ELTE | FACULTY OF INFORMATICS

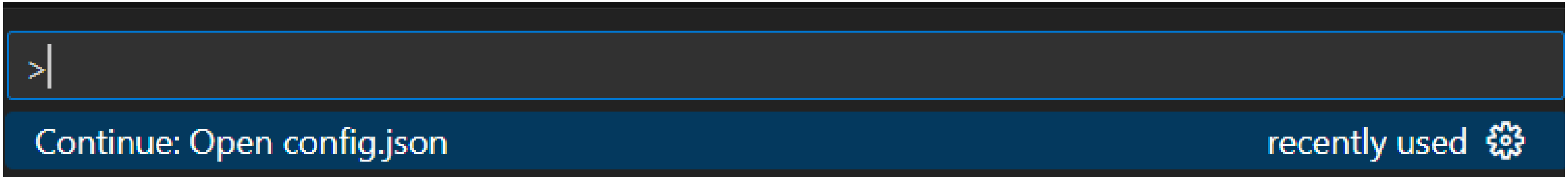
Local Code Assistants, Agents, Coding Agents

Continue Dev

- „Local” Copilot-like extension
- Pros:
  - Privacy
  - Cost(?)
  - Customizable models
- Cons:
  - Speed(?)
  - Accuracy

Continue Dev

- Install from extensions
- Use continue tab
  - Try with OpenAI API key
- Functionality similar to Copilot Chat-mode
- Custom commands/prompts available



>

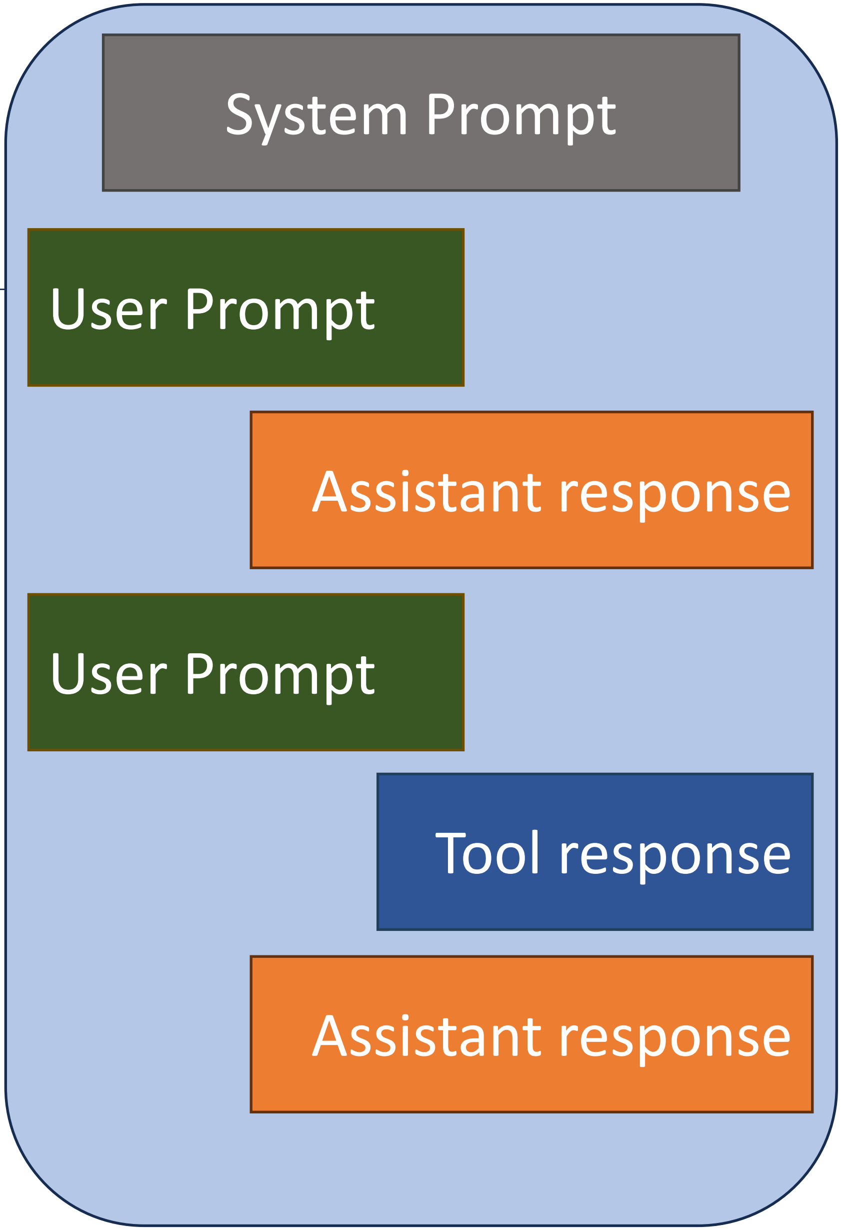
Continue: Open config.json

Continue Dev

- Local LLM-s with Huggingface
- https://huggingface.co/docs/text-generation-inference - Huggingface TGI
- Models at: https://huggingface.co/
- Supported models: https://docs.continue.dev/setup/select-model

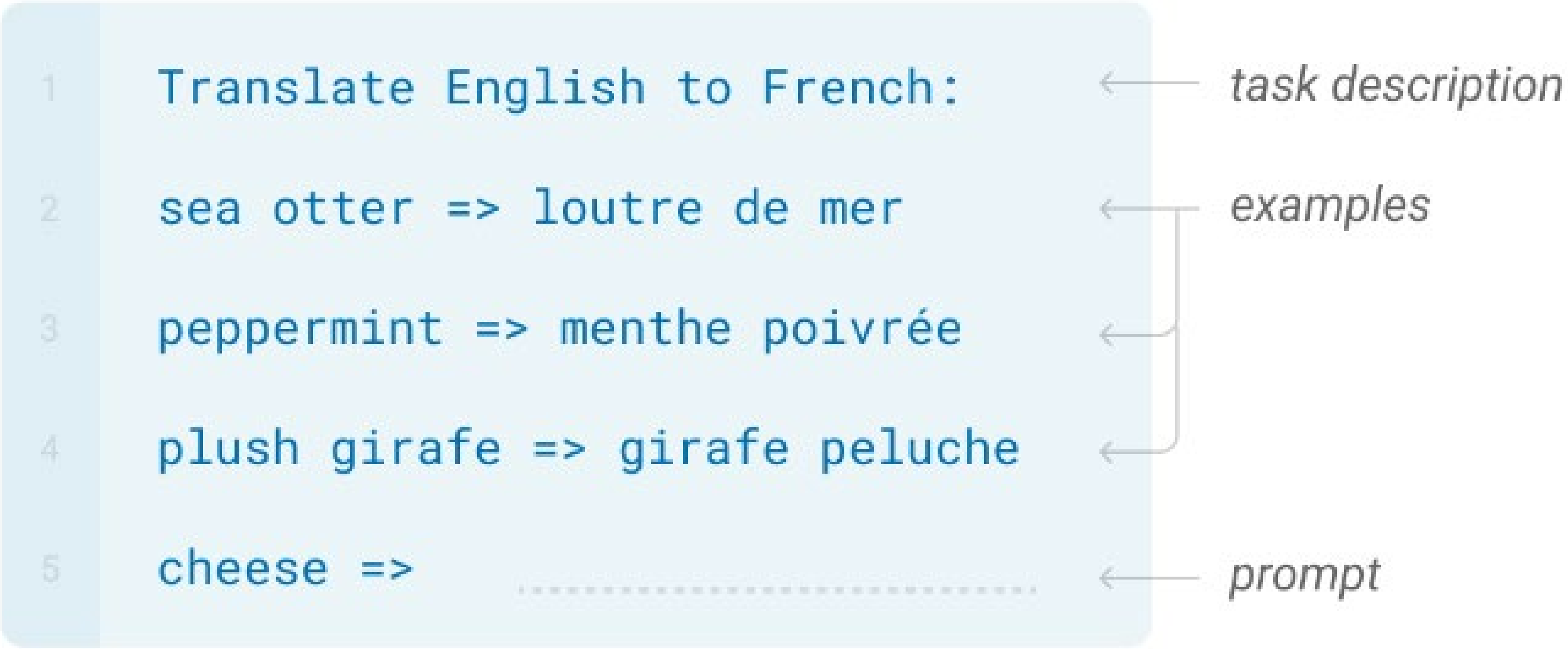
Prompting chat models

- System prompt
  - General definition for the whole conversation, does not get cut out of history
- User-Assistant prompts form the history which is limited. Breaking alternation is often allowed.
- Textual tool outputs might also be included in the conversation



In-Context Learning

- Few-shot prompts include the task definition and a few examples.



```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese =>
```

Source: Brown, et. al. 2020.

Problem Decomposition

- LLMs have limited single-step reasoning capabilities
- They are good at small common sense reasoning steps
- Hard problems should be decomposed into a Chain of Thoughts (CoT), where each thought is just a small step towards the solution.
- Latest models have built-in (by alignment) CoT capabilities

CoT Example

Which one is older Bosch or the Eiffel tower?

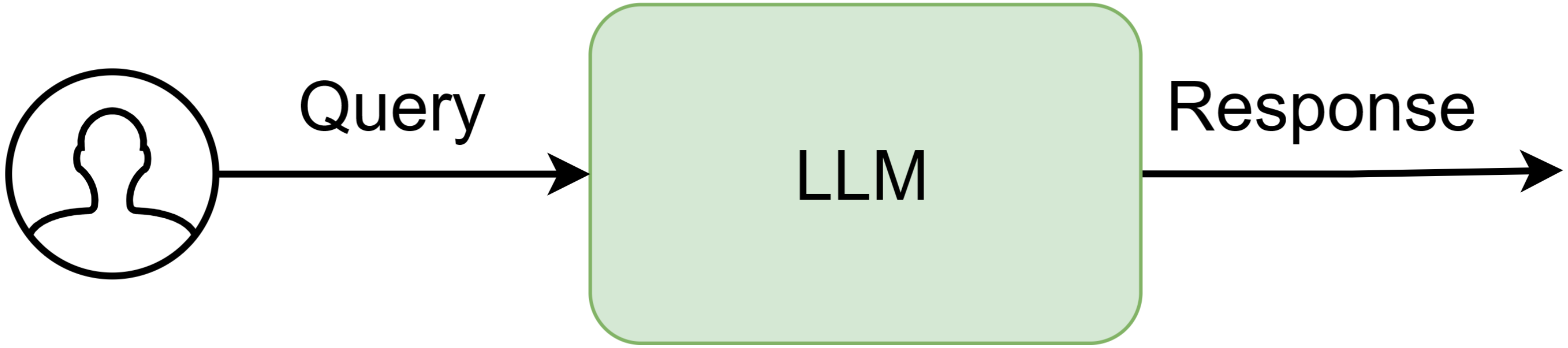
The Eiffel Tower is older than Bosch. The Eiffel Tower was completed in 1889, while Bosch was founded in 1886.

Which one is older Bosch or the Eiffel tower?
Give your answer step-by-step!

1. The Eiffel Tower was completed in 1889.
2. Bosch is a company that was founded in 1886.
3. Therefore, Bosch is older than the Eiffel Tower by 3 years.

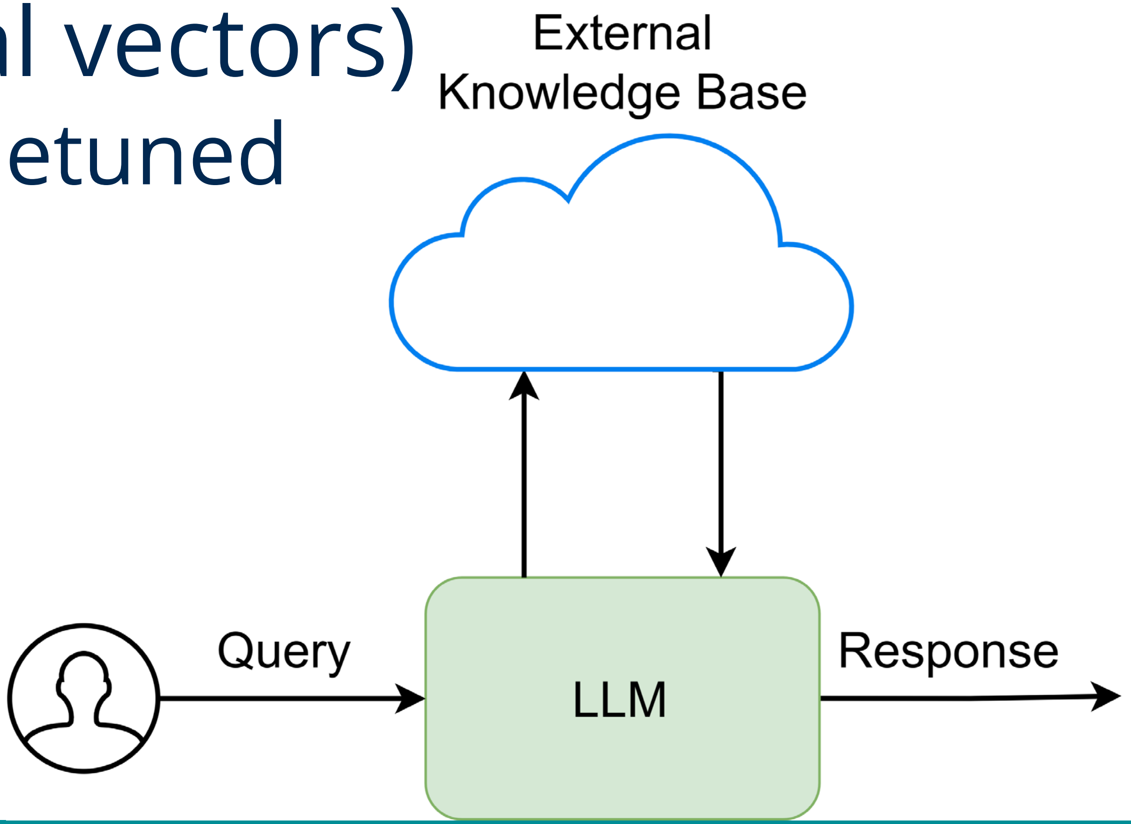
Provide external knowledge

- Specific knowledge is missing from LLMs
- Models frequently hallucinate



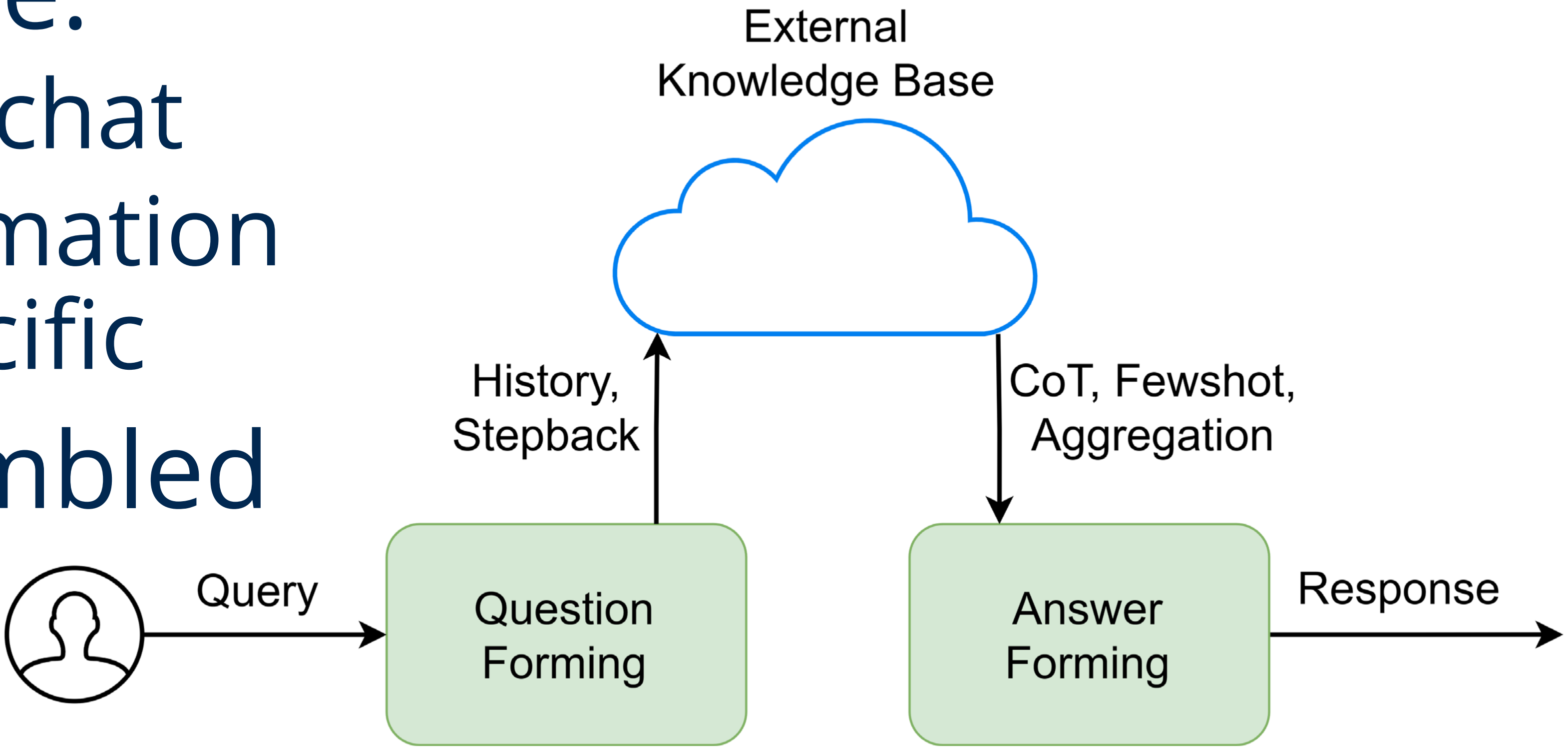
Retrieval Augmented Generation (RAG)

- Using embedding models to encode the meaning of text chunks (1-3k dimensional vectors)
  - Usually encoder-style LLMs finetuned for similarity search
- VectorDB – similarity search
- Hybrid search possible
- Adding knowledge from the same topic to the prompt



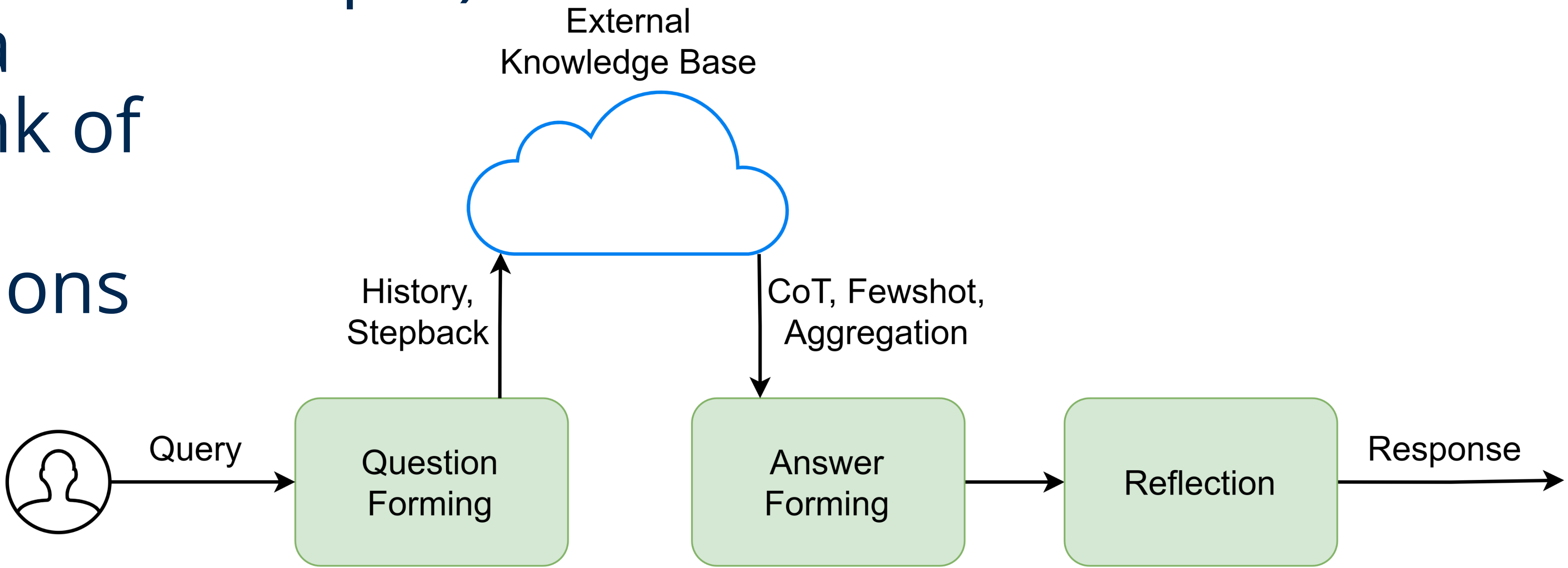
2-Step RAG

- Questions are not always similar to the relevant knowledge:
  - References in the chat
  - Background information is better than specific
- Answers are assembled in a separate step



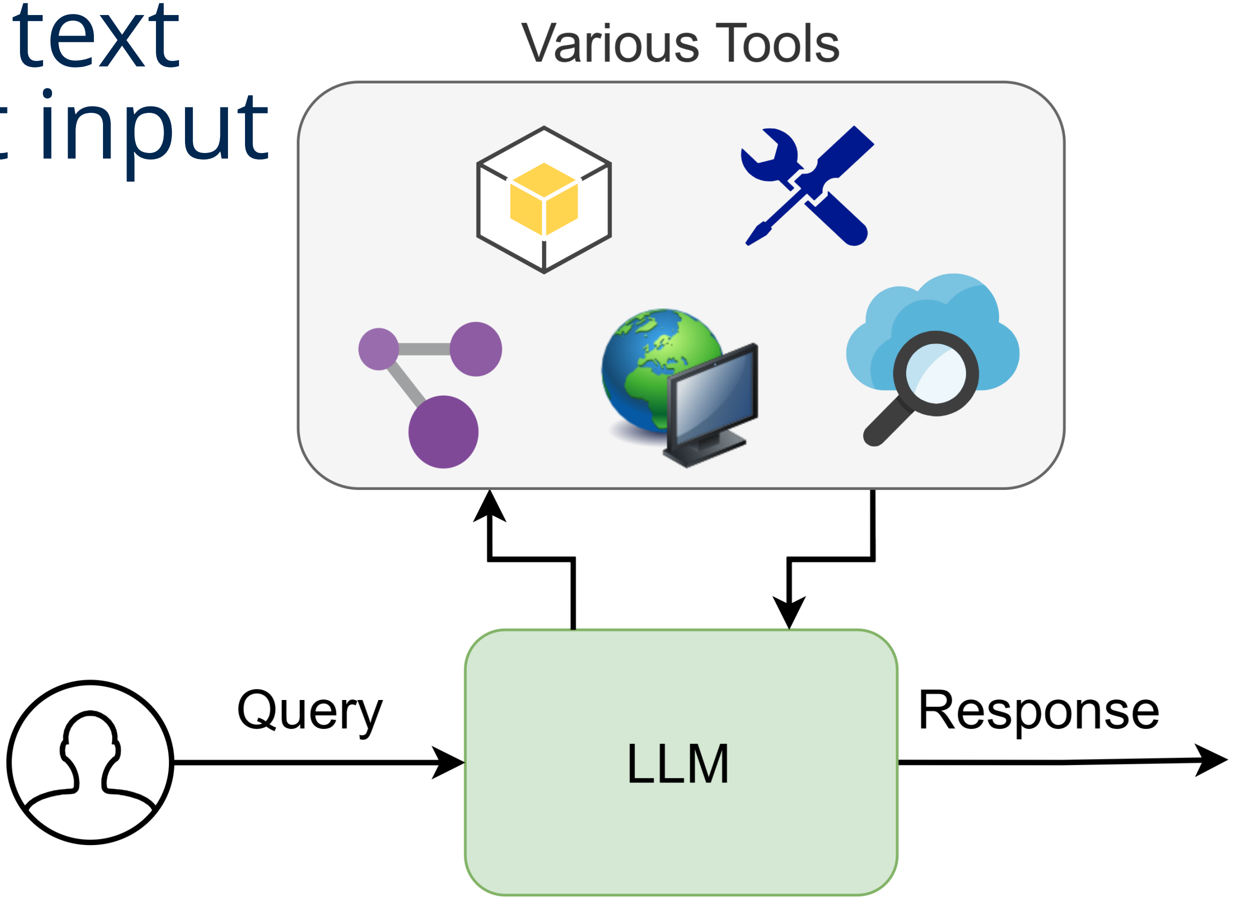
Reflection

- An extra reflection node (which reflects on the suitability of the output) can filter a large chunk of remaining hallucinations

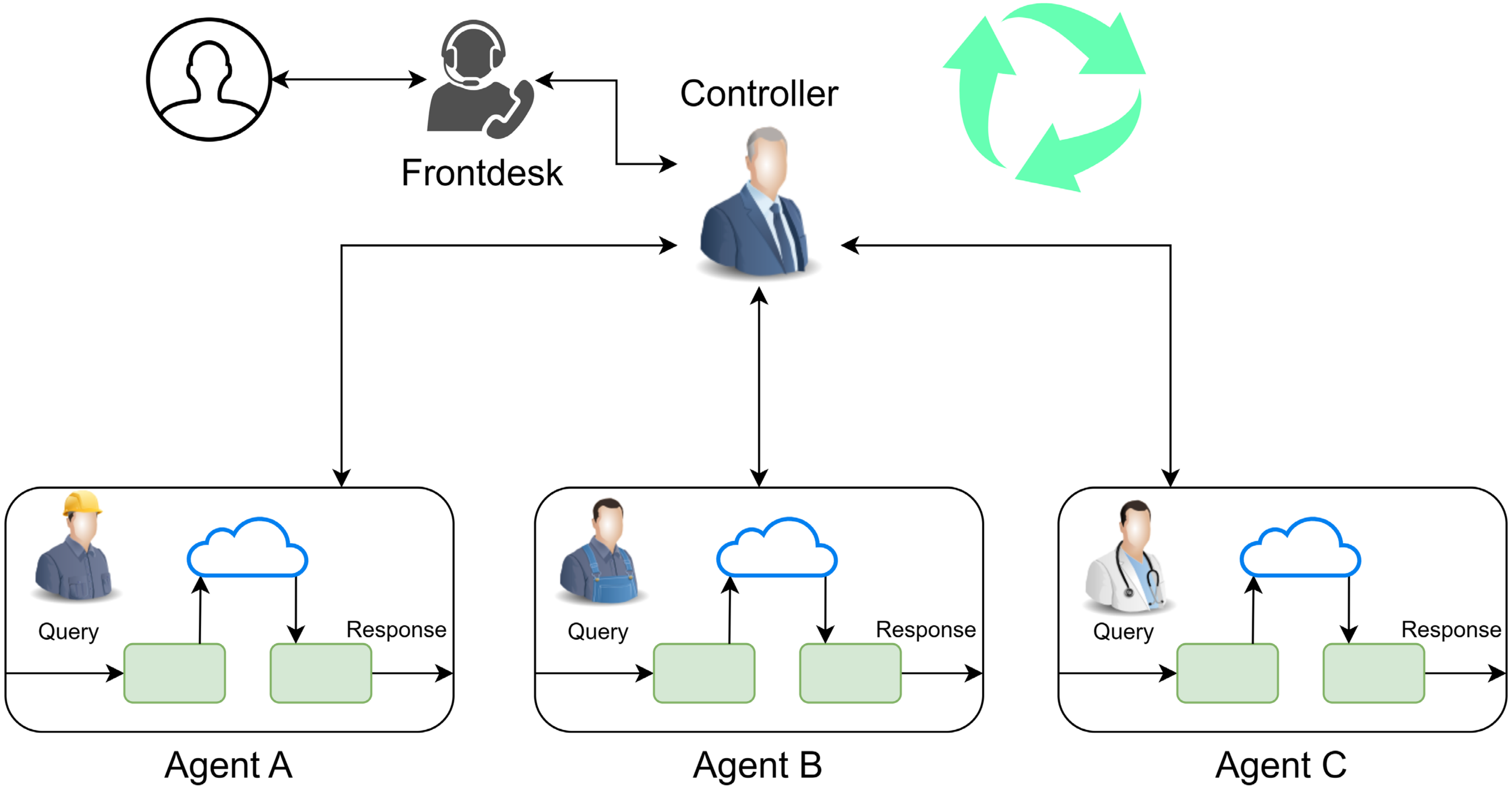


Tooling

- Tools that can be called via text parameters and return text input are suitable for LLM chains
  - Programming kernel
  - Search engines
  - Knowledge graphs
  - Action handlers
  - Application APIs
- JSON is a popular format

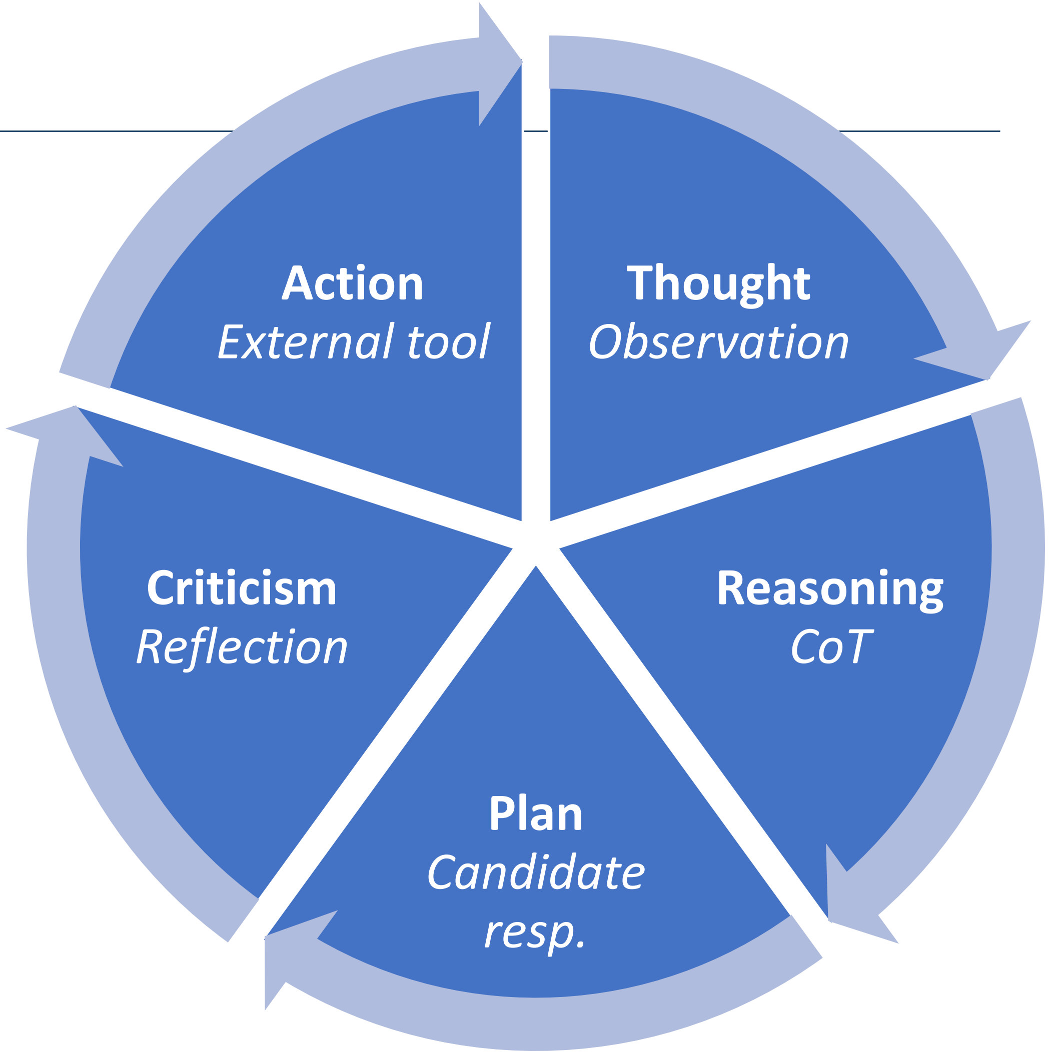


Agents


Controller
Frontdesk
Query
Response
Agent A
Query
Response
Agent B
Query
Response
Agent C

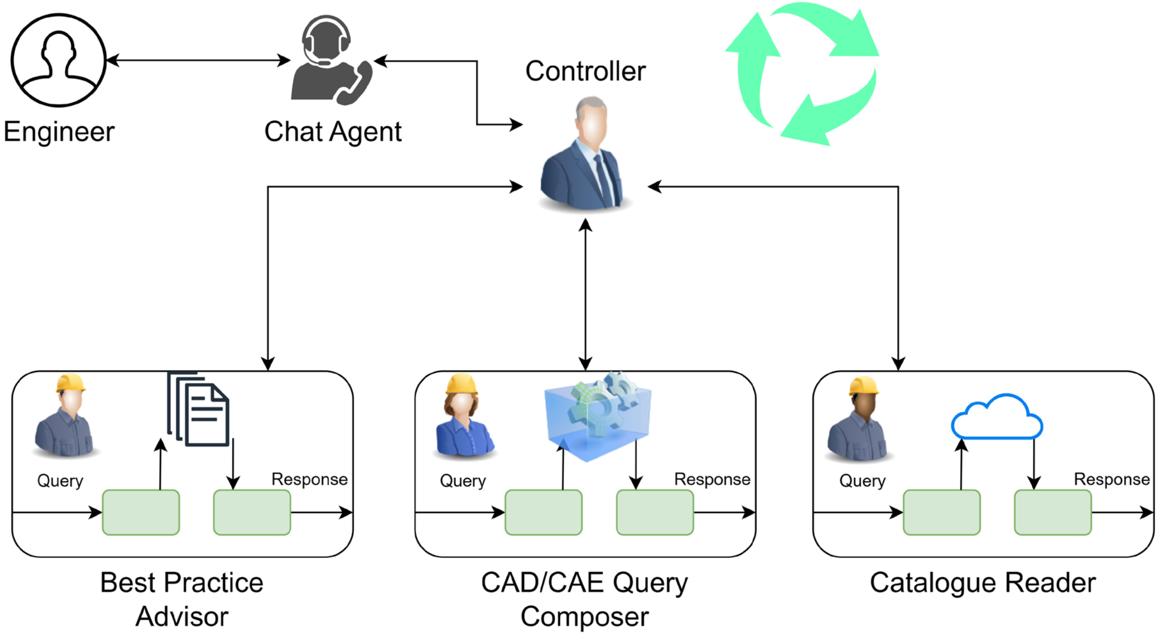
Agents

- Specialist Agents are LLMs with external knowledge/tools that specialize in performing a single task
- Controller is an LLM, that uses a loop of planning, task execution and evaluation



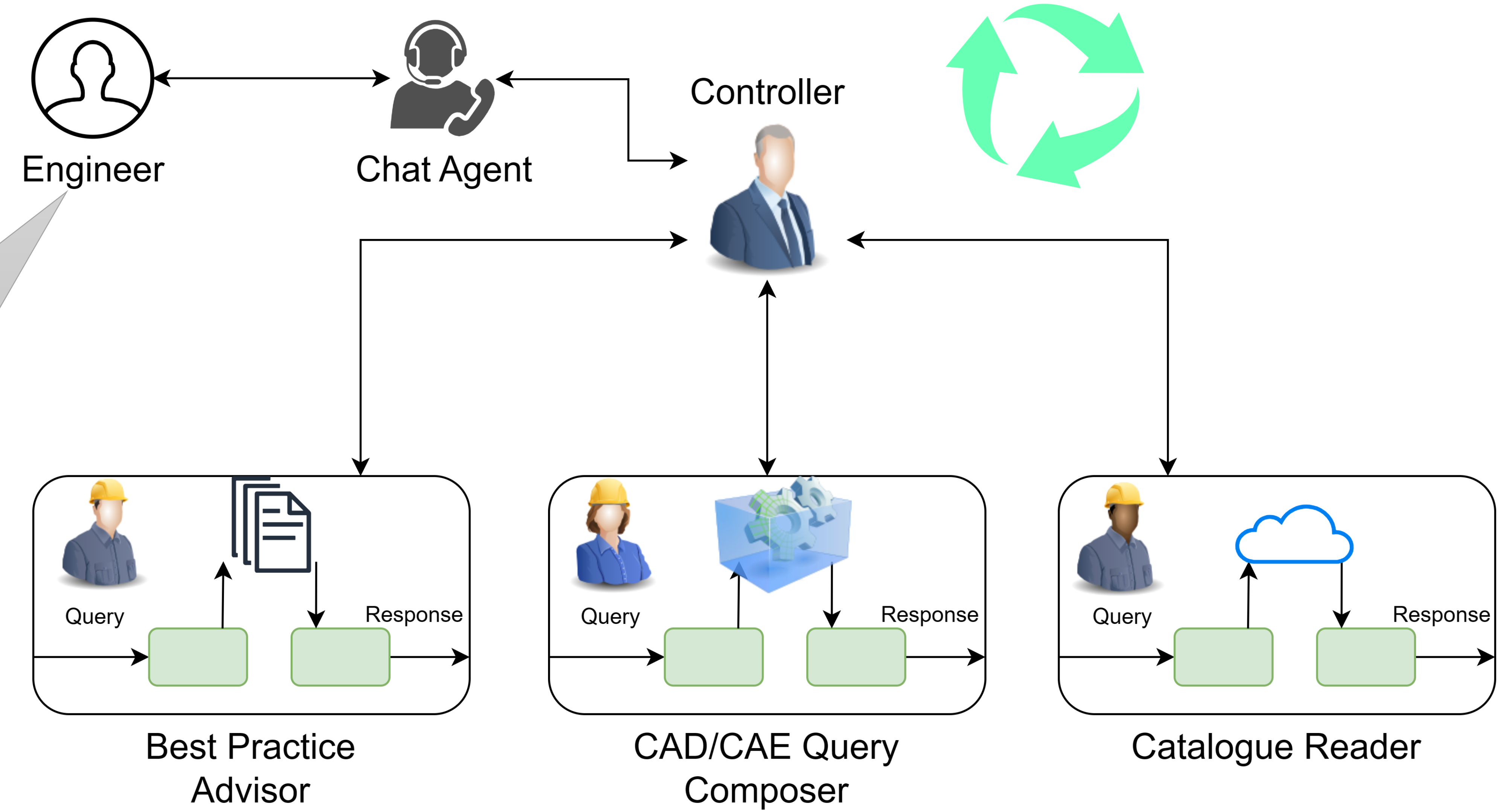
Example: Design Assistant

- Integration of scientific tools and LLM pipelines
- Immature stage: 1st conference on LLM-aided design this June
- Agentified chat systems are viable

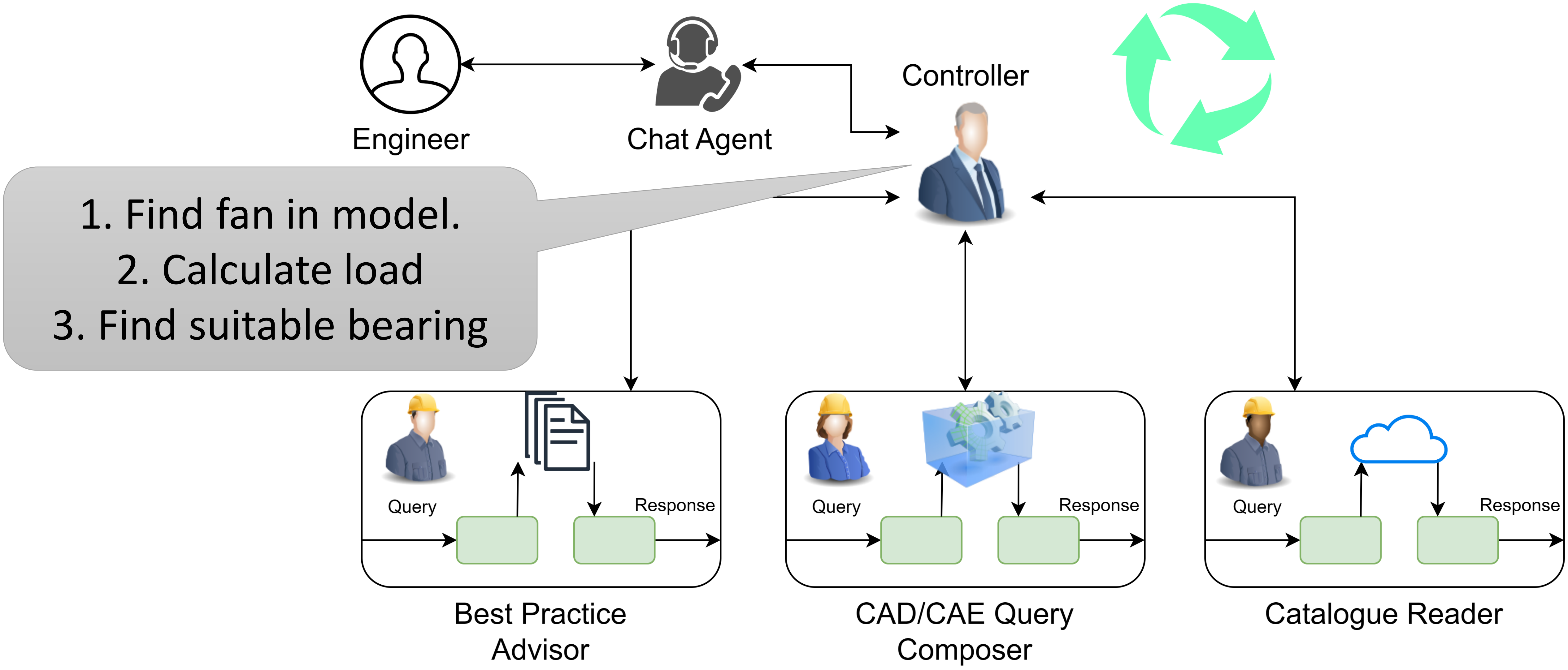


Example: Design Assistant

Provide a bearing for the fan!

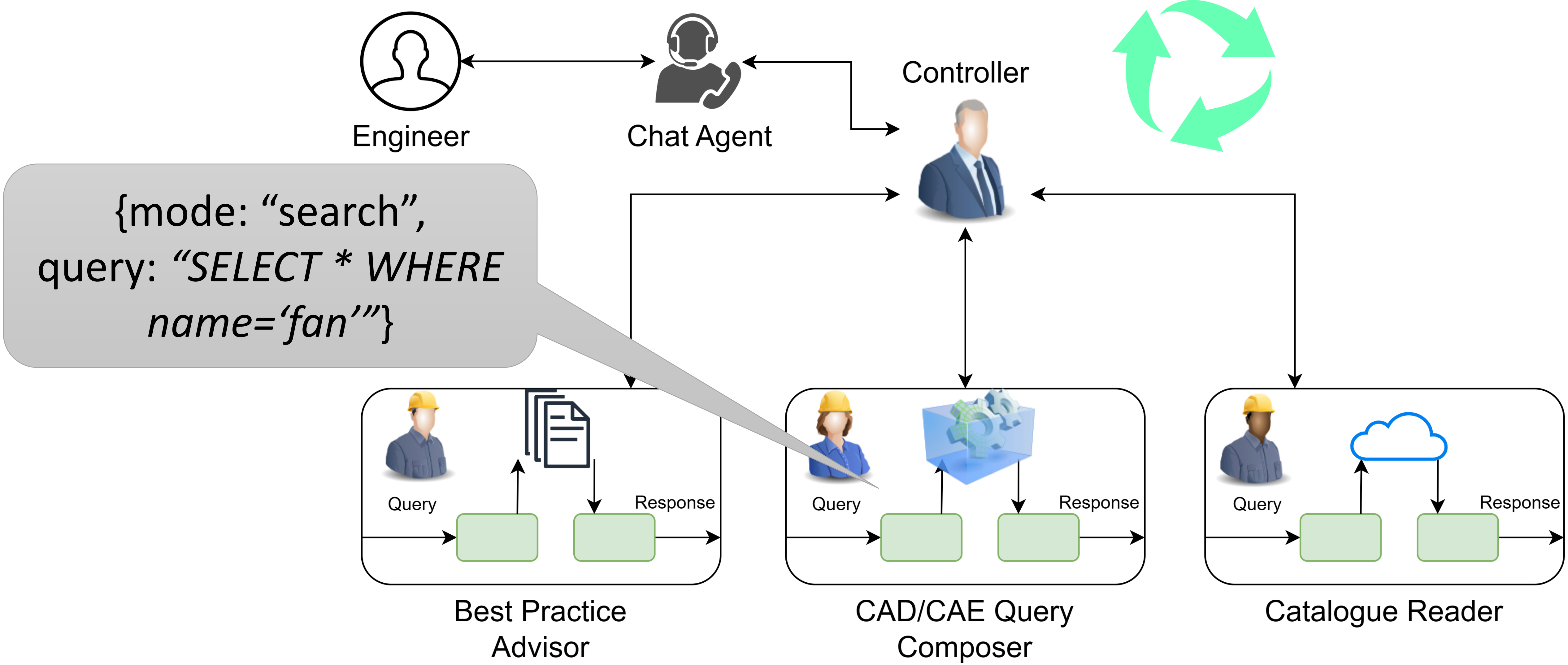


Example: Design Assistant



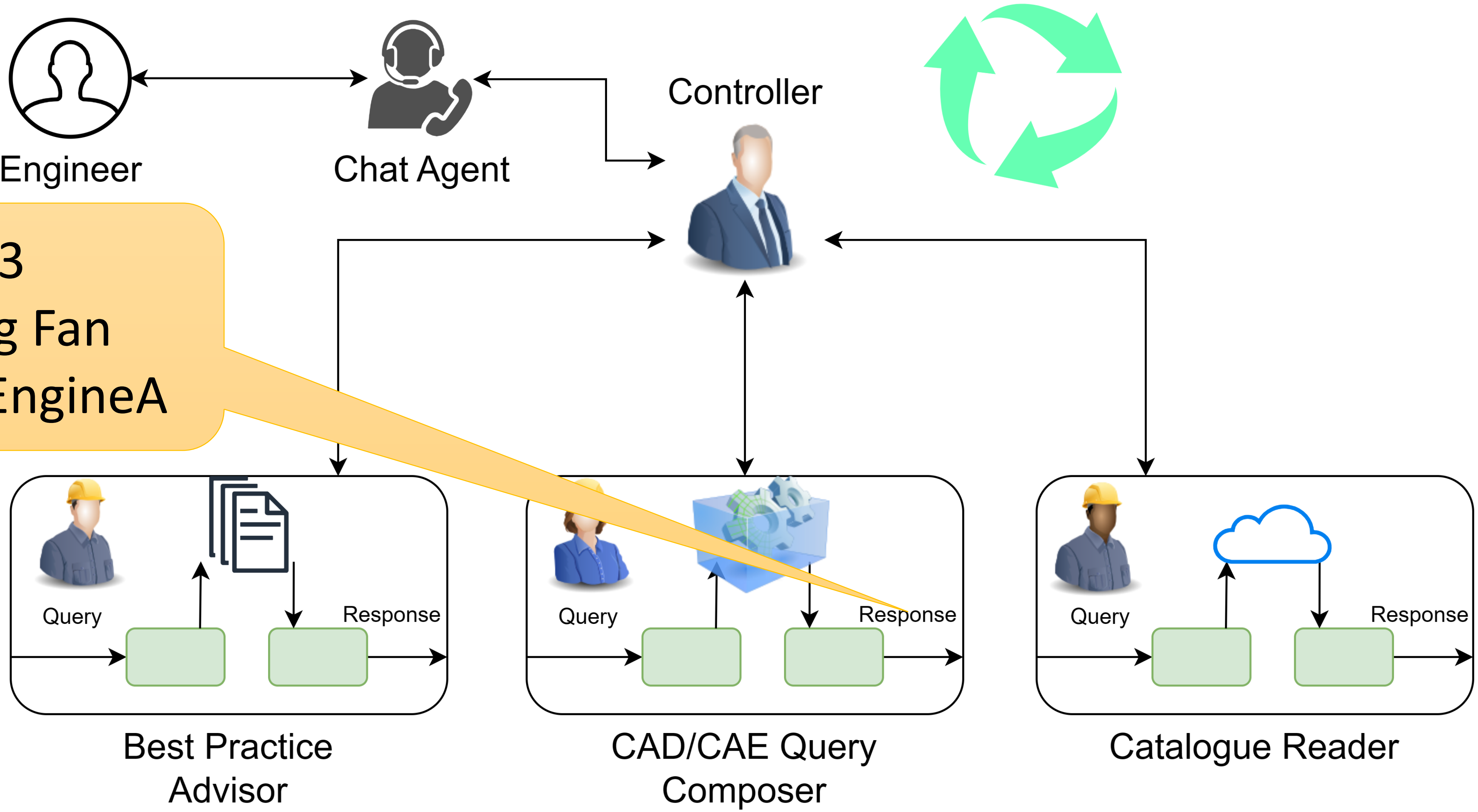
1. Find fan in model.
2. Calculate load
3. Find suitable bearing

Example: Design Assistant



{mode: “search”,
query: *“SELECT * WHERE name=‘fan’”*}

Example: Design Assistant



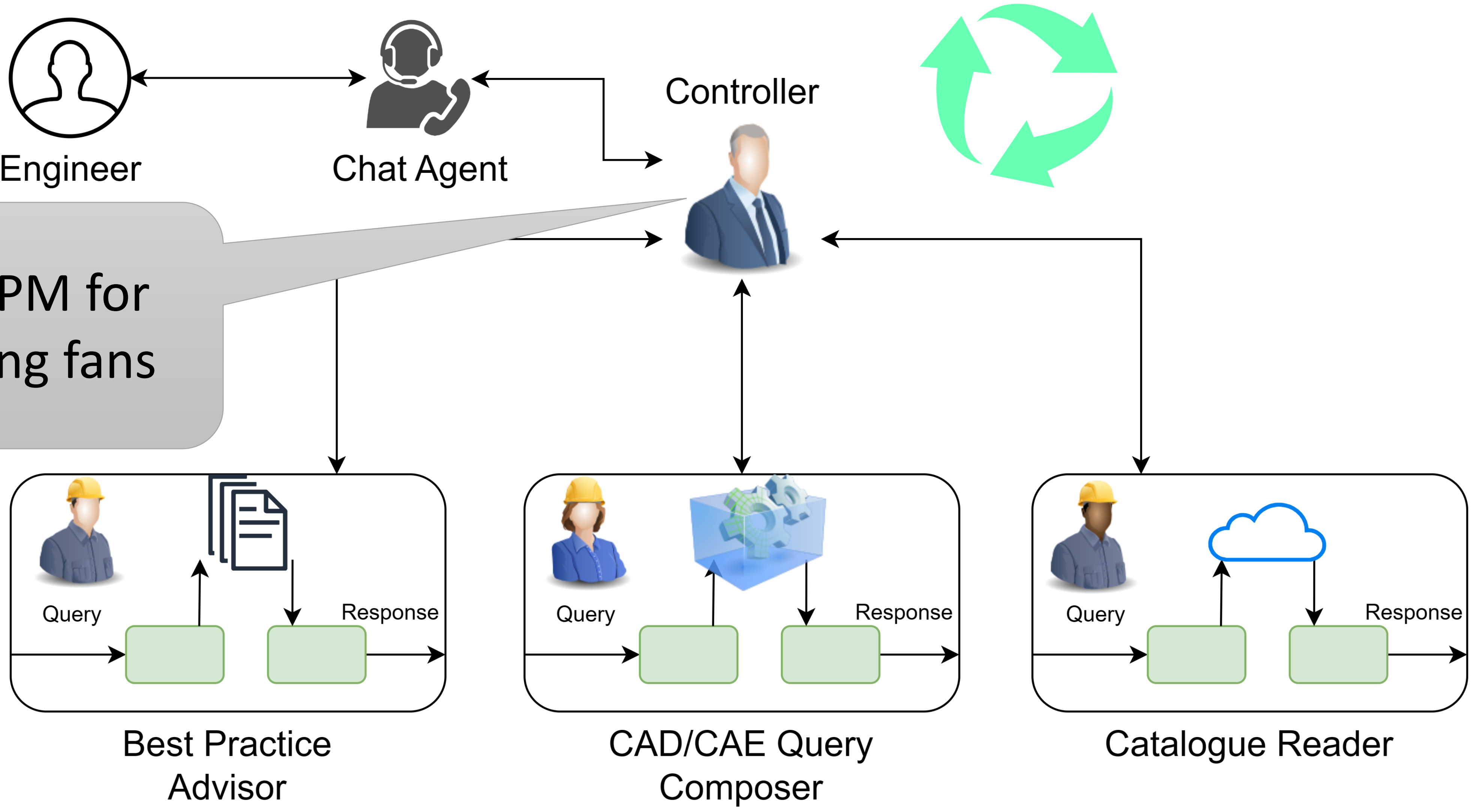
PartID = 23

Type = Cooling Fan

ParentModel = EngineA

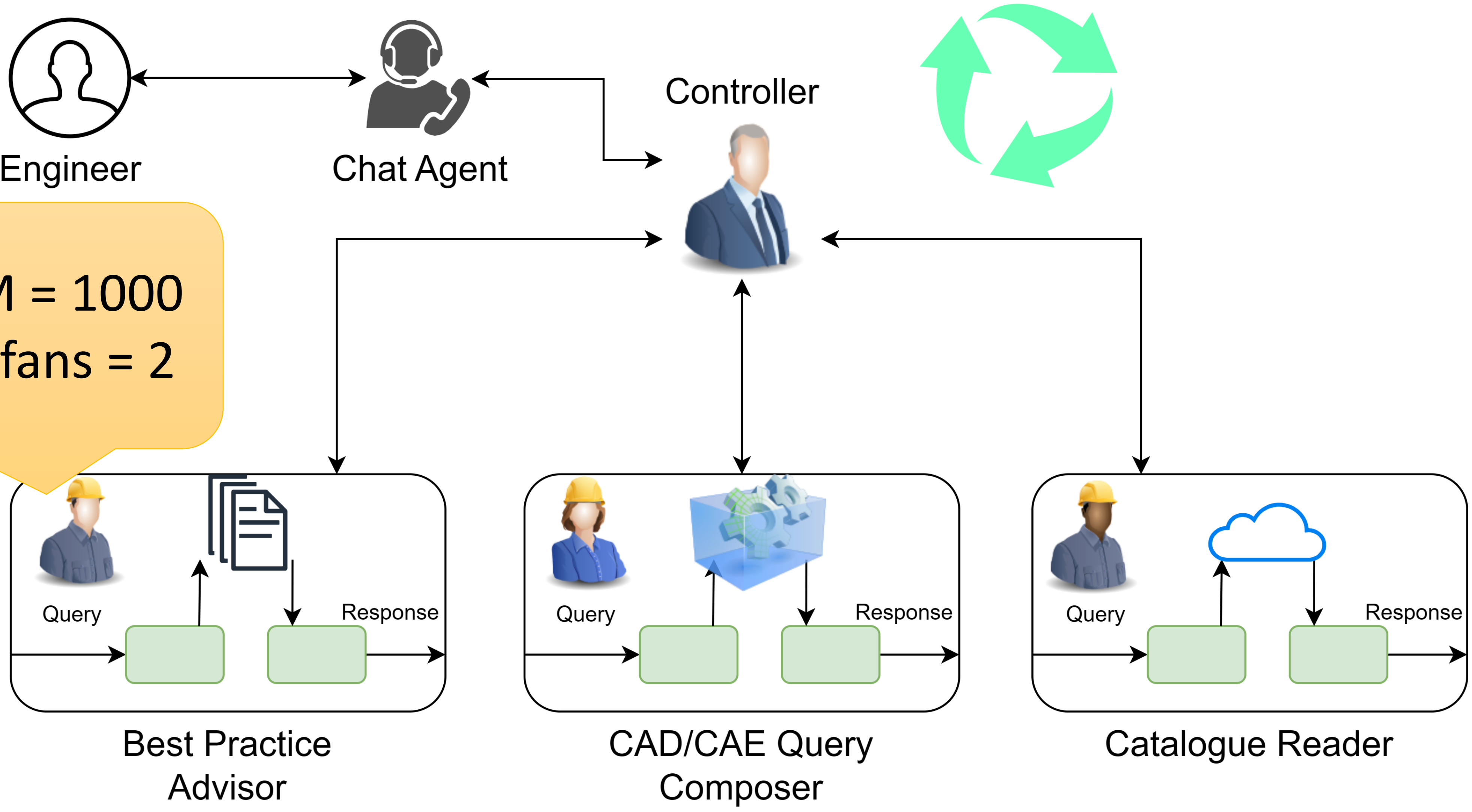
Example: Design Assistant

Find maximal RPM for “EngineA” cooling fans

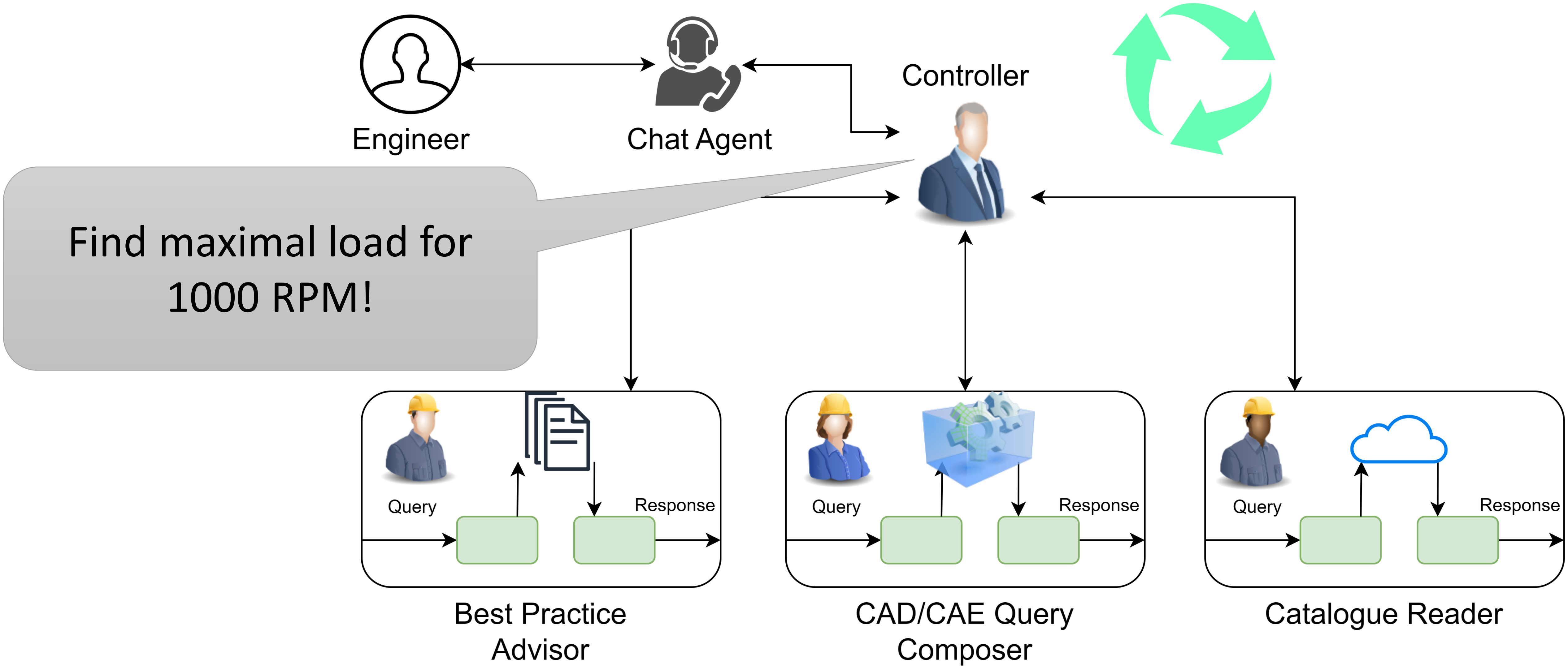


Example: Design Assistant

EngineA maxRPM = 1000
Safety factor for fans = 2

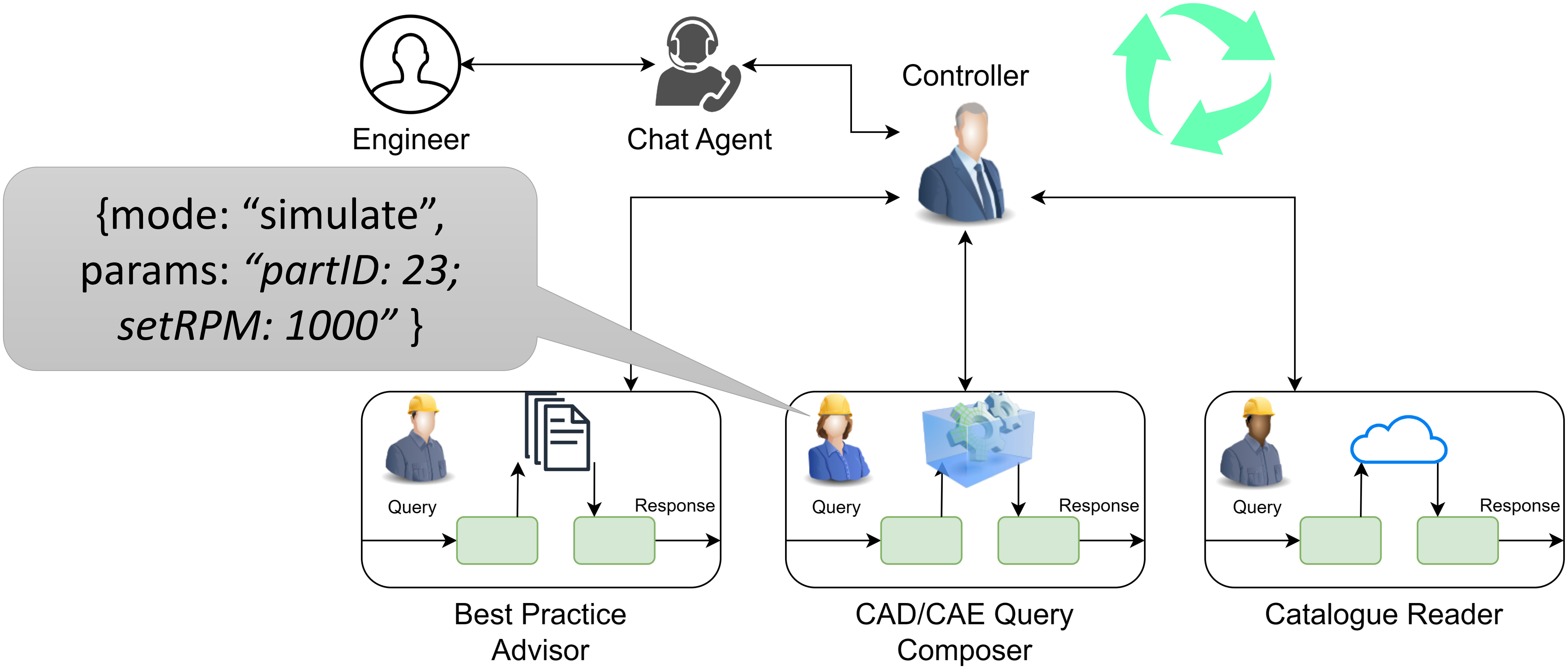


Example: Design Assistant



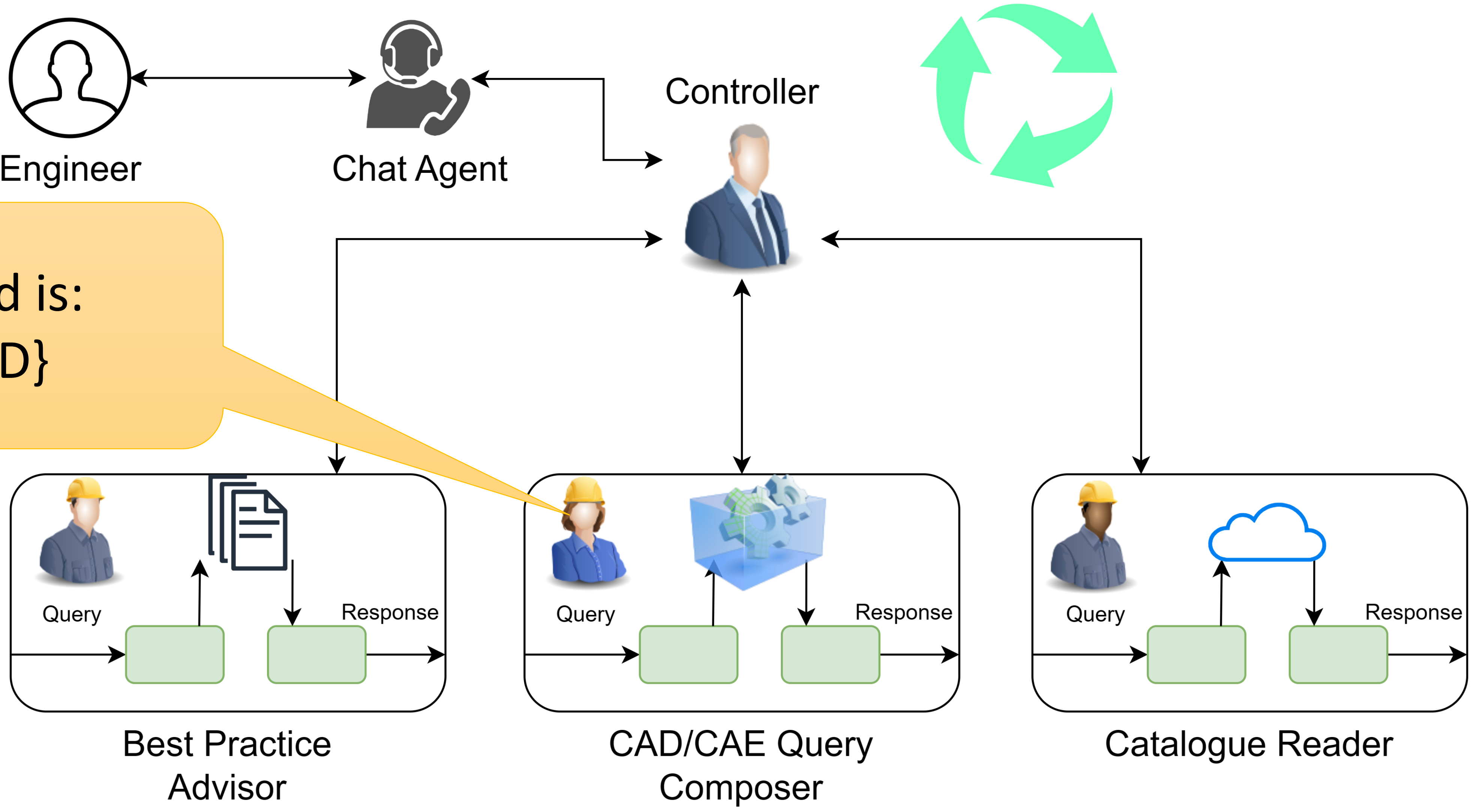
Find maximal load for 1000 RPM!

Example: Design Assistant



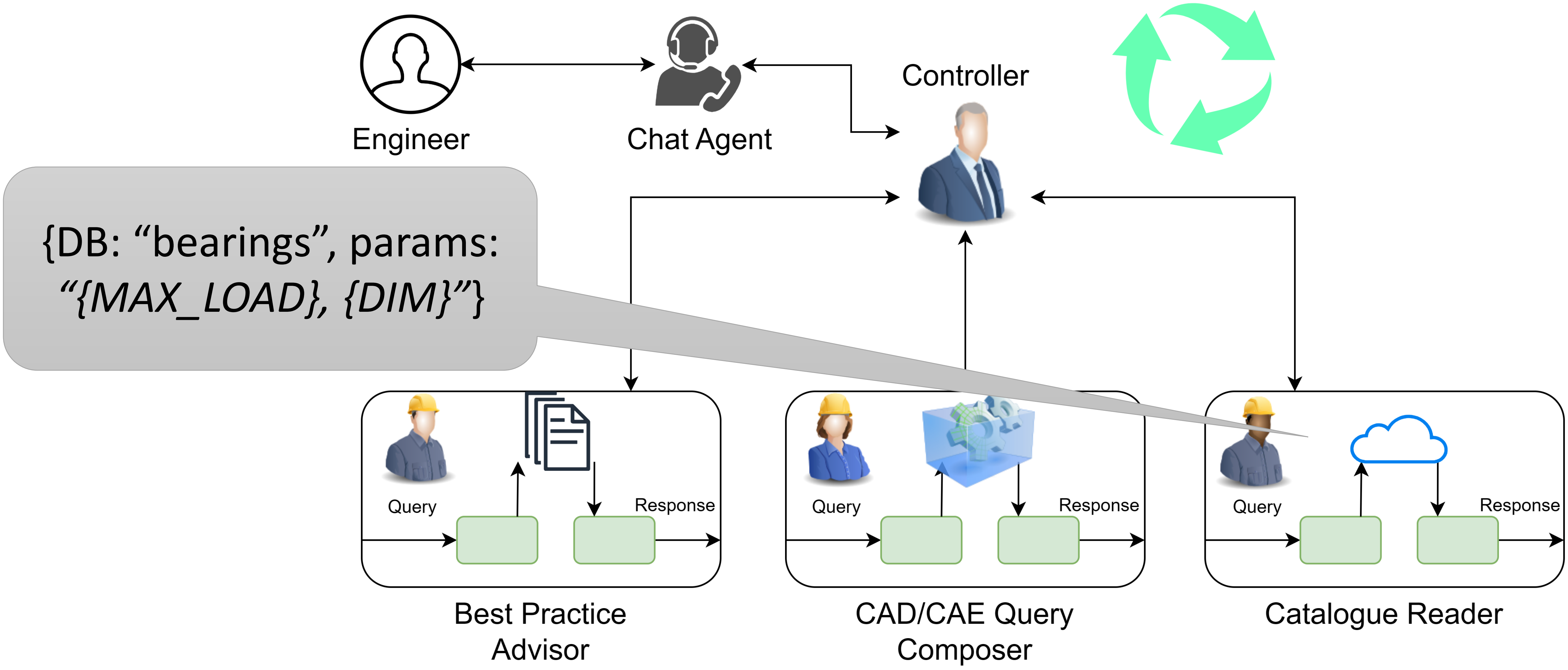
{mode: “simulate”,
params: *“partID: 23;*
setRPM: 1000” }

Example: Design Assistant



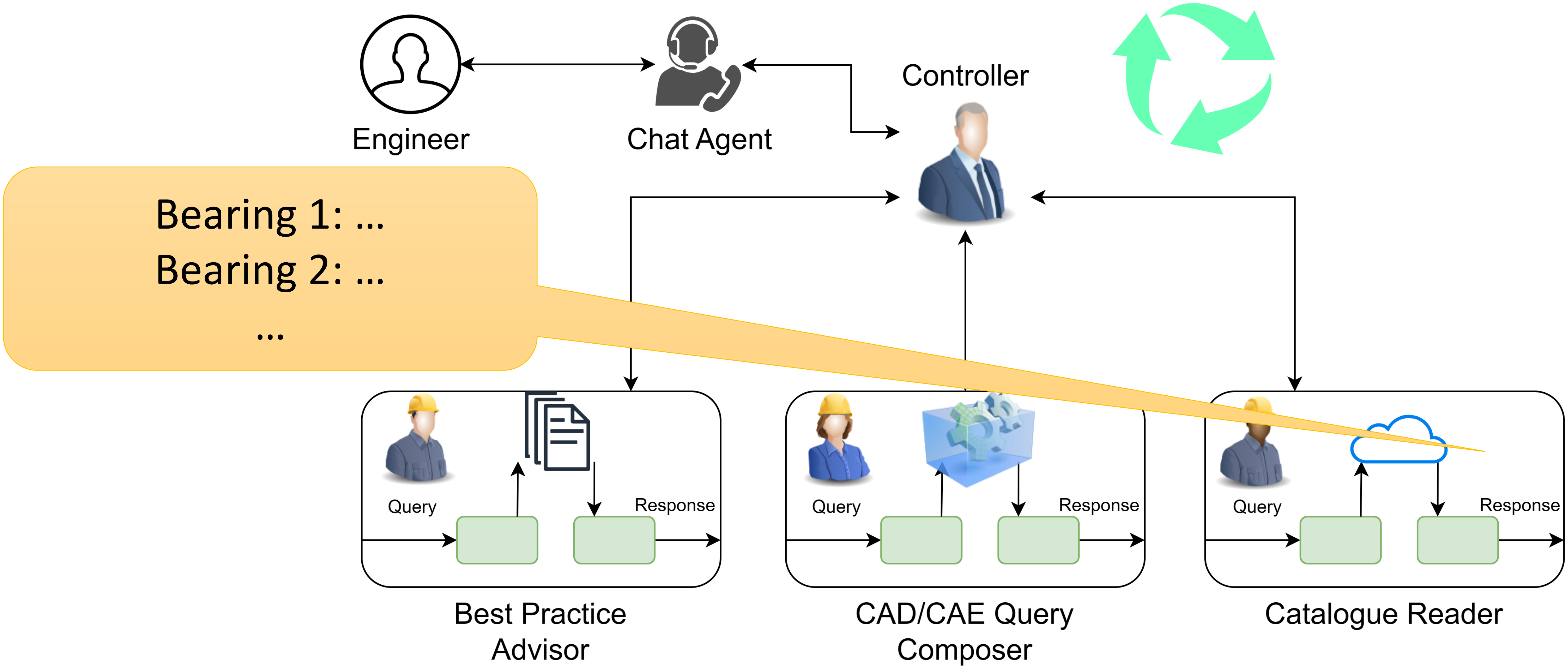
Maximal load is: {MAX_LOAD}

Example: Design Assistant



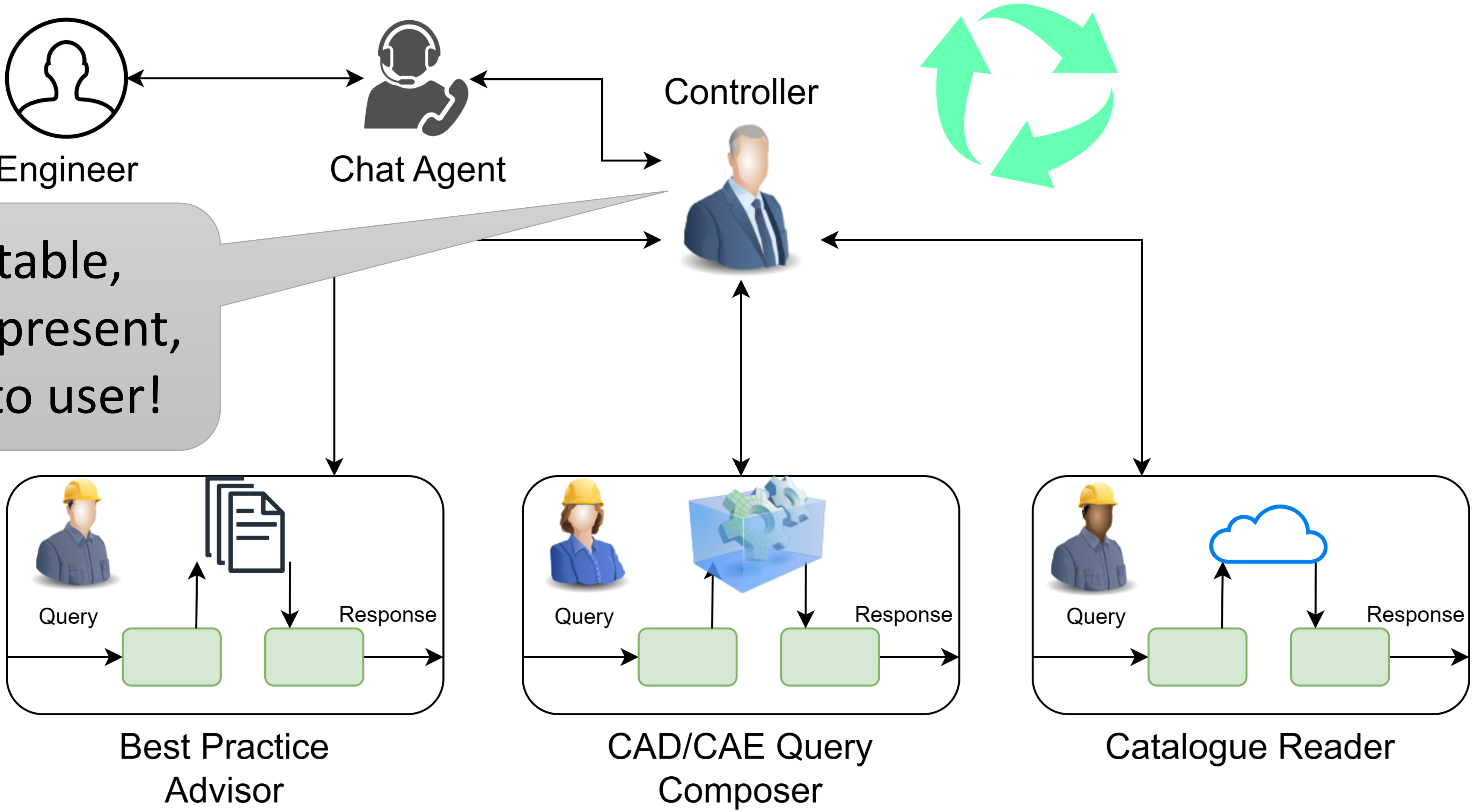
{DB: “bearings”, params: *“{MAX_LOAD}, {DIM}”*}

Example: Design Assistant



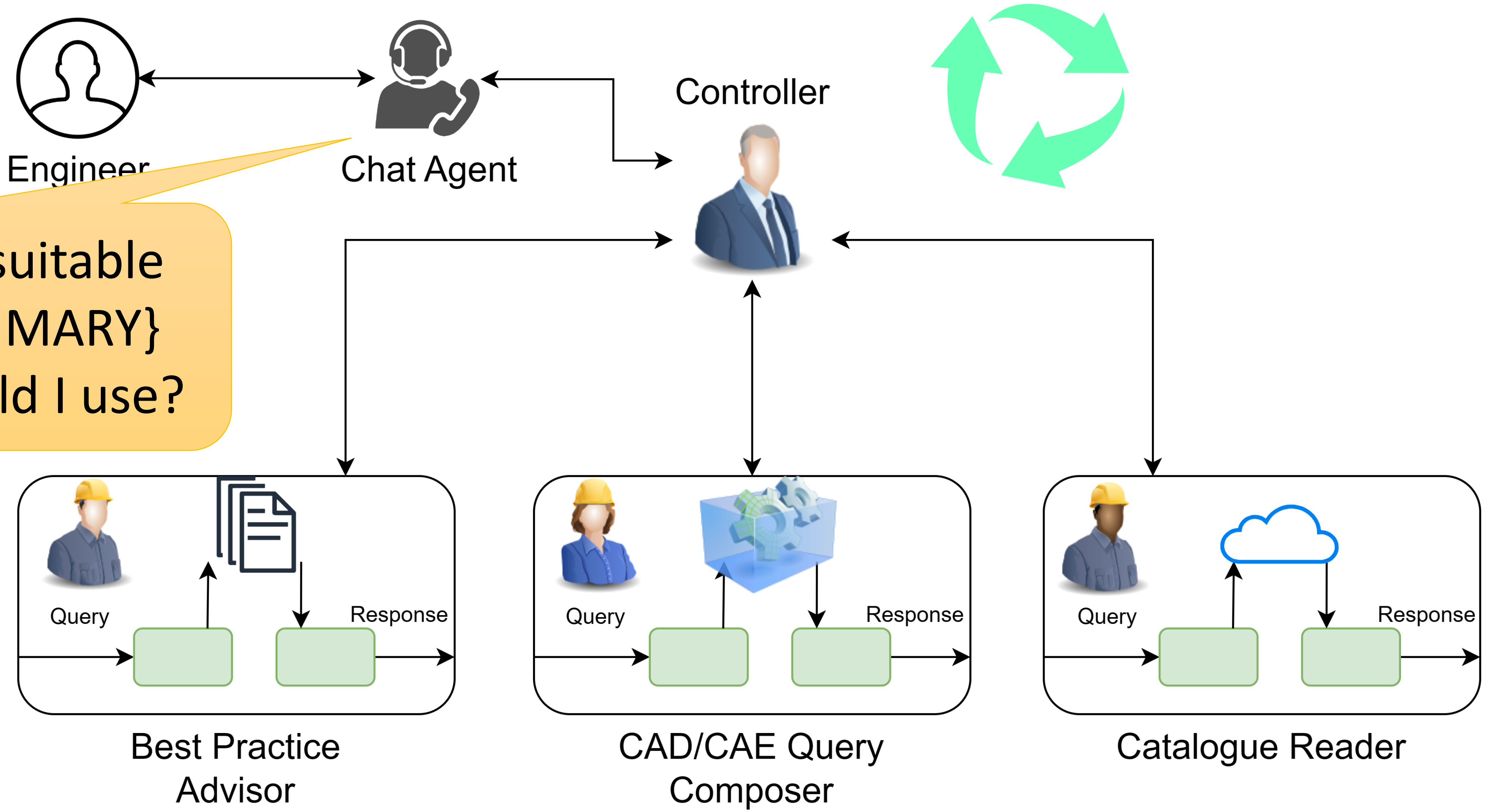
Bearing 1: …
Bearing 2: …
…

Example: Design Assistant



Results are suitable, multiple options present, return decision to user!

Example: Design Assistant



I have found 5 suitable bearings: {SUMMARY} Which one should I use?

Coding Agents

- Planning included
- Tools are mostly code-related
- Might perform RAG on the repository

- Example:
- Pythagora.io

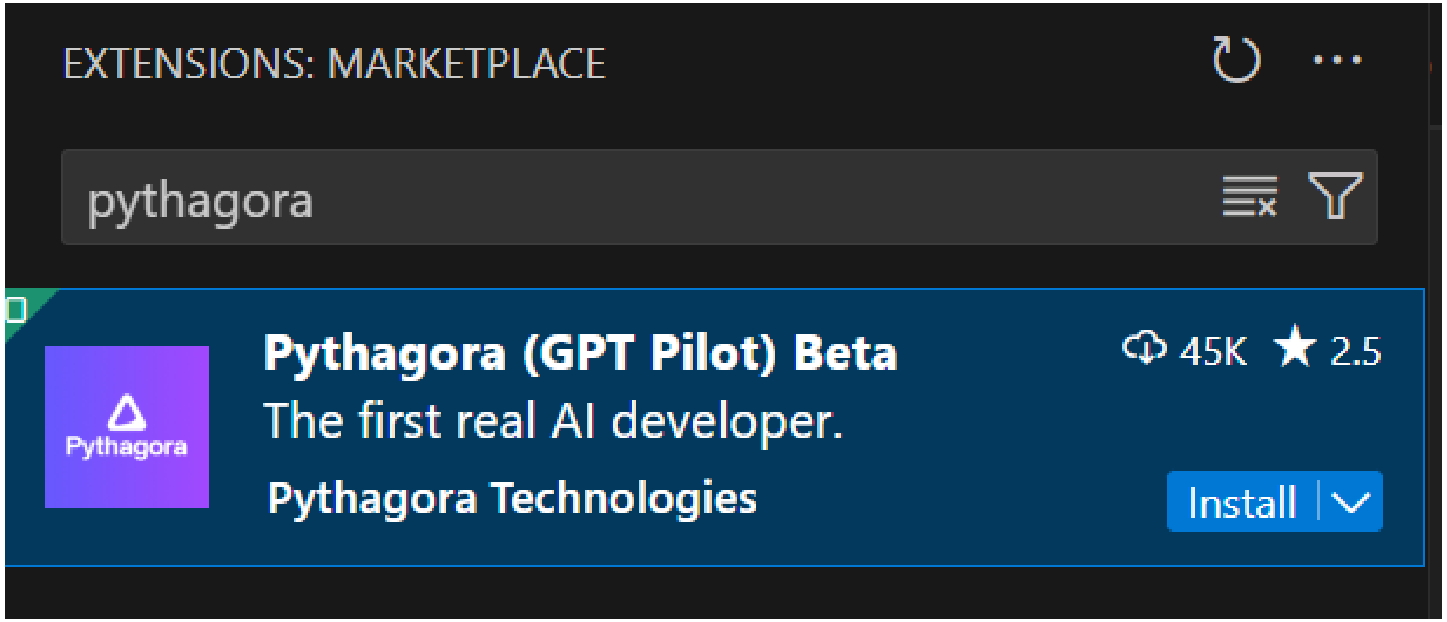
Get an OpenAI API key

- Platform.openai.com
- Dashboard
- API keys
- Create new
- Save it for later

OpenAI

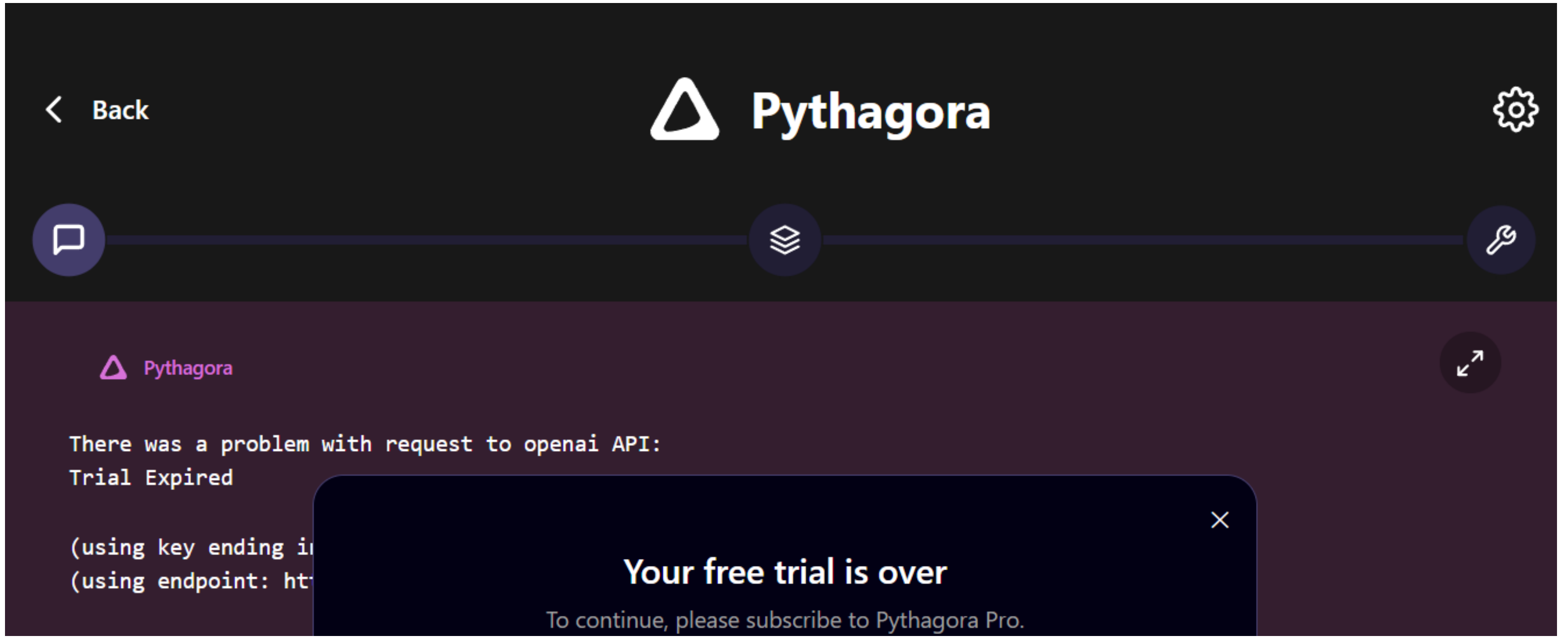
Get the Pythagora Extension

- Install from VSCode Extensions



Get the Pythagora Extension

- Open the Pythagora tab, wait for it to setup
- Click on the Settings cogwheel



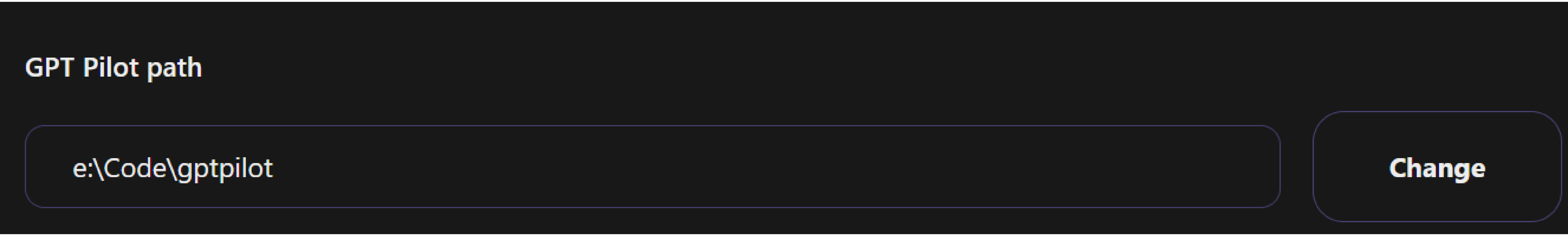
```
There was a problem with request to openai API:
Trial Expired

(using key ending in
(using endpoint: ht
```

Your free trial is over

To continue, please subscribe to Pythagora Pro.

Get the Pythagora Extension

- Set and copy your GPT Pilot path to **any value**
- Navigate to that folder



GPT Pilot path

e:\Code\gptpilot

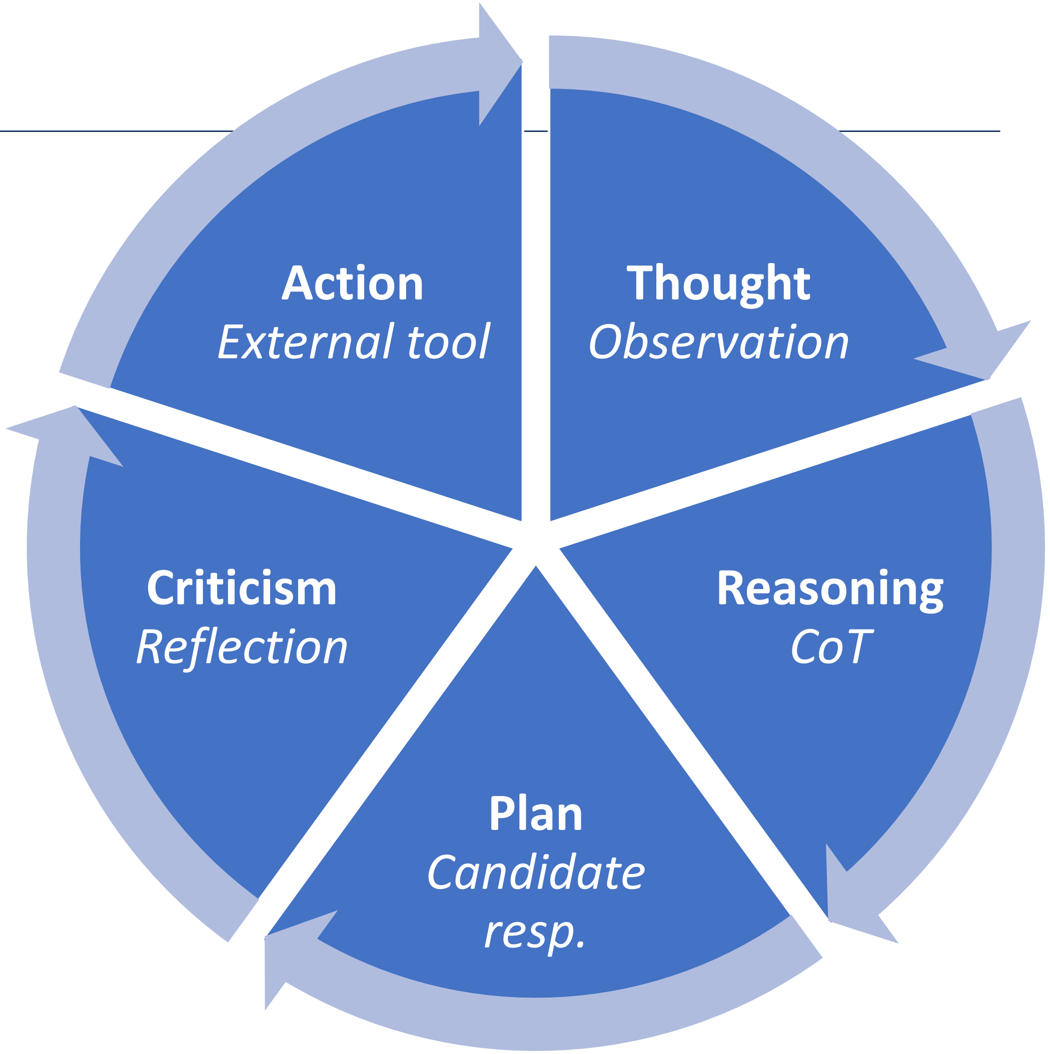
Get the Pythagora Extension

- Edit ./pilot/.env
- **OPENAI_ENDPOINT=https://api.openai.com/v1/chat/completions**
- **OPENAI_API_KEY=<YOUR KEY>**
- **MODEL_NAME=gpt-4o**

- Restart VSCode

Enjoy the interview

- Create an app answering the questions by Pythagora
- Watch it plan, act, revise!



UI Dev Tool

- OpenUI
- Clone from GitHub
  - git clone https://github.com/wandb/openui
  - $env:OPENAI_API_KEY = <your key>
  - cd openui/backend
  - pip install .
  - python -m openui

UI Dev Tool

- Set the model to gpt-4o in the settings
- Draw or screenshot a UI!
- Define local modifications
- Dynamic connections
- Basic animations
- etc