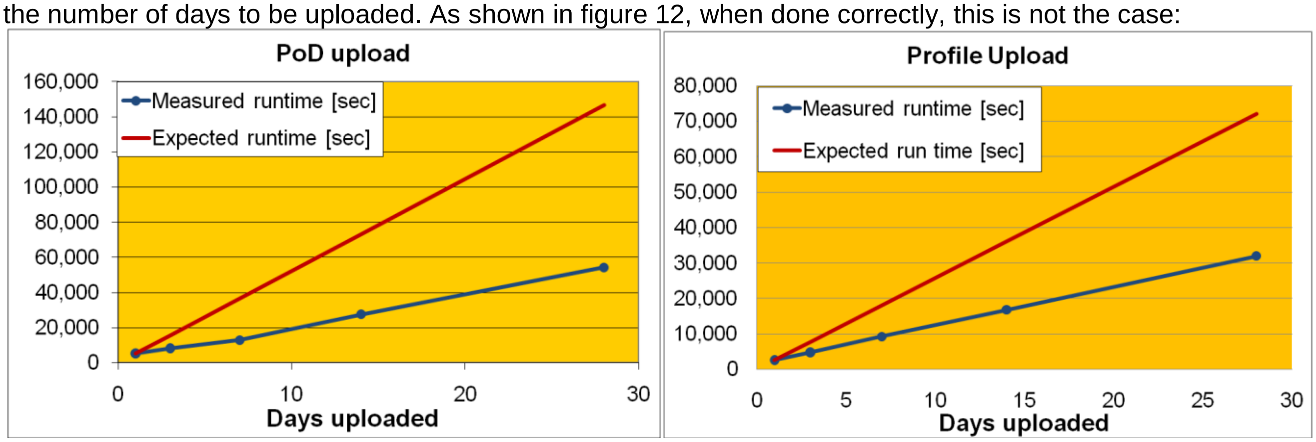


Figure 12: Dependency of total run time from number of days loaded

The achievable throughput is increased by a factor of two to three against the naïve expectation. To achieve this result, it is mandatory to order the values to be transferred first by the profile, then by the day. In this example, the order would be profile A (day 1, day 2,day 3), profile B (day 1,day 2,day 3) and so on. If they are ordered the other way round, the total runtime would be *N* times the time required for one day (Order would be day 1 (profile A, profile B), day 2 (profile A, profile B), day 3 (profile A, profile B)).

All measurements in this section have been performed with uploads of 960,000,000 profile values, except the one where the influence of the block size has been measured where only 96,000,000 values were uploaded (corresponds to one million profiles). In another test series shown in the first chapter of this presentation, it has been confirmed that the total number of profiles uploaded has a strictly linear influence on the runtime. This means that with the same number of concurrent jobs, *N* times more profiles require *N* times more time.

The dependency between the time needed to upload TOU blocks from the MDUS System on the number of blocks loaded per installation was not investigated. There are three reasons for this:

1. The overall scenario assumes that profile values must be uploaded every day, whilst billing is only performed on 20 working days of a month. In scenario 3 however, the upload is only performed during billing and this automatically leaves the weekends free for maintenance tasks.
2. The actual number of blocks is generally low and defined by the rate and is not therefore a technical parameter permittiung configuration.
3. In scenario 1 and scenario 2, the upload is the single most time consuming step, but which is most time consuming while in scenario 3 it is the one with the fastest runtime. Therefore, this should not pose a problem.

General Findings

Processor

Fujitsu's RX600 S5 comes with Intel's Xeon processor series 7500, the Nehalem-EX-processor architecture, and was equipped with 4 processors X7560, 32 cores and 64 threads.

Intel Xeon 7500 architecture implements the INTEL QPI Quick Path Interconnect technology principles to provide sufficient growth, each for Memory capacity, CPU/Memory Transfer Bandwidth, I/O Bandwidth and CPU performance.

Additional performance improvements result with the new Xeon 7500 processors by supporting Hyper-Threading. Using HT, a processor core is enabled to process two different software threads simultaneously, a hyper-threading processor appears as two "logical" processors to the operating system, allowing the OS to schedule two threads or processes simultaneously. Thus throughput can be increased compared to implementations without Hyper-Threading.

This load test demonstrates scaling up to 32 jobs which corresponds to the number of cores and the additional Hyper-Threading improvement is shown by using over 32 jobs.

Memory

Memory requirements were not high in any of the individual steps in the different scenarios; 4 MB per concurrent process was usually enough.

Exact memory requirements are difficult to determine because they depend on the way installations (the technical SAP object that is billed) are distributed between the jobs. This is a two step process: During the first step, installations are distributed between intervals. In the second step, intervals are processed by batch jobs. Generally, there is no fixed relation between the number of intervals and the number of batch jobs, but to provide a good distribution, the number of intervals should be a multiple of the number of batch jobs. The memory itself depends on the number objects per interval and the number of batch jobs.

Figure 13 from TOUB (EDM-TOU) billing describes this behavior effectively. Up to 100,000 installations have been distributed into 64 different intervals and each of the 64 batch jobs processed exactly one interval. This meant that the memory requirements increased with the number of installations up to a maximum value of 1563 installations per interval.

In cases of over 100,000 installations, the interval size was limited to 1000 installations per interval; hence the memory requirements are lower and remain constant.

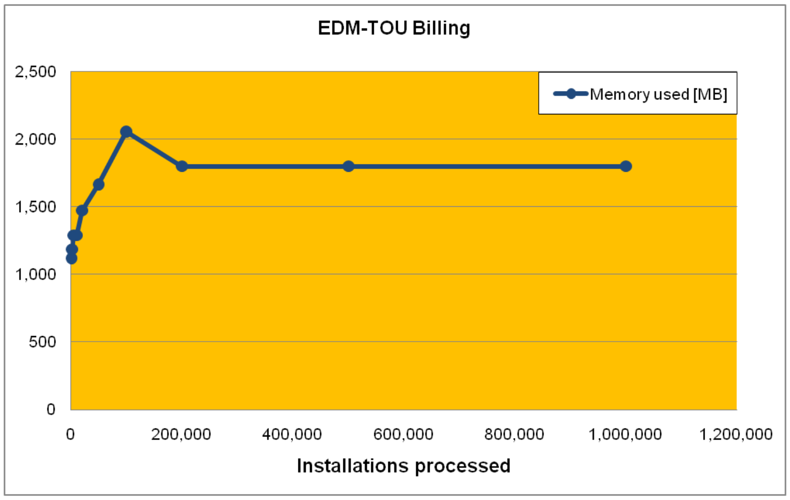


Figure 13 Memory consumption

CPU consumption

The actual consumption of CPU time varied between the different steps that have been measured as can be expected. Some activities require more I/O, others more CPU. Important is to check good scalability and optimal usage of available resources. This is fulfilled in this test, as the example of the RTP billing in figure 14 shows:

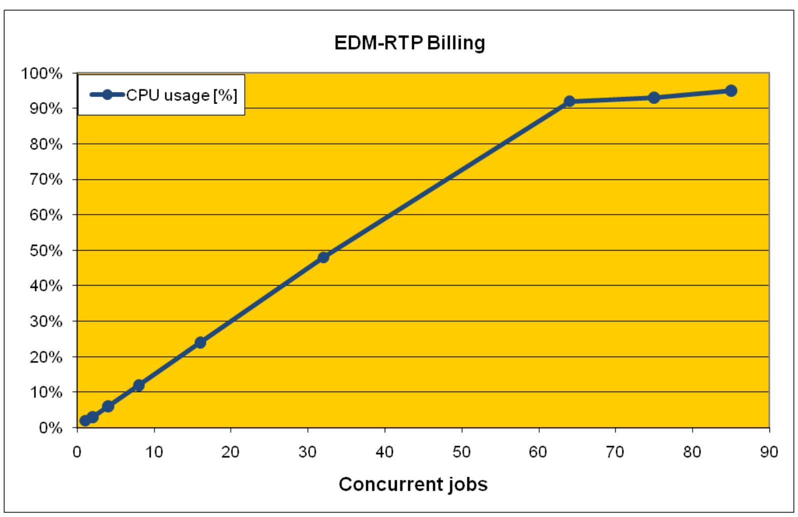


Figure 14:CPU usage dependency on number of concurrent jobs

As the number of jobs increase, the CPU consumption from VMSTAT increases linearly up to above 90%.

BENEFITS OF COMPRESSION

Due to the high volumes of data to be handled and stored on the database, it is an ideal candidate for database compression. SAP Utilities billing can handle large values with high precision (metering data is internally stored with 31 digits) which is not required in the mass market where consumptions above 10 kWh within 15 minutes are very unlikely. On the other hand, measurement accuracy below the range of 1/10000 of a kWh is not required, because at current price levels it would change bills at a sub-cent level. This means that 7 to 8 digits are usually enough to store the data, which provides a good basis for compression.

The SAP Utilities System can potentially benefit in three ways from a good database compression algorithm:

1. Lower storage requirements
2. Decreased I/O volume
3. Less I/O requests

The lower storage requirements are obvious and do not require discussion. Since data is usually stored in data blocks of certain sizes (8 kB in this case), more table rows can be stored in a single compressed data block. Transfer of data between memory and storage system is always done by sending or receiving full data blocks, therefore the I/O volume can also decrease significantly. If the database system keeps data blocks within the memory compressed and performs the decompression only before sending data to the client, more table rows can reside in the same fixed amount of memory. This effect in turn can reduce the total number of I/Os between the memory and the storage.

To demonstrate the usefulness of a powerful compression algorithm such as the one provided by the Oracle Database 11g with Advanced Compression, the tests above have been repeated without compression. The two diagrams below show the differences in terms of run time and database size for an activity.

Figure 15 shows the differences in runtime for the different jobs with and without compression switched on. The tests with the highest throughput have been chosen as examples.

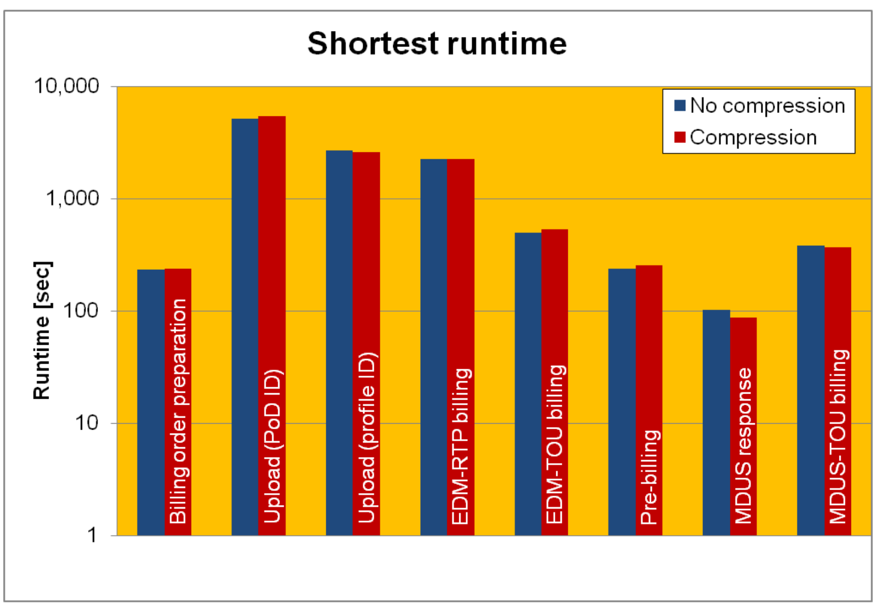


Figure 15: Effect of compression on the run-time

Figure 16 shows the effects of compression on the size of tablespace PSAPSR3 that contains all application tables. Without compression, the size of the database was 1.47 TB. With OLTP compression the size of the database was 732 GB, meaning that in this environment a compression factor of over 2 was achieved.

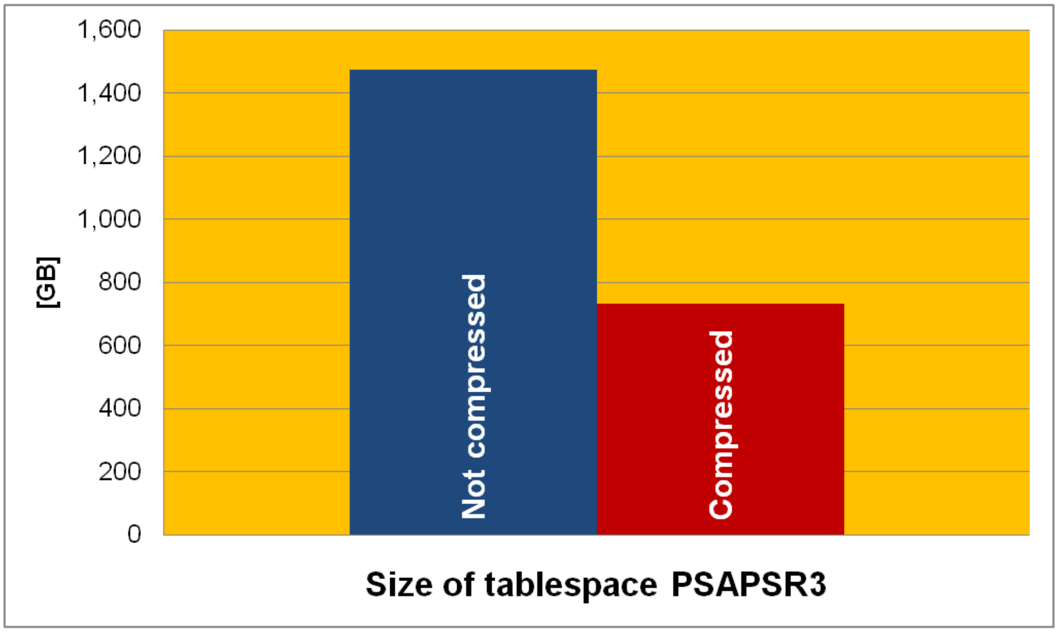


Figure 16: Sizes of tablespace PSAPSR3 before and after compression

CONCLUSIONS

This test has demonstrated the ability of the successful SAP for Utilities solution to handle a very high volume of profile data, which is a common feature of the residential customer market, though so far only a small number of utilities companies in the market currently need to process such volumes.

It was possible to fulfill the requirements using a mid-range server by Fujitsu. The new Advanced Compression Option provided by Oracle Database 11g greatly reduces the storage amount by a minimum factor of 2 without penalizing throughput.

Scalability has been proven in all relevant dimensions, meaning that further scalability can be expected in cases where more time series have to be uploaded or more bills have to be produced within the same timeframe or the same amount in a shorter timeframe.

The results of the test can be used to derive some best practices, such as those for uploading profile values.

DISCLAIMER