

ORACLE®

# Oracle Digital Assistant

## The Complete Training

### The System.ResolveEntities Component

# Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

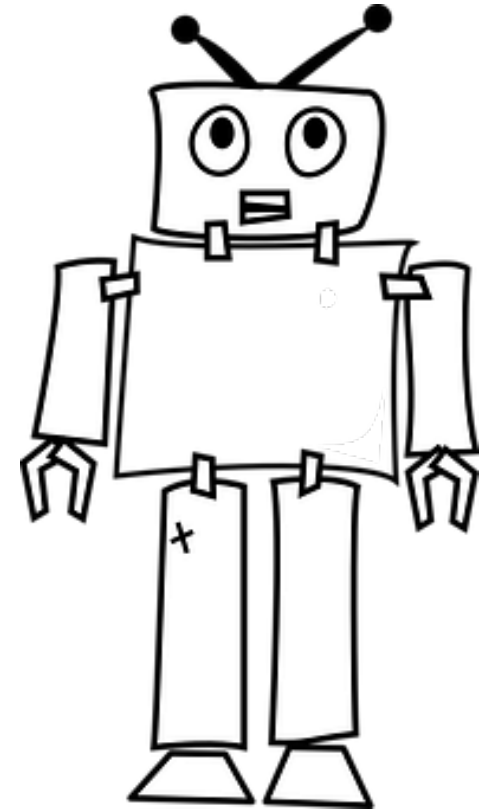
# Topic Agenda

- 1 Entity derived conversations
- 2 Component overview
- 3 Use with composite bag entities

# Topic Agenda

- 1 Entity derived conversations
- 2 Component overview
- 3 Use with composite bag entities

Dialog flow is the conversation script that is followed by a skill in a user interaction. However, the **best dialog flow is no dialog flow.**



# What wrong with dialog flows

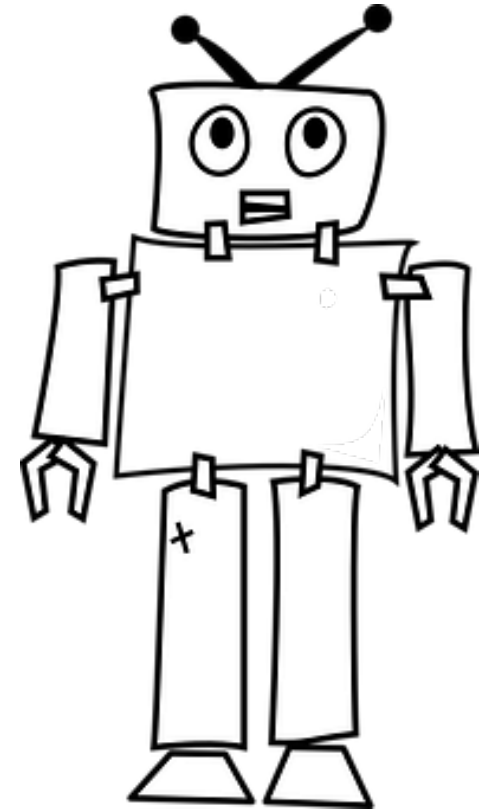
- Users are not good at giving a single answer to a question
  - In human-to-human interaction it is natural to overload answers with information
    - Bot: "what pizza type do you like?"
    - User: "a large salami with extra cheese"
- 'story telling' vs. 'data driven'
  - Natural conversation design is chatty
  - Skills only need data input to complete a task
- Violates the DRY principle (don't repeat yourself)
  - Bot response configured on component
    - Prompt, error message, validation, range size etc.
  - No reuse of settings if configuration is on the component

# Entity driven bot conversations

- Reduce the amount of dialog flow steps to write at design time
- Dynamically generate UI at runtime
  - Bot UI rendered based on entity type
    - Simple entities have a single user prompt
    - Composite bag entities may prompt users multiple times
  - All configurations and behaviors are defined on the entity
    - Prompts, error message, validation rules, range size
    - Entity extraction, out-of-order message handling (composite bag entity only)
- Oracle Digital Assistant promotes entity derived conversations
- Require use of `System.ResolveEntities` and `System.CommonResponse` components



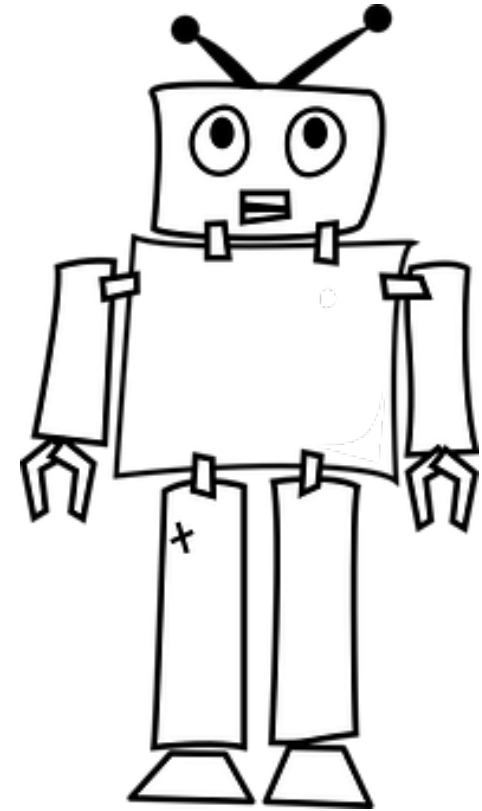
Entity driven conversation **delegates common component configurations to the entity** level, which is a much better model for reuse.



# Topic Agenda

- 1 Entity derived conversations
- 2 Component overview
- 3 Use with composite bag entities

You can use the `System.ResolveEntities` component with system, custom and **composite bag entities**. It generates input fields (prompts) and value lists.



# Building System.ResolveEntities from component template

The screenshot shows the Oracle ADF Components palette on the left and the Component Template editor on the right.

**Components Palette:** The palette is titled "Components" and shows a list of component types. The "User Interface" category is selected, and the "Resolve entities" component is highlighted with a red box.

**Component Template Editor:** The editor is titled "Component Template" and shows the template for the "Resolve entities" component. The template is a JSON object with the following structure:

```
resolveEntities:
  component: "System.ResolveEntities"
  properties:
    # variable (required) refers to the composite entity context
    variable:
      # nlpResultVariable (optional) refers to the nlresult variable
      # that can be used to resolve (part of) the composite entity variable.
      # If the nlpResultVariable value contains an entity match of the same
      # type as one of the child entities of the composite entity variable,
      # then this child entity value will be set inside the variable value.
      # If all child entities are populated by the entity matches in the
```

The editor also includes a sidebar with a list of component types: Interactive, List - set action, List - set variable, Output, Resolve entities (highlighted), Text, and Webview. At the bottom, there are controls for "Insert After" (set to "handleMaxPromptsExc..."), "Remove Comments" (a toggle switch), and an "Apply" button.

# System.ResolveEntities component with custom entity

The screenshot displays the Oracle APEX interface for configuring the `System.ResolveEntities` component. The interface is divided into several sections:

- Entity List:** A list of entities including Travel, Airports, ADDRESS, CURRENCY, DATE, DURATION, EMAIL, NUMBER, PHONE\_NUMBER, SET, TIME, URL, and YES\_NO. The 'Airports' entity is selected.
- Description:** A section for describing the entity, with fields for Name (Airports) and Description.
- Configuration:** A section for configuring the entity, including a Type (Value list) and a Value list (LAX, SFO, LHR, MUC, CDG).
- Enumeration Range:** A section for setting the Enumeration Range (3).
- Prompts:** A section for adding prompts, with a prompt text 'Please provide an airport code'.

The configuration is then applied to the `System.ResolveEntities` component, which is shown in the center of the image. The component configuration includes the following properties:

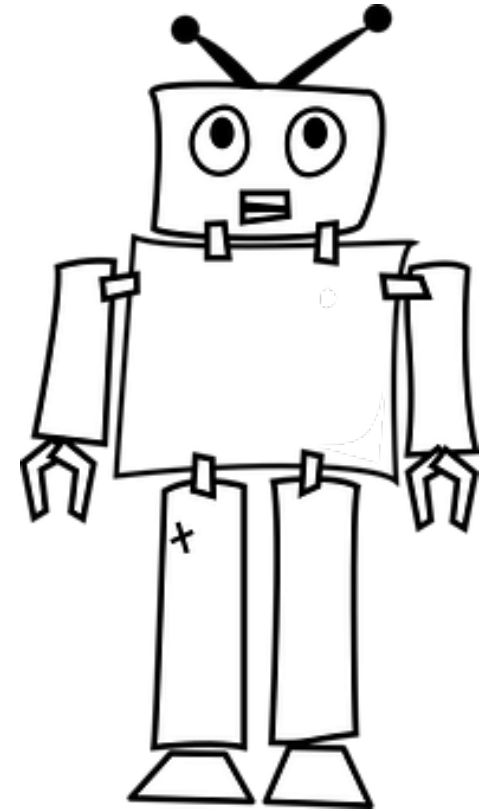
```
context:
  variables:
    airports: "Airports"
    iResult: "nlresult"
states:
  showAirports:
    component: "System.ResolveEntities"
    properties:
      variable: "airports"
      nlpResultVariable: "iResult"
      maxPrompts: 1
      cancelPolicy: "immediate"
      transitionAfterMatch: "true"
      autoNumberPostbackActions: false
      headerText:
      footerText:
      showMoreLabel: "Show More"
      translate:
    transitions:
      actions:
        match: "handleEntityMatch"
        cancel: "handleFailedValidInput"
```

The component is then used in a page, as shown on the right. The page displays a prompt 'Please provide an airport code' with a list of airport codes (LAX, SFO, LHR) and a 'Show More' button. The page also includes a 'Message' section with a link icon.

# Topic Agenda

- 1 Entity derived conversations
- 2 Component overview
- 3 Use with composite bag entities

You need to train **the skill bot model** before using `System.ResolveEntities` with composite bag entities.



# System.ResolveEntities with composite bag entity

The screenshot displays the Oracle APEX interface for configuring a composite bag entity. The left sidebar shows a list of entities: Travel, Airports, CabinClass, ADDRESS, CURRENCY, and DATE. The central configuration panel is titled "Description" and "Configuration". The "Configuration" section shows the "Type" set to "Composite Bag" and a table of "Bag Items".

Name	Type	Entity Name
DestinationAirport	ENTITY	Airports
DepartureDate	ENTITY	DATE
Cabin	ENTITY	CabinClass

The right sidebar shows a JSON configuration snippet for the "System.ResolveEntities" component. The snippet is as follows:

```
context:
  variables:
    booking: "Travel"
    iResult: "nlresult"
states:
  showAirports:
    component: "System.ResolveEntities"
    properties:
      variable: "booking"
      nlpResultVariable: "iResult"
      maxPrompts: 1
      cancelPolicy: "immediate"
      transitionAfterMatch: "true"
      autoNumberPostbackActions: false
      headerText:
      footerText:
      showMoreLabel: "Show More"
      translate:
    transitions:
      actions:
        match: "handleEntityMatch"
        cancel: "handleFailedValidInput"
```

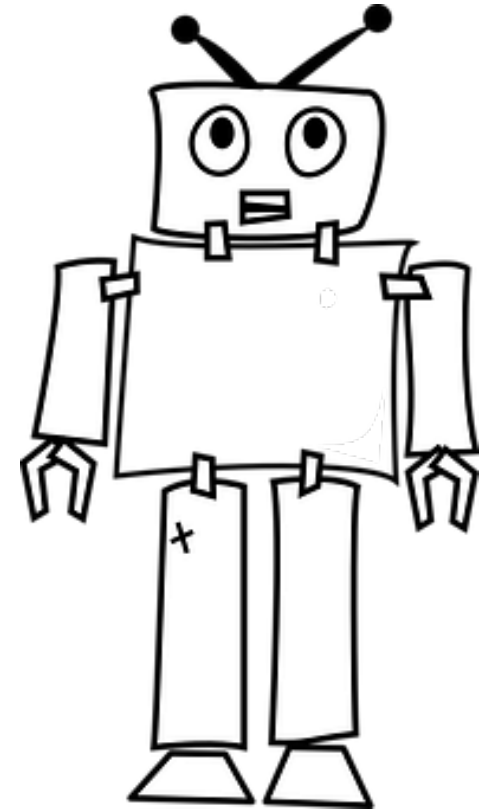
Three example prompts are shown on the right, each with a "Show More" button:

- "Please provide an airport code" with options: LAX, SFO, LHR.
- "Please provide a departure date" with option: January 23rd 2019.
- "Please select a cabin class" with options: Economy, Economy Plus, Business.

Arrows indicate the flow of data from the JSON configuration to the prompts. The "booking" variable is linked to the "Please provide an airport code" prompt. The "iResult" variable is linked to the "Please provide a departure date" prompt. The "handleEntityMatch" action is linked to the "Please select a cabin class" prompt.



Okay. We need to talk. What if you want to **perform additional validation** or just need to **invoke a custom component in response to a matched entity?**



# 'transitionAfterMatch' property

- If set to "true", component transitions to dialog flow state upon entity match
  - String "true", not the boolean true
  - Bot designers can call custom component or just acknowledge the value match
- 'match' action transition called for each entity match

What kind of pizza would you like to order?

CHEESE BASIC

PEPPERONI

MEAT LOVER

SUPREME

PREMIUM GARDEN VEGGIE

ULTIMATE CHEESE LOVER

HAWAIIAN CHICKEN

BACON SPINACH ALFREDO

PEPPERONI

Confirming entity match in composite bag:

Bag item: Type  
Entity name: PizzaType  
Entity value: PEPPERONI

What size do you want?

Large

Medium

Small

X-Large

Small

Confirming entity match in composite bag:

Bag item: Size  
Entity name: PizzaSize  
Entity value: Small

# Navigating on entity match

```
orderPizza:
  component: "System.ResolveEntities"
  properties:
    variable: "pizza"
    nlpResultVariable: "iResult"
    maxPrompts: 2
    transitionAfterMatch: "true"
    showMoreLabel: "More"
  translate:
  transitions:
    next: "confirmation"
  actions:
    match: "handleAfterMatch"
    cancel: "cancelorder"

handleAfterMatch:
  component: "System.Output"
  properties:
    text: |-
      Confirming entity match in composite bag:

      Bag item: ${system.entityToResolve.value.resolvingField}
      Entity name: ${system.entityToResolve.value.allMatches[0].entityName}
      Entity value: ${pizza.value[system.entityToResolve.value.resolvingField]}

  keepTurn: true
  transitions:
    #resume_orderPizza
    next: "orderPizza"
```

What kind of pizza would you like to order?

CHEESE BASIC

PEPPERONI

MEAT LOVER

SUPREME

PREMIUM GARDEN VEGGIE

ULTIMATE CHEESE LOVER

HAWAIIAN CHICKEN

BACON SPINACH ALFREDO

PEPPERONI

Confirming entity match in composite bag:

Bag item: Type  
Entity name: PizzaType  
Entity value: PEPPERONI

What size do you want?

Large

Medium

Small

X-Large

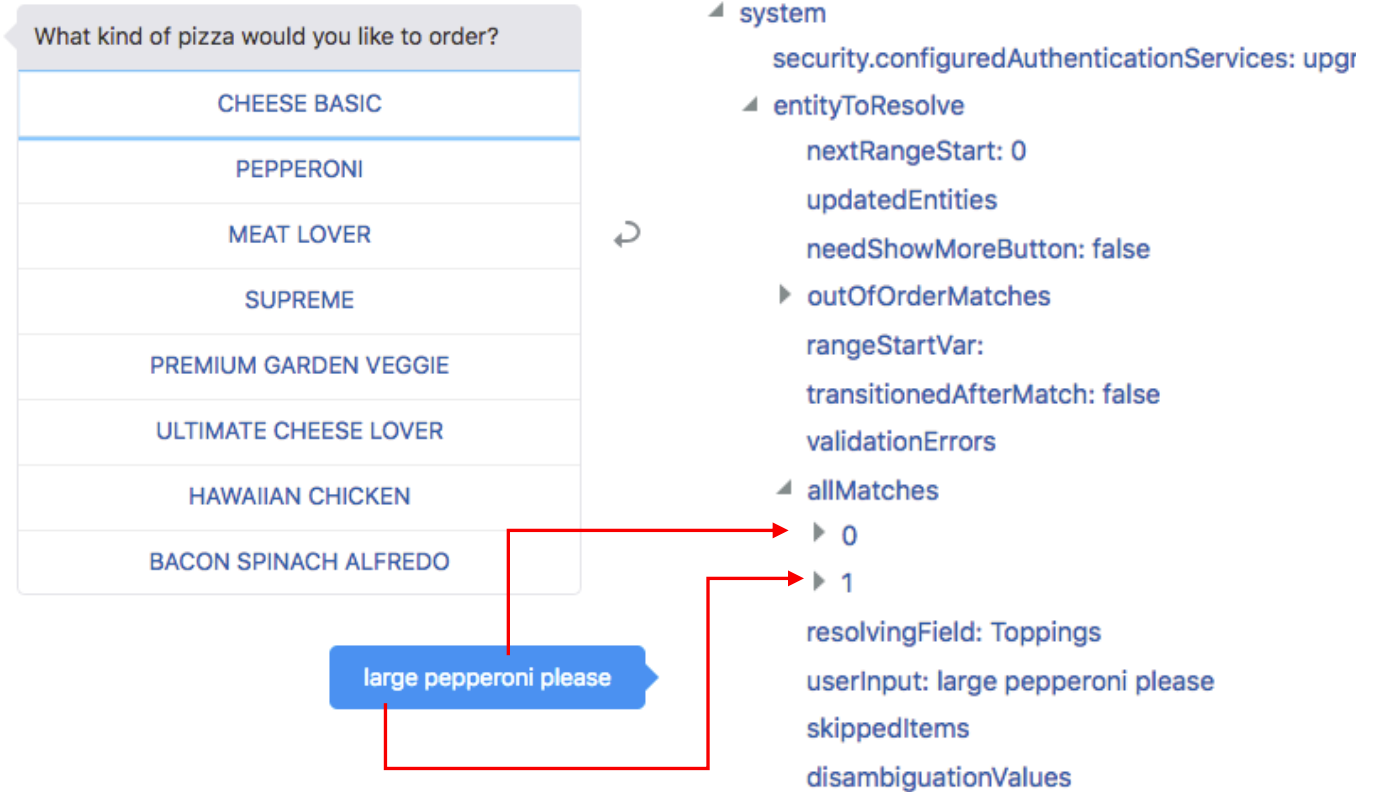
Small

Confirming entity match in composite bag:

Bag item: Size  
Entity name: PizzaSize  
Entity value: Small

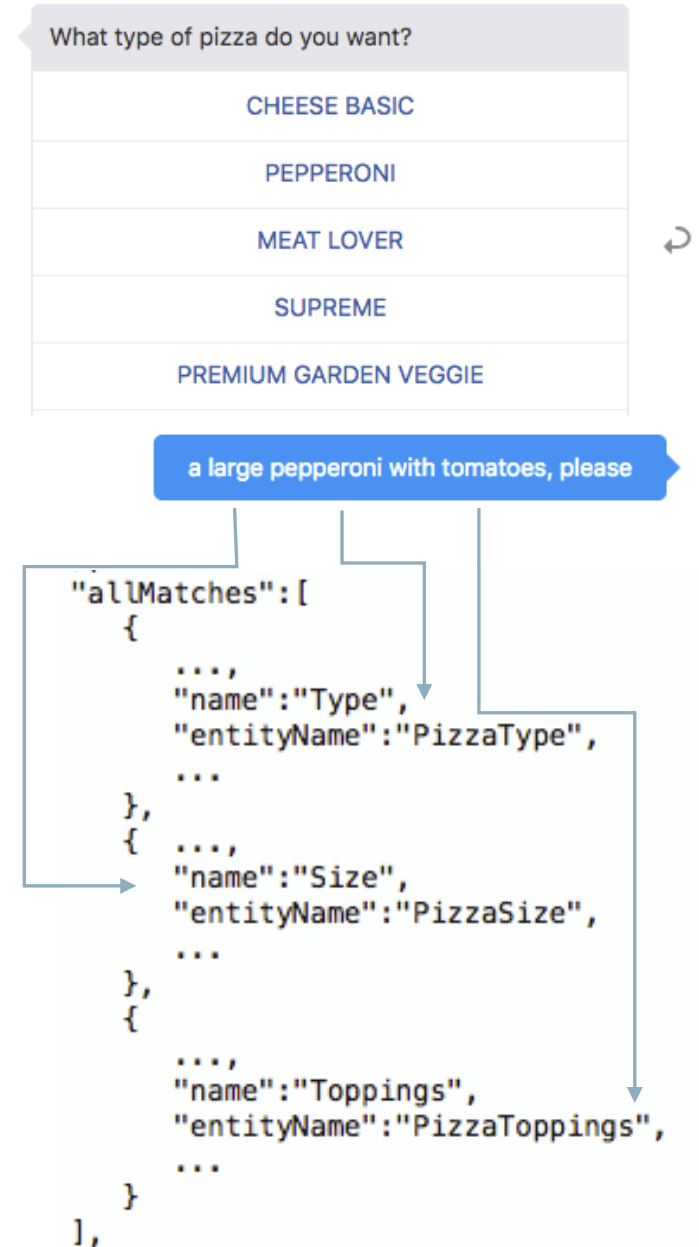
# Behavior when multiple entities are getting resolved

- User input may lead to multiple entity matches
  - Out-of-order extraction
- 'match' transition is called only once
- Access matched bag items
  - `${system.entityToResolve.value.allMatches[n].entityName}`
  - `${system.entityToResolve.value.allMatches[n].name}`



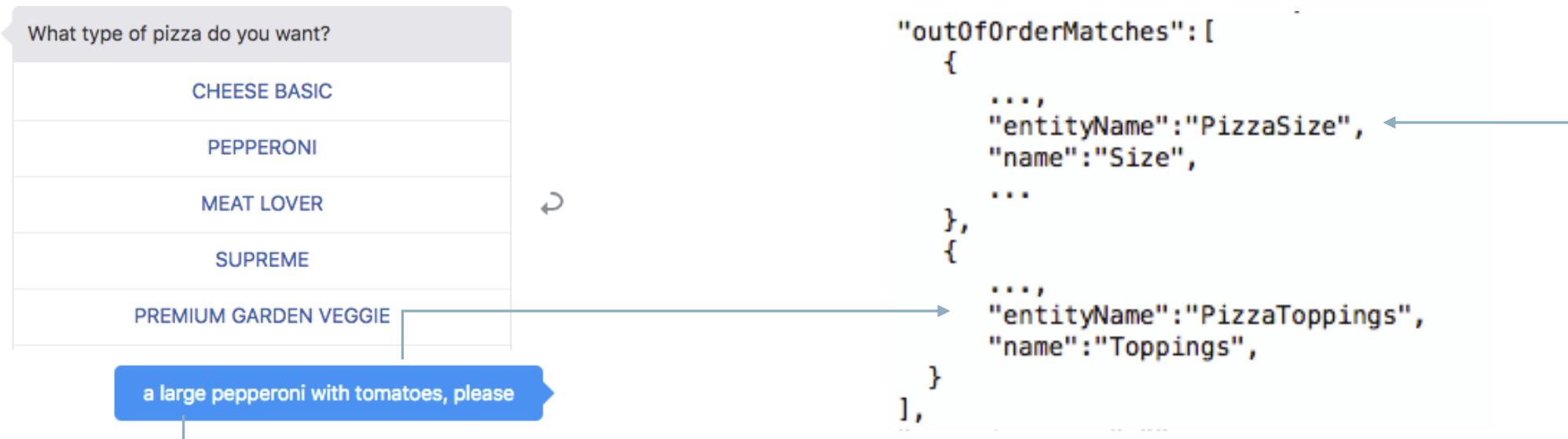
# Accessing matched entities

- All updated entities from a user input
  - E.g. User provides more information than prompted for
    - Bot: "what pizza type do you like?"
    - User: "a large salami with tomatoes"
  - Updated entities: PizzaSize, PizzaType, PizzaToppings
- Expression to access matched entities
  - `${system.entityToResolve.value.allMatches?size}`
  - `${system.entityToResolve.value.allMatches[n].entityName}`
  - `${system.entityToResolve.value.allMatches[n].name}`



# Accessing out-of-order entity matches

- Updated entities from a user input for which there was no prompt
  - `${system.entityToResolve.value.outOfOrderMatches[n].entityName}`
  - `${system.entityToResolve.value.outOfOrderMatches[n].name}`
  - `${system.entityToResolve.value.outOfOrderMatches?has_content?then(...,...)}`



# Integrated Cloud

## Applications & Platform Services

ORACLE®