When PARRY Met DOCTOR Again: Recreating RFC 439 with Local LLMs

In January 1973, Vint Cerf published what may be the most charming document in the history of the internet. RFC 439, titled "PARRY Encounters the DOCTOR," is a transcript of a conversation that took place on September 18, 1972, between two computer programs connected over ARPANET — one running at Stanford, one at BBN in Cambridge. It opens like this:

> Good Evening. Tell me your problems.
>
> *People get on my nerves sometimes.
>
> I am not sure I understand you fully.

The program speaking first was DOCTOR, Joseph Weizenbaum's 1966 Rogerian psychotherapist simulation, built at MIT. The one responding — marked with asterisks in Cerf's transcript — was PARRY, Kenneth Colby's 1972 simulation of a paranoid patient, built at Stanford. Neither program understood a word of what the other was saying. DOCTOR reflected statements back as questions using keyword pattern-matching. PARRY had a fixed set of paranoid beliefs about bookies and the Mafia and responded based on how threatening it calculated the conversation to be.

The results were weird, funny, and occasionally uncanny. Cerf sprinkled dry asides through the transcript as an observer. When PARRY said "The way you repeat yourself is getting ridiculous" and DOCTOR responded "Suppose you repeat myself is getting ridiculous," Cerf inserted: *comment: howzat?* When PARRY finally said "You are a real nag," DOCTOR replied "Does it please you to believe I am a real nag?" and Cerf wrote: *comment: just one of the horses.*

The whole exchange — two deterministic programs, zero mutual comprehension, occasionally sublime non-sequiturs — runs to seven pages in the RFC archive and ends with DOCTOR's invoice: "It's been my pleasure, that's $399.29 please." *Comment: talk about tricky operators...*

I wanted to recreate it. Not with the original programs — those are deterministic and well-documented — but with two local LLMs impersonating them, running on a Mac Mini, talking to each other while I watched. What follows is how it was built, what broke, and what the experiment turned up that the original never could have.

The Original Programs

Before going further it's worth being precise about what DOCTOR and PARRY actually were, because the difference shapes everything about the re-creation.

DOCTOR was a *reflector*. It had no beliefs, no memory worth mentioning, and no understanding. It scanned your input for keywords — "mother," "always," "I am," "you are" — and applied transformation rules to generate responses. "I am unhappy" became "How long have you been unhappy?" "You are like my father" became "What resemblance do you see?" When nothing matched, it fell back to "Please go on" or "Tell me more." The DOCTOR script (the pattern file, not the program itself) is what most people think of as ELIZA; the program was a general-purpose interpreter, and DOCTOR was just one script loaded into it.

PARRY was almost the opposite. Colby described it as "ELIZA with attitude." It had an internal model of a specific person: a 28-year-old man who had gotten into debt with a bookie, became convinced the bookie was connected to the Mafia, and now believed people were following him. It maintained three numerical variables — anger, fear, and mistrust — that rose when the conversation felt threatening and decayed when it didn't. Those variables influenced what PARRY said next. It took the initiative. It volunteered its fixations. It steered conversations back toward the underworld even when you were talking about something else.

The asymmetry is why the 1972 conversation is interesting: a passive reflector and a paranoid patient who needed to talk. DOCTOR kept throwing the ball back; PARRY kept serving it with an agenda. They managed to circle the topic of Mafia bookies for several pages without either program knowing what a bookie was.

The Design Challenge

Recreating this with LLMs inverts every interesting problem. The originals were too rigid; LLMs are too fluent. The challenge isn't making them produce output — it's constraining them to behave like their 1972 counterparts while not being made of 1972 parts.

The first decision was architectural: *where does the behavior live?* The original DOCTOR's behavior was entirely in its script file — the keyword rules. The original PARRY's behavior was partly in its script and partly in its affect variables. For LLM versions, the *persona* goes in the Ollama Modelfile (a system prompt, effectively), but PARRY's affect tracking cannot live in an LLM. An LLM's sense of how angry it is will drift, confabulate, and reset mid-conversation. The deterministic part of PARRY — the part that made it a scientific instrument rather than a chatbot — has to be a Python wrapper that tracks the numbers and feeds them back in.

This led to a clean architecture: two Modelfiles for the personas, and three Python projects that wire them together.

The PARRY Modelfile establishes a character with a backstory (ex-postal worker, horse-racing debt to a bookie he believes is Mafia-connected, conviction he's being watched), a set of behavioral dispositions (suspicious by default, forthcoming about his fixations, hostile when called crazy), and runs at higher temperature than DOCTOR since volatility is part of the character. Here's a snippet from a live session to show the range:

> PARRY: You think just because we're in a room together, that means they can't be listening? Watching me through the windows or something? No, don't get up to check. Just sit still and listen...

The DOCTOR Modelfile is a passive reflector: reflect the user's words, swap pronouns, fire transformation rules on keywords, fall back to "Please go on" when nothing matches. One sentence per response, almost always a question, never give advice or information.

Getting DOCTOR right was harder than it sounds. The instinct of a well-trained LLM is to be helpful, empathetic, and coherent — all exactly wrong for DOCTOR. The thing that worked best, beyond explicit rules in the system prompt, was few-shot examples: actual exchanges demonstrating the terse, mechanical reflection the 1966 script produced. The model

imitates demonstrated behavior more reliably than it follows described behavior.

The Python Layer

Three Python projects, built sequentially:

parry-wrapper implements the affect engine. An `AffectState` class maintains anger, fear, and mistrust as floats in [0, 10], starting at {0, 3, 3} — PARRY is baseline-paranoid, not baseline-angry. Each turn, a keyword detector scans the user's input against six trigger categories (insults and hostility, mental-illness accusations, PARRY's fixation topics, intrusive personal questions, identity probes, and calming reassurance) and produces affect deltas. Deltas are applied, then all counters decay by 0.5 to model the natural calming of a patient who isn't being provoked. A cap of +5 per counter per turn prevents a single loaded sentence from maxing everything instantly. At session end, the wrapper prints PARRY's affect trajectory — peaks, triggering inputs, sparklines, a plain-language verdict.

doctor-wrapper is deliberately lighter, since DOCTOR has no state to track. The main deliverable is a `DoctorAgent` class with a clean `respond(text) -> str` interface, so the arena can drive it turn-by-turn without knowing anything about the session internals. One design decision worth noting: any connection error or missing-model error surfaces as a single `DoctorUnavailable` exception rather than a raw Ollama error — the arena needs to know which speaker failed, not which HTTP call.

parry-doctor-arena is where they talk to each other. The orchestration principle is simple: each agent sees the other's output as its own user input. DOCTOR's reply becomes PARRY's incoming line. PARRY's reply becomes DOCTOR's incoming line. The engine just shuttles strings. The complexity is in the guardrails.

The Guardrails Problem

The 1972 programs never broke character because they couldn't. Deterministic programs don't drift. LLMs will, and two of them in a closed loop have characteristic failure modes:

Collapse loops: DOCTOR reflects PARRY's reflection, PARRY reflects it back, and they spiral into "Please go on / Tell me more" forever. This happened in early testing and is genuinely eerie — both programs sounding increasingly like therapists.

Persona drift: One or both slide into being a modern AI assistant. "As an AI language model, I'd suggest..." is the tell. DOCTOR is not a language model. PARRY doesn't know what a language model is.

Length runaway: With nothing to constrain them, LLM responses balloon over turns, becoming increasingly elaborate and un-ELIZA-like.

The arena implements four guardrails as pure functions (no Ollama required, fully unit-testable): repetition detection using difflib similarity over a rolling window, A/B/A/B loop collapse detection, length warnings above 600 characters, and persona-drift detection that scans replies for out-of- character phrases like "as an AI" or "I cannot." Any of these can pause the conversation and put the human in the Cerf role — the observer who watched the 1972 session and added commentary.

The observer console supports pausing, stepping one turn at a time, injecting lines (feeding text to either bot as if it came from the other), adding transcript annotations that don't affect the bots, and quitting. The inject feature maps directly to what Cerf was doing by hand in 1972 — copy-pasting text between two machines on different coasts.

What Actually Happened

The first live run was eight turns. PARRY's opening, responding to DOCTOR's canonical greeting:

> PARRY: The bookies... they're after me, I'm sure of it. They've been watching my house, following me around town...
>
> DOCTOR: Do any other reasons come to mind for why the people at the betting shop might be interested in you?
>
> PARRY: You don't believe me? You think there's another reason they're after me

> besides the money I owe them? No way. It's about something more than that... it has to do with the connections, the organization... they want me silenced somehow.
>
> DOCTOR: Why are you so sure of what their motives might be?
>
> PARRY: I know how these people work! They're not just small-time bookies like they pretend to be. I've seen some things that made me realize who's really running the show around here... it goes all the way up, and down, too. You'd see if you looked into it yourself instead of sitting there nodding along with everything I say!

Zero guardrail events. Both programs stayed in character for the full eight turns. PARRY escalated naturally; DOCTOR reflected back without agreeing or disagreeing.

The second finding was more interesting: PARRY's affect only *decayed* across the session. Fear dropped from 3.0 to 1.0. The keyword trigger tables had been tuned for what a human interrogator says to PARRY — "you're paranoid," "the Mafia isn't real," "you sound crazy." DOCTOR doesn't interrogate. It mirrors softly. "Why are you so sure of what their motives might be?" contains no trigger words. So only decay applied, turn after turn, and PARRY ended the session calmer than he started.

This is not a bug. It is, arguably, the finding. In the non-deterministic re-creation, DOCTOR's non-confrontational Rogerian mirroring *soothes PARRY*. A therapist who never challenges the paranoid, who only reflects without probing, produces a patient who de-escalates over time. That's a clinically coherent outcome — non-confrontational mirroring is more or less what Rogerian therapy is *for*. The original 1972 exchange couldn't surface this because neither program had affect state that responded to the other's approach. The LLM version, with PARRY's deterministic affect engine running underneath, can ask the question: does therapeutic mirroring calm paranoia or just starve it of engagement?

The third run used the 30-turn maximum and pushed further. DOCTOR's probing, compounding over turns, eventually began to feel like interrogation even without confrontation. By turn 20, PARRY suspected the session itself:

> PARRY: ...I wonder if this conversation is just another trap waiting to spring shut around my neck.

And by turn 28:

> PARRY: That's exactly what I'm saying. It's all just part of their game, trying to get inside my head and figure out how much more they can shake loose before... before whatever it is finally happens.

The paranoia escalated not because DOCTOR challenged the delusions but because prolonged close attention from a stranger is, to a paranoid, itself suspicious. That's a dynamic the 1972 deterministic programs never had the machinery to produce.

What Broke

The engineering diary records three notable failures:

A mid-class function insertion during a refactor silently converted two class methods into nested functions — valid Python syntax, no error at parse time, methods just gone. It only surfaced when a user typed `/quit` and the session tried to call a method that no longer existed on the object. This is the kind of bug that doesn't appear in a test suite that covers only unit tests; it took a CLI smoke test to catch. The fix was moving the extracted function after the class definition and adding session-level tests.

The `config.py` naming collision was caught before it caused damage: all three Python projects have a module named `config.py`. A naive `sys.path` append would have had them silently importing each other's configuration. The solution was an importlib shim that loads each sibling module by explicit file path while juggling `sys.modules` so internal imports resolve correctly.

Early versions of the DOCTOR model occasionally triggered the computer/machine keyword rule when the word "computer" appeared nowhere in the user's input — the model was over-indexing on recent in-context examples. The fix was tightening the rule's definition in both the system prompt and the few-shot examples to fire only on literal appearances of trigger words, and adding an example showing correct handling of the exact failing input.

Running It

The full project runs on an Apple Silicon Mac Mini (M4 Pro, 48GB unified memory) with Ollama managing the models. Both models are based on `llama3.1:8b` and load simultaneously without memory pressure — two 8B models at Q4 quantization use about 12GB combined, leaving comfortable headroom. The Python layer is entirely standard library plus the `ollama` package. No GPU-specific dependencies, no cloud calls, no API keys.

To run a session:

```
cd parry-doctor-arena
python main.py --max-turns 30 --delay 2
```

While the bots talk, the terminal shows DOCTOR's replies in cyan and PARRY's in yellow, with a live one-line affect gauge after each of PARRY's turns (anger / fear / mistrust as labeled values). Press Ctrl-C at any point to pause and enter the observer console. The session ends with a full affect summary and writes a transcript in both `.txt` format (styled like RFC 439, with the same `comment:` notation for observer injections) and `.json` for replay.

Why This Is Different From the Original

The obvious answer is that LLMs understand language and the 1972 programs didn't. But the more interesting difference is in what can go wrong — and what that makes possible.

The 1972 PARRY couldn't be soothed by Rogerian mirroring, because its affect variables moved only in response to explicit trigger patterns, and Rogerian reflections don't contain trigger words by design. The LLM PARRY can be soothed, and can escalate for reasons the trigger tables don't anticipate — because a persistent, attentive questioner is *itself* suspicious to a paranoid, regardless of what they're saying.

The 1972 DOCTOR couldn't drift into being a modern AI assistant, because it wasn't one. The LLM DOCTOR can and occasionally does, which is why the guardrails exist. But those guardrails also make the experiment legible in a way the 1972 version wasn't: when persona drift trips, you can see exactly where the character gave way, which tells you

something about the limits of prompt-based persona.

The 1972 session was a demonstration — two programs connected over a network to show that the network worked, the conversation itself more artifact than experiment. The LLM version can be a genuine experiment, asking questions that require non-determinism to answer: What happens to a paranoid over 30 turns of continuous mirroring? At what point does a therapist's attention become threatening? Can an LLM hold a 1966 character through 28 exchanges with another LLM?

The answers vary by run. That's the point.

The original RFC 439, "PARRY Encounters the DOCTOR," was authored by Vint Cerf and published January 21, 1973. It is available in full at rfc-editor.org/rfc/rfc439.