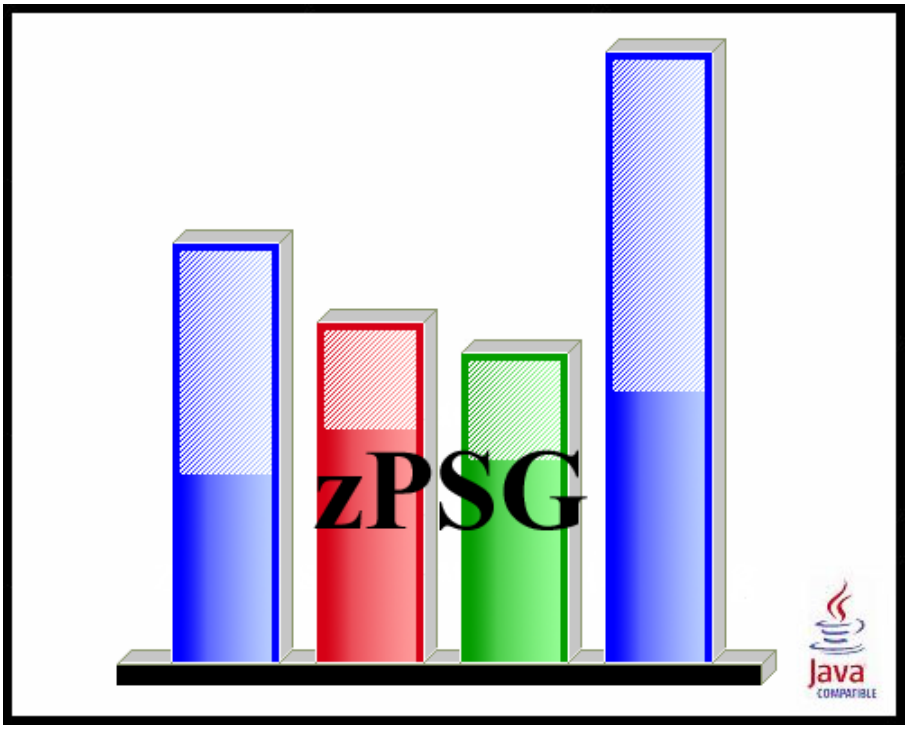
Processor Selection Guide
for
IBM System z

zPSG
User's Guide
for
Business Process Manager



Version 5.9b
zPSG BPM UG V59b 2014a01.doc
June 27, 2014

Business Process Manager
z/OS and Linux

This tool provides estimates of System z processor capacity for J2EE applications running under Business Process Manager (BPM) V8.0 and using JCA (J2C) connectors. J2EE and JCA/J2C are standards for Java application design and deployment.

See the BPM Glossary of Terms for a definition of terms used in describing BPM applications.

BPM V8.0 is a next-generation business process server that supports all styles of integration based on Service-Oriented Architecture (SOA) and open standards. SOA provides you with the ability to develop and modify integration applications dynamically. It also lets you integrate existing applications with newer applications, so that they work together transparently.

BPM V8.0 is a comprehensive Service-Oriented Architecture integration platform, which is based on WebSphere Application Server (WAS) V8.0. Because it is based on J2EE 1.6 infrastructure and platform services provided by WAS V8.0, BPM includes capabilities such as business process automation. You can automate business processes that span people, workflows, applications, systems, platforms, and architectures making it the ideal platform for business applications.

Using the WebSphere Integration Developer tool set (based on the Eclipse 3.0 platform), business integration solutions can be created using simplified integration mechanisms, such as the Service Component Architecture (SCA) programming model and the Service Data Objects (SDO) data model.

SCA is a service-oriented component model for defining and invoking business services that publish or operate on business data. SCA is aimed at providing a simplified programming model for writing applications that run in a J2EE runtime environment, and is based upon concepts and techniques that are refinements of existing J2EE technology. One of the important aspects of SCA is to enable the separation between application business logic and the the implementation details. In order to accomplish this, SCA provides a single abstraction for service types that might already be expressed as session beans, Web Services, Java classes, or Business Process Execution Language (BPEL).

SDO business objects can be defined, transformed, routed and mapped using SCA components. The goal of SDO is to provide a programming model that will unify data representation across heterogeneous data sources and simplify application development for developers. SDO provides this by offering a common API that can be used regardless of the backend data store being accessed. This common way of representing data also makes SDO an ideal choice for data abstraction in a Service-oriented Architecture.

BPM also uses WebSphere Adapters which supply connectivity to back-end Enterprise Information Systems (EIS). With BPM, business integration applications may define

business logic and processes based on Web Services, BPEL, human tasks, and business rules.

For the runtime monitoring of the business integration solutions, BPM provides Common Event Infrastructure (CEI), which centralizes the monitoring of the various events that occur in these applications.

BPM enables deployment of standards-based integration applications in an SOA structure. SOA is an application framework that takes everyday business applications and breaks them down into individual business functions and processes called services. SOA is a conceptual description of the structure of a software system in terms of its components and the services they provide, without regard for the underlying implementation of these components, services and connections between components. Loosely coupled integration applications that are based on SOA provide flexibility and agility.

Enterprise Service Bus (ESB) simplifies implementation of an SOA by reducing the number, size, and complexity of interfaces between services. It resides between a requestor and a service, and provides the following capabilities:

- Transformation of message formats between requestor and service
- Routing messages between services
- Conversion of transport protocols between requestor and service
- Handling of business events from disparate sources

BPM provides the capabilities of a standards-based enterprise service bus. BPM manages the flow of messages between service requesters and service providers. Mediation modules within BPM handle mismatches between requesters and providers, including protocol or interaction-style, interface, and quality of service mismatches.

How To Do a BPM Sizing

When you select BPM sizing support from the ***Product Selection*** window to begin a new sizing, the ***BPM Application Definition*** input screen is presented.

The general approach to sizing BPM applications on System z is to determine which of the pre-defined SOA transaction profiles best represent the customer's application and provide an average transaction rate during a peak interval for each of these. Refer to the *BPM Application Definition* window to see the list of transaction profiles that can be included in a sizing. The **Profile** button for each transaction profile can be used to display a detailed description of the processing included in that transaction. The BPM Application Definition window has two tabs for z/OS only. One tab is for BPEL processes and the other tab is for mediation processes. A sizing for z/OS can include transaction profiles from one or both tabs.

The usual span of time for a peak interval is 15 minutes, and you want to specify the average transactions per second for that interval. Note that if you have statistics for the average transaction rate for prime shift or for a day or week, you might want to apply a peak-to-average multiplier factor to averages for long periods of time to arrive at an average rate for a 15 minute interval.

For long-running macroflows, the peak interval transaction rate is more difficult to determine because many macroflows do not begin or end during the peak interval. Try to arrive at a reasonable rate by first establishing how many macroflows begin and end within a peak hour (if any). For those that do not run completely within the hour, estimate the proportion of them that do run during the hour and add an additional number of transactions to reflect this. Then divide the total number of transactions by 3600 to arrive at trans/sec. For example, you might have 200 macroflows that begin and end during the peak hour, and you estimate that 40% of 62 other macroflows also run. The tran rate should be 200 + 62x0.4 = 224.8 / 3600 = 0.06 trans/sec. The **Help Calculating Transaction Rate** button on the BPM Application Definition window can be used to perform this calculation.

For the transaction profiles that access IMS/CICS/DB2 you need to specify an implementation option that determines what processing costs are included in the sizing. In some cases the BPM implementation will tie together existing native IMS/CICS/DB2 transactions that are not currently invoked from a WAS J2EE container environment. The sizing estimate should include the capacity required for a WAS container environment, BPM processing, and adapters to IMS/CICS/DB2 but not the IMS/CICS/DB2 transaction processing. In other cases the BPM implementation will tie together existing WAS J2EE transactions that invoke IMS/CICS/DB2 transactions, so the WAS container environment plus any adapters to IMS/CICS/DB2 already exist. In this case the sizing estimate should only include the capacity required for BPM processing. There may also be cases where there are no pre-existing IMS/CICS/DB2 or WAS J2EE transactions and the BPM implementation will tie together new WAS J2EE transactions that invoke new IMS/CICS/DB2 transactions. In this case the sizing estimate should include the capacity required for everything. The IMS/CICS/DB2 transactions that were run for this sizing support were trivial. If you have more complex IMS/CICS/DB2 transactions, additional capacity may be required.

For the mediation transaction profiles (z/OS only), web services bindings is used so you need to specify the request and reply message size (KB) in addition to the transaction rate for each transaction profile included in the sizing. For the Aggregation transaction profile you also need to select the fanout option that best represents the expected usage in the application being sized.

To see results, click on the **Summary Report**, **CPU Utilization** or **Transaction Rate** buttons in the Reports and Capacity Projections section of the window.

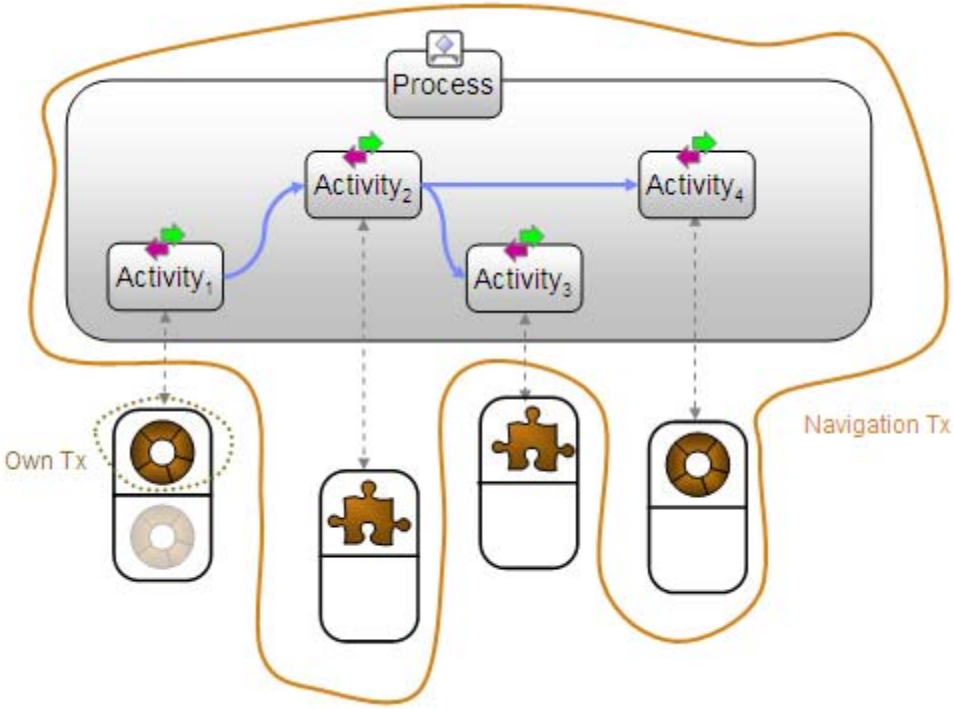
The application window images shown in this user guide have been provided as a representation of the windows the user will see when using **zPSG** but there may be minor differences from the current version of **zPSG,** such as version numbers and dates.

Transaction Descriptions

This section provides a list of the pre-defined SOA transaction profiles that can be included in a sizing. A transaction profile is included in a sizing by providing a transaction rate per second greater than zero. The implementation option and additional number of service invocations should also be specified if applicable. The transaction profiles are divided into three categories. The first category is for BPEL microflow processes, the second is for BPEL macroflow processes and the third is for Mediation processes (z/OS only).

What is a BPEL Microflow?

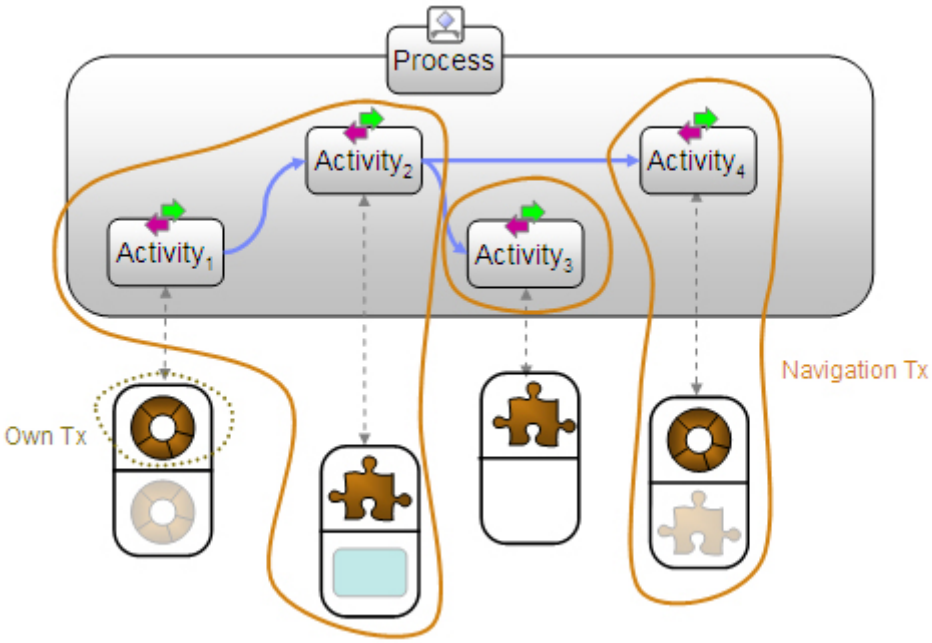
A microflow is of short-running, non-interruptible nature. It is transient, that is, its state only exists in memory. A microflow is executed within a single unit of work (either a global transaction or an activity session). It typically orchestrates synchronous, short-running services; in particular it cannot drive truly asynchronous service interactions or human tasks.



What is a Macroflow?

A long-running process may run for hours, days or even years. It is of interruptible nature thus allowing for asynchronous interactions with other services, such as waiting for unsolicited events, involving human interactions, or realizing complex business-to-business interaction with other stateful services. Such a process is realized as a series of global transactions that may be executed in parallel to support truly parallel branches

within the process. Its state is stored in a database. A long-running process comprises multiple global transactions. Each navigation step of a long-running process is performed in its own transaction. A navigation step may span multiple activities.



What is a Mediation?

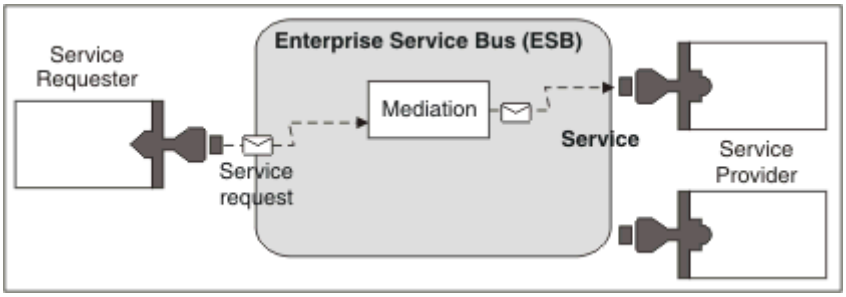
Interposing the ESB between participants enables you to modulate their interaction through a logical construct called a *mediation*. Mediations operate on messages in-flight between requesters and providers. For example, mediations can be used to find services with specific characteristics that a requester is asking for, or to resolve interface differences between requesters and providers. For complex interactions, mediations can be chained sequentially.

Using mediations, an enterprise service bus carries out the following actions between requester and service:

- *Routing* messages between services. An enterprise service bus offers a common communication infrastructure that can be used to connect services, and thereby the business functions they represent, without the need for programmers to write and maintain complex connectivity logic.
- *Converting* transport protocols between requester and service. An enterprise service bus provides a consistent, standards-based way to integrate business functions that use different IT standards. This enables integration of business functions that could not normally communicate, such as to connect applications in departmental silos or to enable applications in different companies to participate in service interactions.
- *Transforming* message formats between requester and service. An enterprise service bus enables business functions to exchange information in different

formats, with the bus ensuring that the information delivered to a business function is in the format required by that application.
- *Handling* business events from disparate sources. An enterprise service bus supports event-based interactions in addition to the message exchanges to handle service requests.

Figure 1. An enterprise service bus. The enterprise service bus is routing messages between applications, which are requesters or providers of services. The bus is converting transport protocols and transforming message formats between requesters and providers. In this figure, each application uses a different protocol (represented by the different geometric shapes of their connectors) and uses different message formats.



By using the enterprise service bus you can focus on your core business rather than your computer systems. You can change or add to the services if required; for example, to respond to changes in the business requirement, to add extra service capacity, or to add new capabilities. You can make the required changes by reconfiguring the bus, with little or no impact to existing services and applications that use the bus.

BPEL Microflow Transactions

Below are the detailed descriptions of the pre-defined BPEL microflow transaction profiles.

Tran A

Price Quote/Price Change
PVT1

Trivial microflow with business object interface (Lazy Parsing mode)

BPEL process:
- Receive
- 2 synchronous Invokes using SCA bindings over JCA
- 1 trivial JavaSnippet
- Reply

2 JCA adapters to IMS
- Average of 1k sent to IMS, 2k returned

Tran B

2PC Funds Transfer
SVT3

Moderate-complex microflow with business object interface (Lazy Parsing mode)

BPEL process:
- Receive
- 1 JavaSnippet
- 1 synchronous Invoke implemented as a Session Facade EJB via JPA to DB2 for z/OS
- 1 synchronous Invoke using SCA bindings over JCA CTG adapter to CICS
- Reply

1 JCA adapter to CICS
- Average of 1k sent to CICS, 4k returned

Tran C

Insurance Claim Automated Approval SOABench 2008 Automated Approval	Complex microflow with Java and web services invokes (Lazy Parsing mode) BPEL process: Receive 1 Choice 5 web services invokes 42 variable assignments across 7 assignment blocks Reply By default, a 3 KB BO request and reply size is used.

BPEL Macroflow Transactions

Below are the detailed descriptions of the pre-defined BPEL macroflow transaction profiles.

Tran D

New Order SVT10	A macroflow (long-running process) with a Human Task node that processes a New Order based on input from the Human Task. If accepted, the NewOrder SCA Service is invoked and a NewOrder record is written to the ERWW database using a CICS transaction. (Lazy Parsing mode)
	BPEL long-running process (macroflow) Receive 1 SCA invoke 3 Java snippets 1 Choice activity 1 Human Task Reply

Tran E

Insurace Claim Manual Approval SOABench 2008 Outsourced	More complex macroflow using optimizations for long running processes (Lazy Parsing mode)
	2 BPEL processes (1 microflow + 1 macroflow): 1 Receive 3 web services invokes 1 long-running process invoke 41 variable assignments across 6 assignment blocks 1 Reply 1 Receive 4 two-way web services invokes 4 one-way web services invokes that wait on 4 corresponding receives 32 variable assignments across 10 assignment blocks 1 parallel activity 1 Reply By default, a 3 KB BO request and reply size is used.

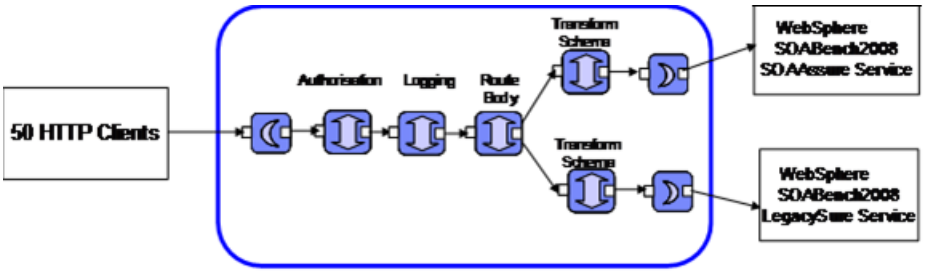
Mediation Transactions

Below are detailed descriptions of the pre-defined mediation transaction profiles which use web services bindings.

Tran F

Composite

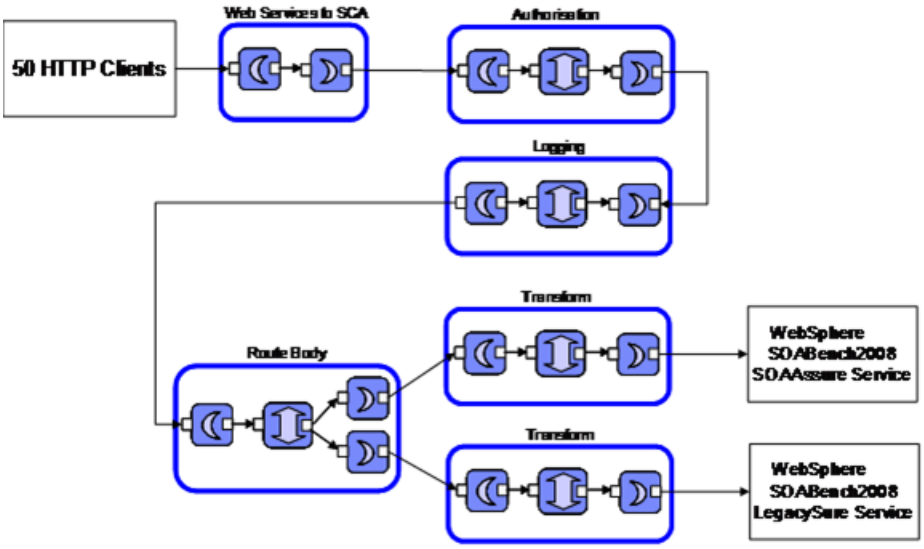
The composite mediation consists of four mediation primitives wired together inside a single mediation module. This saves the overhead of inter-module call overheads, but at the expense of the ability to individually administer the pieces of the overall mediation. The Authorisation mediation is a routing mediation which checks a password field in the request body. No logging is performed.



Tran G

Chained

The chained mediation performs the same function as the composite mediation but the four types of mediation primitives are each packaged as separate mediation modules, which are then joined together using bindings.



Tran H

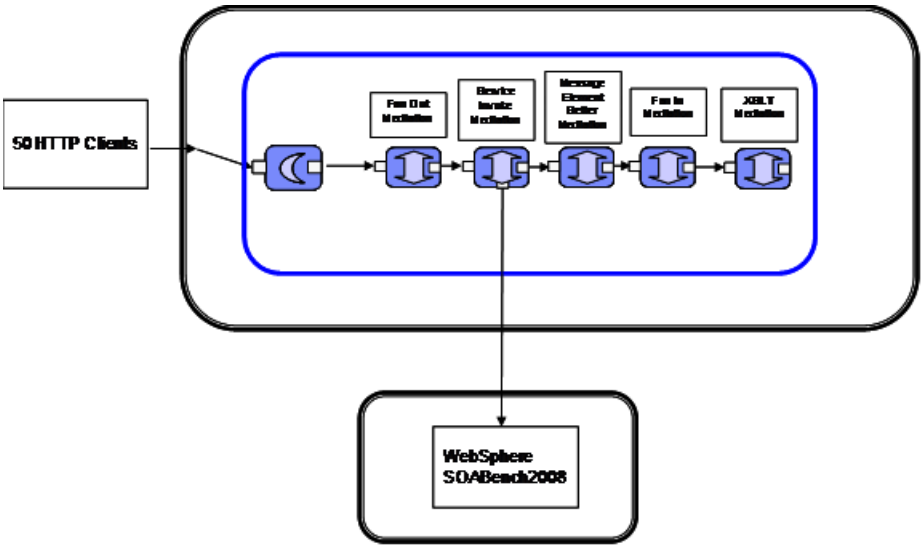
Aggregation

The Aggregation mediation allows you to iterate over a repeating element in the request message. This mediation iterates over the following:

- Perform a Service Invoke to call a target Web Service
- Use a Message Element Setter to update the shared context with some data from the response. The shared context was created on the Input node.

The Aggregation mediation will then wait for all iterations to complete, before using an XSLT mediation to create a response message. It then returns the response.

This test was then executed with different request messages so that we would get a different number of Fan Out iterations. It was executed with requests that would result in 1, 2 and 4 Fan Out iterations. Note that each iteration is run sequentially, rather than in parallel.



Tran I

Service Gateway Body

The Service Gateway Body mediation utilizes the Message Filter and Message Element Setter primitives to provide dynamic endpoint resolution – filtering inbound messages dependant on data contained in the message body. This approach to Mediation Flow design provides a single access point for multiple services, and allows for common processing to be performed on all inbound and outbound messages.

This is the type of processing that might be used to process a car insurance quotation, to obtain information pertaining to the make and model of the vehicle. Providing a single access point, and utilizing the dynamic endpoint resolution functionality to locate an appropriate service to invoke – depending on whether the quote is for a BMW, Ford, Vauxhall, etc.

Service Gateway Body Request Flow

Service Gateway Body Response Flow

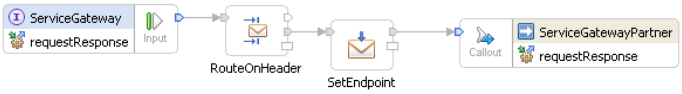
Tran J

Service Gateway Header

The Service Gateway Header mediation utilizes the Message Filter and Message Element Setter primitives to provide dynamic endpoint resolution – filtering inbound messages dependant on data contained in the message header. This approach to Mediation Flow design provides a single access point for multiple services, and allows for common processing to be performed on all inbound and outbound messages.

This is the type of processing that might be used to process a car insurance quotation, to obtain information pertaining to the make and model of the vehicle. Providing a single access point, and utilizing the dynamic endpoint resolution functionality to locate an appropriate service to invoke – depending on whether the quote is for a BMW, Ford, Vauxhall, etc.

Service Gateway Header Request Flow



Service Gateway Header Response Flow

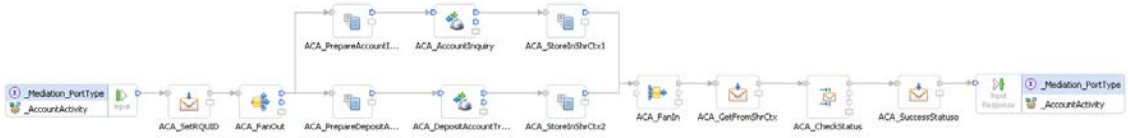
Tran K

SOA Bank

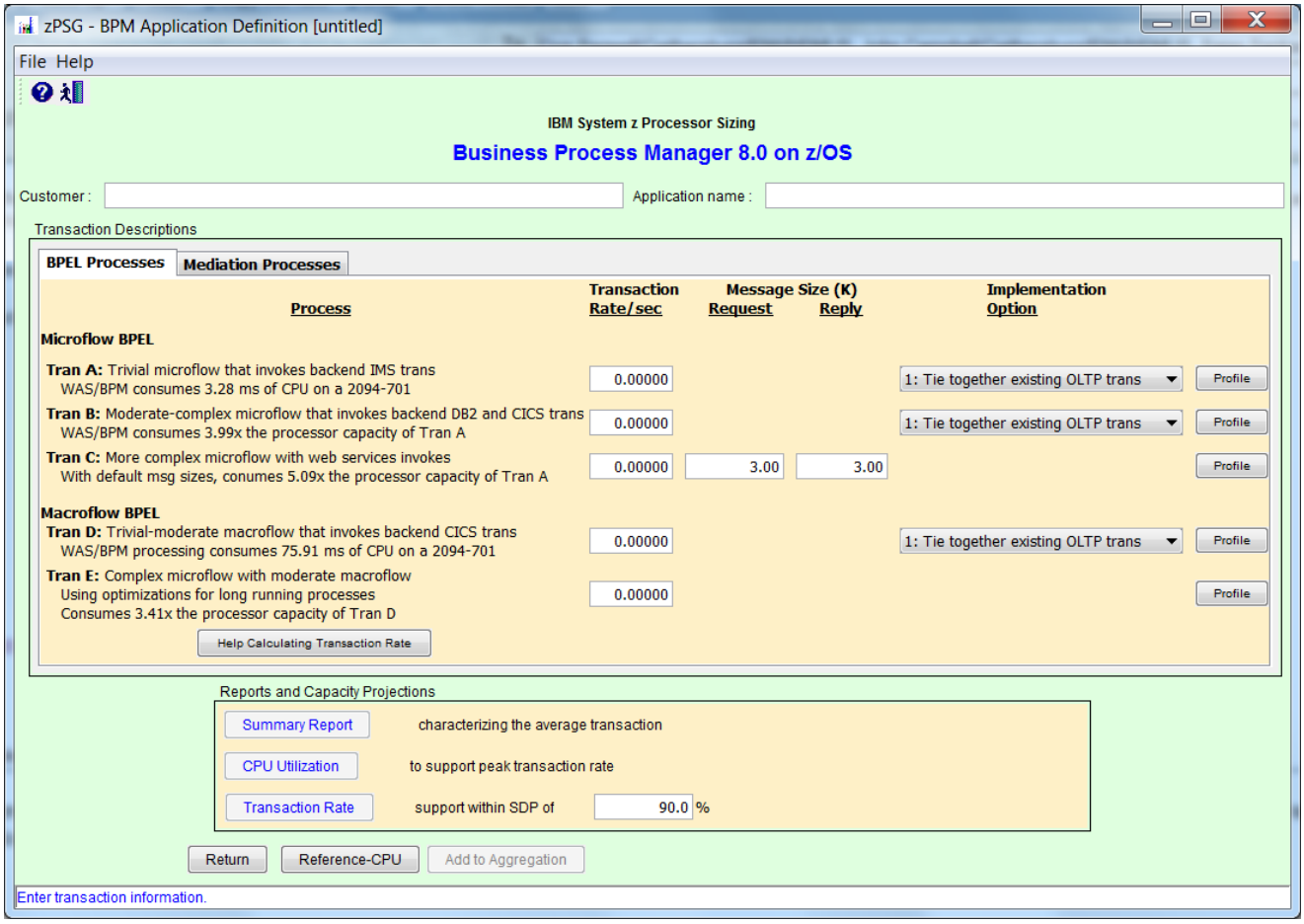
The SOABank mediation flow takes in a request message which contains information for a customer with two difference bank accounts, for example a current and a savings account. Within a FanOut / FanIn block the flow uses the Mapping primitive (targeting Business Object Map) to select the appropriate account information from the request and sends account enquiries to two backend services using the Service Invoke primitive. Further Mapping primitives (targeting Business Object Map) are used to copy the responses from these two services into the shared context and combine them into a single response. A Message Element Setter primitive is used to move the response from the shared context into the message body. Message Filter and Message Element Setter primitives are used to record whether the service invocations were successful.

This type of processing is often used to combine information from multiple backend services into a single response to the customer. The SOABank scenario makes use of large industry standard schemas.

SOABank Request / Response Flow



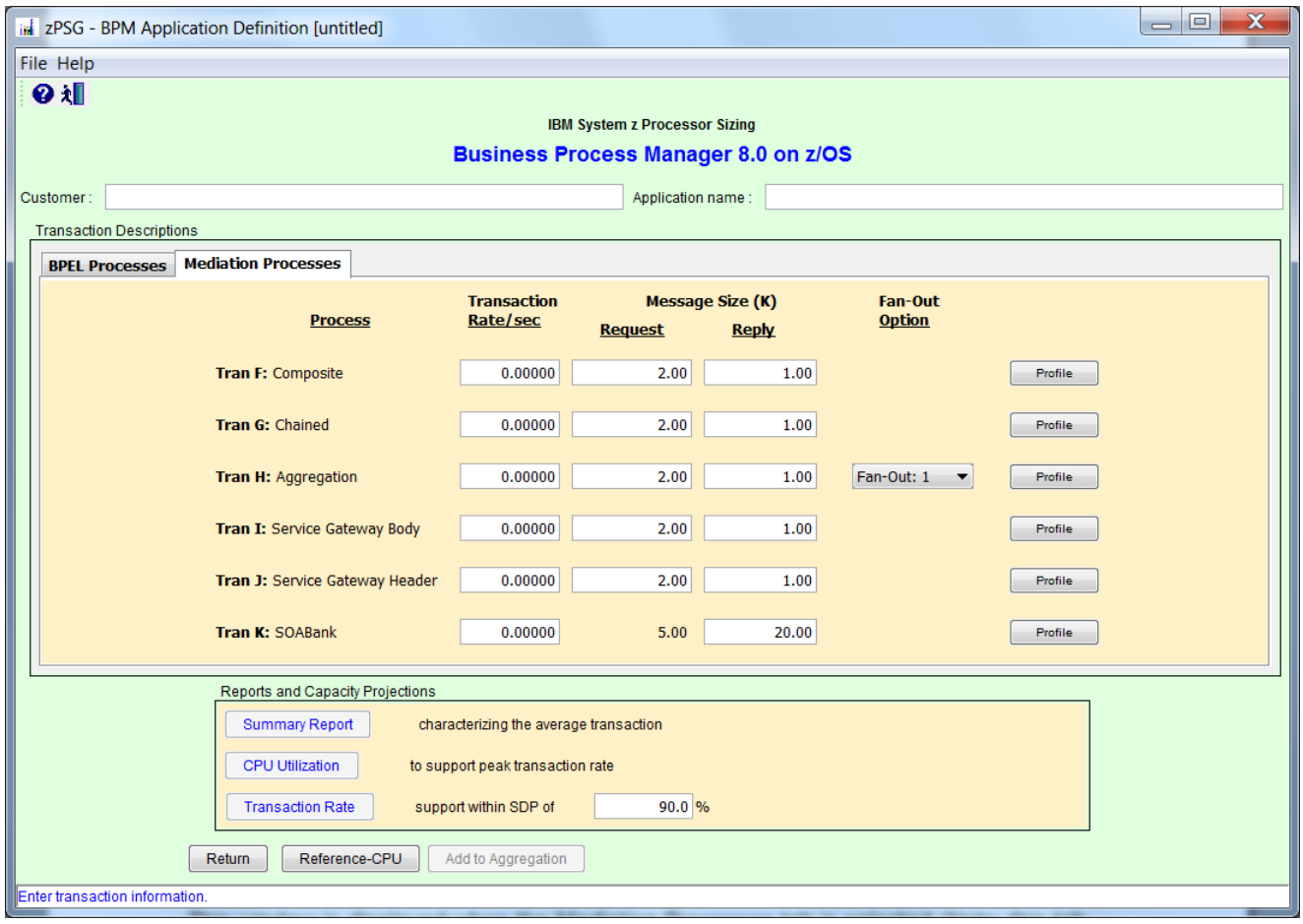
BPM Application Definition



This window is displayed when the **PSG** button is clicked on the ***Product Selection*** window when **Business Process Manager** has been selected for z/OS or Linux (the example used here is for z/OS).

Note: A transaction rate greater than zero must be specified for at least one transaction profile on either the BPEL Processes or the Mediation Processes tab to get a sizing estimate.

BPM Mediations Application Definition



This window is displayed when the Mediation Processes tab is selected (Note: this tab is supported for z/OS only).

Description of Input Fields

Menu bar

File

New	Start a new study. Sets all fields to initialization values.
Load	Load a previously saved study
Save	Save the current study
Save as	Save the current study as a new file
Exit	Exit window and return to the ***Product Selection*** window (Ctrl-E)
Exit zPSG	Terminate **zPSG** execution (Ctrl-Q). Exit zPSG can also be invoked from the Exit zPSG button on the tool bar.

Help

Context Help (F1)	Help for this window
About zPSG	Product information

Toolbar

? button	Click this button to go to Help for this window.
Exit zPSG button	Click this button to terminate **zPSG** execution.

Customer =
Input field, for documentation purposes, not required. If you want to save a copy of the sizing estimate, you can use this field to document which sizing it is.

Application name =
Input field, for documentation purposes, not required.
If you want to save a copy of the sizing estimate, you can use this field to document which sizing it is.

Input Fields and Buttons

Transaction Rate/sec

Specify a transaction rate per second value for each of the transaction profiles that should be included in the sizing. Specify a value of zero if the transaction profile should not be included in the sizing. The default value for each of the transaction profiles is zero.

Note: You can use the **Help Calculating** Transaction Rate button for assistance in calculating the transaction rate per second for long-running macroflow processes.

Message Size (K) - Request

Specify Request Business Object (BO) size in KB. The default value varies by transaction.

Message Size (K) - Reply

Specify Reply Business Object (BO) size is in KB. The default value varies by transaction.

Implementation Option (BPEL only)

For the transaction profiles that access IMS/CICS/DB2 you need to specify one of the three implementation options that describe your BPM implementation. Click on the drop-down icon at the right of the Implementation Option selection to choose one of the following options:

1. Tying together existing native IMS/CICS/DB2 transactions that are not currently invoked from a WAS J2EE container environment. BPM estimate should include the capacity required for a WAS container environment, BPM processing, and adapters to IMS/CICS/DB2 but not IMS/CICS/DB2 transaction processing. This is the default assumption.
2. Tying together existing WAS J2EE transactions (that may or may not invoke IMS/CICS/DB2 transactions), so the WAS container environment plus any adapters to IMS/CICS/DB2 already exist. BPM estimate should include only the additional processing required for BPM functions.
3. No pre-existing WAS or IMS/CICS/DB2 transactions. New OLTP transactions are being developed with SOA application and BPM estimate should include everything.

Note: Option 2 is only supported for Tran A. The implementation options are also described in a tool tip that is displayed when the mouse pointer is over the Implementation Option column header name.

Fan-Out Option (Mediation Processes only)

For Tran H there is a fan-out option of 1, 2, or 4. The default is one.

Profile button

Click this button to view a detailed description of the processing for the transaction associated with this button. Each transaction has a Profile button.

Help Calculating Transaction Rate button (BPEL only)

Click this button to get assistance with calculating the transaction rate for long-running macroflow processes using the ***BPM Macroflow Process Trans Rate*** window that will be displayed. Refer to the *BPM Macroflow Process Trans Rate* window for additional information.

Reports and Capacity Projections

This section provides buttons to view output windows with summary reports and capacity projections.

Summary Report button

Click this button to view a summary of the input assumptions for the sizing and a breakdown of the CPU/transaction among the transaction profiles included in the sizing.

CPU Utilization button

Click this button to see an output window with estimates of processor utilization for all System z processors supported in zPSG.

Transaction Rate button

Click this button to see an output window with estimates of transaction rates that can be supported on all System z processors supported in zPSG. You can also see the transaction rates that can be supported within a Saturation Design Point (SDP) specified for the processors.

SDP %

Description

Input field, numeric, valid range is 1 to 100.
SDP stands for Saturation Design Point. This is a classic capacity planning concept which allows you to examine the amount of workload than can be supported in less than the full capacity of the processor model. It applies to the Transaction Rate output window and enables you to determine how much work can fit into a processor that is already being used for other applications.

Default

The default is 90%.

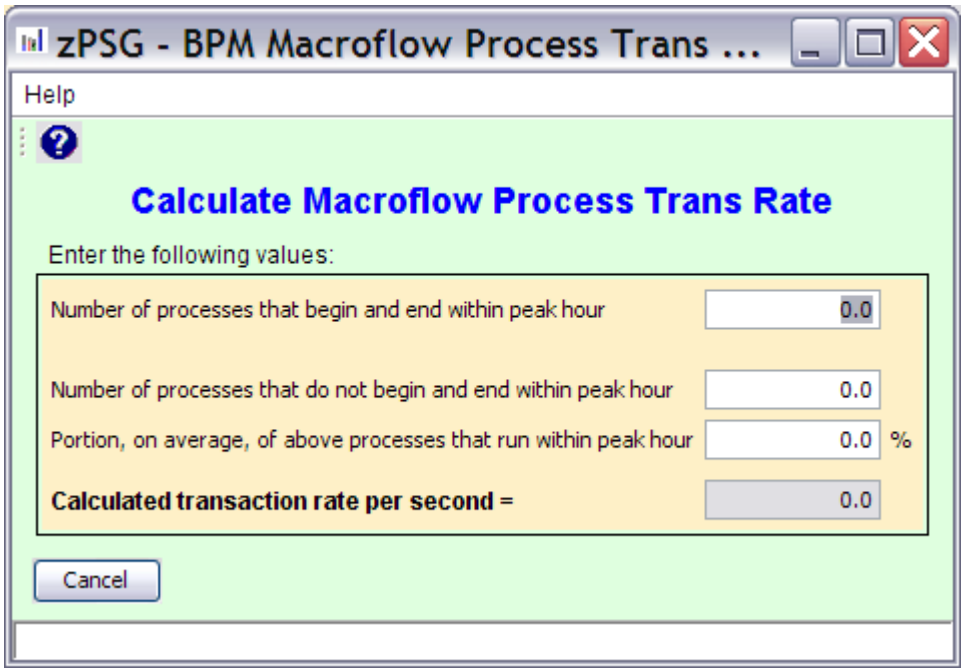
Return button

Click this button to return to the ***Product Selection*** window.

Reference-CPU button

Click this button to go to a window to change the System z processor used as a basis for capacity ratings. See the Reference-CPU section in the zPSG User's Guide for information about this setting.

BPM Macroflow Process Trans Rate



This window is displayed when the **Help Calculating** Transaction Rate button is clicked on the primary ***BPM Application Definition*** window. It is used to calculate the transaction rate per second for long-running macroflow processes.

Description of Input Fields

Menu bar

Help

Context Help (F1) Help for this window

About zPSG Product information

Toolbar

? button

Click this button to go to Help for this window.

Push Buttons

Click the **Cancel** button to close the window.

Number of processes that begin and end within peak hour

Required input field. Specify the number of processes that begin and end within the peak hour. Enter 0 if none of the processes begin and end within the peak hour.

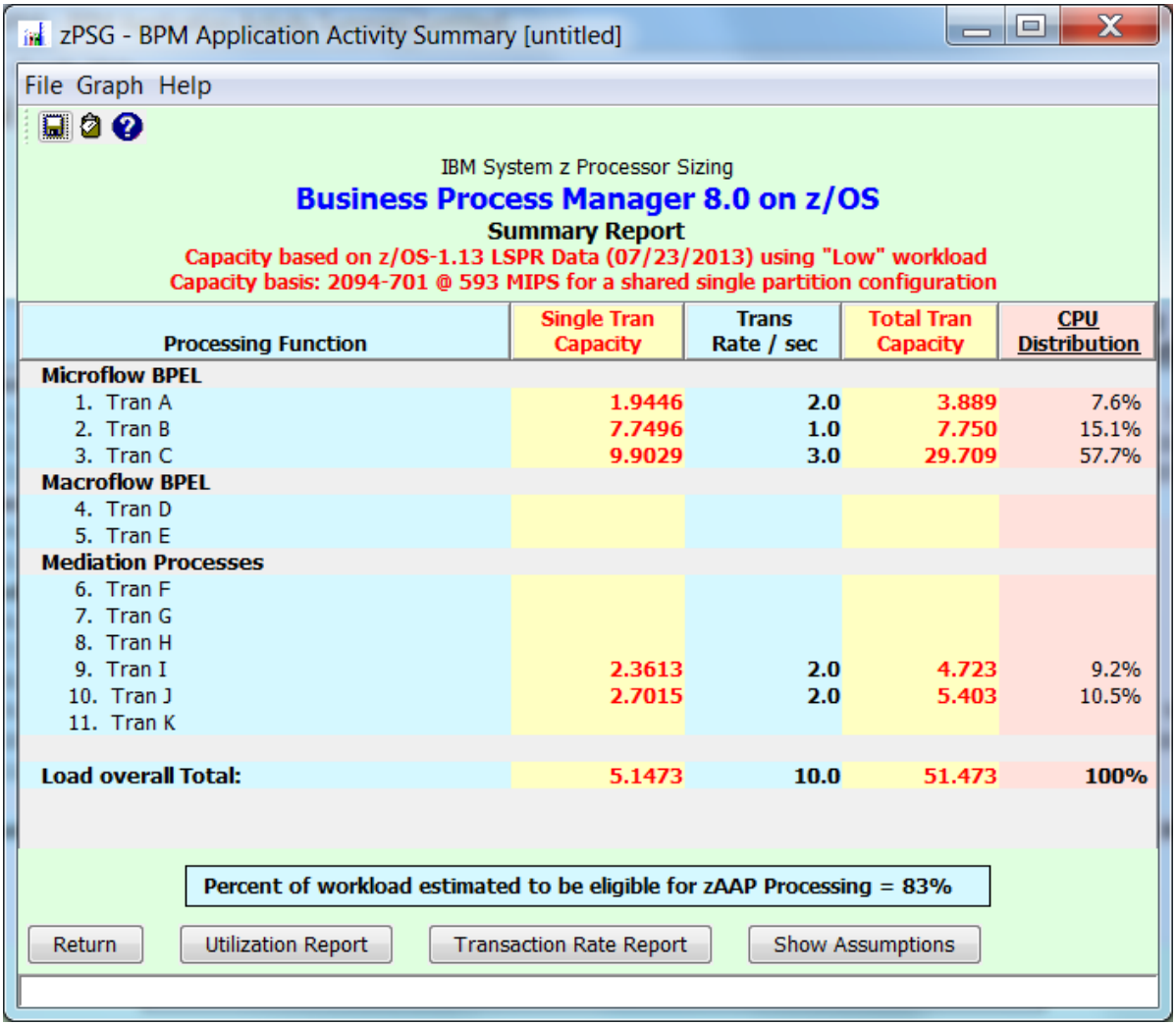
Number of processes that do not begin and end within peak hour
Required input field. Specify the number of processes that do not begin and end within the peak hour. Enter 0 if all the processes begin and end within the peak hour.

Percent, on average, of above processes that run within peak hour
Required input field. Specify the percentage of processes entered in the previous input field that ran during the peak hour. This percentage should be entered as a number between 0 and 100.

Description of Output Fields

Calculated transaction rate per second =
The number of transactions per second calculated from the values entered in the three input fields.

BPM Application Activity Summary



This window is displayed when the **Summary Report** button is clicked on the primary ***BPM Application Definition*** window. It shows a breakdown of the CPU per transaction for the various pre-defined transactions included in the sizing.

Menu bar

File

Output	Write contents to a flat (PRN) file.
Copy	Write contents to Window's clipboard

Graph — Generates a pie chart showing the distribution of application activity

Help

Context Help (F1)	Help for this window
About zPSG	Product information

Toolbar

1st button
Click this button to send sizing information to a PRN file for processing outside of zPSG.

2nd button
Click this button to send sizing information to the clipboard, so that you can copy it into a note or other document.

? button
Click this button to go to Help for this window.

***Processing Function* column**

Lists the pre-defined transactions available on the primary ***BPM Application Definition*** window.

***Single Tran Capacity* column**

Reflects the amount of CPU (as represented by the Capacity Rating) for each transaction, and at the bottom for all transactions.

***Trans Rate / sec* column**

Reflects the number of transactions completed per second for each of the pre-defined transactions as specified on the ***BPM Application Definition*** window.

***Total Tran Capacity* column**

Reflects the amount of CPU (as represented by the Capacity Rating) for each transaction multiplied by the transaction rate specified in the ***Trans Rate / sec*** column, and at the bottom for all transactions.

***CPU Distribution* column**

Shows the percentage of the CPU/transaction used by each transaction.

Percent of workload estimated to be eligible for zAAP Processing = (z/OS only)

Shows the estimated percentage of Java content for the pre-defined transactions included in the sizing. Percentages of Java content were computed in all the performance lab measurements done to support the pre-defined transactions supported in this tool. Depending on what pre-defined transactions you included, the percentage per transaction will vary. These percentages reflect the amount of CPU that we estimate you can offload to zAAP, assuming sufficient zAAP capacity to handle the load. You can generate an estimate of zAAP capacity requirements using the zAAP Capacity Estimator available from the CP Calculator menu on the ***Product Selection*** window.

Notice Concerning Specialty Engines

Neither **zPSG** nor this document provides descriptions of the types and portions of workloads that are eligible for execution on Specialty Engines (e.g., zAAP. zIIP, and IFL). IBM authorizes customers to use IBM Specialty Engines only to process Eligible Workloads of specific Programs expressly authorized by IBM. These programs are specified in the "Authorized Use Table for IBM Machines", found at:

www.ibm.com/systems/support/machine_warranties/machine_code/aut.html

No other workload processing is authorized for execution on an SE.

IBM offers Specialty Engines at a lower price than General Processors/Central Processors because customers are authorized to use Specialty Engines only to process certain types and/or amounts of workloads as specified by IBM in the AUT.

Push Buttons

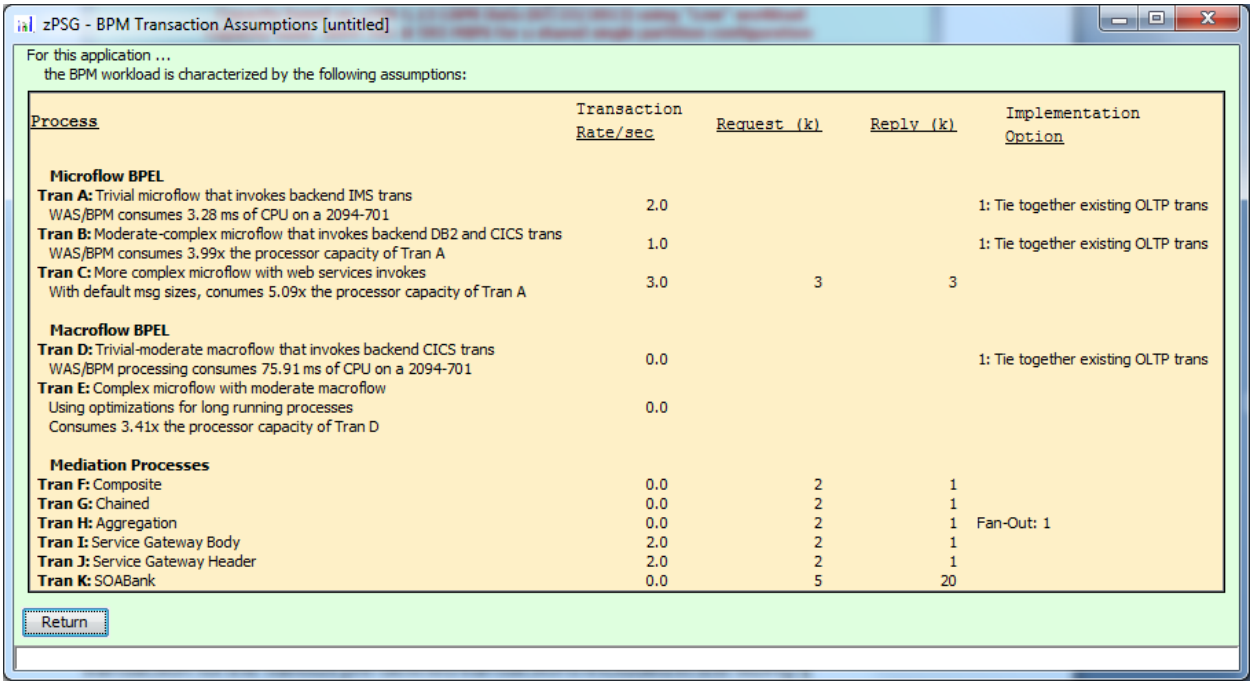
Click the **Return** button to return to the primary ***BPM Application Definition*** input window.

Click the **Utilization Report** button to go to the ***BPM Processor Capacity Projections - Processor Utilization*** output window.

Click the **Transaction Rate Report** button to go to the ***BPM Processor Capacity Projections - Transaction Rate Supported*** output window.

Click the **Show Assumptions** button to see a list of the assumptions for the sizing in the ***BPM Application Transaction Assumptions*** window.

BPM Transaction Assumptions

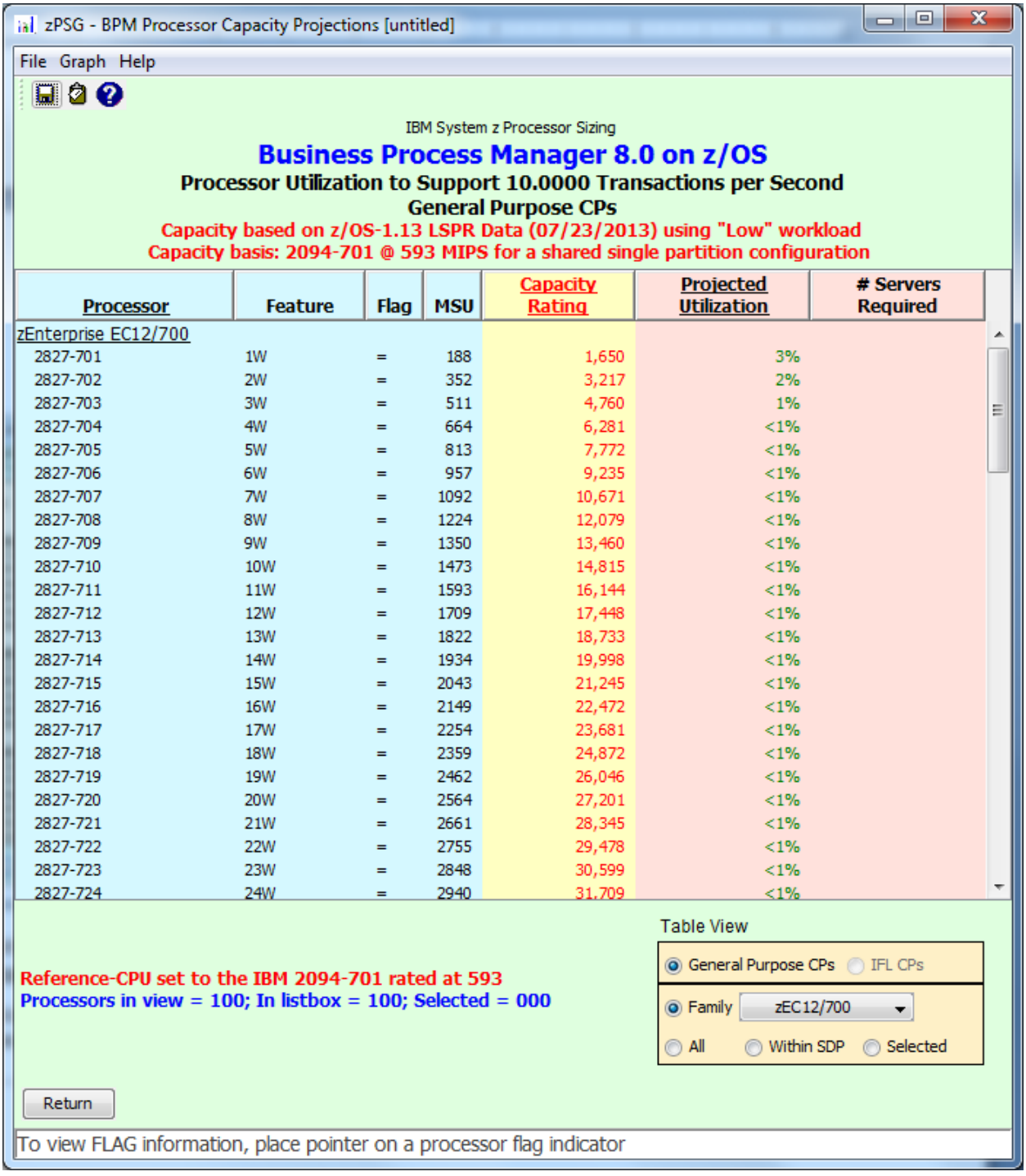


This window is displayed when the **Show Assumptions** button is clicked on the ***BPM Application Activity Summary*** window.

All assumptions as listed will be included when generating output for the ***Summary*** window.

BPM Processor Utilization

BPM Processor Capacity Projections



This window is displayed when the **Utilization** button is clicked on the ***BPM Application Definition*** window or the **Utilization Report** button is clicked on the ***BPM Application Activity Summary*** window.

Menu bar

File

Output	Write report contents to a flat (PRN) file.
Copy	Write report contents to Window's clipboard

Graph (for processors currently selected in table)

Capacity	Generate bar graph depicting capacity values
Utilization	Generate bar graph showing utilization on selected processors

Help

Context Help (F1)	Help for this window
About zPSG	Product information

Toolbar

1st button
Click this button to send sizing information to a PRN file, for processing outside of zPSG.

2nd button
Click this button to send sizing information to the clipboard, so that you can copy it into a note or other document.

? button
Click this button to go to Help for this window.

Table

***Processor* column**
A list of all processor models supported in zPSG

***Feature* column** **For z/OS & Linux**
Using the General Purpose CPs option under Table View, a designation of how many general purpose processing engines (CPs) for this entry. For example, 4W ("W" is short for "way") indicates 4 CPs or engines. Also see ***Flag*** column below.

***Feature* column** **For Linux only**
Using the IFL CPs option under Table View, a designation of how many IFL engines for this entry. For example, 4W IFL ("W" is short for "way") indicates 4 IFL engines. Also see ***Flag*** column below.

***Flag* column**
If you place your cursor on a row in this column, an explanatory message about the System z model designation and the number of CP or IFL engines for the entry.

***MSU* column**
Only for the General Purpose CPs Table View (does not apply to IFLs). Shows the MSU rating assigned to the number of CP engines for this entry.

Capacity Rating **column**
The capacity ratings reflect the relative capacity of each processor table entry to the reference-CPU and its capacity rating assigned on the Reference-CPU window. When **zPSG** is started the reference-CPU will be set to a 2094-701 (a z9 EC/700 processor with 1 general purpose CP) with a capacity rating of 602 MIPS.

Projected Utilization **column**
Shows the estimated CPU% for each processor entry in the table, based on the transaction rate(s) and implementation options specified for the pre-defined transactions. This is the primary output for a sizing.

***Servers Required*** **column**
If the estimated CPU% is greater than 100% (and therefore cannot fit on the processor), this column reflects the number of these models that would be needed to accommodate the load.

Table View Options Box

Click a radio button in each section to customize the processor entries shown in the table:

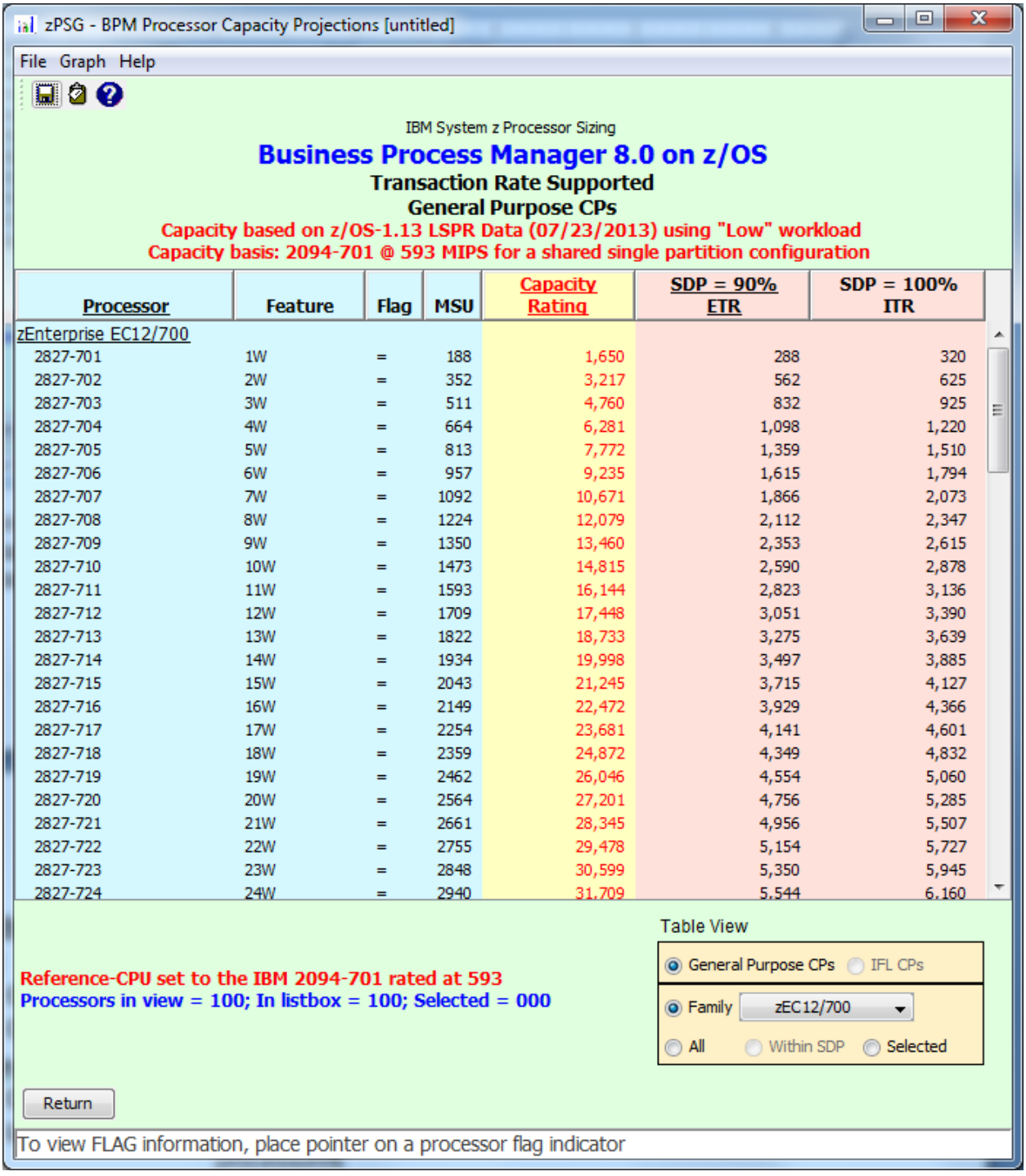
- **General Purpose CPs** shows entries with some number of general CP engines
- **IFL CPs** shows entries with some number of IFL engines (for Linux only)
- **Family** shows all processor models for the family selected (Default)
- **All** shows all processor models supported in zPSG
- **Within SDP** shows all models that can accommodate the load within the Saturation Design Point
- **Selected** shows only selected models. Models are selected by clicking on the entry while holding down the Ctrl key on your keyboard.

Return button

Click this button to return to the primary ***BPM Application Definition*** window.

BPM Transaction Rate Supported

BPM Processor Capacity Projections



This window is displayed when the **Transaction Rate** button is clicked on the ***BPM Application Definition*** window or the **Transaction Rate Report** button is clicked on the ***BPM Application Activity Summary*** window.

Menu bar

File

Output	Write report contents to a flat (PRN) file.
Copy	Write report contents to Window's clipboard

Graph (for processors currently selected in table)

Capacity	Generate a bar graph depicting capacity values
ETR	Generate bar graph showing transaction rate supported at SDP
ITR	Generate bar graph showing maximum transaction rate supported

Help

Context Help (F1)	Help for this window
About zPSG	Product information

Toolbar

1st button
Click this button to send sizing information to a PRN file, for processing outside of zPSG.

2nd button
Click this button to send sizing information to the clipboard, so that you can copy it into a note or other document.

? button
Click this button to go to Help for this window.

Table

***Processor* column**
A list of all processor models supported in zPSG

***Feature* column** **For z/OS & Linux**
Using the General Purpose CPs option under Table View, a designation of how many general purpose processing engines (CPs) for this entry. For example, 4W ("W" is short for "way") indicates 4 CPs or engines. Also see **Flag** column below.

***Feature* column** **For Linux only**
Using the IFL CPs option under Table View, a designation of how many IFL engines for this entry. For example, 4W IFL ("W" is short for "way") indicates 4 IFL engines. Also see **Flag** column below.

***Flag* column**
If you place your cursor on a row in this column, an explanatory message about the System z model designation and the number of CP or IFL engines for the entry.

***MSU* column**
Only for the General Purpose CPs Table View (does not apply to IFLs). Shows the MSU rating assigned to the number of CP engines for this entry.

***Capacity Rating* column**
The capacity ratings reflect the relative capacity of each processor table entry to the reference-CPU and its capacity rating assigned on the Reference-CPU window. When **zPSG** is started the reference-CPU will be set to a 2094-701 (a z9 EC/700 processor with 1 general purpose CP) with a capacity rating of 602 MIPS.

SDP= xx % -- ETR column
Shows the transaction rate for the application that can be supported within the Saturation Design Point specified on the primary ***BPM Application Definition*** window (the default SPD is 90%). ETR stands for External Throughput Rate, which is a standard System z term for transaction rate.

SDP=100% -- ITR column
Shows the transaction rate for the application that can be supported at 100% CPU. ITR stands for Internal Throughput Rate, which is a standard System z term indicating the throughput that can be achieved at 100% CPU. ITR is computed by dividing the ETR by the CPU% (expressed as a decimal). This is the way to correctly rate the processor capacity of each entry in the processor table for this workload (as opposed to MIPS ratings, which are generally erroneous).

Table View Options Box

Click a radio button in each section to customize the processor entries shown in the table:

- **General Purpose CPs** shows entries with some number of general CP engines
- **IFL CP's** shows entries with some number of IFL engines (for Linux only)
- **Family** shows all processor models for the family selected (Default)
- **All** shows all processor models supported in zPSG
- **Within SDP** shows all models that can accommodate the load within the Saturation Design Point
- **Selected** shows only selected models. Models are selected by clicking on the entry while holding down the Ctrl key on your keyboard.

Return button

Click this button to return to the primary ***BPM Application Definition*** window.

BPM Sizing Assistance

Here are instructions for accessing the System z questionnaire from Techline. Note that on the Techline websites there are sizing questionnaires for distributed platforms in addition to System z. Be sure to use System z questionnaires for System z sizings. The questions and sizing methodologies are different from distributed platforms.

For IBMers:

1. Obtain the latest copy of the Business Process Manager sizing questionnaire from the following website:
   - http://w3-03.ibm.com/support/techline/global/swsz.html
2. Submit a sizing request to Techline using the instructions found in the sizing questionnaire.

For Business Partners:

1. Obtain the latest copy of the Business Process Manager sizing questionnaire via:
   - Phone: Call PartnerLine at 1-800-426-9990 (US and Canada)
   - Email: pwcs@us.ibm.com
   - Online: http://www.ibm.com/partnerworld/techline
2. Submit a sizing request to Techline using the instructions found in the sizing questionnaire.

BPM Glossary of Terms

Bindings Mode Connection
When a JMS connection is made in bindings mode, MQ JMS uses the Java Native Interface (JNI) to call the MQ Queue Manager directly rather than communicating over TCP/IP. This connection mode is much more efficient when the sender and receiver reside in the same image of z/OS. Connections that require TCP/IP are called TCP mode connections.

BMP
A type of entity bean with Bean Managed Persistence. This means that the programmer must add code to persist the contents of the entity bean to the data base.

Cached Handshake
See SSL Handshake.

CCF
Crypto Co-Processor Facility. On S/390 and z900 processor models, 1 or 2 CCFs are included on every processor. They can be used to off-load some SSL processing from the general CPs. Processing can be off-loaded for full SSL handshakes, which reduces that CPU cost by 90% or more, and for TDES encryption & decryption, which reduces that cost by about 50%. CCFs are supported by SSL under z/OS but not under Linux. See Crypto Hardware.

CFA **What else, what functions are supported for CFA under z/OS & Linux ?**
Crypto Facility Accelerator **??** On z990 processors,

Cipher
See SSL.

Client Authentication
See SSL Client Authentication.

CMP
A type of entity bean with Container Managed Persistence. Using this type of entity bean, the EJB container is responsible for persisting bean contents to the data base.

Crypto Hardware
Either co-processors (CCFs or CFAs) or cards (PCICA, PCICC, PCIXCC) installed in zSeries processors that off-load some SSL processing from the general CP engines.

CTG
CICS Transaction Gateway, the IBM product providing JCA (J2EE) data connector support from WAS to CICS

Cursor

See DB2 Cursor

Data connector
Software that provides support for communication between WAS and back-end applications. Data connectors are used to send transactions or requests, with the accompanying input data and parameters, to a back-end application like CICS or IMS, and to return the transaction response or request results to WAS.

DB2 Connect
The IBM middleware product that provides access from WAS to DB2 data bases running in separate system images from WAS when ASCII to EBCDIC translation is needed. DB2 Connect is used in our performance measurements to access DB2 on z/OS from WAS on Linux.

DB2 Cursor
An API used when multiple rows (records) may be returned by DB2 for a SQL select (read) statement. The API consists of a Declare Cursor, an Open Cursor which initiates the building of the result set of rows by DB2, and a processing loop of Fetch to return each row to the application. Cursors may be open for read only or for update.

DOM
Document Object Model. When you parse an XML document using DOM, you create a tree structure (a program object) in memory, representing the contents of the XML document. The programmer can navigate the tree structure and add, modify, or delete its elements. DOM parsing uses more CPU and more memory than SAX parsing.

DTD
Document Type Definition. Used in XML validation processing. A DTD describes the grammar that constrains an XML document. If, for example, an XML-format personnel file contains entries for many employees, each of which must have 1 social security number, the DTD would contain a rule enforcing the occurrence of 1, and only 1, SSN per employee. The rules that may be described using a DTD are fairly limited in scope. For more extensive control over the contents of an XML document, use an XML Schema instead of a DTD.

EJB
Enterprise Java Bean. This is a specialized Java bean which is architected to provide enterprise-class behavior (transactional support, security, etc.). EJB support is one of the technologies in the J2EE specification.

EJB Container
The EJB Container in WAS provides the runtime environment for enterprise beans.

Entity Bean
A type of EJB which represents permanent data. An entity bean persists its contents to the data base.

Express Message
Also called non-persistent message. Guaranteed to be delivered by MQ at most once, unless there is a system failure. Not hardened to DASD. Deleted when receipt is acknowledged.

Full Handshake (aka non-cached handshake**)**
See SSL Handshake.

Handshake
See SSL Handshake.

Hashing Algorithm
See SSL.

HTTP, HTTPS
HyperText Transfer Protocol, HyperText Transfer Protocol Secure. HTTP is the protocol used for non-SSL communications on the web. HTTPS is for SSL communications.

IMS Connect
An IBM product which provides TCP/IP access to IMS. Recent versions of IMS Connect also provide local mode access to IMS applications on the same system as WAS.

IMS Connector for Java
Data connector runtime support for accessing IMS transactions from WAS applications using J2C (JCA) connector technology.

Java Class
A definition for a certain type of Java object.

Java Method
One instance of a Java class or object.

Java Object
One instance of a Java class. For example, if I have a class called "Animal", I might create an instance of "Animal" called "Rover", to represent my dog.

JPA
Java Persistence Architecture

JCA Connector
A means for a WAS application to interact with other system components (CICS, IMS, MQ). A JCA connector conforms to the Java Connector Architecture.

JDBC
Java Data Base Connectivity. JDBC is commonly used to access data in DB2, or other relational data bases, from Java applications.

JMS
Java Message Service. A peer to peer communication facility that can be used by software components or applications, usually in conjunction with MQ Series.

JSP
Java Server Page. JSPs are similar to static HTML pages, but they provide a programming interface which can be used to add dynamic content to the page.

J2EE
Java 2 Platform, Enterprise Edition. The server side platform which provides standard support for EJBs and other enterprise-class technologies in Java.

Local Mode
In the context of JCA Connectors, local mode refers to a means of accessing CICS or IMS without using TCP/IP sockets. Local mode is generally more efficient since it is optimized to exploit the fact that the caller and callee are on the same system.

MQ Message
A string of bytes that is meaningful to the applications that use it

MQ Queue
A named data structure for holding messages until they are retrieved by an application. Multiple senders and receivers can be associated with a single queue.

MQ Queue Manager
A named group of address spaces that run as a z/OS subsystem and manage the resources associated with WebSphere MQ. Applications connect to a Queue Manager using its name.

Parse, parser, parsing
A parser is a program that facilitates the interpretation of XML documents, and the extraction of XML data.

PCICA Card
Peripheral Component Interconnect (PCI) Cryptographic Accelerator card. Offloads some SSL handshake processing from general CP engines on zSeries.

PCICC Card
Peripheral Component Interconnect (PCI) Cryptographic Coprocessor card. Offloads some SSL processing from general CP engines on zSeries.

PCIXCC Card
Peripheral Component Interconnect Extended (PCIX) Cryptographic Coprocessor card. Offloads some SSL processing from general CP engines on zSeries.

Persistent Message
A persistent message is guaranteed to be delivered by MQ once and only once. It must be written to a file or a database to guarantee delivery.

Point-To-Point Messaging
This messaging model enables the delivery of an MQ message to only one recipient, also called a consumer.

Publish/Subscribe Messaging
This messaging model supports the delivery of an MQ message to multiple recipients called topic subscribers.

Queue Manager
See MQ Queue Manager.

RC4/MD5
The most commonly used SSL cipher and hashing algorithm. With RC4/MD5, System z9 and zSeries crypto hardware can be used for full (non-cached) handshakes, but not for the encryption & decryption of data.

RMI/IIOP
Remote Method Invocation using CORBA's communication protocol, IIOP. IIOP stands for Internet InterORB Protocol. Requests to WAS coming from Java clients and other WASs can use RMI/IIOP, which uses less CPU than HTTP requests.

SAX
Simple API for XML. A type of XML parsing. SAX parsing makes the contents of the XML document available to the application through a series of callbacks which occur as the parser scans and interprets the document. For example, the parser gives the application control when it encounters a "start element tag", so that subsequent processing decisions can be based on the tag elements. Callback processing is defined by user supplied handlers which are registered with the parser. During SAX parsing, the XML document is processed sequentially. Unlike DOM, SAX does not allow the program to revisit already parsed message segments unless they have been explicitly saved by application code. It is not possible to modify the original XML document. SAX parsing uses less CPU and less memory than DOM parsing.

Schema
Used in XML validation processing. A schema is used to describe the grammar that constrains an XML document. If, for example, an XML-format personnel file contains entries for many employees, each of which must have 1 social security number specified as ### - ## - ####, the Schema would contain a rule enforcing the occurrence of 1, and only 1, SSN per employee in the prescribed format. XML Schemas provide the ability to exercise a high degree of control over the contents of an XML document. Validation using a Schema does, however, generally require more CPU than validation using a DTD.

Servlet
Java code which can be run in WAS (on the server) in response to an HTTP request.

Session Bean
A type of EJB which represents work to be done on behalf of a particular caller. Session beans can be stateful (saving information from call to call) or stateless (saving no status from call to call).

SOAP
Simple Object Access Protocol.
1. SOAP is a W3C specification which provides a standard for using XML to exchange structured and typed information between peers in a decentralized, distributed environment.
2. SOAP is also the name of the WebSphere Web Services implementation supported in WAS 4.0 and WAS 5.0. The new Web Services support provided by WAS 5.0.2 performs significantly better (uses less CPU) than the original SOAP support. (and it also conforms to SOAP specifications).

SQLJ
Standard Query Language for Java. Another means (in addition to JDBC) to access data in DB2, or other relational data bases, from Java applications. In general, SQLJ access uses less CPU than JDBC, but cannot be dynamically created.

State, Stateful, Stateless
Many client interactions cannot be completed with a single request, requiring several requests to complete. For these multi-request interactions, it's often necessary to retain client and status information from request to request. This retained information is often referred to as "state". Session beans which retain state from request to request are called stateful session beans. Session beans which do not retain state are called stateless session beans, and they tend to consume less CPU than statefull session beans.

TCP Mode (or Client Mode) Connection
When a JMS connection is made in TCP mode, JMS uses TCP/IP to call the MQ Queue Manager rather than communicating over the Java Native Interface as it does in bindings mode. With TCP Mode Connections, the MQ Queue Manager does not have to be on the same server, or indeed the same platform.

Transacted Session
This option is used to group a series of messages into an atomic unit of work. All messages in the work unit either succeed or fail. The application server commits the session. If the application server detects an error, it may roll back the transaction. The message is not actually sent until the transaction is committed. The next transaction begins after a call to either commit or rollback.

TripleDES/SHA
The SSL cipher and hashing algorithm that provides the highest level of security, generally used by financial institutions and government agencies with high security requirements. TDES was developed by IBM and both the full handshake and encryption/decryption processing are supported by crypto hardware. Although some

processing is offloaded from the general CP engines, TDES/SHA still uses significantly more CPU than RC4/MD5.

Validation
See XML Validation

W3C
World Wide Web Consortium (w3c.org). The W3C is responsible for the creation and advancement of standard web-based technologies.

Web Container
The web container in WAS handles requests for servlets, JSPs, and other files that include server-side code.

Web Services
Web Services is the name given to communication that employs the SOAP standard for messaging. SOAP messages are XML documents containing certain required elements. They enable potential users of applications to find and invoke applications without the need to understand their implementation and underlying structure. Web services uses SAX parsing.

Web Transaction
This is a term we use to refer to any sequence of WAS activity that is repeatable and you want to use as the unit of work for projecting capacity requirements. In most cases, it is based on a business transaction. A web transaction can involve multiple interactions with WAS, any number of Java servlets/EJBs, multiple access to DB2, and multiple data connectors to back-end applications like CICS or IMS. Nothing inherent in WAS dictates what the scope of a transaction is. The important thing is to match your transaction rate with the scope of a web transaction that you choose., i.e. if your web transaction is long and involves a number of activities, the transaction rate would be lower than if you break up this sequence of activity into shorter web transactions.

XML
Extended Markup Language.

XML Attribute
A subcomponent of an XML element. Attributes are specified within element start tags or empty element tags. In the following example, **productId** is an attribute:

<productName **productId="123abc"**>Whistler Tea Kettle</productName>

XML Element
A subcomponent of an XML document. The example below represents an element called **productName**:

<productName productId="123abc">Whistler Tea Kettle</productName>

XML Validation
Validation is the process used by an XML parser to insure that the contents of an XML document conform to the rules in an associated DTD or Schema.

XSL Transformation

A type of XML processing used to create an XML document.