

The many roles of AI in Software.

March 2026 edition

reliableai.github.io

Fabio Casati · University of Trento

The plan for today

01 — Set up L1 notebooks

Make sure we can run L1 notebooks (no audio or image for now please)



02 — Roles of AI in software

Understand the roles of AI in software – think before you implement



03 — Making good individual calls

Understand how to make “good” individual calls to AI



04 — Stateless vs stateful interactions

Understand nuances of stateless vs stateful interactions and context management



Three Roles for AI

01

AI as a Component

AI is embedded inside your solution — a model call in the pipeline. You design around its strengths and limitations: prompts, guardrails, evaluation, fallback logic.

02

AI as a Coding Agent

AI helps you write, review, and refactor code. It accelerates development but the output is traditional software — deterministic, testable, fully under your control.

03

AI as a User

AI consumes and produces information through your system. It calls your APIs, reads your docs, and acts on your data. You need to design for this new kind of consumer.

When Do You Need AI as a Component in your System?

No AI

Fully specifiable

- Behavior is specified, reviewed, and approved
- Outputs are deterministic and testable
- You want guarantees. Adding AI adds complexity without value
- *eg: Billing logic, form validation, API routing, access control*

AI as a Convenience

Easier in English.

A NEW CATEGORY!!

- The behavior can be fully written down as rules
- But a prompt is simpler than writing code
- English becomes the implementation language
- Outputs are known — you can still test correctness

AI as a System Component

Hard to specify.

- Behavior is too complex or contextual for rules
- Hard to write conventional tests — no expected output
- The value comes precisely from what's hard to specify
- Eg: summarization, intent, creative generation, open-ended Q&A

AI as a System Component

Think before you build:

1. What are you building – how “reviewable” and fixed you want your output
2. Who are you building for

If the answer to #2 is “humans”, think harder

Four Challenges/Opportunities of Working with LLMs

01

Getting a single call right — How do I craft a prompt that reliably produces the output I need?

02

Managing a conversation — How do I give the model the right context across turns without blowing up cost?

03

Long-term memory — How do I remember a user across sessions — their facts, preferences, history?

04

Enterprise / community knowledge — How do I connect knowledge across users while respecting privacy?

Later lessons (if time allows)