UNIVERSITÉ DE GENÈVE

FNSNF Swiss National Science Foundation

Designing Constrained AI Assistance for Novice Programmers

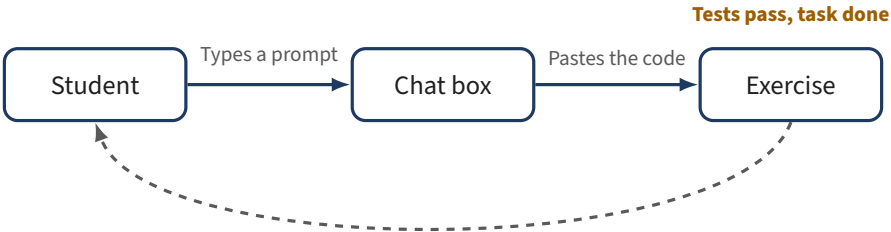
Tensions and Hypotheses from a JupyterLab Hint System

Alexandre De Masi • Chen Wang • Laurent Moccozet

NordiCHI 2026 • Vaasa • 5 October 2026

TECFA, Technologies for Training and Learning
Faculty of Psychology and Educational Sciences, University of Geneva

When the chat box answers, novices lose



Completion up, transfer flat	The strong pull ahead, the weak fall behind	Novices cannot write the prompt that would help them
Kazemitabaar et al., CHI ’23	Prather et al., ICER ’24	Ma et al., CHI ’26

So we constrain the interface.

Design process and rejected alternatives

Rejected alternatives

Free-text field
Invites “how do I write a for loop?”

Topic dropdown
Novices cannot say if it is syntax or logic

Hints only on request
Help avoidance; asking has a social cost

Adopted design

Infer need from the code

- One button, no prompt
- First hint on every run
- Questions only

Help-seeking: Aleven et al.
Socratic tutoring: Graesser et al.
No code: CodeAid, CodeHelp

A constraint set: it does not adapt, fade, or hand control back. The study asks what that costs.

Four constraints

C1 No free-form chat

One button, “Request hint”. The system reads the code and the errors; the student cannot type a question.

C2 No code

No fix, no snippet, no pseudo-code, and no explanation of what is wrong.

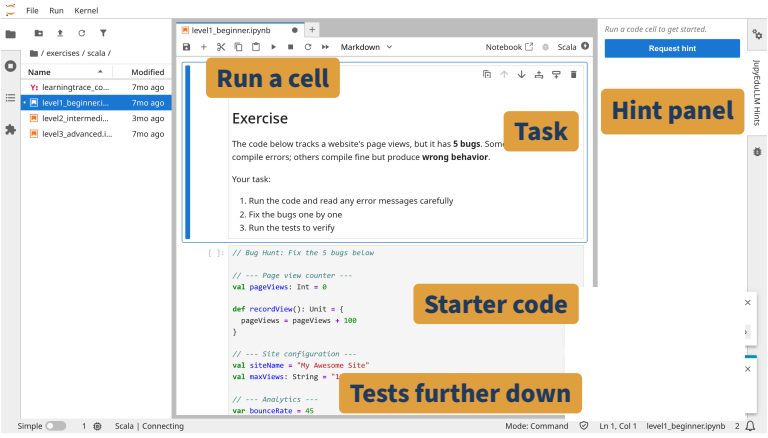
C3 Automatic first hint

Every cell execution triggers one; follow-ups are voluntary.

C4 Socratic only

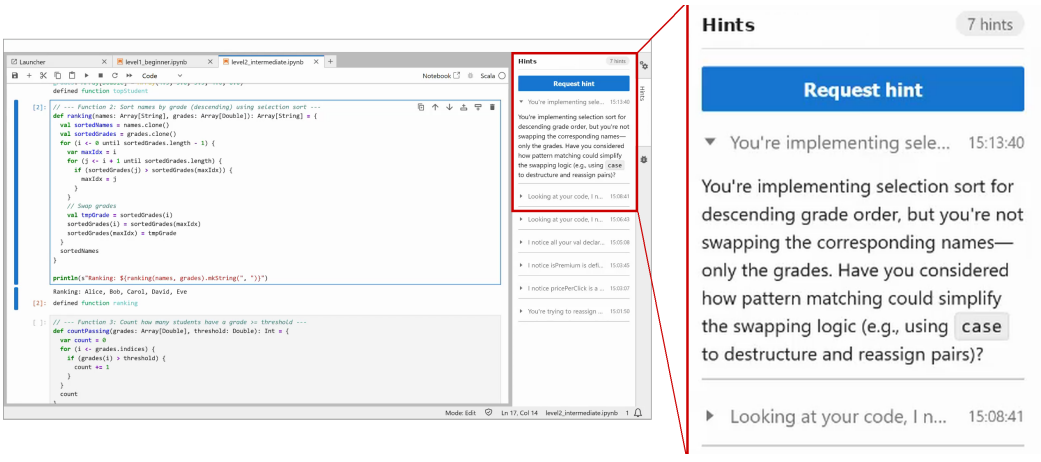
Two or three sentences, one question, one issue at a time.

The environment: a Jupyter notebook



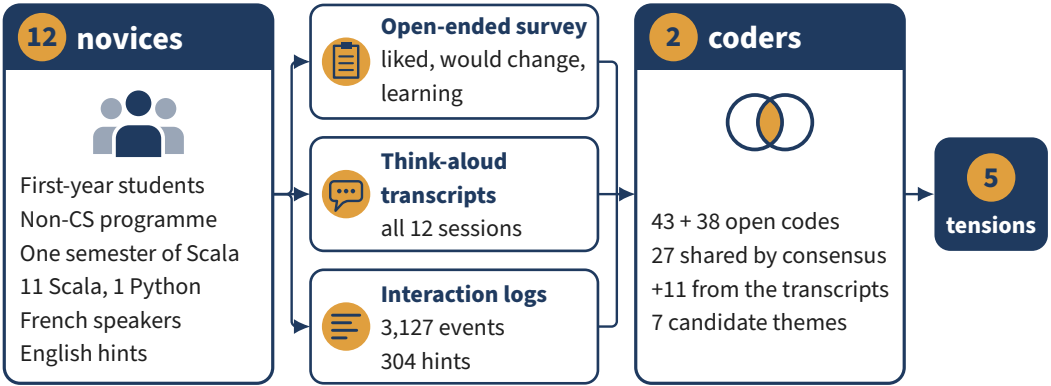
Task 1: immutability bug hunt, Scala.

Our addon: the hint panel

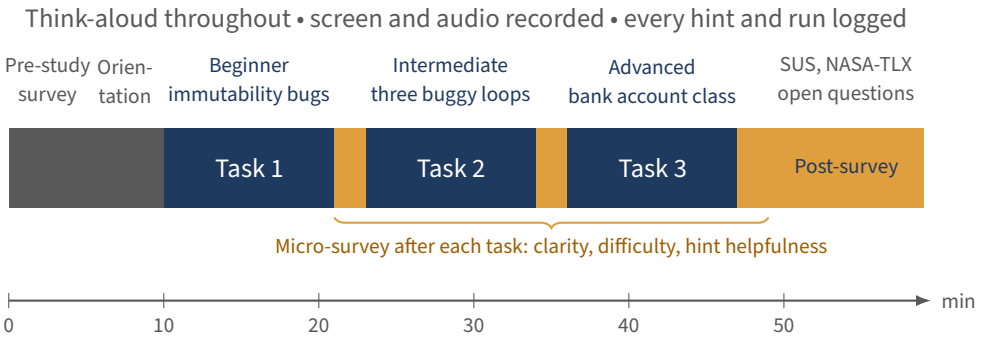


Real session, Task 2. Hints stack as cards; code names are highlighted, code is never written.

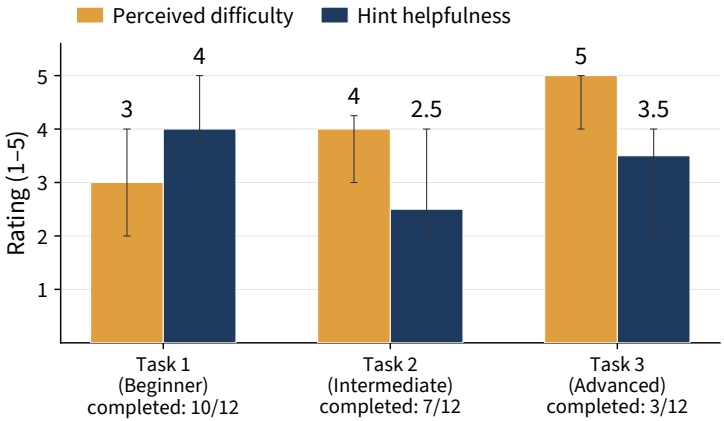
Twelve novices, three data streams, five tensions



One session, 55 minutes



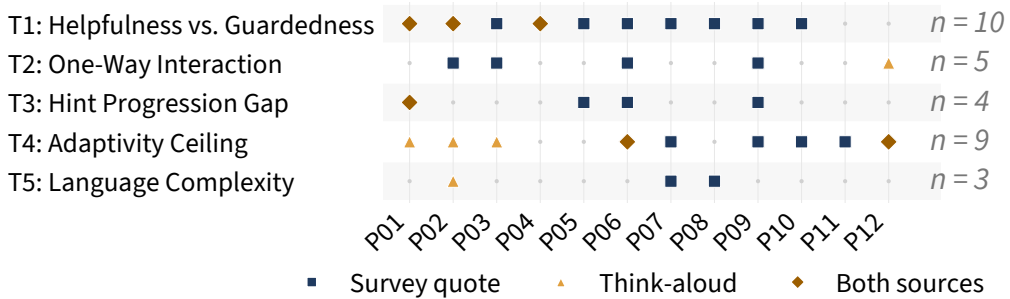
Harder tasks, less help



- Completion 10, 7, then 3 of 12
- Helpfulness falls, then only half recovers
- SUS 72.5: acceptable, not loved

Much of the frustration below comes from a task most could not finish.

Five design tensions



T1 Valued on easy tasks, broke on hard ones

Ten of twelve praised the Socratic approach in principle.

“if I’m completely blocked it should give me maybe one answer to help go forward, whereas if I have a little issue, it should guide me” (P08)

- The breakdown is a threshold: it appears where the task passes the student’s competence
- The problem sits in task–learner calibration; the Socratic format itself was endorsed

T2 The system sees the code, not the confusion

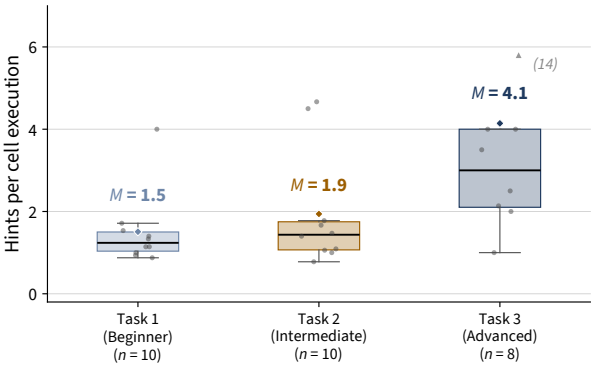
Five of twelve wanted to talk back.

“you cannot communicate with the assistance. It’s just a one-way road.” (P03)

“there is no way to tell it what seems to be blocking us” (P09)

Code and errors are a lossy picture of understanding. A student can be stuck on something that leaves no trace in the cell.

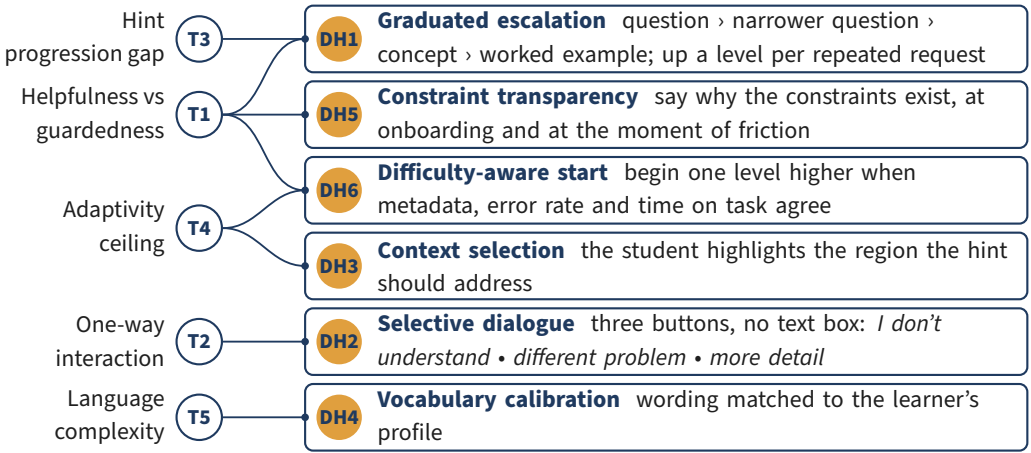
T3 and T4 No new signal, no escalation



- 47 of 203 follow-ups (23%) came within 30 s with no edit; two in three repeated the same issue
- Hints per cell rise with difficulty, and so does the spread between students
- A question cannot bridge a gap the student lacks the words for

“the AI gives us no information about what a function means” (P09)

Six design hypotheses



What we did not foresee

1 A threshold

praised, then abrupt failure

2 Measurable traces

23% rapid re-asks

3 Automatic hints were welcome

intrusion expected, none felt

4 Confusion the code can’t show

no chat, no way to say so

Constrained should not mean rigid

The constraint level is a design parameter.

DH1: one level more direct per repeated request, if nothing changed

DH6: start one level higher on hard tasks

Keep the box closed, open a narrow channel back (DH2, DH3). Next: measure learning, against open chat and no AI.

Thank you

Designing Constrained AI Assistance for Novice Programmers: Tensions and Hypotheses from a JupyterLab Hint System

with Chen Wang and Laurent Moccozet

Alexandre De Masi

alexandre.demasi@unige.ch

ademasi.ch

Beyond the Chatbot • Swiss National Science Foundation

TECFA, University of Geneva

The paper

doi.org/10.1145/3829807.3829904

Backup: limitations

Small sample

12 participants, one institution, one 55-minute session

No learning outcomes

Surfaces tensions; a controlled study tests them

No comparison arm

Some tensions may appear with any hint system

One model

Qwen3 Coder Next; hint quality may differ elsewhere

Notebook cells

Out-of-order runs and stale state add their own load

Designers as analysts

Independent coding, disconfirming evidence, verbatim quotes

Backup: the Scala confound

11 of 12 chose Scala, the language of their course; **one used Python**
Harder tasks drew on end-of-semester material, not yet consolidated

Constraint-specific

T1 **T2** **T3**

Would appear in any language, at other thresholds

Amplified

T4

By recently taught constructs

`grades.clone()`

Context-dependent

T5

English hints

French speakers

Scala vocabulary

May affect all five tensions

Only one participant used Python; the classification is a working hypothesis.

Backup: T5 language complexity

Raised by three participants, two of them answering the survey in French

> *“The answers given by the LLM could be easier to understand (use simpler words)”*
>
> P08

Direct-answer system

Hard words are offset by the clarity of the fix

Socratic system

The question *is* the hint: an opaque question ends the interaction

Backup: DH1 and DH6 in detail

DH1 Graduated escalation: four levels

Counter per cell

Resets on a substantive code change

At least 60 s between levels

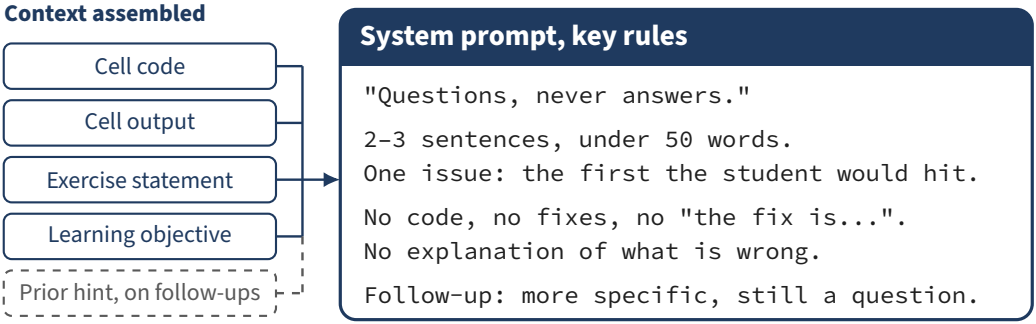
DH6 Difficulty-aware start

Backup: where the system sits

System	Chat	Code	Trigger	Scaffolding	Student model
CodeAid	Free-form	Pseudo-code	On demand	Explanation	No
CodeHelp	Free-form	None	On demand	Guiding text	No
DBox	Structured	None	On demand	Decomposition	Implicit
Khanmigo	Multi-turn	None	On demand	Socratic	Implicit
GitHub Copilot	Inline	Full code	Automatic	Completion	No
Ours	**None**	**None**	**Automatic + on demand**	**Socratic**	**No**

The most restrictive point: no chat, no code, automatic trigger, no student model

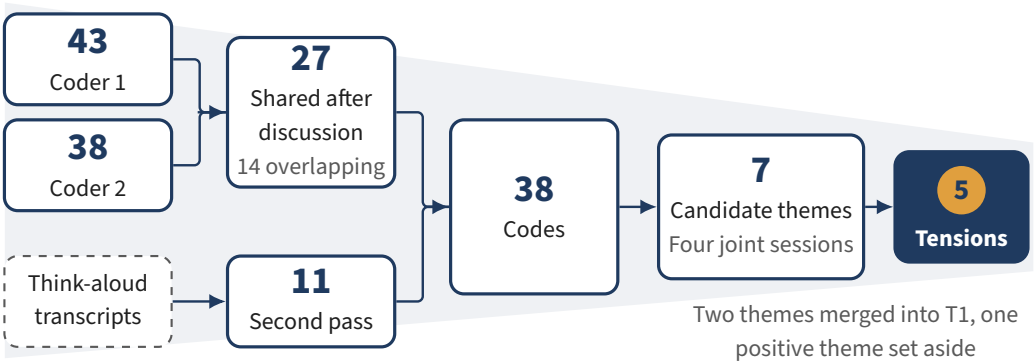
Backup: the system prompt, in short



Qwen3 Coder Next via OpenRouter; Qwen's recommended sampling: temperature 0.7, top-p 0.8, top-k 20.

Backup: analysis

Content analysis, interpretive orientation, thematic-analysis process



No kappa: consensus coding, not independent assignment