



Teaching, Learning, Doing and Conversing: How They Fit Together

Charles Rich
Worcester Polytechnic Institute

Computer Science Department
Interactive Media and Game Development Program

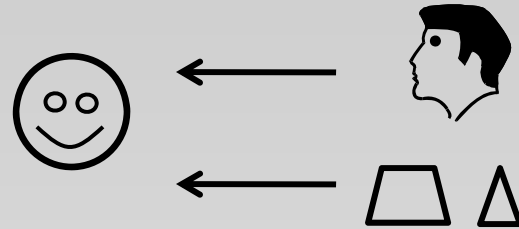
Architecture



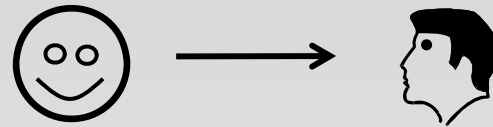
- symmetry
- balance
- modularity
- economy
- ...

Lifelong Learning Companion

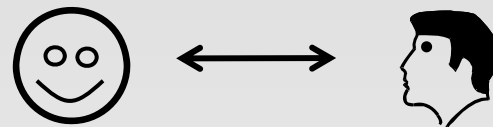
- Learning



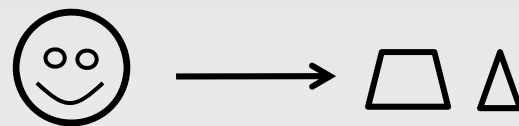
- Teaching



- Conversing



- Doing



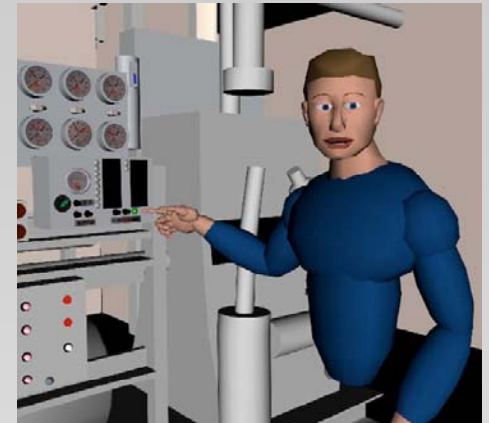
Example Systems

- PACO “Pedagogical Agent for Collagen”
 - Jeff Rickel, Neal Lesh, Charles Rich, Candace L. Sidner, Abigail Gertner
 - successor to STEVE
 - unified intelligent tutoring & collaborative dialogue
- COLLAGEN “Collaborative Agent”
 - Charles Rich, Candace L. Sidner, Neal Lesh
 - unified intentional, attentional and linguistic aspects of dialogue

STEVE → PACO

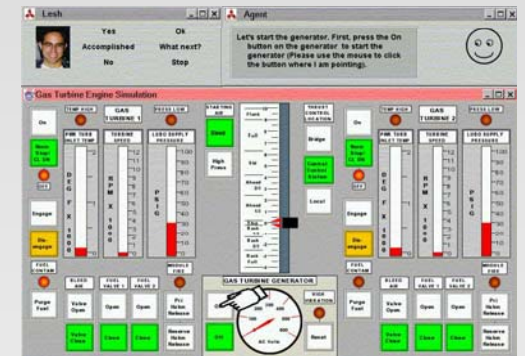
■ STEVE

- *hierarchical task network* used to keep track of both student and teacher goals/actions
- but tutorial *dialogue* implementation was ad hoc



■ PACO

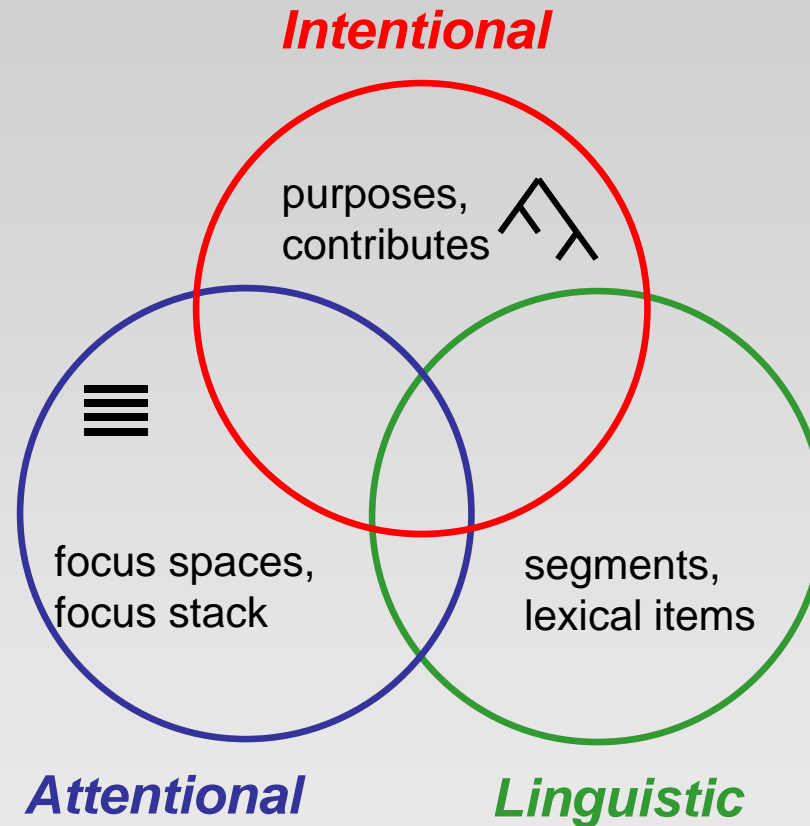
- re-implemented STEVE using COLLAGEN
- task network also drives the dialogue (by adding pedagogical goals/actions)



COLLAGEN

- Key architectural features:
 - utterances treated as actions
 - symmetric treatment of user and system actions/utterances
 - discourse generation treated as inverse of interpretation
 - plug-ins for response generation

Collaborative Discourse Theory



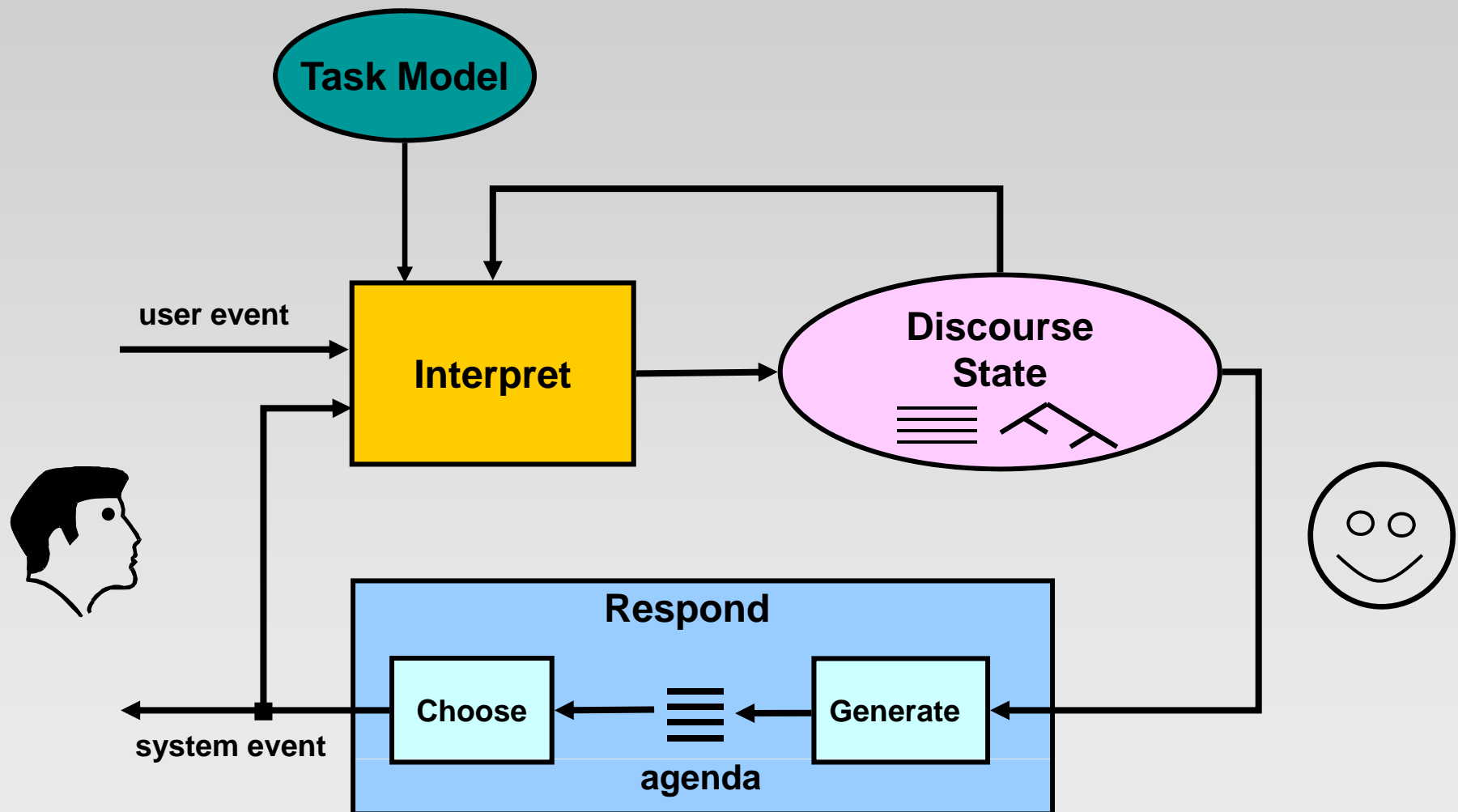
[Grosz, Sidner, Kraus, Lochbaum 1974-1998]

Contributes (Subgoals)

- An action/utterance directly contributes to the purpose of a segment if it:
 - *achieves* the purpose
 - is a *step* in the plan for the purpose
 - identifies the *recipe* to be used to achieve the purpose
 - identifies *who* should achieve the purpose
 - identifies a *parameter* of the purpose

“knowledge preconditions” [Lochbaum, 1988]

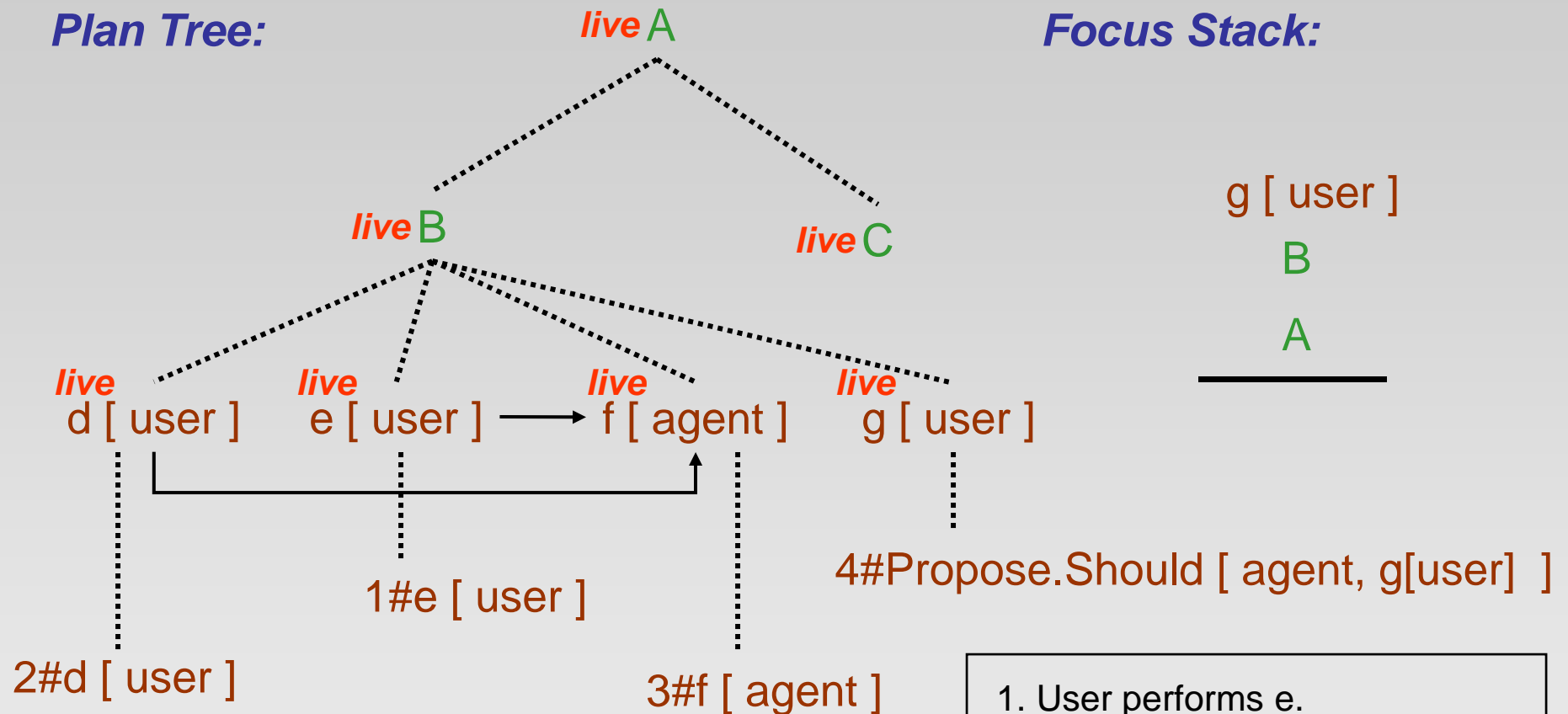
COLLAGEN Architecture



Discourse Interpretation

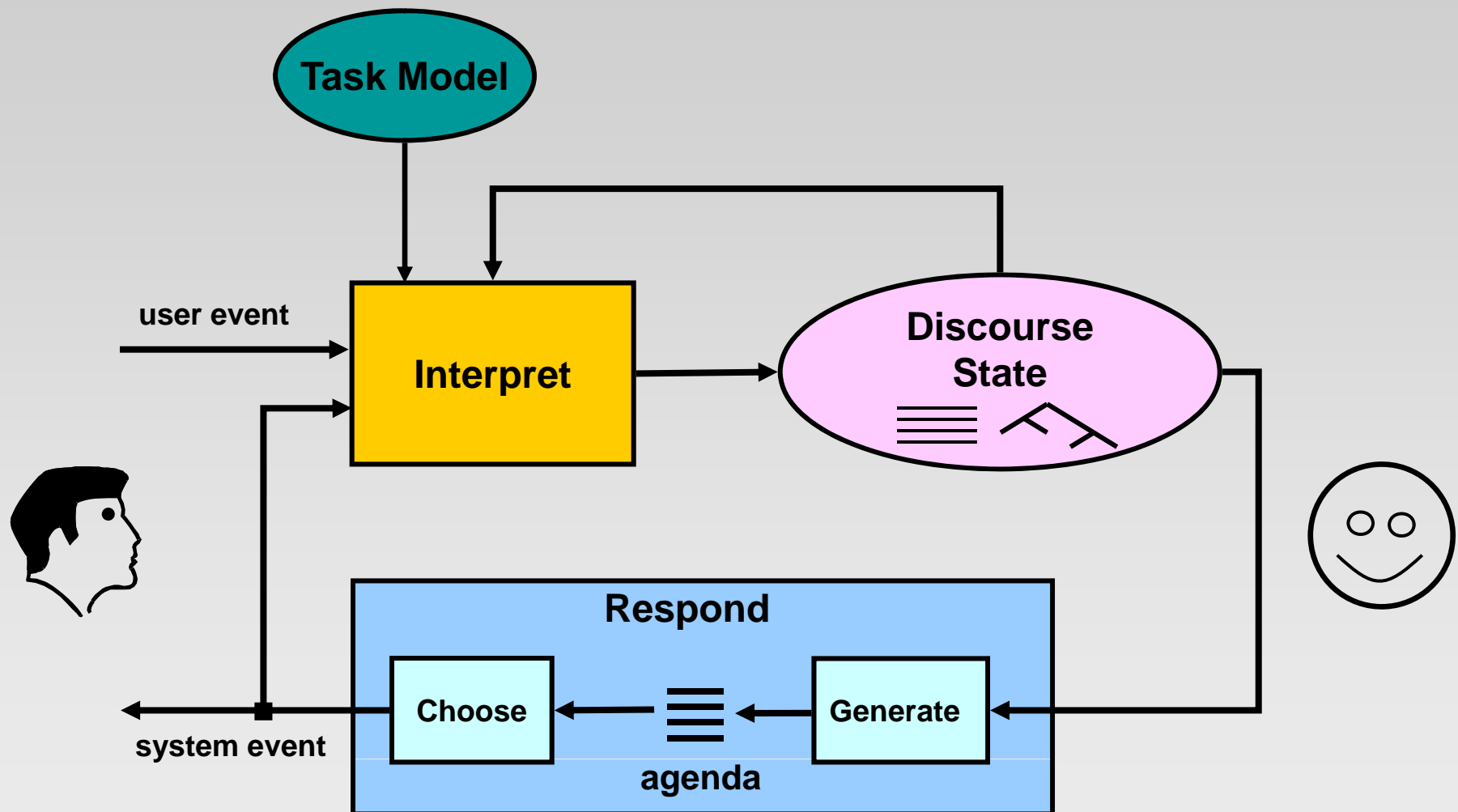
Plan Tree:

Focus Stack:



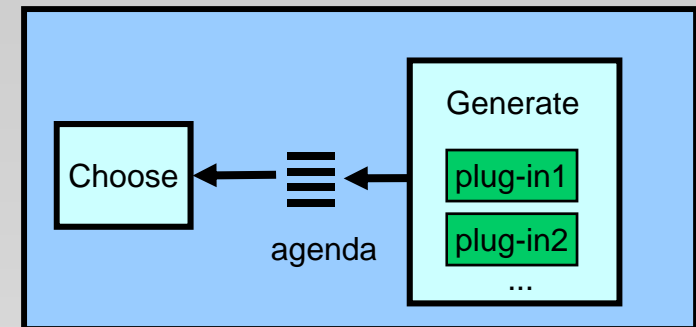
1. User performs e.
2. User performs d.
3. Agent performs f.
4. Agent says "Please perform g."

COLLAGEN Architecture



Discourse Generation

“Plug-in Architecture”

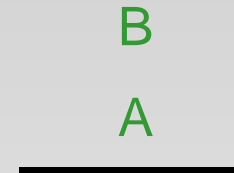
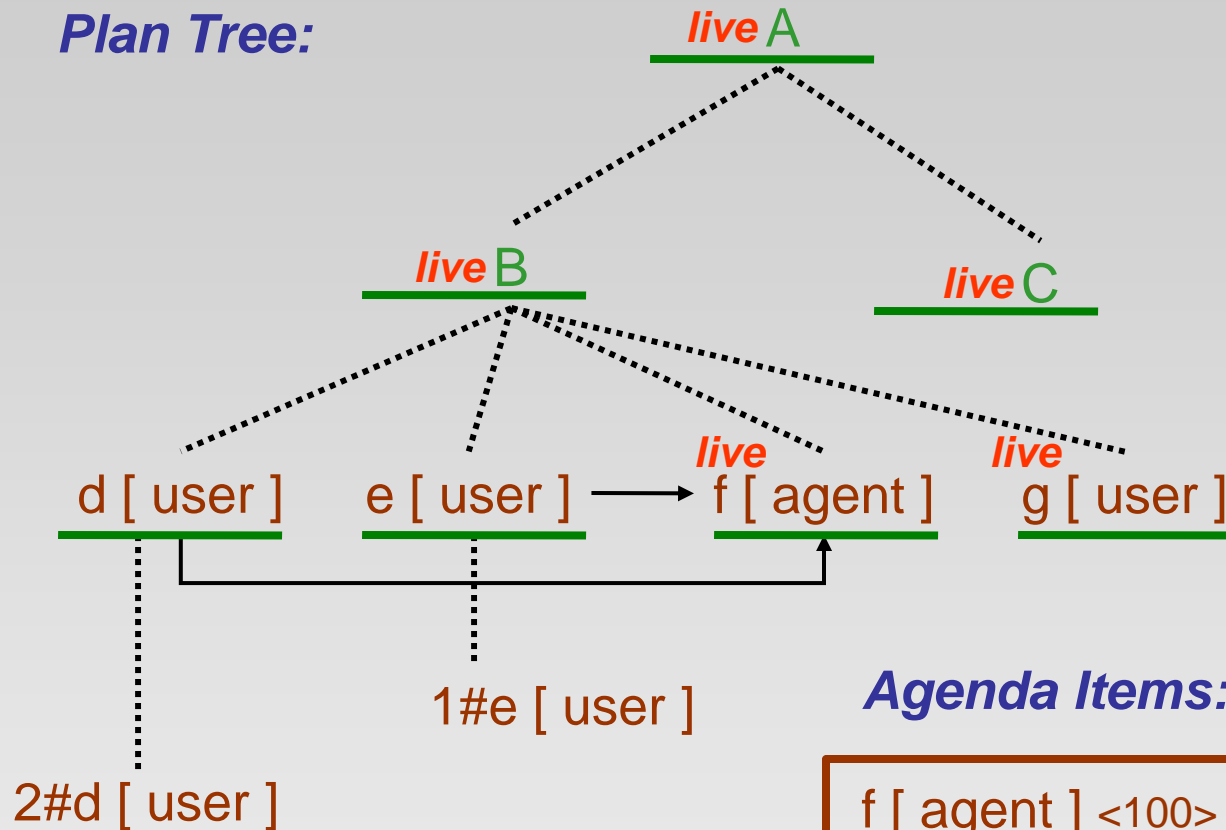


- each plug-in is applied to each node of plan tree
- plan tree nodes visited in focus stack order
- each plug-in application returns zero or more agenda items containing:
 - a (partially specified) system action/utterance which would contribute to purpose of given plan node
 - a priority (for choosing best agenda)

Discourse Generation

Plan Tree:

Focus Stack:



Agenda Items: <priority>

f [agent] <100>

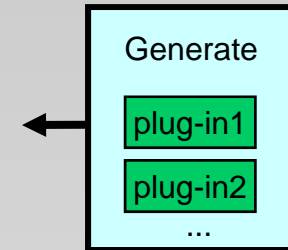
Propose.Should [agent, g[user]] <50>

Propose.Should [agent, A] <50>

Propose.Should [agent, C] <50>

Plug-in: ProposeShould

Discourse Generation Plug-ins



- Design Methodology:
 - each plug-in embodies a single response *principle*
 - all agenda items should be “reasonable” responses
 - priorities determine agent’s style and personality

Example Response Principles

- *Execute* live primitive actions which are constrained to be performed by system. <priority 100>
- *Ask* the user to execute live primitive actions which are constrained to be performed by user. <priority 50>
- *Explain* how to achieve goals which user has never achieved by herself. <priority 150>

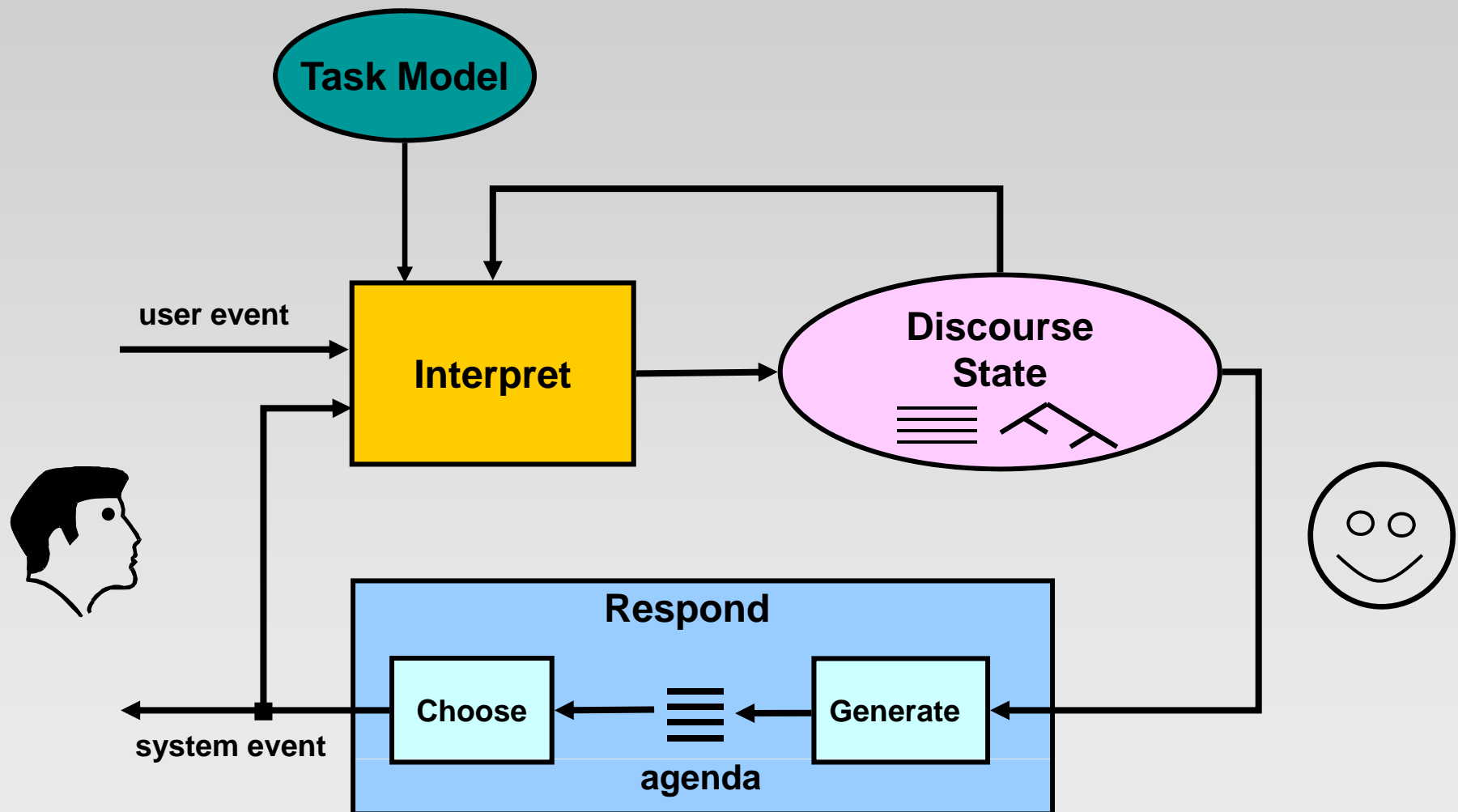
PACO Plug-ins

Default Plug-ins: Execute, ProposeShouldUser, ProposeShouldJoint, AskWho, AskWhat, AskHow

Error Correction Plug-ins: PositiveFeedback, StopInterruption, Predecessors, Precondition, NearMiss, Repeated, PrivateFocus, UnexpectedLive

Teaching Plug-ins: TeachStep, TeachApplicable, TeachInitiative, TeachSuccessful, TeachConfirm, NextExercise

PACO Architecture



PACO Tutorial Dialogue Example

The screenshot displays a software interface for a Gas Turbine Engine Simulation. At the top, there is a dialogue window with two panes. The left pane, titled 'Lesh', shows a portrait of a man and a set of response options: 'Yes', 'Accomplished', 'No', 'Ok', 'What next?', and 'Stop'. The right pane, titled 'Agent', contains a text box with the instruction: 'Let's start the generator. First, press the On button on the generator to start the generator (Please use the mouse to click the button where I am pointing).' and a simple smiley face icon.

Below the dialogue window is the 'Gas Turbine Engine Simulation' interface. It features two identical engine control panels for 'GAS TURBINE 1' and 'GAS TURBINE 2'. Each panel includes a 'TEMP HIGH' indicator, a 'PRESS LOW' indicator, and a 'STARTING AIR' indicator. The 'GAS TURBINE 1' panel shows 'PWR TURB INLET TEMP' (0-2000 DEG F), 'TURBINE SPEED' (0-12000 RPM), and 'LUBO SUPPLY PRESSURE' (0-100 PSIG). The 'GAS TURBINE 2' panel shows similar metrics. Each engine has an 'On' button, a 'Norm Stop/CL DN' button, an 'Engage' button, and a 'Dis-engage' button. There are also 'FUEL CONTAM' and 'MODULE FIRE' indicators, and buttons for 'Bleed', 'High Press', 'Bridge', 'Central Control Station', and 'Local'.

In the center of the simulation interface is a 'THRUST CONTROL LOCATION' scale ranging from -4 to 10. A hand cursor is pointing at the 'On' button of the 'GAS TURBINE GENERATOR' section, which also includes a 'HIGH VIBRATION' indicator and a 'Reset' button. The 'GAS TURBINE GENERATOR' section features a large circular gauge labeled 'AC Volts' with a scale from 0 to 600. Below the gauge are buttons for 'Valve Open', 'Valve Close', 'Open', 'Close', 'Open', 'Close', 'Pri Halon Release', and 'Reserve Halon Release'.

Confirming the Tutor's Role in the Simulation

1. Tutor says "Our first exercise is very simple."
2. Tutor sets the state of the simulator.
3. Tutor says "Let's start the generator."
4. Tutor says "The first step is to press the generator on button."
5. Tutor says "Please press the generator on button."
6. Student presses the generator on button.
7. Tutor says "Great."
8. Tutor says "We confirm successfully starting the generator by noticing a positive reading on the volt meter."
9. Tutor says "Now, do you want to try a more complex task that includes starting the generator?"
10. Student says "Yes."
11. Tutor sets the state of the simulator.
12. Tutor says "Let's engage engine one."
13. Tutor says "The first step is to start the generator."
14. Tutor says "Let's start the generator."
15. Tutor says "You take it from here."
16. Student presses the disengage button.
17. Tutor says "Please stop disengaging the engine."
18. Student says "What next?"
19. Tutor says "Please press the generator on button."
20. Student presses the generator on button.
21. Tutor says "Great."
22. Tutor says "These engines need a supply of compressed air in order to be started."
23. Tutor says "The next step is to open the bleed valve to tap into the air produced by the generator."
24. Tutor says "Please open the bleed valve."
25. Student presses the engage button.
26. Tutor says "Whoops, it's too soon to do that."
27. Student opens fuel valve one.
28. Tutor says "That was a correct action, but I wasn't expecting you to do it now."
29. Tutor says "Let's go back to opening the bleed valve."
30. Student opens the bleed valve.
31. Tutor says "Right."
32. Tutor says "Now, notice that the fuel contamination light is on."
33. Tutor says "The next step is to press the purge button to clear fuel line contamination."
34. Tutor says "Please press the purge button."
35. Student presses the purge button.
36. Tutor says "Great."
37. Tutor says "We confirm successfully clearing fuel line contamination by noticing that the fuel contamination light is off."
38. Tutor says "The next step is to press the engage button again."
39. Tutor says "Please press the engage button."
40. Student presses the engage button.
41. Tutor says "Great."
42. Tutor says "We succeeded in engaging engine one."

What About Learning?

- Learning *procedural* knowledge (i.e., adding to task model)
 - same collaborative discourse framework used for user to teach/instruct system
 - cf. Andy Garland, AAI-FSS'00, KCAP'01
 - modest generalization capabilities go a long way (especially if system can ask questions)
 - cf. Bootstrap Learning project (DARPA)

Summary / Morals

- **Reuse**

- discourse state used both for interpretation and generation

- **Symmetry**

- actions and utterances
- system and user

- **Modularity**

- response generation plug-ins

Questions?

Discussion?