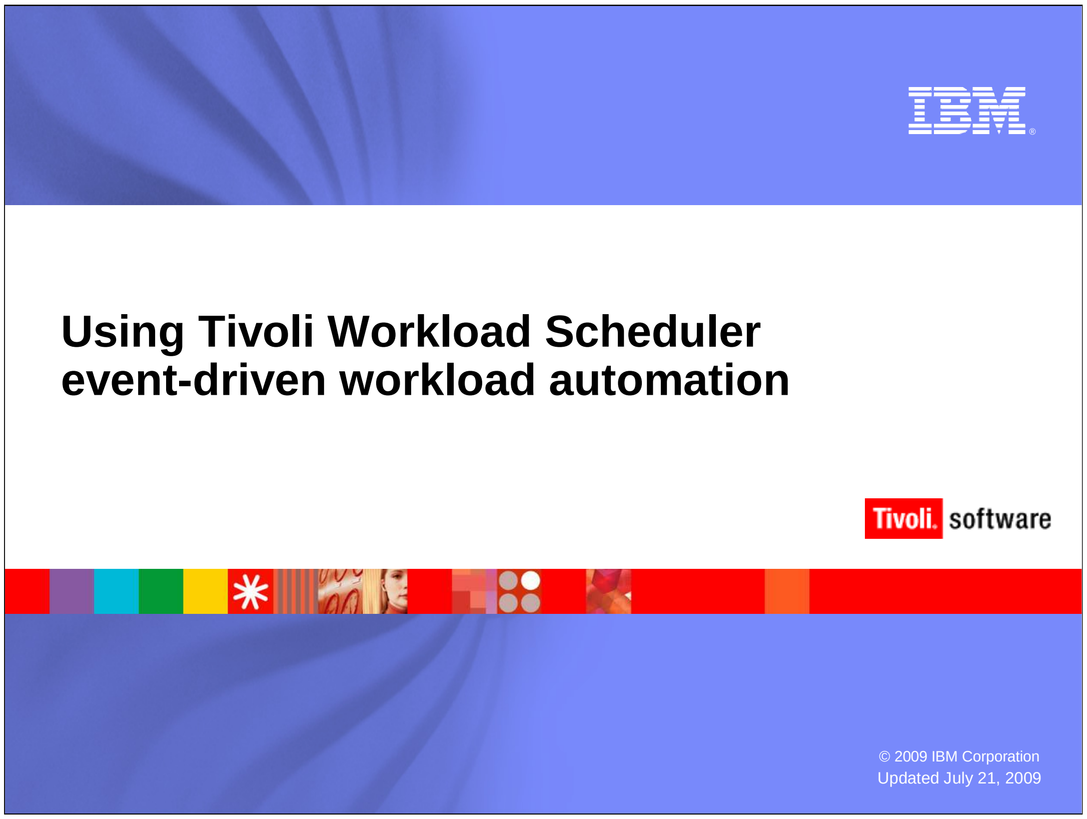
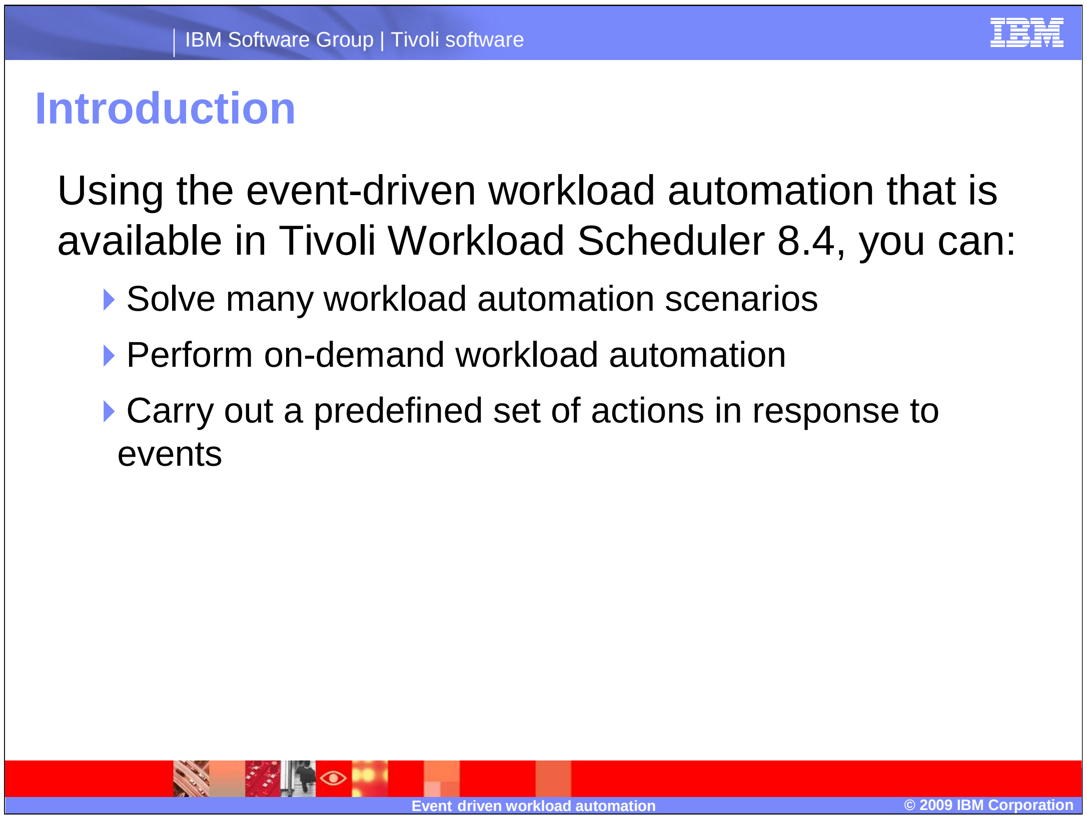


Using Tivoli Workload Scheduler event-driven workload automation

Tivoli software

In this module, you will learn how to use the new Event Driven Workload Automation feature of IBM Tivoli Workload Scheduler.



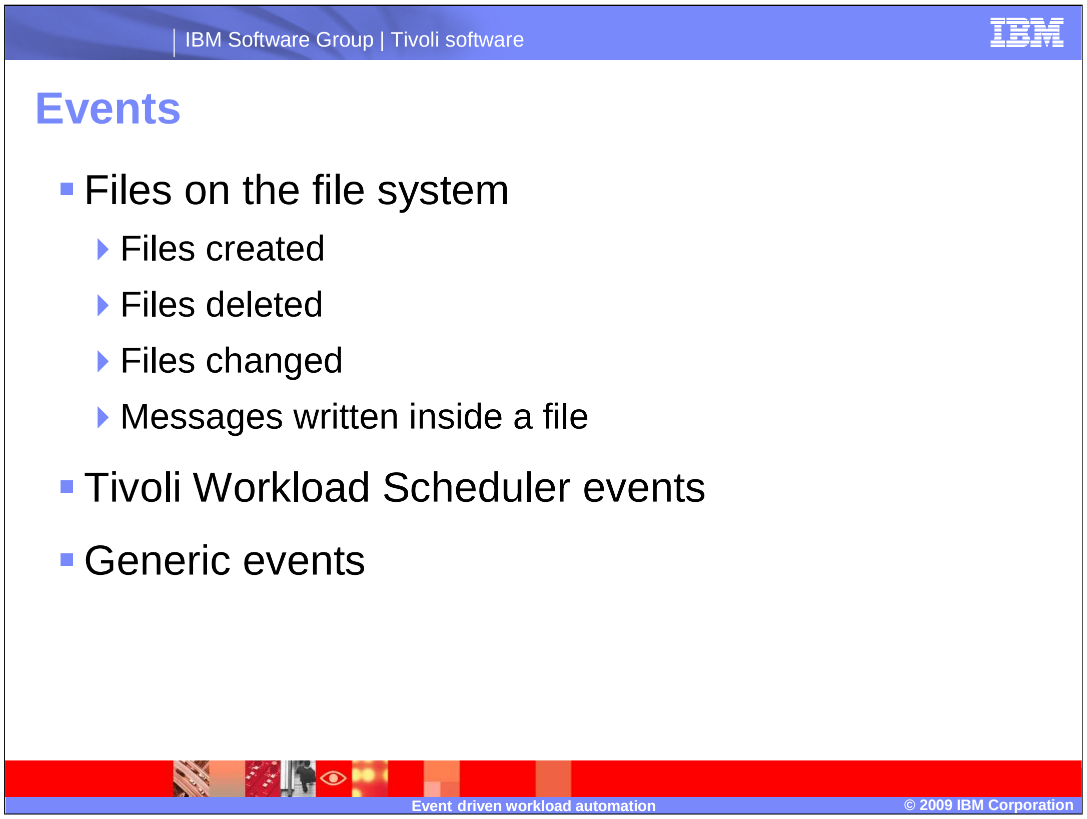
Introduction

Using the event-driven workload automation that is available in Tivoli Workload Scheduler 8.4, you can:

- Solve many workload automation scenarios
- Perform on-demand workload automation
- Carry out a predefined set of actions in response to events

Event-driven workload automation, added in Tivoli Workload Scheduler 8.4, performs on-demand workload automation and plan-based job scheduling. This automation defines rules that can trigger on-demand workload automation.

The object of event-driven workload automation in Tivoli Workload Scheduler is to carry out a predefined set of actions in response to events that occur in the environment. The focus of this session will be to show how you can solve workload automation scenarios by using event driven workload automation.



Events

- Files on the file system
  - Files created
  - Files deleted
  - Files changed
  - Messages written inside a file
- Tivoli Workload Scheduler events
- Generic events

Tivoli Workload Scheduler uses built-in event plug-ins to react to conditions in three areas:

The first area is File system events.

Tivoli Workload Scheduler uses the file system events plug-in to react to changes on the file system of a Workload Scheduler fault tolerant agent. You can pass information about the file system change to event rule actions. Four types of file system events include:

File created, which triggers when a file is created.

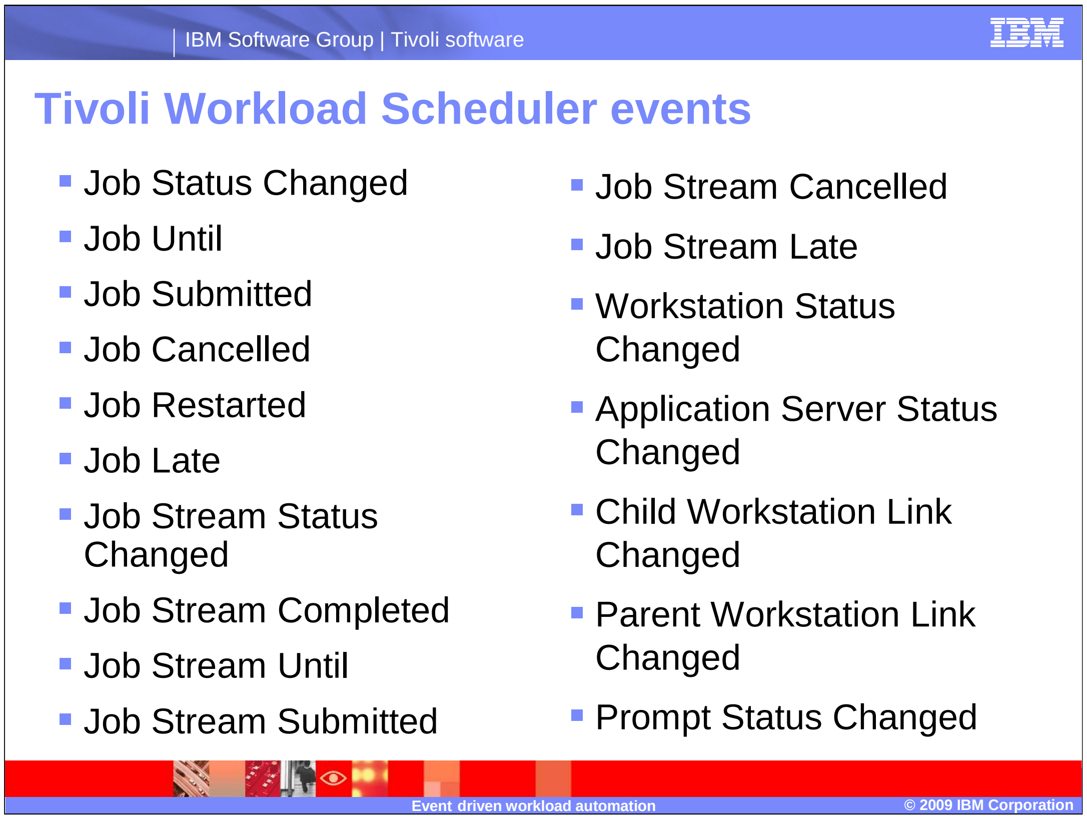
File deleted, which triggers when a file is deleted.

Modification completed, which triggers when a file does not change after two successive checks.

Log message written, which triggers when a log file contains the text you define.

The second area is Tivoli Workload Scheduler events. You can automatically react to changes in the state of objects in a Tivoli Workload Scheduler production plan.

Finally, Generic events are captured and forwarded to Tivoli Workload Scheduler by the send event command. You can use these events for most purposes not covered by the other event types.



Tivoli Workload Scheduler events

- Job Status Changed
- Job Until
- Job Submitted
- Job Cancelled
- Job Restarted
- Job Late
- Job Stream Status Changed
- Job Stream Completed
- Job Stream Until
- Job Stream Submitted
- Job Stream Cancelled
- Job Stream Late
- Workstation Status Changed
- Application Server Status Changed
- Child Workstation Link Changed
- Parent Workstation Link Changed
- Prompt Status Changed

Event rules can detect changes to elements of the Tivoli Workload Scheduler environment in areas shown on the slide.

•Job Status Changed

•Job Until

•Job Submitted

•Job Cancelled

•Job Restarted

•Job Late

•Job Stream Status Changed

•Job Stream Completed

•Job Stream Until

•Job Stream Submitted

•Job Stream Cancelled

•Job Stream Late

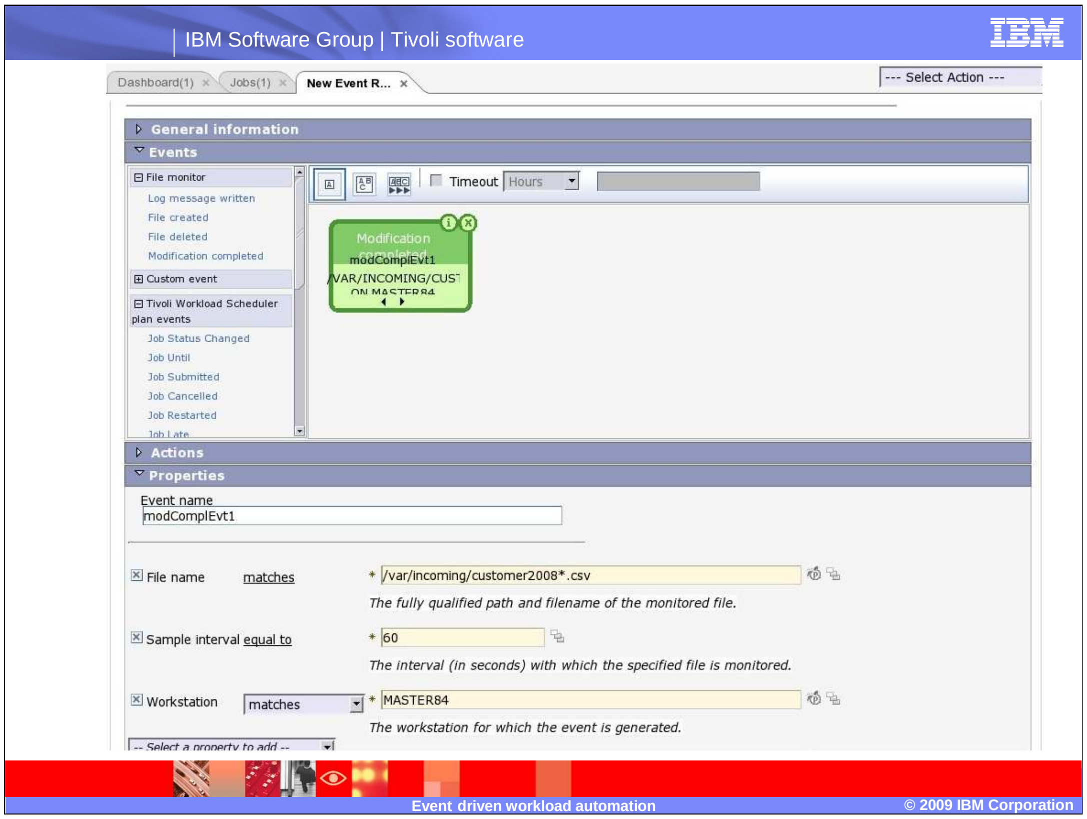
•Workstation Status Changed

•Application Server Status Changed

•Child Workstation Link Changed

•Parent Workstation Link Changed

•Prompt Status Changed


IBM Software Group | Tivoli software

Event name: modComplEvt1

File name matches * /var/incoming/customer2008*.csv

The fully qualified path and filename of the monitored file.

Sample interval equal to * 60

The interval (in seconds) with which the specified file is monitored.

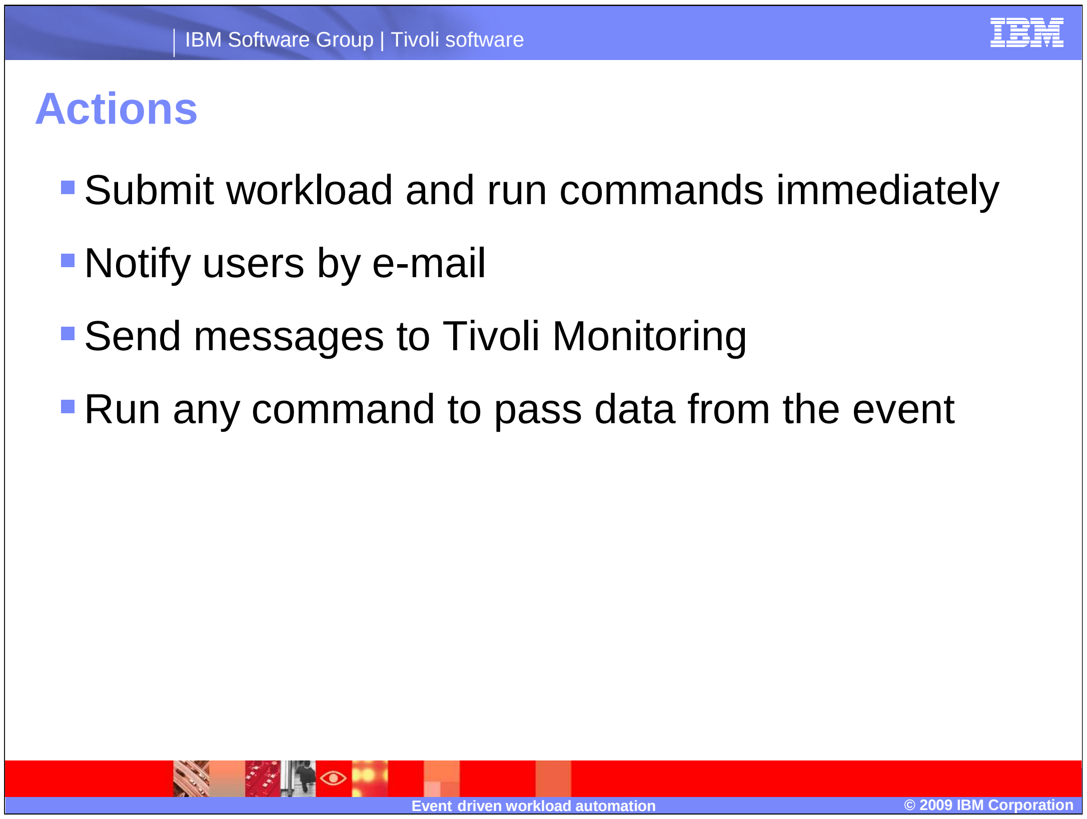
Workstation matches * MASTER84

The workstation for which the event is generated.

-- Select a property to add --

Event driven workload automation © 2009 IBM Corporation

This slide shows the Tivoli Dynamic Workload Console event rule editor. Using the editor, you can graphically design Workload Scheduler event rules to be saved in the Tivoli Workload Scheduler database.

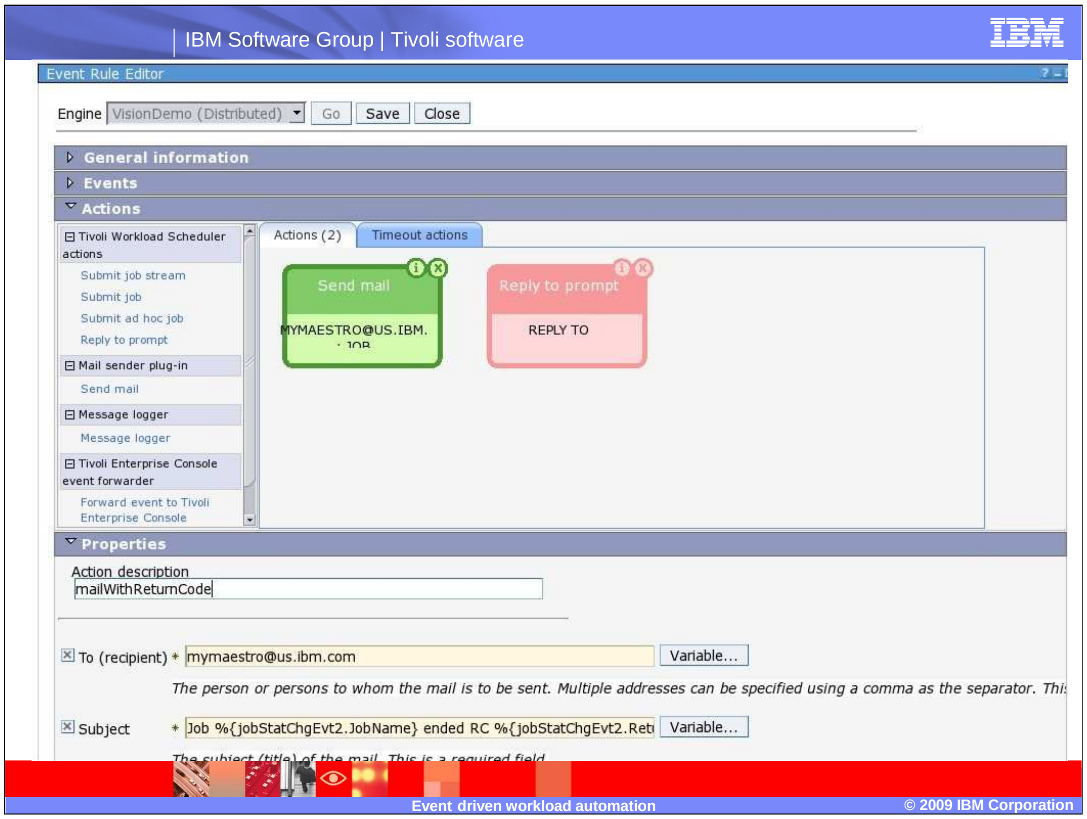


Actions

- Submit workload and run commands immediately
- Notify users by e-mail
- Send messages to Tivoli Monitoring
- Run any command to pass data from the event

Actions include submitting workload and run commands immediately, notifying users by e-mail, or sending messages to Tivoli Monitoring.

An action can also run a generic command.



Action description
mailWithReturnCode

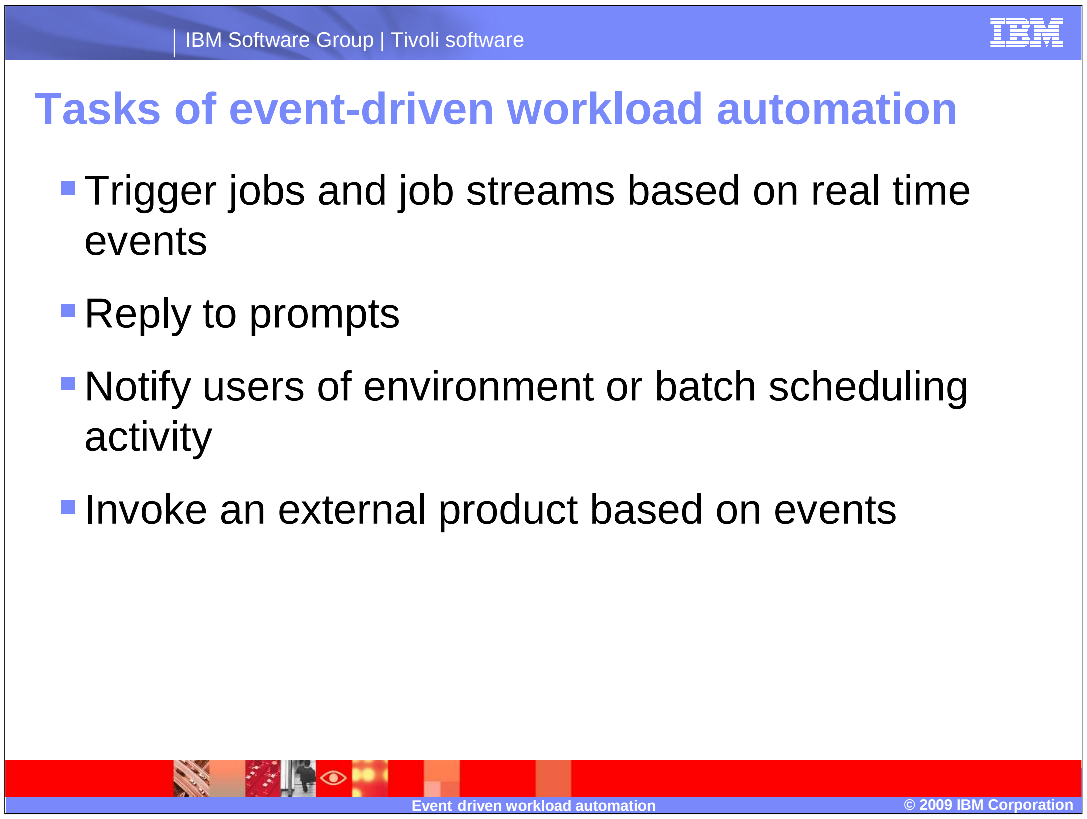
To (recipient) * mymaestro@us.ibm.com

The person or persons to whom the mail is to be sent. Multiple addresses can be specified using a comma as the separator. Thi

Subject * Job %{jobStatChgEvt2.JobName} ended RC %{jobStatChgEvt2.Ret

The subject (title) of the mail. This is a required field

Here you can see the Dynamic Workload Console event rule editor, showing the Actions panel with an e-mail action highlighted.



Tasks of event-driven workload automation

- Trigger jobs and job streams based on real time events
- Reply to prompts
- Notify users of environment or batch scheduling activity
- Invoke an external product based on events

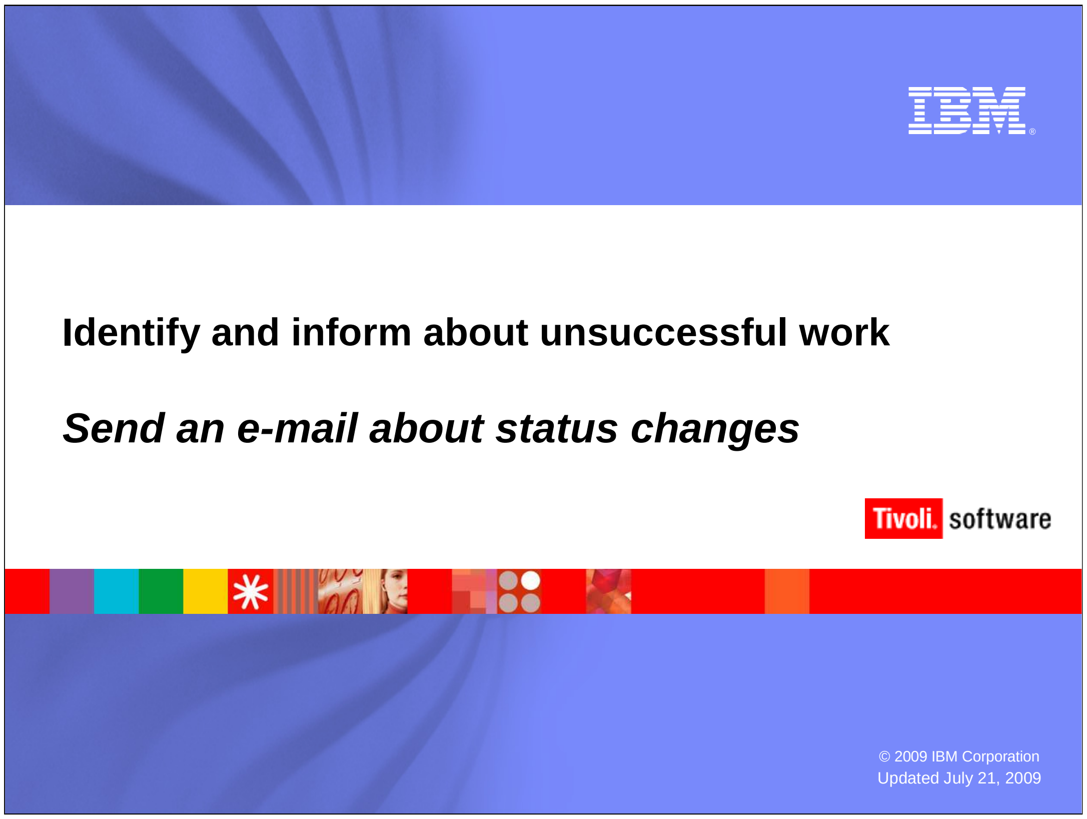
The main tasks of event-driven workload automation are:

One, to trigger the execution of batch jobs and job streams based on the reception or combination of real time events.

Two, to reply to prompts

Three, to notify users when anomalous conditions occur in the Tivoli Workload Scheduler scheduling environment or batch scheduling activity, and

Four, to invoke an external product when a particular event condition occurs.

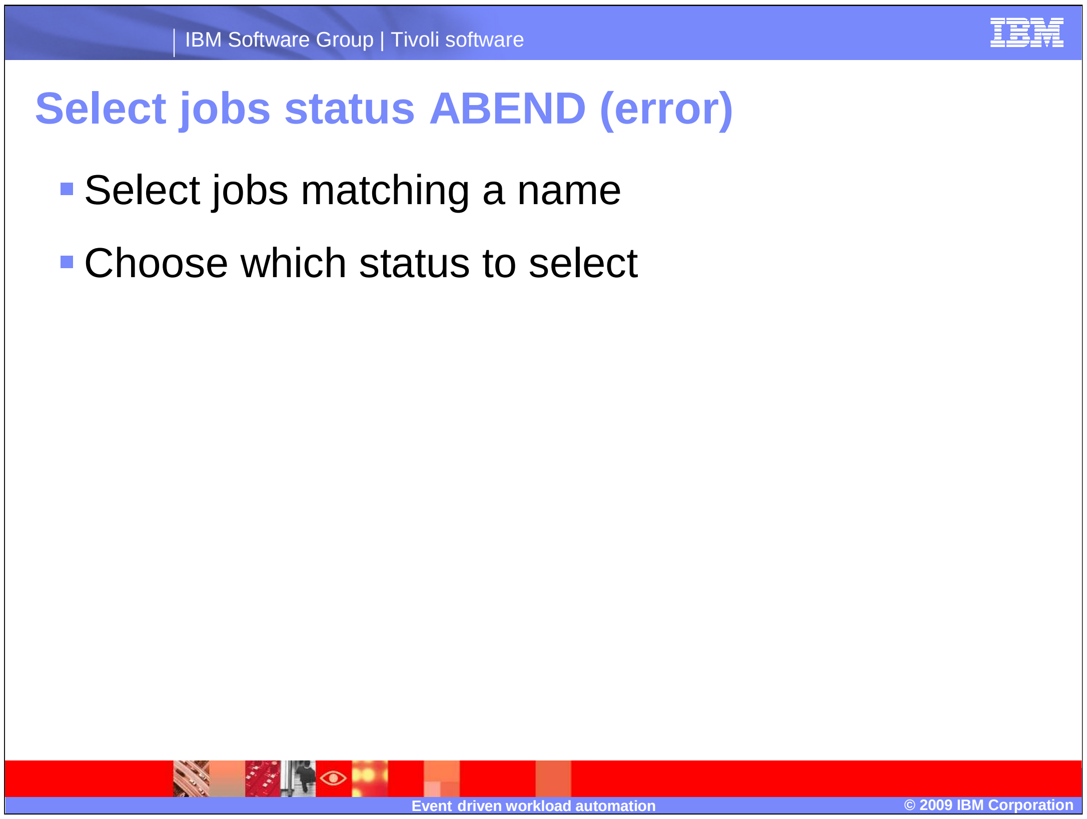


Identify and inform about unsuccessful work

Send an e-mail about status changes

Tivoli. software

Now take a look at some examples of event rule scenarios. With a simple event rule, you can send an e-mail containing information about work that changes status.

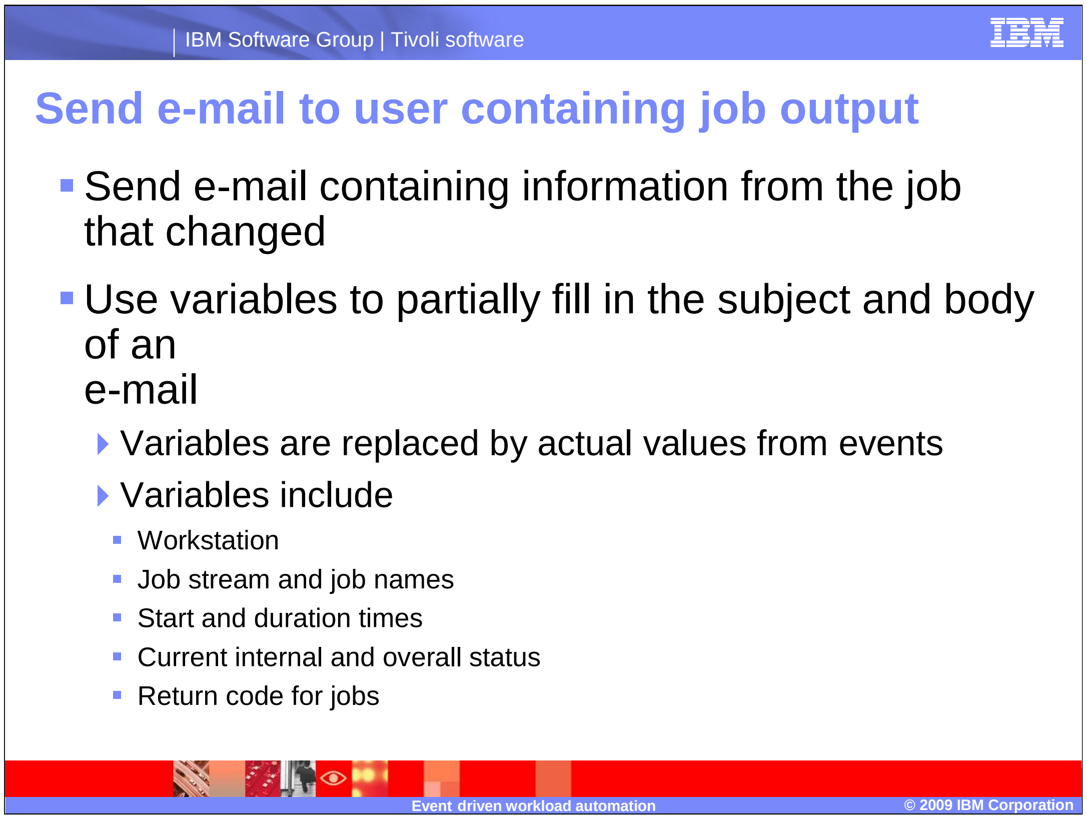


Select jobs status ABEND (error)

- Select jobs matching a name
- Choose which status to select

To build the notification event rule, select jobs matching a particular naming convention and an error status such as ABEND.

The first part of this event rule is selecting which work and which status or statuses will cause the event to be triggered. Here you might select some jobs or job streams based on their names and select the states that will cause the trigger.

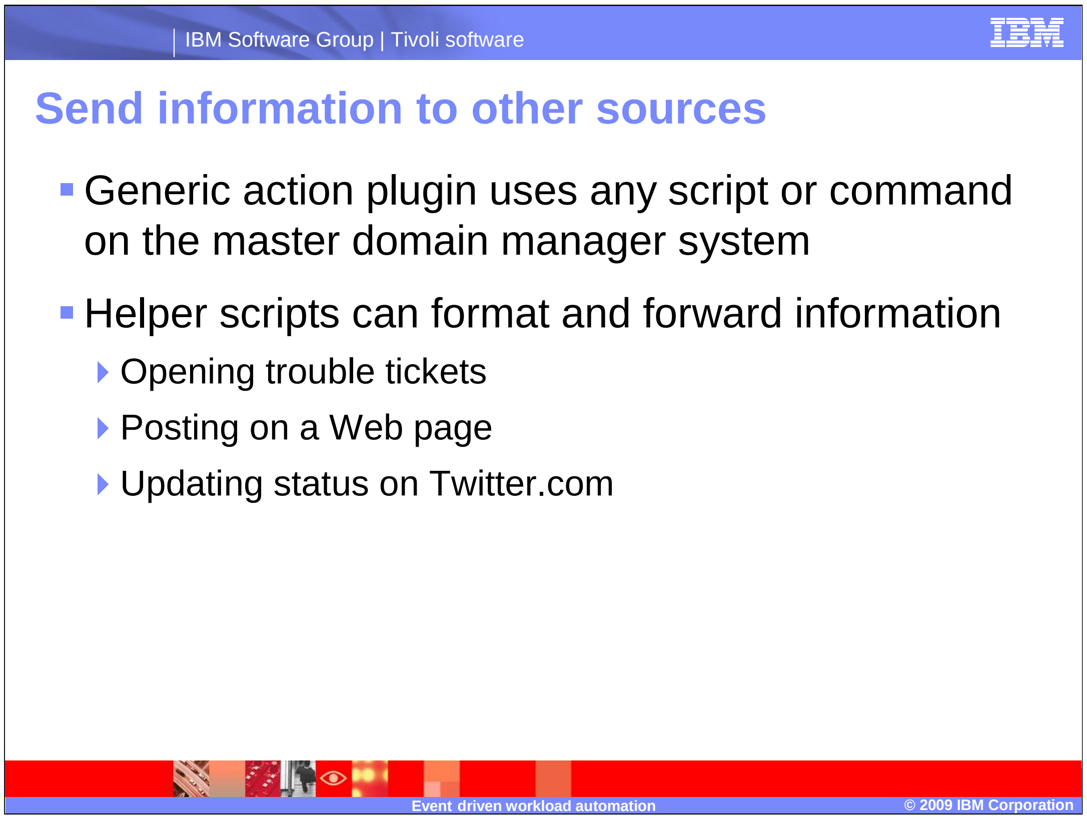


Send e-mail to user containing job output

- Send e-mail containing information from the job that changed
- Use variables to partially fill in the subject and body of an
  e-mail
  - Variables are replaced by actual values from events
  - Variables include
    - Workstation
    - Job stream and job names
    - Start and duration times
    - Current internal and overall status
    - Return code for jobs

Using one of the built-in action functions, you can send an e-mail containing information from the job that changed. Selecting the e-mail action and filling in the properties will complete this event rule.

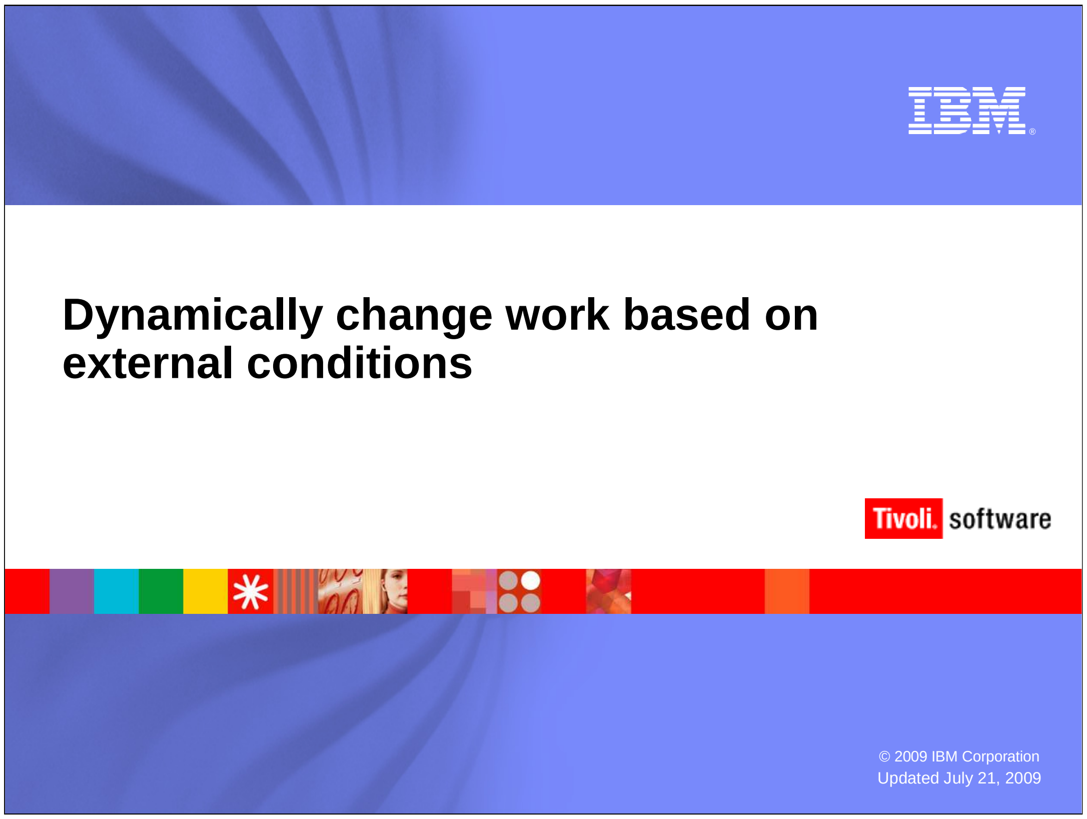
You can use variables to partially fill in the subject and body of the e-mail. These variables, which you can see by clicking the Variables button in the editor, are replaced by actual values from the event that triggered the action at the time of the trigger. Variables include the workstation, job stream, and job names, the start and duration times, the current internal and overall status, and the return code for jobs.



Send information to other sources

- Generic action plugin uses any script or command on the master domain manager system
- Helper scripts can format and forward information
  - Opening trouble tickets
  - Posting on a Web page
  - Updating status on Twitter.com

Using the Generic action plug-in, you can send information externally by using any script or command on the master domain manager system. For example, a small helper script can take information and format it for opening trouble tickets posted on a Web page or for updating your status on Twitter.

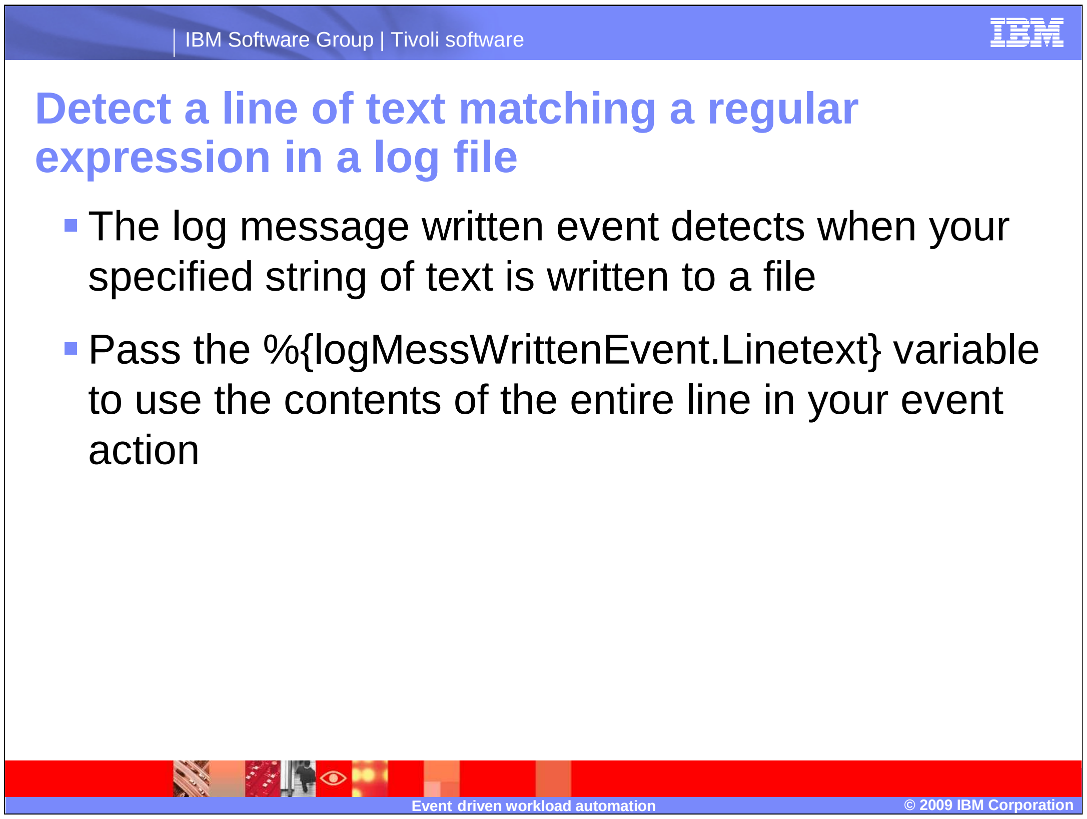


Dynamically change work based on external conditions

Tivoli. software

© 2009 IBM Corporation
Updated July 21, 2009

Using the event triggers and the variables provided by them, you can dynamically change the way work runs in your environment.

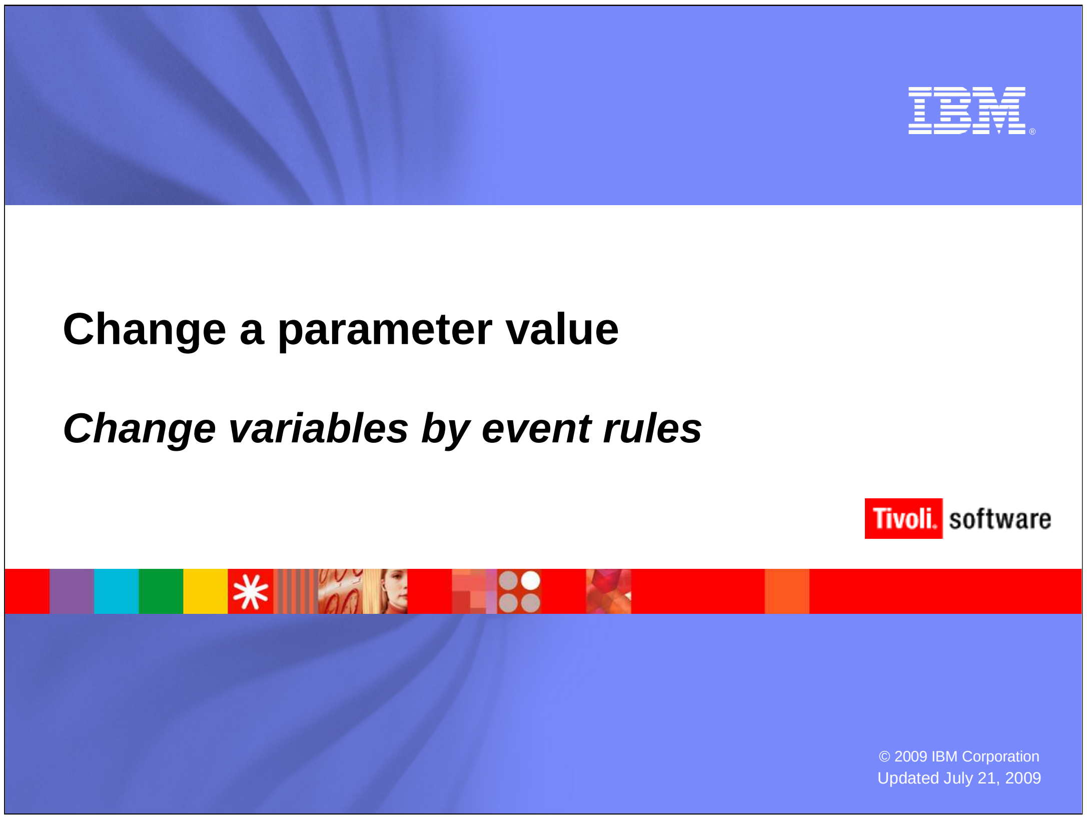


Detect a line of text matching a regular expression in a log file

- The log message written event detects when your specified string of text is written to a file
- Pass the %{logMessWrittenEvent.Linetext} variable to use the contents of the entire line in your event action

One of the events in an event rule is the Log message written event. This event detects when your specified string of text is written into a log file you select. You can use the contents of the entire line in your event action.

The **log Message Written Event dot Line text** variable contains the entire line of text detected by the log message written event of the event rule. You can pass this variable to your event action so that the action can run in context of the log message event that caused it to be triggered.



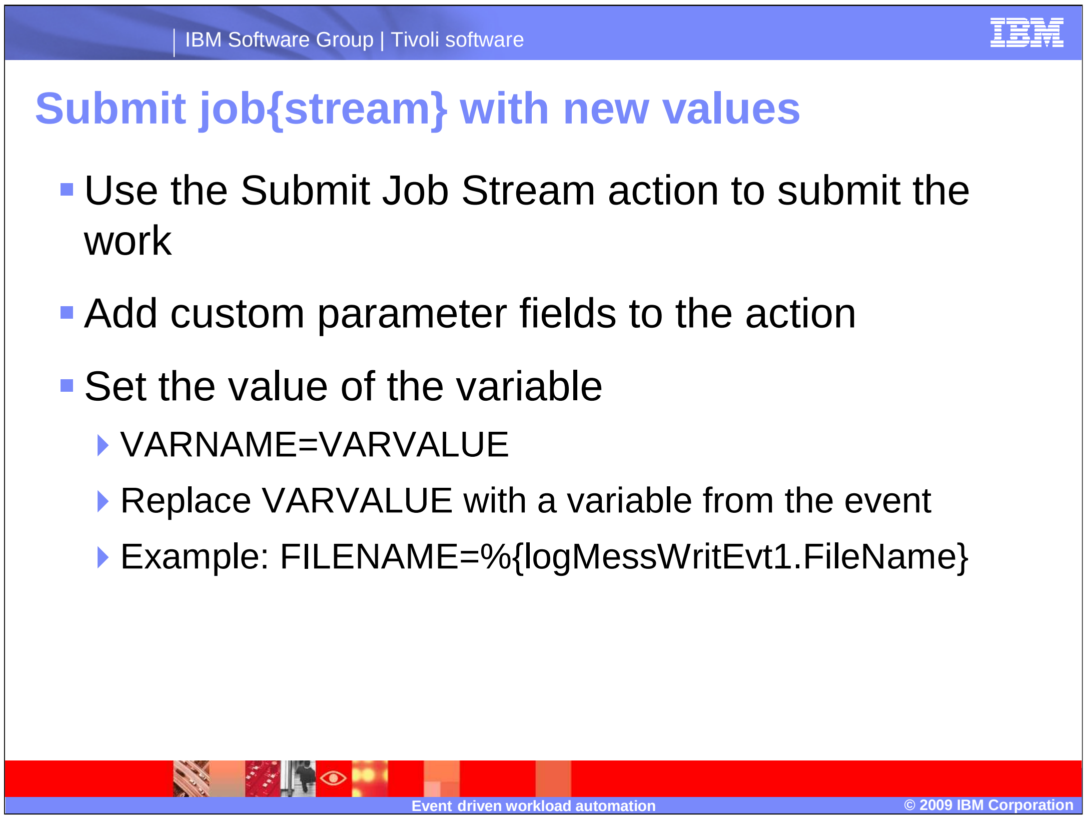
Change a parameter value

Change variables by event rules

Tivoli. software

© 2009 IBM Corporation
Updated July 21, 2009

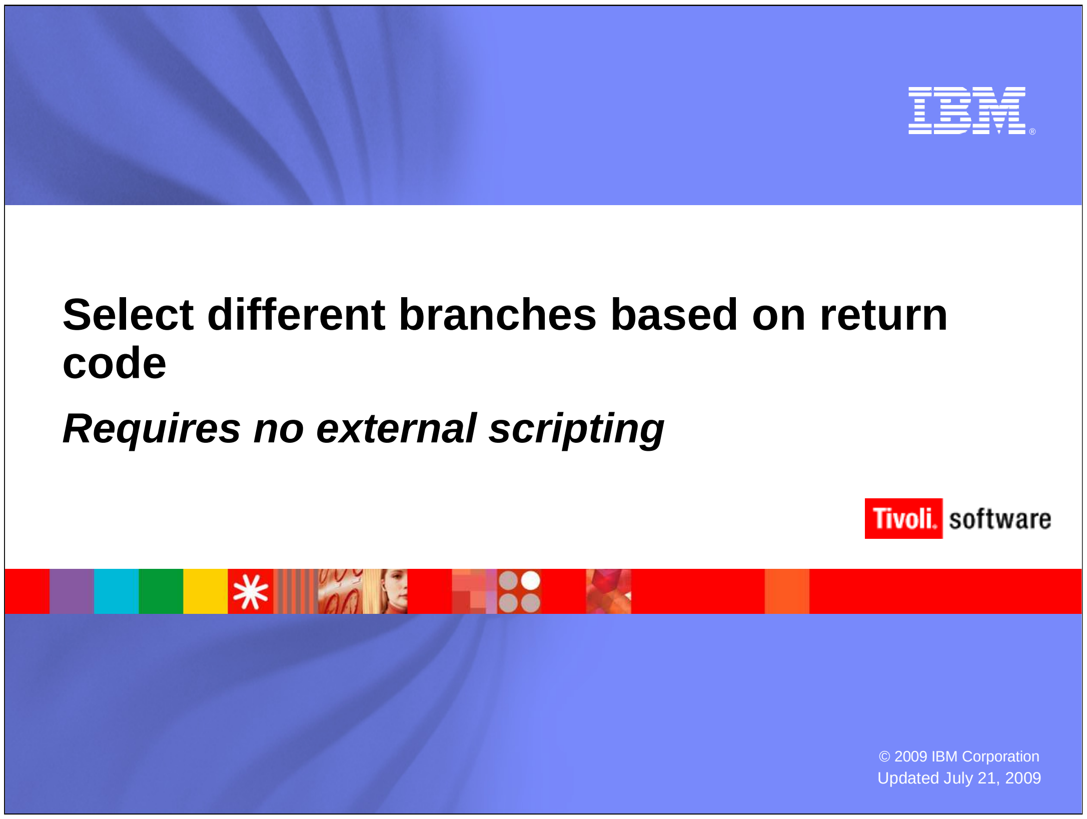
Variables used as part of job definitions can be changed immediately by event rules. These variables take their values from events in the event rule.



Submit job{stream} with new values

- Use the Submit Job Stream action to submit the work
- Add custom parameter fields to the action
- Set the value of the variable
  - VARNAME=VARVALUE
  - Replace VARVALUE with a variable from the event
  - Example: FILENAME=%{logMessWritEvt1.FileName}

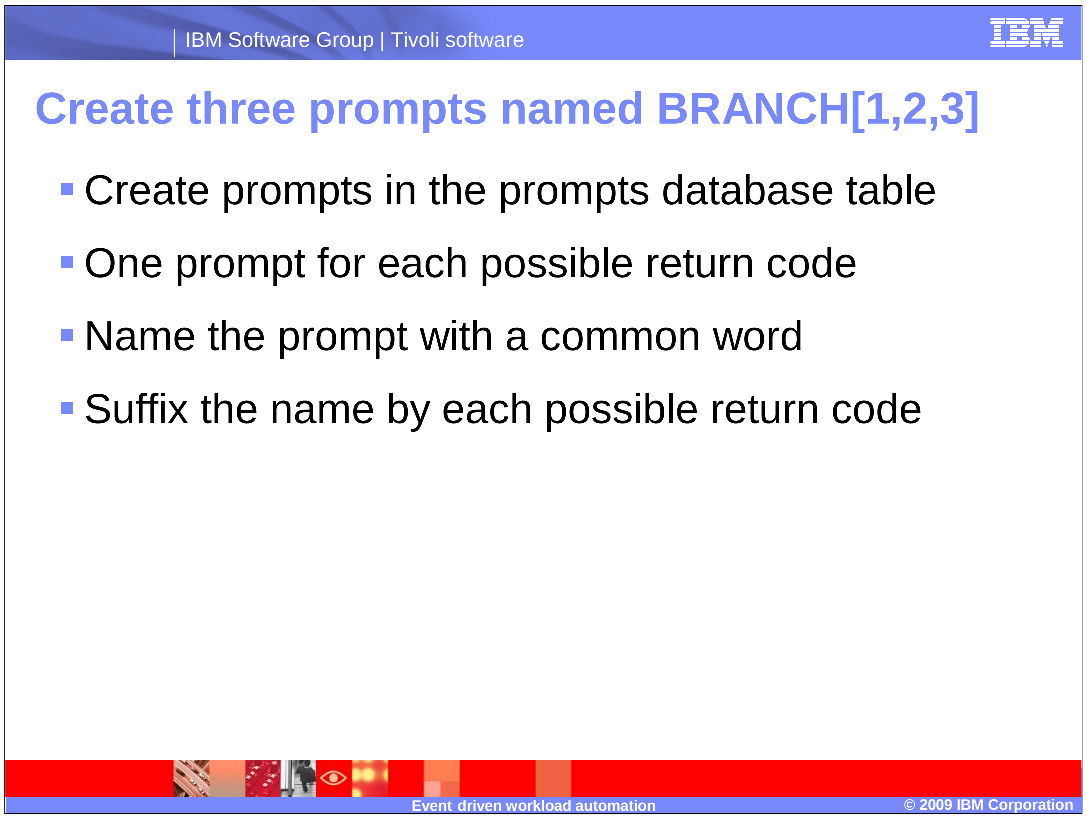
Use the Submit Job Stream action to submit the work. Add Custom parameter fields to the action and set the value of the variable by specifying its name and value in the format VARNAME=VARVALUE. Replace VARVALUE with a variable from the event. For example, to set a jobs **File name** variable to the name of a file that has been written, use FILENAME=log Message Written Event 1 dot File Name.



Select different branches based on return code

Requires no external scripting

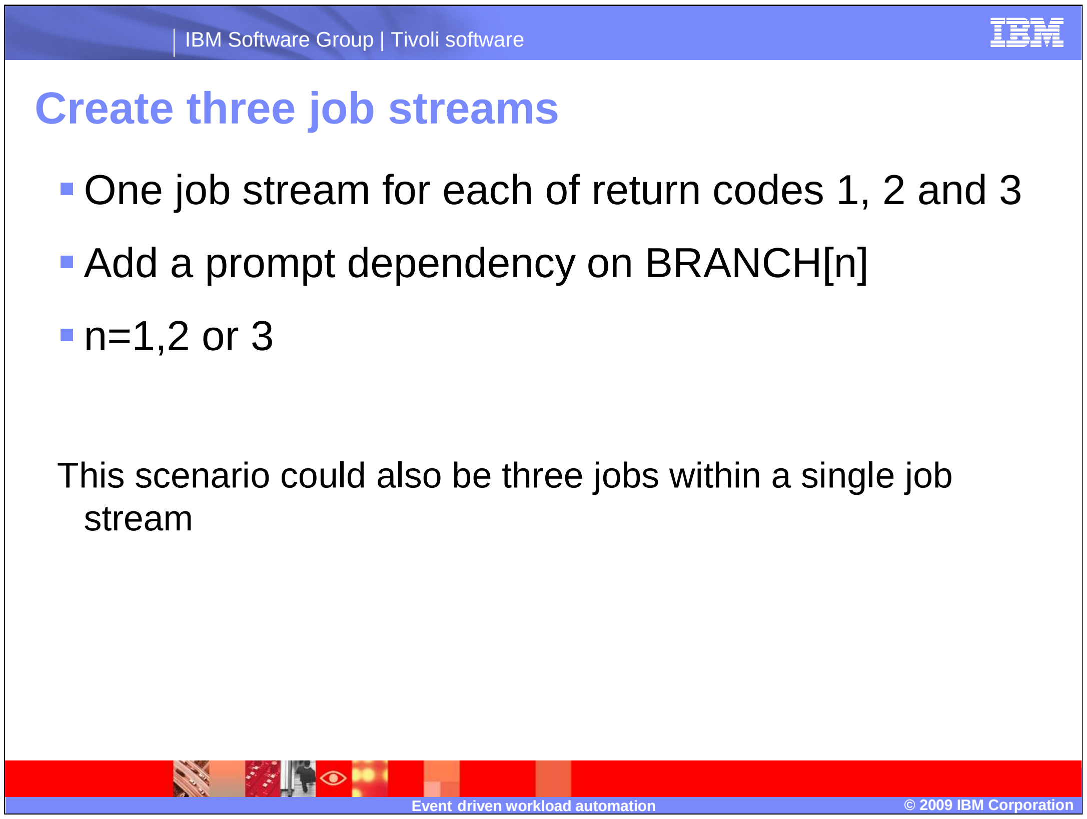
Sometimes different jobs need to be executed depending on decisions made by other jobs that are already running as part of the production plan. You can use one of several methods to perform this scenario; this one requires no external scripting or coding.



Create three prompts named BRANCH[1,2,3]

- Create prompts in the prompts database table
- One prompt for each possible return code
- Name the prompt with a common word
- Suffix the name by each possible return code

You will need some prompts in the prompts database table, one for each possible return code. The name of the prompt is a common word, suffixed by the possible return codes of the decision-making job. Here, for example, we use the word *Branch* followed by the numbers one, two, three.



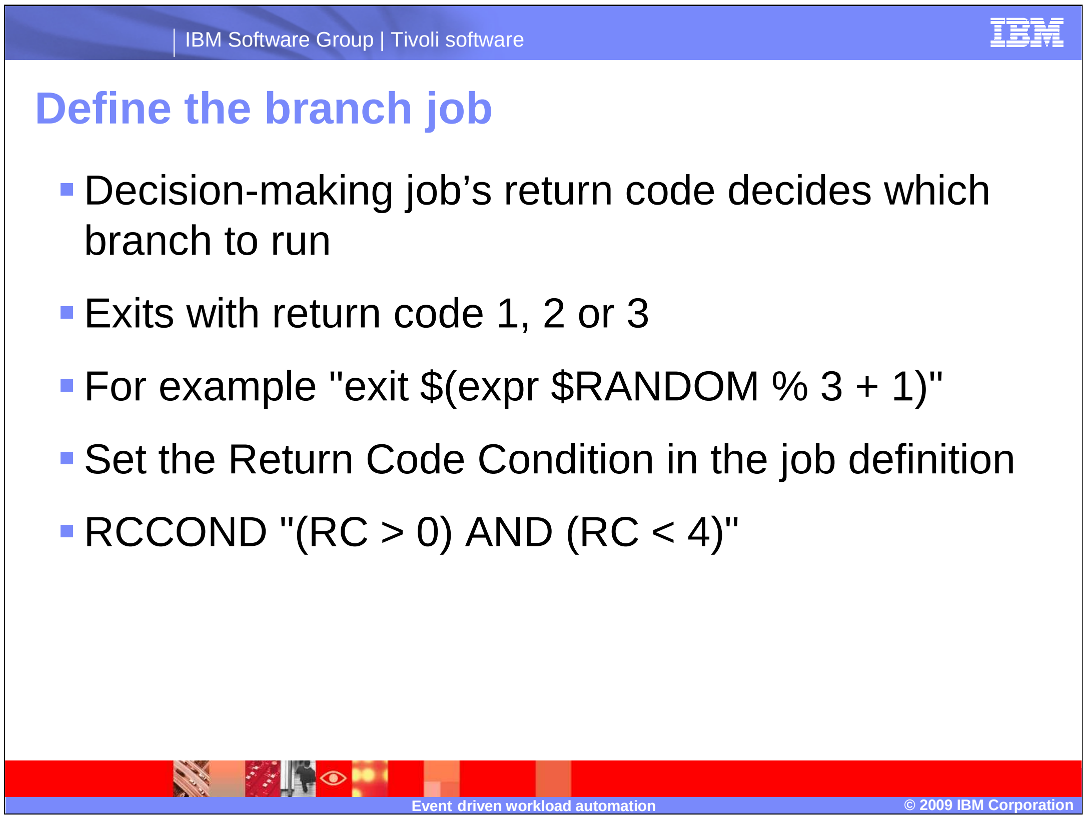
Create three job streams

- One job stream for each of return codes 1, 2 and 3
- Add a prompt dependency on BRANCH[n]
- n=1,2 or 3

This scenario could also be three jobs within a single job stream

In this scenario we have three possibilities, one for each of return codes 1, 2 and 3. Add to each one a prompt dependency on BRANCH 1, BRANCH 2, or BRANCH 3.

For each of the three job streams, add a single prompt dependency. The prompt is the BRANCH prompt created earlier. The number is the return code of the decision-making job, in this case, one of BRANCH 1, BRANCH 2 or BRANCH 3.



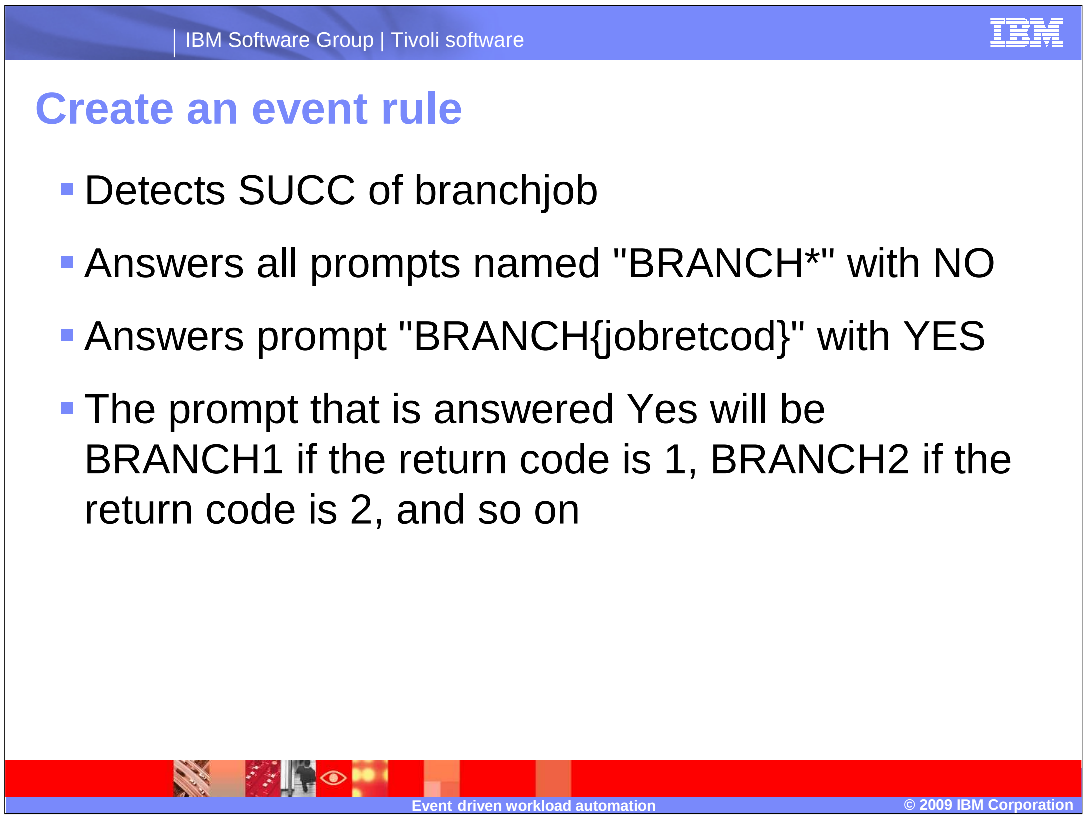
Define the branch job

- Decision-making job’s return code decides which branch to run
- Exits with return code 1, 2 or 3
- For example "exit $(expr $RANDOM % 3 + 1)"
- Set the Return Code Condition in the job definition
- RCCOND "(RC > 0) AND (RC < 4)"

The decision-making job, or branch job, is the one having the return code that will later be associated with the branch to run.
This particular branch job exits with return code of 1, 2 or 3.

If you want to simulate a branch job in your test environment, you can create a job that exits with a random return code between one and three inclusively.

The return codes 1, 2 or 3 will represent a successful job. You need to set the Return Code Condition on this job. Use an RC CONDition specifying return codes greater than zero and less than four to signify a successful return.



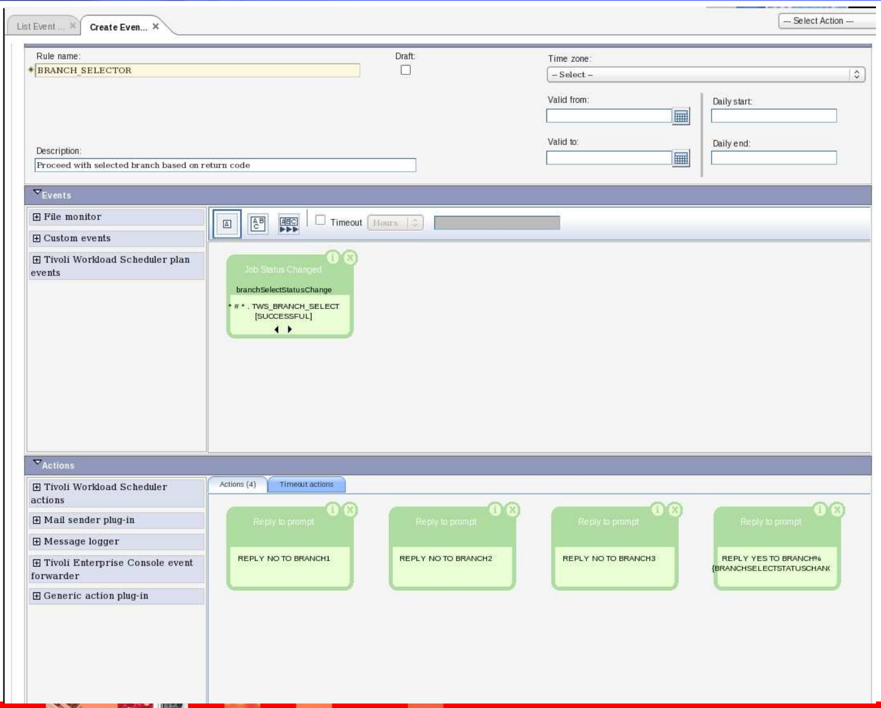
Create an event rule

- Detects SUCC of branchjob
- Answers all prompts named "BRANCH*" with NO
- Answers prompt "BRANCH{jobretcod}" with YES
- The prompt that is answered Yes will be BRANCH1 if the return code is 1, BRANCH2 if the return code is 2, and so on

Now that you have defined the decision-maker and the work to be run, you can create the event rule that makes everything happen.
The event, or trigger, in the event rule will be the successful completion of the branch job. Remember that return codes 1, 2 or 3 are success conditions as defined in the job.

The first action in the event rule answers all prompts starting with BRANCH with a *No* reply. This way operators can see that the prompts have been handled, and that they do not remain in the ASKED state.

The second action in the event rule answers a single prompt with a *Yes* reply, allowing the dependent job stream to run. In the action definition, choose the Answer a prompt action. In the name field, enter BRANCH, followed immediately by a variable. Click the variables button and select the Return code option. Now, the name of the prompt that is answered *Yes* will be BRANCH1 if the return code is 1, BRANCH2 if the return code is 2, and so on.



Rule name: BRANCH_SELECTOR

Draft: ☐

Time zone: – Select –

Valid from:

Daily start:

Valid to:

Daily end:

Description: Proceed with selected branch based on return code

Job Status Changed
branchSelectStatusChange
* # * . TWS_BRANCH_SELECT [SUCCESSFUL]

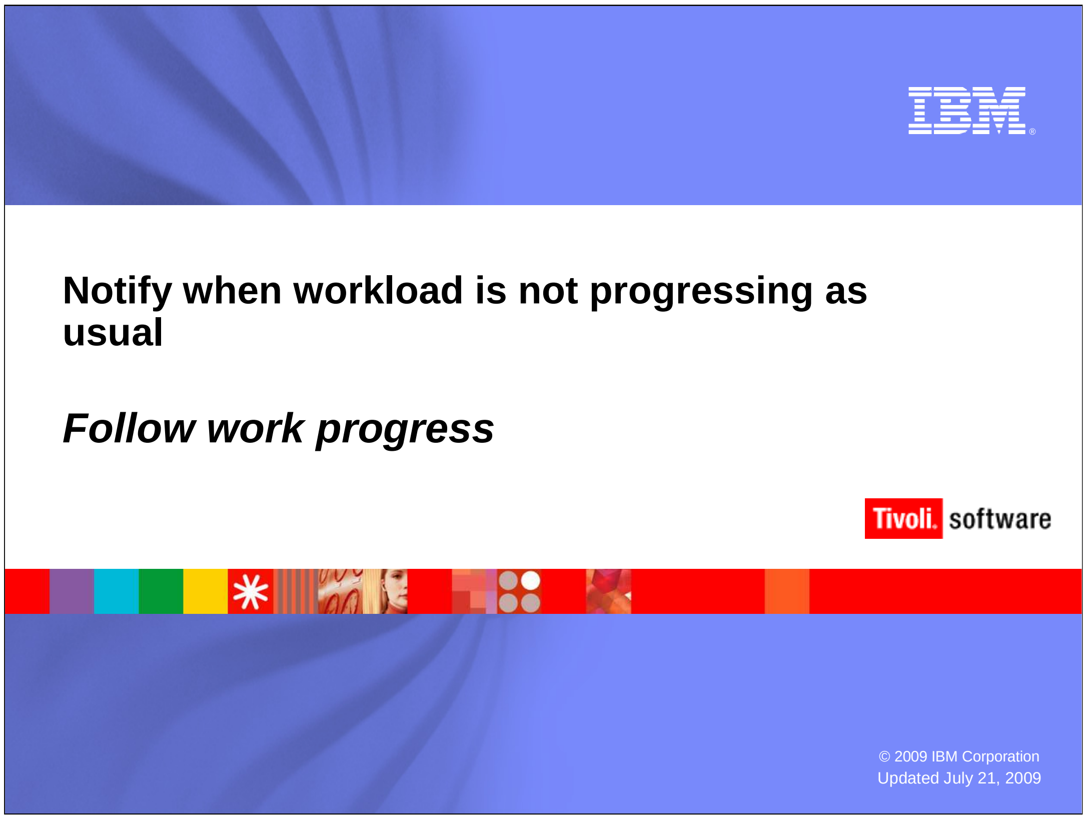
Reply to prompt
REPLY NO TO BRANCH1

Reply to prompt
REPLY NO TO BRANCH2

Reply to prompt
REPLY NO TO BRANCH3

Reply to prompt
REPLY YES TO BRANCH%{BRANCHSELECTSTATUSCHAN

Here is an example of a branch job event rule in the Dynamic Workload Console event rule editor.

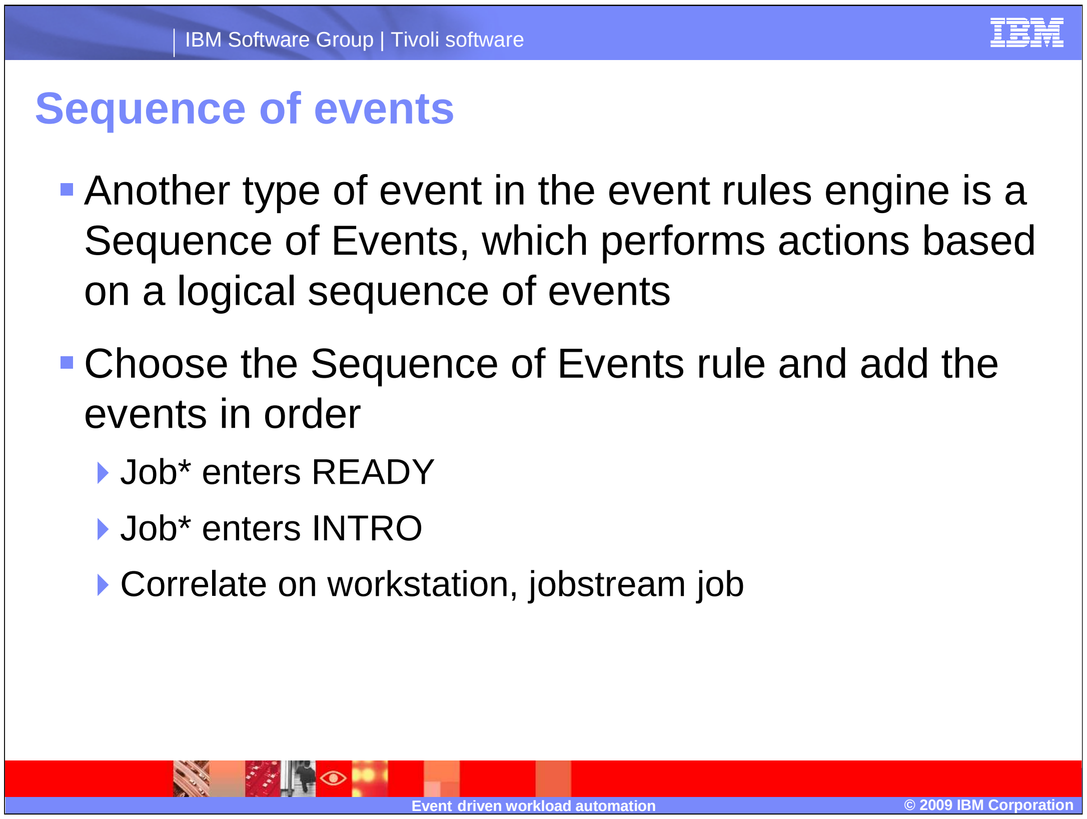


Notify when workload is not progressing as usual

Follow work progress

Tivoli. software

An event rule can be written to follow the progress of a workload through its change of status and notify or perform other actions when the change ends at an unexpected state.


IBM Software Group | Tivoli software

Sequence of events

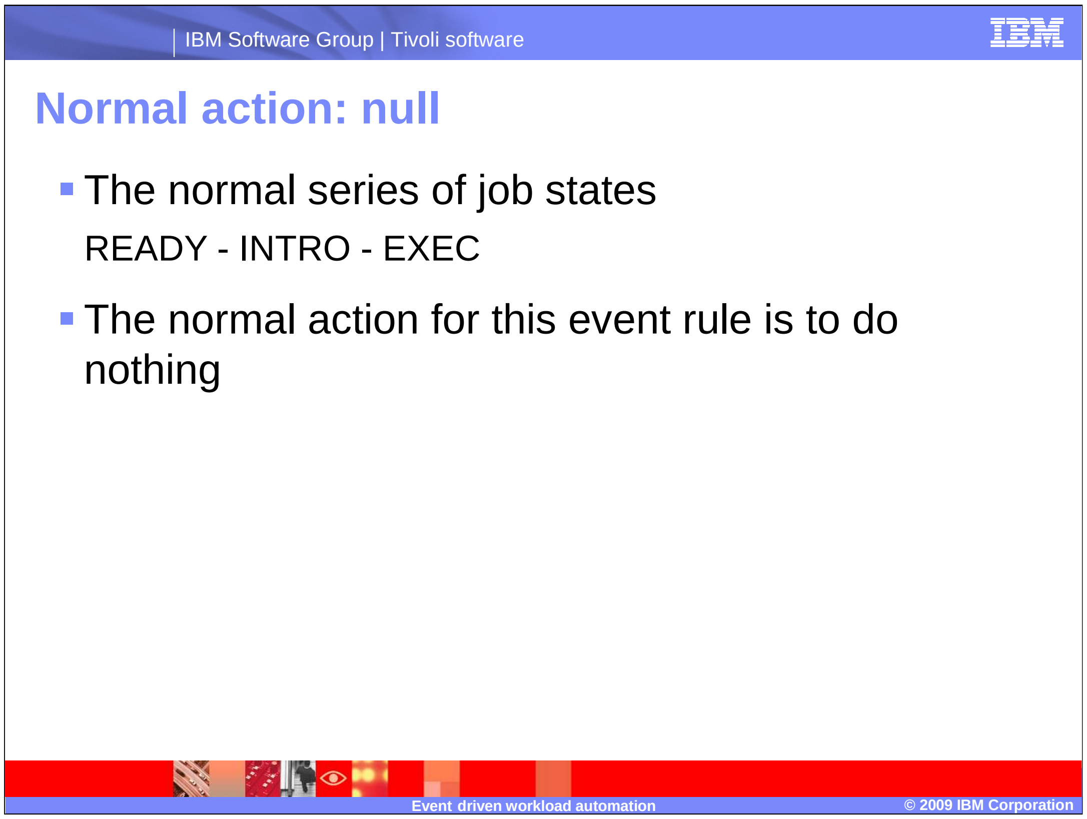
- Another type of event in the event rules engine is a Sequence of Events, which performs actions based on a logical sequence of events
- Choose the Sequence of Events rule and add the events in order
  - Job* enters READY
  - Job* enters INTRO
  - Correlate on workstation, jobstream job

Event driven workload automation © 2009 IBM Corporation

Another type of event in the event rules engine is a Sequence of Events. Using the sequence, actions are based on a logical sequence of events happening in the Tivoli Workload Scheduler environment. For this event rule you take advantage of the knowledge of normal TWS job processing to form a notification.

Choose the Sequence of Events rule and add the events in the following order.

The first event is the one that detects when a job enters the READY state. This state means all job dependencies have been satisfied, and the job should start to execute soon. After a job is READY, it enters the INTRO state, meaning the job is being prepared to execute. The second action should detect jobs that have the INTRO state.
Make sure the two actions are correlated on Workstation, Job Stream, and Job to be certain that the sequence of events always applies to the same job within each sequence.



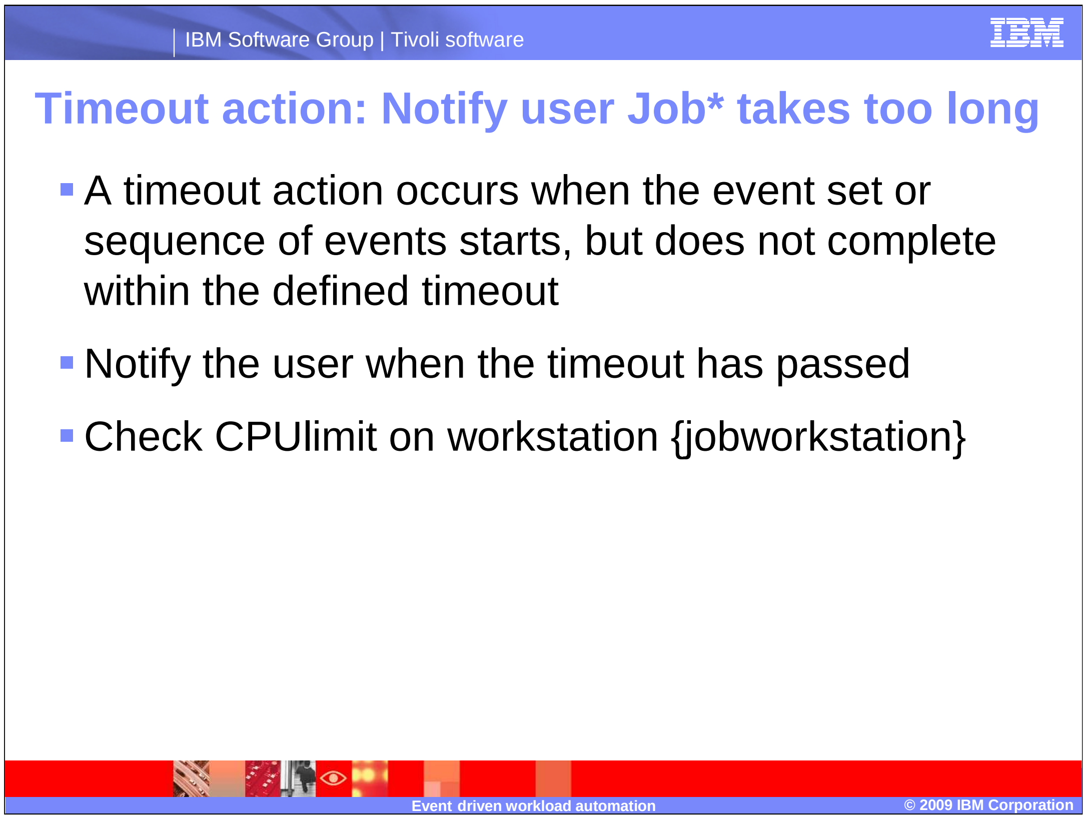
Normal action: null

- The normal series of job states

  READY - INTRO - EXEC

- The normal action for this event rule is to do nothing

Since the normal series job states for any job is READY, followed by INTRO, followed by EXEC, the normal action for this event rule is to do nothing.

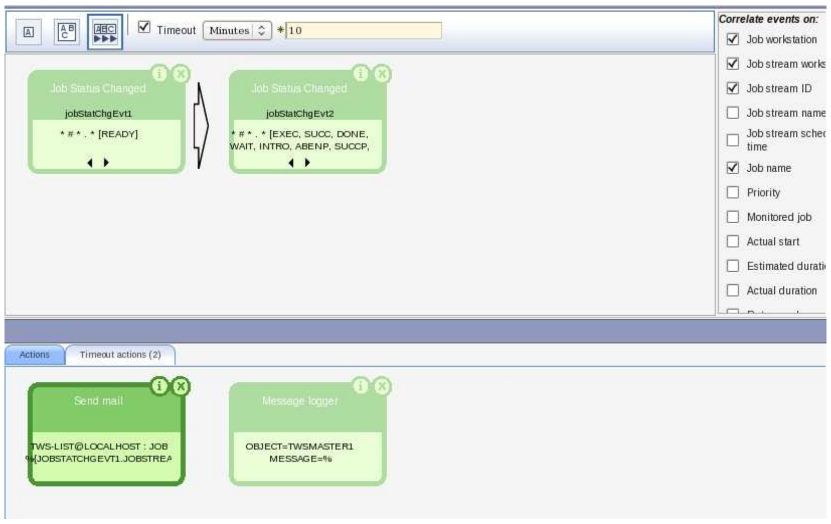


Timeout action: Notify user Job* takes too long

- A timeout action occurs when the event set or sequence of events starts, but does not complete within the defined timeout
- Notify the user when the timeout has passed
- Check CPUlimit on workstation {jobworkstation}

For this event rule we use the timeout action. In a timeout action, the action occurs only when the event set or sequence of events starts but does not complete within the defined timeout. In this event rule you want to notify (send an e-mail) to the user when the timeout has passed. The notification might include information that a job has been READY, probably because the Workstation limit is too low.

Ready too long event



☑ Timeout Minutes * 10

Correlate events on:

☑ Job workstation
☑ Job stream works
☑ Job stream ID
☐ Job stream name
☐ Job stream sched time
☑ Job name
☐ Priority
☐ Monitored job
☐ Actual start
☐ Estimated durati
☐ Actual duration

Job Status Changed
jobStatChgEvt1
* # * . * [READY]

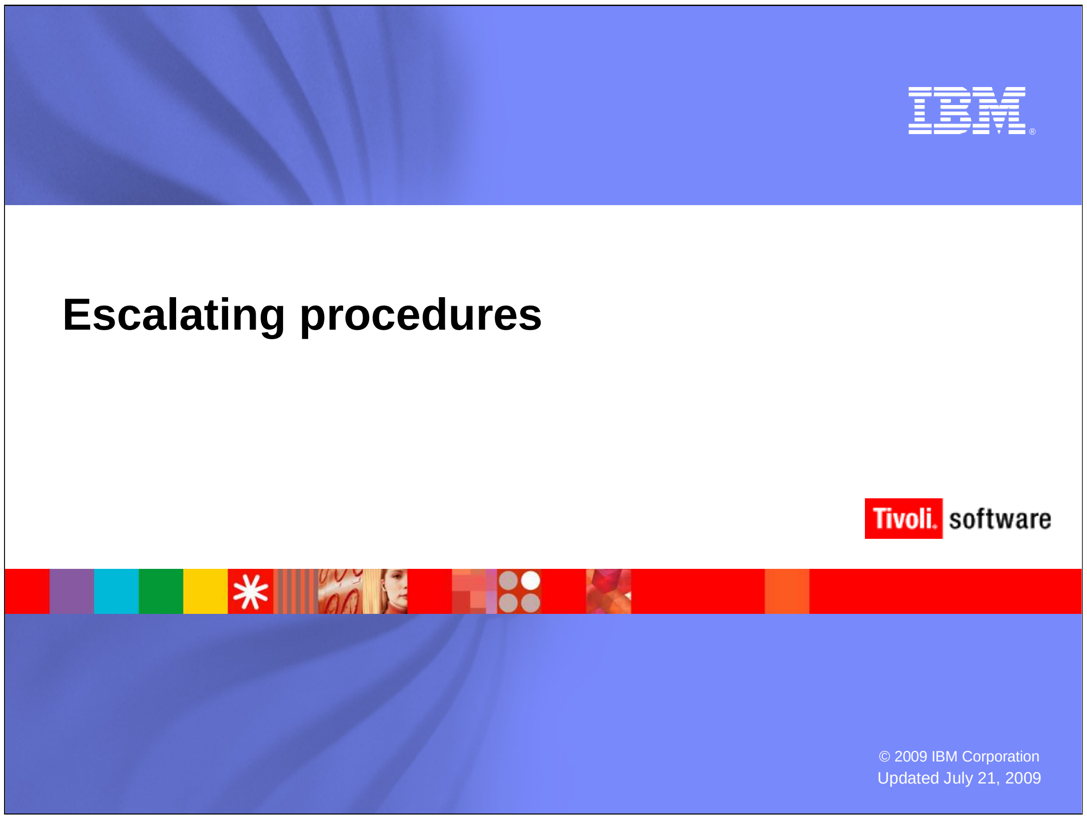
Job Status Changed
jobStatChgEvt2
* # * . * [EXEC, SUCC, DONE, WAIT, INTRO, ABENP, SUCCP,

Actions | Timeout actions (2)

Send mail
TWS-LIST@LOCALHOST : JOB %{JOBSTATCHGEVT1.JOBSTREA

Message logger
OBJECT=TWSMASTER1
MESSAGE=%

This example illustrates a ready too long event rule in the event rule editor on the Workload Console.

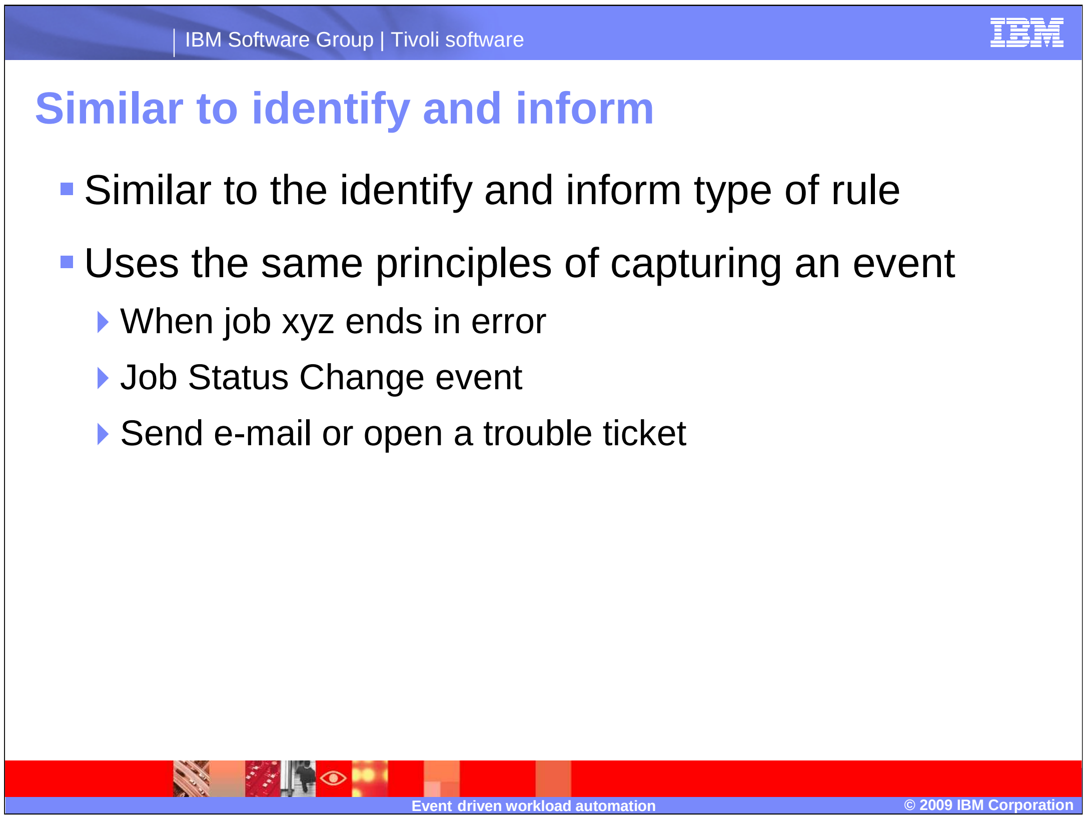


Escalating procedures

Tivoli. software

© 2009 IBM Corporation
Updated July 21, 2009

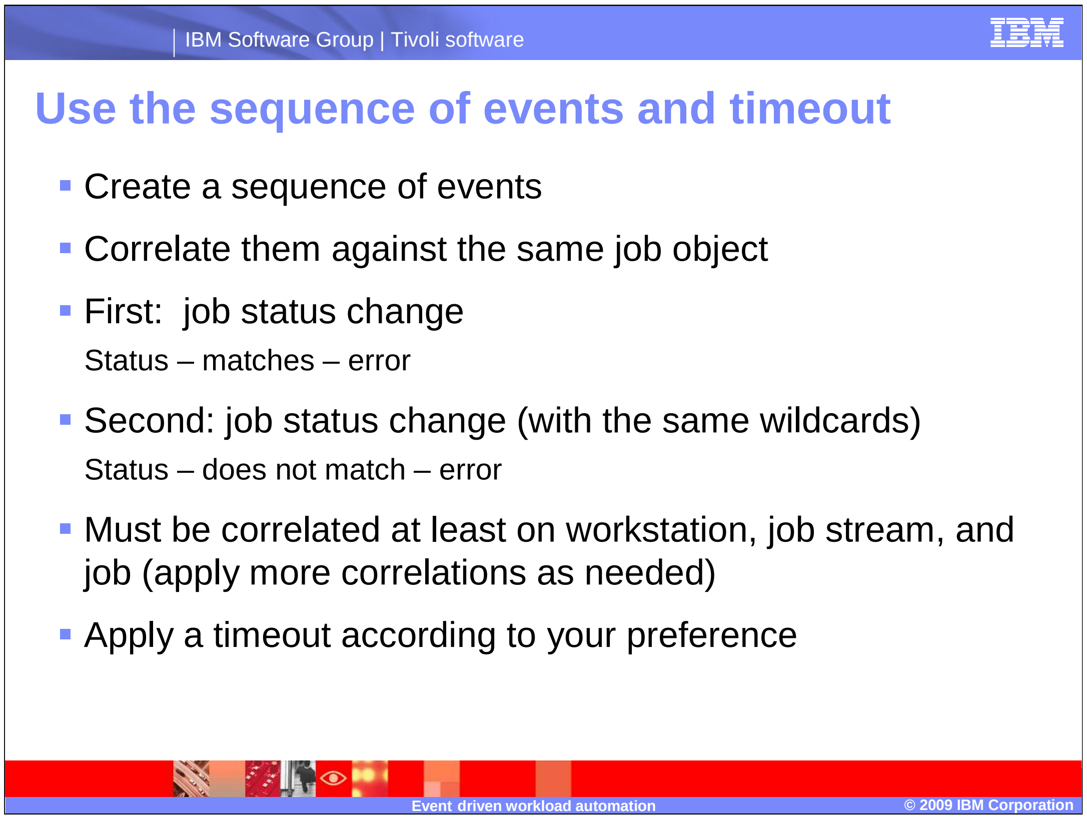
You can use the sequence of events with the timeout action features in event rules to make an escalated procedures rule.



Similar to identify and inform

- Similar to the identify and inform type of rule
- Uses the same principles of capturing an event
  - When job xyz ends in error
  - Job Status Change event
  - Send e-mail or open a trouble ticket

The escalation rule is similar to the identify and inform rule. This escalation rule uses the same principles of capturing an event, such as when job xyz ends in error with a Job Status Change event, and sending an e-mail or opening a trouble ticket.



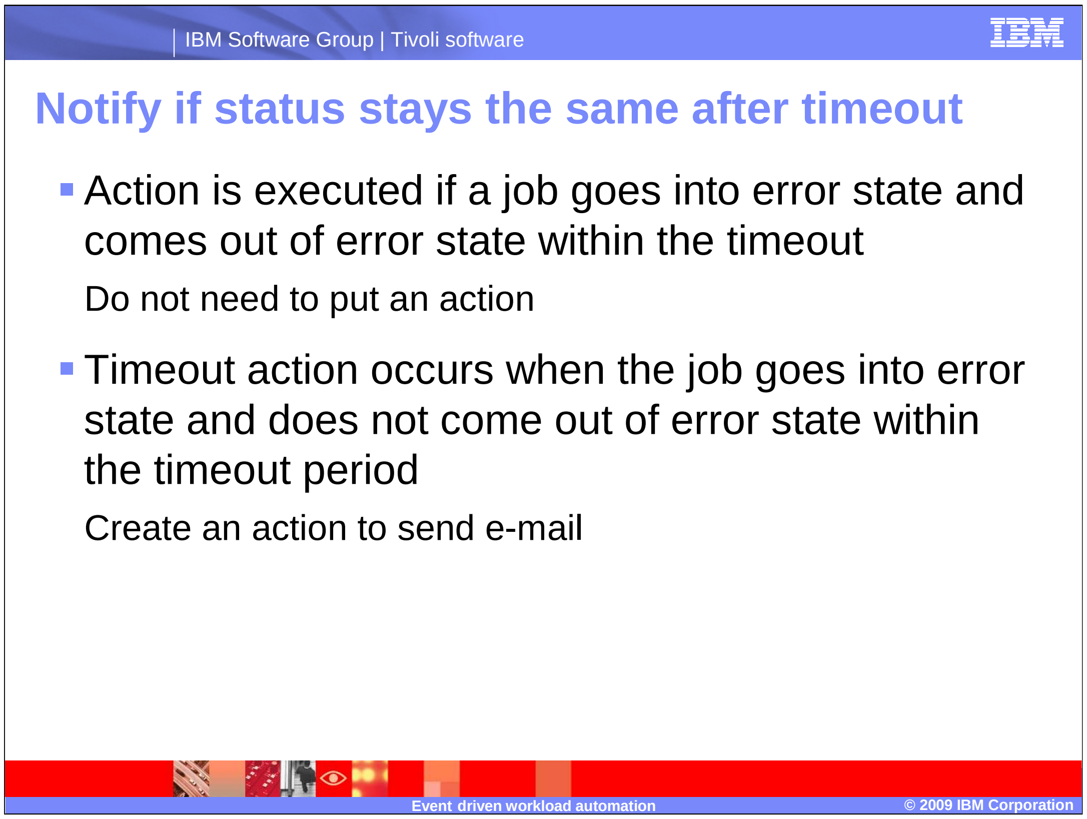
Use the sequence of events and timeout

- Create a sequence of events
- Correlate them against the same job object
- First: job status change

  Status – matches – error
- Second: job status change (with the same wildcards)

  Status – does not match – error
- Must be correlated at least on workstation, job stream, and job (apply more correlations as needed)
- Apply a timeout according to your preference

Instead of using a single event, such as when job xyz ends in error with a Job Status Change event, you can create a sequence of events and correlate them against the same job object. The first event is a job status change representing a job in error or status matches Error. The second event is also a job status change (with the same wildcards) but representing the same job not in error, with status does not match Error. The two events must be correlated at least on the workstation, job stream, and job. You can apply more correlations as needed. Apply a timeout according to your preference.



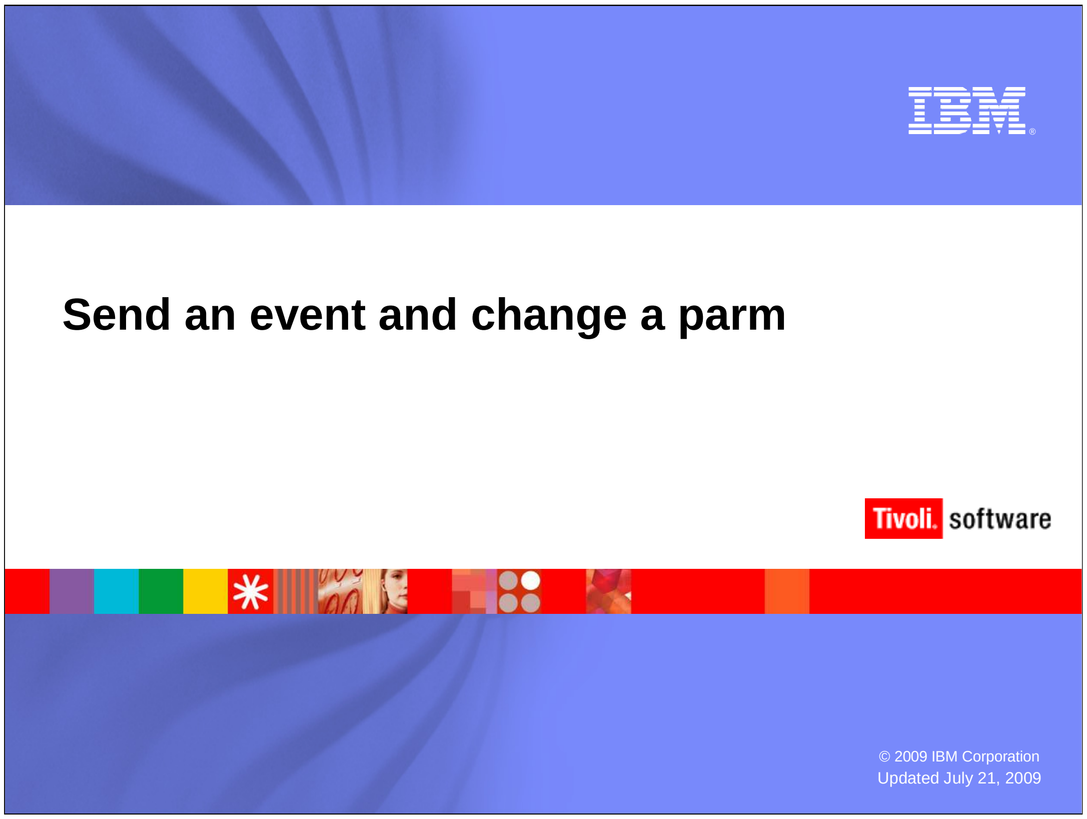
Notify if status stays the same after timeout

- Action is executed if a job goes into error state and comes out of error state within the timeout

  Do not need to put an action
- Timeout action occurs when the job goes into error state and does not come out of error state within the timeout period

  Create an action to send e-mail

The action for this event rule is executed if a job goes into error state and comes out of error state within the timeout. You can choose not to put an action. The timeout action, on the other hand, occurs when the job goes into error state and does not come out of error state within the timeout period. Here you can create an action to send e-mail or perform some other function.

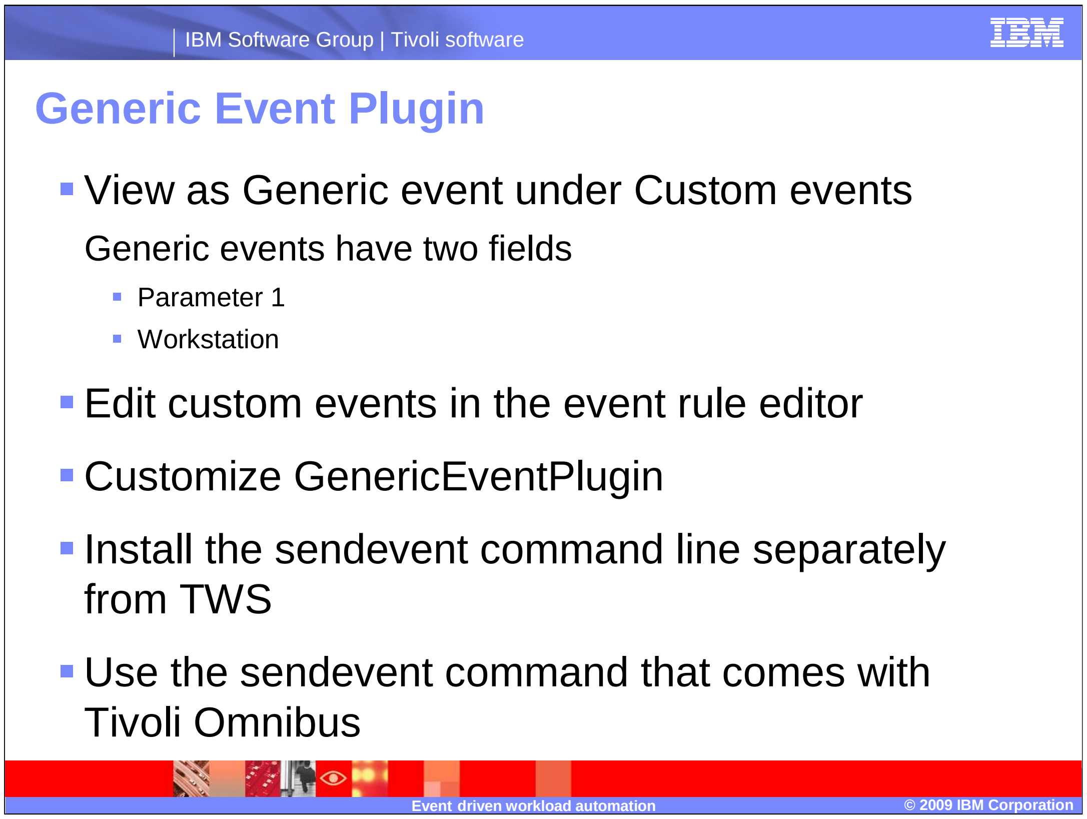


Send an event and change a parm

Tivoli. software

© 2009 IBM Corporation
Updated July 21, 2009

The event-driven workload automation engine also has a few command-line utilities that are shared with other Tivoli products. Tivoli Workload Scheduler contains, in addition to the file monitor and Tivoli Workload Scheduler events, a generic event plug-in. This plug-in is fed information from the send event command. You will use sendevent to relay information to TWS jobs through the parms utility.



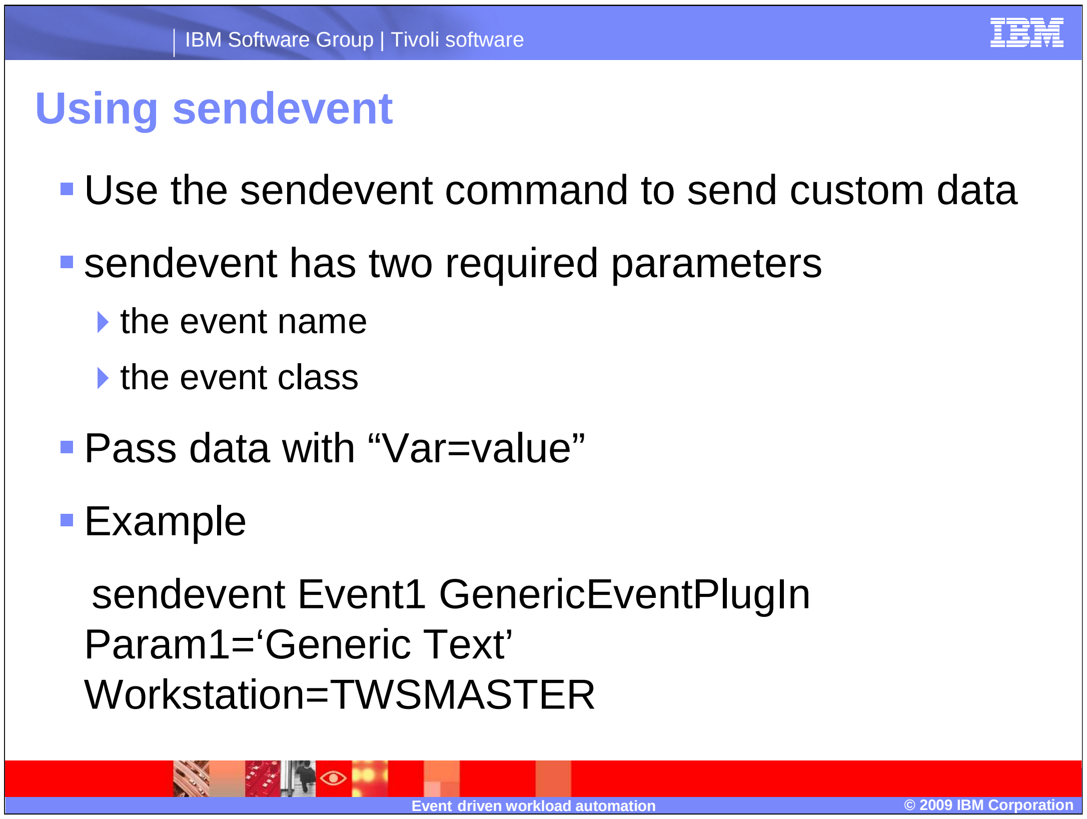
Generic Event Plugin

- View as Generic event under Custom events

  Generic events have two fields
  - Parameter 1
  - Workstation
- Edit custom events in the event rule editor
- Customize GenericEventPlugin
- Install the sendevent command line separately from TWS
- Use the sendevent command that comes with Tivoli Omnibus

In the event rule editor, you can see the Generic event under Custom events. In the generic event, there are two fields: Parameter 1 and Workstation. You will fill in those fields with data later.

You can customize the GenericEventPlugin to provide other data to the event-driven automation scenario. You can install the sendevent command line separately from TWS or use the sendevent command that comes with Tivoli Omnibus. Customization is part of another detailed session and outside the scope of this session.

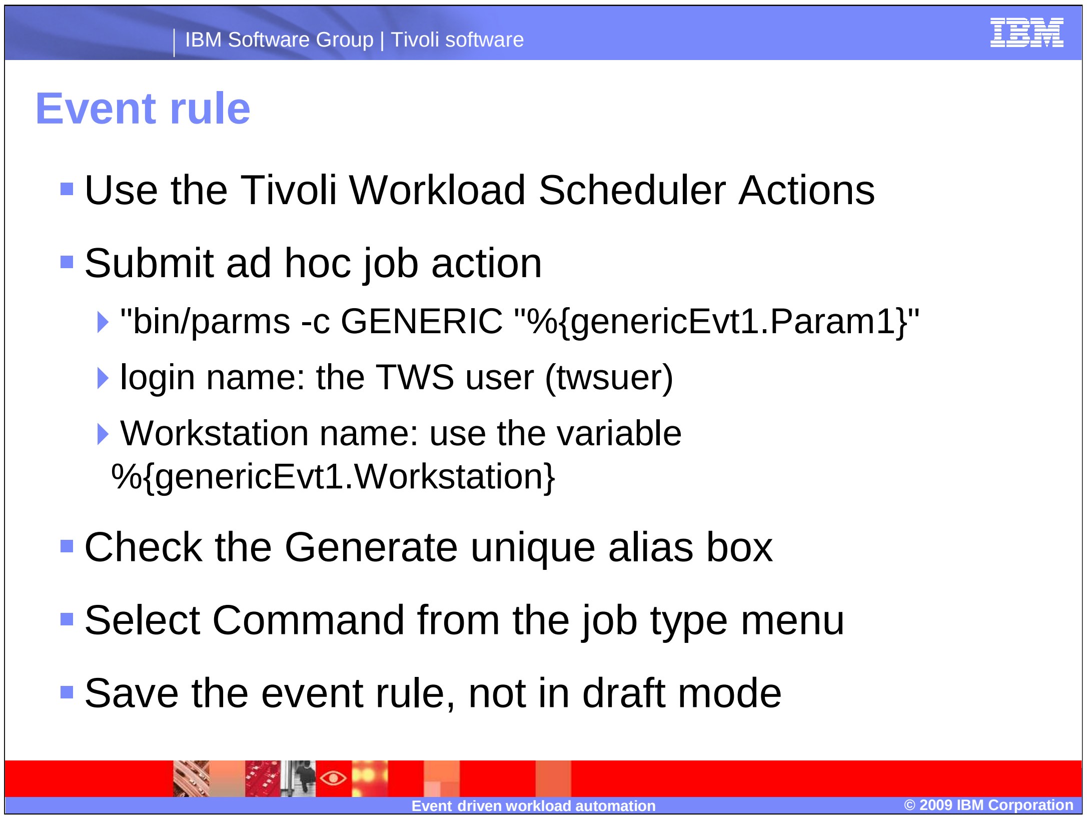


Using sendevent

- Use the sendevent command to send custom data
- sendevent has two required parameters
  - the event name
  - the event class
- Pass data with “Var=value”
- Example

  sendevent Event1 GenericEventPlugIn Param1=‘Generic Text’ Workstation=TWSMASTER

To feed data to the event rule, you use the sendevent command. The sendevent command has two required parameters: the event name and the event class. You will also pass data using variables. For example,
sendevent Event1 GenericEventPlugIn Param1=“Some Text” Workstation=TWSMASTER.



Event rule

- Use the Tivoli Workload Scheduler Actions
- Submit ad hoc job action
  - "bin/parms -c GENERIC "%{genericEvt1.Param1}"
  - login name: the TWS user (twsuer)
  - Workstation name: use the variable %{genericEvt1.Workstation}
- Check the Generate unique alias box
- Select Command from the job type menu
- Save the event rule, not in draft mode

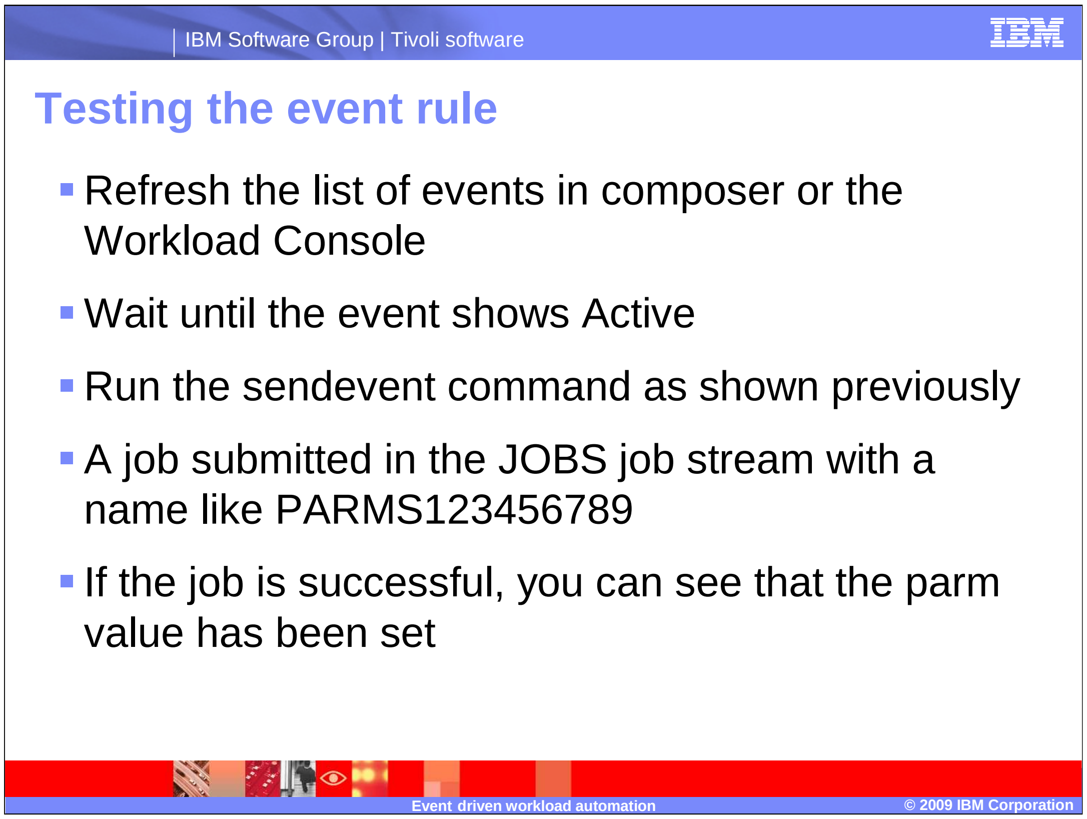
Now that you can feed data through to an event, use the data in an event action. Use Tivoli Workload Scheduler Actions: submit ad hoc job action.

For the command, use "bin/parms -c GENERIC "percent brace generic event one dot parameter 1 brace"

For the login name, use the TWS user.

For the workstation name, use the variable percent brace generic event one dot workstation.

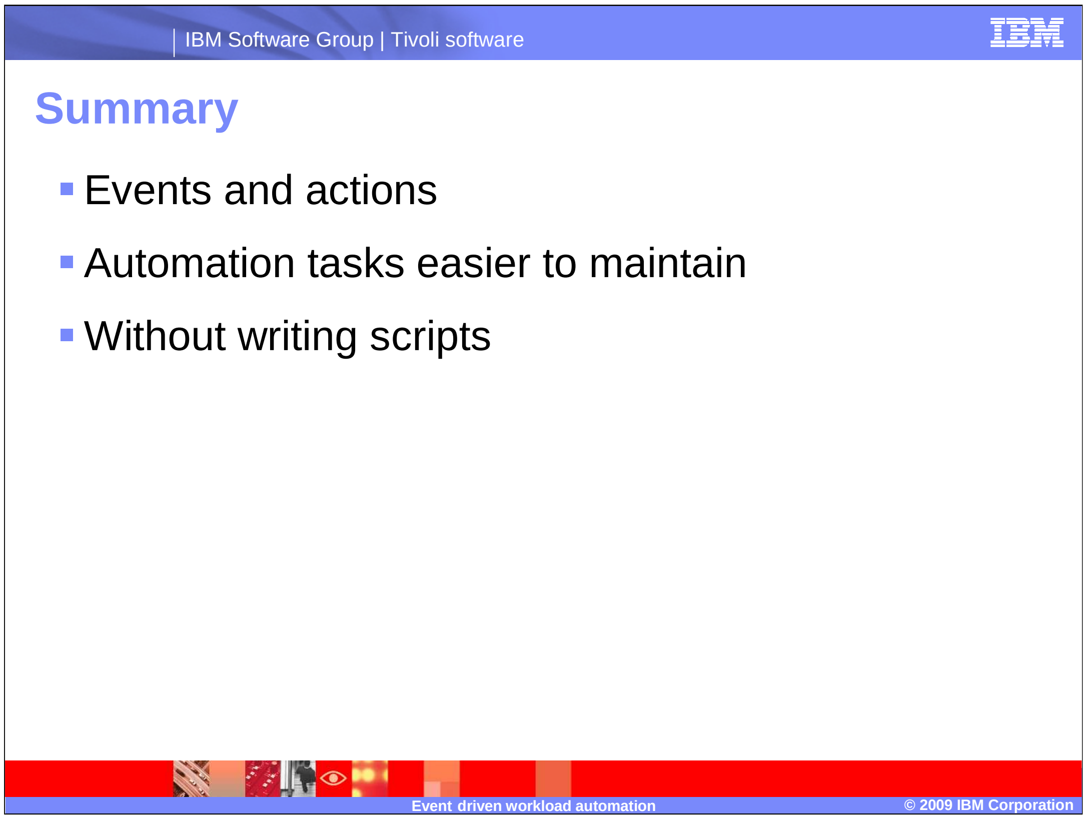
Select the Generate unique alias check box. Select Command from the job type menu. Save the event rule, making sure that it is not in draft mode.



Testing the event rule

- Refresh the list of events in composer or the Workload Console
- Wait until the event shows Active
- Run the sendevent command as shown previously
- A job submitted in the JOBS job stream with a name like PARMS123456789
- If the job is successful, you can see that the parm value has been set

In a few minutes, the event rule will be deployed and you will be able to test the result. Refresh the list of events in composer or the Workload Console and wait until the event shows Active. Then, run the sendevent command as shown previously. You will see a job has been submitted in the JOBS jobstream with a name like PARMS123456789. If the job is successful, you can see that the parameter value has been set by running the parms utility on the workstation.



IBM Software Group | Tivoli software

Summary

- Events and actions
- Automation tasks easier to maintain
- Without writing scripts

Event driven workload automation

© 2009 IBM Corporation

In this session you have learned about using Tivoli Workload Scheduler's event-driven workload automation to add easy-to-maintain automation to tasks that otherwise might have required intensive shell scripting or programming.

Trademarks, copyrights, and disclaimers