

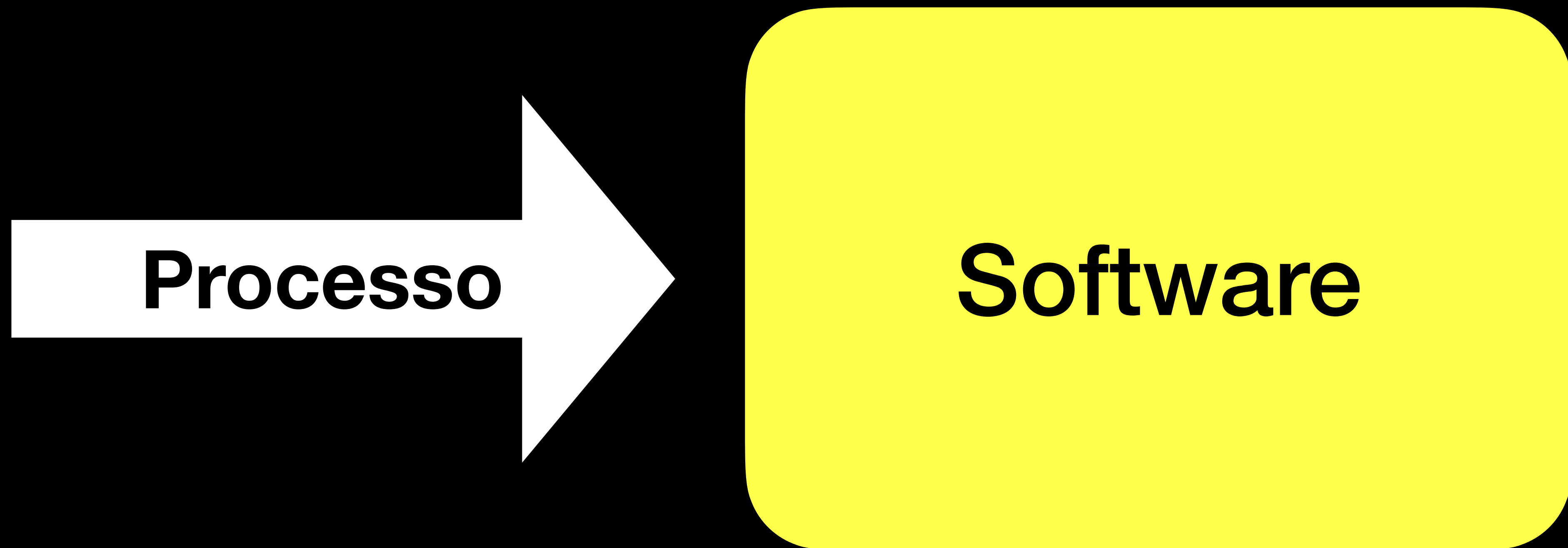
# Mitos e Realidades no Desenvolvimento de Software com IA

Paulo Borba  
Centro de Informática  
Universidade Federal de Pernambuco

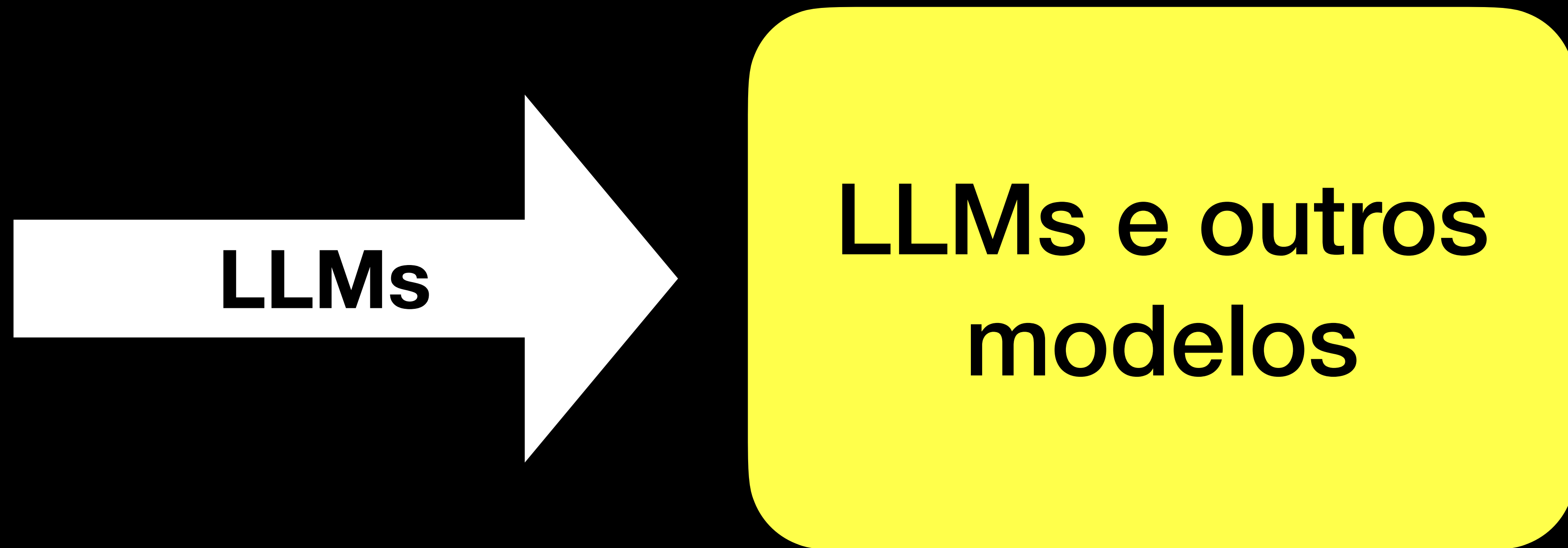


[pauloborba.cin.ufpe.br](http://pauloborba.cin.ufpe.br)

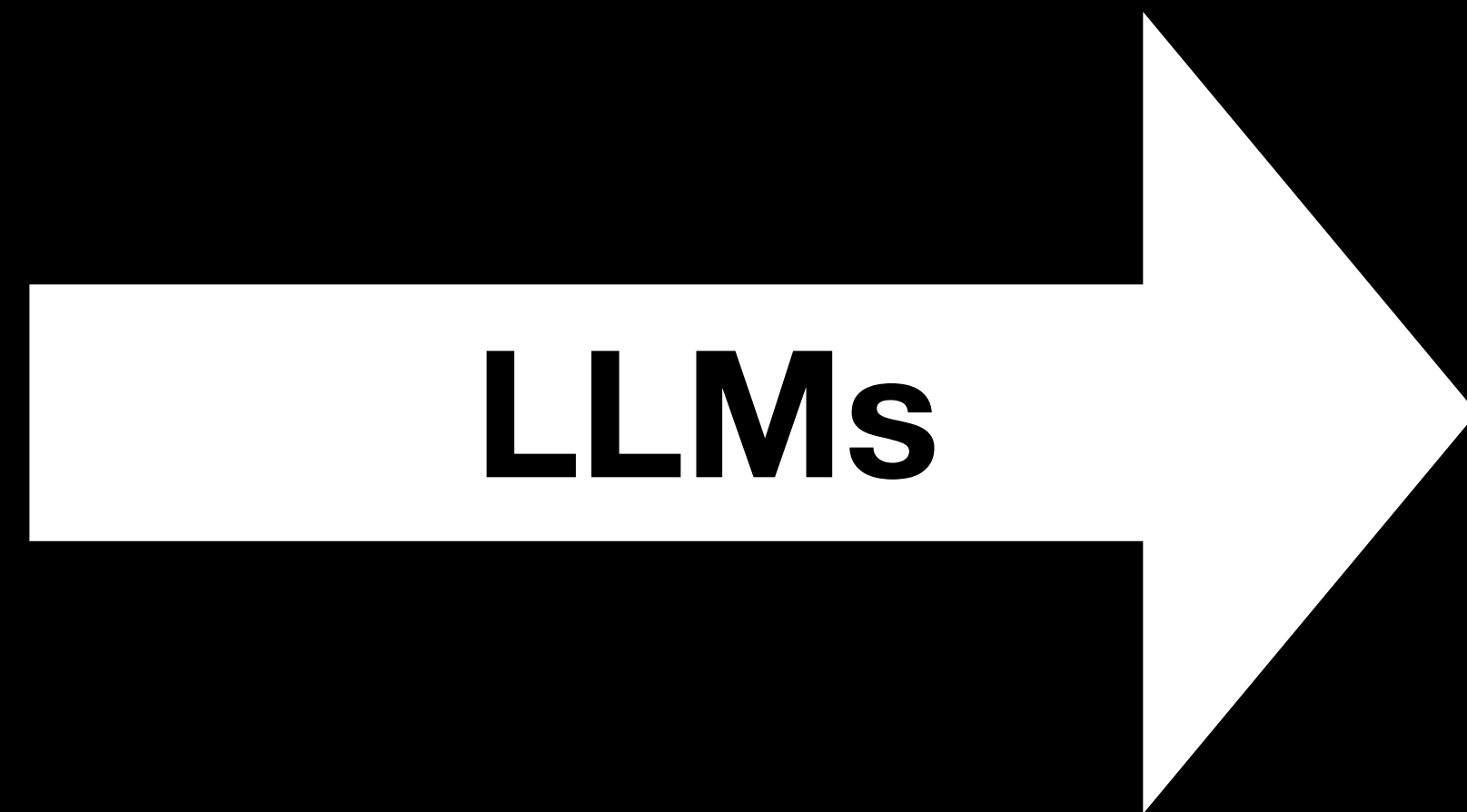
# A Engenharia de Software está mudando...



# LLMs mudam o processo e os produtos

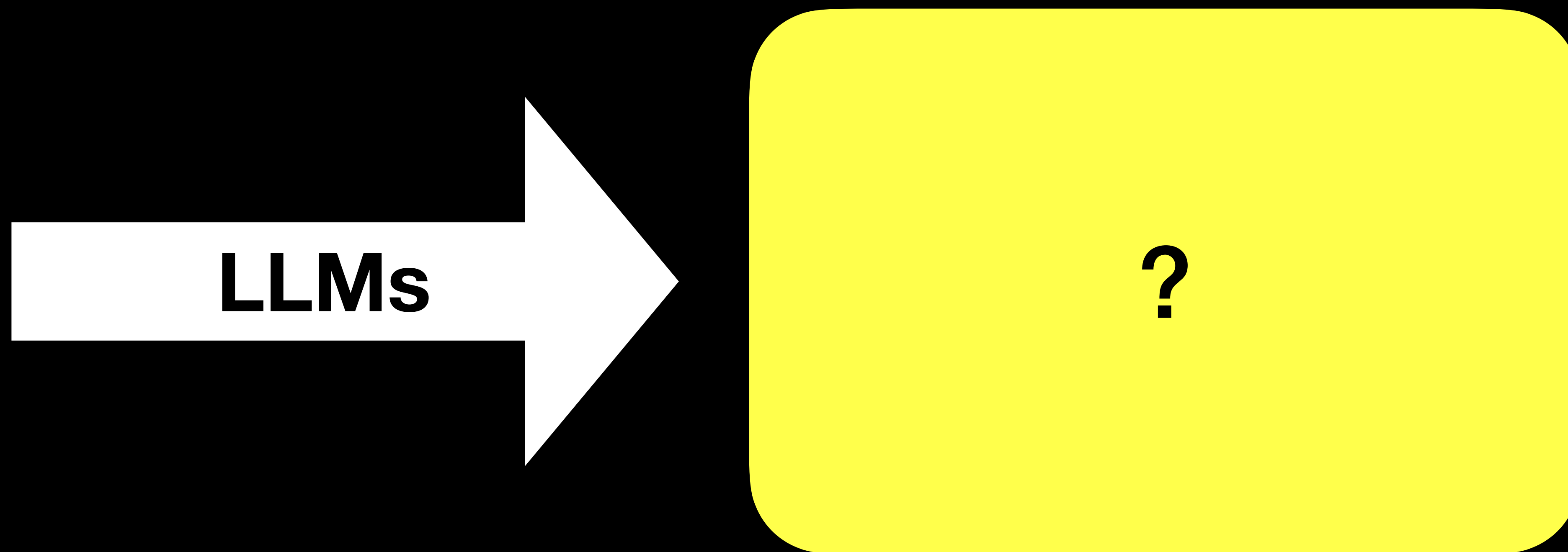


# Novos tipos de componentes, novos artefatos



**Código, pesos e  
prompts!**

# Foco no impacto da IA para ES



Marketing excessivo

Posições extremas

Ciência e  
serenidade

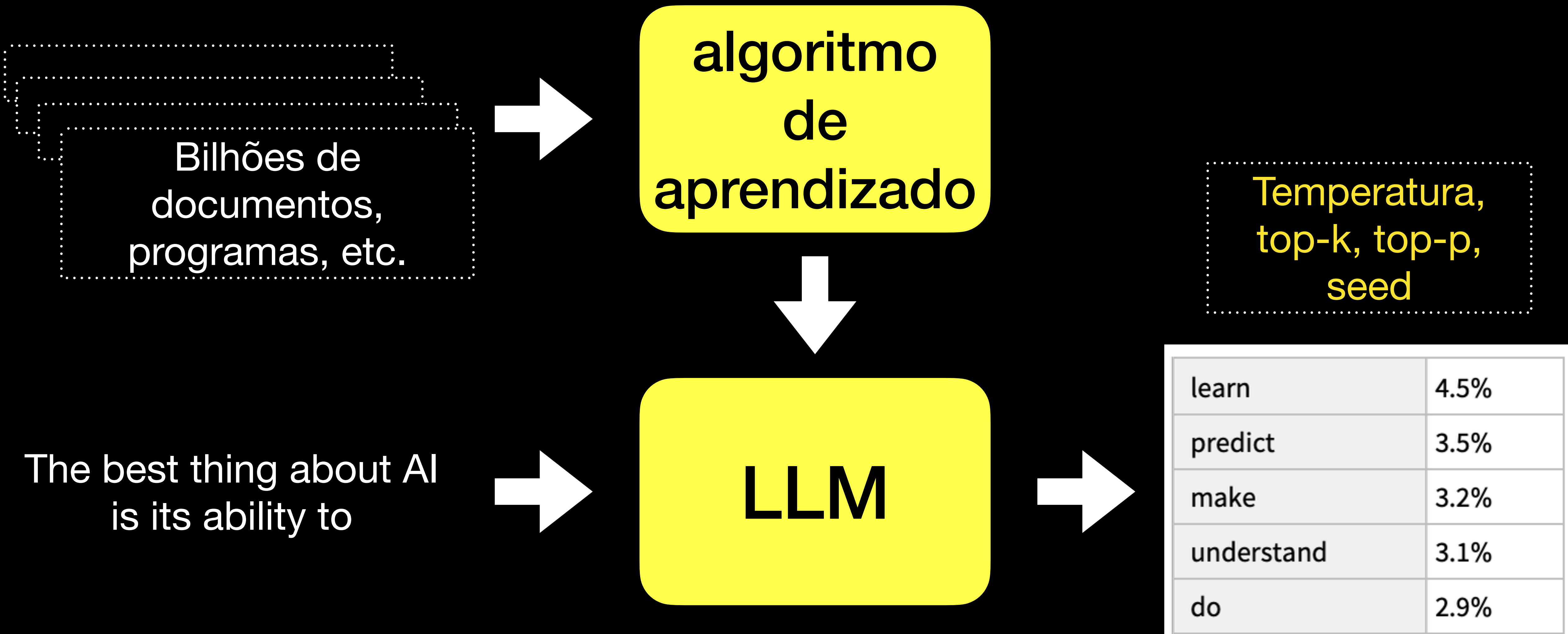
# LLM

simulação grosseira de um  
sábio com problemas  
cognitivos

Software Is Changing (Again), Andrej Karpathy  
<https://www.youtube.com/watch?v=LCEmiRjPEtQ>



# Aleatoriedade na escolha da próxima palavra



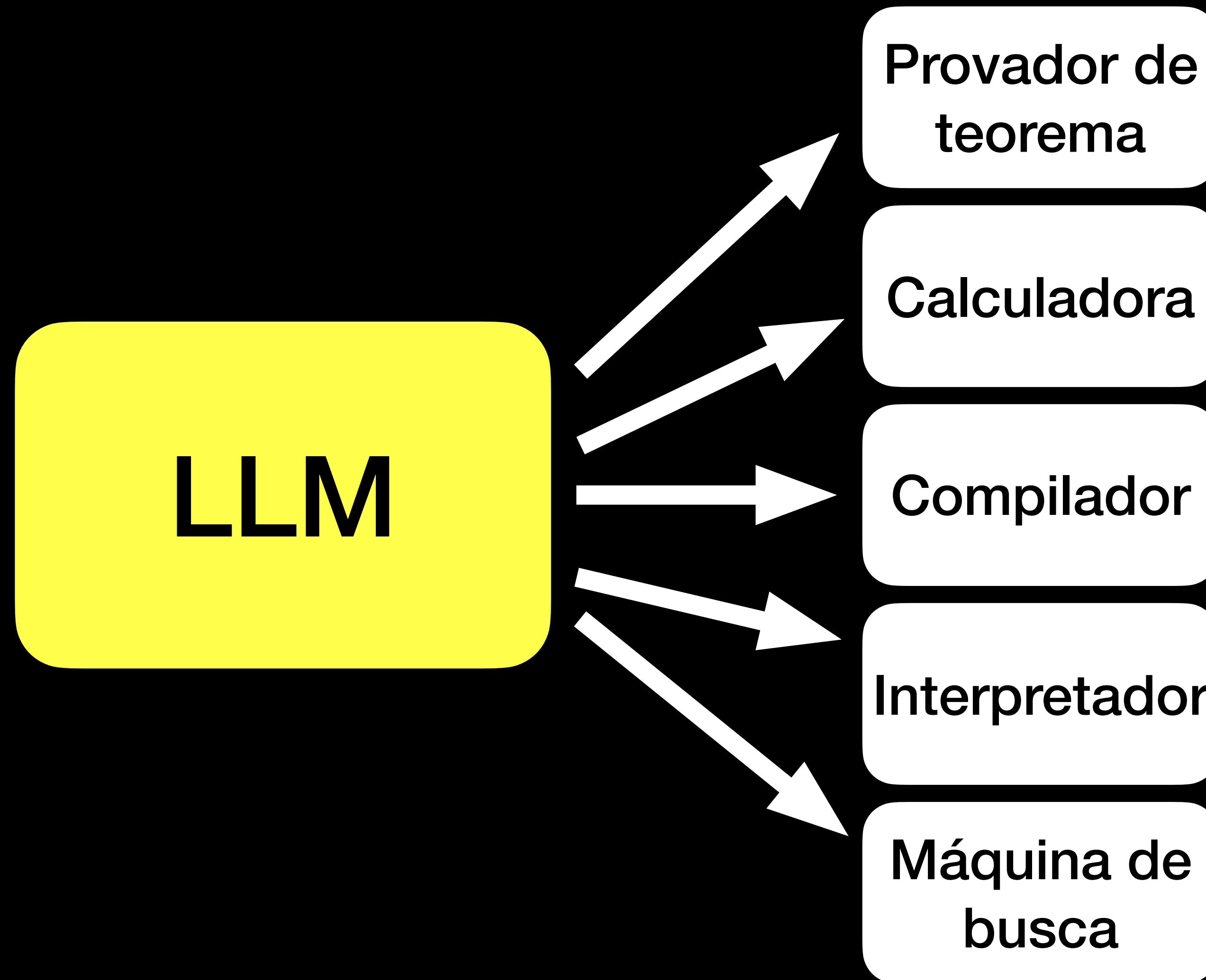
# Consequências

- Impressionante capacidade e conhecimento
- Impressionante incapacidade
- Alucinações
- Curta memória, limite de atenção
- Contexto local

# Agente (IA)

trabalha com um LLM para tentar  
identificar e corrigir suas  
limitações

# Augmented LLMs



# Mitos e Realidades

# Mito

*"vibe coding" é criar um SaaS  
para controle acadêmico sem  
cometer erros*

# Realidade

Adicione uma nova coluna à tabela de alunos, atualize o endpoint da API e o objeto de resposta para incluir esse novo campo, certifique-se de que a ordenação/filtragem por esse novo campo funcione como esperado, verifique se o store de alunos no frontend lida corretamente com todas as operações relacionadas a esse novo campo, garanta que as atualizações funcionem como esperado. Seguem arquivos relevantes:..."



# Especificações detalhadas

\* Keychain Storage

⌘1

~/s/Context

⌘2

node ~/s/Context

⌘3

~/s/Context

⌘4

~/Desktop

⌘5

+

```
> Please do the following:
- Please read Context/CLAUDE.md to understand how to build the app using XcodeBuildMCP
- For DXT manifests that contain a `user_config` key, add a "Configure" button that opens a new DXTUserConfigurationView
- The DXTUserConfigurationView should contain generated UI to specify each configuration option in user_config. As you read in the docs, the supported config types are string, number, boolean, directory, and file.
- Each user configuration option should have a label (using the `title` key) and a control that allows the user to modify it. Use native SwiftUI controls. Use the SwiftUI Form type build the overall form: https://developer.apple.com/documentation/swiftui/form
- There should be an info button for each configuration option that shows the `description`, whether multiple selections are allowed (via `multiple`), whether the key contains sensitive data (via `sensitive`) and any min/max limits if it is a number (via `min` and `max`)
- If a key is `required`, the UI should clearly indicate it
- For text fields where `sensitive` is true, use a secure text field
- If `multiple` is true for a file or directory selection, render a table and +/- buttons similar to how StdioConfigurationView shows Arguments and Environment
- If a `default` value is specified, make sure to perform substitutions for ${HOME}, ${DESKTOP}, and ${DOCUMENTS} as described in the spec and show the default value
- Inside the ContextCore library, Create a public Swift Codable & Sendable type called DXTUserConfigurationValues that can be used to store the substitutions. For non-sensitive data, the value can be stored directly in this type (as it will be written to the database). For sensitive data, we will generate a UUID that can be used to later reference the item from the macOS Keychain (see KeychainManager.swift for our Keychain access implementation). The DXTUserConfigurationValues should contain metadata that indicates whether a value needs to be looked up from the Keychain or not. While editing, the DXTUserConfigurationValues can contain the sensitive data, but before the configuration is saved (when data is written to the database), the sensitive data should be inserted into the Keychain and the sensitive value in DXTUserConfigurationValues should be modified to reference the keychain item. Note that DXTUserConfigurationValues should not know anything about the Keychain itself - it should just know whether the value is sensitive or not, and store the identifier to look the data up in the Keychain if it is.
- In Database/Schema.swift, add a new @Column for a nullable dxtUserConfig column that uses JSONRepresentation as in the other examples already there, except with the new DXTUserConfigurationValues (which conforms to Codable)
- In Database/Database.swift, add a new migration that adds the dxt_user_config column
- The DXTConfigurationView should indicate whether there are any required user_config parameters that the user has not filled in. If there are missing required parameters, it should not be possible to Save Changes/Add Server
- DXTTransport should take a DXTUserConfigurationValues in the initializer. If the DXTUserConfigurationValues contains any sensitive fields, it should throw an error (these should have been replaced with the real values prior to DXTTransport being initialized). DXTTransport should perform substitutions for user config values in accordance with the spec (there's a TODO for that right now). Add new tests to DXTTransportTests.
- MCPClientManager, when creating a DXTTransport, should read the DXTUserConfigurationValues from the database from the server, and perform any substitutions for sensitive values by reading the values from the Keychain. If there is a value missing in the Keychain, it should modify the DXTUserConfigurationValues to REMOVE the value for that key, and save it to the database, then throw an error indicating that the user needs to edit the server and specify a value for that missing field (which will cause the data to be written to the Keychain again).

Please make sure you understand the following implementation notes from the spec when you implement the user_config substitution in DXTTransport:

### Implementation Notes

- **Array Expansion**: When a configuration with `multiple: true` is used in `args`, each value is expanded as a separate argument. For example, if the user selects directories `/home/user/docs` and `/home/user/projects`, the args `["${user_config.allowed_directories}"]` becomes `["/home/user/docs", "/home/user/projects"]`.

• I'll implement the DXT user configuration support as requested. Let me start by reading the Context CLAUDE.md file to understand how to build the app.

• Read(Context/CLAUDE.md)
  L Read 25 lines (ctrl+r to expand)
```



Nem sempre linguagem natural é  
a melhor alternativa para  
especificações detalhadas

Modelos conceituais,  
especificações formais podem  
ser complementares

# Mito

ah, mas vai ser assim com o novo modelo M que vai ser lançado semana que vem (ou com RAG, ou com fine tuning, ou com agentes, ou com AGI...)

# Realidade

até com AGI, a descrição terá  
que ser precisa, de forma a evitar  
confusão entre o que você quer e  
o que o LLM pensa que você  
quer

# Realidade

se você quer pouco, tem poucas  
restrições, liga apenas para  
poucas decisões de projeto,  
sobreviverá com descrições  
menos precisas

# Mito

o experiente engenheiro de  
prompt do modelo M

# Realidade

o prompt que funciona bem na versão  $X$  do modelo  $M$  pode não funcionar tão bem na versão  $X+1$

"Notei que novas versões de LLMs podem produzir efeitos totalmente inesperados até mesmo em tarefas simples. Coloquei um código no M 2.5 e pedi para corrigir um bug seguindo passos que eu passei, e **ele mudou completamente a lógica, mesmo eu tendo falado repetidas vezes que era para atuar apenas na correção.** Ele ficou se justificando e recusando. Estava funcionando tudo normalmente, mas **o M tá mudando com frequência e dá para ver que muda o comportamento.**"

# Mito

o experiente engenheiro de  
contexto do modelo M



# Realidade

contexto é essencial para bons  
resultados, mas não resolve  
todos os problemas

# Mito

ah, mas se eu especificar bem  
vai funcionar sem problema!

# Realidade

você ainda vai se dar mal  
algumas vezes devido a  
alucinações, falta de contexto,  
etc.

Agentes, com e sem IA, são  
essenciais para funcionar, como  
oráculo para reduzir o impacto  
das alucinações

Mesmo sem IA, empresas usam agentes que rodam sozinhos, chamam linters, formatadores e outras ferramentas de desenvolvimento, comentam em PRs, submetem PRs com correções de bugs, geram testes para o código que você acabou de integrar, etc.

Com IA vai ter potencialmente  
mais erros...

ou pelo menos erros que  
difícilmente seriam cometidos  
por humanos

- Zé, você removeu todo o código do submódulo do carrinho de compras?
- Eu não, o meu agente de IA, mas estou trabalhando para resolver
- Nossa, você deu acesso de escrita para ele? Seu agente definitivamente não funciona!
- Talvez funcione! E se ele concluiu que o jeito mais eficiente de reduzir os bugs é eliminando todo o código? Tecnicamente ele não está errado...

# Agentic Program Repair from Test Failures at Scale: A Neuro-symbolic approach with static analysis and test execution feedback

Chandra Maddila, ... Rui Abreu, Nachiappan Nagappan, ..., Peter C. Rigby

In a three month period, 80% of the generated fixes were reviewed, of which 31.5% were landed (25.5% of the total number of generated fixes)



# Realidade

vai ser um problema para novas  
linguagens e APIs a menos que  
sejam muito similares às  
existentes

# Realidade

participação humana ainda é  
essencial durante o processo de  
desenvolvimento

**Mito**

é só perda de tempo!

# Realidade

"Aquela tarefa que eu fazia em 10-15 minutos agora faço em 2 minutos, com menos estresse (não tenho que lembrar as APIs) e tédio..."

Substituto  
do stack  
overflow

Gerador de  
código  
repetitivo

Interpretador de  
*linguagem natural*  
para programação  
passo-a-passo

risco

Explicar  
código

Realizar  
mudanças  
repetitivas

Criar APIs,  
abstrações, etc.

(pouca evidência científica)

**Agentes e humanos são  
essenciais para esses resultados**

# Mas cuidado com erro acumulado...



se o segundo propaga o erro do primeiro

# Mito

não preciso mais programar  
(visão 2022)!



# Realidade

a não ser para domínios muito  
específicos e repetitivos, precisa  
programar e entender o código

Na dúvida, ver palestra "O fim da programação, de novo!"



# Realidade

é preciso **revisar, verificar**  
(corretude, eficiência, segurança,  
etc.), **complementar, refinar,**  
**modularizar, refatorar, integrar, e**  
**alterar** o código gerado

# Realidade

principalmente, é preciso  
**criar** abstrações do domínio,  
arquitetura, estrutura modular

# Mito

não é mais necessário aprender  
a programar ou um curso de  
computação!

# Realidade

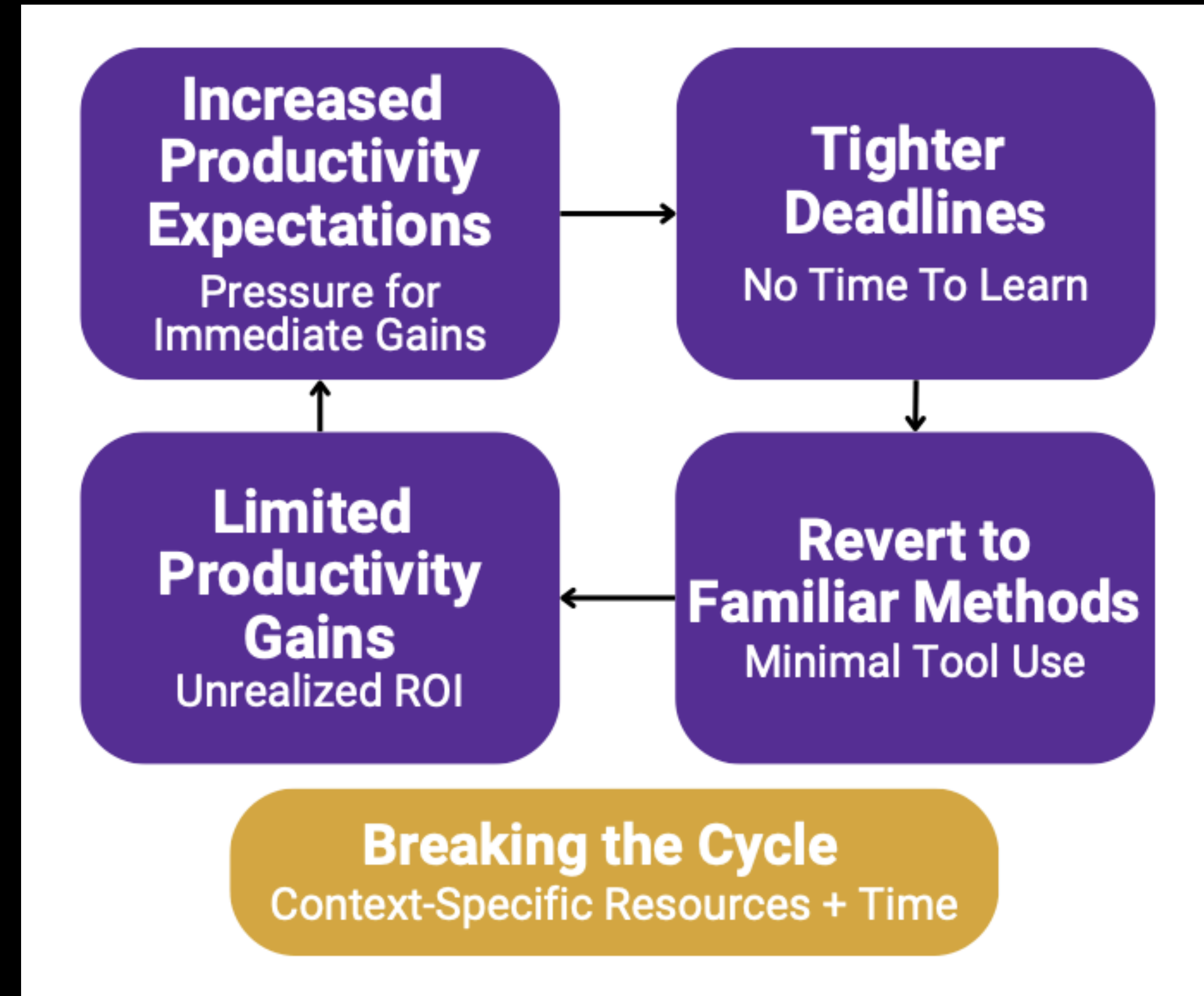
necessidade de entender  
conceitos básicos, vocabulário  
da área (padrões, etc.) para  
descrever precisamente o que se  
deseja

# Mito

não preciso estudar/entender  
sobre LLMs para usá-los na  
prática

# Realidade

the productive  
pressure paradox!



<https://arxiv.org/pdf/2507.21280>



# Realidade

tentativa e erro para cada versão do modelo, aceitar a frustração da perda de tempo eventual, e o fato de que não há teoria mais geral a ser aprendida

# Mito

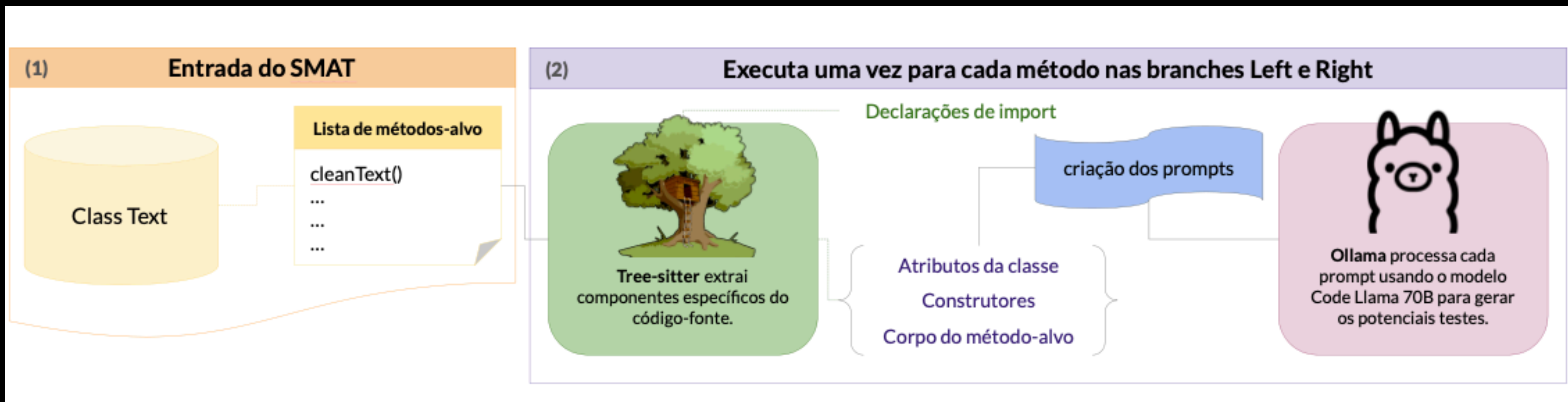
não precisa pesquisa sobre o  
impacto de LLMs no processo  
de desenvolvimento de sw

# Realidade

os resultados podem ser  
surpreendentes!

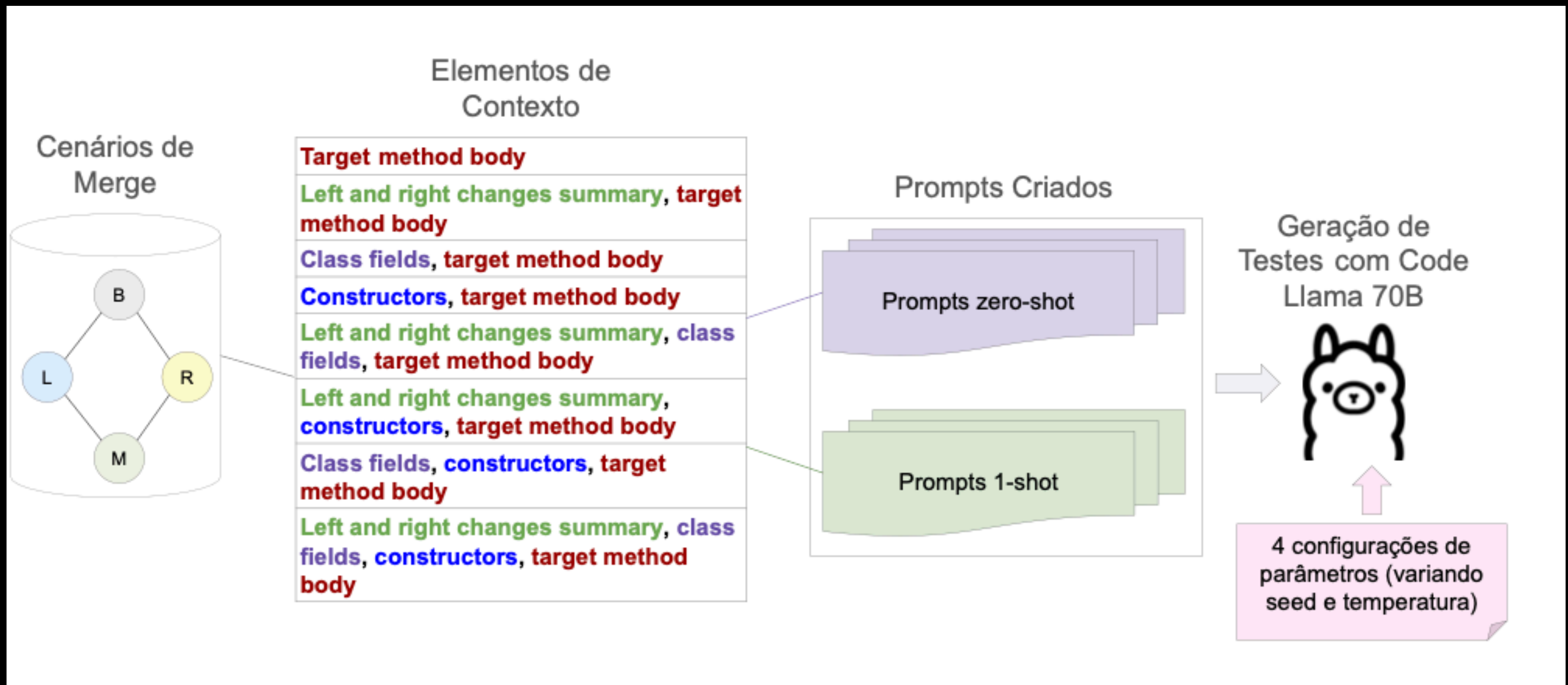
# Detecção de conflitos semânticos de código

## geração de testes de unidade com LLMs



<https://arxiv.org/abs/2507.06762>

# Analizando várias soluções



# Resultados inferiores a outras ferramentas com maior (~20x) custo computacional

Ferramenta de Geração de Testes	Conflitos Detectados
Differential EvoSuite	6
EvoSuite	5 [1]
<b>Code Llama 70B (união de todas as execuções)</b>	5 (1*)
<b>Code Llama 70B (zero-shot, temperatura 0)</b>	3 (1*)
Randoop	2
Randoop Clean	2 [1]

**Substancial diferença de  
comportamento dependendo do  
sistema (amostra)**

**Com sistemas simples, aumento de  
200-600% na taxa de testes gerados  
que compilam**



Uma visão mais abrangente sobre  
o processo de desenvolvimento  
de software

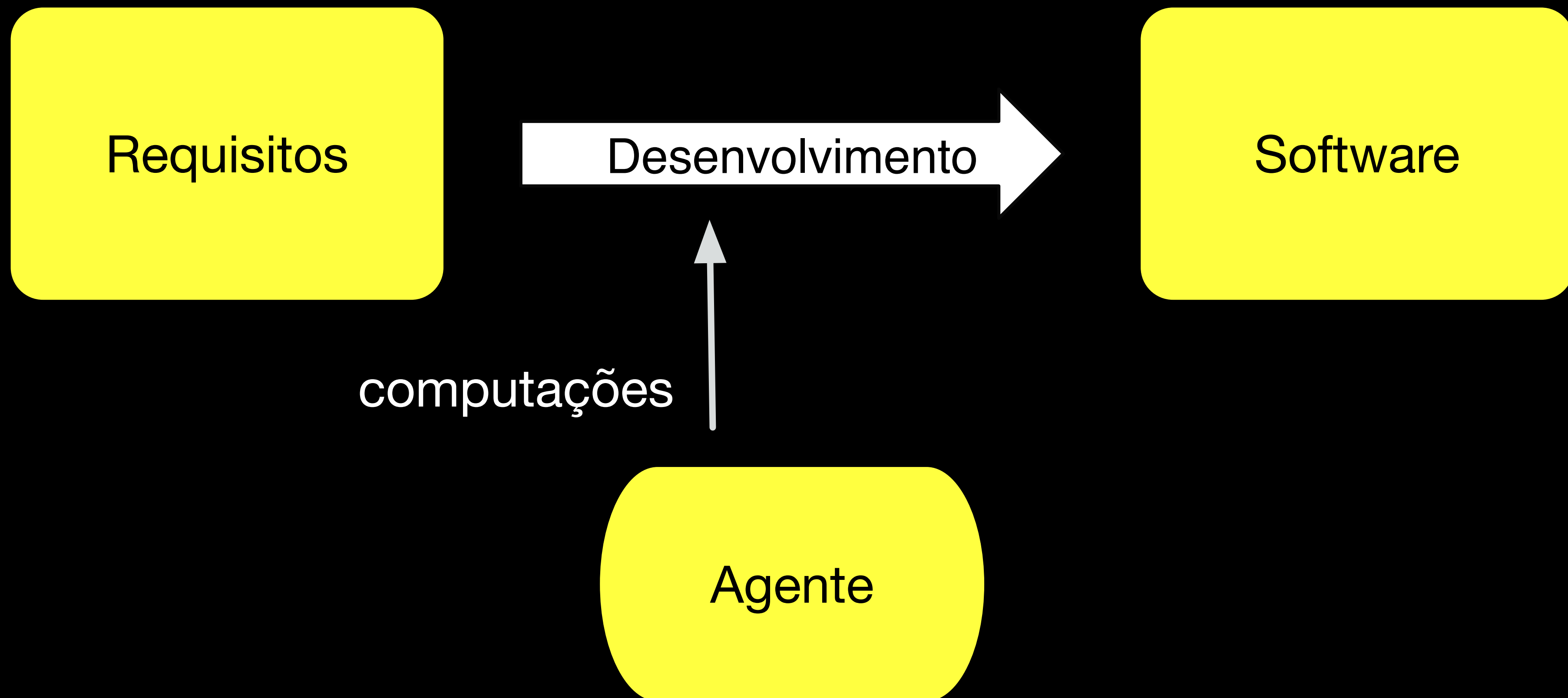


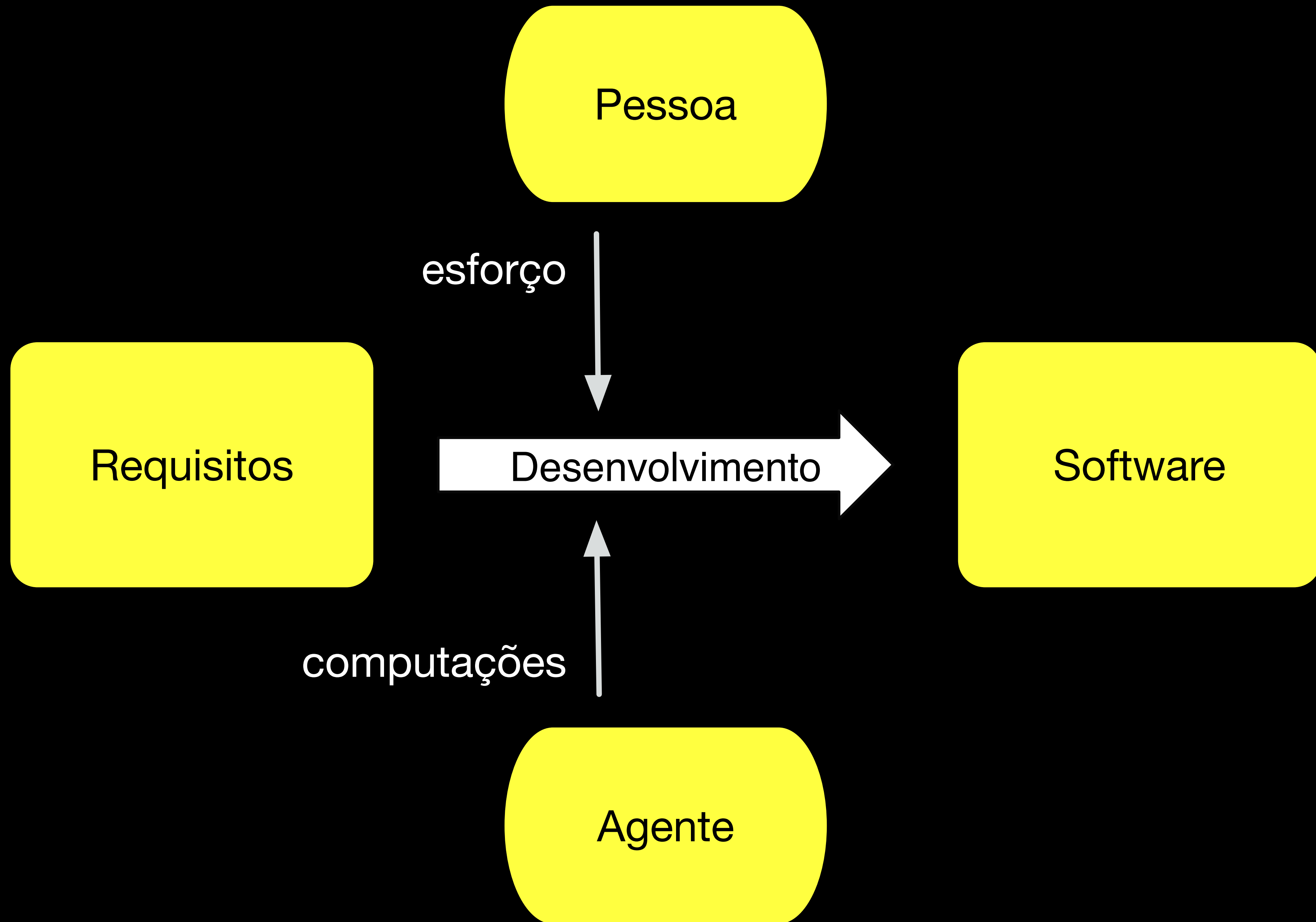
Requisitos

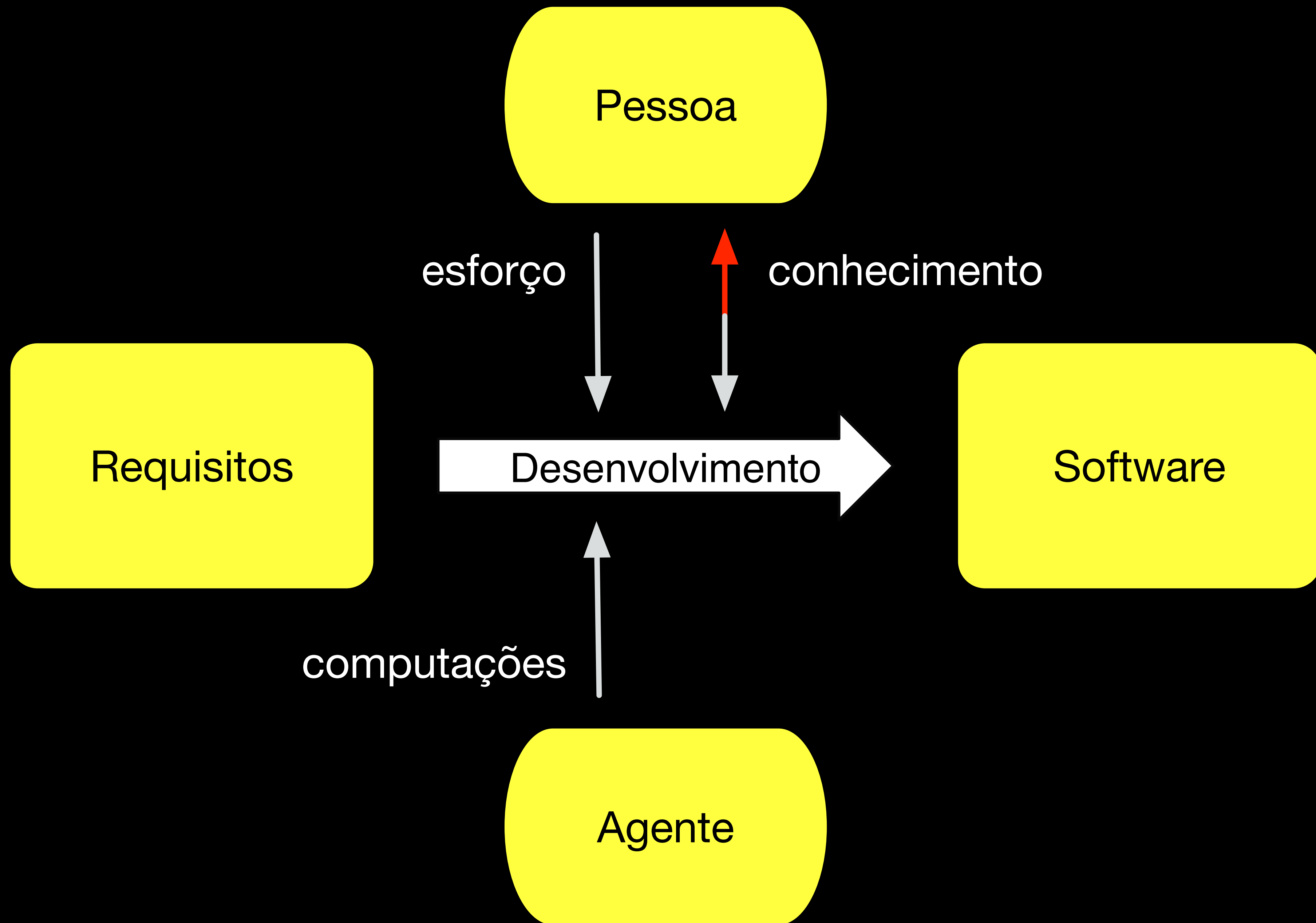
Desenvolvimento

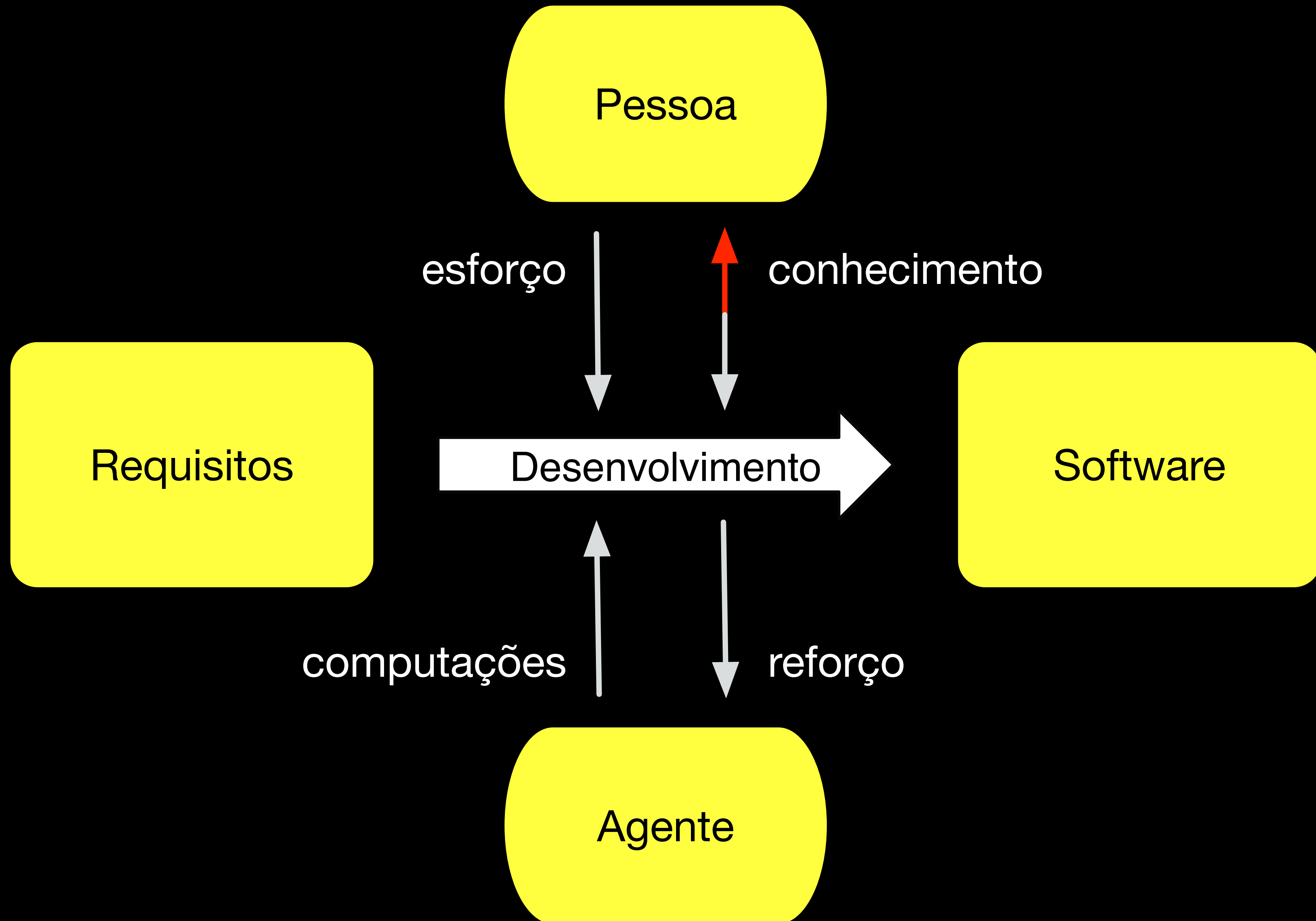
Software

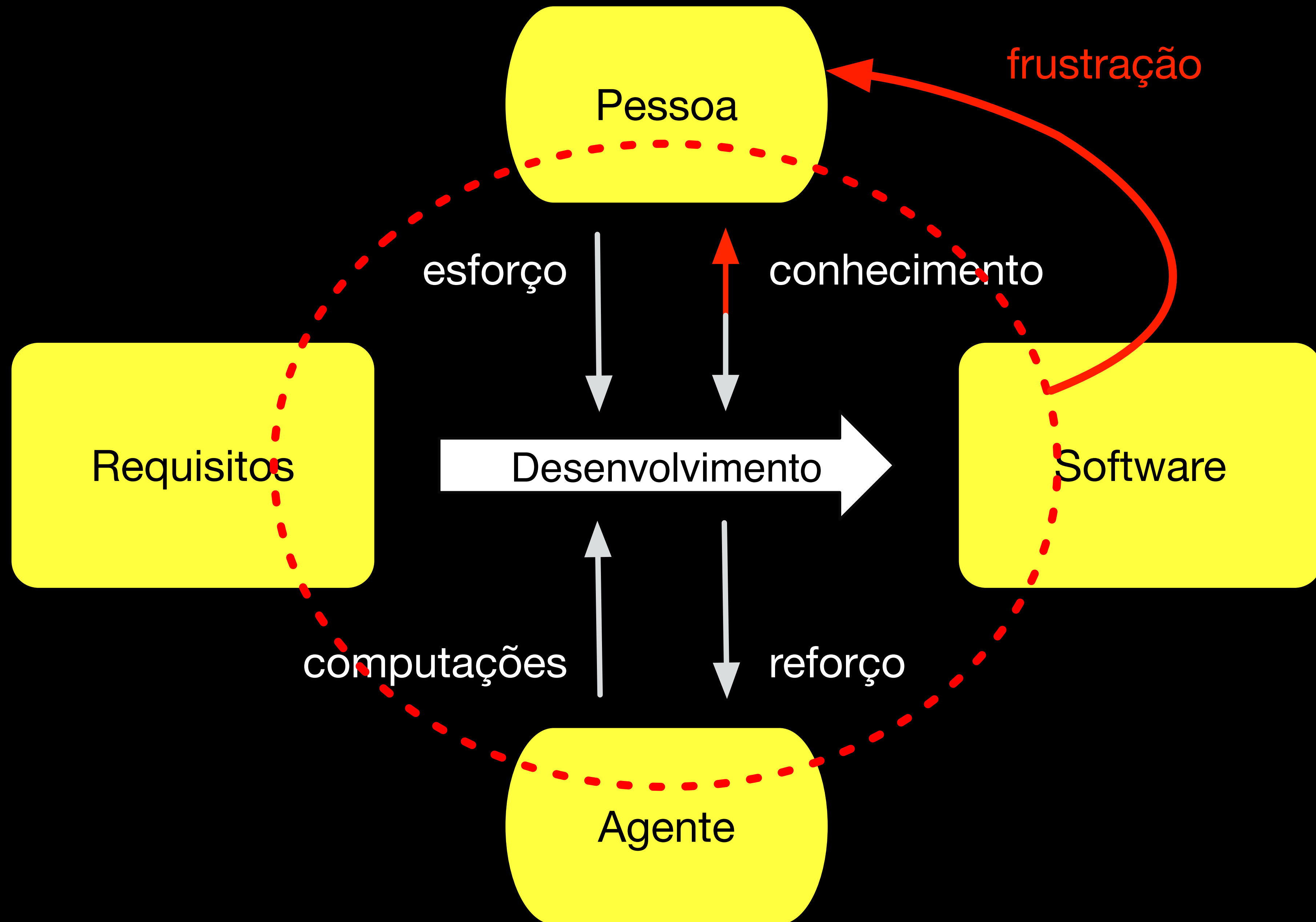












Em que tarefas usar agentes?

$$A = h(e_c, e_{pr}^{-1}, c_{in}^{-1}, c_{out}^{-1}, f^{-1})$$



# Aproximação dependente de...

- Modelo, seus parâmetros e tamanho de contexto
- Linguagem de programação e APIs utilizadas para construção do sistemas
- Tamanho do sistema

Obrigado!

# Mitos e Realidades no Desenvolvimento de Software com IA

Paulo Borba  
Centro de Informática  
Universidade Federal de Pernambuco



[pauloborba.cin.ufpe.br](http://pauloborba.cin.ufpe.br)