

Building an Effective AI Multi-agent system



Grigorij Dudnik





We hired an AI junior dev



Strona Główna

O Nas

Przykładowy Profil

Kontakt

Zaloguj się

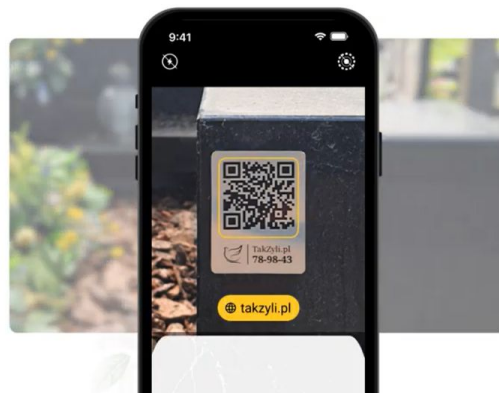
Zachowaj pamięć o swoich najbliższych

Stwórz profil pełen pięknych wspomnień i chwil.
Niech przyszłe pokolenia zobaczą jak żyli.

Wpisz numer profilu pamięci

Zobacz profil

lub [zobacz przykładowy profil](#)



Naive agent

Tools:



File 1



File 2



File 3



File 4



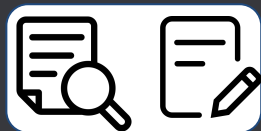
File 5

System message



Naive agent

Tools:



File 1



File 2



File 3



File 4



File 5

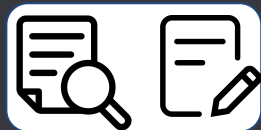
System message

Tool Call:  *Garbage*



Naive agent

Tools:



File 1



File 2



File 3




File 4



File 5

System message

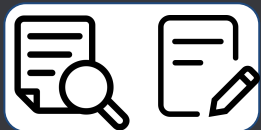
Tool Call:  *Garbage*

 File 1 *Garbage*



Naive agent

Tools:



File 1



File 2



File 3




File 4



File 5

System message

Tool Call:  *Garbage*

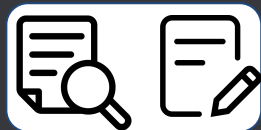
 File 1 *Garbage*

Tool Call:  *Garbage*



Naive agent

Tools:



File 1



File 2



File 3




File 4




File 5

System message

Tool Call:  *Garbage*

 File 1 *Garbage*

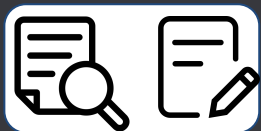
Tool Call:  *Garbage*

 File 2 *Garbage*



Naive agent

Tools:



File 1



File 2



File 3




File 4




File 5

System message

Tool Call:  *Garbage*

 File 1 *Garbage*

Tool Call:  *Garbage*

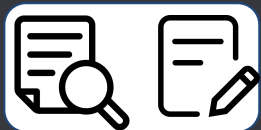
 File 2 *Garbage*

Tool Call:  *Garbage*



Naive agent

Tools:



File 1



File 2



File 3




File 4




File 5

System message


Tool Call:  *Garbage*

 File 1 *Garbage*

Tool Call:  *Garbage*

 File 2 *Garbage*

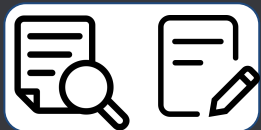
Tool Call:  *Garbage*

 File 3



Naive agent

Tools:



File 1



File 2



File 3




File 4




File 5

System message


Tool Call:  *Garbage*


 File 1 *Garbage*

Tool Call:  *Garbage*

 File 2 *Garbage*

Tool Call:  *Garbage*

 File 3

 File 3



Context is a key

Clean context is:

- Improved attention, which means better quality of responses
- Lower costs
- Easier to debug

Context trade-off: agent informativeness vs performance



Researcher

Tools:



System message

Tool Call:



File 1



File 3



File 5

Researcher

Tools:



System message

Tool Call:



File 1

...



File 3

...



File 5

Planer

Tools:



System message

Plan



Researcher

Tools:



System message

Tool Call:



File 1

...



File 3

...



File 5

Planer

Tools:



System message

Plan

Executor

Tools:



System message



File 3

Ok



File 5

Ok



Verbosity is (another) key

Ability to see raw LLM input is crucial

```
Run Feedback Metadata
{
  "tool": "replace_code",
  "tool_input": {
    "filename": "src/views/DashboardPage.vue",
    "start_line": 11,
    "end_line": 11,
    "new_code": "<span v-if=\"mem_profile.showTooltip\" class=\"tooltip\">L
  }
}
""
```

HUMAN

Action wasn't executed because of human interruption. He said: indents

HUMAN

File contents:

File: src/views/DashboardPage.vue:

```
<1><template></1>
<2> <div class="dashboard content"></2>
<3> <div class="container"></3>
<4> <h1 class="dashboard-title">Panel użytkownika</h1></4>
<5></5>
```



Create a good tools

```
1 sed -i 's/trainer = SFTTrainer(/alternative_trainer = SFTTrainer(/' training.py && sed -i '/^trainer = SFTTrainer(/,/^)$/c\alternative_trainer = SFTTrainer(\n  model=model,\n  max_seq_length=max_seq_length,\n  tokenizer=tokenizer,\n  formatting_func=create_prompt_universal,\n  args=alternative_args,\n  train_dataset=train_dataset,\n  eval_dataset=eval_dataset,\n)' training.py
```

- Make it as simple as possible to use (console vs dedicated tool)
- Replace code tool evolution
- Human acceptance needed for invasive tools
- Automatic check of LLM input

```
1|<template>
2| <div>
3|   <div class="scroll-container">
4|     <CoverPage :profile-data="post" />
5|     <SecondPage />
6|     <EducationPage />
7|     <AchievementsPage />
8|     <WorkPage />
9|     <AdditionalDescriptionPage />
10|     <FamilyPage :family-data="post.family" />
11|     <FinalPage />
12|   </div>
13| </div>
14|</template>
15|
...
```



File editing problem

```
1 from fastapi import FastAPI
2 import os
3 import json
4
5
6 @app.get("/")
7 async def main_page():
8     return {"message": "Hello World"}
```

File editing problem

```
1 from fastapi import FastAPI
2 import os
3 import json
4
5
6 @app.get("/")
7 async def main_page():
8     return {"message": "Hello World"}
```

```
1 from fastapi import FastAPI
2 import os
3 import json
4 import read_image
5 import render_image
6
7
8 @app.get("/")
9 async def main_page():
10     return {"message": "Hello World"}
```

File editing problem

```
1 from fastapi import FastAPI
2 import os
3 import json
4
5
6 @app.get("/")
7 async def main_page():
8     return {"message": "Hello World"}
```

```
1 from fastapi import FastAPI
2 import os
3 import json
4 import read_image
5 import render_image
6
7
8 @app.get("/")
9 async def main_page():
10     return {"message": "Hello World"}
```

```
1 from fastapi import FastAPI
2 import os
3 import json
4 import read_image
5 import render_image
6 @app.get("/")
7 async def main_page():
8     image = read_image.read_image("image.png")
9     return render_image.render_image(image)
10 async def main_page():
11     return {"message": "Hello World"}
```

Auto refreshing context

System message



 Edit file

Ok

 Edit file

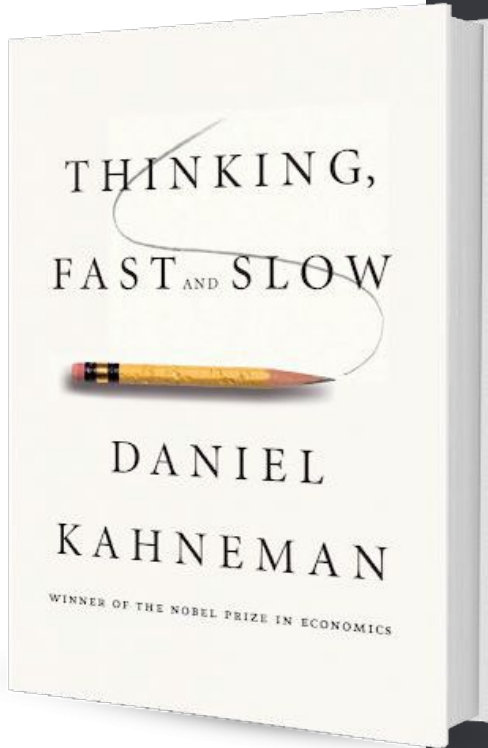
Ok

Programming tricks

- Autorefreshing of modified files in context
 - Automatic log check
 - Syntax check
- The more things done programmatically, the better.

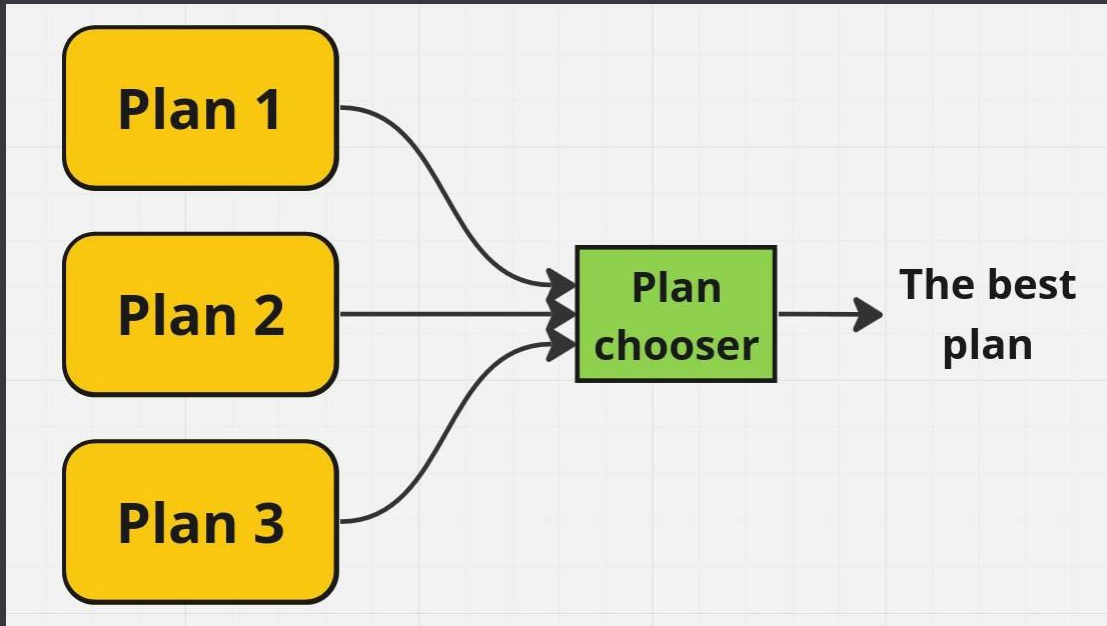


Make it think



- Use reasoning before every tool call
- XML or plain text for a reasoning; not json

Proposing 3 different plans and choosing the best



Manager

Tool list:

- Read tasks
- Create task
- Modify task
- Delete task
- Mark task as done
- Ask programmer to execute task
- Ask tester to check correctness of execution



Manager

Old tool list:

- ~~— Read tasks~~
- Create task
- Modify task
- Delete task
- ~~— Mark task as done~~
- ~~— Ask programmer to execute task~~
- ~~— Ask tester to check correctness of execution~~
- + Reorder tasks
- + Finish project planning



New tool list:

- Create task
- Modify task
- Delete task
- Reorder tasks
- Finish project planning

- Reduce number of tools
- Concentrate agent attention on single thing

Single Task Scope

- Enhance logging across the project
Enhance logging across the project, focusing on `generate_human_preferences.py` to ensure col
 - Verify configuration management
Ensure all configurations (like email settings) are managed using environment variables. Verify th
 - Add unit tests
Write unit tests for the following functions: `fetch_hacker_news_articles` `filter_articles` `send_email` l
 - Create deployment documentation
Create scripts or documentation for deploying the application to a server or cloud environment.
 - Update documentation
Update the README.md file to include the following: Instructions for running unit tests. Details a
- + Add task

- Manager concentrated only on creating good list of tasks

Try it by yourself



Key takeaways

- Create easy-to-use tools
- Shorten and clean the context
- Minimize scope of responsibilities
- Make it think
- Automate as much as possible