

# AI for software development: A reality check

Dec 4, 2023

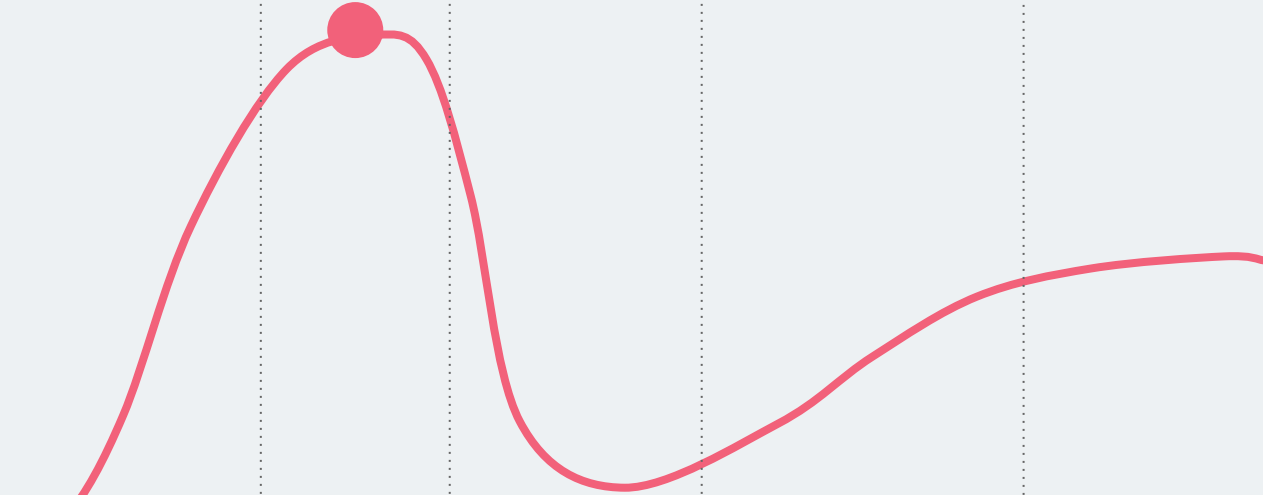
Birgitta Böckeler

@bboeckel@toot.thoughtworks.com 

# Generative AI

## AI-Augmented Software Engineering

Expectations



Innovation Trigger

Peak of  
inflated  
expectations

Trough of  
disillusionment

Slope of  
Enlightenment

Plateau of  
productivity



**Are you super excited  
about this?**

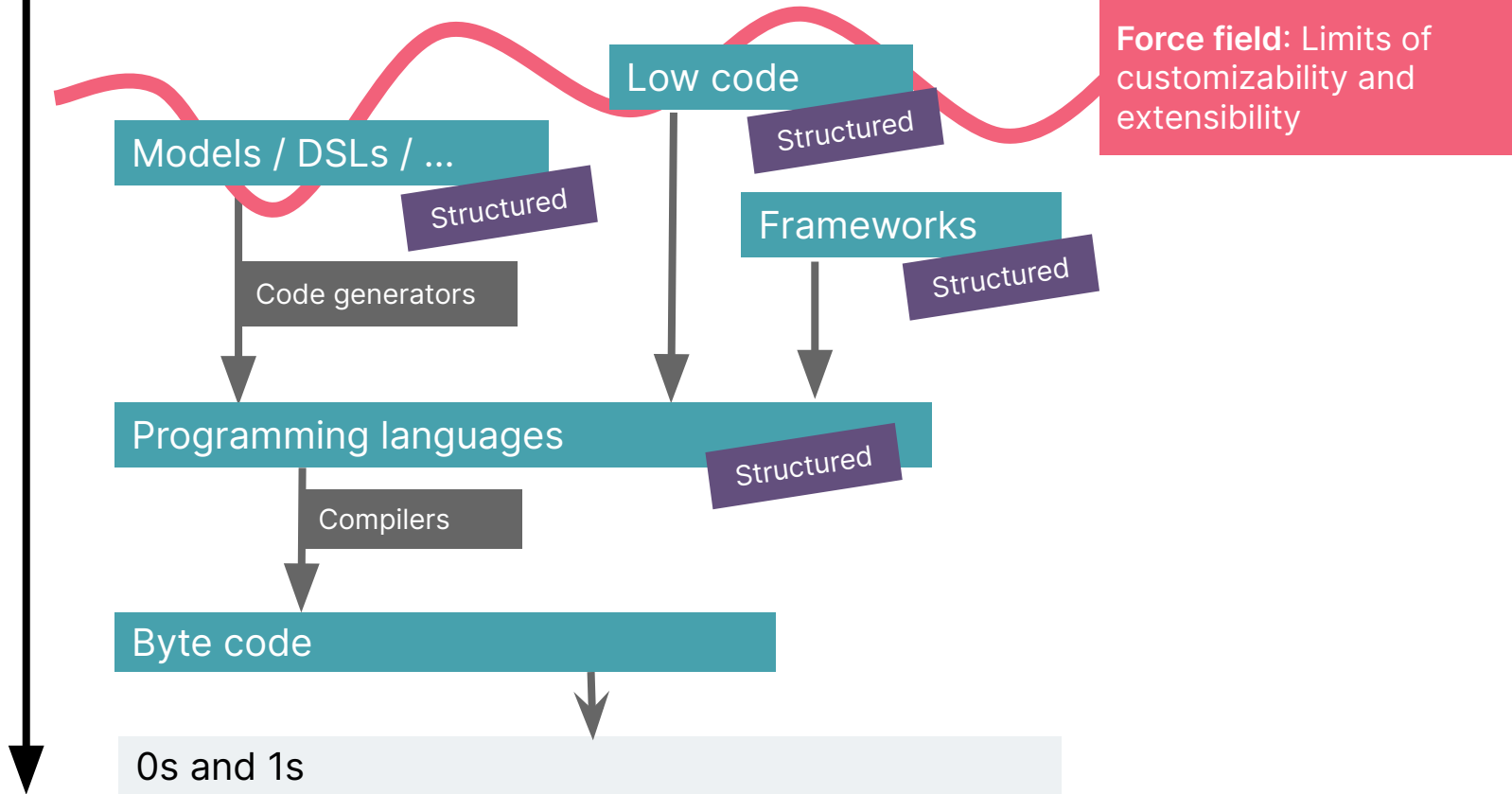
**Are you skeptical?**



# What's new here?



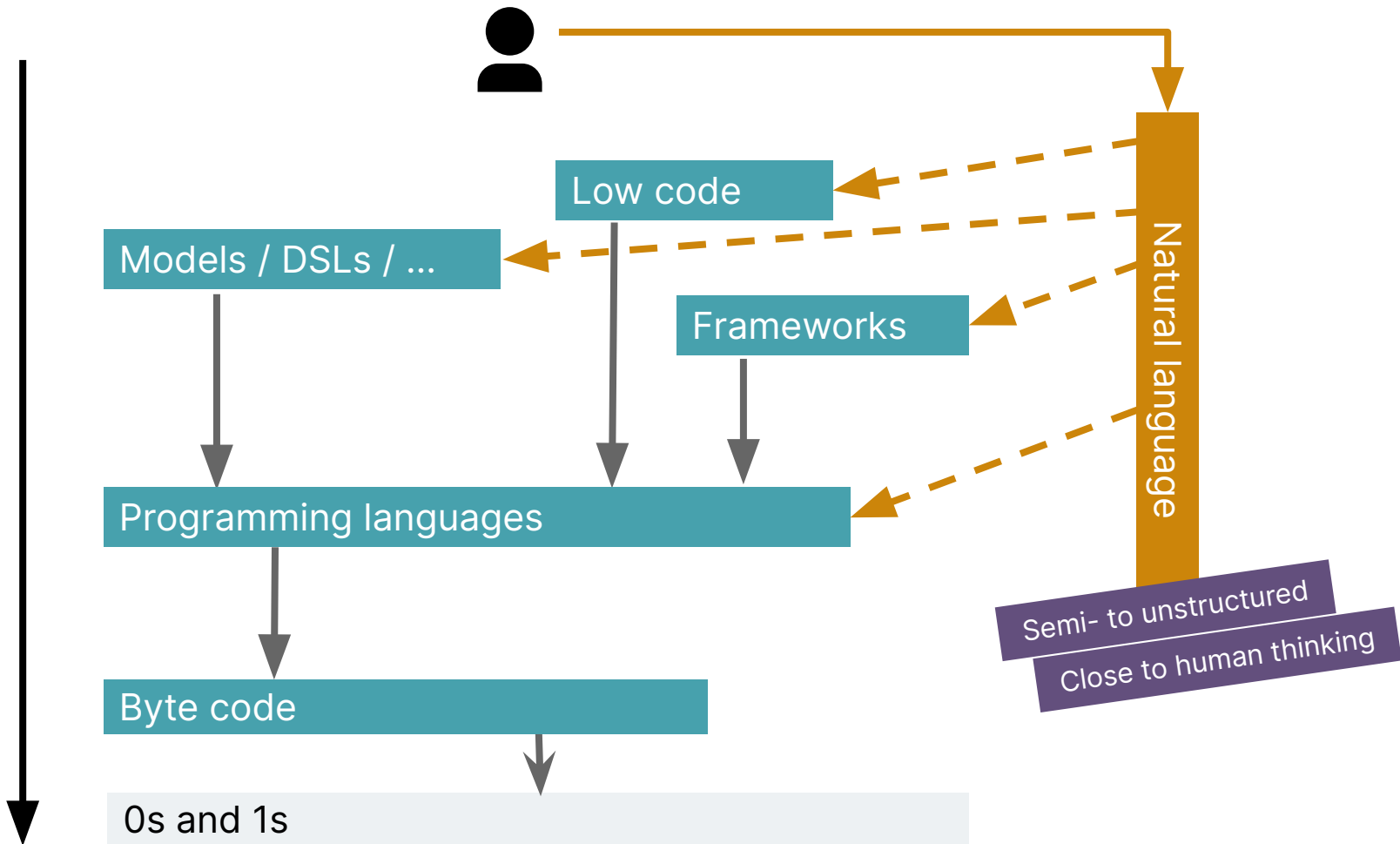
High level of abstraction



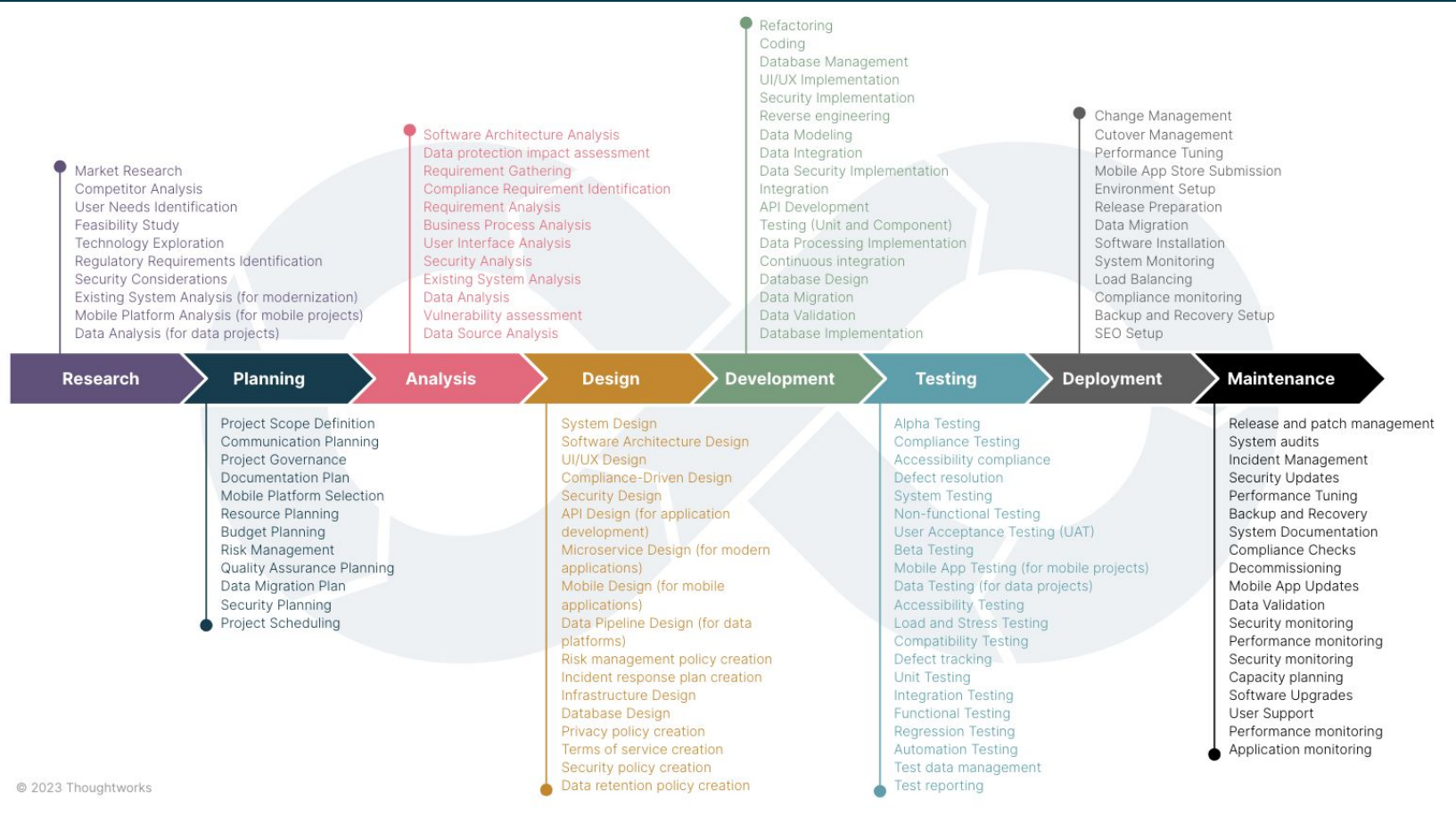
Force field: Limits of customizability and extensibility

High level of technical detail

High level of abstraction



High level of technical detail



**Coding**





# Coding assistance: The core features

## In-line assistance

```
public getReactions(): Reaction[] {  
    return _.chain(this.list)  
        .map(m => {  
            return m.reactions;  
        })  
        .flatten()  
        .value();  
}
```

## Chat assistance

typescript, with a list of messages, and each message has an id, a user id, a text, and a list of emojis that were used as a reaction and how many times. With that list of messages, I want to do a bunch of calculations:

1. How many users posted messages?
2. How many messages did each user post?
3. Which emoji was used the most as a reaction?

Please write some Typescript code for me that represents this model of messages, and has functions to calculate those 3 calculations.



Sure, here's some Typescript code that represents the model of chat messages you described, along with functions to calculate the three requested calculations:



typescript

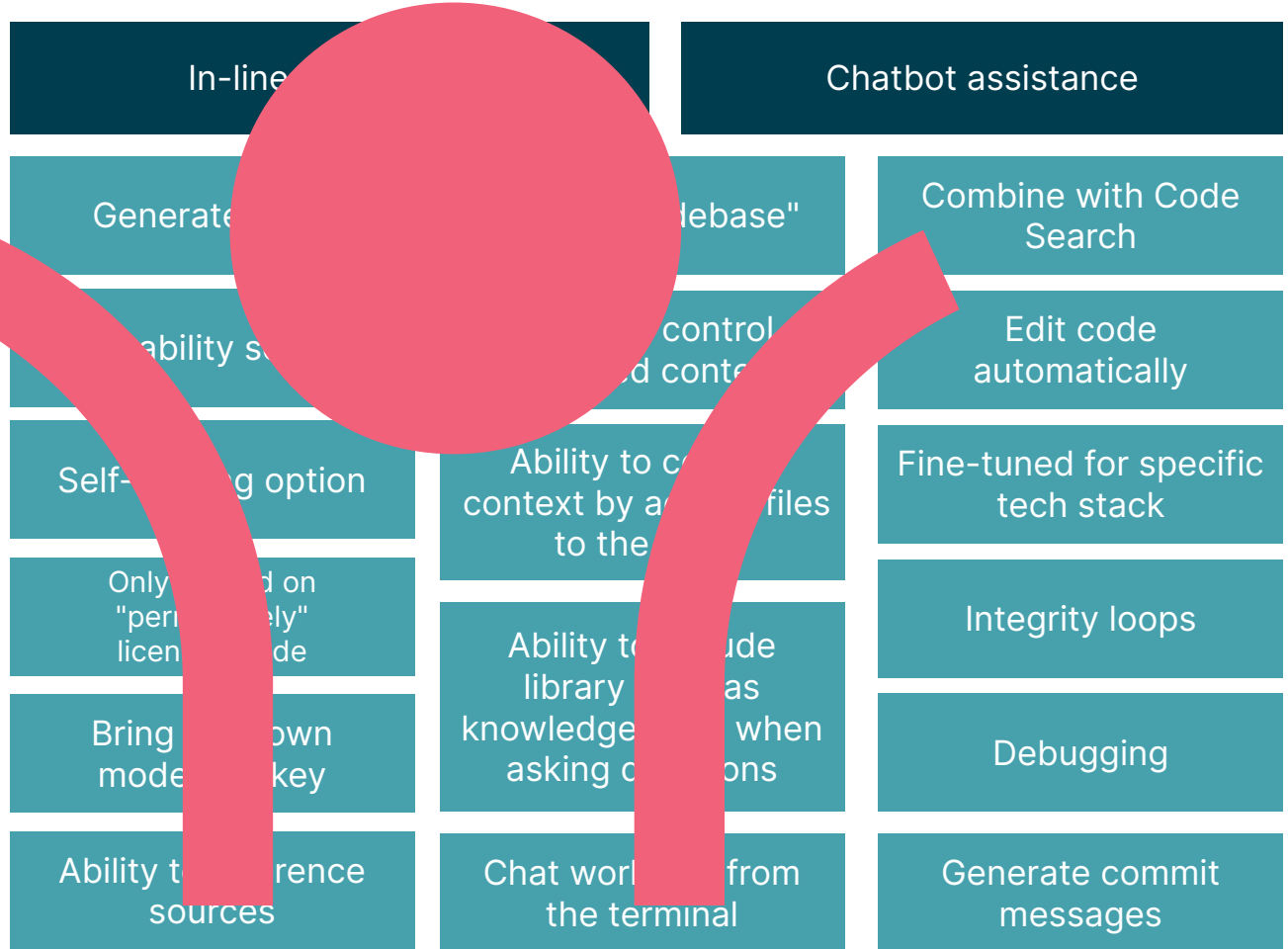
Copy code

```
interface EmojiUsage {  
    emoji: string;  
    count: number;  
}  
  
interface Message {  
    id: string;
```

**But  
there's  
a lot  
more!**

In-line assistance		Chatbot assistance
Generate tests	"Ask the codebase"	Combine with Code Search
Vulnerability scanning	Ability to control searched context	Edit code automatically
Self-hosting option	Ability to control context by adding files to the chat	Fine-tuned for specific tech stack
Only trained on "permissively" licensed code	Ability to include library docs as knowledge base when asking questions	Integrity loops
Bring your own model API key		Debugging
Ability to reference sources	Chat workflow from the terminal	Generate commit messages

**But  
there's  
a lot  
more!**



# How useful are coding assistants?

- “It depends”



Experience of the developer(s)



Prevalence of the tech stack



Type of problem



Isolation and size of the problem



Problem's tolerance for errors



Existing codebase

...

**“AI is not  
good software.  
It is pretty  
good people.”**



Ethan Mollick

<https://www.oneusefulthing.org/p/ai-is-not-good-software-it-is-pretty>

# Data exploration



# Translating natural language into queries

The screenshot shows the Google Cloud Metrics Explorer interface. At the top, there's a search bar with the text "Search (/) for resources, docs, products...". Below that, the "Metrics explorer" header is visible. A modal window titled "Help me modify" is overlaid on the chart area. The modal contains a PromQL query: `max(rate(run.googleapis.com:request_count{monitored_resource="cloud_run_revision", response_code!="200", project_id="cymbal-next-23-395321"}[${__interval}])))`. Below the query, it asks "What would you like to do to this query?" and provides a text input field with the text "Show me all services impacted by this error". An example below the input field reads "Ex: Show for each zone".

Google Cloud

Cymbal Next 23

Search (/) for resources, docs, products...

Search

Metrics explorer

CHART TABLE BOTH

Help me modify

```
max(rate(run.googleapis.com:request_count{monitored_resource="cloud_run_revision", response_code!="200", project_id="cymbal-next-23-395321"}[${__interval}])))
```

What would you like to do to this query?

Show me all services impacted by this error

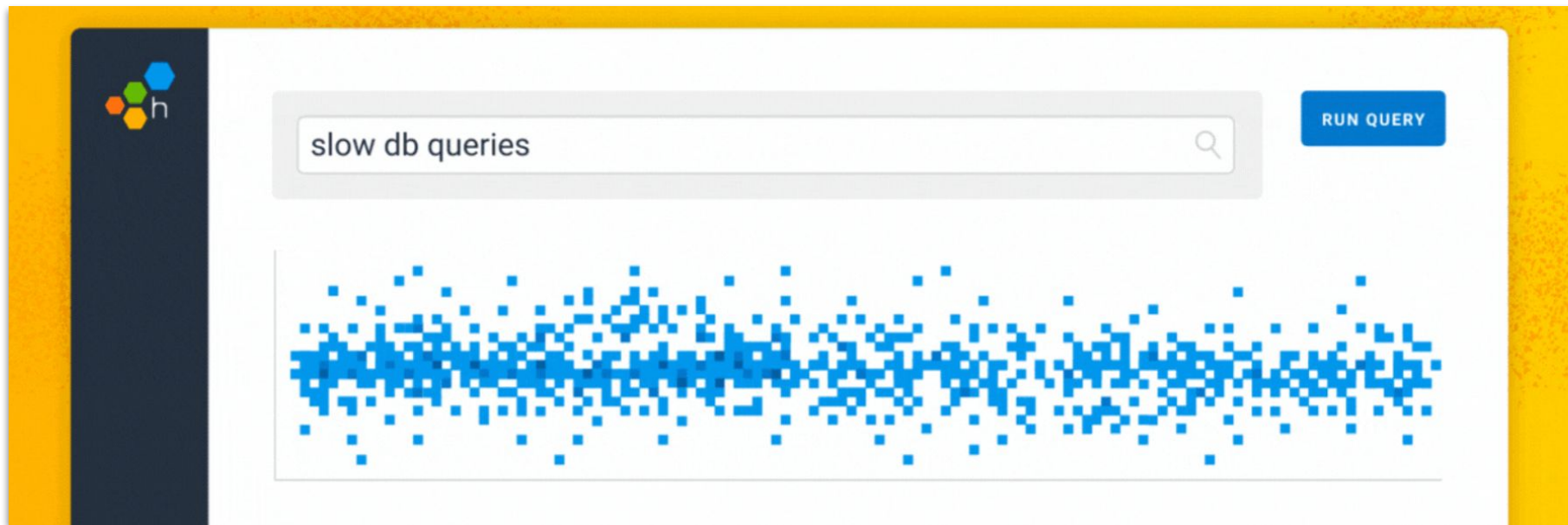
Ex: Show for each zone

UTC-6 10:05 AM

Queries > Q

PromQL Q

# Translating natural language into queries



The screenshot displays the Honeycomb query assistant interface. On the left is a dark sidebar with the Honeycomb logo (four colored hexagons: orange, green, blue, yellow) and a white 'h'. The main area features a search bar containing the text 'slow db queries' and a magnifying glass icon. To the right of the search bar is a blue button labeled 'RUN QUERY'. Below the search bar is a large scatter plot consisting of numerous small blue squares, representing the results of the query.



<https://www.honeycomb.io/blog/honeycomb-natural-language-querying-query-assistant>



# Translating natural language into queries

## Team AI Demo

About Team AI

Story brainstorming

Story generation

Tasking

Knowledge Base

Test DB Exploration

</> Schema

```
1 Stores(StoreID, StoreName, StoreLocation, StoreManager, Store
2 Products(ProductID, ProductName, ProductDescription, Category
3 Suppliers(SupplierID, SupplierName, ContactPerson, PhoneNumbe
4 Employees(EmployeeID, FirstName, LastName, StoreID, Position,
5 Inventory(InventoryID, ProductID, StoreID, QuantityOnHand);
6 Sales(SaleID, ProductID, StoreID, EmployeeID, SaleDate, Sale
7 Customers(CustomerID, FirstName, LastName, Email, PhoneNumbe
8 CustomerPersonas(PersonaID, PersonaName, PersonaDescription);
9 OnlineOrders(OrderID, CustomerID, OrderDate, DeliveryAddress,
10 OnlineOrderDetails(OrderDetailID, OrderID, ProductID, Quantit
11 Interests(InterestID, InterestName, InterestCategory);
12 CustomerInterests(CustomerID, InterestID);
```

Chatbot

which store has the most number of inventory items?

The Huiboo Store 4 has the most number of inventory items.

*QUERY USED: ' SELECT s.StoreName, SUM(i.QuantityOnHand) AS TotalInventory FROM Stores s JOIN Inventory i ON s.StoreID = i.StoreID GROUP BY s.StoreName ORDER BY TotalInventory DESC LIMIT 1; '*

Ask a question

SAMPLE QUESTIONS: which store has the most number of inventory items? ||  
how many stores have the polo shirt product?

# Translating natural language into queries

## Team AI Demo

About Team AI

Story brainstorming

Story generation

Tasking

Knowledge Base

Test DB Exploration

</> Schema

```
1 Stores(StoreID, StoreName, StoreLocation, StoreManager, Store
2 Products(ProductID, ProductName, ProductDescription, Category
3 Suppliers(SupplierID, SupplierName, ContactPerson, PhoneNumbe
4 Employees(EmployeeID, FirstName, LastName, StoreID, Position,
5 Inventory(InventoryID, ProductID, StoreID, QuantityOnHand);
6 Sales(SaleID, ProductID, StoreID, EmployeeID, SaleDate, Sale
7 Customers(CustomerID, FirstName, LastName, Email, PhoneNumbe
8 CustomerPersonas(PersonaID, PersonaName, PersonaDescription);
9 OnlineOrders(OrderID, CustomerID, OrderDate, DeliveryAddress,
10 OnlineOrderDetails(OrderDetailID, OrderID, ProductID, Quantit
11 Interests(InterestID, InterestName, InterestCategory);
12 CustomerInterests(CustomerID, InterestID);
```

Chatbot

which store has the most number of inventory items?

The Huiboo Store 4 has the most number of inventory items.

*QUERY USED: ' SELECT s.StoreName, SUM(i.QuantityOnHand) AS TotalInventory FROM Stores s JOIN Inventory i ON s.StoreID = i.StoreID GROUP BY s.StoreName ORDER BY TotalInventory DESC LIMIT 1; '*

Ask a question

SAMPLE QUESTIONS: which store has the most number of inventory items? || how many stores have the polo shirt product?

# Documentation



# It's not always about the output



The video player shows a presentation slide with a green background. On the left, a stick figure is in a thought bubble. The text on the slide reads: "Surface and understand COMPLEXITY." and "I just heard Alice *explain* that complex thing. AGAIN." A small number "2" is in the bottom right of the slide. To the right of the slide, the event information is displayed: "# LEAD DEV", "9th June 2017", "London, UK", and a small video thumbnail of the speaker, Birgitta Boeckeler. The video player interface includes a title, channel information (LeadDev, 47.1K subscribers), and interaction buttons (Subscribe, Like, Dislike, Share, Download, and a menu icon).

Surface and understand  
**COMPLEXITY.**

I just heard Alice *explain* that complex thing. AGAIN.

2

# LEAD DEV  
9th June 2017  
London, UK

We're Agile, We Don't Do Documentation – Birgitta Boeckeler | The Lead Developer UK 2017

LeadDev  
47.1K subscribers

Subscribe

124

Share

Download

© 2023 Thoughtworks



**Sometimes the  
journey is the  
destination,  
not the artifact**

# ChatGPT summarising 190 commit messages



Here is a summary of the changes made to the application based on the provided changelog:

- Chat Analytics Introduction:** The application underwent a substantial restructure, starting with the introduction of "ChatAnalytics." This module may contain the core functionality related to the analysis of chat data.
- Data Handling and Refactoring:** Several changes were made to how data was handled, beginning with the switch to 'allData' in multiple steps. An endpoint was also added to serve all data at once. A series of refactoring changes improved code organization, such as creating subfolders, moving methods, and renaming response objects.
- Aggregation and Visualization Enhancements:** There were multiple updates for the aggregation of user data, including aggregate by country and office. New charts were added for role, tenure, and total experience. Additionally, the chart's visual elements were enhanced, like increasing the width of the activity-by-date chart, adding a bar to indicate duration of threads, and improving the layout of top threads.
- Improvements in Analysis Metrics:** Several new analysis metrics were added, such as the number of messages in a time period, percentage of messages in the activity histogram, and the addition of median number of messages per week. Filtering was implemented to exclude underrepresented dimensions and messages not written by humans. There were also changes to how certain parameters were considered, like including "unknown" senders and not [considering](#) "Unknowns" in member demographics.
- Backend and Performance Enhancements:** The application saw several performance improvements, most notably through caching. Initially, a crude server cache was introduced, which was later extended to file storage. The data cache also started to include tokens and person information. The application also moved from `'ts-node'` to `'nodemon'`, and `'esbuild'` was introduced to shorten the feedback loop.
- Error Handling and Testing:** Error handling was improved with the addition of HTML error displays. Several test-related updates were also made, including fixing tests and the introduction of Jest for source maps.
- UI and Usability Improvements:** Several changes were made to improve the application's usability, such as creating an index page with input fields, showing a date when data was pulled, and offering options to use and clear cache. The PDF export functionality was improved with nicer page breaks.
- General Maintenance:** As part of general housekeeping, there were numerous minor fixes and cleanups, updates to the TODOs, fixing of transpile errors, and updates to the README. Styling was also updated, and the number of top threads displayed was increased from 5 to 10.

# New ways to discover documentation



# is there a concept of static functions in python?

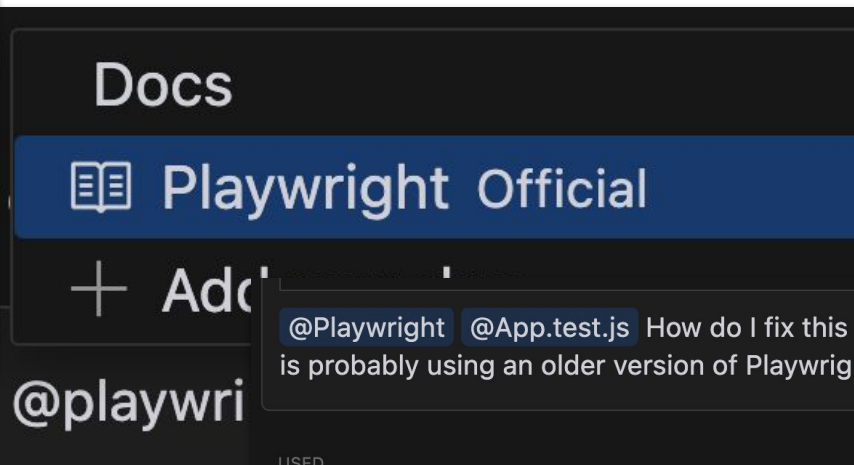
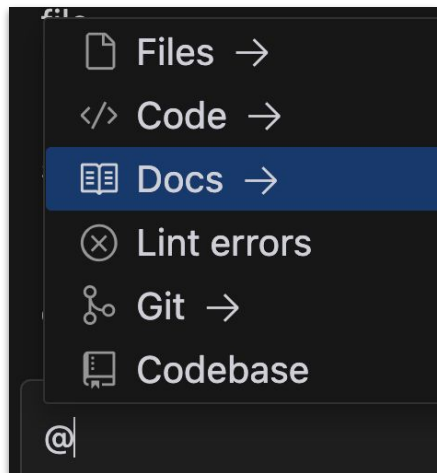
Yes! In object oriented programming languages such as C++, Java, JavaScript etc., classes contain both instance methods and static methods. Instance methods are bound to specific objects whereas Static methods belong to the class itself. For eg;

Java Code:

```
public class ExampleClass {  
    public static void printMessage() {  
        System.out.println("Hello World");  
    }  
}
```



# Including reference docs



@Playwright @App.test.js How do I fix this test? beforeAll does not exist, so this test is probably using an older version of Playwright?

USED

[Rerun without context](#)

App.test.js Current file Playwright Docs

DOCS PAGES

writing-tests intro running-tests codegen-intro

The issue you're facing is that the `beforeAll` function does not exist in the current version of Playwright. It seems like the test code you provided is using an older version of Playwright that had the `beforeAll` function.

To fix this test, you can replace the `beforeAll` function with the `beforeEach`



[CURSOR.SO](https://cursor.so)

# On-the-fly documentation



# “Ask the codebase”



where am I calculating the gender distribution?



Reading

[TS ChatAnalytics.spec.ts](#)



Reading

[TS ChatApiResponses.spec.ts](#)



Done



07:50 PM



The gender distribution is calculated in two places in the codebase:

1. In the `getAllData` method in the `ChatAnalytics` class, the gender distribution of members is calculated. This is done by iterating over the members of a chat space and categorizing them by their gender. The results are stored in `result.members.demographics.absolute.genders` and `result.members.demographics.relative.genders` on lines 129 and 136.
2. In the same method, the gender distribution of senders (people who have sent messages in the chat

**bloop.**

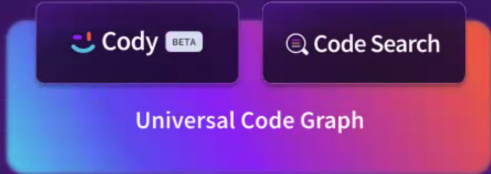
**Combining LLMs  
with other tools  
and data unlocks  
more potential**



# Teaming an LLM up with tools that understand code

WHAT IS SOURCEGRAPH?

Sourcegraph is a code AI platform that makes it easy to read, write, and fix code—even in big, complex code bases.



Universal Code Graph

Code LLM

**bloop.**

lockPeo  
etAllD where am I calculating the gender distribution?

Includ  
) .toBe  
ll({

calculating gender distribution

Reading TS ChatAnalytics.spec.ts ↗

Reading TS ChatApiResponses.spec.ts ↗

Reading TS ChatAnalytics.spec.ts ↗

Reading TS ChatApiResponses.spec.ts ↗

Done ☰

The gender distribution is calculated in two places in the codebase:

1. In the `getAllData` method in the `ChatAnalytics` class, the gender distribution of `members` is calculated. This is done by iterating over the members of a chat space and categorizing them based on their gender. The results are stored in `result.members.demographics.absolute.genders` and `result.members.demographics.relative.genders` respectively.
2. In the same method, the gender distribution of senders (people who have sent messages) is calculated.



**Data quality  
still matters**

# Writing user stories



# Find gaps in our thinking earlier.

The screenshot displays the Thoughtworks Team AI Demo interface. At the top left is the Thoughtworks logo. Below it is the title "Team AI Demo". A navigation bar contains five tabs: "About Team AI", "Story brainstorming" (which is selected), "Story generation", "Tasking", "Knowledge Base", and "Test DB Exploration".

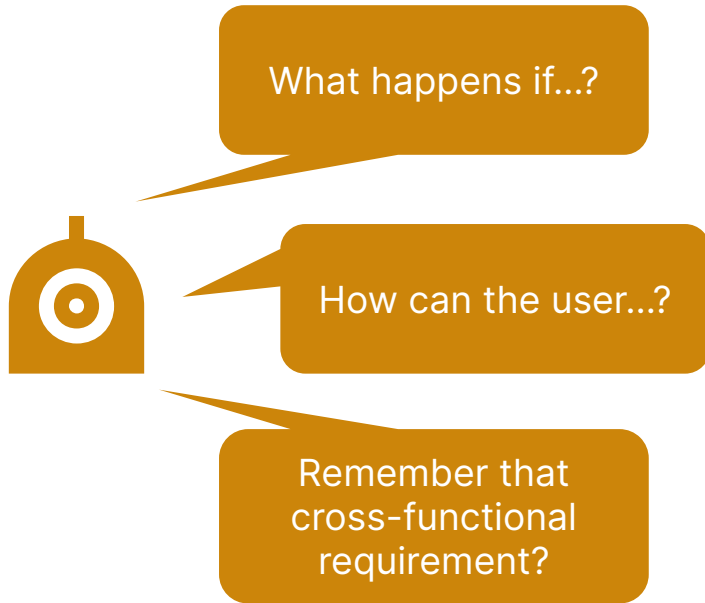
The main content area is divided into two columns. The left column contains:

- User story**  
SAMPLE STORY: Plan activity for a specific date: The user should be able to add a new planned activity to the application, for a specific customer, with a date and some notes describing what they want to do.
- A text input field with a cursor.
- Business Context**  
Preloaded, reusable context about the domain
- You're part of a team building a web application that will allow salespeople in a company to organise their work and document their customer relationships.**
- Users:** Sales Managers who are looking after a portfolio of restaurants/hotels/caterers/traders, looking to build a personal relationship and increase the revenue their respective customers are generating.
- Functionality:** Plan customer visits and document them, view plans on a map and execute unplanned visits for nearby customers. It needs to work on mobile phones and with variable

The right column contains a tab labeled "Assistant" with a small icon to its left.



# Find gaps in our thinking earlier.



**“AI is not  
good software.  
It is pretty  
good people.”**



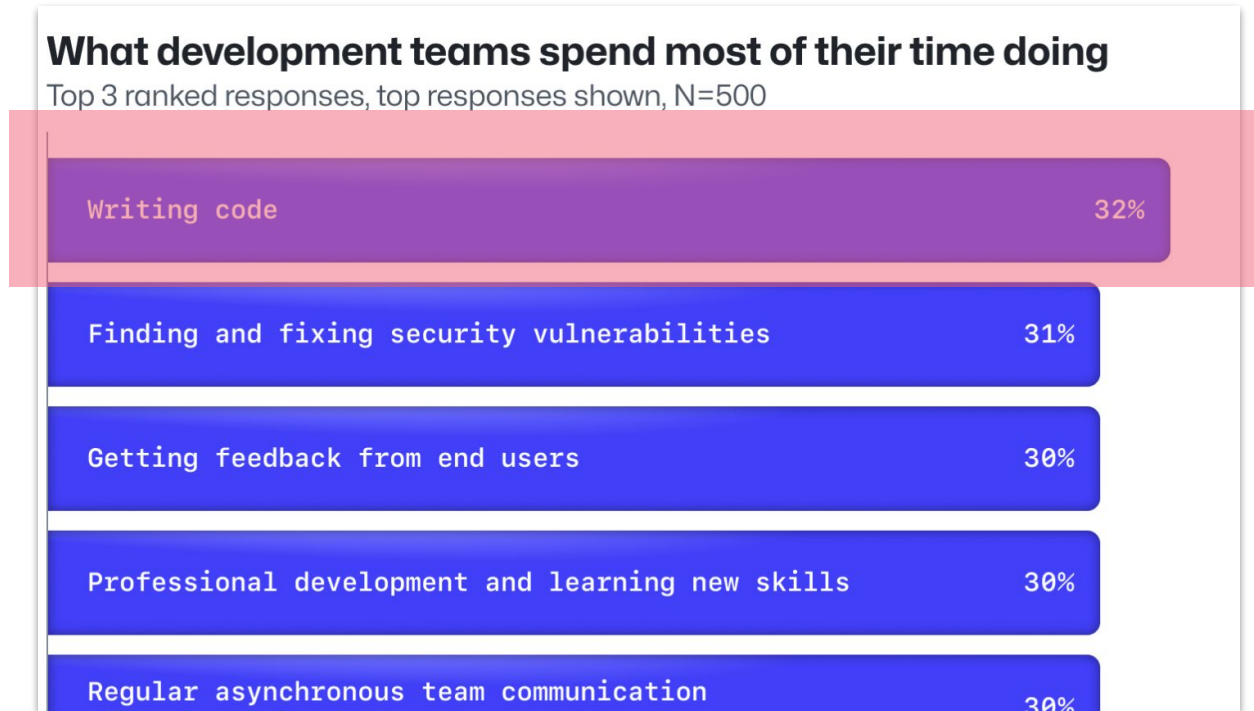
Ethan Mollick

<https://www.oneusefulthing.org/p/ai-is-not-good-software-it-is-pretty>



**How can this impact  
delivery performance  
overall?**

# Coding is less than 40% of what a team spends time on.



<https://github.blog/2023-06-13-survey-reveals-ai-impact-on-the-developer-experience/>

# GitHub's numbers are being misinterpreted.



/ **Blog**

Menu ▾

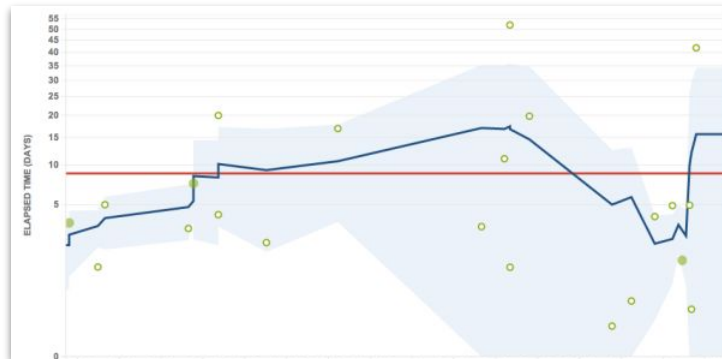
**Try GitHub Copilot**

group without Copilot).

- The striking difference was that **developers who used GitHub Copilot completed the task significantly faster—55% faster** than the developers who didn't use **GitHub Copilot**. Specifically, the developers using GitHub Copilot took on average 1 hour and 11 minutes

# A heuristic: Potential impact on cycle time

	Assumptions			
Scenario	Part of cycle time spent on coding	Part of coding supportable with coding assistant	Rate of faster task completion with coding assistant	Time saved in cycle time
Very optimistic	40%	60%	55%	<b>13%</b>



# Look beyond speed metrics

**Developer flow and experience do matter**

**Test coverage**

**Onboarding time and general staff mobility**

**Extensibility and maintainability**

**Debugging time**

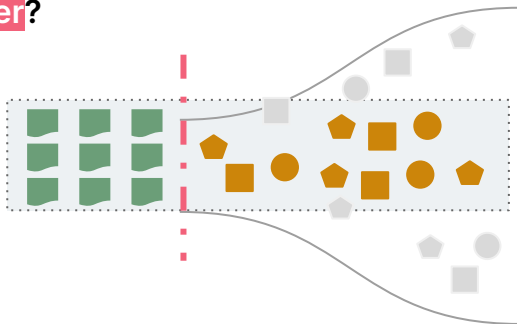
# Look beyond coding assistance



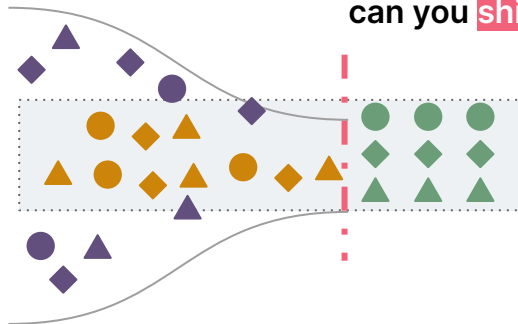


# Look at the whole system

If you can code faster,  
can you **fill the backlog**  
**faster**?



If you can code faster,  
can you **ship faster**?



More code, more  
problems: can you  
keep your **technical**  
**debt** in check?



If you can produce  
more features, how  
are you tracking their  
**value**?



## Are you super excited about this?



Data quality still matters.



Sometimes the journey is the destination.



Keep the whole system and the users in mind



“AI is not good software”



Combining LLMs with other tools unlocks more potential.



There is lots of room for improvements



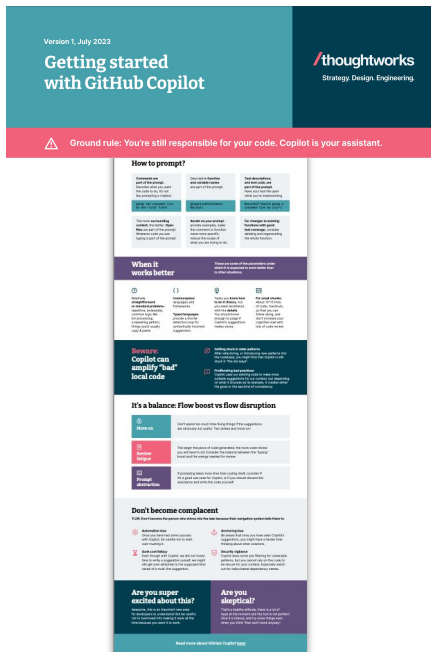
It's about more than coding



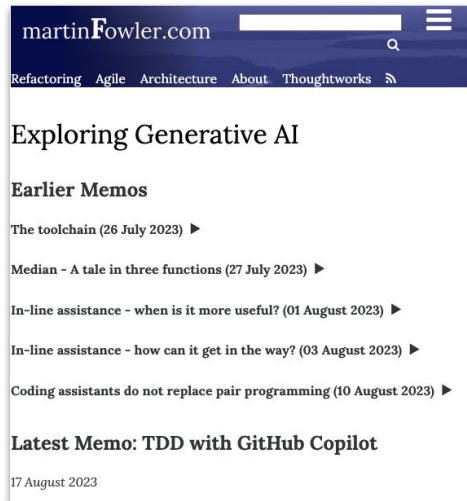
## Are you skeptical?

# Read more

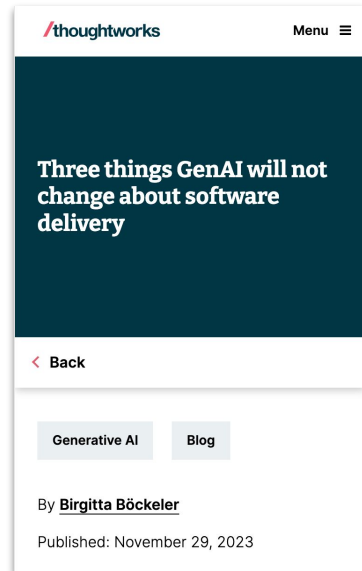
<https://www.thoughtworks.com/insights/blog/generative-ai/getting-started-with-github-copilot>



<https://martinfowler.com/articles/exploring-gen-ai.html>



<https://www.thoughtworks.com/insights/blog/generative-ai/three-things-GenAI-will-not-change-about-software-delivery>





**Come to our booth to chat  
and play with our demo**



Google's Duet AI,  
prompt "harness energy",  
style "Vector art"

# Thank you!

 **thoughtworks**

© 2023 Thoughtworks

