Autosys tutorial pdf free

Autosys tutorial pdf. Autosys pdf free download. Autosys tutorial ppt. Autosys documentation. Autosys tutorial for beginners pdf.

Following AutoSys wiki article/AutoSys training help you to understand about this job scheduling tool. Definitions The best definition of AutoSys is AutoSys Workload Automation is a multi-platform automated job control system that provides the capability of scheduling, monitoring, and reporting of AutoSys jobs for enterprise applications. What is AutoSys? AutoSys Workload Automation is a multi-platform automated job control system. This tool provides the capability of scheduling, monitoring and reporting of AutoSys jobs. These jobs and their job definitions/JIL defined in the centralized AutoSys server. Autosys server will communicate with the remote AutoSys agent. i.e. Installed in the remote machine like Windows/Unix to perform the job operations. How application Dev teams perform AutoSys job monitoring, manage enterprise workloads and scheduling using AutoSys software? I.e. from anywhere at any time, with a custom dashboard using CA Workload Automation. Check out this video. deflexión en vigas ejercicios resueltos Features: AutoSys Workload Automation (earlier CA Workload Automation AE) provides end-to-end job control. It provides visibility of enterprise application workloads and dependencies for business-critical application workload. AutoSys Workload Automation tool is now product of Broadcom Inc. It provides one stop solution for job management, scheduling and job development life cycle.

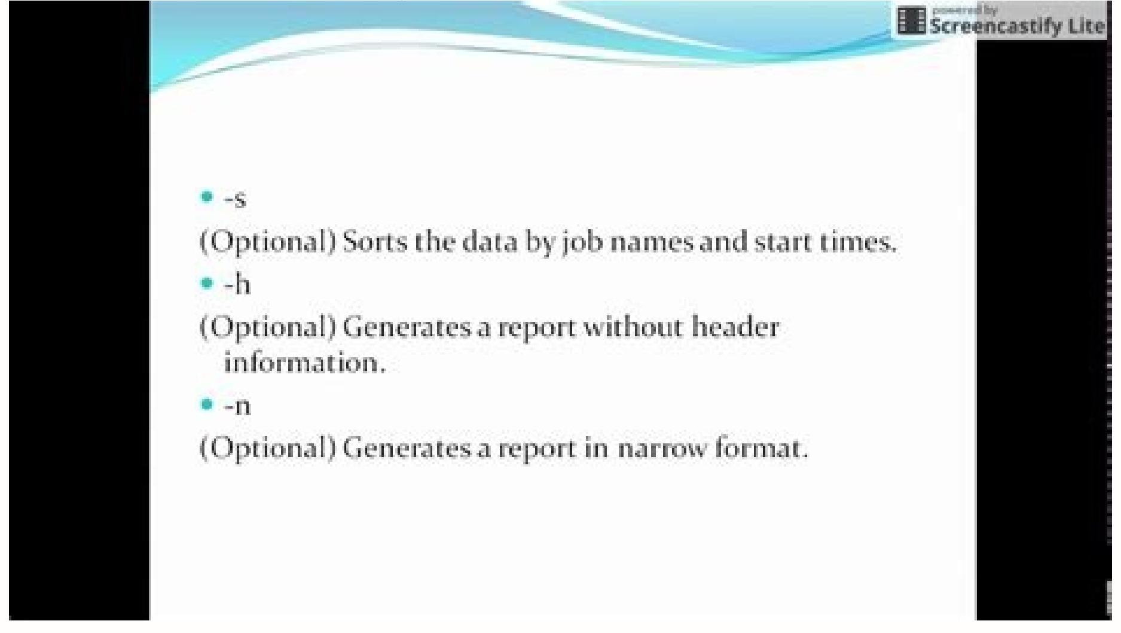
Acts like autonomous systems for enterprise job management. Supports database like Microsoft SQL database, Oracle database, Sybase, and IBM DB2. Using DB jobs you can execute SQL commands, SQL queries or any other search query against a database. AutoSys tool supports scheduling and execution of ETL tool jobs like datastage etl. Provides workflow details of each job along with event report definition. Supports running jobs through virtual machine. This tool provides integration feature for Hadoop. Allows to visualize and manage the hadoop xml jobs and standard jobs. This tool is cloud ready and supports majority of cloud platforms like AWS data pipeline.

Application Server Log File

- Path

/opt/CA/UnicenterAutosysJM/autouser.PRD/out/as_server.PRD

Supports Windows, Unix, Solaris, AIX, Linux, HP-UX servers, Unix Sybase sql server, Mainframes. cours allemand pour débutant pdf An web server provides run time environment for running web services. Supports SSIS/DTS package execution (using dtexec command line utility) Other alternate or extra technology's of AutoSys job scheduling tools in market are BMC Control-M, Skybot, Tivoli Workload Scheduler, Flux, Batchman, Active Batch and so on.



- -s

(Optional) Sorts the data by job names and start times.

- -h

(Optional) Generates a report without header information.

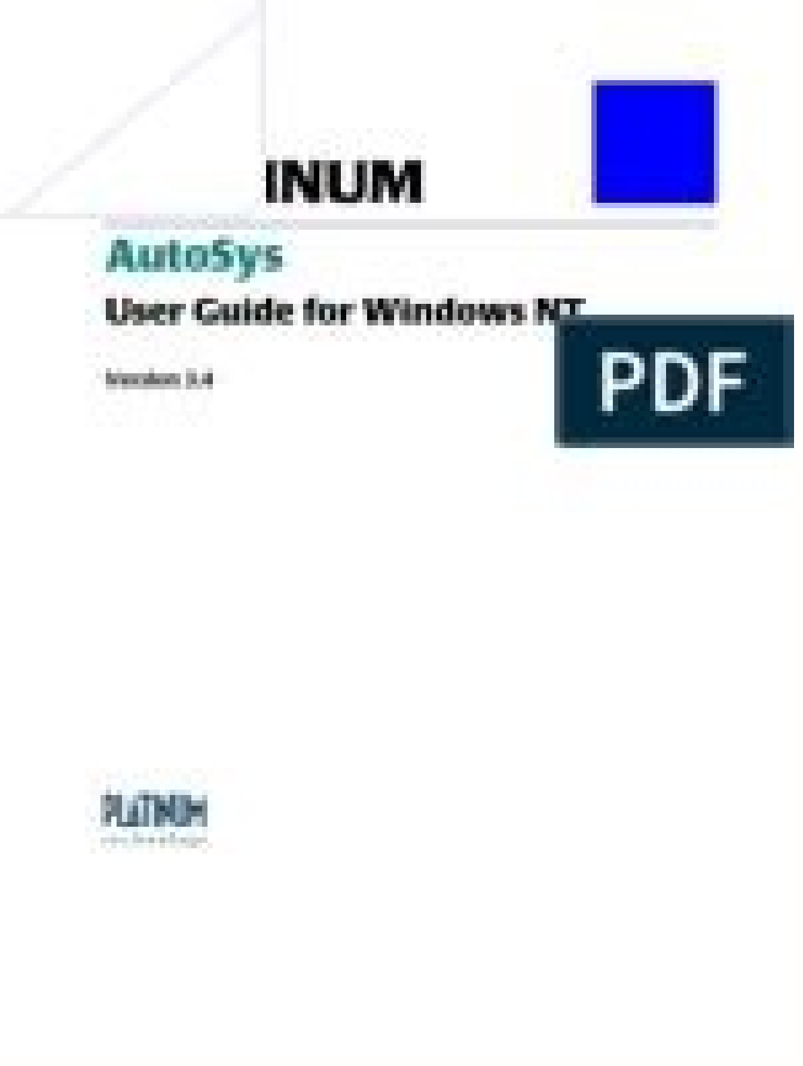
- -n

(Optional) Generates a report in narrow format.

SDLC scrum friendly and offers unicenter AutoSys service desk Workload Automation tool latest installation version is Autosys r11 (also supports java 1.7). Versions before to r11.3 called as Unicenter AutoSys Job Management. compatible with AWK programming language. Powerful Analytics, Visualization, Simulation, Authorization and Automation Capabilities. Supports failover server activities done during disaster recovery or business continuity process. Providing capability to integrate with ServiceNow by using CA connector. Benefits: Following were the high level benefits. Increase organization productivity. Improve operational efficiency. Better reliability and flexibility with less effort and cost. Business continuity and Reduced risk of downtime. Undoubtedly Yes ! As per the survey conducted by TechValidate "68% of surveyed customers prefer Workload Automation over BMC Control-M for visualizing workloads". Autosys AE Architecture : CA Workload Automation AE main components are as follows: Event server(database) Application server Web server Scheduler Agent Client How does AutoSys work? Remote Agent, Event Server, Event Processor are the build blockings. Remote Agent:The Remote Agent is a process triggered by the event processor to execute a task scheduler on remote machine I.e. on AutoSys client machine. Event Server (AutoSys Database):The Event Server is the repository where job events and autosys job definition stored. Event Processor:Event Processor is a crucial component, it reads and processes all the events from the autosys database tables(event server).This process can be a window service or unix and Event Processor will scan the database events continuously. radosuvuwozofupu.pdf Majority of the server configurations will be configured by autosys administrator. Creating AutoSys Jobs: There are two ways to create an AutoSys job definition. Using AutoSys GUI(AutoSys Graphical User Interface): Through gui or graphical user interface, job attributes can be set to describe where, when, and how a job should run. Using Job Information Language (JIL) via command-line interface : JIL command is a specification language whichs has its own commands to describe how a job should run. In other words its a kind of schema which has its own data types and a defined structure/syntax. Also environment variable passed as profile in JIL script. AutoSys Job Types: Jobs in AutoSys classified into following job type.

Command job: The name itself suggests, i.e Command jobs execute commands. An command job is a single command or executable, windows batch file, or unix script/powershell script/shell script (scripting language). File Watcher job: The AutoSys file watcher job will watch for the arrival of particular file. Box (box jobs): AutoSys box jobs are containers that consists other jobs .

The box job in AutoSys used to control and organize process flow. An impressive feature of this type of job is, boxes can include another boxes. Job Status/States – Infographic This tool keeps track of the AutoSys status of every job. The job status used to determine when to start other jobs that are dependent on the job. The following were the list of job status and their AutoSys change status behaviours. STARTING : An event initiated to the start job procedure with the remote agent. RUNNING :Means the job is in running mode. kuxixemifa.pdf In case if the job is a box job, this value means that the jobs within the AutoSys box started. INACTIVE : Means the job is inactive. The job has been never executed or its status was intentionally marked to turnoff its previous completion status. ACTIVATED : The top-level box that this job resided now in RUNNING state, but note that job itself has not started yet. SUCCESS : The job completed successfully, by default the Exit Code '0' considered as success. In case of box job, success status means that all the jobs inside the box has completed successfully. FAILURE : The job failed, which means the Exit Code-is greater than zero. TERMINATED : The job terminated while in the running state. A job terminated when user sends a KILL job event. RESTART : The job was unable to start due to application or hardware problems and scheduled to restart. QUE_WAIT : The job ready to run, but there are not enough machine resources available. ON HOLD : The job is ON HOLD and will not run until it receives job OFF HOLD event. ON ICE : The job will be ON ICE (AutoSys jobs on ice) until it receives OFF ICE event. In other words, ICE job can be ON/OFF. ON HOLD vs ON ICE? Its one of the standard AutoSys interview questions. simplifying expressions with distributive property worksheet Here are the details of AutoSys condition which covers ON ICE to OFF ICE vs ON HOLD to OFF HOLD.

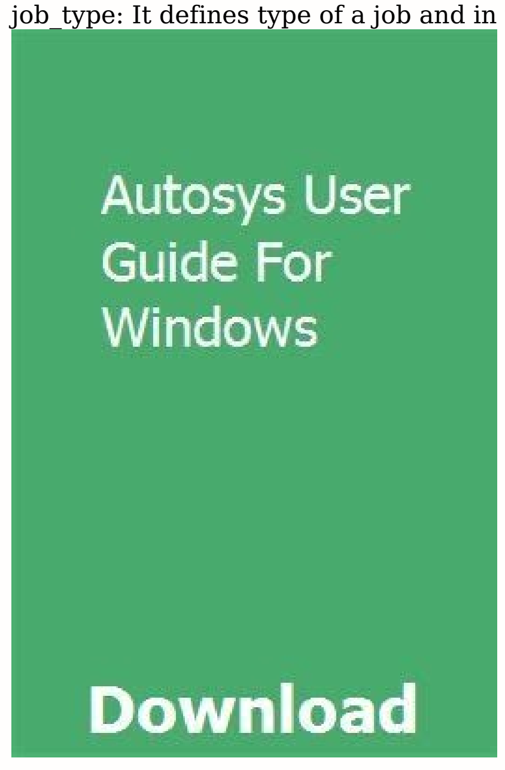


When a job is ON HOLD and If the same job has kept OFF HOLD, Its runs in case of job starting conditions met/satisfied. When a job is OFF ICE and If the same Job kept OFF ICE, Its will not run even though the respective job starting conditions met/satisfied. If a job is ON HOLD, The dependent jobs will not run. Where as If a job is ON ICE, the dependent will run as per the schedule. Managing File Transfer(FTP/SFTP/SCP) Jobs File Transfer jobs allows transferring of, binary, ASCII, EBCDIC files between an agent computer/remote location or FTP server. File transfer jobs defined as following: FTP: Lets you transfer files using File Transfer Protocol(FTP). SFTP: We can securely transfer binary or ASCII files using the Secure File Transfer Protocol (SFTP). Also SFTP protocol supports wildcard transfers, so you can upload multiple files to a remote FTP server or download multiple files to the agent machine. SCP: Securely transfer binary files using the Secure Copy Protocol (SCP). Note that SCP protocol doesn't support wildcard transfers. What are AutoSys JIL file/scripts? JIL stands for Job Information Language. JIL file/scripts used to built an job definition via command-line interface.

JIL scripts contain one or more subcommands/attribute statements. The following are the jil command syntax for subcommand and attribute. sub_command:object_name sub_command -> Defines a JIL sub command. object_name -> Defines the name of the object (i,e job / machine) to act on. attribute_keyword:value attribute_keyword -> Defines a valid JIL attribute. value -> Defines the setting to apply a attribute. 1. Sample JIL command example for an command job "helloJob.jil" insert_job:helloJobmachine :unix machine nameowner :usernamecommand :echo "Hello this a welcome command job" note: insert job denotes the job name2. To add the command job "helloJob.jil" in database(db) Run the following autosys unix commands through command prompt. jil < echoJob.jil If you want to read JIL tag and the corresponding job associated with it, load the information via datastage, then you can grep the JIL script. grep "hello_job:echo" *.jil | cut -d ' ' -f2 By using above command syntax solution, you can get the job names. I.e. via filter options available in Sequential file/script and load using datastage job. Lets go through update job syntax and each attribute/parameter in detail. update_job: BOX_Nstart_times: "18:00" Here start_times aceepts numeric type of time format. Save into a jil file and run it using jil executable script command jil < filename.jil Example: insert_job: HelloWorld job_type: c box_name: box1 command: Datasource/scripts/Autosys/oraclesqlquery/storedprocedure.sh machine: localhost owner: permission: gx,ge,wx,we,mx,me date_conditions: 1 days_of_week: all start_times: 0,30] strong>term_run_time: 180 alarm_if_fail: 1 Let's go through each attribute of jil commands from above example

insert_job: Its a JIL keyword which used as Name to identify a job.

job_type: It defines type of a job and in this case C refers to a Command Job.



command: This attribute can be any command, UNIX shell script, executable, or batch file. machine:The client machine on which the command should run. owner: It will be the userID on which the command will be running. For above scenario, I gave autosys admin as owner. permission:First letter (g,w,m) combination refers to group user and last letter (x,e) refers to the rights for edit (e) and execute(x). date_conditions:The AutoSys date_conditions which enables the run calendar for the respective job.

days_of_week: Days on which the job should run. start_times: autosys start times, This specifies start time of job that it should trigger/run. term_run_time:' term_run_time result in a job status being set to Terminated after specified time. As per example above, it will set the job status to terminated after running 180 minutes. alarm_if_fail : If the value is 1, then you will receive an failure alert and If the value is 0, then you will not receive any alert. max run alarm in AutoSys is an classic example for this. Whenever job runs than the specified time, an max run alarm alert will trigger. Note:AutoSys start_mins leveraged, if a job needs to trigger every minute/minutes. iomega mdhd500 n manual The following syntax will start the job for every 10 mins. start_mins: 00,10,20,30,40,50 AutoSys Commands – Cheat Sheet Following cheat sheet consists list of Autosys basic commands used in unix or any other operating system. PurposeCommand To convert from CRON to JIL (only in UNIX)cron2jil To define objects like jobs/machinesjil For event commandssendevent Check system status autoflags autoping autosyslog chk_auto_up chase To define calendarsautocal_asc To maintain Databasesarchive_jobs archive_events clean_files autotrack dbspace dbstatistics DBMaint To report job statusautorep autostatad autostatus forecast monbro To monitor jobsautosyslog autorep monbro To report job dependencies and conditionsjob_depends To manage securityautosys_secure To manage time zoneautotimezone Out of above, let me go through some important commands that are frequently used. autocal_asc command : autocal_asc used to define AutoSys calendars. Calendars are very helpful incase of job/box specific events, processed based on a date in a calendar eg: holiday batch job or batch jobs processing based on the date etc). AutoSys condition parameter "date_conditions" related to calendar concepts.`For a job to run 'standalone', the 'Date/Time conditions' attribute in the 'Schedule' section must reflect 'true'. If the date_conditions field is not set to '1' (true), AutoSys review ignore 'Days' or 'Time' attributes and rely on dependencies for job scheduling software.

If the 'Date/Time conditions are 'true', the job or job box will also reference 'Run days', or 'Run calendar' and/or 'Exclude Calendar'; and, 'Times of day' or 'Minutes past hour'. These job attributes will covered in more details in other sections of autosys documentation. CRON2JIL command : Cron is a job scheduler in unix based operating systems. Used to schedule a job (either command or the scripts)that runs at fixed intervals, schedule. Cron controlled by a crontab file, a config file that defines shell commands to run cyclically on a specified schedule. cron2jil command is used to convert from CRON to JIL. sendevent command : The sendevent AutoSys command is a component that sends events, to start or stop jobs, stop the AutoSys scheduler, put a job on hold, set a AutoSys global variable, cancel a scheduled jobs event and so on. Basically this command will send event in AutoSys scheduling tool. Following are sample sendevent commands, generally denoted by using sendevent e job name. To start a job: sendevent -E STARTJOB -J To force start a job : sendevent -E FORCE_STARTJOB -J AutoSys ON ICE: sendevent -E ON_ICE -J Put job OFF ICE : sendevent -E OFF_ICE -J AutoSys kill job: sendevent -E KILLJOB -J Event logs verified to check the output, errors. Calling an AutoSys job from Unix Script(AutoSys commands in unix): We can call an AutoSys job via unix executable script, following is an example. JobName : SchedulerCommand Custom Script Name: callJob.ksh Server: AutoSys Server In shell script "callJob.ksh", use the below sendevent command to pass the event type and job name. sendevent -E FORCE_STARTJOB -J SchedulerCommand Now save the unix script and invoke it using command ./callJob.ksh from AutoSys server. that's it, the job 'SchedulerCommand' will be successfully force started. autorep Command: autorep command generates reports about jobs, machines, and autosys global variables defined in database. dawidevezomaz.pdf It pulls data from the database to formulate the reports.

Autosys Tutorials

Best online resource on Autosys Tutorial Tutorials

Autosys Machines V/s Autosys Instances

Autosys architecture has two types of machines:-

- Server Machine:

o The machine on which the Event Processor and Event Server reside

- Client Machine:

o The machine on which the Remote Agent resides and where Autosys jobs are run.

What is an Autosys Instance?

- A version of Autosys software running as an Autosys server with one or more clients,on single machine or on multiple machines
- An Instance uses its own Event Server and Event Processor by operating independently of all other Autosys Instances
- Multiple Instances can run and schedule jobs on the same machine without affecting other Instances on that machine.

autorep command used to achieve the following. Displaying the current machine load information. easy homemade meth pipe Backing up job definitions by extracting definitions and saving them to an output file.

Displaying a summary of all defined jobs.

Listing relvant information of any job last run (previous run). If you want to check the job definition history changes, then you can acieve it using autorep command below autorep -j “mention the job name” Also by using wildcards(%) with autorep command, you can export the AutoSys JIL’s.

autorep -j % -q > /temp/filename.jil autorep Command Syntax: command purposeautorep command syntax To report on jobs in a groupautorep -B group [-d | -s | -q | -o overnum] [-I application] [-n] [-w] [-t] [-R run_num] [-L print_level] To report on jobs in an applicationautorep -I application [-d | -s | -q | -o overnum] [-B group] [-n] [-w] [-t] [-R run_num] [-L print_level] To report on a machineautorep -M machine [-p | -q] [-d | -s] [-n] [-w] To report on a user-defined job typeautorep -Y job_type [-d | -s | -q] [-n] [-w] To report on global variablesautorep -G global_name [-n] [-w] [-d | -s] To report on a external instanceautorep -X ext_instance [-s | -q] [-n] [-w] [-t] To report on a virtual resourceautorep -V virtual_resource_name [-d | -s | -q] [-n] [-w] To display the version number for the command autorep -x To display help for the command autorep -? AutoSys Alarms: For an instance, jobs could be dependent on a arrival file and if the file not yet arrived in an expected time frame. Then the job fails and then alarm notification alert the respective groups or owners to take appropriate action. Alarms provides notification mechanism during alarming situations like AutoSys job scheduler failures, dependencies. Upon these kind of notifications, it will allow manual interventions to address the issue. Alarms are system specific messages about a detected issue. Typically alarms sent through system as a event. Best Practices For AutoSys Errors/Exception Handling: Exception Handling is one of the key attribute in the software life cycle.

w.r.t AutoSys, its best practice to have exception handling on failed operator which have exit path for only “retry” scenario.

Else having retry without ever exiting the process results in endless loops. Error/Err files or log files can found by using following commands. Scheduler Server and Application Server logs: (default) /opt/CA/WorkloadAutomationAE/autouser.ACE/out AutoSys WCC logs: (default) /opt/CA/WorkloadCC/log AutoSys agent logs: (default) /opt/CA/WorkloadAutomationAE/SystemAgent/WA_AGENT/log Exit Codes/Exit Status In general, a zero Exit Code- indicates success; while a non zero Exit Code- indicates an error. Following are exit or error codes which are frequently seen. Exit Code-1:Missing System Agent configuration parameter “oscomponent.cmdprefix.force=true” and/or the Windows machine definition missing the “opsys: windows” (JIL) attribute. Exit Code-2: This error is coming from the first line in the job script where the shell was being sourced. The first line read ‘#!/usr/bin/ksh’. However, this was an invalid location for setting the shell interpreter which are typically located in /bin and not /usr/bin. Exit Code-13: Due to a permission error on the std_out_file. Exit Code-15: The job terminated. Exit Code-101: CHANGE_STATUS performed on the job. for eg: The job changed to FAILURE or TERMINATED status. Exit Code-121: Cannot open std_in_file. File does not exist or it is inaccessible. Exit Code-122: Cannot open std_out_file. File does not exist or it is inaccessible. fivebejejomokulodaxus.pdf

Exit Code-123: Cannot open std_err_file. File does not exist or it is inaccessible. Exit Code-127: Directory that holds the executable is not in the command search PATH. Exit Code-256: Unable to execute the command. Exit Code-512: Incorrect command options. Exit Code-655/-655 SYSTEM_ERROR: STARTJOB failures because auto_remote will not start. -656 NO_EXIT_CODE: exit_code field in database initialized to this. -657 PROCESS_MIA: Set by a chase-generated FAILURE event. for eg: chase cannot find the process. Oracle E-Business Suite Jobs Oracle job scheduler defined to run Oracle E-Business Suite workload. It is illustrated in the following ways. Request Set Job : To run multiple programs in oracle application. Single Request Job : To run single programs in oracle application. Copy Single Request Job : copying the existing Single Request Job and running it under the AutoSys agent. broderson ic 80 parts manual CA Workload Automation Reviews from subject matter experts posted in itcentralstation.com. In this tutorial, I tried to cover as vast i can when compared to other AutoSys tutorials. Hope you liked AutoSys user guide topics covered in this post , please feel free to share above autosys reference manual. Interview FAQ AutoSys Workload Automation is a multi-platform automated job control system that provides scheduling, monitoring, and reporting of AutoSys jobs. To report on global variables, use below commandautorep -G global_name [-n] [-w] [-d | -s] STARTING, RUNNING, INACTIVE, ACTIVATED, SUCCESS, FAILURE, TERMINATED, RESTART, QUE_WAIT, ON HOLD, ON ICE JIL stands for Job Information Language.JIL file/scripts used to built an job definition via command-line interface.JIL scripts contain one or more subcommands/attribute statements. There are two ways to create an AutoSys job definition.1) Using AutoSys GUI(AutoSys Graphical User Interface).2) Using Job Information Language (JIL) via command-line interface. File watcher job will watch for the arrival of specific file defined. By running command “autorep -j”. where -j refers to job name. t-_rex_fights_dinosaurs_dominator_edition_mod_apk.pdf By applying wildcards(%) with autorep command, we can export the AutoSys JIL’s.autorep -j % -q > /temp/filename.jil 1)Login to autosys wcc client url.2)Locate the job you want to mark it as success.3)Click on send event and select ‘mark as success’ from the drop-down list, then the job marked as success.You can perform the same through “send event” command in server command line. Define the python script as command like “/usr/bin/python” and then pass the script as “Argument”. Autosys commands 0.1d 24 August 2010 i | P a g e T C S P u b l i c 2 | P a g e T C S P u b l i c Document Release Note Document Details Name Version No. Description Autosys 0.1d The documents briefs the the Autosys system states and commands, and it also lists the JIL sub-commands and job attributes. Revision Details Action Taken (Addition/Deletion/ Modification) Previous Page No. New Page No. Revision Description Created by : Stephen Saldanha Authorised by: Date: 24 August 2010 Date: 3 | P a g e T C S P u b l i c About this Document Purpose This document has been written to help you define jobs using Autosys, and monitoring and managing these jobs. Intended Audience The following are the intended audience for this document: • Oracle DBA • System Administrators • Operations personnel References Autosys Reference Guide 4 | P a g e T C S P u b l i c Contents 1 Overview ....5 2 Autosys Commands ....7 3 JIL ....24 The total number of pages in this document, including the title page, is 38.

5 | P a g e T C S P u b l i c 1 Overview CA AutoSys is a full job scheduling and management system for NT systems. AutoSys lets you create simple or complex sets of instructions to automatically execute at regular intervals. AutoSys lets you trigger jobs by date and time, file arrival, and other criteria. If you decide to use AutoSys in a UNIX environment, it has a feature that lets you automatically convert cron job files for use on AutoSys. Like a mainframe scheduling product, AutoSys automatically monitors your jobs. If necessary, AutoSys can automatically restart jobs that have failed, were cancelled, or were aborted for a variety of reasons. For mission critical enterprise environments, AutoSys also has a high availability option that will perform an unattended rollover to a backup server if your primary server fails. CA AutoSys extends its management capabilities to Windows, Linux, UNIX, AIX, HPUX, and z/OS environments. What is a Job? • A job is a single action that can be performed on a valid Autosys client • A job is defined by assigning it a name and specifying the attributes that describe its associated behavior. These attributes form the Autosys job definition What are the types of Autosys jobs? 1. 64497221089.pdf Command jobs : Command jobs are basic jobs .It runs a specified command when the start conditions are met 2. File Watcher Job : These jobs are run based upon the following conditions a. 9.2 classifying chemical reactions worksheet answer key A file’s existence. b. A file being unchanged in size for a defined time period c. A file reaches a specified size 3. Box jobs: These jobs can be used to group jobs in a controllable branch stream. They are basically a container of jobs with like starting conditions eg Date and/or time or job dependencies. These jobs do not itself perform any actions but can trigger other actions. The diagram below explains a job structure 6 | P a g e T C S P u b l i c 7 | P a g e T C S P u b l i c 2 Autosys Commands Command Description You can execute AutoSys commands at the system prompt, using options and arguments to specify exactly what you want them to do. You can also embed many of these commands within shell scripts. The Autosys commands are explained in detail below archive_events archive_events removes data from various AutoSys database tables that are older than the specified number of days. You use this command to prevent the AutoSys database from becoming full. archive_events will optionally copy the information to an archive directory before deletion. 1 archive_events {-n num_of_days | -j num_of_days | -l num_of_days | -s num_of_days } [-A] [-d directory_name] [-B batch_size] [-D dataserver:database | -D TNSname] [-t timeout_in_secs] Options 1 -n num_of_days Indicates that records older than the num_of_days should be deleted from the event table in the database. -j num_of_days Indicates that the information older than the num_of_days should be deleted from the job_runs table in the database. -l num_of_days Indicates that the autotrack information older than the num_of_days should be deleted from the audit_info and audit_msg tables in the database. -s num_of_days Indicates that the job resource usage information older than the num_of_days should be deleted from the svarchive_tbl table in the database. -A Indicates that information is to be copied to an archive file before being deleted; otherwise, the information is discarded. -d directory_name Indicates a user-specified directory in which the archived data should be stored. If this option is omitted, data is archived to the default directory named $AUTOUSER/archive. -B batch_size Specifies the batch size—the number of events to be archived at a time. -D data_server:database Indicates the name of the Sybase dataserver, and the specific database within it, from which events are to be archived. -D TNSname Indicates the TNS alias name of the Oracle dataserver from which events are to be archived. -t timeout_in_secs Specifies the number of seconds to wait before timing out your SQL connection. 8 | P a g e T C S P u b l i c Examples 1 1 To copy all events in the events table older than 5 days to the default archive file, and delete it from the database, enter this: archive_events -A -n 5 2 To copy all job_runs statistics older than 5 days to a specified archive directory, and delete them from the database, enter this: archive_events -A -d /my_archive -j 5 autocal Autocal starts the Graphical Calendar Facility.The autocal command is used to start up the AutoSys Graphical Calendar Facility to define and maintain AutoSys calendars. Calendars are lists of dates that you can use to schedule the days on which jobs should, or should not, run Example 1 To start the Graphical Calendar Facility, enter this: autocal & autocal_asc 1 Autocal_asc adds, deletes, and prints custom calendar definitions. Autocal_asc provides a text-based, command line mechanism for creating, deleting, and printing custom calendars, which may be used to specify the days on which to start jobs, or days on which a job should not be started (e.g., holidays). Each calendar has a unique name and a list of days. Once created, calendars can be referenced in a job definition. Example 1 To start autocal_asc to begin entering calendar information, follow these steps: 1 Enter the following command: autocal_asc The following messages will appear: Utility to Add/Delete or Print entries in Calendar Calendar Name: 2 At this point you can enter the name of an existing calendar you want to edit, or the name of a new calendar. After entering the name, the following message appears: Add (A) or Delete (D) or Print (P)? 3 If you want to add a date to the calendar, enter this: A The following prompt appears: Date (MM/DD/[YY]YY [HH:MM]): 4 Using the following format, enter the date and, optionally, the time: MM/DD/[YY]YY [HH:MM] where: MM is the month, DD is the day, [YY]YY is the year, HH is the hours, in 24-hour format, and MM is the minutes. This prompt is repeated as long as dates are entered in response to the prompt.

5 To complete the additions, press without specifying a date. This action will also exit the autocal_asc utility.

9 | P a g e T C S P u b l i c autocons The autocons command starts up the AutoSys Operator Console for monitoring AutoSys jobs in real-time. Example 1 To start the Operator Console, enter this: autocons & autoflags 1 The autoflags command prints out the AutoSys version and release number, the database being used, and the operating system. autoflags [-a | -i | -o | -d | -v | -r | -h | -n] Options 1 -a : Displays all autoflags information to standard output. -I : Displays the AutoSys tape ID number to standard output. -o : Displays the operating system to standard output. -d : Displays the database type to standard output, either SYB for Sybase or ORA for Oracle. -v : Displays the AutoSys version number to standard output. -r : Displays the AutoSys release number to standard output. -h : Displays the hostid to standard output to standard output. -n : Displays the hostname to standard output to standard output. Example 1 To view all AutoSys and system configuration information, enter this: autoflags -a autoping 1 autoping verifies that the server and client machines are properly configured and are communicating successfully. It also checks and verifies that the Remote Agent and the Remote Agent’s database connection are functioning correctly. autoping -m {machine|ALL} [-A] [-D] Options 1 -m machine|ALL : The name of the machine to be verified. This must be a real machine, and must be listed in the /etc/hosts file on the machine from which the command is issued. ALL checks all machines. -A : Send an alarm if problems are detected. -D : Check the database connections on the specified machine(s).

Example 1 To check whether the machine “venice” is properly configured for AutoSys, and that its Remote Agent can function properly, enter this: autoping -m venice If successful, the following will display: AutoPinging Machine [venice] AutoPing WAS SUCCESSFUL! 10 | P a g e T C S P u b l i c To check all machines and verify their database access, enter this: autoping -m ALL -D If successful, the following will display: AutoPinging Machine [venice] AND checking the Remote Agent’s DB Access. AutoPing WAS SUCCESSFUL! AutoPinging Machine [rome] AND checking the Remote Agent’s DB Access. AutoPing WAS SUCCESSFUL! autosc 1 It starts the AutoSys Graphical User Interface (or GUI) and displays the GUI Control Panel. Example 1 To start the AutoSys Graphical User Interface, you enter: autosc & autostatus 1 autostatus writes the current status of the specified job or the current value of a global variable to standard output. This facility is especially useful in two circumstances: • When an application needs to know the status of another job. • When complex starting conditions are required that are beyond the scope of the starting conditions that can be easily specified in the job definition. autostatus {-J job_name | -G global_name} [-S autoserv_instance] Options 1 -J job_name Specifies the name of the job whose status needs to be determined.The current status is returned to standard output. -G global_name Specifies the name of a global variable that has been set using the sendevent command or the Send Event dialog. The value of the global variable is returned to standard output. -S autoserv_instance Specifies the three-character code of the AutoSys instance to be queried. The default is the value of $AUTOSERV from the current environment. Examples 1 1 To check the current status of the job named “test_install” in the current instance of AutoSys, enter this: autostatus -J test_install A one-word response to this command displays on standard output, such as the following: SUCCESS autosyslog 1 autosyslog is used to view either the Event Processor log file or the Remote Agent log file for the specified job. autosyslog [-e | -J job_name] [-p] 11 | P a g e T C S P u b l i c Options 1 -e Indicates that the Event Processor log is to be monitored. -J job_name Indicates that the Remote Agent log for the specified job_name is to be viewed. -p Specifies to append messages to the output file if anything in the profile file failed, if commands were not executed, or environment variables or definitions were not set. Examples 1 1 To view the Event Processor log, enter this on the command line of the system where the Event Processor is running, or where it ran last: autosyslog –e 2 To view the Remote Agent log of the last run of the “test_install” job, you would issue the following command on the command line of the machine where the “test_install” job ran: autosyslog -J test_install 3 To view the output file with profile information, enter this: autosyslog -j job_name -p This command will display the log file first, appending the profile output, if there is any. If no profile output file exists, the profile output section will be empty, for example: -- OutPut from File: auto_rem_pro.491.216.1 -- autosys_secure 1 It maintains the AutoSys Edit and Exec superuser ownerships, remote authentication methods and AutoSys database password. Also maintains NT user IDs and passwords, which are required for AutoSys jobs to run on NT client machines. autosys_secure Or autosys_secure [-q] {-a | -c | -d} -u _or_domain [-o old_password] -p password Options 1 -h Displays help. Use this option to get help on the autosys_secure command-line options. -q Specifies to run in quiet mode and not print any messages to the screen. -a Specifies to add a user ID and password. You must also supply the –u and -p options. -c Specifies to change an existing user password. You must also supply the -u, -o, and -p options. -d Specifies to delete an existing user password. You must also supply the -u and -o options. -u _or_domain Specifies the NT user whose password you are entering. NT user names can be from 1 to 20 characters in length and can contain all characters except the following:* [ ] + : ; " < > . , ? / \ | -o old_password 12 | P a g e T C S P u b l i c Specifies the password for an existing user. If you are changing a password or deleting a user ID and password, you must supply this option. If the password is NULL, enter NULL. -p password Specifies the password for the _or_domain that you want entered in the AutoSys database. Example 1 To start ecure in interactive mode, enter this: autosys_sautosys_secure The autosys_secure options display. autotimezone 1 autotimezone lets you query the timezones table, and add and delete timezones table entries. The timezones table contains entries that you can specify in a job definition using the timezone attribute autotimezone [-a entry_name value] [-c entry_name value] [-t timezone_name] [-d entry_name] [-q entry_name | sql_pattern] [-l] Options 1 -a entry_name value Adds an Alias entry to the timezones table. An Alias entry associates a name with a time zone. -c entry_name value Adds a City entry to the timezones table. -t timezone_name Adds a time zone entry to the timezones table. -d entry_name Deletes an entry from the timezones table. -q entry_name | sql_pattern Queries the timezones table for the setting of a specific alias, city, orzone. -l Lists all entries in the timezones table Examples 1 1 The following command adds a city named “San-Jose” to the timezones table: autotimezone -c San-Jose US/Pacific 2 The following command deletes the city named “San-Jose” from the timezones table: autotimezone -d San-Jose 3 The following command queries the timezones table for all entries beginning with the letter “d”: autotimezone -q d% The output from this command would be similar to the following: Entry Type Zone -- Dallas City US/Central Denver City US/Mountain Detroit City US/Central 13 | P a g e T C S P u b l i c a 1 utotrack 1 autotrack tracks changes to the AutoSys database (e.g., job definition changes, sendevent calls, and job overrides) and writes this information to the database. autotrack [-D data_server:database | -D TNSname] [-u 0|1|2] [-l] [-h|H] [-v] [-F "from_time"] [-T "to_time"] [-U user_name] [-m machine] [-J job_name] [-t A|B|C|J|M|O|S|T] Options 1 -D data_server:database Indicates the name of the Sybase dataserver, and the specific database within it, to be searched for the specified information. -D TNSname Indicates the TNS alias name of the Oracle dataserver to be searched for the specified information. -u tracking_level Updates the level of detail that autotrack writes to the database, using Level 0, 1, or 2. -l Displays the currently set tracking level (0, 1, or 2). -h|H Displays the autotrack usage summary. -v Verbose reporting. -F "from_time" Reports changes or events that occurred from this date and time forward; the format is “MM/DD/[YY]YY HH:MM”.

-T "to_time" Reports changes or events that occurred up to this date and time; the format is “MM/DD/[YY]YY HH:MM”.

-U user_name Reports changes or events initiated by the specified user. -m machine_name Reports changes or events initiated from the specified machine. -J job_name Reports on the specified job. The “%” character may be used in the job name as a wildcard. -t autotrack_type Examples 1 The amount of detail written to the database (and, thus, available to query against) is determined by the autotrack tracking level. Level 2 tracking provides much more detail than does level 1, as shown in examples 1 and 2. 1 The following query requests verbose reporting about a job named “NightlyReport.” autotrack -J NightlyReport -v If the autotrack tracking level had been set to 1, the output of this request would resemble that shown below. 11/21 10:04:54 job definition change :: 11/21 10:05:49 job definition change command: date :: 11/21 10:06:29 14 | P a g e T C S P u b l i c sendevent issued If the tracking level had been set to 2, autotrack would have written much more detail to the database. Thus, the same query autotrack -J NightlyReport -v would produce a report that includes the entire job definition with changes indicated by an asterisk. 2 To view all the changes that occurred to the job “NightlyReport” after 1 a.m. on November 12, 1997, enter this: autotrack -F "11/12/1997 01:00" -J NightlyReport 3 To view all changes made by user “sue” over the weekend of November 16 and 17, 1997, enter this: autotrack -U sue -F "11/16/1997 01:00" -T "11/17/1997 23:59" autorep autorep lists a variety of information about jobs, machines, and global variables currently defined in the AutoSys database.

You can use it to list a summary of all currently defined jobs, or to display current machine load information. autorep serves as a problem tracking tool by listing all relevant event information for the last run of any given job, or a specified job run. You can also use it to extract job definitions in JIL script format and save them to an output file for later re-loading into AutoSys, as a means of backing up job definitions. autorep {-J job_name | -M machine_name | -G global_name} [-s | -d | -q | -o over_num] [-r run_num][-L print_level] [-t] [-D data_server:database | -D TNSname] Options 1 -J job_name Indicates that a Job Report is desired. job_name specifies the name of the job on which to report. -M machine_name Indicates that a Machine Report is desired, which lists the machine’s Max Load, Current Load, and Factor. machine_name specifies the machine on which to report. -G global_name Indicates that a global variable report is desired, listing the variable name, value, and last modification date. global_name specifies the name of a global variable that has been set using the sendevent command or the Send Event dialog. In the specification, you can use ALL or wildcard characters. -s Indicates that a Summary Report is desired. This is the default. -d Indicates a Detail Report is desired.For a Job Report, all events from the last run of the requested job will be listed. -q Indicates a Query Report is desired, providing the current job or machine definition, in JIL format, as it exists in the AutoSys database. -o over_num Indicates an Override Report is desired, providing the overrides for the specified override number (over_num) and job_name -r run_num Indicates a report is desired for a specific job run (run_num). -t Requests that the time zone, if one is specified in the job definition, appear in the report. -L print_level (Job reports only.) Indicates the number of levels of nesting for boxes for which the specified information should be listed. 15 | P a g e T C S P u b l i c -D data_server:database Indicates the name of the Sybase dataserver, and the specific database within it, to be searched for the specified information.

-D TNSname Indicates the TNS alias name of the Oracle dataserver to be searched for the specified information. Example1 1 The following summary report is for a run of the Nightly_Download example, This is the command that requests the report: autorep -J Nightly_Download Job Name Last Start Last End ST RunPri/Xit ________________ ____ _________ Nightly_Download 11/10/1997 17:0011/10/1997 17:52SU 101/1 Watch_4_file 11/10/1997 17:0011/10/1997 17:13SU 101/1 filter_data 11/10/1997 17:13 11/10/1997 17:24SU 101/1 update_DBMS 11/10/1997 17:24 11/10/1997 17:52SU 101/1 chase 1 1 chase determines from the AutoSys database which jobs are in the STARTING or RUNNING state, and on which machine chase [-A] [-E] Options 1 -A Indicates that if chase detects any inconsistencies (i.e., jobs that should be running, but are not) it sends alarms to the AutoSys RDBMS(s). -E Indicates that if a job and the job’s Remote Agent are not running on the client machine, but the database indicates they should be, chase puts the job in FAILURE status, triggering the job to be restarted if the job definition includes the n_retrys attribute. Example 1 If a job is running longer than expected and you suspect it may have abnormally ended (but still shows as “running”), you should run chase. To verify that the job is running, receive an alarm if there is an inconsistency, and restart the job if necessary, enter this: chase -A -E chk_auto_up 1 1 It inspects the environment variables and configuration files, then determines if the AutoSys database (Event Server) and Event Processor are running. 1 chk_auto_up [-Q] Options 1 -Q Indicates that the command should output just the exit code without any descriptive message. Example 1 16 | P a g e T C S P u b l i c To check that the database and Event Processor are up and to view the results on your monitor, enter this: chk_auto_up chk_cond (SP) 1 AutoSys Stored Procedure 1 The chk_cond (SP) prints a report containing diagnostics about a job having starting conditions that are based on another job that does not exist. chk_cond job_name Options 1 job_name Specifies the name of the job against which diagnostics should be run. Example 1 A job named “jobA” has the following conditions specified: condition: success(jobB) and success(jobC). But, “jobC” does not exist. To print out diagnostics for all jobs in a Sybase dataserver, enter this: 1> chk_cond 2> go The following would be displayed to standard output: Job Missing_Condition_Job --------- ------------------------------- jobA jobC clean_files 1 The clean_files command deletes old Remote Agent log files from the various machines. clean_files -d days Options 1 -d days Specifies that log files older than the number of days will be removed. Example 1 To start clean_files and delete all Remote Agent log files older than 1 day, enter this: clean_files -d 1 cron2jil The cron2jil command converts each line in a UNIX crontab file to a corresponding JIL script (*.jil file) and, if necessary, a run calendar file(*.cal file). cron2jil -f crontab_file [-d output_directory] [-i include_file] [-m machine] [-p prefix] Options 1 -f crontab_file Specifies the name of the crontab formatted file. -d output_directory Specifies the directory to which the *.jil and *.cal files should be written. -i include_file 17 | P a g e T C S P u b l i c Specifies the name of a file containing JIL statements that are to be included in every generated *.jil file. -m machine Specifies the name of the machine on which the translation should occur. -p prefix Specifies a prefix that should be inserted before each job’s name. Example 1 To translate a crontab file with the name “daily”, enter this: cron2jil -f daily dbstatistics 1 1 dbstatistics performs the following tasks: • It update statistics in the database for optimal performance by invoking the Sybase Transact- SQL command update statistics. For Sybase databases, it updates the statistics for the event, job, job_status, and job_cond tables. For Oracle, it computes statistics for all of the AutoSys tables. • dbstatistics runs the AutoSys dbspace command to check the available space in the database. dbspace prints a summary of the free space versus the used space in the database. If the amount of free space is insufficient, dbspace issues warning messages and generates a DB_PROBLEM alarm. eventor 1 1 Use the eventor command to bring up the Event Processor (and, optionally, the Shadow Event Processor), also referred to as the “event demon.” eventor runs in the background, by default. eventor [-f] [-n] [-q][-G] [-M shadow_machine] [-Z $ZTEAMDIR] Options 1 -f Specifies that the Event Processor should run in the foreground, and all of its output should be sent to the display from which the command was issued.

-n Specifies that eventor is not to run the chase command on start-up. -q Specifies that eventor should run in quiet mode, meaning that after the Event Processor has been started, eventor should not execute the tail -f command on the event_demon log file. -G Starts up the Event Processor in Global AutoHold mode. -M shadow_machine Specifies that a Shadow Event Processor should be started on the machine named shadow_machine. -Z $ZTEAMDIR Sets the ZTEAMDIR environment variable, if it is not already set Examples 1 1 To start the Event Processor under normal circumstances, enter this: eventor 2 To start the Event Processor on the local machine, and a Shadow Event Processor on the machine named “jupiter”, enter this: 18 | P a g e T C S P u b l i c eventor -M Jupiter gatekeeper 1 1 gatekeeper is an interactive utility you can use to maintain AutoSys license keys located in the AutoSys database. Example 1 To start the license utility, enter this: gatekeeper job_depends 1 1 job_depends provides detailed reports about the dependencies and conditions of a job. This command can be used to determine the current state of a job, its job dependencies, and (for boxes) nested hierarchies as specified in the job definition, and a report of what jobs will run during a given period of time. job_depends [-c | -d | -t] [-J job_name] [-F from_date/time] [-T to_date/time] [-L print_level] [-D data_server:database | -D TNSname] Options 1 -c Prints out the current state of a job and the names of any jobs that are dependent on this job. -d Prints out the starting conditions for a job; no indication of the current status of the job is provided.

-t Time Dependencies. Prints out the starting conditions for a job. -J job_name Indicates the job on which to report, where job_name is the name of the target job. -F "from_date/time" Indicates the report start date and time, where from_date/time is the date and time of the first job in the report. -T "to_date/time" Indicates the report end date and time, where to_date/time is the date and time of the last job in the report. -L print_level Indicates the print level for the report, where print_level is any valid numeric value specifying the number of levels of nesting to display for a box job. -D data_server:database Indicates the name of the Sybase dataserver, and the specific database within it, to be searched -D TNSname Indicates the TNS alias name of the Oracle dataserver to be searched for the specified information. Example 1 1 To display a report on the current condition status of a job named “jobX”, you would issue the following command: job_depends -c -J jobX You would see a report similar to the following displayed to standard output: ____ Start Dependent Job Name Status Date Cond? Cond? Jobs? -------- ------ ---------- ------ --------- 19 | P a g e T C S P u b l i c jobX INACTIVE No No Yes Dependent Job Name ------------------ jobY ____ This report shows that “jobX” has no date or starting conditions, but another job, “jobY” is dependent on it.

2 To display a report on the current condition status of a job named “jobA”, which does have starting conditions and dependencies, you would issue the following command: job_depends -c -J jobA 3 To display a report on the box job named “job_bxA” showing all the nested levels of jobs and boxes within this job, you would enter the following command: job_depends -d -J job_bxA 4 To display a report on jobs that are scheduled to run on New Years day, you would enter the following command: job_depends -t -J ALL -F "01/01/1998 00:00" -T "01/02/1998 00:00" monbro 1 1 monbro runs a monitor or report (browser) that has already been defined, either using either jil or the GUI. monbro -N name [-P poll_frequency] [-D data_server:database| -D TNSname] [-q] Options 1 -N name Specifies the name of the monitor or report (browser) to be run. -P poll_frequency Applies to monitors only, and indicates the time interval (in seconds) to sleep between polls of the database. -D data_server:database Specifies the name of the Sybase dataserver, and the specific database within it, from which to retrieve events and the monitor or report definition. -D TNSname Specifies the TNS alias name of the Oracle database from which to retrieve events and the monitor or report definition. -q Specifies that you want to display monbro definitions in JIL format. Example To run a monitor called “mon1” which is defined in the default database, enter: monbro -N mon1 Sample output with the q option: -monbro -N mon1 -q insert_monbro: xxx mode: m all_events: Y job_filter: a sound: N alarm_verif: N insert_monbro: xxx2 20 | P a g e T C S P u b l i c mode: b all_events: N alarm: Y all_status: N running: N success: Y failure: Y terminated: N starting: N restart: N job_filter: b job_name: box currun: N after_time: "11/11/1997 12:12" record_sounds 1 1 This utility records sounds for playback in monitors. It stores the sounds in files located in the $AUTOUSER/sounds directory. record_sounds AutoSys_password Example 1 To record sounds, be sure the workstation you are on is equipped for sound, has a microphone plugged in, and is set up correctly, then, enter this: record_sounds sendevent 1 Sends events to AutoSys for a variety of purposes, including starting or stopping AutoSys jobs, stopping the Event Processor, and putting a job on hold. This command is also used to set AutoSys global variables or cancel a scheduled event. 1 sendevent -E event [-S autoserv_instance] [-A alarm] [-J job_name] [-s status] [-C comment] [-P priority] [-M max_send_trys] [-q job_queue_priority] [-T "time_of_event"] [-G "global_name=value"] [-k signal_number(s)] [-u] Options 1 -E event Specifies the event to be sent. This option is required.

Any one of the following events may be specified: STARTJOB Start the job specified in -J job_name if the starting conditions are satisfied. KILLJOB Kill the job specified in -J job_name. DELETEJOB Delete the job specified in -J job_name from the database. FORCE_STARTJOB Start the job specified in -J job_name, regardless of whether the starting conditions are satisfied. JOB_ON_ICE Puts the job specified in -J job_name “on ice.” JOB_OFF_ICE Takes the job specified in -J job_name “Off Ice”. Jobs that are taken “Off Ice” will not start until the next time their starting conditions are met. 21 | P a g e T C S P u b l i c JOB_ON_HOLD Puts the job specified in -J job_name “On Hold”. When a job is “OnHold”, it will not be started, and downstream dependent jobs will not run. JOB_OFF_HOLD Takes the job specified in -J job_name “Off Hold.” If the starting conditions are met, the job will be started. CHANGE_STATUS Forces a change in the status of the job specified in -J job_name. STOP_DEMON Stops the Event Processor (event demon). This is the only way to stop the Event Processor. CHANGE_PRIORITY Changes the Job Queue Priority of the job specified in -J job_name to the priority specified by the -q priority arguments.

COMMENT Attaches a message to the event, for informational purposes only. ALARM Sends an alarm. Generally alarms are generated internally; however, using this event, users and programmers can send alarms to alert operators. SET_GLOBAL Sets an AutoSys global variable. This event is sent with a high priority so the Event Processor will process the variable before it is referenced by any jobs at runtime. SEND_SIGNAL Sends a signal to a running AutoSys job. -S autoserv_instance Specifies the three-character AutoSys instance (e.g., ACE) to which the event should be sent. -A alarm Specifies the name of the alarm to be sent. This option is only used when the specified event is ALARM; it is required when using this event. -J job_name Specifies the name of the job to which the specified event should be sent -s status Specifies the status to which the job specified in -J job_name should be changed. -C comment Specifies a textual comment that is to be attached to this event for, documentation purposes only. -P priority Specifies the priority to be assigned to the event being sent. -q job_queue_priority Specifies the new queue priority to be assigned to the job. -T "time_of_event" Specifies the date and time when the event should be processed. -G "global_name=value" Specifies the name and value of a global variable when a SET_GLOBAL event is sent. -k signal_number(s) or signal_name For processes running on UNIX, this argument specifies the signal number.

-u Cancels the event specified in the -E event option. Examples 1 1 To start a job named “test_install” that has no starting conditions (and therefore must be started manually), enter this: sendevent -J test_install -E STARTJOB 22 | P a g e T C S P u b l i c 2 To force a job to start named “wait_job”, which is waiting on the completion of another job, and explain the reasons for your action, enter this: sendevent -J wait_job -E FORCE_STARTJOB -C "tired of waiting,forced it" 3 To change the status of a job called “ready_to_run” to ON_HOLD to prevent its execution, and to assign the command a high priority so it will be sent immediately, enter this: sendevent sendevent -J ready_to_run -E JOB_ON_HOLD -P 1 When you want the above job to run, enter this: sendevent -J ready_to_run -E JOB_OFF_HOLD 4 To prevent a job called “lock_out” from running between the hours of 11:00 a.m. and 2:00 p.m., a pair of sendevent commands could be used to place it on hold during that time. (These same sendevent commands could be placed in a job that is run daily to perpetuate this condition on a regular basis.) To put the job on hold at 11:00 a.m., enter this: sendevent -J lock_out -E JOB_ON_HOLD -T "11/08/1997 11:00" To take the job off hold at 2:00 p.m., enter this: sendevent -J lock_out -E JOB_OFF_HOLD -T "11/08/1997 14:00" 5 To write a comment into the Event Processor log file, enter this: sendevent -E COMMENT -C "have not received EOD files - an hour late again" 6 To stop the Event Processor at 2:30 a.m. on November 9, 1997 (it is always a good idea to attach a comment to this event), enter this: sendevent -E STOP_DEMON -T "11/09/1997 02:30" -C "stopped for upgrade" 7 To change a job called “resource_hog” to a lower priority (it is currently at 1 and is not yet running), and to only issue the command 5 times, rather than letting it try indefinitely, enter this: sendeventsendevent -J resource_hog -E CHANGE_PRIORITY -q 10 -M 5 8 To kill a job named “wrong_job” which is running on another AutoSys instance called “PRD”, enter this: sendevent -J wrong_job -E KILLJOB -S PRD 9 To set a global variable named “today” having a value of “12/25/ 1997”, enter this: sendevent -E SET_GLOBAL -G "today=12/25/1997" 10 To delete the global variable named “today”, enter this: sendevent -E SET_GLOBAL -G "today=DELETE" 11 To send the Unix signal number 1 to a job named “RunData”, enter this: sendevent -E SEND_SIGNAL -J RunData -k 1 12 To cancel all unprocessed JOB_OFF_HOLD events for a job named “RunData”, enter this: sendevent -E JOB_OFF_HOLD -J RunData -u xql 1 xql is the AutoSys-supplied utility that accesses the Sybase dataserver from any properly configured client. xql -U user_name -P password [-S server] [-D database] [-c "command_string" | -f input_file] [-d delimiter] [-l] 23 | P a g e T C S P u b l i c [-T timeout_interval] Options 1 -U user_name Specifies the name of the Sybase user to log in as. -P password Specifies the Sybase password for the specified user_name. -S server Specifies the name of the Sybase dataserver to be accessed. -D database Specifies the specific Sybase database to be accessed.

-c "command_string" Specifies an SQL statement to be passed to Sybase and executed in “batch”, rather than interactive mode. -f input_file Specifies a text file containing SQL statements to be passed to Sybase, to be executed in batch -d delimiter Specifies the delimiter to be used for output, which is written to standard output. -l Specifies that a long listing is desired, meaning that the output should be displayed as one column name (i.e., field name) with its corresponding value per line. -T timeout_interval Specifies a period of time after which xql will terminate the session if no activity has occurred. Example The following examples assume that the Sybase account and password are “autosys” and “autosys” respectively. The examples also assume that the dataserver defaults to AUTOSYSDB, and the database defaults to “autosys”. 1 To select the job ID and job name (the field names are assigned by AutoSys) from the job table in the default dataserver and database, enter this (using the “autosys” user and the “autosys”, or the appropriate, password): xql -Uautosys -Pautosys Then, at the xql prompt, enter this: xql>>[AUTOSYSDB][autosys] 1> select joid, xql>>[AUTOSYSDB][autosys] 2> job_name xql>>[AUTOSYSDB][autosys] 3> from job; Assuming that there are only three jobs, this will be the output: joid job_name -------- ---------------------------- 101 tester 106 test1 107 domail.tibet To exit the interactive session, enter this: xql>>[AUTOSYSDB][autosys] 1> exit 24 | P a g e T C S P u b l i c 3 JIL Jil runs the Job Information Language (JIL) processor to add, update, and delete AutoSys jobs, machines, monitors, and reports. Also used to insert one-time job override definitions.The jil executable is the language processor for the AutoSys Job Information Language (JIL).

Using JIL (the language itself), you can define and update jobs, monitors, reports, and machines. The jil command can be used in one of two ways: • To automatically submit job definitions to the AutoSys database. You do this by redirecting a JIL script file to the jil command. • To interactively submit job definitions to the AutoSys database. You do this by issuing the jil command only and entering JIL statements at the provided jil>> prompts. To exit interactive mode, enter “exit” at the prompt, or press . jil [-q] [-S autoserv_instance] [-V none | job | batch] Options 1 -q Specifies that jil should be run in “quiet” mode and that it should only output error messages. -S autoserv_instance Specifies the three-character AutoSys instance name, e.g. ACE, (and therefore the RDBMS) to which to apply the definition(s). -V none | job | batch Specifies whether or not the JIL processor should verify that jobs specified in the job dependency condition for the job actually exist in the AutoSys database. Examples 1 1 To redirect a text file named “job1” containing JIL statements into jil, enter this: jil < job1 2 To redirect a text file named “job1” containing JIL statements into jil and prohibit the JIL processor from verifying the existence of specified jobs in its job dependencies, enter this: jil < job1 -V none JIL Sub-commands JIL sub-commands are used to establish if you are creating, updating, or deleting a job. When using the AutoSys GUI, the same instructions are conveyed by entering values in various fields, and/or by pressing different buttons in the GUI’s dialog boxes. Job Attributes AutoSys job attributes are used to specify everything from the name of a new job to the specific exit conditions which must be “successful” in order for the job to be considered completed. Job attributes can be defined using JIL statements, which are input to the jil command, or they can be defined using the AutoSys Graphical User Interface (GUI). Regardless of method, the attributes are virtually the same. 25 | P a g e T C S P u b l i c alarm_if_fail 2 JIL Attribute Indicates whether an alarm should be posted to the Event Processor if the job fails or is terminated.

alarm_if_fail: toggle toggle can be y or 1 for yes; or n or 0 for no. Example 2 To set the job currently being created or updated to post an alarm if it fails or is terminated, enter this: alarm_if_fail: y auto_delete 2 JIL Attribute Indicates whether the job should be automatically deleted after completion.

If auto_delete is set to 0, AutoSys will immediately delete job definitions only if the job completes successfully. If the job does not complete successfully, AutoSys will keep the job definition for seven days before automatically deleting it. auto_delete: value where value can be any number of hours; 0 indicates immediate deletion, while -1 indicates that the job should not be deleted. Example 2 To set the job to be automatically deleted 5 hours after completion, enter this: auto_delete: 5 auto_hold 2 Job Attribute This feature is only for jobs that are in a box. When a job is in a box, it inherits the starting conditions of the box.

This means that when a box goes into the RUNNING state, the box job will start all the jobs within it (unless other conditions are not satisfied). By specifying “yes” to AutoHold On, AutoSys automatically changes the job state to ON_HOLD when the box it is in begins RUNNING. To start the job, take the job off hold by sending the JOB_OFF_HOLD event. This is done with the AutoSys sendevent command. auto_hold: toggle toggle can be y or 1 for yes; or n or 0 for no. Example 2 To set the job to be automatically placed on hold, enter this: auto_hold: y 26 | P a g e T C S P u b l i c avg_runtime (JIL only) 2 Job Attribute 2 Indicates an average run time (in minutes) for a job that is newly submitted to the AutoSys database; it establishes this value in the absence of the job having been run multiple times. avg_runtime: value where value can be any number of minutes, to include decimal numbers. Example 2 To set the average run time for a new job to be five minutes, enter this: avg_runtime: 5 box_failure 2 Job Attribute Specifies the conditions to be interpreted as a box failure. The Box Completion Conditions appears in the Job Definition dialog only when you select a box job, and when you are opening an existing box job definition. box_failure: conditions where conditions can specify any of the dependencies described in the Examples 2 1 To set the status of the box currently being created or updated to FAILURE if “JobA” fails or “JobB” fails, but ignoring if “JobC” fails, enter this: box_failure: failure(JobA) OR failure(JobB) 2 To set the status of the box currently being created or updated to FAILURE only if all three jobs fail, enter this: box_failure: failure(JobA) AND failure(JobB) AND failure(JobC) box_name 2 Job Attribute 2 Indicates the name of the box in which this job is to be placed. box_name: name where name can be any string of up to 30 alphanumeric characters, plus the underscore character (_). Example 2 To specify that the job currently being created or updated should be put in the box named “Box1”, enter this: box_name: Box1 27 | P a g e T C S P u b l i c box_success 2 Job Attribute Specifies the conditions to be interpreted as a box success. box_success: conditions where conditions can specify any of the dependencies Examples 2 1 To set the status of the box currently being created or updated to SUCCESS only when “JobA” succeeds or “JobB” succeeds, but ignoring the status of “JobC”, enter this: box_success: success(JobA) OR success(JobB) 2 To set the status of the box currently being created or updated to SUCCESS only if all three jobs succeed, and they are the only jobs in the box, enter nothing. This is the default behavior of box jobs. 3 To set the status of the box currently being created or updated to SUCCESS only if jobs “JobA” and “JobB” succeed, and “JobC” completes, regardless of its status, enter this: box_success: success(JobA) AND success(JobB) AND done(JobC) box_terminator 2 Job Attribute This attribute specifies whether the box containing this job should be terminated if the job fails or terminates box_terminator: toggle where toggle can be y or 1 for yes; or n or 0 for no. Example 2 To specify that if the job currently being created or updated fails, the box it is in should be terminated, enter this: box_terminator: y chk_files 2 Job Attribute 2 This resource check specifies the minimum amount of file space that must be available on designated file system(s) for the job to be started.

One or more file systems, specified with full pathnames or directory names, and their corresponding sizes, can be specified. If multiple file systems are specified, separate them with a single space. chk_files: file_system_name size [file_system_name size]... file_system_name = The full pathname of the file system where the file space will be needed, and environment variables exported in the profile can be used in the pathname. Size = Is the file space needed (in kilobytes). Many file_system_name size pairs can be specified, separated by aspace.

Example To specify that the job currently being created or updated should have 100K of space available on the file system named “rootfs” and 120K of space available on the file system named “auxfs1”, enter this (using the full pathname): 28 | P a g e T C S P u b l i c chk_files: /rootfs 100 /auxfs1 120 command 2 Job Attribute The command attribute can be the name of a command, shell script, or application program that is to be run on the client machine (when all necessary conditions are met). command: command_name command_runtime_args command_name = The command_name can be the name of any command, shell script, or application program executable. command_runtime_args = Any runtime arguments. Examples 2 1 To specify that the UNIX date command is to be executed, enter this: command: /bin/date 2 If the /bin directory is included in the search path, either in the /etc/auto.profile or in the user-defined profile, the UNIX date command can be specified to execute by entering this: command: date 3 To specify that the “Backup” script in the /usr/common directory is to be executed, enter this: command: /usr/common/Backup Or If the /usr/common directory is included in the runtime environment path of the job being defined, enter this instead: command: Backup 4 To specify that the “Backup” script in the /usr/common directory is to be passed today’s date (that has been set as the global variable named “RunDate”), you could enter this: command: /usr/common/Backup -D $$RunDate 5 To remove all files from the /tmp subdirectory under the directory specified in the “MY_BACKUPS” global variable, you could enter this: command: rm $${MY_BACKUPS}/tmp/* condition 2 Job Attribute Description 2 When using the condition attribute, any number of job dependencies can be specified. All dependencies must evaluate to “true” before the dependent job will be run. condition: [(]condition[)][{AND | OR }[(] condition [)]]... where conditions can specify any combination of the dependencies Examples 2 1 This is the job dependency specification for a job which is to run only if the job named “DB_BACKUP” succeeds: condition: success(DB_BACKUP) 2 If “JobC” should be started only when both “JobA” and “JobB” complete successfully or when both “JobD” and “JobE” complete,regardless of whether they failed, succeeded, or terminated, specify the following dependency in the job definition for “JobC”: condition: (success(JobA) AND success(JobB)) OR (done(JobD) AND done(JobE)) 29 | P a g e T C S P u b l i c date_conditions 2 This attribute specifies whether or not there are date and/or time conditions for starting this job. If it is set to “no”, the remainder of the date/time related attributes will be ignored. If set to “yes”, the date can be specified using the days_of_week attribute, or the specific dates can be specified by associating this job with a custom calendar, created using the Graphical Calendar facility or the autocal_asc command. date_conditions: toggle where toggle can be a y or 1 for yes; or n or 0 for no. Example 2 To specify that starting date and time conditions are to be in effect, enter this: date_conditions: y days_of_week 2 Job Attribute Indicates the days of the week when the job will be run. One or more days can be selected, or all days can be selected. days_of_week: {day [,day]... | all} where day can be any of the following: mo (Monday), tu (Tuesday),we (Wednesday),th (Thursday),fr (Friday),sa (Saturday), su (Sunday) or all can be specified for every day of the week. Examples 2 1 To specify that the job should be run only on weekdays, enter this: days_of_week: mo, tu, we, th, fr 2 To specify that the job should be run every day of the week, enter this: days_of_week: all delete_box 2 JIL Sub-command The delete_box sub-command deletes the specified box and all the jobs in that box. delete_box: box_name Where box_name must be a box currently defined in the AutoSys database. Example 2 To delete a box named “Box1” and all jobs inside it, you would specify the following sub-command in the JIL script: delete_box: Box1 30 | P a g e T C S P u b l i c delete_job JIL Sub-command The delete_job sub-command deletes the specified job from the AutoSys database. Even if the job is already scheduled to run, it will not be run. delete_job: job_name where job_name must be a job or box currently defined in the AutoSys database. Example 2 To delete a job called “Job1”, you would specify the following sub-command in the JIL script: delete_job: Job1 description 2 Job Attribute Specifies a description for the job; for documentation purposes only. description: text where text can be any string of alphanumeric characters, up to 255 characters. Spaces can be included. Example 2 To specify that the job is an incremental daily backup of the database, enter this: description: "incremental daily backup of the database" exclude_calendar 2 Job Attribute Indicates the name of the custom calendar to be used for determining the days of the week on which this job will not run. exclude_calendar: calendar_name where calendar_name must be the name of a custom calendar that has already been created. Example 2 To specify that the job can be run on any day except those days specified in the “holiday” calendar, which you have previously defined, enter this: exclude_calendar: holiday heartbeat_interval 2 Job Attribute Specifies the frequency (in minutes) at which this job’s command is expected to issue a heartbeat. Heartbeats are AutoSys’s way of monitoring a job’s actual progress. heartbeat_interval: mins where mins specifies the number of minutes; any reasonable number is acceptable. 31 | P a g e T C S P u b l i c Example 2 To set the heartbeat to be expected every 2 minutes, modify your program to call the heartbeat routine every 2 minutes or less by entering the following: heartbeat_interval: 2 insert_job 2 JIL Sub-command 2 The insert_job sub-command adds a new command, box, or file watcher job definition to the AutoSys database. insert_job: job_name job_name = The unique job identifier used throughout AutoSys. It can be from 1 to 30 alphanumeric characters, and is terminated with white space. Example 2 1 The following example creates a command job, specifying only the essential job attributes. The job is called “time_stamp”, is to run on the real machine “tibet”, and simply executes the time_stamp.sh shell script. To create this definition, enter the following sub-command and job attributes in the JIL script: insert_job: time_stamp machine: tibet command: time_stamp.sh The job_type attribute is optional when defining a command job. To specify a command job, enter this: job_type: c job_load 2 Job Attribute Specifies the relative amount of processing power the job will consume. job_load: load_units where load_units specifies the relative load of the job, and can be any arbitrary value within the user-defined range of possible values (which are also arbitrary). Example 2 To set the job load for a job that typically uses 10% of the CPU, with arange of possible load values from 1-100, enter this: job_load: 10 job_terminator 2 Job Attribute This attribute specifies whether the job should be terminated if the box it is in fails or terminates. job_terminator: toggle where toggle can be y or 1 for yes; or n or 0 for no. 32 | P a g e T C S P u b l i c Example 2 To specify that if the box containing the job currently being created or updated fails, the job should be terminated, enter this: job_terminator: y job_type 2 Job Attribute Specifies whether the job is a command job, file watcher job, or box job.

job_type: type where type can be any one of the following: c (command) f (file watcher) b (box) Example 2 To set the job currently being created or updated to be a box job, enter this: job_type: b machine 2 Job Attribute Specifies the client machine where the job will be run, under the control of the Remote Agent. machine: {machine_name [, machine_name]...| ‘machine_chooser_script’} where machine_name can be any real machine, virtual machine, or set of real machines.

The name can be up to 80 characters. Examples 2 1 To specify that the job be executed on either of the machines named “tibet” or “socrates”, enter this: machine: tibet, socrates 2 To run the svload program at runtime to determine which machine to use, enter this: machine: ’svload -a alg [-v virt | -l list] -p profile’ 3 To run the script /usr/local/bin/my-machine-chooser at job runtime to determine which machine to use, enter this: machine: ’/usr/local/bin/my-machine-chooser’ max_exit_success 2 Job Attribute 2 Specifies the maximum UNIX exit code with which the job can exit and still be considered a success by AutoSys.

max_exit_success: exit_code where exit_code can be any integer representing a UNIX exit code. 33 | P a g e T C S P u b l i c Example 2 To set the job to be considered successful when exiting with any exit code of “2” or less, enter this: max_exit_success: 2 max_run_alarm 2 Job Attribute Specifies the maximum run time (in minutes) that a job should require to finish normally. max_run_alarm: mins where mins can be any integer; it represents the maximum number of minutes the job should ever require to finish normally. Example 2 To set the job to be considered as running too long if it runs for more than an hour and a half, enter this: max_run_alarm: 90 min_run_alarm 2 Job Attribute 2 Specifies the minimum run time (in minutes) that a job should require to finish normally. min_run_alarm: mins where mins can be any integer; it represents the minimum number of minutes the job should ever require to finish normally. Example 2 To set the job to be considered as completing too quickly if it runs for less than an hour and a half, enter this: min_run_alarm: 90 n_retrys 2 Job Attribute Specifies how many times, if any, the job should be restarted after exiting with a FAILURE status n_retrys: attempts where attempts can be any integer between 1 and 20. Example 2 To set the job to be automatically restarted up to 5 times after an application failure (not system or network related), enter this: n_retrys: 5 override_job 2 JIL Sub-command The override_job sub-command specifies that a one-time override be applied to a particular job, for the indicated attributes 34 | P a g e T C S P u b l i c override_job: {job_name | job_name delete} attribute_keyword: {value | NULL} job_name = Must be a job or box currently defined in the AutoSys database. There is no default. Delete = Used to cancel a previously specified job override.

NULL = Used to delete or negate any currently existing value for the indicated attribute_keyword. Examples 2 1 To specify a one-time job override for the job named “job1” to change the standard output file, enter the following sub-command and attribute in the JIL script: override_job: job1 std_out_file: /usr/out/run.special 2 To specify a one-time job override for the job named “jobA” to delete its job dependency condition and change the standard output file, enter the following sub-command and attributes in the JIL script: override_job: jobA std_out_file: /usr/out/run.special owner 2 Job Attribute 2 Specifies the owner of the job. The owner is the user who invoked jil or the GUI Control Panel to define the job.

This user will own all jobs defined during the session, and will have edit permission on the jobs. owner: { | user} where can be any valid user with an account on the specified machine, which must be a real, not a virtual machine. Example 2 For the Edit Superuser to change the owner such that “chris” on any machine in the network can edit the job, and the job’s command will run with the permissions of “chris”, enter this: owner: chris permission 2 Job Attribute 2 The AutoSys permission scheme is based on the same permissions used in native UNIX. permission: permission [, permission] When a job is first created, the user ID is retrieved from the environment and attached to the job. Then the current value of the owner’s umask is used to supply default permissions to the job. The umask “write” permission is used as the default “edit” permission of the job, and the umask “execute” permission is used as the default “execute” permission of the job.These are the possible values for the permission attribute: Gx = Group Execute Ge = Group Edit Mx = Execute by any authorized users, regardless of the machine they are on Me = Edit by any authorized users, regardless of the machine they are on Wx = World Execute We = World Edit 35 | P a g e T C S P u b l i c Example 2 To set the job to allow anyone to execute it, but to allow only members of your group to edit it, enter this: permission: ge, wx priority 2 Job Attribute Specifies the queue priority of the job. priority: priority_level where priority_level can be any integer 0 or larger. priority_level 0 indicates that the job should always be run immediately, regardless of the current machine load. Examples 2 1 To set the job to always run, regardless of the current load on the client machine, accept the default which is 0. 2 To set the job to run with the highest priority, while not overriding the machine load control mechanism, enter this: priority: 1 3 To set the job to run in the background when the machine load is low, enter this: priority: 100 profile 2 Job Attribute Specifies the profile that is to be sourced by the Bourne shell before the specified command is executed. profile: pathname pathname = The full pathname of the profile file to be sourced in order to establish the job’s runtime environment. Variable substitution cannot be used. Example 2 To set the user’s profile called le in their home directory called /usr/home, enter this: my_profiprofile: /usr/home/my_profile run_calendar 2 Job Attribute 2 Indicates the name of the custom calendar to be used when determining the days of the week on which a job will run. run_calendar: calendar_name where calendar_name must be the name of a custom calendar that has already been created. Example 2 To specify that the job should be run on the last business day of the month, as specified in the previously created custom calendar named “last_business”, enter this: run_calendar: last_business 36 | P a g e T C S P u b l i c run_window Job Attribute Indicates the time span during which the job will be allowed to start run_window: "time-time" where time-time must be entered in quotes, using the format "hh:mm-hh:mm" where the hh specifies hours, in 24-hour format, and the mm specifies minutes. Example 2 To specify that the job should be allowed to start only between 11:00 p.m. and 2:00 a.m., regardless of other conditions, enter this: run_window: "23:00-02:00" start_mins 2 Job Attribute Indicates the number of minutes past the hour, every hour, on the specified days or dates, when the job will be started. start_mins: mins [, mins]... where mins must be a number 0–59, representing the number of minutes past each hour when the job will be run.

Example 2 To specify that the job be run at a quarter past and a quarter before each hour, enter this: start_mins: 15, 45 start_times 2 Job Attribute Indicates the times of day, in 24-hour format, on the specified days or dates, when the job will be started. start_times: "time [, time]..." where time must be specified using the format "hh:mm" where the hh specifies hours, in 24-hour format, and the mm specifies minutes. Example 2 To specify that the job be run at 10:00 a.m. and 2:00 p.m. on every specified day or date, enter this: start_times: "10:00, 14:00" std_err_file 2 Job Attribute Specifies the file to which the standard error file’s output should be redirected std_err_file: pathname Enter the std_err_file keyword and the full pathname for the standard error file.

37 | P a g e T C S P u b l i c Examples 2 1 To set the file /tmp/test.err to receive standard error file output for the job, enter this: std_err_file: /tmp/test.err 2 To append new information to the error file, enter: std_err_file: >>/tmp/test.err std_in_file 2 Job Attribute 2 Specifies the file to which the standard input file for the job should be redirected. std_in_file: pathname Enter the std_in_file keyword and the full pathname of the standard input file. Examples 2 1 To set the file named /tmp/test.in to be read as the standard input file, enter this: std_in_file: /tmp/test.in 2 To set the file named /tmp/today’s_date.in to be read as the standard input file, set a global variable named “Today” (using sendevent or the Send Event dialog) to be today’s date, then enter this: std_in_file: /tmp/$${Today}.in std_out_file 2 Job Attribute Specifies the file to which the standard output file should be redirected. std_out_file: pathname Enter the std_out_file keyword and the full pathname of the standard out file.

Examples 2 1 To set the file named ut to receive standard output for the job, enter this: /tmp/test.ostd_out_file: /tmp/test.out 2 To append new information to the output file, enter: std_err_file: >>/tmp/test.out term_run_time 2 Job Attribute Specifies the maximum run time (in minutes) that a job should require to finish normally. term_run_time: mins where mins can be any integer; it represents the maximum number of minutes the job should ever require to finish normally. Example 2 To set the job to be automatically terminated if it runs longer than 90 minutes, enter this: term_run_time: 90

timezone 2 Job Attribute Allows you to schedule a job based on a chosen time zone. When the timezone attribute is specified in a job definition, the time settings in that job are based on the zone time zone timezone: zone 2 Zone = Either a time zone recognized by the operating system or a case-insensitive string of characters corresponding to an entry in the timezones table. Example 2 To set the time zone for a job definition to Chicago time, enter this: timezone: Chicago To set the time zone for a job definition to Pacific time, enter this: timezone: US/Pacific If you specify a time zone that includes a colon, you must quote the time zone name if you are using JIL, like this: timezone: "IST-5:30" update_job 2 JIL Sub-command The update_job sub-command updates an existing command, box, or file watcher job definition in the AutoSys database. update_job: job_name job_name = The unique job identifier used to define the original job to AutoSys. Example 2 To change a pre-existing command job called “time_stamp” to run on the real machine “paris”, rather than on the originally specified machine, enter the following sub-command and job attribute in the JIL script: update_job: time_stamp watch_file 2 Specifies the file for which this file watcher job should watch. The name of the file to watch for must be a legal UNIX filename, and it must identify the full pathname of the file. watch_file: pathname pathname = The full pathname of the file for which to watch. Examples 2 1 To set the file watcher to watch for a file named /tmp/batch.input, enter this: watch_file: /tmp/batch.input 2 To set the file watcher to watch for a file whose name has been assigned to a global variable named “file_1”, enter this: watch_file: $${file_1}

watch_file_min_size 2 Job Attribute Specifies the watch file minimum size (in bytes) which determines when enough data has been written to the file to consider it complete. watch_file_min_size: bytes where bytes can be any integer; it represents the minimum number of bytes in the file before it is considered complete. Example 2 To set the file to be considered complete when it reaches 10K bytes (assuming the file has reached “steady state” as well), enter this: watch_file_min_size: 10000 watch_interval 2 Job Attribute 2 Specifies the interval (in seconds) at which the file watcher job will check for the existence and size of the watched-for file. watch_interval: seconds where seconds can be any integer; it represents the time interval between checks of the file existence and file size. Example 2 To set the file to be checked for a steady state every two minutes, enter this: watch_interval: 120