SAP SuccessFactors

PUBLIC

Document Version: 1H 2025 – 2025-04-04

SAP SuccessFactors HCM Suite Boomi Development Standards Guide

THE BEST RUN SAP

Content

1 Introduction

In order to ease collaborative development and low cost of ownership of Boomi projects, it is important that some basic standards are followed. This will ensure projects are consistently organized and any team member can easily understand what a process does and how it does it.

Boomi is a very graphical, “self documenting” environment but there it is still possible to create processes that are incomprehensible to newly assigned personnel. The cost of ownership of such processes escalates exponentially and the ability to reuse them is greatly diminished if standards are not followed.

Ultimately, processes should be created so that others can easily understand and maintain them, and this document is meant to provide guidelines and best practices to achieve this.

2 Boomi Customer Accounts

2.1 Boomi Project Organization

Certain nuances of Boomi functional behavior drives development procedures. Thoughtful design and maintenance processes should be followed. The following functional behavior is considered when managing projects:

It is a best practice to create a separate folder for every process.

Copies of processes and components represent new components ids. New Ids will lose their association and extension values of a deployed production components. This means that copying/branching all components of a process will lose its associations with the deployed original processes.

Processes and all dependent components, sub processes are deployed as a snapshot

- Deploying of other processes that share components have no effect on an existing process deployment

It can be difficult to merge changes in multiple process elements • Process steps, components, sub processes

- Use of shared components across main process copies can help but induces dependency complexities.
- There is not yet the ability to cut/paste steps/component references within a process.
- Version control does support visioning of each component
- Processes and components associate with the current dependent component
- Rolling back a component version does NOT roll back dependent components
- You can compare deployed versions of the same process in the Deployment tab

Top level should contain only shared components.

Processes related to a specific 3rd party system integration should be in their own folder with the process specific components.

If a subset of processes share components, create another level of folder for each process.

- For example, ADP folder may have common components but ADP sub folders contain specific integrations to ADP versions.

2.1.1 Component Explorer Organization

1. Organize Top Level Folders
2. Stable, shared component folder(s)
3. Connections are resources constrained by Boomi licenses. Do not bury them in sub folders. Keep them at the top level.
4. Connections should created once and shared across all processes. This will reduce the connector license usage.
5. Do not have copies of connections for test/prod tenants.

6. Expose required connection tenant, username and passwords as “extensions” and set the extension values in the dev, prod and test environments. This prevents development credentials from being accidently used in production as defaults and keeps the production credentials exclusively on the production atom.

SAP SuccessFactors Components Reference Library:

1. This folder should appear in new customer accounts.
2. It contains utility components useful for development
3. Separate Top Level folders for each release “Wave”.
4. Use a separate folder within for each process .
5. Add folders for shared components at the same level as the processes that use them. For example, multiple processes for integrations to the same 3rd party system that share operations, connections..., profile/schema definitions and other components connections...
6. Use “Environments” (atoms) to manage Dev, QA and Production deployments
7. “Extensions” for each environment control connectivity to individual system tenants for each development stage.
8. Encrypted endpoint passwords are stored exclusively on the target atom. Only privileged users can access the Production environment for deployment and credential management.
9. Try not to branch/copy process for Dev, QA Copied processes lose their associations with processes deployed to environments.
10. Treat first release of a process as the Production Copy. This eliminates the need to branch for production.
11. Deploying process to QA environment upon version release to QA.
12. Continue development on same process copy as required.
13. If branching is required, try to share unchanged components across versions.

2.2 Project Lifecycle Management

2.2.1 Environments

Name: Dev (optional)

Description: Used for development. Developers can use this or their local ATOMs for testing. By using a shared development environment, developers are able to share connector licenses.

Attachments: Customer or Developer Hosted ATOM

Name: Test

Description: Used for end to end testing. Jobs are deployed and scheduled to run regularly.

Cloud Attachment: Assign to Boomi Cloud or SF Boomi Prod Cloud Environment Variables: Should point to the SF test instance, test formIds and 3rd party test systems.

Name: Prod

Cloud Attachment: Assign SF Boomi Prod Cloud Environment.

Extensions: Should point to the SF production instance, production formIds and 3rd party production systems. Production credentials should only reside here and not be exposed in stand-alone connections in the Project Explorer.

2.2.2 Environment Extension Variables

Environment Extensions have several important purposes:

1. They allow seamless staged development, managing deployments from Dev to Test to Production
2. Packaging Integration Packs require extensions.
3. Share Connector Connection components and consistently expose the attributes in each process. At minimum expose endpoint, user name, password, and Adhoc connection timeout (SFAPI connector only)
4. Connection Settings - these allow definition of the Dev/Test/Prod instances endpoints and credentials.

   > Note
   >
   > You should not create copies of processes and connections for each stage of development. Extensions should be used and only one connection should be defined in the project and be shared by all processes.

5. Dynamic and "typed" Process Properties - Dynamic Properties are the old, non typed string properties.
6. Cross Reference Tables - are useful for doing value tranlations from source to target systems. In the case of SFAPI picklist values, we should be using the picklist common component and not Cross Reference tables. Otherwise changes to a picklist will break integrations.
7. Object Definitions and Data Maps - These will only be supported for the SFAPI when connector is public. It will not work for the Private Production connector. When available, this will allow for customization.

2.3 Overall Security Considerations

We need to ensure that we are meeting security requirements.

- For file level integrations, traditionally we encrypt data files that reside in SFTP directories. Files are decrypted prior to processing and clear versions are deleted after processing has completed. Boomi's Data Process component has capabilities to PGP Encrypt and Decrypt files. Note, even though the SAP SuccessFactors webservice API will eliminate this need for the SAP SuccessFactors side of any integration, file integration will still often be used on the third-party system side of any integration. For example, files are required to send sensitive payroll data to ADP and Ceridian.
- A Best Practice involving Certificates is to use separate ones for the Production and Test environments. Additionally, PGP certificates can be made configurable through the Extensions button when in the Process Build mode. Within the Atom Management page, developers can then adjust the Certificate used based on the Boomi environment within in the Environment Extensions link for that given environment you are specifying.
- Similarly, with keys, a developer can make use of the Environment Extensions variables to use different Connectors when adjusting between different environments. Within the Boomi Build mode, choose to use a Connection that specifies a Testing environment as a default, and then when deploying to Production, choose to override that Test Connection within Atom Management to correspondingly match Production. This will greatly minimize the potential of a security and data privacy issue resulting from using Production data in Test scenarios.
- Ensure that any logging that is implemented does not log sensitive employee data to the atom nor does it send email alerts that contain and sensitive data.
- Boomi accounts need to be audited for appropriate access for who has authorization to change and deploy processes to a production atom. Security reviews for bespoke process implementations may be in order.

- Solution templates will be reviewed for security considerations.
- The Boomi Find Changes component has deemed itself invaluable because SAP SuccessFactors cannot provide delta feeds that contain data modified since a specified timestamp anchor point. This component requires a history file that boomi stores on the atom in the /work/cdc directory. It is stored as unencrypted plaintext. We need to push boomi to support encryption of this file or implement our own component using Groovy and PGP.

2.4 Boomi Process Reporting Overview

- The Process Reporting Boomi screen is found under the "Manage" banner menu item, and it shows the Processes that are currently running or have previously been run. This section is very valuable for performing troubleshooting on any Processes that may have failed during execution. See Boomi Reference material for more details on the interface elements and their descriptions.
- When clicking on a Process that has been run, a new window pane appears on the screen showing details of specific "Inbound" and "Outbound" data from the Process execution run, broken out by all applicable Connections that were used. The individual record/document breakdown data that is displayed shows at least basic details, but can show customized fields as well based on usage of the Tracked Fields (please reference the related documentation section for more information).
- • View the Process Log information for even more details about the process execution.
  - o Pay particular attention to the logging statements regarding the Connector shapes. For the SAP SuccessFactors API Connector, very important troubleshooting details available include the environment that was requested, the exact query that was run, and any potential Object Form override variable that was in use.
  - Adding Notify shapes within Boomi Build mode allows you to print custom logging statements, which can give even more help in troubleshooting a Processes activity.

3 Boomi Processes

3.1 Overview of Boomi Processes

Boomi Processes are the graphical representation of the path that a document takes from the point at which it is received by Boomi, to the point at which it is sent to one or more destinations.

Using Boomi, you can:

- Use an inbound connector to retrieve data from one source. The source can be an on-premise or web application, or a data source such as disk, mail, HTTP, FTP, SFTP, or database.
- Perform various actions on that data by using execution steps, logic steps, maps, and map functions.
- Use an outbound connector to send data to one or more destinations. The destination(s) can be an application or a data source such as disk, mail, HTTP, FTP, SFTP, or database.

Boomi applies the Build concept as a means to organize and control data processing. Integration requires data structuring to enable communication between disparate applications. An integration service should allow structured data types to be extracted, manipulated, validated and forwarded. In order to integrate applications directly from the web without coding, Dell Boomi uses a built-in visual interface (a "visual designer") to create and direct process flows.

The Build page is where you are able to do the following:

- Create, edit and delete components
- Build integration processes
- Set up an Atom
- Set up a Molecule
- Access the reference guide

3.2 Creating Self-Describing Processes

This section describes authentication, authorization and integration with the API.

The graphical nature of Boomi lends itself well for creating self-documenting work products. Even so, more requirements are need to ensure that as we create reusable processes, the details of complex processes may not be obvious unless documentation disciplines are followed. A trained Boomi tech should be able to quickly understand what a process is doing without having to drill down to every Data Process and Map component.

Note due the the graphical and agile nature of Boomi development, it is also possible to avoid the process of developing a Design Document but instead, model your process in Boomi with extensive component descriptions and canvas notes for your initial design. Next generate your profiles and maps and you have a great start at your design. If necessary, export maps to XLS and take screen shots of your processes and you have a great start if more detailed formal documentation is required.

Create your work product so that when you look at it months from know, it will be clear what it does and how:

- **General Guidelines** - ensure descriptive names are used. Avoid abbreviations when possible and properly include both the abbreviation and the description, ie. Employee Central (EC).
- **Process Naming** - Names should call out the endpoints being connected to. For example SAP Payroll to EC Compensation. Ideally the title should include the entities and/or high level processes involved.
- **Component Naming** - Component titles often appear on the process canvas where they are especially useful so name accordingly.
- **Description/Subtitles** - Always fill in the "Enter Description Here" field/sub title for processes and components unless the component is so self explanatory, a novice will know what it's specific application is.
- **Use of Notes** - Use the Add Note capability to describe branches and other regions of the process. Some components provide very generic names on the canvas, for example "Data Process" so it is especially important to Add Notes near these components.
- **Cross Reference Tables** - If cross reference tables use cryptic codes as the input/output pairings, add a Description column that has human readable descriptions for each table entry. When using Cross Reference Lookup map functions, use descriptive values for the input/output parameters. This is because Boomi does no show the title of the lookup table when the function is displayed in the Functions area of the map.
- **Don't hide business logic**.-Expose business logic on process canvases where possible, as it usually is. Decision and Business Rules steps are designed for this purpose. Where necessary, business logic can be implemented in Custom Map Functions but proper naming and component descriptions are required. Note that for a newcomer analysing a process, it can be difficult to discover logic hidden in Map Functions.
- **Avoid Scripting** - It is tempting to implement logic in JavaScript or Groovy, especially for personnel proficient in such languages. This should be avoided because of the greater cost of ownership that results. It is usually the case where any scripted logic can be replaced with graphical Boomi components, steps and standard Map Functions.

When it is determined that scripting is absolutely necessary, consider implemented a general use component that can be unit tested, shared and maintained separate from the project. This is the case for the SAP SuccessFactors Common Component library that is reused for processes across many customer accounts with functions varying from sorting, de-duping, date arithmetic and picklist handling.

3.2.1 Process Naming Conventions

Constant naming conventions are an important element for minimizing cost of ownership of a project. The obvious benefits are to help personnel maintaining project to have a shorter learning curve during their first encounter with it. Some other factors driving the import of proper naming are:

- Process names are exposed in Process Reporting and Alerts hence they have exposure to broad audiences
- Process and Component names are exposed in Integration Packs, Process Extensions, and Process Libraries. Naming in this case because a topic for Product Management as they can be exposed to customers. Names and prefixes are important as all extensions in an environment are listed together. In order to decipher which extensions go with which process or Integration Pack, a consistent prefixing mechanism is required. Component Names can be exposed as extensions and which will be mixed with other extensions in an environment. It is important to know what extension goes with what process.
- When viewing Boomi Manage -> Process Reporting, process names appear in the logs.

- Main processes must be deciphered from “No Data” sub processes which will have their own entries. (No Data sub processes have the Start step configured with No Data vs. Pass Through and are granted their own line in the Process Reporting Log)
- You may have multiple main processes running at the same time. In this case multiple “No Data”. Naming conventions are required to visual associate which processes are related and which are sub vs. main processes.

One general guideline of a component or step name is that it should be meaningful and self-explanatory to any newcomer. For example, a process that implements an employee extract from Employee Central to Microsoft Active Directory should not be named “AD”.

> ⓘ Note
>
> This example is called out because such a process name was found in an actual customer account.

Name: {System Name or Abbreviation)} {Process Definition}

Description: {Detailed description of the process}

Examples: PeopleAnswers Order Request (in this case, “PA” might not be an ideal process name without more context)

ADP Payroll Extract

LN to SF Background Search Results Import

Don’t use “Main” as a prefix. Only use “Sub” as Main simply wastes real estate on the Process Reporting page.

3.2.2 Sub-Process Naming Conventions

When examining a log file, it should be obvious which sub-processes go with which main processes.

Name: **Sub**: {System Abbreviation)} {Process Definition}

Description: {Detailed description of the sub process including parameters passed in if any}

Examples:

Sub: SF Load Document Cache

Sub: LN Background Search Results Import

Sub: NS Results Import

Sub: SAP Cost Center Extract

3.3 Common Design Patterns

The use of Common Components and Reference Library templates follows the software engineering practice of using design patterns. Rather than reinvent the wheel for every project, apply existing design patterns to solve an integration problem. Others familiar with the design pattern will thank you later when they take over maintenance of the process.

A common design pattern within the fold is the Message Transformation design pattern applied to Recruiting Assessment orders. This use case accepts an inbound order event to produce an assessment order in HR-XML format. This pattern can be reused for everything from drug screening orders to online competency assessment exams. The basic pattern requires intercepting a message from a Job Application, joining in Job Requisition and other information, sending the HR-XML to the 3rd party system and updating the Job Application with status information. This pattern can be reused from everything to People Answers, SHL and Talent Plus. Having all these implementations re-invented for every application is not practical.

3.3.1 Scripting for XML Parsing and Modification

XML Files can be manipulated by parsing the file using the Data Process Step. The incoming XML file can be made pass through the Data Process with custom scripting added to it. In the custom scripting, we can traverse through the XML file by targeting a particular node or XPath for Manipulation.

3.3.2 Testing Mode

Test mode allows you to quickly test processes from within the Build menu without needing to deploy the process to an Atom and wait for results to be posted back to the Manage menu's Process Reporting page. Test mode is intended to provide immediate feedback on process configuration for use during initial development or production debugging. Test mode provides results and detailed logging information for each step to assist with troubleshooting.

There are a few important things to understand about Test mode:

- When executing a process in Test mode, the process actually runs. Data actually runs through the process so records will be moved, created, modified, deleted, etc. according to the process configuration.
- Run the process in the Test Atom Cloud or install a local Atom on which the process can run during development and testing. The process does not need to be deployed to that Atom. You cannot deploy processes to the Test Atom Cloud. (For more information see topics about deploying processes.)
- Test the Start step by itself to verify connection credentials and to verify that the filters and/or criteria you specified are returning what you expect.
- If you enter Test mode and you have made a change to the process without saving it, you will receive a message. You can choose one of the following: Revert to and reload the previously saved process (but your recent change will be lost), save your recent change and proceed to Test mode, or cancel Test mode so that you can continue to work on your recent change.
- The log and data files produced during Test mode are temporary. Once you exit Test mode this information is lost. Test mode executions are not displayed on the Manage menu's Process Reporting page.
- Test mode is limited to a maximum of 10 files (documents) per inbound connector step. If more than 10 files or documents are retrieved by an inbound connector, an error message appears. If this error occurs on the Start step, you will have the option to Retry just the 10 documents that are originally picked up and forward them through the rest of the process. If this occurs on an inbound Connector step later in the process, you will

receive a standard error message. You should configure your connector accordingly to process less data for future tests.

- Test mode is limited to a maximum aggregate data size of 1 MB across the entire process execution. If the size is exceeded, an error message appears. You should configure filters and/or other selection criteria in the inbound connector operations or profile to limit the size of data read into the process.

3.3.3 Further Reading: Enterprise Integration Patterns

This book provides excellent treatment of the Design Patterns topic, in general:

Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions Gregor Hohpe, BOBBY AUTOR WOOLF

3.4 Custom Scripting Guidelines

Groovy should be used, not Javascript. This allows for a consistent skillset required for maintenance and provides the power of Java libraries to scripts. The one exception is when you want to dynamically execute code stored in Cross Reference Table Extensions. Only Javascript can be used for this design pattern.

Scripting should be avoided. First try to use standard boomi functions, common shared components, and/or cross reference lookup tables.

If you still must use custom scripting, attempt to develop a general purpose script that can be reused for other projects.

Keep business logic on the canvas vs. buried in custom map function if possible which it usually is.

Note for reusable integrations, it is a good idea to try to use Cross Reference Tables for business logic since they can be exposed as an Environment Extension. In these cases, note that the Business Logic step supports Cross Reference lookups.

3.5 Error Logging

Boomi has powerful error logging and alerting functionality. Errors are categorized by two types:

1. 1. Infrastructure errors due to down systems, connectivity issues, etc.
2. Data errors due to missing mandatory fields, mis-formatted values, value length exceeded, etc.

Ideally these errors should be reported with different categories though that is not always easy. The best way to report errors is to use the Alert step. This can provide values in email alerts as well as the Process Reporting log file.

3.6 Error Handling

- Use a Try Catch step to the left of map steps. Do this in conjunction with having data restrictions declared in the Data Profiles (minimum length, etc). The Catch branch should be directed to an Alert where the following can be reported:
  - Current Data - the data document that caused the error. Please consider security/data privacy when using this element. It might be required to capture only specific profile elements such as employee id, email address.
  - Try/Catch Error Meta Document property - this contains a description of the error.
  - The default number of tries should be configured. It is generally recommended to have the number of tries set to a minimum of 4.
- For the profiles after a map, please make sure to follow the specifications document in changing the min occurs/ max occurs. This can be used to reject records if the min occurs of any field is 0.
- Avoid using business rules for rules that can be handled in XML/Flatfile profile restrictions.

3.7 Data Cleansing

Detecting data errors can be done in several ways.

3.7.1 Cleanse Component

- This allows data elements to be cleansed based on restrictions defined in the respective Data Profile
- The Rejected branch can be directed to an Alert step similar to above.

3.7.2 Decision Component

- Direct the rejected branch to an Alerts
- Note the Business Rules Meta Document Property contains messaging information
- Note when implementing decisions for Connector Operation results, it is useful to store the Request document into a document property to the left of the operation. A decision can be used to detect the error in the Response document. The Alert can then lookup elements in the Request document

4 Boomi Sub-Components

4.1 Overview of Boomi Sub-Components

Boomi Sub-Components (also referred to as "steps") perform all the individual activities found within a Process. This section covers many of the available components in more detail, including naming conventions and best practices when using these components when building SAP SuccessFactors integrations.

4.2 Connector Connection

4.2.1 Naming Conventions

Connection naming is flexible due to use of Environment Extensions. When using Boomi environments to manage connection credentials and other settings, the connection contents will be variable based on the target deployment environment. You can include endpoint and tenant ID Info or keep generic. One thing to note is that the connection(s) in the Component Explorer will be used for Run as Test and for operation Import functions.

Name: {System Abbreviation} {Connector Type}

Connection Description: {Short description of the connector use}

Examples:

- SF Instance Connection
- SF SFTP Connection
- ADP FTP Connection

4.3 Connector Operation

4.3.1 Naming Conventions

Name: {System Abbreviation (optional)*} {Entity Name (optional)} {Operation Use(optional)} {Action}{Connector Type} Operation

Description: {Short description of the connection operation} *System abbreviation only required for generic connectors such as Web Services Client or SFTP connectors.

Examples:

- Hireright Create Order
- Put Overwrite SFTP Operation
- Put Re-name FTP Operation
- ManageContactIn MaintainBundle EXECUTE

4.3.2 Operation Filters

Operation filters are those which are used to filter the data based on the criteria that is being provided. Depending on the report/data that is being pulled, we can add any number of filters. Filter fields will be selected whenever the operations are being developed.

You can create multiple filters and add multiple parameters and mix static and dynamic values. One common dynamic value to use in the Start step is to create a filter that extracts records where the "last modified date" is greater than some value, and pass in the special Last Run Date value as the parameter. Steps to add filters:

- Create or edit an Operation component.
- Click the Options tab.
- (Optional) Click the Import button and use the wizard to generate the XML profile and configure the operation.
- At the bottom of the page, add a filter.
- Configure the filter with a user-defined name, the field to filter against and a comparison operator.

4.3.3 Filter Naming Conventions

Name: {fieldname) {Operator} {Expected Value in ALLCAPS (optional)}

Examples:

- lastmodified>=TODAY
- paycomponent=BASEPAY
- picklistId= Filter values can also be provided based on process properties. Here is an example for the same:
- lastmodified>=TODAY
- paycomponent=<PROCESS_PROPERTY_PAYCOMPONENT>.
- picklistId=

In the above example, a new process property named PROCESS_PROPERTY_PAYCOMPONENT will be added before the connection object is created, which will provide values to the paycomponent filter.

4.4 Profiles

```
Connector Type> <Instance Name>* <Operation> <"Request" or "Response"> ie. SAP By
Design ManageContactIn MaintainBundle EXECUTE Request
```

4.4.1 Profile Constraints

4.4.2 SF Connector Request Profiles

SF Connector Request Profiles will be generated if the Boomi process is trying to UPSERT data into the BizX. In no other scenarios there will be an Request Profiles being generated. In such an UPSERT scenario, there will be both Request and Response profiles generated.

4.4.2.1 Naming Conventions

Name: {System Abbreviation} {Entity Name} {Operation}

Request Description: {Short description of the request profile}

Examples:

- SF User Update Request
- SF Job Application Update Request

4.4.3 SF Connector Response Profiles

SF Connector Response Profiles normally will be generated in response to a Query Operation. If Boomi process has to produce an output file, first step is to identify the fields that are part of the Output and generate a Query Operation to produce the output file.

Response Profile will hold the output of the Query that was made.

Response profiles will also be created when Boomi is performing an UPSERT operation in the BizX.

4.4.3.1 Naming Conventions

Name: {System Abbreviation} {Entity Name} {Operation} Response

Description: {Short description of the response profile}

Examples:

- SF User Query Response
- SF Picklist Query Response

4.5 Maps

4.5.1 Map Naming Convention

`<Source Profile Name> To <Target Profile Name>`

Instance specific required only if customized and more than one instance applicable in the Boomi account. When imported from a specific tenant, It is a good practice to include the tenant ID and possibly the version number (ie 1210) used for the import operation. If not in the title, at least include this information in the description.

Examples:

- If you are creating an ADP Passthru 01 output file from the ADHOC ECTINT profile, you can use the mapname as:
- ADP PassThru 01 Output – ADHOCECTINT TO 01FlatFile
- If you are creating an Output File for the Benefit Focus from the ADHOC ECTINT profile, you can use the mapname as:
- BenefitFocus Output – ADHOCECTINT TO BFFlatFile
- In some scenarios you have to process the output file further. Let us consider an situation, where after generating the 02 ADP PassThru file, you want to process it further to validate some of the fields based on the first process output you can use the mapname as:
- ADP PassThru 02 Output –Field Processing – ADHOCECTINT TO 02FlatFile

4.6 Cross Reference Tables

The cross reference table is a data structure that is used to replace a run-time lookup computation with a much simpler lookup operation. The gain in processing speed can be significant, because retrieving a value from memory is much faster than performing a database or other connector lookup.

Cross reference table lookups are most often performed in map functions (under the Lookup category) but can also be used as parameter values in all process steps that use parameters, such as connectors (including the Start step), Decision, Set Properties, Message, Program Command, and Exception steps.

Some common uses of a cross reference table are:

- A simple value translation between System A and System B, such as item codes, units of measure, status codes, or any other type of code
- Reusable translations (for example, U.S. state abbreviations)
- Switch/case logic (simple if/else)
- Atom-specific map default values
- Atom-specific connection default values (Start step criteria)
- To parametrize any process step with atom-specific values for deployment to multiple locations or customers

The cross reference table allows you to retrieve one or more "values" based upon the value of one or more "lookup references". However, within the Cross Reference Table component configuration, you do not designate the lookup reference(s) or value(s). These are chosen when you reference the cross reference table within a map function or other step parameter. This allows the same column(s) to operate as either the lookup reference or the value, depending on the context.

The cross reference table is easy to use and requires no coding. It is comprised of a set of data elements (or values) that are organized using a model of rows and columns.

5 Change History

Learn about changes to the documentation for SAP SuccessFactors HCM suite Boomi Development Standards Guide in recent releases.

2H 2023

Type of Change	Description	More Info
Changed	We moved the Change History to the end of the guide.	Introduction [page 4]

Q2 2017 – Present

Type of Change	Description	More Info
None	We did not update this document.	

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
    - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
    - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

www.sap.com/contactsap

THE BEST RUN SAP