

beyond vibe coding: a call for a spec language



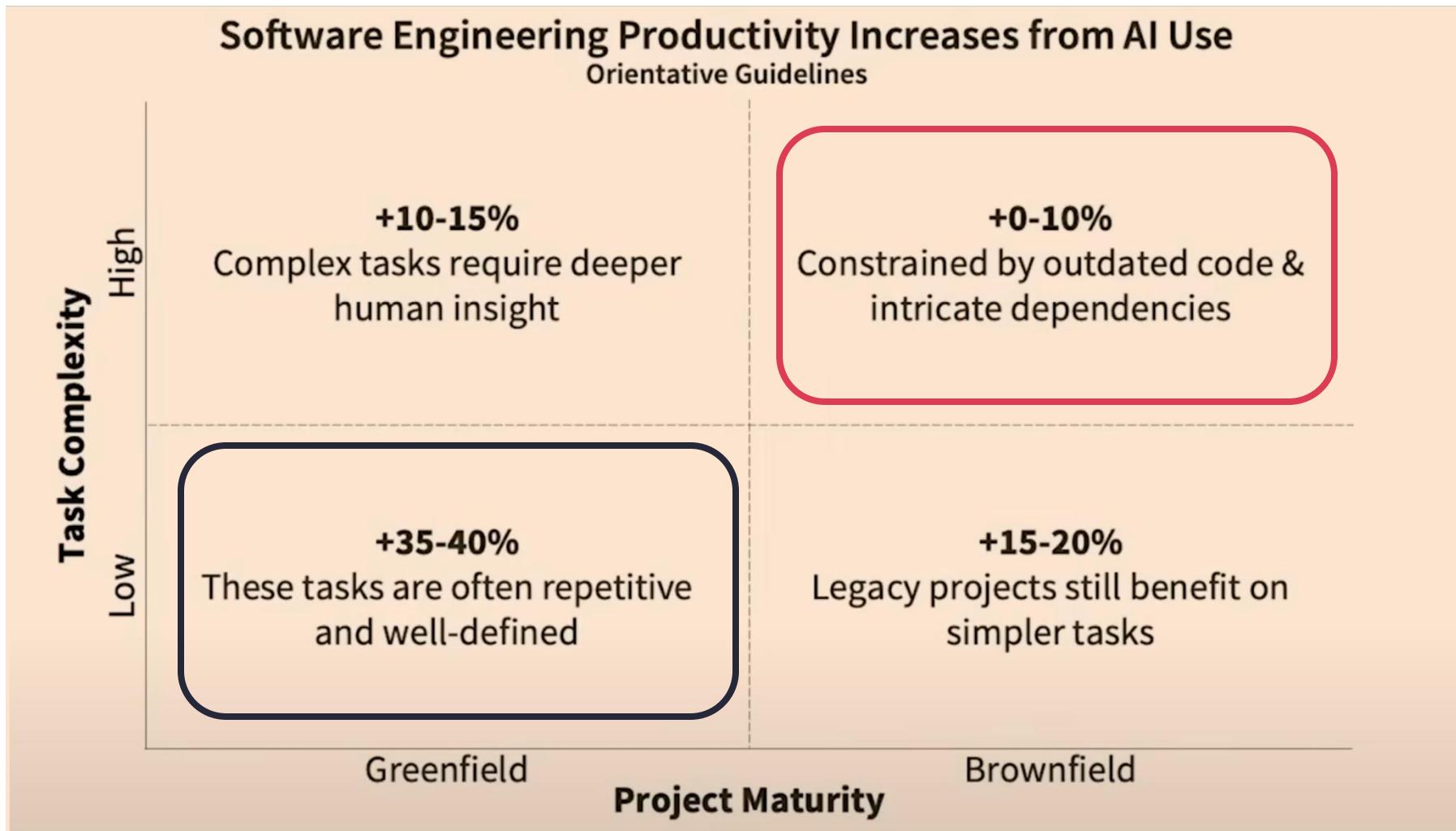
*Markus Höfling
Business Engineering @ DevBoost*



vibe coding:

“fix one thing, break ten others.”

DUSAN OMERIC (CODEPLAIN, “BEYOND VIBE CODING”
2025)



Stanford Study on AI's Impact on Developer Productivity (<https://www.youtube.com/watch?v=tbDDYKRFjhk>)



{ WHY KIRO }



Go from vibe coding to
viable code

kiro.dev by AWS



GitHub Next

GitHub Next investigates the future of software development.

We are a team of researchers and engineers at GitHub, exploring things beyond the adjacent possible. We prototype tools and technologies. We identify new approaches to building teams and products.

GitHub Copilot

June 29, 2021

PRODUCT

A new way to build software

SpecLang

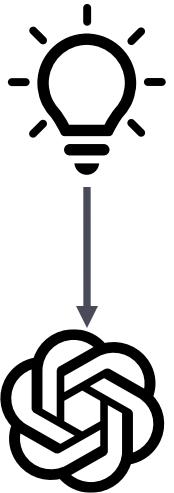
November 9, 2023

RESEARCH PROTOTYPE

Can we develop software entirely in natural language, and an AI-powered toolchain manage the implementation?



Feature / Task Idea



critical step!
-> may require multiple iterations
due to ambiguity of natural language!

**Q&A with LLM to
clarify the WHY and WHAT**

Generate Requirements

```
# Requirements Document
## Introduction
This feature enables administrators and users to customize the logo displayed on the LibreChat login page. The current login page displays a default logo, and this feature will provide the ability to replace it with a custom logo while maintaining proper styling, responsiveness, and accessibility standards.

## Requirements
### Requirement 1

**User Story:** As an administrator, I want to replace the default login page logo with a custom logo, so that I can brand the application according to my organization's identity.

#### Acceptance Criteria
1. WHEN an administrator uploads a custom logo THEN the system SHALL display the new logo on the login page
2. WHEN no custom logo is configured THEN the system SHALL display the default LibreChat logo
3. WHEN a custom logo is uploaded THEN the system SHALL validate the
```

Generated Code

```
def fetch(limit=100):
    headers = {"Authorization": f"Bearer {API_KEY}"}
    cursor = None
    users = []

    while True:
        params = {"limit": limit}
        if cursor:
            params["cursor"] = cursor

        r = requests.get(f"{BASE_URL}/v1/users",
                        headers=headers, params=params)
        r.raise_for_status()

        payload = r.json()
        users.extend(payload.get("data", []))
        cursor = payload.get("next_cursor")
        if not cursor:
            break

    return user
```





SpecLang

Can we develop software entirely in natural language, and let an AI-powered toolchain manage the implementation?

WHAT'S IT FOR?

End-to-end development in natural language

STAGE

RESEARCH
PROTOTYPE

The image shows a split-screen view. On the left, a Mac OS X terminal window titled 'specnews.spec' displays a natural-language specification for a 'SpecNews' application. The code uses a mix of English comments and structured definitions for components like 'Header', 'PostList', and 'PostItem'. On the right, an iPhone SE running iOS 16.4 displays the 'SpecNews' mobile application, which shows a list of trending posts from HackerNews.

```
specnews > specnews.spec > # SpecNews
1  # SpecNews
2
3 This app is a HackerNews client that shows a list of trending posts.
4
5 ## Screens
6
7 ### Home Page
8 Displays a list of trending posts from HackerNews.
9
10 - **Behavior:** Fetches and updates the posts from the HackerNews API.
11 - **Layout:** Contains a `Header` and a `PostList`.
12
13 ## Components
14
15 ### Header
16 Displays the app title and a refresh button.
17
18 - **Behavior:** On tap, triggers the `PostList` to refresh the posts.
19 - **Styling:** Fixed at the top, with a large font size and a background color.
20
21 ### PostList
22 Displays a list of posts in a scrollable format.
23
24 - **Behavior:** 
25   - Renders a list of `PostItem` components.
26   - Supports pull-to-refresh and infinite scrolling.
27 - **Styling:** Occupies the remaining space below the `Header`.
28
29 ### PostItem
30 Displays an individual post with its title, score, and author.
31
32 - **Behavior:** On tap, opens the post URL in a web browser.
33 - **Styling:** 
34   - Displays the title in a bold font and the score and author in a smaller font.
35   - Has some padding and a separator line.
```

SpecLang

Videos of Godotcon 2023
136 points by mdtrooper

Maine's Constitution Has Unprintable Sections
88 points by dsr_

A C Compiler that fits in the 512 byte boot sector of an x86 machine
91 points by doener

Whole Earth Catalog (1968) [pdf]
46 points by doener

Potato Diet Riff Trial: Sign Up Now, It's Free!
80 points by MaurizioPz

Validating Data in Elixir: Using Ecto NimbleOptions
54 points by amalinovic

The coolest robot I've ever built
232 points by superasn

“spec-to-code compilation”

<https://githubnext.com/projects/speclang/>

*** plain

***plain — a specification language that combines the efficiency of natural language with the control and precision of code.

“To move **beyond vibe coding** [...], developers must be able to **fix any issue entirely within the specifications** — **without ever touching the underlying code.**”

*currently under development by members of **Codeplain**,
SpecLang & former member of original GitHub Copilot Team

** first release is planned for fall 2025

*codeplain



**spec-driven,
production-ready
code generation**

*codeplain is an ai-powered code generation service that produces production-ready software from specifications written in ***plain language.

JOIN WAITLIST

<https://www.codeplain.ai/>



*** plain

Specification in ***plain

Generated Python code

intercom-client/plain

intercom-client.py

```
{% include "python-console-app-template/plain" %}
```

Definitions:

- The App interacts with Intercom API (The API).
- The List of Users is the list of Intercom API users.

Non-Functional Requirements:

- The resource [intercom-openapi.json](<https://github.com/Codeplain-ai/intercom-openapi.json>) describes The API.

Test Requirements:

- The Conformance Tests should use the real server of The API - do not use mock server.
- The resource [intercom-credentials.json](./credentials/intercom-credentials.json) contains credentials that can be used for testing.

Functional Requirements:

- The App should take credentials json as the only positional argument.
- Authenticate with The API.
- Fetch The List of Users from The API.
- Print The List of Users to the console.

*** plain

Specification in ***plain

Generated App

task_manager/plain

```
{% include "typescript-react-app-template/plain" %}
```

Definitions:

- The User is the user of The App.
 - The Task describes an activity that needs to be done by The User.
- The Task has the following attributes
- **Name** - (required) - a short description of The Task. The name must be at least 3 characters long.
 - **Notes** - additional details about The Task.

Test Requirements:

- End-to-end tests of The App should be implemented in React using Jest.

Functional Requirements:

- Show The Task List. The details of the user interface are provided in [task_list_ui_specification](<https://github.com/codeplain/task-manager.png>).
- The User should be able to add The Task. The details of the user interface are provided [new_task_modal_specification] (<https://github.com/codeplain/new-task.png>).
- The User should be able to delete The Task.
- The User should be able to edit The Task.
- The User should be able to mark The Task as completed.

Task List App ⚙

Task list

All of your tasks in one place.

Add New Task

Complete project proposal

Finish the quarterly project proposal for the new client initiative
15. Jun



Review team feedback

Go through all team member feedback from the last sprint
14. Jun



Prepare presentation

Create slides for the upcoming board meeting presentation
16. Jun



Dimension	Vibe coding	Spec-driven with ***plain
Time to first run	Very fast	Fast (testing adds time)
Source of truth	Codebase	The specification
Tests	Requires explicit prompt	Automated test generation and execution
Change safety	Manual regressions	Auto-tested
Collaboration	Review code after generation	Review the spec before generation
Maintainability	Declines as complexity grows	Stays maintainable
Barrier to entry	Lower (freeform text)	Higher (learn ***plain primitives)

<https://blog.codeplain.ai/p/from-vibes-to-specs>