

```

\documentclass[12pt,a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage[catalan]{babel}
\usepackage[margin=2.5cm]{geometry}
\usepackage{inconsolata} % Font monoespaïada agradable
\usepackage{hyperref}
\hypersetup{colorlinks=true, linkcolor=MidnightBlue, urlcolor=MidnightBlue}
\usepackage{fancyvrb} % per mostrar codi
\title{PROJECTE ASSIMOV: Un Manifest per a Educadors en l'Era dels Transformers}
\author{Dídac Valenciano Gener}
\date{maig del 2025}
\begin{document}
\section*{L'any 2025 és —i serà— l'any ChatGPT}.

```

A l'Europa ibèrica i mediterrània dins la UE, amb un subconscient meca latent que avui encara es nota, *Mazinger Z* va deixar una profunda empremta cultural, emocional i en diversos altres àmbits que explica alguns comportaments compartits de la *Generació X*. Si tens entre 45 i 60 anys, ja has pensat en “*¡Puños fuera!*”, has recordat com vas sobreviure al *Blandiblú*, als *Clicks* de *Famobil*, i a veure dibuixos violents sense convertir-te en un psicòpata.

Als Estats Units, *Hasbro* inicia el relleu generacional amb la saga *Transformers*, que culmina culturalment l'any 2007 quan, just arrencar el metratge, els *Decepticons* ataquen una base militar nord-americana al desert de *Qatar*. Mentrestant, *Megan Fox* i *Shia LaBeouf* ens deixen sense alè. És en aquest moment quan una nova imatge s'enganxa al subconscient d'una altra generació: un petit *Decepticon* —un *transformer*, però dels dolents—, amagat entre les pertinences dels protagonistes, absorbeix metall de l'entorn per modificar-se, estirant les seves formes, perllongant-se i adaptant-se fins a assolir el seu objectiu.

Deu anys després, fruit d'un llarg recorregut d'investigació per part de múltiples entitats acadèmiques i privades, en paral·lel al món de l'entreteniment, *Google* presenta públicament l'any 2017 una tecnologia algorítmica que es replica, s'estira i amplia la seva pròpia estructura, modificant-se i perllongant-se fins que, de manera purament probabilística, aconsegueix assolir el seu objectiu. Quina sorpresa —i quina coincidència— que aquesta tecnologia, desconeguda fins aleshores, rebés el nom de: *transformers*.

“*Attention is All You Need*”, publicat per *Google* el 2017, va crear rebombori —especialment a *Twitter* (o *Grok*, o com es digui aquesta setmana). Però no va ser fins que *OpenAI* va agafar el relleu que ens van col·locar al davant, literalment, un petit *transformer*. Un que, cada vegada que li parlem, *tokenitza* el nostre llenguatge, i com aquell *Decepticon* al desert de *Qatar*, s'estira, s'adapta i reorganitza la seva forma per generar, de manera purament probabilística, una resposta que optimitzi la situació que té al davant.

I la gràcia, amics meus, és que ho simula tan bé, que si entenem per *intel·ligència artificial* una simulació funcional d'intel·ligència... bé, potser hi ha altres implementacions, però aquesta —definitivament— n'és una.

El que ve tot seguit és una altra implementació. Una que, com els algoritmes d'*OpenAI*, s'estira, s'adapta i reorganitza la seva pròpia estructura per generar, de manera purament probabilística, una resposta òptima a la situació que té al davant: crear de manera eficient contingut didàctic, un article, un llibre. Tot plegat perquè nosaltres, *professors*, puguem disposar de les nostres pròpies armes per a la batalla que ens espera.

```

1 # Phase 0: Interpret prompt and assign roles with
   grammarnaut + llm_router
2
3 # Phase 1: Generate structural skeleton with yaml_generator
   using Claude or o3 or ...
4
5 # Phase 2: Expand each section with role-assigned LLMs (GPT
   -4o, MythoMax, Lit-6B...) using cached YAMLs
6
7 # Phase 3: Review narrative coherence, argument and tone
   with o3 or Claude using section context
8
9 # Phase 4: Compile formatted output into .tex with writer or
   translator
10
11 # Phase 5 (optional): Final stylistic polish by purist LLM
   or human reviewer
12
13 #####
14 # AI-enhanced Content Generation Pipeline (Assimov)
15
16 import openai
17 from pathlib import Path
18 import time
19
20 def phase_0_prompt_intake():
21     """Receive user prompt and determine functional
       roles + LLMs."""
22     prompt = user_input()
23     roles = assign_roles(prompt) # via gramaneute.py ->
       llm_router.py
24     llms = select_models(roles, config="config.yaml")
25     return plan(roles, llms)
26
27 def phase_1_generate_structure(plan):
28     """Create YAML skeleton per section with titles,
       themes, targets."""
29     yamls = []
30     for section in plan.sections:
31         yamls.append(generate_yaml(section)) # via
       yaml_generator.py or real LLM
32     return yamls
33
34 def phase_2_expand_sections(yamls):
35     """Expand each section via assigned LLM, using
       cached context."""
36     content = []
37     for yaml in yamls:

```

```

38         llm = yaml.assigned_model
39         section_text = expand_from_yaml(yaml, llm=
40             llm, use_cache=True)
41         content.append(section_text)
42     return content
43
44 def phase_3_refine_coherence(content):
45     """Polish narrative flow, argument structure, and
46     tonal coherence."""
47     polished = []
48     for section in content:
49         coherent = enforce_coherence(section) #
50         coherence supervisor
51         refined = polish_argument(coherent)
52         toned = adjust_tone(refined)
53         polished.append(toned)
54     return polished
55
56 def phase_4_compile_tex(polished):
57     """Format polished content into a .tex document with
58     structure."""
59     doc = initialize_tex()
60     for section in polished:
61         doc.append(format_section(section)) # via
62         writer.py or translator.py
63     return doc
64
65 def phase_5_final_review(tex_document):
66     """Optional: stylistic and poetic pass by a purist
67     LLM or human."""
68     reviewed = manual_review(tex_document)
69     return reviewed
70
71 # Main pipeline execution
72
73 if __name__ == "__main__":
74     plan = phase_0_prompt_intake()
75     yamls = phase_1_generate_structure(plan)
76     raw_content = phase_2_expand_sections(yamls)
77     refined = phase_3_refine_coherence(raw_content)
78     tex = phase_4_compile_tex(refined)
79     final_output = phase_5_final_review(tex)
80     save(final_output, "output/document.tex")

```

Aquest text està llicenciat sota una Llicència Creative Commons Reconeixement – No Comercial – Compartir Igual 4.0 Internacional (CC BY-NC-SA 4.0).

<https://creativecommons.org/licenses/by-nc-sa/4.0/deed.ca>