

ORACLE

Visual Flow Designer Fundamentals

Session Agenda

- 1 About dialog flows
- 2 Visual Flow Designer overview
- 3 Visual dialog mode architecture
- 4 Tips

Session Agenda

- 1 **About dialog flows**
- 2 Visual Flow Designer overview
- 3 Visual dialog mode architecture
- 4 Tips

About dialog flows

Define the narrative of a bot-user conversation in a skill

- Scripts the happy and unhappy paths

A dialog flow typically begins when a user message resolves to an intent

Use dialogs to prompt for user input or to display messages

Update variables

- User input
- Entity slotting

Remote backend queries and updates

Expense my business trip to New York in May 2022

I understand you want to create an expense report for a "business trip to New York". Did I get this right?

Yes, you did get this right

No, let me change the title

Yes, you got it

From your message I am taking it that the business trip was in May 2022. Shall I use this date?

Yes, use May 2022

No, let me change it

I want to change it

Ok. So, what do you want me to put as a date?

May 15 2022

About dialog flows in digital assistant

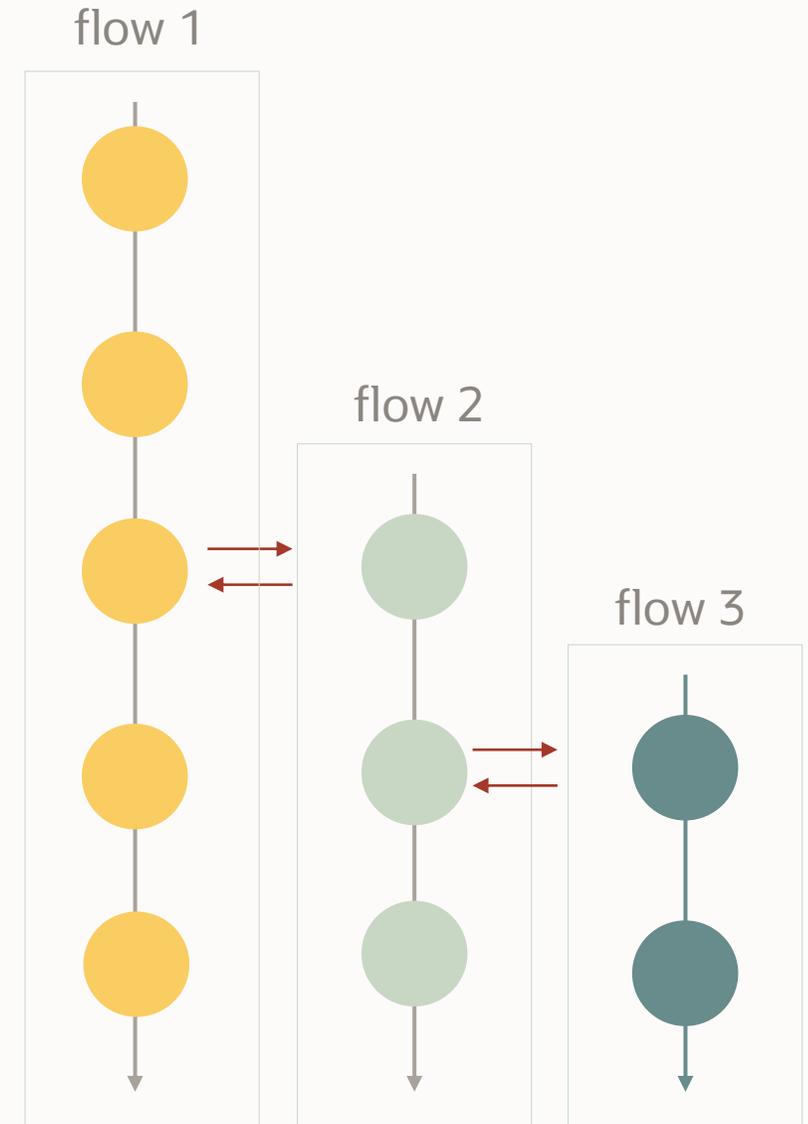
Flows make the conversation

Dialog flows in Oracle Digital Assistant can consist of a single flow or multiple flows chained together through invoke-flow calls

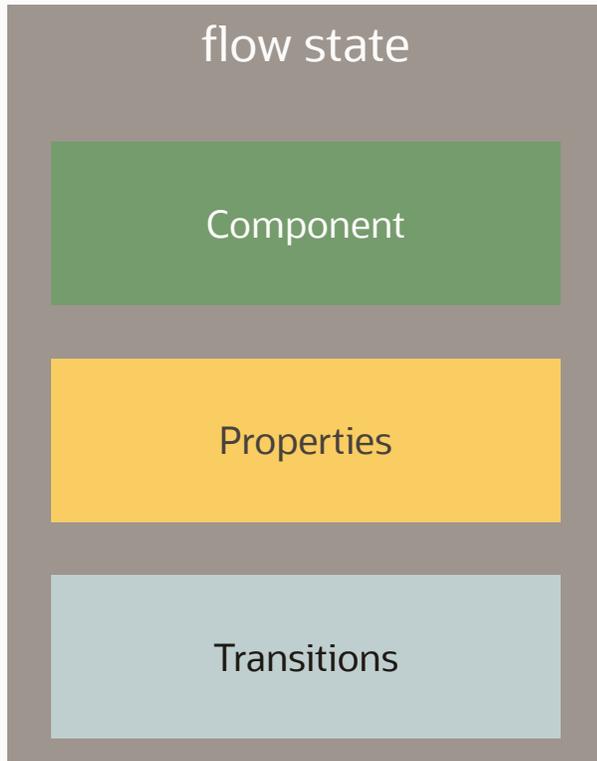
Flows consist of flow states that are visited in a specific order determined by user interaction with a skill

Each flow state refers to a configured component to

- render a user input prompt
- display user messages
- perform logical or functional operations



Flow states



flow states execute logic or render a user interface

Component types

- Built-in
- Custom

Properties

- Pass information into a component
- Used to update variables defined in a flow

Transitions

- next – navigates to a pre-defined next state
- action – conditionally navigates to a next state

About the Visual Flow Designer

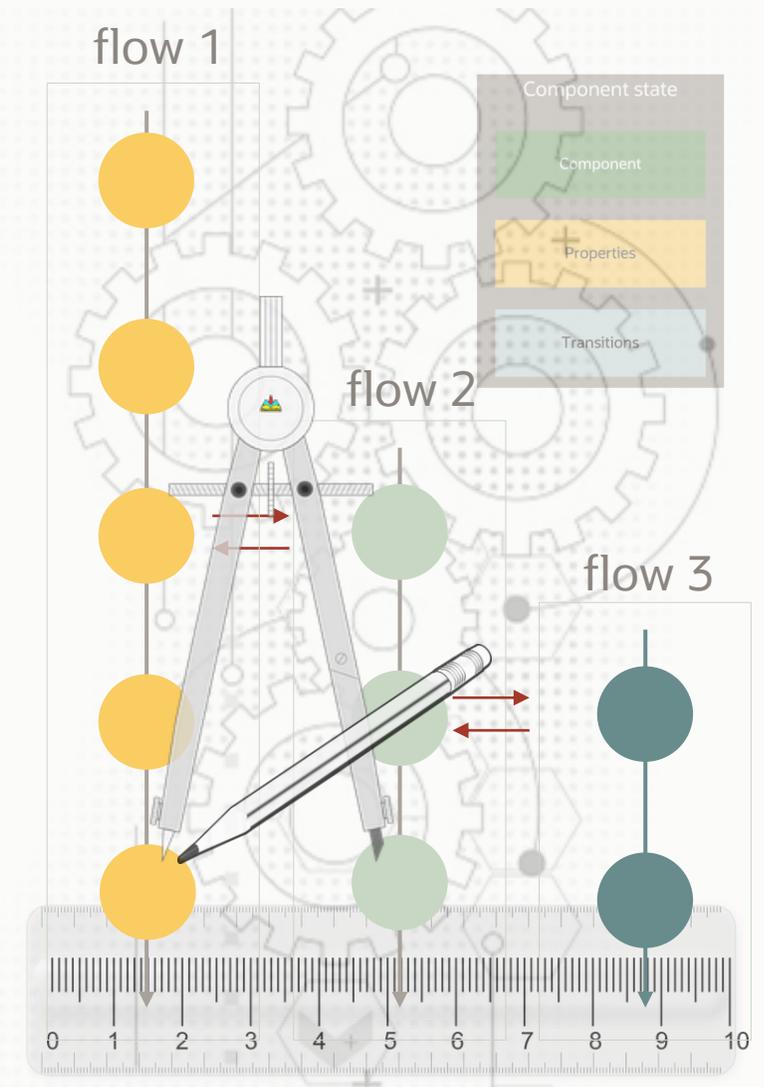
Declarative and visual tool to create and modify flows

Template driven state creation

- Driven by what you want to do (e.g., send a message)

For the component used in a state it does

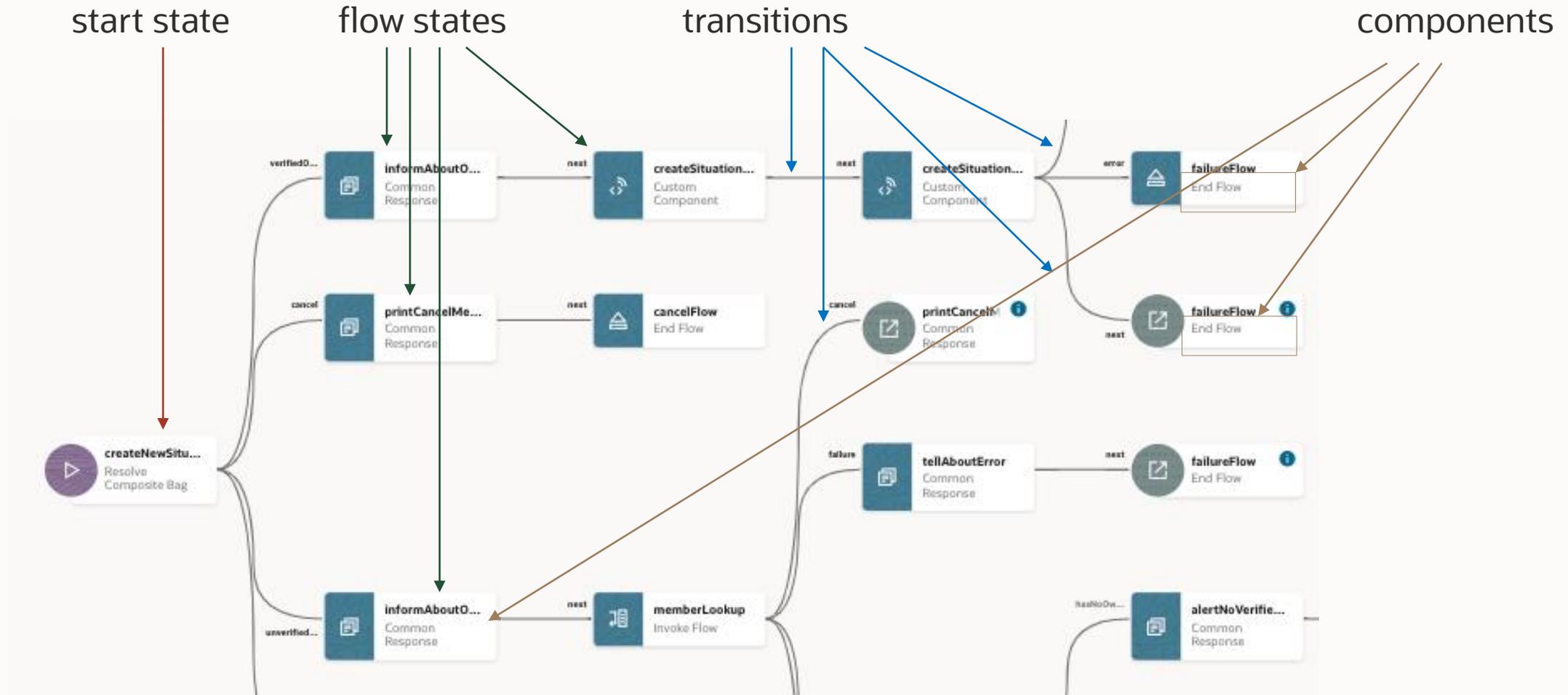
- Introspect and displays component properties
- Supports static values and expressions (Apache FreeMarker)
- Allows creating variables that will be updated by the component



Session Agenda

- 1 About dialog flows
- 2 **Visual Flow Designer overview**
- 3 Visual dialog mode architecture
- 4 Tips

Visual flow designer example and terminology



Getting started

Create Skill ×

Display Name

Name

Skill Version Platform Version

Dialog Mode [?]
 YAML Visual

Primary Language (Natively-Supported) [?]

One-Sentence Description

The visual flow designer is available in skills that were created using the **Visual** dialog mode option

The dialog mode option cannot be changed once the skill is created

YAML option no longer recommended

- Code centric development

Main Flow

The screenshot shows the 'Flow Designer' interface with two tabs: 'Flow Designer (2)' and 'Main Flow'. The 'Flow Designer (2)' tab is active and contains a '+ Add Flow' button, a 'More' dropdown, a 'Filter' input field, and a 'Sort By Alphabetical' dropdown. Below these is a list of flows, with 'Main Flow' highlighted in green and 'UnresolvedIntent' listed below it. The 'Main Flow' tab is also visible, showing 'Events' and 'Skill Variables' sections. The 'Events' section has an '+ Add Event Handler' button and a list of event categories: '> Intent Events (0)', '> Built-In Events (1)', and '> End Flow Action Events (0)'.

Always there, cannot be deleted

Maps intents and other events to flows

Allows to define global variables

- Variable values are kept until user session expires

Can pass information to mapped flows using input parameters



Creating a flow

+ Add Flow More ▾ Events Skill Variables

Create Flow

General

Name
intent.reg.expenses.create

Description
Implements the process of creating an expense report and optionally creating expense items

Intent Name ⓘ
reg.expenses.create ▾

Advanced Configurations

Parameters (0) **+** Add Parameter ▾

No parameters are available

Open created flow afterwards

Input
Output **Create**

There is no limit in the number of flows that can be created within a skill

Flows can be mapped to an intent name but don't have to

- Flows mapped to an intent are started when the intent resolves in response to a user message

Flows can call other flows, pass information and receive information

- Input- Output parameters

Creating input, output parameters and variables

intent.reg.expenses.create

Flow **Configuration**

> General

+ Add Parameter ▼

> Parameters (1)

+ Add Variable

> Variables (3)

> Events Mappings

Use Configuration tab visible when selecting a flow

Input parameters and variables

- Must have a unique name within the flow
- Have a type: primitive, complex, entity
- Can have an initial value set

Input parameters and variable values defined for a flow are reset when exiting the flow

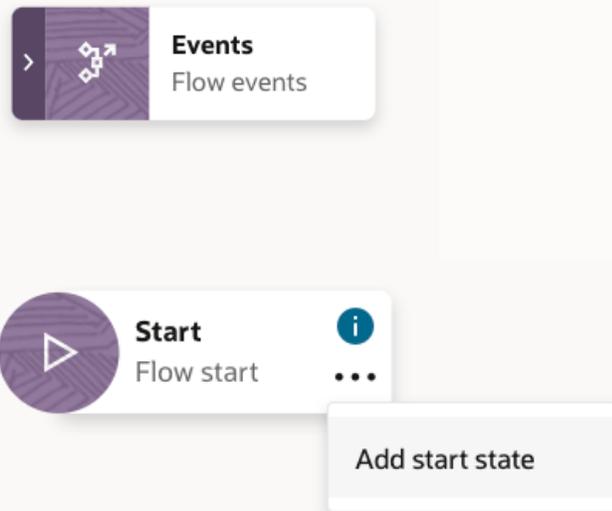
Output parameters

- Have a type: primitive, complex, entity
- Declaratively referenced when ending a flow

Adding states to a flow

intent.reg.expenses.create

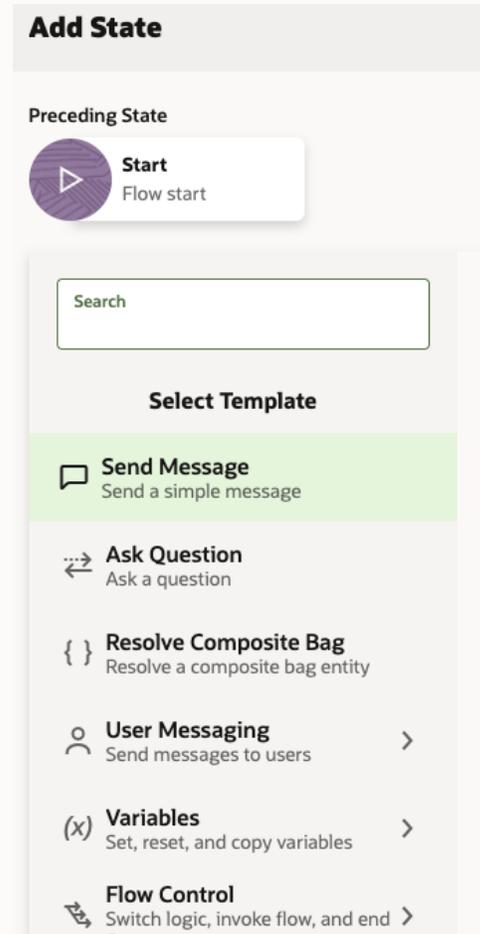
Flow Configuration



The screenshot shows a configuration interface for a flow. At the top, there are two tabs: 'Flow' (selected) and 'Configuration'. Below the tabs, there are two state cards. The first card is titled 'Events' with the subtitle 'Flow events' and a purple icon of a play button with a plus sign. The second card is titled 'Start' with the subtitle 'Flow start' and a purple play button icon. To the right of the 'Start' card is an information icon (i) and a three-dot menu icon. A context menu is open over the three-dot menu, showing the option 'Add start state'.

Choose "Add state" on the "Start" state to begin composing the conversation flow

Adding states to a flow

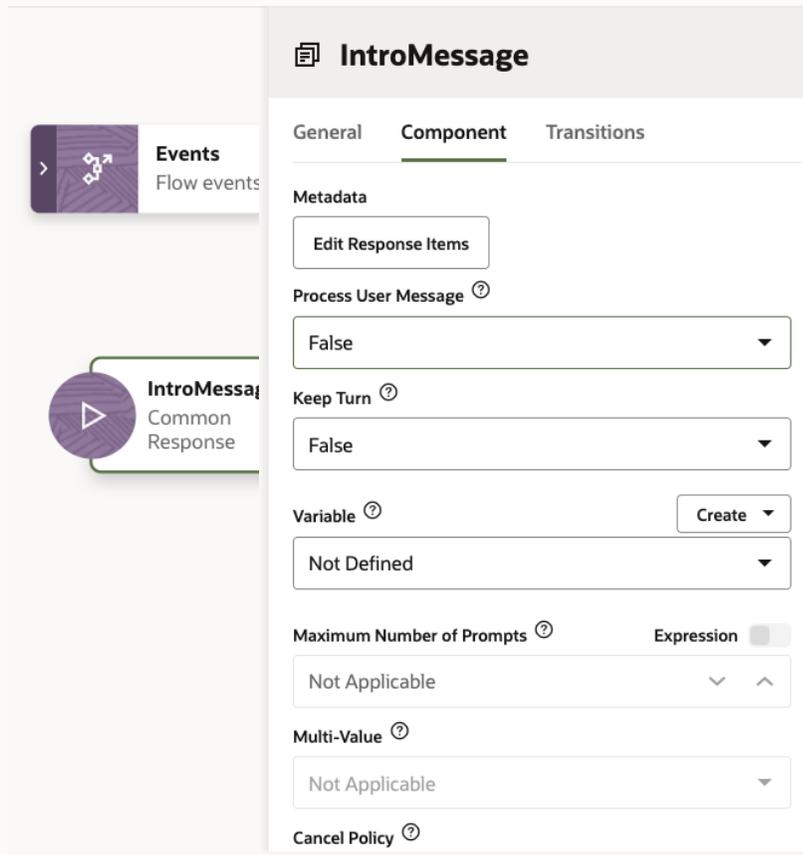


Choose "Add state" on the "Start" state to begin composing the conversation flow

A library of state template allows developers to select the functionality to add

- E.g., display a message, set variables, invoke a subflow

Adding states to a flow



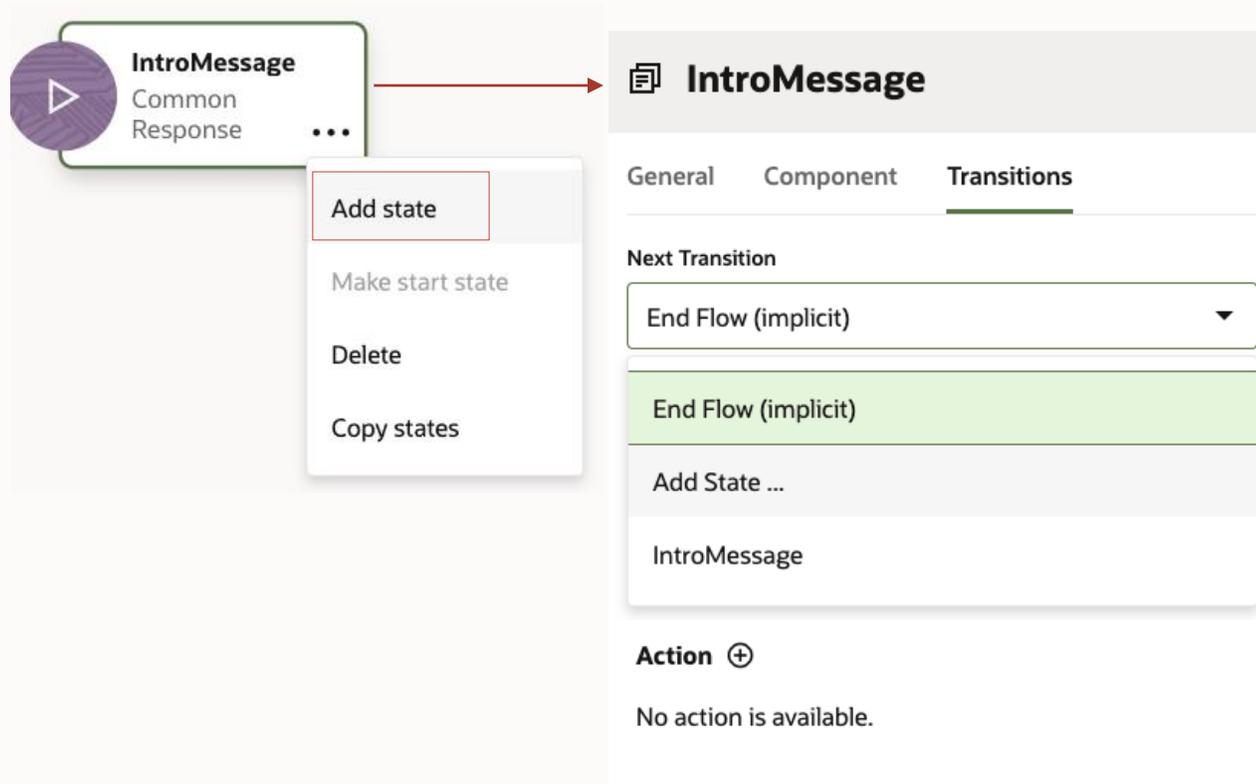
Choose "Add state" on the "Start" state to begin composing the conversation flow

A library of state template allows developers to select the functionality to add

- E.g., display a message, set variables, invoke a subflow

Selecting the added state displays a properties pane to configure the state's component properties and next navigation target (transition)

Adding transitions



The screenshot illustrates the process of adding a transition to a state in Axure RP. On the left, a state card for "IntroMessage" (Common Response) is shown with a context menu open. The "Add state" option is highlighted with a red box. An arrow points from this menu to the "Transitions" tab of the "IntroMessage" component's configuration panel on the right. In this panel, the "Next Transition" dropdown is set to "End Flow (implicit)". Below the dropdown, a list of available transitions is shown, with "End Flow (implicit)" selected and highlighted in green. Other options include "Add State ..." and "IntroMessage". At the bottom, the "Action" section shows a plus icon and the text "No action is available."

Transitions determine the next state to visit within a flow

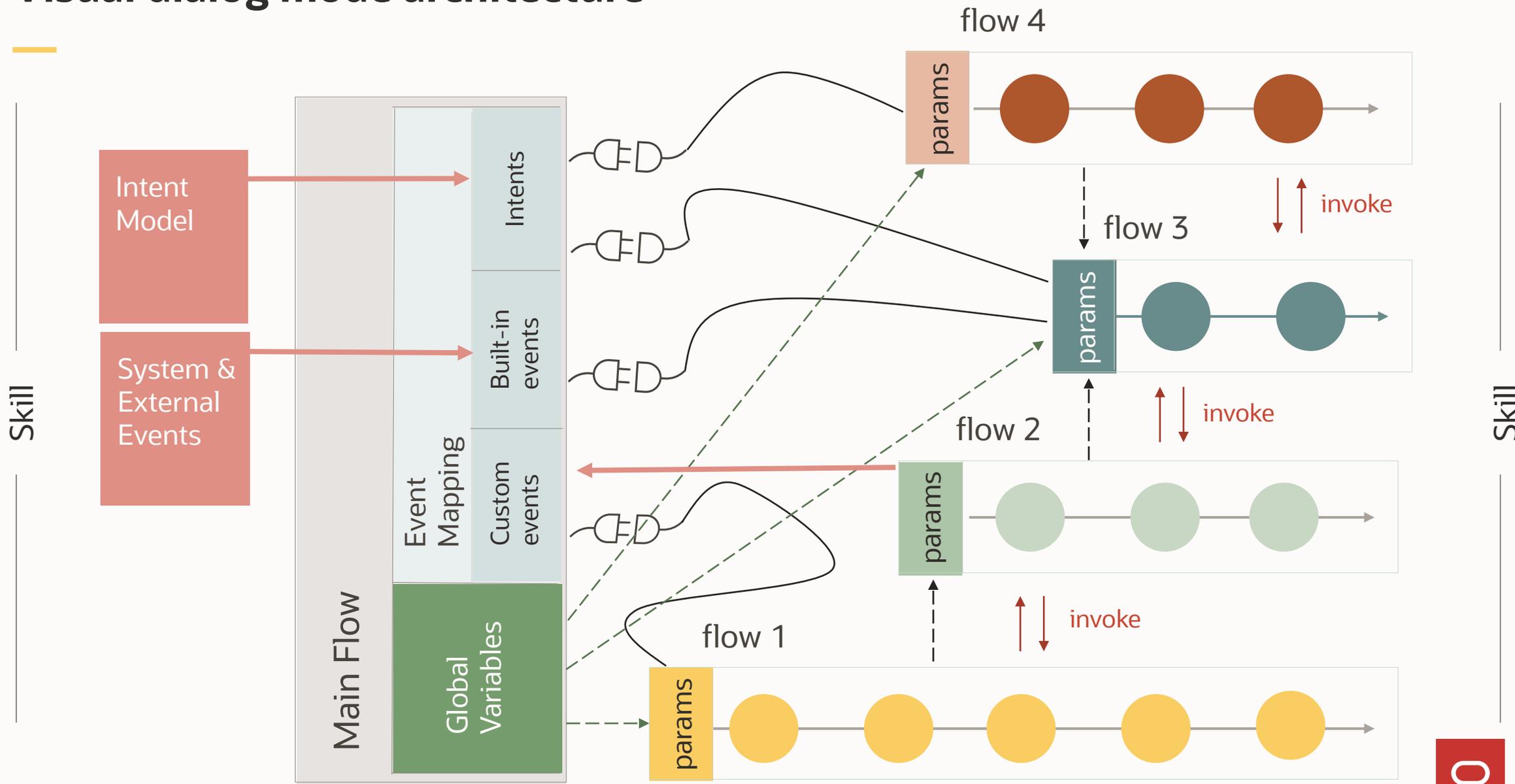
- Static navigation using "Next Transition"
- Conditional navigation using "Actions"

Components associated with a state determine the available actions

Session Agenda

- 1 About dialog flows
- 2 Visual Flow Designer overview
- 3 **Visual dialog mode architecture**
- 4 Tips

Visual dialog mode architecture



Session Agenda

- 1 About dialog flows
- 2 Visual Flow Designer overview
- 3 **Visual dialog mode architecture**
- 4 Tips

Recommendation for getting started

” Build conversation flows incrementally

Select a conversation to build

- E.g., file an expense, check orders

Create the intent for this conversation

Create entities needed for and in the conversation

Create a flow for the intent

Create and call subflows for conversation logic that can be reused from other flows

Build self contained flows

” Think 'modularization' when building flows

Use input parameters and output parameters for communication between a parent and the subflow

Use action strings in end flows to indicate to the parent flow what to do next

ORACLE