Logo

Logo

# Informatica Cloud Naming Conventions

* [Agents and Agent Groups](#agents-and-agent-groups)
  + [Secure Agent](#secure-agent)
    - [Naming Agent Groups](#naming-agent-groups)
      * [Name Structure](#name-structure)
    - [Naming Agents](#naming-agents)
      * [Name Structure](#name-structure-1)
* [ICS Artifacts](#ics-artifacts)
  + [Connections](#connections)
  + [Tasks](#tasks)
    - [Name Structure](#name-structure-2)
    - [Examples](#examples)
  + [Mappings](#mappings)
    - [Name Structure](#name-structure-3)
    - [Examples](#examples-1)
  + [Task Flows](#task-flows)
    - [Name Structure](#name-structure-4)
    - [Examples](#examples-2)
  + [Schedulers](#schedulers)
    - [Name Structure](#name-structure-5)
    - [Examples](#examples-3)
  + [Custom Views](#custom-views)
* [ICRT Naming Conventions](#icrt-naming-conventions)
  + [Connectors](#connectors)
    - [Name Structure](#name-structure-6)
    - [Examples](#examples-4)
  + [Connections](#connections-1)
    - [Name Structure](#name-structure-7)
    - [Name Variants](#name-variants)
  + [Processes](#processes)
    - [Data Access Processes](#data-access-processes)
    - [General Purpose Processes (Utility Services)](#general-purpose-processes-utility-services)
    - [Business Process Flow Naming Convention](#business-process-flow-naming-convention)
    - [Integration Process Flow Naming Convention](#integration-process-flow-naming-convention)
    - [Names Aligned with the DI Process Architecture](#names-aligned-with-the-di-process-architecture)
    - [IPD Process and Guide Variables](#ipd-process-and-guide-variables)
    - [Variable Examples](#variable-examples)
      * [Process Titles](#process-titles)
    - [Recommended process Title Structure](#recommended-process-title-structure)
      * [Main Process](#main-process)
      * [Sub Processes](#sub-processes)
      * [Example of retrying process Title](#example-of-retrying-process-title)
    - [Example Initialize and Set Process Title](#example-initialize-and-set-process-title)
      * [Main Process](#main-process-1)
      * [Subprocess](#subprocess)
    - [Example Update Process Title](#example-update-process-title)
      * [Success](#success)
      * [Fault](#fault)
  + [Process Objects](#process-objects)
  + [Guides](#guides)
    - [Examples](#examples-5)

## Agents and Agent Groups

### Secure Agent

Integration tasks can run on Secure Agent groups or the Hosted Agents. Secure Agents can be grouped for load balancing and high availability. When you add/register new secure agent to your org it will default to own agent Group and use Host name as the agent name. You may see something like this in Informatica Cloud Administration

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Environment Name | Status | Host Name | Platform | Version | ... |
| Informatica Cloud Hosted Agent | Up and Running |  |  |  |  |
| agent1 (1) |  |  |  |  |  |
| -- agent01 | Up and Running | agent01 | Linux64 |  |  |
| iclab (1) |  |  |  |  |  |
| -- iclab | Up and Running | iclab | Linux64 |  |  |

Agents can be used separately or put into groups which can be used to execute integration tasks on agent grouped based on regions or functional requirements.

It is recommended to rename your groups and agents with logical names rather than using physical host names names. Agents can be then migrated more easily to a different host. Environment migrations from DEV to QA to PROD will be also easier as they will not require renaming of the references used in the design artifacts such as connections and processes.

Example Regional Layout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Environment Name | Status | Host Name | Platform | Version | ... |
| Informatica Cloud Hosted Agent | Up and Running |  |  |  |  |
| NA (2) |  |  |  |  |  |
| -- na\_agent\_01 | Up and Running |  | Linux64 |  |  |
| -- na\_agent\_02 | Up and Running |  | Linux64 |  |  |
| EMEA (2) |  |  |  |  |  |
| -- emea\_agent\_01 | Up and Running |  | Linux64 |  |  |
| -- emea\_agent\_02 | Up and Running |  | Linux64 |  |  |
| GLOBAL (2) |  |  |  |  |  |
| -- global\_agent\_01 | Up and Running |  | Linux64 |  |  |
| -- global\_agent\_02 | Up and Running |  | Linux64 |  |  |

In above example of Informatica Cloud Organization is used to manage integration processes across several regions where each region supports specific set of integrations and region specific connections.

Another possible grouping of agents can be on functional or specific integration needs basis i.e.

* Batch. vs. Real Time
* Enterprise/Division Level
* Internal/External/Hybrid Integrations
* Sizing levels such as Small,Medium,Large

#### Naming Agent Groups

Using upper case for groups makes it easier to know if the Design object references a group or a specific agent.

* use upper case
* separate words by underscores
* standardize and document abbreviations

##### Name Structure

<group\_logical\_name>\_<number>

#### Naming Agents

* use all lower case
* use underscore to separate words
* logical name can indicate os/sizing and other environment characteristics

##### Name Structure

<group>\_<agent\_logical\_name>\_<number>

|  |  |
| --- | --- |
| Example | Name Structure Variant |
| na\_aws\_east\_01 | <geography>\_<hosting>\_<region>\_<agent\_number> |
| na\_east\_sm\_suse\_01 | <geography>\_<hosting>\_<region>\_<agent\_number> |
| na\_rt\_01 | <geography>\_<rt\_or\_batch>\_<agent\_number> |

Another organizational layer of Informatica CLoud is ability define and use Informatica Cloud Sub-Organizations Sub Organizations can be used to further isolate different group of integration assets as well manage more granular access to specific integrations or hierarchy driven by corporate organizational hierarchy.

Administrators of parent organizations manage and monitor child organizations from one place. See [Organization Hierarchies](https://app3.informaticacloud.com/saas/v389/docs/EN/index.htm#page/cc-cloud-administer/Organization_Hierarchies.html#3_12_8_1)

## ICS Artifacts

### Connections

Connection Names are often specific to a connector used by the connection some names can simply reflect connection system name but some other connection may have many instances instances for some connections simple name suffice and some other connection names may need to describe their logical or physical attributes

* maximum name length is 100 characters
* use uppercase for abbreviations
* Standardize and document abbreviations
* logical names can have further structure to define connection specific sub type
* suffix connection with location or other specific attributes needed to distinguish specific connection
  + old/new for migrations
  + version
  + region or division
  + location

Generic Name Structure

<ConnectionType>\_<logical\_name>\_<flag(s)>

Examples:

|  |  |  |
| --- | --- | --- |
| Connection Type | Recommended Name | Examples |
| Database | <db\_type>\_<logical\_name>\_<schema\_name> | ORCL\_siebel\_contact\_center |
| Flat File | FF\_<logical\_name>\_<folder\_name> | FF\_salesforce\_lookups |
| ODBC | ODBC\_<DS Name> | ODBC |
| SAP | SAP\_<type>\_<logical\_name>\_<region> | SAP\_BAPI\_COMPANYCODE\_GETDETAIL\_NA, SAP\_IDOCR\_MATMAS04 |
|  |  |  |

### Tasks

#### Name Structure

* Maximum name length is 100 characters
* Prefer to use underscore as a word separator
* It is allowed to choose all lover case or mixed case, it is desirable to keep convention consistent for different types of tasks

DSS\_<SOURCE>\_<TARGET>\_<NAME>\_(INS|UPD|DEL|UPSERT)\_<VERSION>

#### Examples

* DSS\_Salesforce\_Workday\_Accounts\_UPSERT
* DSS\_Coupa\_SAP\_Requisition\_NA\_INSERT\_01
* DSS\_Coupa\_SAP\_Requisition\_NA\_UPDATE\_01

### Mappings

* Maximum name length is 200 characters
* Prefer to use underscore as a word separator
* It is allowed to choose all lover case or mixed case, it is desirable to keep convention consistent for different types of mappings

#### Name Structure

# for mappings that have single source target  
M\_<SOURCE>\_<TARGET>\_<NAME>\_<VERSION>  
  
# mappings that use multiple Sources can contain entity names in the name  
M\_<SOURCE>\_<ENTITY>\_to\_<TARGET>\_<NAME>\_<VERSION>

#### Examples

* M\_CDI\_Accounts\_Contacts\_to\_SFDC
* M\_CDI\_Accounts\_FF\_incremental

### Task Flows

#### Name Structure

* Maximum name length is 100 characters
* Prefer to use underscore as a word separator
* It is allowed to choose all lover case or mixed case, it is desirable to keep convention consistent

TF\_<NAME>\_<VERSION>

#### Examples

* TF\_CDI\_Acount\_Contacts\_to\_SFDC
* TF\_Hierarchy\_Change\_Processing

### Schedulers

* Maximum name length is 100 characters
* Prefer to use underscore as a word separator
* It is allowed to choose all lover case or mixed case, it is desirable to keep convention consistent once you define Specific prefix and name structure

#### Name Structure

Schedule\_<TIMEZONE>\_<Frequency>\_<TIME>  
 or  
 SCH\_<TIMEZONE>\_<Frequency>\_<TIME>

#### Examples

* Schedule\_Five\_Minutes
* Schedule\_Hourly
* Schedule\_PST\_Daily\_11PM

### Custom Views

Note: taken from KB, but needs review and andjustments

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Name Stucture | Example | Description |
| Activity Log | View\_ | View\_DSS\_SFDC\_Success | This view will display the activity log of all successful DSS tasks which contain SFDC keyword |
| Connection | View\_ | View\_SFDC\_User01 | This view will display all the SFDC type Connections created by User01 |
| Task | View\_ | View\_SFDC\_ACCOUNT\_User01 | This view will display all the tasks containing the keyword SFDC\_ACCOUNT created by User01 |
| Task Flow | View\_ | View\_SFDC\_ACCOUNT | This view will display all the task flows containing the keyword SFDC\_ACCOUNT |

## ICRT Naming Conventions

ICRT Naming Conventions have different specific constraints, these tend to be less strict than in ICS.

* Valid characters are specific to an object type
* Specific for Object Type
* Usually enforced by code reviews

### Connectors

* Maximum name length is 128 characters
* No Strict Name structure as names are often derived from the consumed service name which we do not control
* Descriptive Name may but does not have to contain API version
* Version defined as connector parameter or tag, both are acceptable depending on the service versioning model

#### Name Structure

<NAME>-<VERSION>

#### Examples

* IRefreshWebService
* SFDC-SOAP-API

### Connections

* Maximum name length is 128 characters
* Match the Name of the Service and Data Source + additional information such as Version, Target Agent
* Connection may have different characteristics, thus different name structure can be used based on the connection type

#### Name Structure

<CONNECTOR>-<Version>-<Module>-<LOCATION>

#### Name Variants

<Connector>-CventWeserviceSoap  
<Connector>-<LogicalSystemName>  
SFDC-Atlas  
SFDC-MDM  
SFDC-ATS  
  
<Connector>-<Module>-<TargetAgentOrGroup>  
Workday-V2-ResourceManagement-agent01

### Processes

ICRT Processes may have many different purposes that also drive their naming conventions

Maximum name length is 80 characters

#### Data Access Processes

Typically Encapsulate CRUD operations on managed entities

* CreateEntity
* UpdateEntity
* UpsertEntity

#### General Purpose Processes (Utility Services)

Follow General Web Service and SOA naming conventions

* The utility-centric context is found in application services involving operations that encapsulate cross-cutting functions, such as event logging, exception handling, or notification. These reusable services need to be labeled according to a specific processing context, agnostic in terms of any particular solution environment. For example, a utility service might be named Notify.
* An entity-centric context is established in a business service that represents a specific business entity, such as an invoice or a purchase order. The labeling of entity-centric business services is often predetermined by the entity name. For example, a service may simply be named Invoice or Customer.
* Task-centric contexts are required for services modeled to encapsulate process logic. In this case, the thread that ties together the grouped operations is a specific activity being automated by the service logic. Therefore, the use of verbs in service names is common. For example, a task-centric service may be called GetProfile or ProfileRetrieval, if that accurately represents the task's scope.

#### Business Process Flow Naming Convention

* Name the process with a descriptive title. The name should be descriptive of what the process does (ex: MissingClinicalDataCase).
* Add the prefix (MP-) to name if it is a master process. A master process is generally called directly from outbound message. It also contains the error handling.
* Add the prefix (ER-) for error handling processes.
* Add the prefix (SP-) for processes called by a master process.

#### Integration Process Flow Naming Convention

To be reviewed - Need peer review

* Name the process with a descriptive title. The name should be descriptive of what the integration process does (ex: SendAdHocDataMessage, MapOrderRoleData).
* Add the prefix (INT-) to all processes that are integration specific.
* Add the prefix (INT-MP-) to name if it is an integration master process. An integration master process controls the overall function of the integration. It is can be called by a business process or directly by a workflow outbound message. It also contain the general error handling of the integration, including retries.
* Add the prefix (INT-SP-) for integration processes called by an integration master process.
* Add the prefix (INT-XDM-) for integration processes responsible for data mapping.
* Add the prefix (INT-SXDM-) for integration processes responsible for data mapping that are reusable by multiple data mapping processes.

#### Names Aligned with the DI Process Architecture

<Source>-<Target>-<UseCase>-<Component>

* SFDC-Workday-Upsert-Project-Handler
* SFDC-Workday-Upsert-Project-Job
* SFDC-Workday-Upsert-Project-ETL

#### IPD Process and Guide Variables

There are essentially three types of variable in IPD processes

|  |  |
| --- | --- |
| Type | Name Structure |
| Input | in\_<name> |
| Temporary | tmp\_<name> |
| Output | out\_<name> |

Input Variables define an input of the process, guide or service. Temporary variables are used by the process to perform data transformation or store state information and intermediary results. Output variables define output data pf the Process or service. You should use prefix for all process/guide variables which makes makes it easier to distinguish between process object attributes and variable in expression editor as well as distinguish the container of the variable (temp, in, out). We recommend to standardize commonly used variables to represent certain process runtime data across all processes.

#### Variable Examples

* tmp\_process\_title - use this variable to store updated process title, this is very useful when you want to track process state and or progress information as well as other tracking information like entity or transaction id associated with the process
* tmp\_script\_out - some function used in the process expressions do not generate output as they alter state of the process or interact directly with the engine in such case this variable is used to store result of such activity example of such function is the ipd:setProcessTitle()

Standardized input parameters for sub Processes

* in\_parent\_process\_id Should contain the parent process if the process from another process
* in\_main\_process\_id Should contain the main job or integration process id

##### Process Titles

Make the process Titles consistent across the board the process title should follow these rules

* Always include actual process name and make it first part of the process title
* Include IDs of main item or when the process takes care of single item
* When process handles list of items rather than single one, include only count of items and indicate progress if desired
* Include Status information to indicate step of the process or final status
* Include Error Message or its part when process faults
* Indicate that process is retrying activities and how many retries it performed when applicable
* Be aware of size limit of process title (256 bytes) (make sure you trim it to fit, especially when reporting error message to process title)
* Define standardized process title structure for your project
* There Are some cases when updating a process title is not desired
  + Time critical, hight throughput processes (Updating process title has DB I/O overhead, so keep that in mind when using this technique to improve process tracking)
  + in-memory processes (when you set process logging level to none, it is not persisted and setting process title does not provide any value)

Example of process taking care of single item as it changes through the steps and status changes for a process that handles single item or item hierarchy (related items with single main parent)

pim\_item\_creation [ActiveRecords-Catalog:Article:50617@1000] [itemNo:1437605384345] [Loading Item Details]  
pim\_item\_creation [ActiveRecords-Catalog:Article:50617@1000] [itemNo:1437605384345] [GS1 Attributes]  
pim\_item\_creation [ActiveRecords-Catalog:Article:50617@1000] [itemNo:1437605384345] [EAN]  
pim\_item\_creation [ActiveRecords-Catalog:Article:50617@1000] [itemNo:1437605384345] [Dimension Attributes]  
pim\_item\_creation [ActiveRecords-Catalog:Article:50617@1000] [itemNo:1437605384345] [GTIN]  
pim\_item\_creation [ActiveRecords-Catalog:Article:50617@1000] [itemNo:1437605384345] [Final Validation]  
pim\_item\_creation [ActiveRecords-Catalog:Article:50617@1000] [itemNo:1437605384345] [Final Approval]  
pim\_item\_creation [ActiveRecords-Catalog:Article:50617@1000] [itemNo:1437605384345] [Approved-Merged]

#### Recommended process Title Structure

##### Main Process

Main Process is a process that contains core orchestration of the Job Intgeration Pomponent, main process is as service directly triggered by scheduler, inbound event, JMS or File listener

MP\_Process\_Name id:{$objectId} step:{$logicalStepName} status:{$status} {$errorMessage}

##### Sub Processes

Sub Process is a typically re-usable processs serving generic function in the process chan invocation, it is usually not invoked directly (except unit testing) it partcipates in orchestration and there are several sub-types of subProcess

* Data mapping (transformation) process
* Utility process, facilitates re-usable operations

SP\_Process\_Name oid:{$objectId} pid:{$parentProcessId} mid:{$mainProcessId} status:{$status} {$errorMessage}

##### Example of retrying process Title

Process\_name oid:0014B00000LEWW6 pid:123456 mid:344666 status:RETRY:activityname[1]  
Process\_name oid:0014B00000LEWW6 pid:123456 mid:344666 status:RUNNING  
Process\_name oid:0014B00000LEWW6 pid:123456 mid:344666 status:FAULTED Failed to Lock Record  
Process\_name oid:0014B00000LEWW6 pid:123456 mid:344666 status:COMPLETED

#### Example Initialize and Set Process Title

##### Main Process

Assign to $temp.tmp\_process\_title

let $id\_label := concat("sfid:",$temp.tmp\_object\_id)  
let $title := "MP-Order-Processing"  
let $parts := ($title,$id\_label)  
return string-join($parts," ")

##### Subprocess

let $id\_label := concat("sfid:",$temp.tmp\_object\_id)  
let $pid\_label := concat("pid:",$input.in\_parent\_process\_id)  
let $mid\_label := concat("pid:",$input.in\_main\_process\_id)  
  
let $title := "SP-Order-Update-Status"  
let $parts := ($title,$id\_label,$pid\_label,$mid\_label)  
return string-join($parts," ")

#### Example Update Process Title

You should be assigning to tmp\_script\_out as the ipd:setProcessTitle function returns true/false only

##### Success

let $parts := ($temp.tmp\_process\_title, "status:COMPLETED")  
return ipd:setProcessTitle(substring(string-join($parts, " "),1,255))

##### Fault

let $error := $output.faultInfo[1]/reason  
let $parts := ($temp.tmp\_process\_title, "status:FAULTED", $error)  
let $process\_title := substring(string-join($parts, " "),1,255)  
return ipd:setProcessTitle($process\_title)

### Process Objects

Process object are logical data structures defines in ICRT, or received from inbound calls. Process objects are used both in processes and within service connectors.

* Maximum name length is 128 characters
* Process Objects are often derived and directly named after underlying data structures used by the Connectors and Systems they map to a specific JSON or XML data elements and attributes
* Process Objects are logically names after the data which that store (ex. OrderRoles).
* Process Objects used within service connectors can contains the suffix (AnonymousData).

### Guides

* Use descriptive names
* Use space as the word separator
* Use Tags to classify guides
* For test harness Guides, align the naming with the processes or connectors they test

#### Examples

* After Meeting with Contact
* Initial Lead Qualification
* Atlas-ATS-Objects-Upsert-guide