Tivoli Business Systems Manager
Operation and Planning and Control Release Notes
Version 1.5
# Contents

**Preface** .......................................................... v
  - Accessing Publications Online ........................................... v
  - Ordering Publications ..................................................... v
  - Providing Feedback about Publications ................................. v
  - Contacting Customer Support ........................................... vi

**Release Notes** .................................................. 1
  - Overview of Operation Planning and Control (OPC) ...................... 1
  - Integration With Tivoli Business Systems Manager ...................... 3
  - Tivoli OPC - Tivoli Business Systems Manager Setup Requirements ........ 4
  - Architecture .................................................................. 5
  - Event Flow .................................................................. 6
  - Discovery ..................................................................... 7
  - Configuration Tasks ....................................................... 7
  - Collection of Current Plans .............................................. 7
  - Mapping Applications and Operations in Tivoli Business Systems Manager ........ 7
  - Mapping Plans to Schedules and Jobs .................................. 10

**Appendix A. OPC Error Codes and Messages** ................. 13
Preface

Tivoli Business Systems Manager (TBSM) provides a robust, system management functionality. In support of the operational perspectives of that functionally, TBSM installs components on both the OS/390 and NT platforms.

Accessing Publications Online

The following sections describe how to access publications online, order publications, provide feedback about publications and contact customer support.

The Tivoli Customer Support Web site (http://www.tivoli.com/support/) offers a guide to support services (the Customer Support Handbook); frequently asked questions (FAQs); and technical information, including release notes, user’s guides, redbooks, and white papers. You can access Tivoli publications online at http://www.tivoli.com/support/documents/. The documentation for some products is available in PDF and HTML formats. Translated documents are also available for some products.

To access most of the documentation, you need an ID and a password. To obtain an ID for use on the support Web site, go to http://www.tivoli.com/support/getting/.

Resellers should refer to http://www.tivoli.com/support/smb/index.html for more information about obtaining Tivoli technical documentation and support.

Business Partners should refer to Ordering Publications for more information about obtaining Tivoli technical documentation.

Attention: The following note is an example of exceptional information. If your documentation requires similar, exceptional information, add it in the appropriate section (however, it is likely that your documentation does not require any additional notes or addenda). In all instances, remove this Attention element.

Note: For NetView OS/390 customers, additional support is also available on the NETVIEW CFORUM (Customer Forum) through the IBMLink system. This forum is monitored by NetView developers who answer questions and provide guidance. When a problem with the code is found, you are asked to open an official problem management record (PMR) to get resolution.

Ordering Publications

Order Tivoli publications online at http://www.tivoli.com/support/Prodman/html/pub_order.html or by calling one of the following telephone numbers:

- U.S. customers: (800) 879-2755
- Canadian customers: (800) 426-4968

Providing Feedback about Publications

We are very interested in hearing about your experience with Tivoli products and documentation, and we welcome your suggestions for improvements. If you have comments or suggestions about our products and documentation, contact us in one of the following ways:

- Send e-mail to pubs@tivoli.com.
Fill out our customer feedback survey at http://www.tivoli.com/support/survey/.

Contacting Customer Support

If you need support for this or any Tivoli product, contact Tivoli Customer Support in one of the following ways:


- Submit a PMR electronically through the IBMLink™ system. For information about IBMLink registration and access, refer to the IBM Web page at http://www.ibmlink.ibm.com.

- Send e-mail to support@tivoli.com.

- Customers in the U.S. can call 1-800-TIVOLI8 (1-800-848-6548).


When you contact Tivoli Customer Support, be prepared to provide the customer number for your company so that support personnel can assist you more readily.
Overview of Operation Planning and Control (OPC)

Tivoli Systems Operation Planning and Control (OPC) is a robust production workload scheduling system for the OS/390 platform. Through the creation of plans, long term and current (daily), applications, calendars, and special resources, production workloads can be scheduled to run automatically, taking into consideration dependencies, calendar issues (weekends, holidays), and special processing requirements. The Tivoli OPC environment consists of one system in the complex acting as a controlling system, which runs the Tivoli OPC controller, where automatic planning and control takes place, and all others acting as trackers. The tracker acts as the communication link between the local image where the tracker runs and the Tivoli OPC controller. Optionally, you can designate one of the tracker systems as a backup controller that can be switched to should problems arise on the primary controller system. Tivoli OPC incorporates information from several sources, including code in SMF and JES exits, to monitor a job’s progress through the system and report it back to the controller. Abend conditions, run times, and several other pieces of information are gathered and used for reporting status.

Tivoli OPC describes an application as a unit of production work. An application consists of a list of operations, which may refer to batch jobs, started tasks, or other functions. Tivoli OPC administrators maintain the definitions of these applications and operations, and can produce a variety of reports describing the configuration, state, and projected workflow of the applications. The Current Plan (also known as the Daily Plan) conveys the currently planned applications and operations, expected run times, and operation dependencies.
The following figure is an example of a Current (Daily) Plan.

<table>
<thead>
<tr>
<th>APPLICATION ID</th>
<th>APPLICATION OWNER</th>
<th>OPERATION TEXT</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAYBACK</td>
<td></td>
<td>Backups</td>
</tr>
<tr>
<td>PAYSTR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPUA_000 PAYBACK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPUA_000 PAYSTR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPUA_000 CPU001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPUA_000 CPU002</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PAYTAX</td>
<td></td>
<td>Payroll tax withholding</td>
</tr>
<tr>
<td>PAYTAX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPUA_005 TAX2WP</td>
<td></td>
<td>W2 Processing</td>
</tr>
<tr>
<td>CPUA_010 TX2MED</td>
<td></td>
<td>Medicare</td>
</tr>
<tr>
<td>CPUA_010 TAXSOC</td>
<td></td>
<td>Social Sec.</td>
</tr>
<tr>
<td>PAY4GLR</td>
<td></td>
<td>Payroll 401K contrib.</td>
</tr>
<tr>
<td>PAY4GLR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPUA_005 PWELFS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPUA_010 PWELFY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PAYDIEU</td>
<td></td>
<td>Payroll print jobs.</td>
</tr>
<tr>
<td>PAYDIEU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPUA_005 PAYCHECK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPUA_010 PAYBET</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPUA_015 PAYBONUS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMY_009</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tivoli Business Systems Manager (TBSM) represents each distinct application in the Current Plan as a Batch Schedule object within the object hierarchy. Each operation within an application is visible within the Batch Management Summary window.

Typical Tivoli OPC environments include thousands of applications and hundreds of thousands of operations. Managing this level of volume requires grouping the applications into sets by functions and management responsibilities within the organization. Using pattern-matching rules established by the administrator, the applications are grouped into sets that reflect both the physical location of the applications (i.e. where the applications run in the host environment) and the logical location or locations, which are determined by the business practices of your organization. These logical locations are reflected in a Line of Business view within TBSM.

The applications and operations in a Current Plan must be examined to determine how the applications are grouped into sets and Lines of Business views. Configuring both of these processes involves identifying the patterns in the data, which articulate the naming and organizational conventions of your business.

TBSM receives messages directly from the Tivoli OPC application regarding the status of jobs and the gathering of start and end times. The calculation for lateness has been eliminated.
Through two distinct points of integration, TBSM represents and monitors batch-related applications and operations controlled by Tivoli OPC and seamlessly integrates these objects with all other business objects monitored by TBSM.

The following sections of this document explain the various components of Tivoli Business Systems Manager’s interface with Tivoli OPC.

Integration With Tivoli Business Systems Manager

During this release of Tivoli Business Systems Manager (TBSM) there are two points of integration with Tivoli OPC. The first point applies a source for capturing updates made to the current plan, outside of the daily planning process, to a baseline snapshot created during the initial discovery process (via the processing of the Current Plan). This reduces the need for the continual running of any reports used to gather changes for the Current Plan.

The second integration point captures and processes alert conditions recognized by Tivoli OPC and forwards the information to TBSM. Included in this would be the meaningful ‘started’ and ‘ended’ messages that are Tivoli OPC specific (replacements for the generic IEF403I and IEF404I operating system messages), and the capturing of exception situations via WTO traps.

TBSM monitors Tivoli OPC for messages that cause several different error/alert conditions.

Console WTO’s are used to gather the following data:

**DURATION**

This error condition is met when an operation in the current plan is active for an unexpectedly long time. The alert threshold is based on the following formula:

\[
\text{estimated duration} \times \text{feedback limit} \div 100 = \text{length of run time that generates the alert}
\]

\[
\text{estimated duration} = \text{duration field from the operation definition or from workstation definition if non specified on operation}
\]

\[
\text{feedback limit} = \text{LIMFDBK value on the JITOPTS initialization statement}
\]

The alert for this error condition is also generated when an operation has been submitted, but has not started within 10 minutes.

**ERROROPER**

Error condition indicated by an operation in the current plan that has completed with an ‘ended-in-error’ status (e.g. abend, JCL error).

**LATEOPER**

An operation in the current plan reaches its latest start time without having a status of ‘started’, ‘completed’, or ‘deleted’ (latest start times are calculated during current plan creation).

**OPCERROR**

Indication of a TIVOLI OPC subtask or subsystem ending unexpectedly (applicable only to a tracker).

**QLIMEXCEED**

OPC/ESA subtask queue exceeds a threshold value (applicable only to a tracker).
RESCONT

Operation has been waiting on a resource for the length of time specified on the RESOPTS parm keyword value CONTENTIONTIME.

All of the messages produced by the preceding conditions include Job Name (Operation), Application Name, and input arrival Date/Time.

The capture of Start/Stop times for Tivoli OPC managed jobs (operations) are done through Tivoli OPC USER EXIT 7 (EQQUX007). This method uses Tivoli Business Systems Manager External Data Interface (EDI) to send data directly from Tivoli OPC to Tivoli Business Systems Manager via the Object Pump.

The combination of the WTO’s and Tivoli OPC USER EXIT 7 (with the EDI) enables Tivoli Business Systems Manager to monitor Tivoli OPC and its activities in a consistent and effective manner.

**Tivoli OPC - Tivoli Business Systems Manager Setup Requirements**

In the controller task (and in the backup controller task):

- ALERTSWTO(DURATION,ERROROPER,LATEOPER,OPCERROR,QLIMEXCEED,RESCONT)

In the tracker task:

- ALERTS WTO(OPCERROR,QLIMEXCEED)

**Note:** Alert statements may already exist in client environments where Tivoli OPC has been active prior to Tivoli Business Systems Manager implementation. In those cases, the alert initialization statements that do not currently utilize the WTO option would need to be modified.

Current initial statement:

- ALERTS GENALERT(OPCERROR,QLIMEXCEED)
- ALERTS MLOG(ERROROPER)

Modified initial statement:

- ALERTS GENALERT(OPCERROR)
- ALERTS MLOG(ERROROPER)
- ALERTSWTO(DURATION,ERROROPER,LATEOPER,RESCONT,OPCERROR,QLIMEXCEED)

When the setup is configured in this manner and any of the alert conditions are met, a WTO is written to the console, which can be captured via console traps in the TBSM Source/390 Object Pump. The traps would be set for the seven messages. See [OPC Error Codes and Messages](#) on page 13 for information about the OPC STC. Specific Job Name information would need to be extracted from the job specific messages (DURATION, ERROROPER, LATEOPER, RESCONT) once they are sent to N/T enabling the alerts to be forwarded to the monitored object. OPCERROR and QLIMEXCEED are OPC/ESA task related error conditions and could be forwarded directly to the STC object for Tivoli OPC (controller or tracker).
Architecture

The aforementioned integration points are accomplished through the use of the EDI and Tivoli OPC USER EXIT 7, EQQUX007, as well as console traps for several new exception messages. EQQUX007 is the Operation Status Change exit that acquires control at two critical points. First, whenever an operation in the current plan changes status and secondly when an operation is added to the current plan via a function other than daily planning. The valid status changes are the following:

- Arrived at workstation
- Completed
- Ended with errors
- Interrupted
- Ready for processing
- Started
- Waiting

The Tivoli Business Systems Manager (TBSM) assembler code, which resides in the USER EXIT, gets control at these two points and appropriate messages are created and sent to TBSM via the EDI. In the case of a status change, if the status is 'started' or 'completed', a message containing critical Tivoli OPC information is sent. The critical pieces of information included are application name and input arrival time, which allows precise identification of an occurrence of an operation in TBSM. Input arrival time for an operation is used as part of a key (Operation Name is the other) by Tivoli OPC to differentiate one occurrence of an operation from another. In the case of current plan additions, a message containing information about the operation being added is sent (via the EDI) and the operation can be added to the TBSM representation of the current plan therefore keeping it current.

The following illustrates the Tivoli OPC Architecture:
Event Flow

When any of the alert conditions are met, a WTO is written to the console, which can be captured via console traps in the TBSM Source/390 Object Pump. The traps would be set for seven messages and the corresponding OPC STC. For more information on these messages, see “OPC Error Codes and Messages” on page 13. Specific job name information would need to be extracted from the job specific messages (DURATION, ERROROPER, LATEOPER, RESCONT) once they are sent to NT, so that alert icons could be forwarded to the monitored object. OPCERROR and QLIMEXCEED are OPC task related error conditions and could be forwarded directly to the STC object for Tivoli OPC (controller or tracker).

The following illustration depicts the OS/390 Processing Environment event flow:
Discovery

The initial discovery is done via the Tivoli OPC Current Plan report. This is the process being used today. However, for the future, instead of using a report to gather this information, we implement a Program Interface (PI) directly to Tivoli OPC via routine EQQYCAIN. This enables us to attain information like PREDEESSOR/SUCCESSOR relationships by relating data from the daily/long term plans and the entire database. In order to integrate the OPC/ESA applications and operations into Tivoli Business Systems Manager, the TBSM Administrator must accomplish several tasks during the discovery session. These tasks are divided into the following categories: configuration and operation.

Configuration Tasks

The configuration tasks are concerned with specifying how the Tivoli OPC Current Plans are to be collected on a daily basis, which plans are used as input to Tivoli Business Systems Manager, and how the applications and operations within the plans are to be organized and monitored. These tasks are done before any monitoring can be implemented, and are then revisited only when changes are to be made in the set of plans to input or in how the data is to be organized.

Collection of Current Plans

Collecting the current plans from Tivoli OPC must be done at least on a daily basis or whenever there are significant intra-day changes to a plan. The collection process can be initiated either from an automation task on the OS/390 image(s) where Tivoli OPC is running, which copies the plan to the Tivoli Business Systems Manager central database server, or by scheduled tasks. Either way, the plans end up as files that reside on the central database server.

Each Current Plan you load should be copied into the DATA\OPC subdirectory of the installation directory for Tivoli Business Systems Manager on the central database server. Each plan must have a unique name, and this name should correspond to the system where the plan was generated. For example, if a particular Tivoli OPC plan that runs in a Complex known as “Northern” contains payroll applications, an appropriate name for the file would be “NORTHERN PAYROLL.”

Mapping Applications and Operations in Tivoli Business Systems Manager

The applications and operations in a Current Plan must be examined to determine how the applications are to be grouped into sets and Lines of Business. You must determine which operations represent key batch jobs that are to be monitored directly for start/stop events (all operations are monitored for abends). Configuring both of these processes involves identifying the patterns in the data that articulate the naming and organizational conventions of the business.

As mentioned before, applications in Tivoli OPC are represented as objects of the Batch Schedule class within Tivoli Business Systems Manager. These schedules must be placed in the physical hierarchy in Batch Schedule Set objects under the Complex in which the Tivoli OPC applications run in order for proper monitoring to be established in the Complex.
following illustration depicts the physical hierarchy.

In order to create this hierarchy automatically as the Current Plan is loaded, the TBSM Administrator must define both the hierarchy to receive the applications and the data to look for in the application definitions of the Current Plan, which identifies their locations.

The definition of the hierarchy is accomplished by creating and applying a dynamic object hierarchy definition file. This is a plain text file that uses a set of macros to define hierarchies by specifying the beginning of a distinct hierarchy (BEGIN_DYNA_OBJ_PATH), each distinct level of the hierarchy (DYNA_OBJ_PATH), and the end of the hierarchy (END_DYNA_OBJ_PATH). The following Dynamic Object Hierarchy Definition file example depicts a dynamic object hierarchy definition file.

```plaintext
include (BusinessObject.sqi)
BEGIN_DYNA_OBJ_PATH(NORTHERN PAYROLL, Group all payroll applications)
    DYNA_OBJ_PATH (ENT, ABC Insurance Corporation)
        DYNA_OBJ_PATH (COMP, Northern)
            DYNA_OBJ_PATH (BCYS, PAYROLL, OPC applications owned by PAYMSTR)
END_DYNA_OBJ_PATH (NORTHERN PAYROLL)
BEGIN_DYNA_OBJ_PATH (Batch Print Jobs, Groups all batch print jobs)
    DYNA_OBJ_PATH (LOB, System Support)
        DYNA_OBJ_PATH (LOB, Printers)
        DYNA_OBJ_PATH (LOB, Batch Jobs)
END_DYNA_OBJ_PATH (Batch Print Jobs)
```

The general form of the BEGIN_DYNA_OBJ_PATH macro is the following:

```
BEGIN_DYNA_OBJ_PATH(name, description)
```

The name uniquely identifies the path, and the description is strictly for documentation. In the previous dynamic object hierarchy definition file example the first path is named “NORTHERN PAYROLL” and its description is “Group all payroll applications.” The second path is named “Batch Print Jobs” and its description is “Group all batch print jobs”.

The general form of the DYNA_OBJ_PATH macro is the following:

```
DYNA_OBJ_PATH(classid, name, description)
```
The classid is a valid identifier for an object class within the Tivoli Business Systems Manager data model. A list of these classes for aggregation of schedules is provided in the following table. The name parameter is the name of the object that is created at this point in the hierarchy. Every subsequent line that has a DYNA_OBJ_PATH macro defines an object that is a child of the object defined on the previous line. The description parameter is used to populate the object description field.

**Note:** For the physical aggregation of schedules, the hierarchy must be BUSC, ENT, COMP, and BCYS. Also, please note that in the previous example the indentation is only for visual clarity.

The general form of the END_DYNA_OBJ_PATH macro is the following:

```
END_DYNA_OBJ_PATH(name)
```

This macro is called to finish the definition of the dynamic object path. The name parameter should match that specified by the preceding BEGIN_DYNA_OBJ_PATH.

The sample batch hierarchy illustration previously mentioned shows a dynamic object path that defines a Line of Business hierarchy. Note that for Lines of Business, the top level object is always the class “LOBC”, and each subsequent level of the hierarchy is the class “LOB”, which is the classid for the Line of Business class.

The following table describes the Batch Schedule Class Hierarchy.

<table>
<thead>
<tr>
<th>CLASS ID</th>
<th>CLASS NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUSC</td>
<td>BusinessObjectContainer</td>
<td>The overall container for all Enterprises.</td>
</tr>
<tr>
<td>ENT</td>
<td>Enterprise</td>
<td>The aggregation of all of an organization’s data center complexes. This is the top-level customer object in a Tivoli Business Systems Manager object model.</td>
</tr>
<tr>
<td>COMP</td>
<td>Complex</td>
<td>The aggregate set of defined OS/390 hosts and resources that span hosts (such as OPC) within an organization.</td>
</tr>
<tr>
<td>BCYS</td>
<td>BatchScheduleSet</td>
<td>Aggregate object containing Batch Schedules that have been grouped by the OPC dynamic load or manually by the customer.</td>
</tr>
<tr>
<td>BTCY</td>
<td>BatchSchedule</td>
<td>A collection of batch jobs managed by a scheduling system, such as Tivoli OPC.</td>
</tr>
</tbody>
</table>

Once the dynamic object hierarchies have been defined, the mapping of the Tivoli OPC applications to these hierarchies has to be defined. In this process the administrator must identify the patterns in the OPC Current Plans that are used to aggregate the applications. Each of these patterns is called a Pattern, “OPC Application Pattern,” and consists of the following fields:

**OPC_PATH_NAME**
Name of the dynamic object path to aggregate matching applications to.

**OPC_PLAN_PATTERN**
Wildcard expression matched to the names of the defining daily plan file.

**OPC_APP_PATTERN**
Wildcard expression matched to the application ids.
**OPC_OWNER_PATTERN**

Wildcard expression matched to the application owner fields.

**OPC_DESC_PATTERN**

Wildcard expression matched to the application description fields.

The patterns are defined similarly to the dynamic object paths, via the OPC_APP_PATTERN macro. In fact, the pattern entries can be in the same file as the dynamic object path definitions.

The wildcard expressions support the following notation:

- **%** Multiple character wildcard. Matches any number of characters (including zero).
- **_** Single character wildcard.
- **[..]** Any single character that matches the characters between the brackets. Also supports ranges of characters, for example "[0-9]" specifies any digit between 0 and 9. Can also be used to specify any character not matching the set or range if the first character after the "[" is a "^" (caret). Use "[[%]" and "[_]" to match a single literal "%" or "_" character.

You can see sample OPC_APP_PATTERN macros in the following example of a Tivoli OPC Application Pattern.

**OPC_APP_PATTERN** (NORTHERN PAYROLL, NORTHERN.PAYROLL)

**OPC_APP_PATTERN** (Batch Print Jobs, %, %PRINT%, %, %PRINT%)

The general form of the **OPC_APP_PATTERN** macro is the following:

**OPC_APP_PATTERN**(PATH_NAME, PLAN_PAT, APP_PAT, OWNER_PAT, DESC_PAT)

Once the dynamic object paths and OPC Application Patterns have been defined, they are compiled and loaded into the central database using the command line utilities: **clsql** and **ISQL**.

**Mapping Plans to Schedules and Jobs**

Once the plan files reside on the central database server, they are loaded into the database using the **OPCParseDailyPlan** utility. Typically, this is the last step of the task that copies the plan files from OS/390 to the central server, but this can be done separately.

The command line for this utility is the following:

**OPCParseDailyPlan** options... daily-plan-file

The options are the following:

- **-language**
  Optional parameter that specifies an alternate language to use for keyword parsing. The default is EN_US (US English). The other currently supported languages are:
  - DEU (German)
  - EN_UK (UK English)
  - ES (Spanish)

- **-np**
  Skip the parsing of the data. This can be done if the daily plan file was parsed by a previous run of the utility, but there was a problem with the load to the database.
-nl  No load of the parsed data is performed. This can be done for diagnostic purposes to examine the output of the parsing phase prior to loading.

-nm  No mapping of the loaded data to objects is performed. As with -nl, this is mostly a diagnostic option that allows examination of the loaded data prior to mapping it to schedules and jobs within the object repository.

-v   Run in verbose mode. Normally, only errors are reported on the command line.

-E   Use a trusted database connection. This is typically used when running the utility as a scheduled task on the central database server in order to avoid having user names and passwords displayed in ways that could inadvertently be exposed to unauthorized users.

-User  User name for the database login.

-Password  Password for the database login.

-Sserver  The database server.

The daily-plan-file is the name of file containing an Tivoli OPC Current Plan report to be processed.

After the plan has been successfully loaded, it is processed for mapping the applications and operations to the schedules and jobs within the object repository. The applications are monitored as schedules and Lines of Business following the standard management practices of Tivoli Business Systems Manager.
OPC Error Codes and Messages

This section provides information on the message layout and provides some examples (using real data. It is important to note that the following examples are standard “out of the box” message formats. Clients do have the ability to modify the message layout, which is necessary information that is needed prior to implementation. The following messages apply to the Tivoli OPC error codes and accompanying messages.

**DURATION:**

**EQQE038I**

LONG DURATION FOR JOB JOBNAME(JNUM), IN APPLICATION APPL, WORK STATION = WSID, IA = ARRTIME

Example:

- **EQQE038I** LONG DURATION FOR JOB TESTBRL2(JOB04039), 149 N APPLICATION SAMAPP, WORK STATION = CPU2, I IA = 0005091240

**EQQE039I**

LONG TIME ON INPUT QUEUE FOR JOB JOBNAME(JNUM), APPL = APPLID, WORKSTATION = WSID, IA = ARRTIME

Example:

- **EQQE039I** LONG TIME ON INPUT QUEUE FOR JOB TESTBRL1(JOB07220), APPL = BRIAN3, WORKSTATION = CPU2, IA = 0005301455

**ERROROPER:**

**EQQE036I**

JOB JOBNAME(JNUM) ENDED IN ERROR EC. PRTY=PRI, APPL = APPL, WORK STATION = WSID, IA = IA

Examples:

- **EQQE036I** JOB XPC01K01(JOB07839) ENDED IN ERROR S522. PRTY=5, APPL = OPC01K01P, WORK STATION = CPU1, IA = 9909070600
- **EQQE036I** JOB XPCPLCK (JOB06996) ENDED IN ERROR JCL. PRTY=5, APPL = OPCPLK01P, WORK STATION = CPU1, IA = 990907060

**LATEOPER:**

**EQQE037I**

JOB JOBNAME(JNUM), IN APPLICATION APPL, IS LATE, WORK STATION = WSID, IA = ARRTIME
Example:
- EQQE037I JOB DKDREI91( ), IN APPLICATION DKDREI91ZPA, IS LATE, WORK STATION = CPEA, IA = 9912191800

OPCERROR:
**EQQZ045W**
OPC/ESA SUBTASK SUBTASK ENDED UNEXPECTEDLY
Examples:
- **EQQZ045W** OPC SUBTASK EVENT MANAGER ENDED UNEXPECTEDLY C OPCC
- **EQQZ045W** OPC SUBTASK EVENT MANAGER ENDED UNEXPECTEDLY

QLIMEXCEED:
**EQQZ106W**
*PERCENT % OF QUEUE QNAME IN USE PERCENT % OF QUEUE QNAME IN USE*
Example:
- **EQQZ106W** 100 % OF QUEUE WTRQ IN USE

RESCONT:
**EQQQ515W**
OPERATION *ADID OPID, JOB N IA IADATE IATIME* HAS WAITED
Examples:
- **EQQQ515W** OPERATION DKDREI56ZPA CPEA_005, DKDREI56 IA 991220 1800 HAS WA RESOURCE EREORG FOR 0 MINUTES.
- **EQQQ515W** OPERATION BED99000ZPAZZ050 CPEA_025, BEDICP02 IA 991220 1800 H RESOURCE ETAPE FOR 14 MINUTES.