Context Engineering in AI-Powered KiCad Automation

Ethan Chien

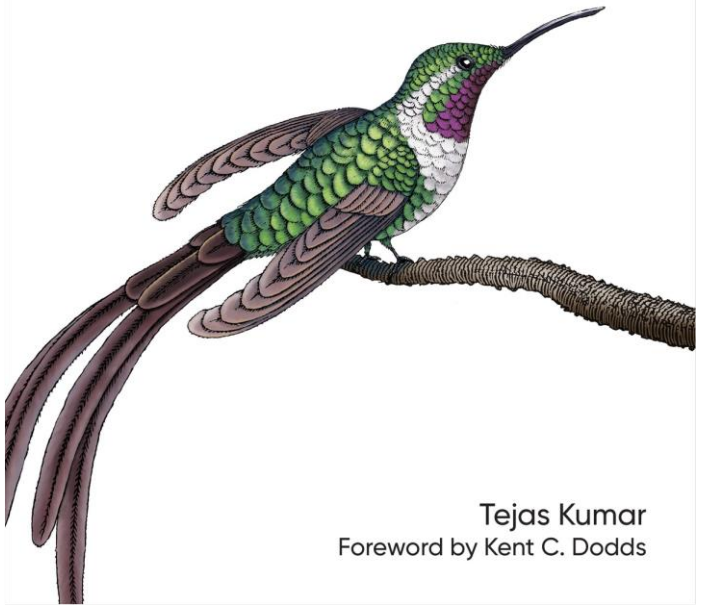
Intelligent Manufacturing Center, Huaqiu

华秋
www.huaqiu.com

O'REILLY®

Fluent React

Build Fast, Performant, and Intuitive Web Applications



Tejas Kumar

Foreword by Kent C. Dodds

O'REILLY®

AI Engineering

Building Applications with Foundation Models

Chip Huyen

Services for a Changing World

RESTful Web APIs

Leonard Richardson & Mike Amundsen

Foreword by Sam Ruby

O'REILLY®

Talk outline

AI-Powered KiCad Automation: How to

- Context-aware Chatbot
- Combine GenAI with Chatbot
- Shaping the business with LangGraph

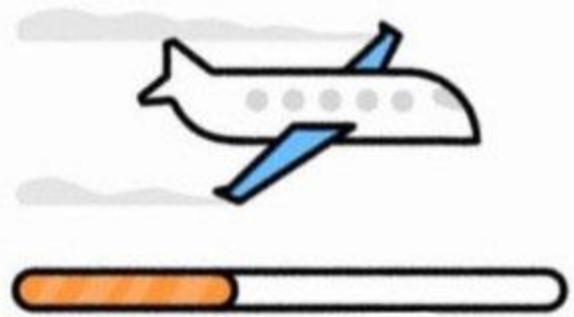
Context Engineering in Action

- MCP explained
- The 100-LOC kicad-mcp-server Insight

Lesson Learned

地球上最快的東西

豹



飛機

光

alt+tab

ALT TAB

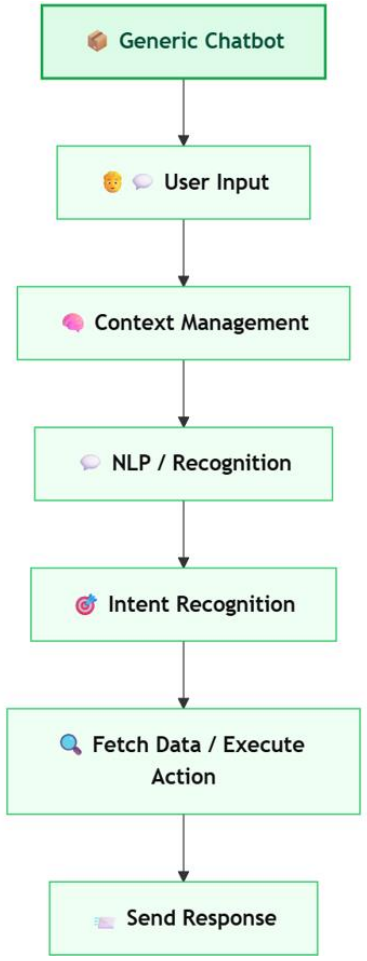
\+

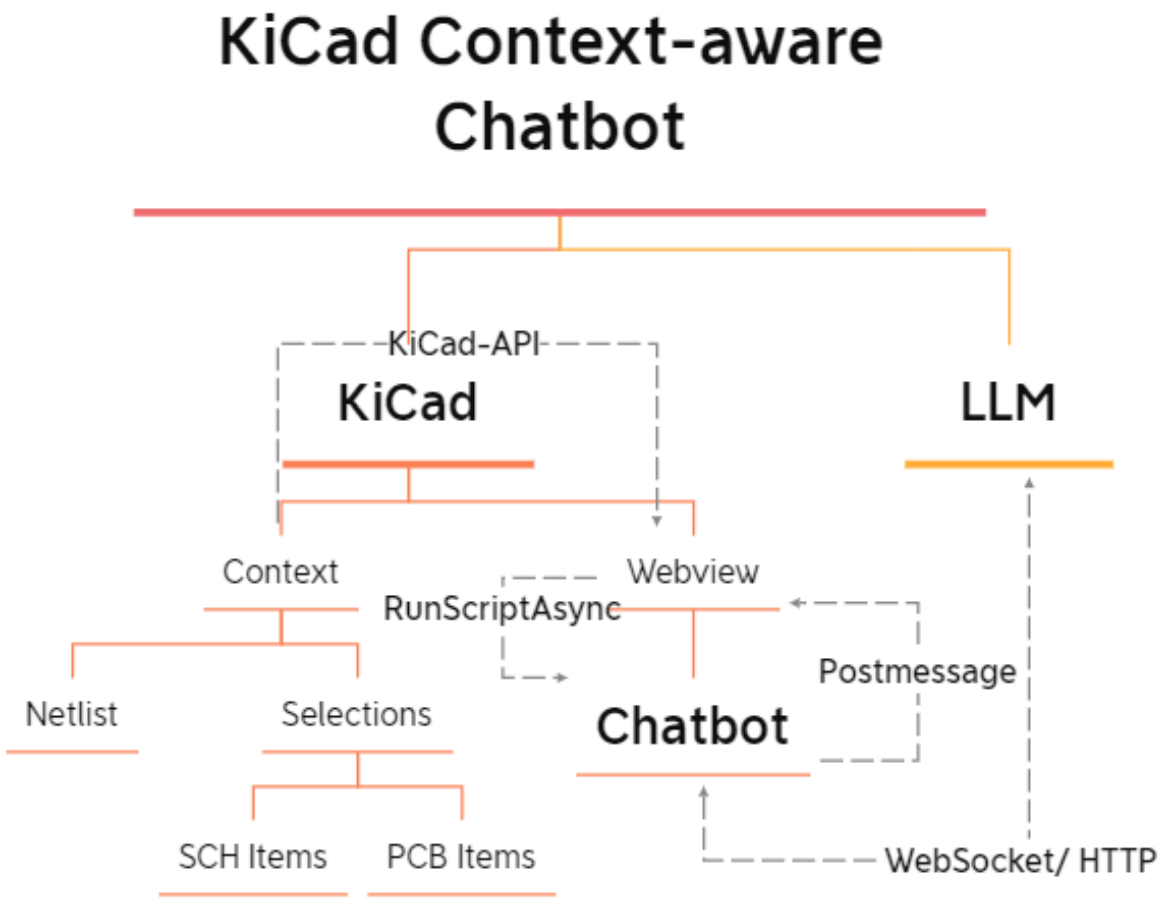
CTRL C/V

=

Claude

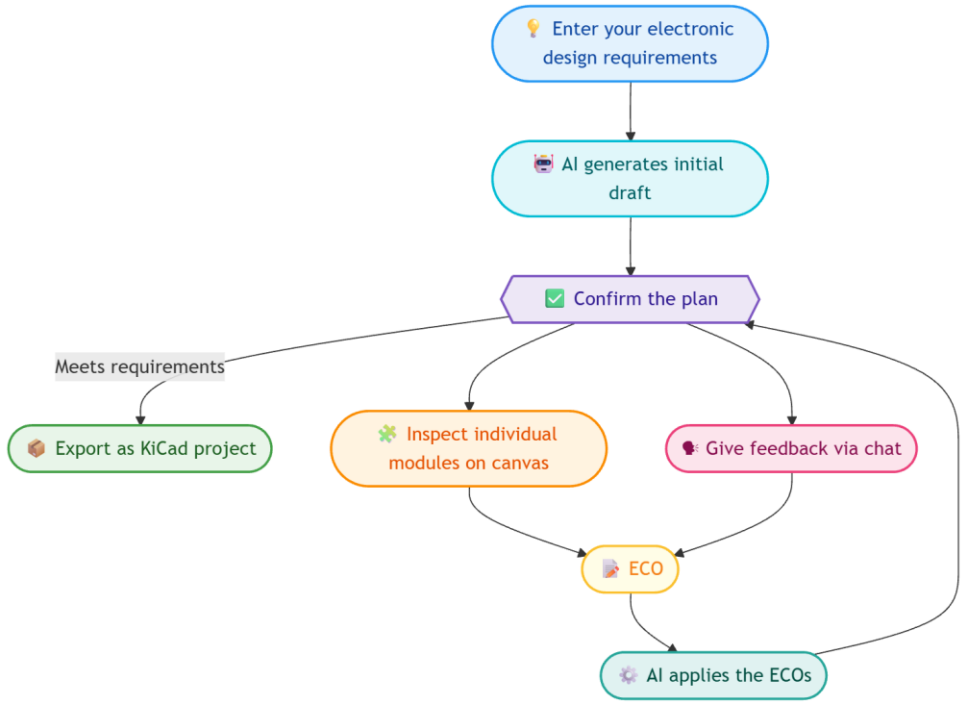
KiCad Context-aware Chatbot

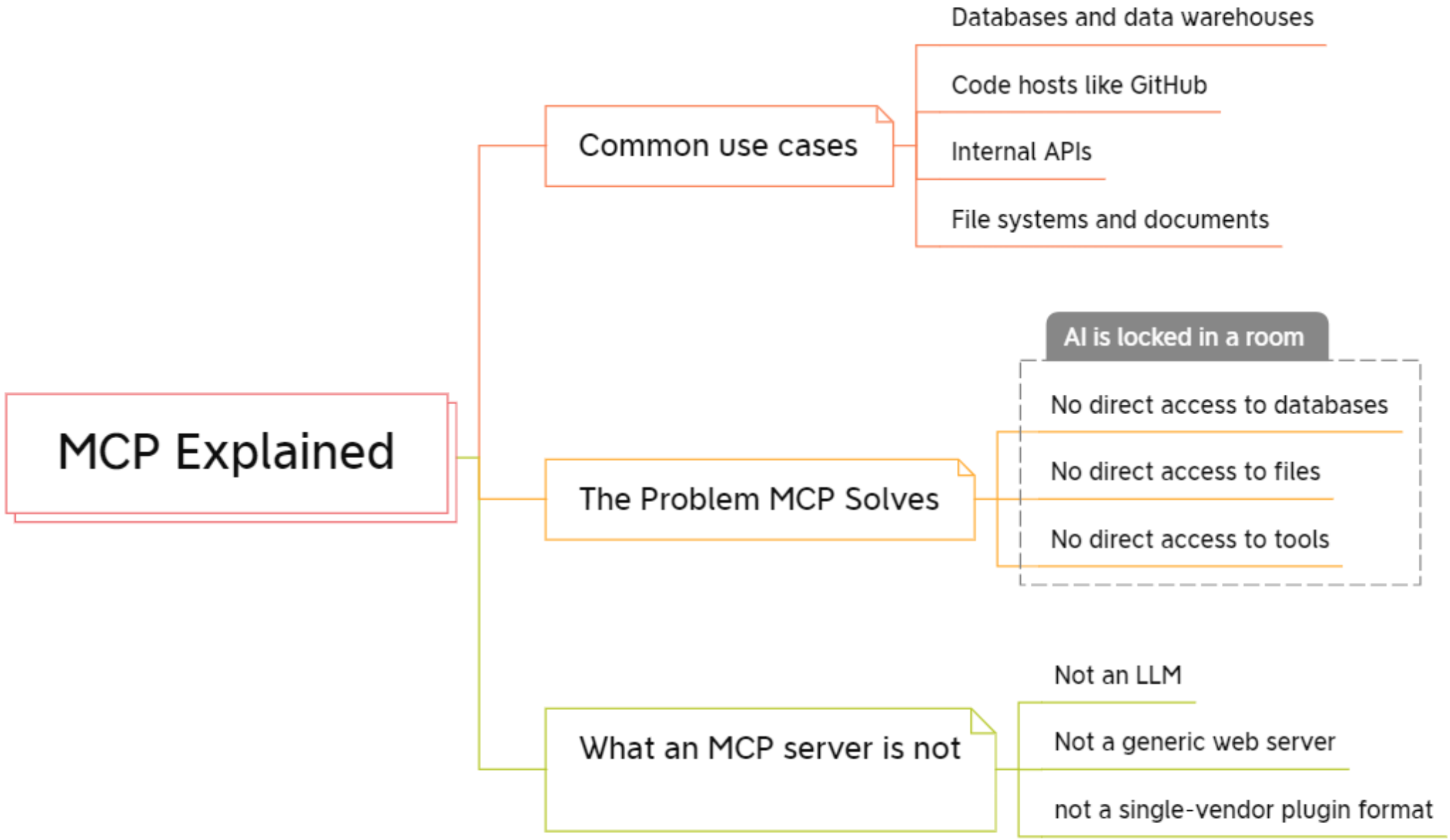




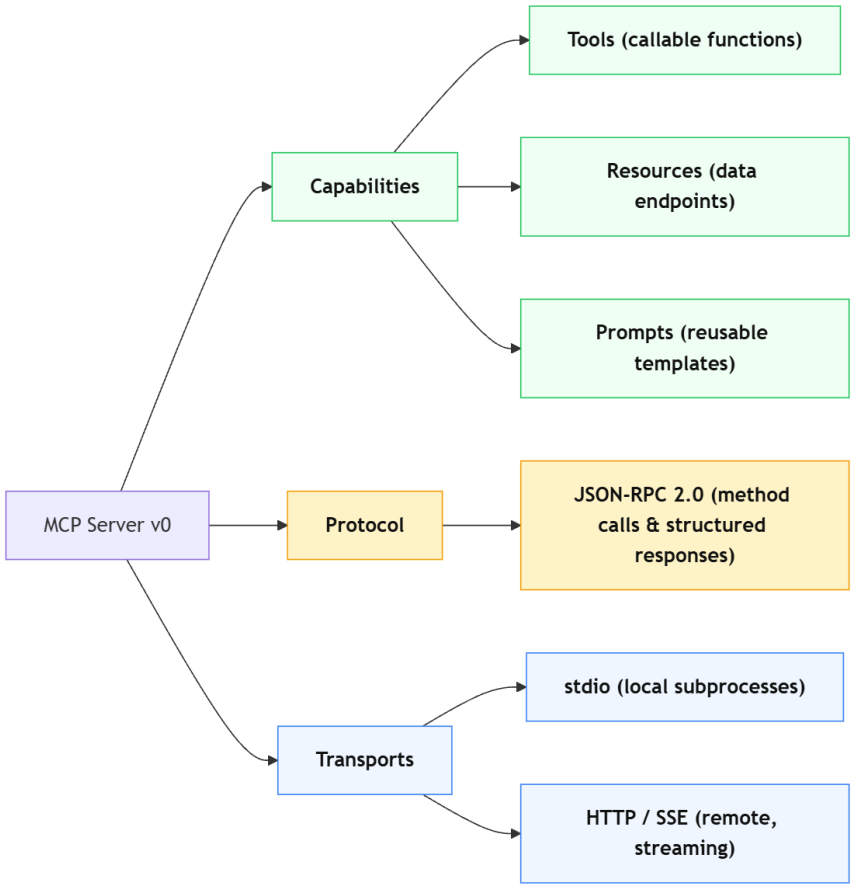
Intention to Schematic: Tech Stack & Workflow

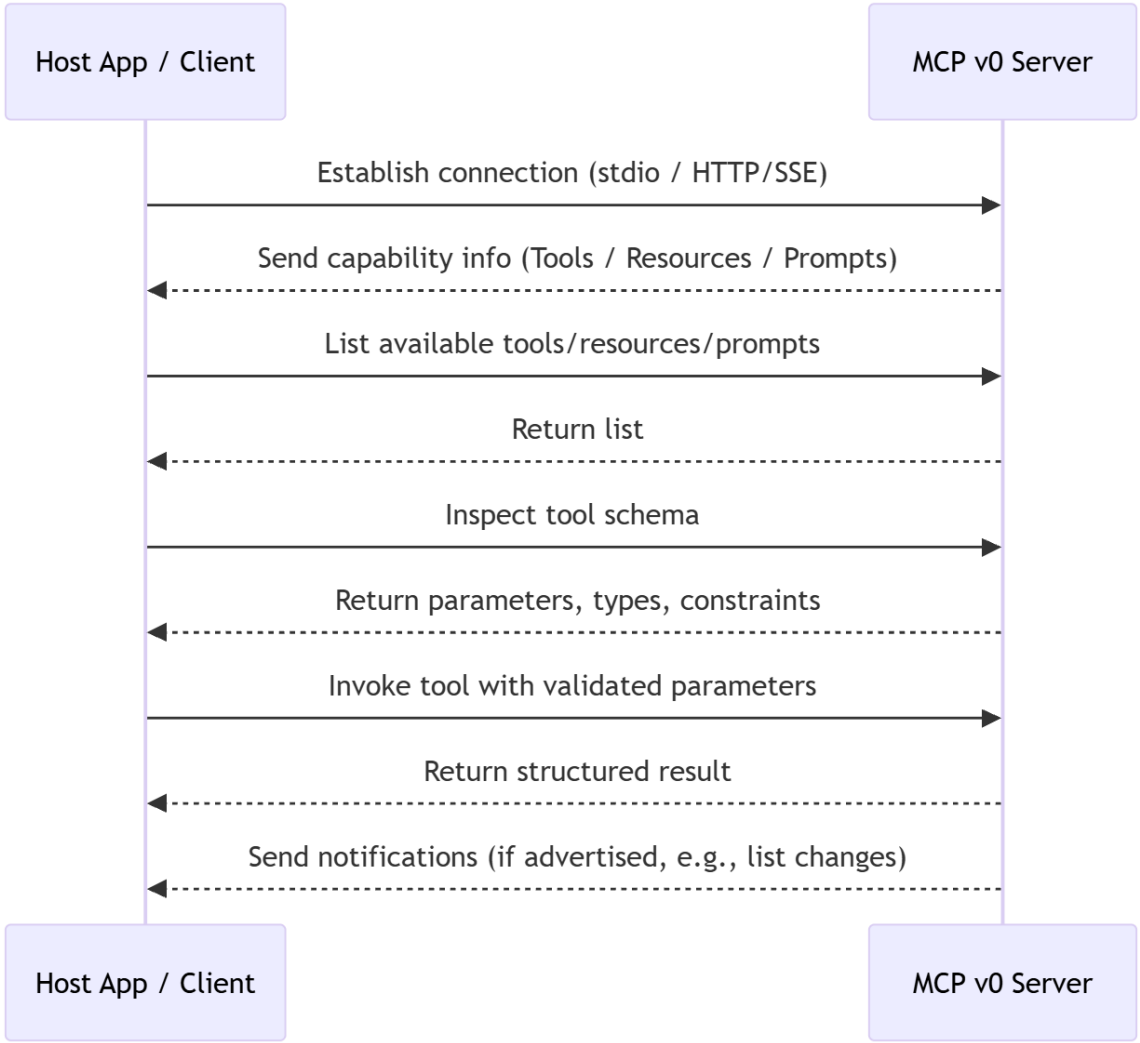
Intention to Schematic Tech Stack	
Frontend	💡 Next.js: input, canvas, chat
Backend	⚙️ FastAPI (Python backend): handle requests & workflows
AI / Design Logic	🤖 LangGraph: draft generation & apply ECOs
AI Assistant UI	🗣️ CopilotKit: React hooks, integrates with LangGraph
Design Export	📦 Node.js + KiCad generator: export finalized designs

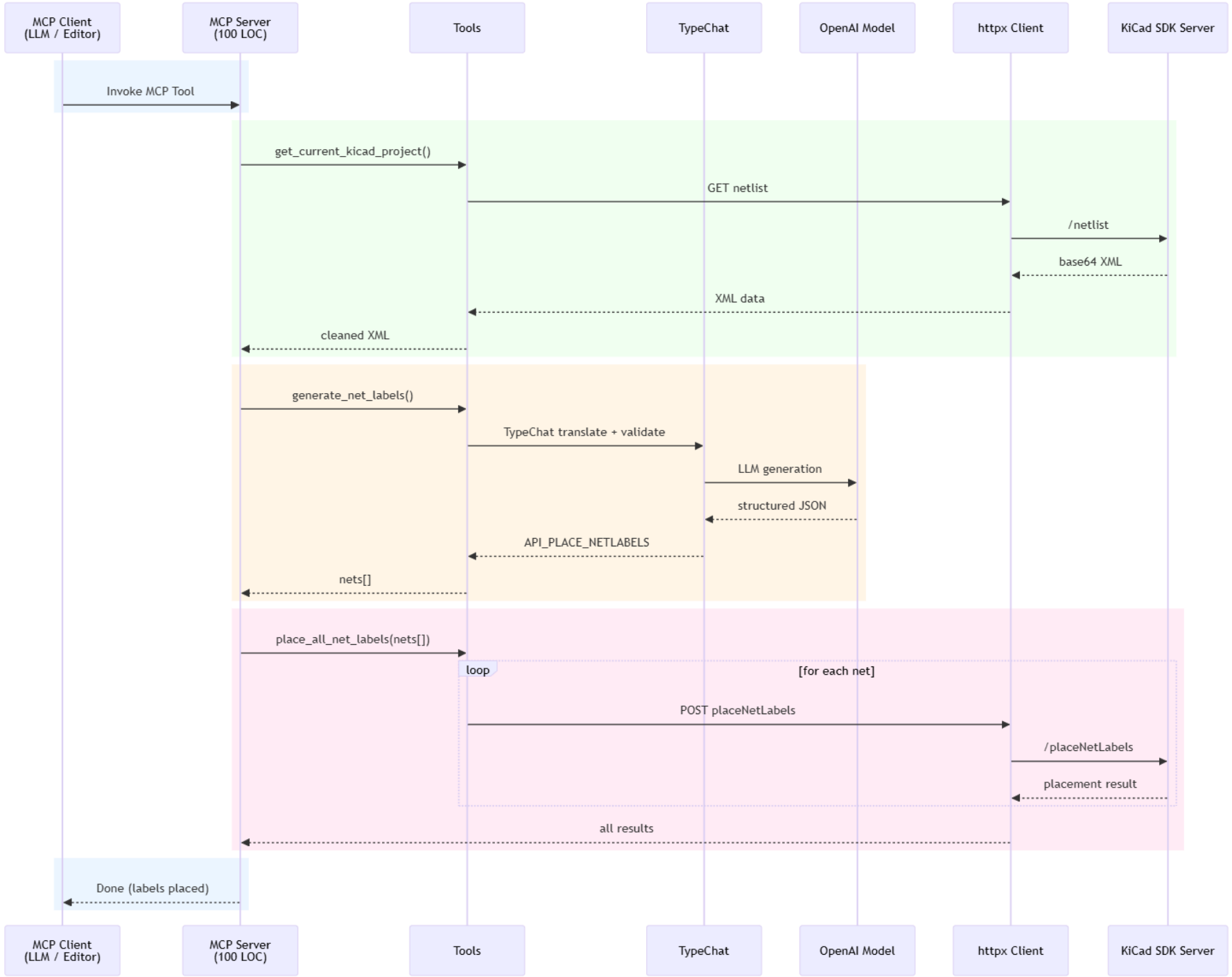


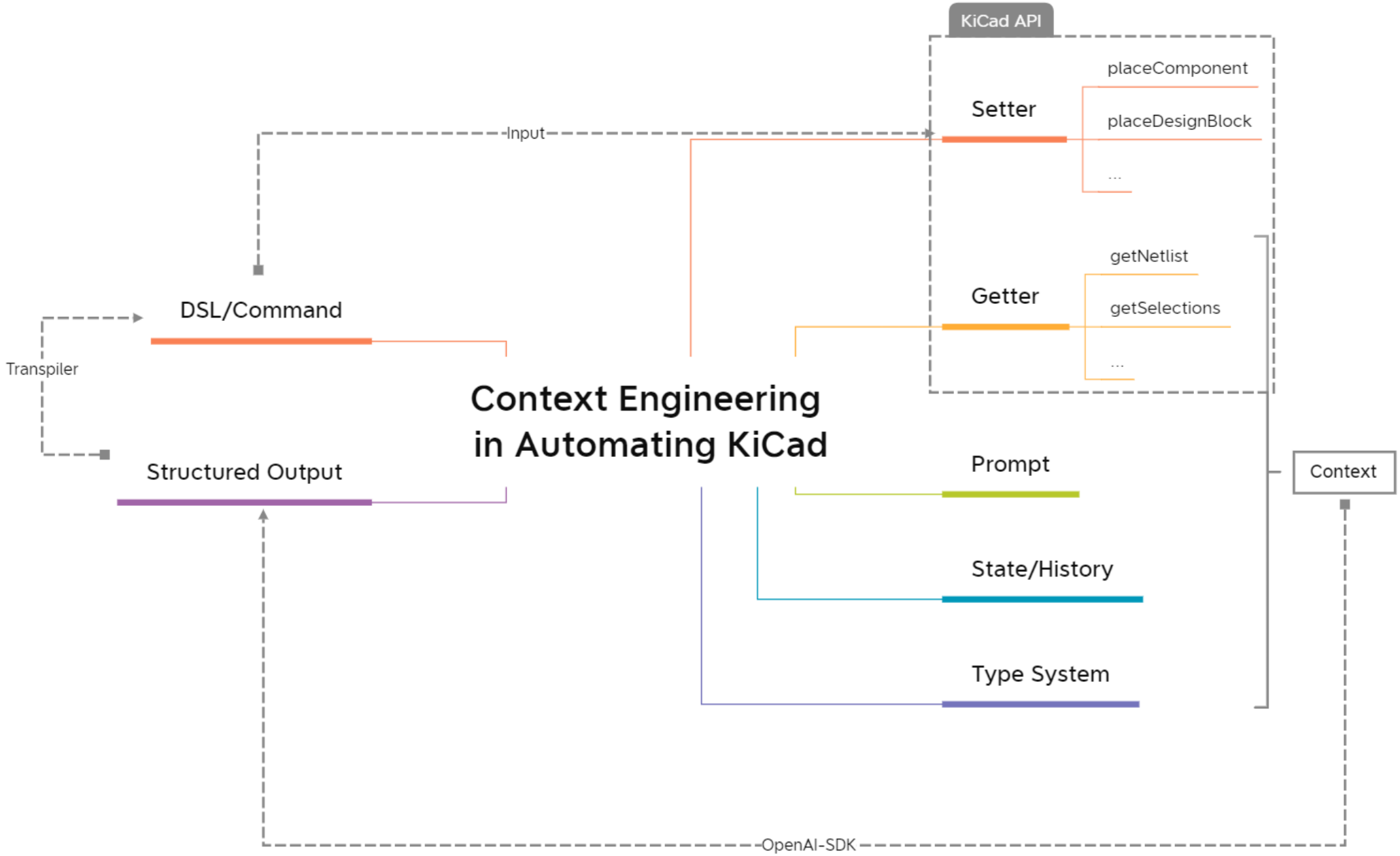

MCP Explained
Common use cases
Databases and data warehouses
Code hosts like GitHub
Internal APIs
File systems and documents
The Problem MCP Solves
AI is locked in a room
No direct access to databases
No direct access to files
No direct access to tools
What an MCP server is not
Not an LLM
Not a generic web server
not a single-vendor plugin format

How an MCP server works





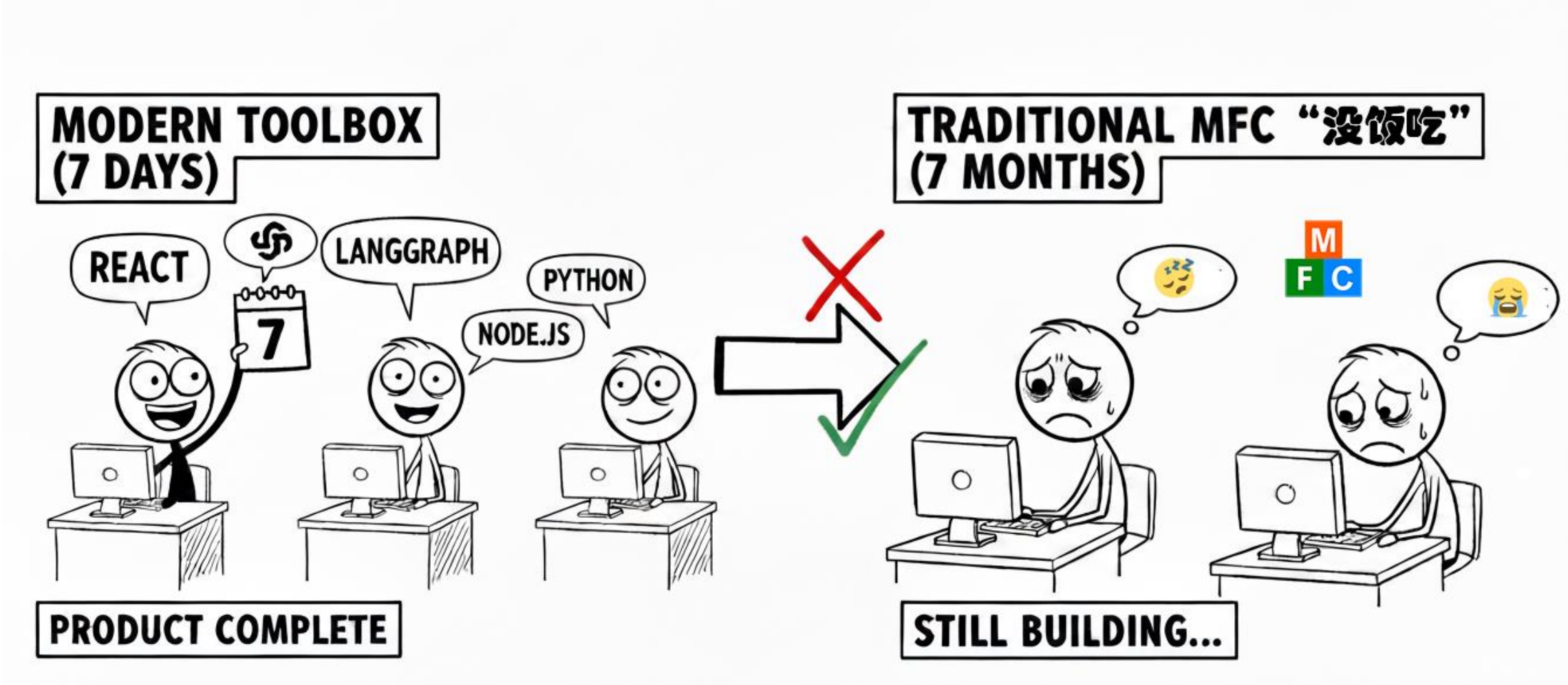

MCP Client (LLM / Editor)
MCP Server (100 LOC)
Tools
TypeChat
OpenAI Model
httpx Client
KiCad SDK Server
Invoke MCP Tool
get_current_kicad_project()
GET netlist
/netlist
base64 XML
XML data
cleaned XML
generate_net_labels()
TypeChat translate + validate
LLM generation
structured JSON
API_PLACE_NETLABELS
nets[]
place_all_net_labels(nets[])
loop
[for each net]
POST placeNetLabels
/placeNetLabels
placement result
all results
Done (labels placed)
MCP Client (LLM / Editor)
MCP Server (100 LOC)
Tools
TypeChat
OpenAI Model
httpx Client
KiCad SDK Server


KiCad API
placeComponent
Setter
placeDesignBlock
...
Input
getNetlist
Getter
getSelections
...
DSL/Command
Transpiler
Context Engineering
in Automating KiCad
Prompt
Context
Structured Output
State/History
Type System
OpenAI-SDK

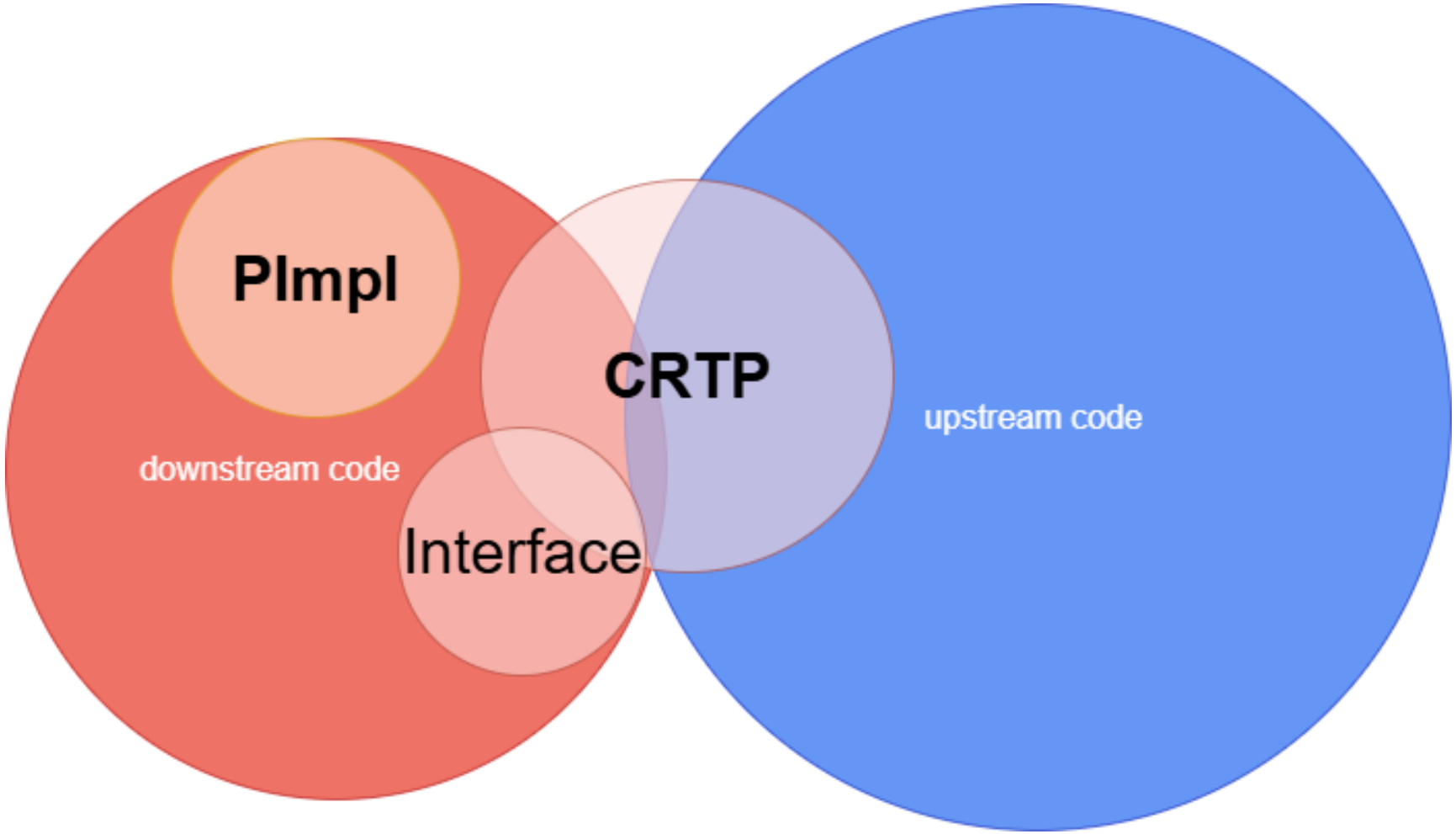
Lesson Learned: Static Type Checking for GenAI

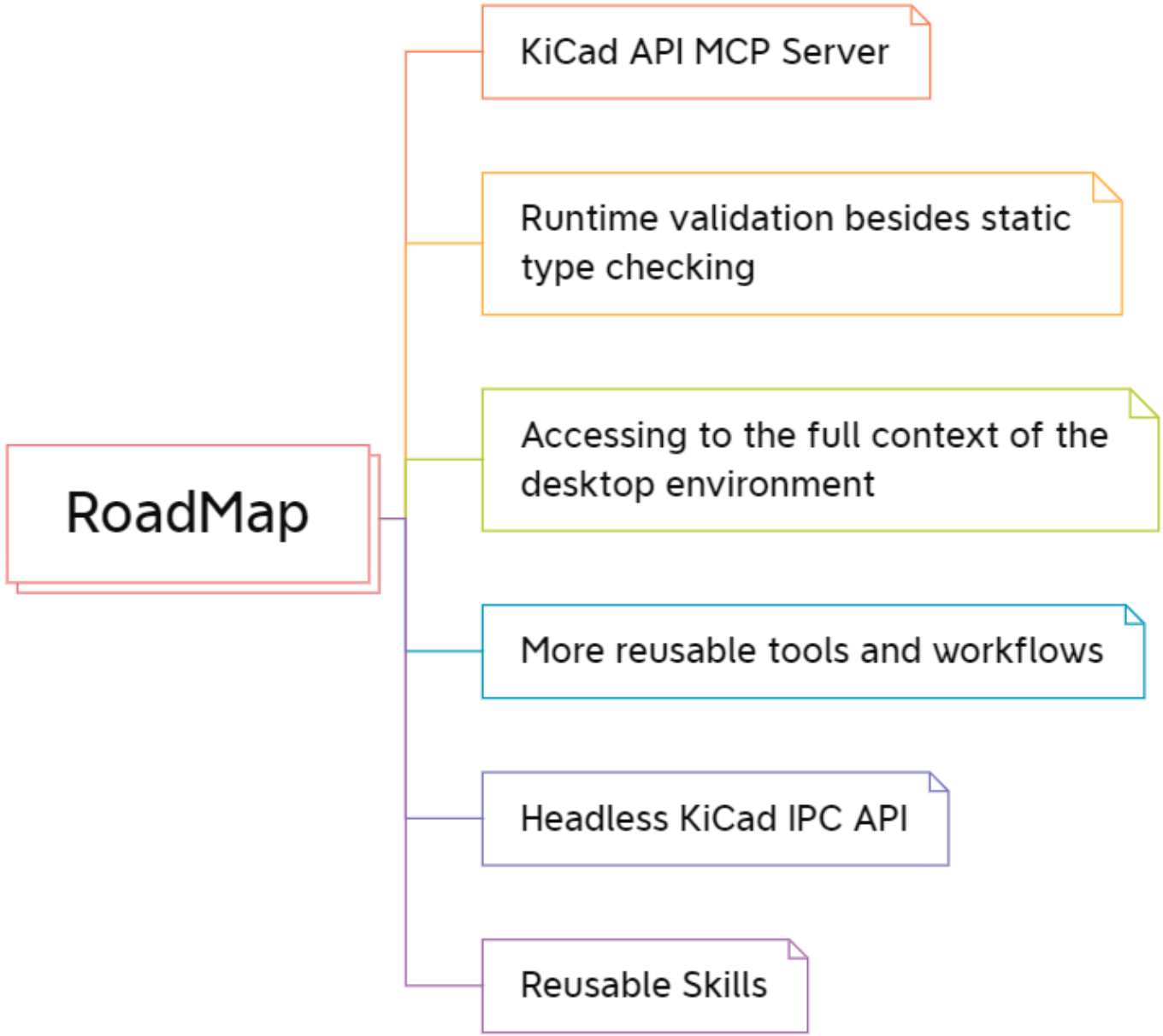
Feature / Tool	JSON Schema	Pydantic	TypeChat
Language support	Multi-lang	Python	TypeScript
LLM integration maturity	High (standardized)	Medium	High (LLM-native)
Runtime repair capability	✗ No	✗ No	✓ Yes (via LLM)
Custom validation logic	⚠ Limited	✓ Rich	✓ via TS types
Ease of definition	⚠ Verbose	✓ Clean	✓ TS-native
Standardization level	✓ RFC spec	✗ No std	✗ No std
Portability	✓ Excellent	✗ Poor	✗ Poor
Ecosystem maturity	✓ Mature	✓ Mature	⚠ Early stage
Best use case	Cross-language schema enforcement	Python-based pipelines	TypeScript LLM apps needing auto-fix

Lesson Learned: Frameworks as Toolbox



Lesson Learned: #include "business.cpp"




RoadMap
KiCad API MCP Server
Runtime validation besides static type checking
Accessing to the full context of the desktop environment
More reusable tools and workflows
Headless KiCad IPC API
Reusable Skills

Questions?

Context Engineering in AI-Powered KiCad Automation

THANK YOU

Ethan Chien

Intelligent Manufacturing Center, Huaqiu

华秋
www.huaqiu.com