Frame-Based Dialogue Systems

Natalie Parde
UIC CS 421

Components of Frame-based Dialogue Systems

Slots: Specify what the system needs to know

- **Slot fillers** are constrained to values of specific semantic types
  - In the travel domain, these could be *cities*, *dates*, *airlines*, or *times*

These types may have hierarchical structures

- Date
  - *Month*
  - *Year*
  - *Day*
  - *Weekday*

Components of Frame-based Dialogue Systems

Slot	Type	Question Template
ORIGIN CITY	city	“From what city are you leaving?”
DESTINATION CITY	city	“Where are you going?”
DEPARTURE TIME	time	“When would you like to leave?”
DEPARTURE DATE	date	”What date would you like to leave?”
ARRIVAL TIME	time	“When do you want to arrive?”
ARRIVAL DATE	date	“What day would you like to arrive?”

Control Structure for Frame-based Dialogue

Goal

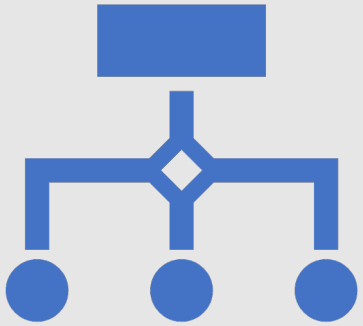
1. Fill the slots in the frame with the fillers the user intends
2. Perform the relevant action for the user

The system achieves its goal by asking questions of a user

- Typically these questions are constructed using pre-specified question templates associated with each slot of each frame

Control Structure for Frame-based Dialogue

- The system continues questioning the user until it can fill all slots needed to perform the desired task
- It might attach **condition-action rules** to slots to reduce monotony
  - If a user has specified a flight destination city, it may automatically fill the hotel destination slot with that value as well



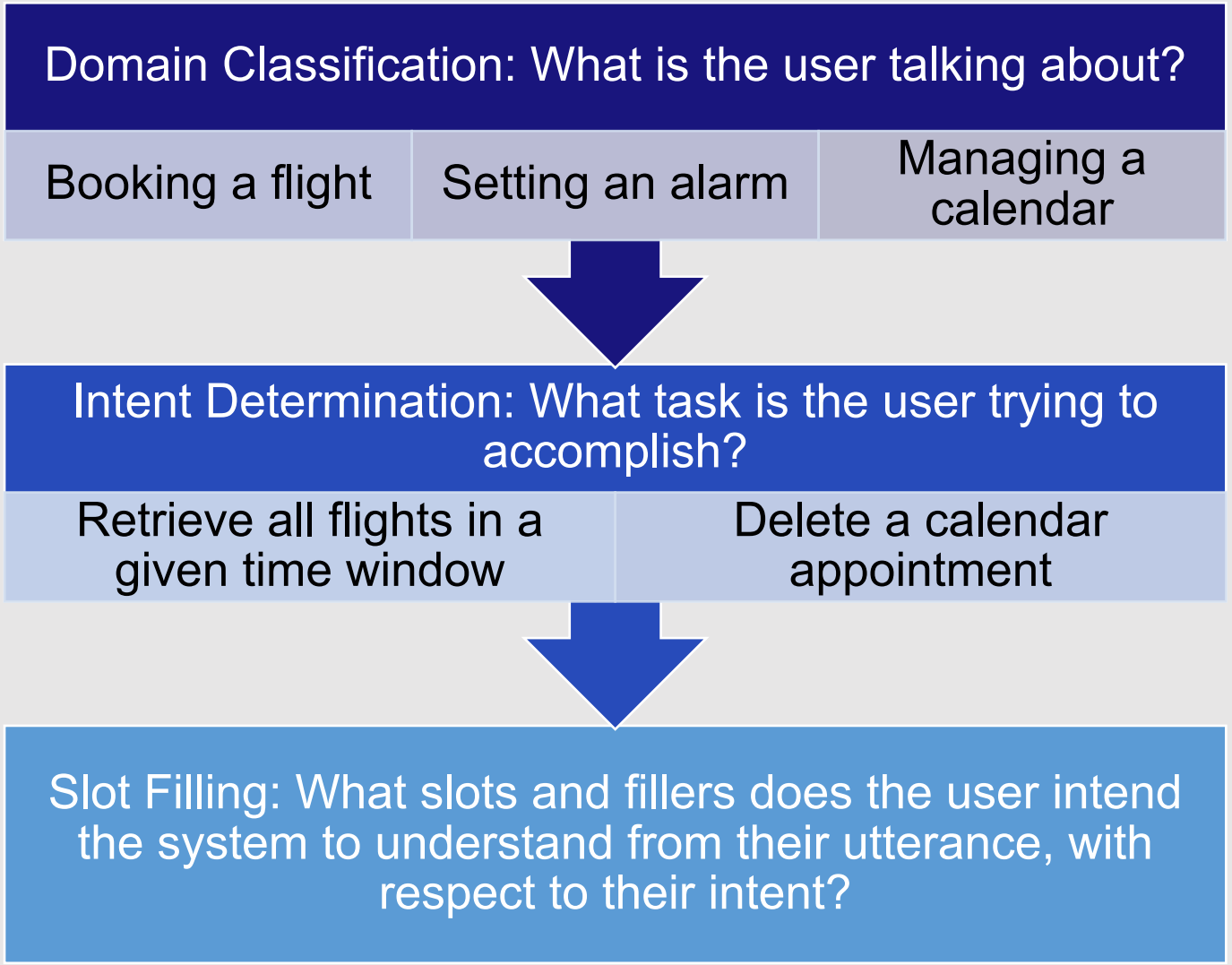
Control Structure for Frame-based Dialogue

- Many domains require multiple frames!
- Dialogue systems must be able to **disambiguate** which slot of which frame a given input is supposed to fill, and then switch dialogue control to that frame
- This can be done using **production rules**
  - Different types of inputs and recent dialogue history match different frames
  - Control is switched to the matched frame
- Once the system has enough information, it performs the desired task (e.g., querying a database of flights) and returns the result to the user

Natural Language Understanding in Frame-based Dialogue Systems

- In a frame-based dialogue system, natural language understanding is necessary for performing three tasks:
  - **Domain classification**
  - **Intent determination**
  - **Slot filling**

Natural Language Understanding in Frame-based Dialogue Systems



Example Frames and Values

Show me the morning flights from Chicago to Dallas on Thursday.

Domain:	AIR-TRAVEL
Intent:	SHOW FLIGHTS
Origin-City:	Chicago
Origin-Date:	Thursday
Origin-Time:	morning
Destination-City:	Dallas

Wake me tomorrow at 6

Domain:	ALARM-CLOCK
Intent:	SET-ALARM
Time:	2020-08-24 0600

Natural Language Understanding for Slot Filling in Frame-based Dialogue Systems

In many commercial applications, slots are filled using handwritten rules

wake me (up)? | set (the|an) alarm | get me up → Intent: SET-ALARM

Rule-based systems often include large quantities (thousands!) of rules structured as semantic grammars

Semantic Grammar: A context-free grammar in which the left-hand side of each rule corresponds to the semantic entities (slot names) being expressed

Semantic grammars can be parsed using any CFG parsing algorithm

Other systems use supervised learning for slot filling

Semantic Grammar

SHOW → show me | i want | can i see

DEPART_TIME_RANGE → (after | around | before) HOUR | morning | afternoon | evening

HOUR → one | two | three | four | … | twelve (AM|PM)

FLIGHTS → (a) flight | flights

AMPM → am | pm

ORIGIN → from CITY

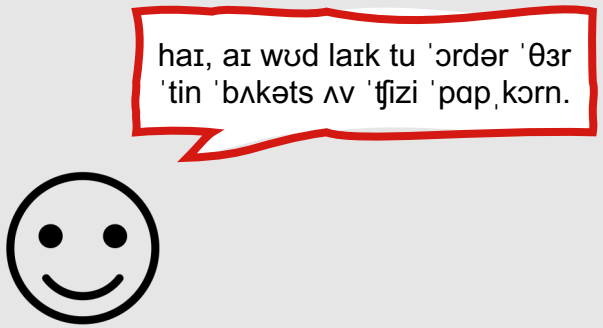
DESTINATION → to CITY

CITY → Chicago | Dallas | Denver | Phoenix

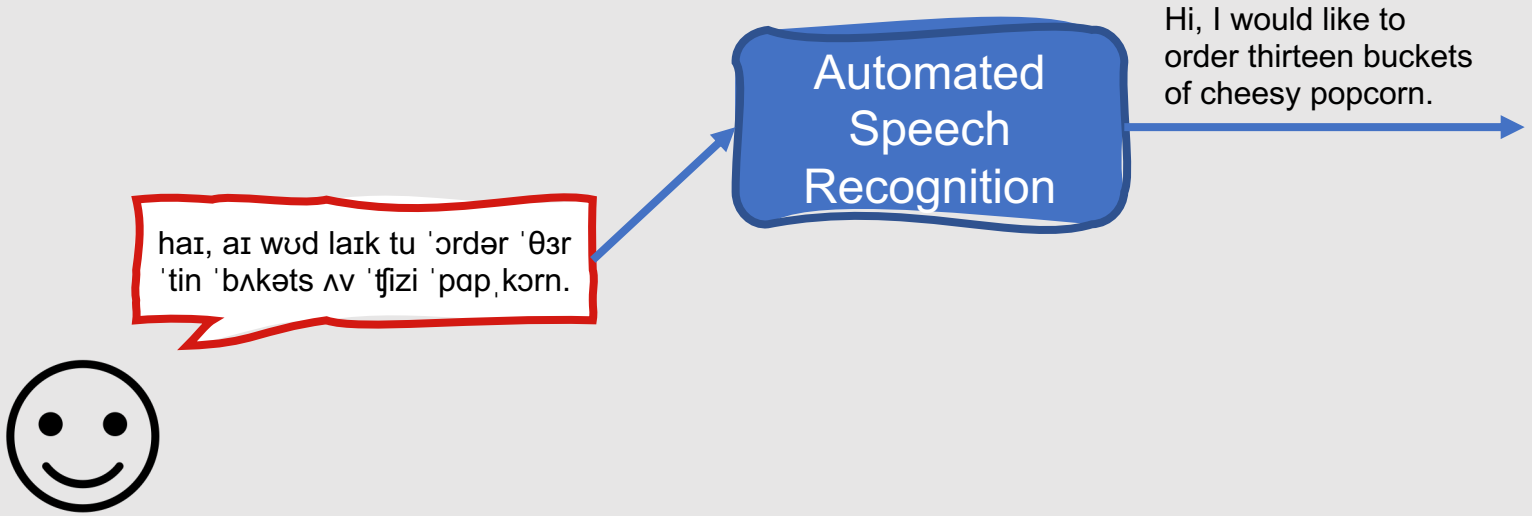
Other Components of Frame-based Dialogue Systems

- **Automated Speech Recognition:** Converts audio input to a string of text
  - May or may not be constrained based on the current dialogue domain and/or intent
- **Natural Language Generation:** Produces the utterances that the system outputs to the user
  - Frame-based systems typically use **template-based generation**
    - What time do you want to leave ORIGIN-CITY?
    - Will you return to ORIGIN-CITY from DESTINATION-CITY?
- **Text to Speech Synthesis:** Converts a string of text to an audio output
  - May be done at runtime or using prerecorded statements or phrases

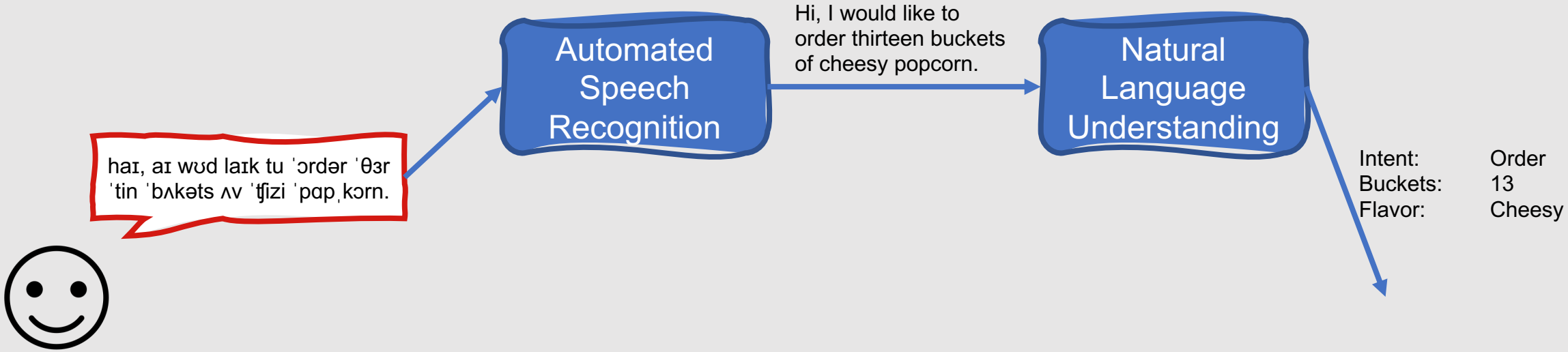
Dialogue State Architecture



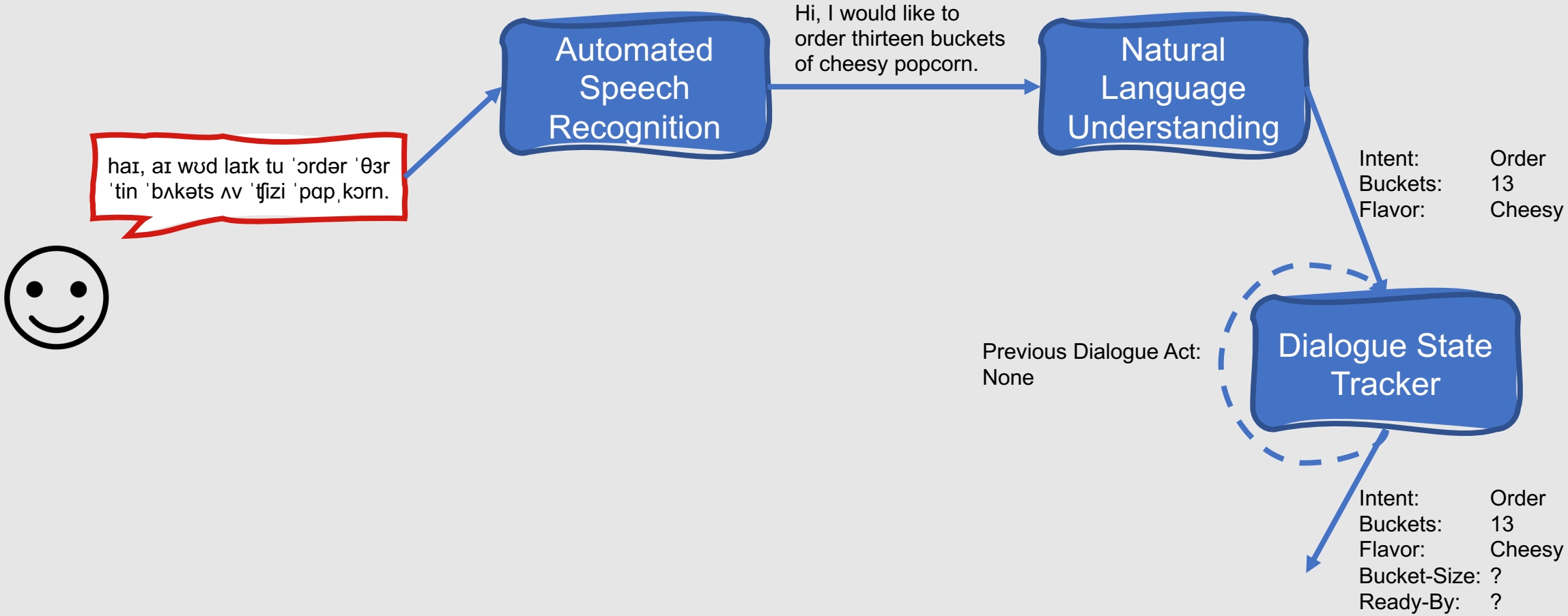
Dialogue State Architecture



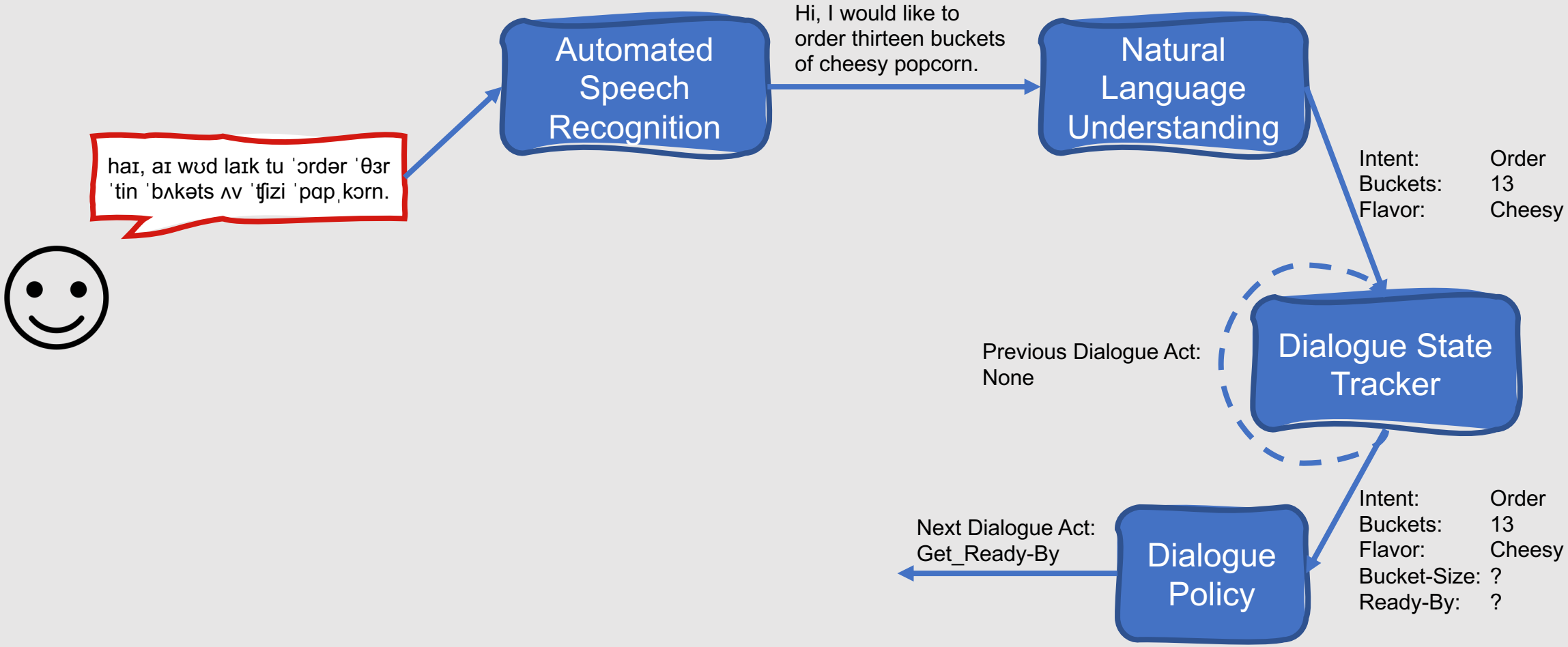
Dialogue State Architecture



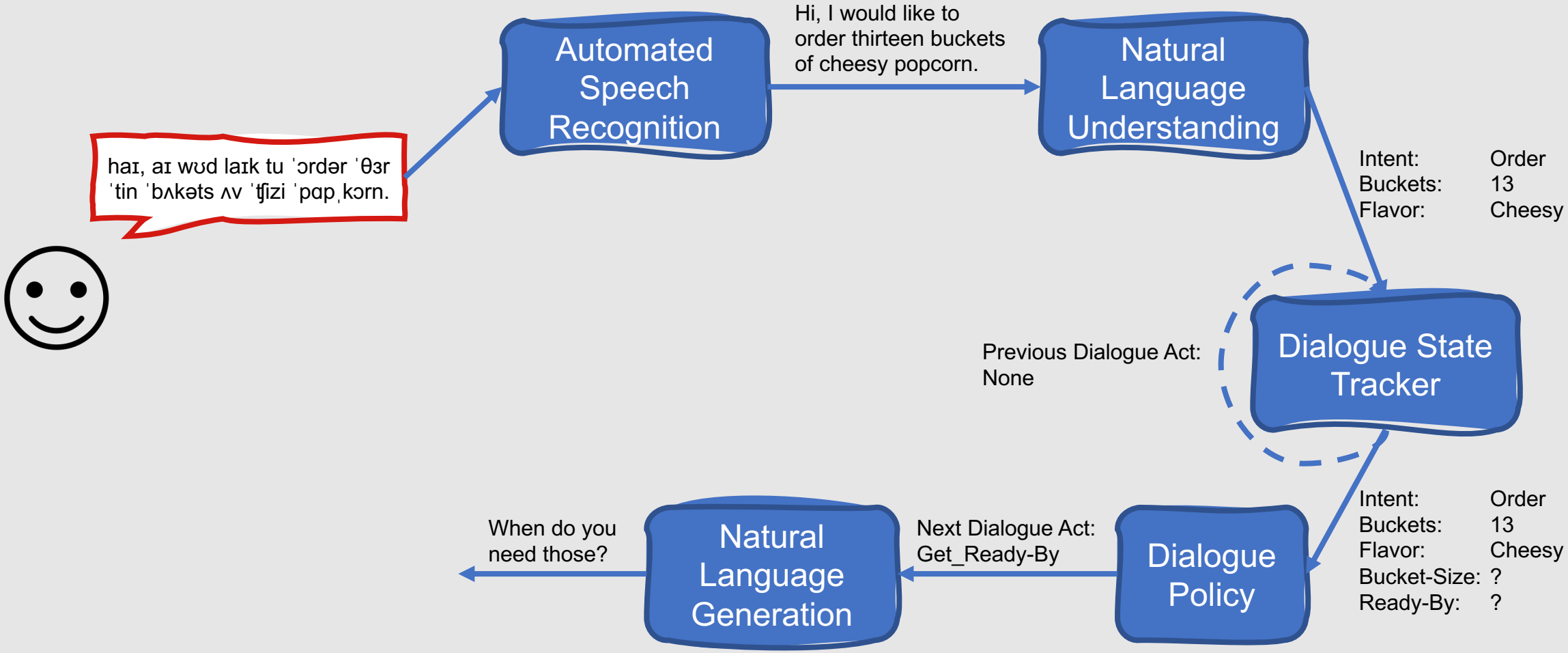
Dialogue State Architecture



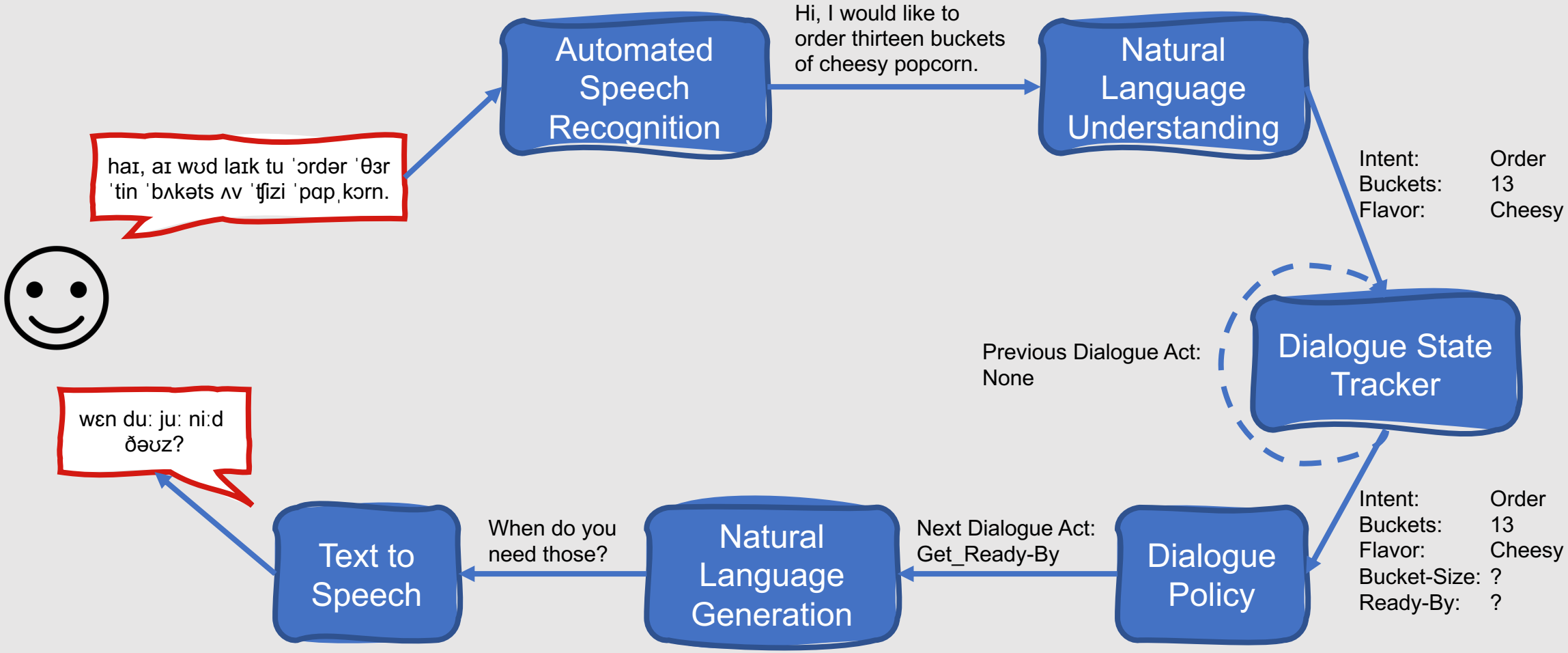
Dialogue State Architecture

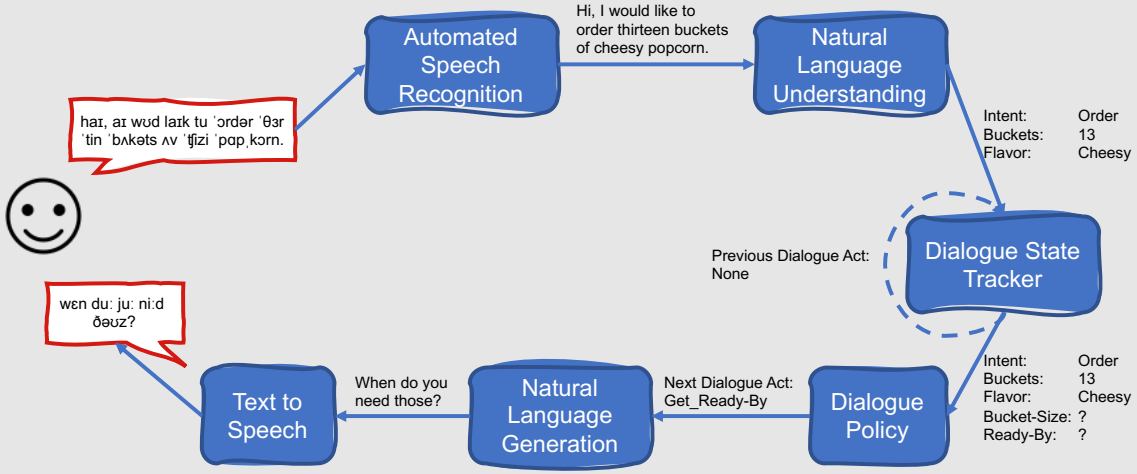


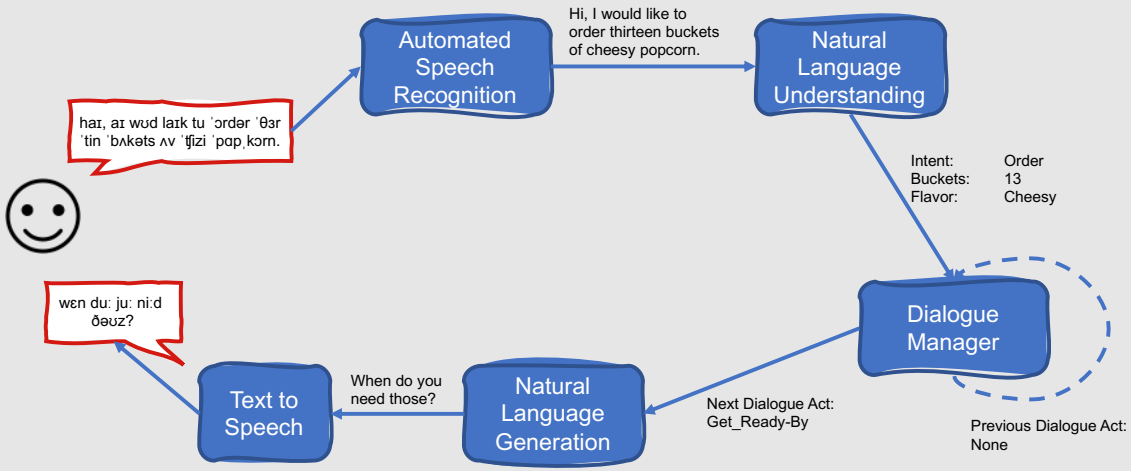
Dialogue State Architecture



Dialogue State Architecture







The dialogue state tracker and dialogue policy are sometimes grouped together as a single dialogue manager.

Automated Speech Recognition

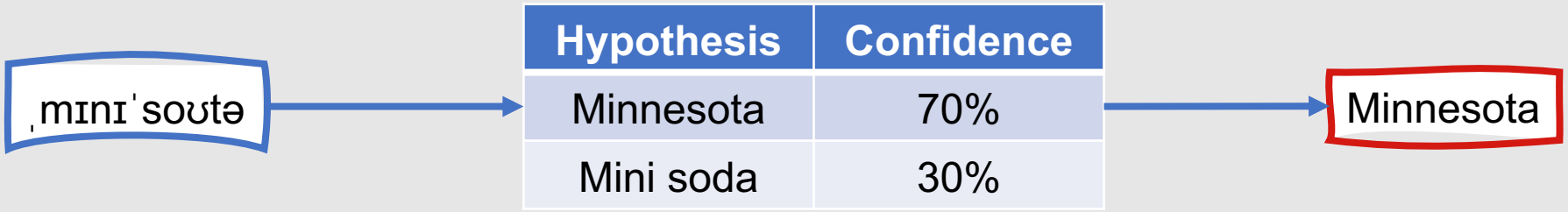
- Input: Audio
- Output: Transcribed string of words
- Can be optimized for domain-dependent dialogue systems by constraining the vocabulary to a fixed, smaller set of relevant words

Automated Speech Recognition

- Very small vocabulary for a given dialogue state → finite state grammar
- Larger vocabulary needed for dialogue state → n-gram language model with probabilities conditioned on the dialogue state
- State-specific language models are **restrictive grammars**
  - Few options for user → user has **less initiative**
  - More options for user → user has **more initiative**

Automated Speech Recognition

- ASR systems need to work quickly (users are often unwilling to wait for long pauses while their input is processed)
  - Prioritizing efficiency may necessitate constraining the vocabulary
- Generally return a confidence score for an output text sequence
  - Dialogue system can use this score to determine whether to request clarification, or move forward on the assumption that the sequence is correct



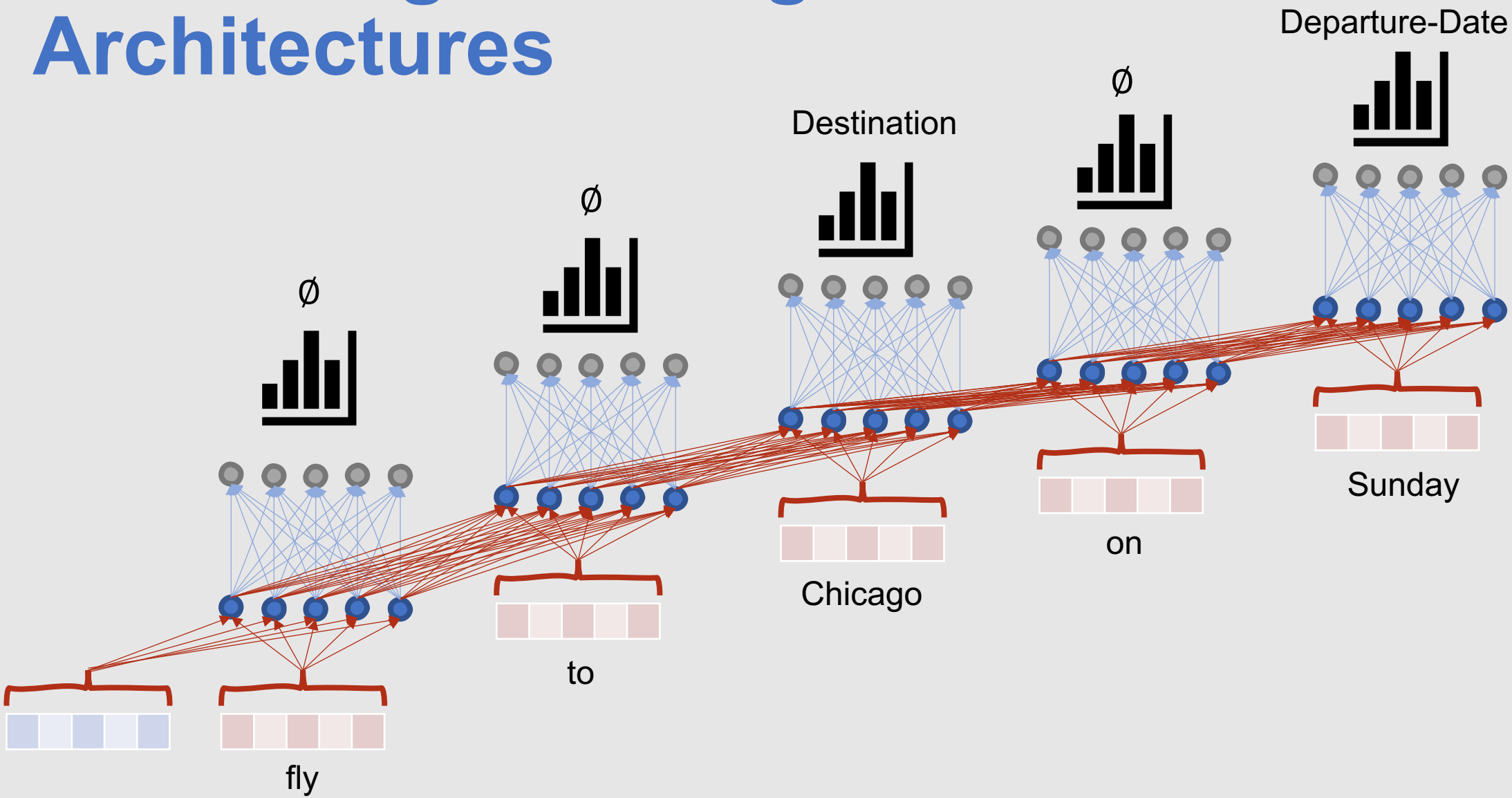
Natural Language Understanding

- Similar to the simple frame-based architecture
  - Slot fillers are extracted from the user's utterance
- However, generally uses machine learning rather than rules

Slot Filling in Dialogue State Architectures

- Special case of supervised semantic parsing
  - Labeled training set associates each sentence with the correct set of slots, domain, and intent
- Many possible ways train a classifier for this purpose
- One method:
  - Train a sequence model to map from input words to slot fillers, domain, and intent

Slot Filling in Dialogue State Architectures



Slot Filling in Dialogue State Architectures

- Domain and intent can be determined via:
  - One vs. many classifier
  - Adding domain+intent as the desired output for the final end-of-sentence token in the sequence labeler

Common Industrial Approach to ML-based Slot Filling

- **Bootstrapping!**
- Start with a seed simple, rule-based system
- Predict labels for new user utterances
- Train classifier on new utterances with predicted labels
- Repeat as needed

Dialogue State Tracker and Dialogue Policy

- **Dialogue State Tracker:** Maintains the current state of the dialogue
  - Most recent dialogue act
  - All slot values the user has expressed so far
- **Dialogue Policy:** Decides what the system should do or say next
  - In a simple frame-based dialogue system, the system may just ask questions until the frame is full
  - In more sophisticated dialogue systems, the policy might help the system decide:
    - When to answer the user's questions
    - When to ask the user a clarification question
    - When to make a suggestion

Dialogue Acts

- Dialogue-state systems make use of **dialogue acts**
- Different types of dialogue systems require that different types of dialogue acts are labeled
  - Dialogue tagsets tend to be task-specific

Sample Dialogue Act Tagset

Tag	Valid System Act?	Valid User Act?	Description
Hello(a=x, b=y, …)	🙂	🙂	Open a dialogue and give info a=x, b=y, …
Inform(a=x, b=y, …)	🙂	🙂	Give info a=x, b=y, …
Request(a, b=x, …)	🙂	🙂	Request value for a given b=x, …
Reqalts(a=x, …)		🙂	Request alternative with a=x, …
Confirm(a=x, b=y, …)	🙂	🙂	Explicitly confirm a=x, b=y, …
Confreq(a=x, …, d)	🙂		Implicitly confirm a=x, and request value of d
Select(a=x, a=y)	🙂		Implicitly confirm a=x, and request value of d
Affirm(a=x, b=y, …)	🙂	🙂	Affirm and give further info a=x, b=y, …
Negate(a=x)		🙂	Negate and give corrected value a=x
Deny(a=x)		🙂	Deny that a=x
Bye()	🙂	🙂	Close a dialogue

Sample Annotated Dialogue

Tag	Valid System Act?	Valid User Act?	Description
Hello(a=x, b=y, …)	🙂	🙂	Open a dialogue and give info a=x, b=y, …
Inform(a=x, b=y, …)	🙂	🙂	Give info a=x, b=y, …
Request(a, b=x, …)	🙂	🙂	Request value for a given b=x, …
Reqalts(a=x, …)		🙂	Request alternative with a=x, …
Confirm(a=x, b=y, …)	🙂	🙂	Explicitly confirm a=x, b=y, …
Confreq(a=x, …, d)	🙂		Implicitly confirm a=x, and request value of d
Select(a=x, a=y)	🙂		Implicitly confirm a=x, and request value of d
Affirm(a=x, b=y, …)	🙂	🙂	Affirm and give further info a=x, b=y, …
Negate(a=x)		🙂	Negate and give corrected value a=x
Deny(a=x)		🙂	Deny that a=x
Bye()	🙂	🙂	Close a dialogue

Speaker	Utterance	Dialogue Act
U	Hi, I am looking for somewhere to eat.	
S	You are looking for a restaurant. What type of food do you like?	
U	I'd like an Italian restaurant somewhere near the lake.	
S	Petterino's is a nice Italian restaurant near the lake.	
U	Is it reasonably priced?	
S	Yes, Petterino's is in the moderate price range.	
U	What is the phone number?	
S	The phone number for Peterino's is 123-456-7890	
U	Okay, thank you. Goodbye!	

Sample Annotated Dialogue

Tag	Valid System Act?	Valid User Act?	Description
Hello(a=x, b=y, …)	🙂	🙂	Open a dialogue and give info a=x, b=y, …
Inform(a=x, b=y, …)	🙂	🙂	Give info a=x, b=y, …
Request(a, b=x, …)	🙂	🙂	Request value for a given b=x, …
Reqalts(a=x, …)		🙂	Request alternative with a=x, …
Confirm(a=x, b=y, …)	🙂	🙂	Explicitly confirm a=x, b=y, …
Confreq(a=x, …, d)	🙂		Implicitly confirm a=x, and request value of d
Select(a=x, a=y)	🙂		Implicitly confirm a=x, and request value of d
Affirm(a=x, b=y, …)	🙂	🙂	Affirm and give further info a=x, b=y, …
Negate(a=x)		🙂	Negate and give corrected value a=x
Deny(a=x)		🙂	Deny that a=x
Bye()	🙂	🙂	Close a dialogue

Speaker	Utterance	Dialogue Act
U	Hi, I am looking for somewhere to eat.	Hello(task=find, type=restaurant)
S	You are looking for a restaurant. What type of food do you like?	Confreq(type=restaurant, food)
U	I'd like an Italian restaurant somewhere near the lake.	Inform(food=Italian, near=lake)
S	Petterino's is a nice Italian restaurant near the lake.	Inform(name="Petterino's", type=restaurant, food=Italian, near=lake)
U	Is it reasonably priced?	Confirm(pricerange=moderate)
S	Yes, Petterino's is in the moderate price range.	Affirm(name="Petterino's", pricerange=moderate)
U	What is the phone number?	Request(phone)
S	The phone number for Peterino's is 123-456-7890	Inform(name="Petterino's", phone="123-456-7890")
U	Okay, thank you. Goodbye!	Bye()

Natural Language Generation

- In simple frame-based dialogue systems, sentences are produced from pre-written templates
- In more sophisticated dialogue systems, the natural language generation component can be **conditioned on prior context** to produce more natural-sounding dialogue turns

Text to Speech Synthesis

- Inputs:
  - Words
  - Prosodic annotations
- Output:
  - Audio waveform

What is prosody?

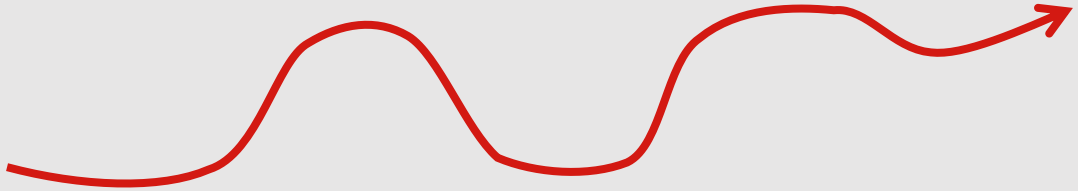
- **Prosody:** Elements of speech such as **intonation**, **tone**, **stress**, and **rhythm**
- Often carries hints regarding:
  - A speaker's emotional state
  - The type of utterance being spoken
  - The presence of sarcasm
  - The focus of the utterance

Common Prosodic Trends

Question: Is there any popcorn left?

Sarcasm: I just love having no popcorn.

Anger: You ate my last bucket of popcorn!



Emphasis: So, caramel or cheesy popcorn?

Spoken Dialogue Systems vs. Text-based Dialogue Systems

- Automated speech recognition and text to speech synthesis are only necessary in **spoken dialogue systems**
  - Dialogue systems which accept spoken input and produce spoken output
- Other dialogue systems can eliminate those components, moving directly from:
  - Input to natural language understanding
  - Natural language generation to output

Text-based Dialogue State Architecture