Opsware™ Process Automation System

Version 7.0

Process Automation System Concepts Guide

For Process Automation System Central and Studio users

This Guide

This Concepts Guide is intended to introduce you to the basic components and ideas of Opsware Process Automation System (PAS).

It is organized into the following sections:

- Product concepts
- What you can do with PAS
- PAS Architecture and Administration

For advanced concepts, in Help for PAS Studio, see the section on “Advanced Ops Flow Authoring.”

Updating documentation

Documentation enhancements are a continual project at Opsware. You can update the documentation set at any time using the following procedure (which is also available in the PAS readme file).

To obtain PAS documentation

1. On The Opsware Network Web site (https://support1.opsware.com/support/index.php), log in with the Opsware Network account name and password that you received when you purchased PAS.
2. On the **Support** tab, click the **Product Docs** subtab.
3. Under **Quick Jump**, click **Process Automation System**.
4. Under **Process Automation System**, click **ZIP** beside **PAS 7.0 Full Documentation Set**.
5. Extract the files in the .zip file to the appropriate locations on your system:
   - For the tutorials to run, you must store the .swf file and the .html file in the same directory.
   - To obtain the repository that reflects the state of the flow at the start of the tutorial, unzip the file Exportof<preceding_tutorial_name>.zip.
   - To obtain the scriptlet for the tutorial that includes using scriptlets, click the scriptlet .txt file name.
   - To update your Central or Studio Help:
     a. Under **Help Files**, and then click **Studio Help File Bundle** or **Central Help File Bundle**.
     b. In the **File Download** box appears, click either **Open** or **Save**.
     c. Extract the files to the Opsware\PAS home directory, in either the **\Central\docs\help\Central** or **\Studio\docs\help\Studio** subdirectory, overwriting the existing file.

What PAS is

PAS is a system for creating and using actions in structured sequences (Ops flows) that maintain, troubleshoot, repair, and provision your Information Technology (IT) resources by:

- Checking the health of, diagnosing, and repairing networks, servers, services, software applications, or individual workstations.

- Checking client, server, and virtual machines for needed software and updates and, if needed, performing the needed installations, updates, or distributions.
- Performing repetitive tasks such as checking status on internal or external Web site pages.

In PAS Central, users can:

- Run Ops flows
- Administer the system
- Extract and analyze data resulting from the flow runs

In PAS Studio, authors can:

- Create, modify, and test Ops flows, including flows that run automatically, as scheduled.
- Create new operations

  You can create operations within Studio and run them within PAS Central.

  You can also create operations that execute outside Central, in a remote action service. You do so in a development environment that is appropriate to the task, then associate the code that you have created with an operation that you create in Studio.
- Specify which levels of users are allowed to run various parts of flows.

PAS concepts

Ops flows

Ops flows are logically linked sequences of steps that are associated with operations. An Ops flow can perform any task that you can program, on any computer anywhere on an intranet, extranet, or the Internet. For example you might have flows that:

- Check to see whether a service is running, and if not, restart it.
- Triage a Web application to determine which tier is causing performance problems.
- Find errors in configuration files and modify them to correct values.

Ops flows are stored in the Library.

A **subflow** is a flow that is used as a step in another flow. The flow that contains the subflow step is the **parent flow.**

Flow runs

A run is a single execution of an Ops flow in Central. A run can be interrupted, resumed, or handed off to another Central user. The PAS administrator manages flow runs. Flow runs collect data that enables the administrator, managers, and executives, to analyze performance of their IT system.

You can view information that is collected from all of a flows runs, or from all the runs of more than one Ops flow. Collecting such data provides the basis for a higher-level analysis of the state of the IT system and its components.

Repositories

A repository is a hierarchical structure of XML files that constitute a set Ops flows, operations, remote action service references, IActions (if any are used), and system accounts and other configurable objects: domain terms, scriptlets, selection lists, system evaluators, system filters, system properties.

Authors typically work in Studio on a separate repository from the Central repository, then publish their work to the Central repository. By a combination of publishing and updating to and from a Central repository, authors can share their work with each other. Flows can also be published from one Central repository to another, so that you can test flows on a staging Central server before publishing it to the production Central server.

PAS Library

The Library is a folder structure in Studio and Central. In Central, the Library shows a repository's collection of flows. In Studio, the Library displays the entire repository, including flows, operations, remote action services, and system accounts and other configurable objects.

Anatomy of an Ops flow

Steps are the basic unit of an Ops flow. Each step is created from an operation.

- An Ops flow step's operation uses input data to perform a task. From the performance of the task and optional subsequent processing, the operation obtains results.
- The operation has several possible responses, one of which is chosen depending on what its results were.
- Each response is connected by a transition to one of the possible next steps in the flow.

Thus the choice of response determines which is the next step in a particular run of the flow.

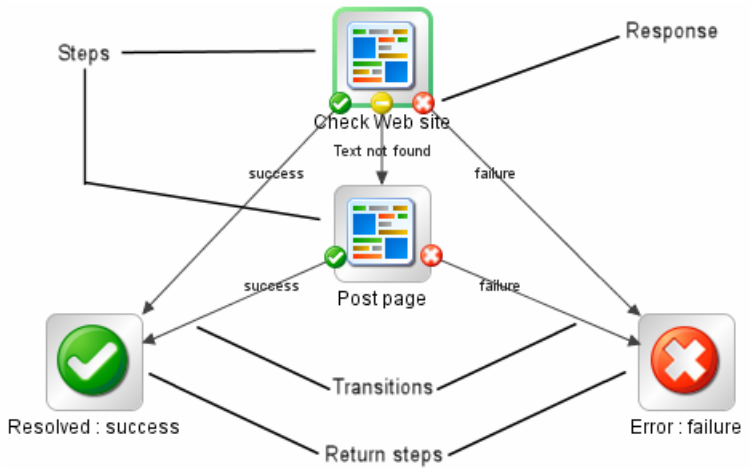


Figure 1 - Parts of a flow

Now let's look at the flow in more detail.

Parts of an Ops flow, in detail

Operations

Operations do the work of the flow. Each operation is the template for the steps that are created from it. For example, a step might be created from an operation that checks a Web page to see whether it contains specific text. Another step in the same flow might be an instance of an operation that copies a file. You could use these two steps in combination to update a Web page.

An operation is an action and, optionally, subsequent manipulation of the data that the action produces. An Ops flow is, technically, an operation, so a step can be created from an Ops flow as well. This means that an Ops flow can contain another flow as a subflow.

The following are examples of tasks that an operation can accomplish:

- Check the status of a service on a computer
- Place or retrieve a file with the ftp put or get commands
- Launch an installing program on a list of computers simultaneously.
- Query a URL for its availability (using an HTTPGet operation).
- Run a SQL query against a particular database table (using a SQL Query operation).

Inputs

Inputs give the operation the data that they need to act upon. For example, an operation to check a Web page needs to know which page to check (mysite.com/mypage.htm) and what text to look for ("needed text"). And the copy-file operation needs a source location and a destination.

Inputs define the means by which operations obtain the data they need to do their work.

Each input is mapped to a variable, whose value can be set in any of several ways:

- In a user prompt, the value is entered by the person running the Ops flow
- The flow author can set the input's value to a specific, unchanging value
- The value can be obtained from another step.
- One of a collection of variables (called "flow variables") and data values that are available to the entire flow can be assigned to the input.

Which element you add an input to can determine when the input's value is obtained:

- An input for a flow obtains a value before the first step runs.

  When possible, it is best practice to set any inputs that the Ops flow needs and that are not produced by processing within the flow in the Ops flow's properties, thus making these input values available to the Ops flow before it begins to run.

- An input for a step obtains a value before the step's operation runs. The operation processes a step input's value only during the execution of that

step. In another step that is based on the same operation, the operation processes the second step's value for the input of the same name.

- When you change an input for the operation in the library, all instances of the operation that you subsequently create reflect the change that you made.

Outputs

There are several kinds of outputs for flows, steps, and operations: raw results, results, and responses.

- Raw results are a collection of key value pairs representing the raw data that was returned from an operation executed in the context of a flow.

  For example, if you execute the ping operation on a windows XP machine, you will get the following results:

```
{
   Code = "0"
   Error String = ""
   Output String =
         "Pinging apple.com [17.254.3.183] with 32 bytes of data:

      Reply from 17.254.3.183: bytes=32 time=24ms TTL=244
      Reply from 17.254.3.183: bytes=32 time=24ms TTL=244
      Reply from 17.254.3.183: bytes=32 time=25ms TTL=244
      Reply from 17.254.3.183: bytes=32 time=26ms TTL=244

      Ping statistics for 17.254.3.183:
         Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
      Approximate round trip times in milli-seconds:
         Minimum = 24ms, Maximum = 26ms, Average = 24ms"
}
```

  You can filter the values of the raw results to create a more refined result. For example, if in the above data you are interested only in the average latency of the ping operation, you can select the Output String field of your operation and specify a series of filters which extract the '24ms' token at the end of the string. For more information, see Creating advanced inputs.

- A result is an output string that a step's operation produces.

  A step or Ops flow can have only one result. The result can be produced by an operation scriptlet or obtained from one of the fields in the raw results. You can use results to annotate strings or send them to the local or global context.

- Responses are the outcomes that are connected, each response by one transition, to a following step. Because you link a response to a subsequent step, the response is the determination of what the flow does next. Return steps are an exception: Return-step responses return an outcome for the entire flow.

  The following table shows which output types pertain to operations, steps, and flows, respectively. Note, however, that you cannot change the handling of outputs for sealed operations.

The following table shows which output types pertain to operations, steps, and flows, respectively. Note, however, that you cannot change the handling of outputs for sealed operations.

	Raw results	**Results**	**Responses**
Operations	✓	✓	✓
Steps		✓	
Flows	✓	✓	✓

Responses

Responses are the possible outcomes of the operation. A “get Web page” operation has three responses: “page not found”, “text not found”, and “success” (if we got the page and it has the desired text). A “copy file” operation might have just “success” and “failure” for its responses.

Transitions

Our read Web page operation may need to respond differently if the Web page can’t be found, if the page is there but the text isn’t present, or if the page is there and the desired text is present. A transition leads from an operation response to one of the possible next steps, so that the operation’s response determines what the next step.

Flow variables

Flow variables are variables that store data obtained by operations or their scriptlets for use in other flows or other steps within a flow.

- **Global flow variables** are available for all flows (both parent flows and subflows) within the current run. When a step in a subflow stores a value in a global flow variable, the variable’s value becomes available to the parent flow and any other subflows.
- **Local flow variables** are available only to the current flow, whether it is a parent flow or subflow.

Checkpoints

A checkpoint in a flow step gathers the state of the flow at that step and saves it to the Central database. The flow's state comprises all the data necessary to completely reconstruct the run in case of a system crash. If you have configured a failover/run recovery cluster for Central and the Central server running a flow fails, other nodes in the cluster resume the flow runs that were taking place on the failed Central server. If a flow has no checkpoints, the run is resumed at the start of the flow. But if there are checkpointed steps in the flow, the run is resumed from the last checkpoint that was reached when the server failed.

A little deeper dive on steps and operations

Now that you're familiar with the basic parts of an Ops flow, let's look at Ops flow steps and operations in more detail.

What happens when a step is executed

When a step is carried out, the following sequence of actions is executed:

1. Input values are obtained from the data sources that have been defined for the inputs and are made available to the step's operation.
2. The step's operation is carried out.

   For details on how information is obtained by, flows through, and can be modified within an operation, see the preceding section, Operations: architecture and information flow.
3. If the operation has a scriptlet, the operation's scriptlet is executed.

   The operation's scriptlet can do the following:

   d. Select the operation response.
   e. Set the operation's primary result.
   f. Make changes to local and global flow variables.
4. If the operation's scriptlet has not already set the operation's primary result, the operation's primary result is set now.
5. If the operation's scriptlet has not already selected the operation's response, the response is selected now, using the operation's evaluation rules for selecting a response.
6. Step results are extracted.
7. If the step has a scriptlet, that step scriptlet is executed.

   The step scriptlet can do the following:

   a. Select the operation response.
   b. Make changes to local and global flow variables.

   Note: The step scriptlet can not set the step's primary result.
8. The transition that is associated with the selected response is selected.
9. The next step is executed, using this same sequence.

Steps and operations, compared

Steps are the building blocks of Ops flows. Each step in a flow is created from an operation, which the step is an instance of.

Because the step is an instance of the operation:

- The step inherits the inputs, flow variables, and properties of the operation.
- Changing a step's input or local variable changes the input for the operation or subflow when that step runs. The operation itself remains unchanged. Thus you can specify different inputs in different steps created from a single operation.
- If you change the operation, you are modifying the template that is the basis for all the steps associated with that operation. The steps that have been created

from that operation are either changed also or broken, depending on what you have changed. In turn, all the flows that contain those steps may be broken due to the operation's having changed.

For more information on what the relationship between steps and operations means when you are creating an Ops flow, see Help for Studio, in the section on Ops flow architecture and concepts.

Operations: The models for steps

In addition to its responses, an operation is made up of the following:

- **Command:** This dictates most of what the operation actually does. (The operation may also contain a scriptlet, which processes data.) The command may be any of several types of commands, including command-line, an http operation, or a script to run. For example, an operation might get a directory listing, check to see if a service is running, execute a Web service, or run a UNIX vmstat command.
- **Results:** When a command runs, it returns data, called results. For example, the results returned by a dir command include a file list. A ps command results are a list of processes. Most commands have more than one result, returning data such as a return code, standard output (stdout), and error output (stderr). Our get http page operation needs results for the http return code (200, 302, 404, etc) and the data on the page.

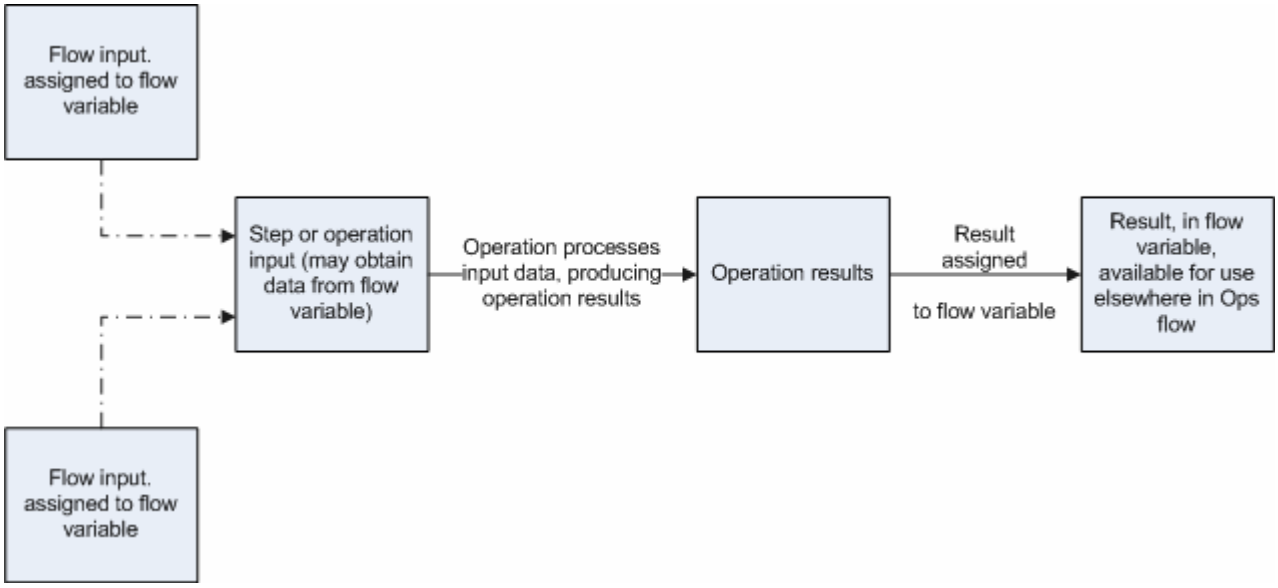
  Some commands can return a lot of data that we may want to use later on in our flow. Using a single command, an operation to get memory statistics on Linux can tell us how much memory is free, how much total memory is available, how much swap memory is being used and much more.
- **Filters:** Figuring out what response to take based on the data that a command returns may require filtering a key piece of data out of a result or creating a scriptlet that manipulates the data.
- **Rules:** Rules evaluate the operation's results to determine which operation response to take when the step runs. Rules can evaluate any of the results fields, the data strings, return codes, or error codes.

  A given response for an operation is selected when the conditions described in the response's rule match the data that you have specified in or extracted from the one of the results fields. The rules that you create are comparisons or matches between a string of text that you describe in the rule and the results field that you select for the rule.

  Consider the get Web page operation in our example:

  - The "page not found" response would be selected if the http return code were 404.
  - The "text not found" response would be selected if text to check were not contained in the data on the page.
- **Scriptlets:** Scriptlets (written in JavaScript or Perl) are optional parts of an operation that you can use to manipulate data from either the operation's inputs or results for use in other parts of the operation or flow.

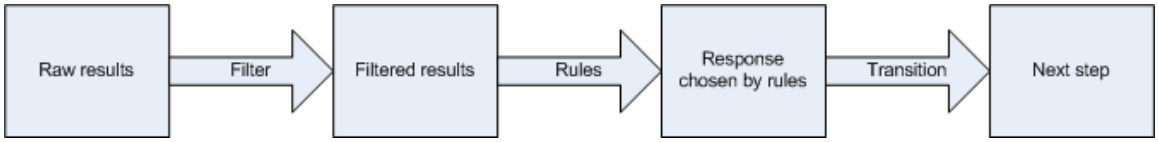
The following diagram shows how operations can get values from flow, step, and their own operation inputs, and how data in operation results can be passed to flow variables, thus becoming available to other parts of the Ops flow.



Apply this diagram to a flow that runs the Windows command-line dir command on a directory:

10. Flow inputs could provide two needed pieces of data: the host computer and the name of the directory to run the dir command against
11. These two input data items could be stored in flow variables named “host” and “directory,” respectively.
12. The operation inputs could obtain the values from the flow variables.
13. After the operation performs its task based on the inputs, the step could assign the operation results to another flow variable, which another operation could use in another step in the flow.

When you create a step from an operation, each of the operation’s responses must be the starting point for a transition to another step. Thus during a particular run of the Ops flow, the rules that evaluate the operation’s results determine the response of the operation, which determines the transition that is followed and therefore the path that the run follows through the steps.



Note: You can also set the operation’s result to supply values in fields to the result of a step created from that operation, then in turn set that step result to supply those values in fields to the Ops flow’s result. You can use the Ops flow result to pass the values to other flows.

Operations: architecture and information flow

An operation is a defined sequence of actions associated with a step in an Ops flow. When we consider a step that has a flow associated with it as a subflow, we say that the subflow is a type of operation. However, the following description is limited to a single operation.

The parts of an operation are:

- Core functionality (called the *core*), which encapsulates the business logic of the operation

  For Web operations, the core functionality is the IAction interface execute method and the method's parameters, which provide data to and influence the behavior of the execute method.

  All input values are copied to the local or global context before execution of the operation. Input values can also be bound to the local or global context.

  The core might map input onto raw results.

- Further processing of raw results by a scriptlet (optional)
- Determination of a response

The *context* is a container that is independent of the step and holds various values that can be exchanged with the step at various points (see the following diagram). There are two kinds of context: global and local. Local context exists for the duration of the step; global context exists for the duration of the Ops flow. You can pass values to and from the local or global context.

In information flow through an operation, as shown in the following diagram, these parts of an operation play roles:

- Raw results

  If the IAction interface is part of the core functionality, the raw results are the data and state.

- Scriptlet

  An optional interpretive program that may be executed at the end of a step. The scriptlet often evaluates the raw results of the operation and produces the output data of the operation.

- Output data

  The data produced by the operation, if any.

- Response

  The evaluation of the operation's output and the resulting determination of the transition from among the possible transitions for the operation's step.

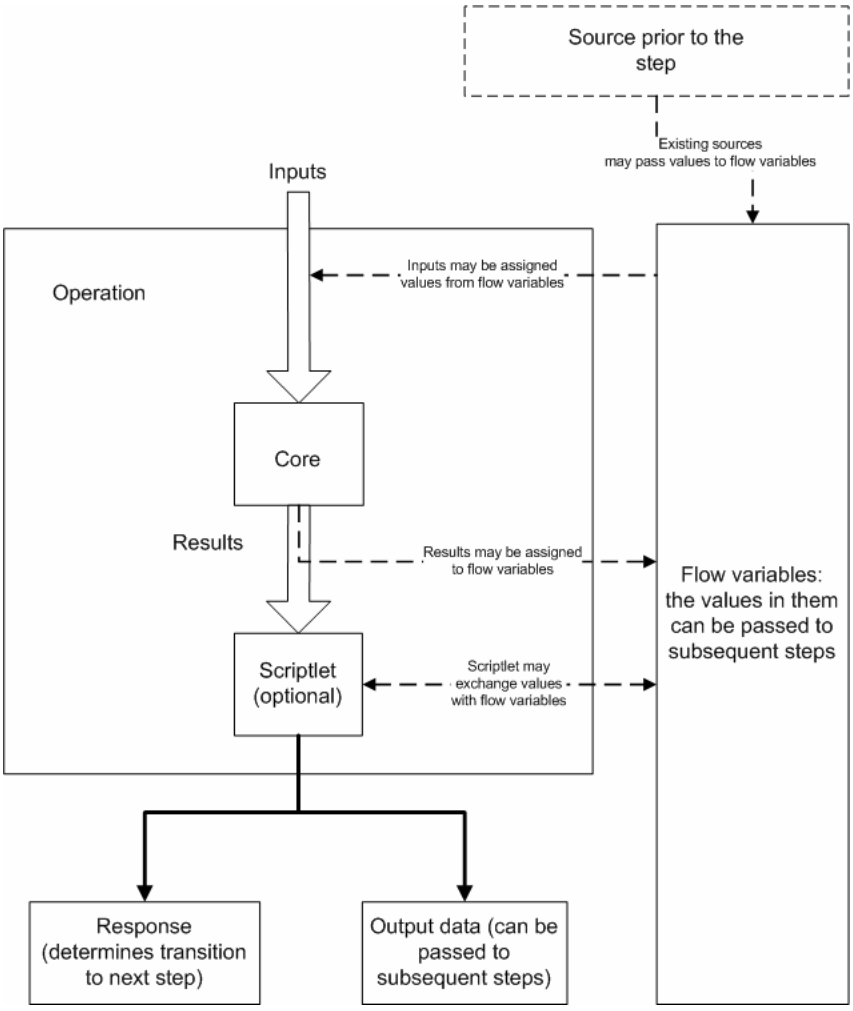


Figure 2 - Information flow through an operation

Note: Operations that are provided in the iConclude folder are sealed—that is. you cannot modify them other than to supply fixed, specific values for inputs.

PAS architecture and administration

PAS includes the following components:

- PAS Central

  Central is the Web-based application in which you execute flows and review the information that the Ops flows have extracted on the health of the IT resources against which the flows have run.

- PAS Studio

  Studio is the standalone application in which you create Ops flows or import them (in “Accelerator Packs”).

- A database, which stores data related to results from running Ops flows.
- Accelerator Packs and Ops flow content, which are ready-to-use Ops flows and operations with which you can build more Ops flows. Each Accelerator Pack is specific to a platform or software application, such as Oracle, Red Hat, etc.
- Remote action services

Remote action services (RASs), which are services that are installed with PAS, are the media by which PAS operations that are carried out outside PAS. Running outside of PAS includes one or more of the following:

- Running remotely or automatically
- Interacting with platforms, environments, or applications that are not directly accessible from PAS

Once you have installed a RAS, then in Studio you can configure a reference to the RAS. Operations that use the RAS to run remotely access the RAS through the reference that you create in Studio.