The Grammarnaut

A narrative and functional figure for orchestrating language models

Dídac Valenciano Gener

May 2025

Abstract

This text, originally conceived as a functional *prompt* to generate the structure of a Python script, evolved into a narrative figure that defines and humanizes the orchestration process of language models within the **Assimov Project**. The **Grammarnaut**, or *good shepherd (of LLMs)*, symbolizes the interface between human intention and the algorithmic distribution of functional roles in a multi-model system.

The Grammarnaut, or "good shepherd" (of LLMs) – Prompt to indicate the structure of a Python script

A "good shepherd" listens to the user's prompt and sends this prompt to an LLM, which suggests a definition of the functional roles and selects the LLMs for each role, based on the original prompt.

Which models fit best for each role is something the good shepherd decides depending on the user's physical and financial possibilities.

The good shepherd shows the user a summary of how the task requested in the original prompt will be carried out. And if the user hasn't indicated it, the good shepherd also provides an estimate of how many top-level sections there will likely be, and how many pages per section and in total, according to what the good shepherd considers appropriate, as well as what has been negotiated with the various functional-role LLMs.

The user may make some adjustments to suit their taste and indicate to "proceed" with the process, or stop if that becomes necessary.

In the next step, the good shepherd contacts an LLM again with the previously gathered information, and the LLM returns some general YAMLs describing the text's setting, target, and narrative style, as well as one YAML for each top-level section, detailing the title, full skeletal structure (with all headings), topics, key phrases or ideas, specific skills (tables, graphs, pies, DALL · E drawings, sex scenes, ...) and the intended objective — all provided by the various functional roles previously mentioned.

The good shepherd shows them to the user. The good shepherd always asks the user for the target language of the text if this hasn't been done yet, to avoid surprises.

Once the user has reviewed the whole structure and made any adjustments needed to better match their expectations, the good shepherd requests the API keys for the various platforms where the "good LLMs" reside, along with an approximation of the cost and duration of the process.

If the keys had already been entered beforehand, the shepherd simply asks for permission and informs the user of the approximate cost. Neither the user nor the good shepherd want a local system, because Mallorca is already hot enough and electricity is expensive here.

The good shepherd suggests that using caches may increase the cost but improve quality, and that parallel execution is possible without too much quality loss — and therefore faster, though not better.

The user chooses how they prefer the task to be "cooked", based on these options of time, added costs, efficiency, and refinement of results.

The good shepherd sends the corresponding YAMLs to the caches, with the user's approval, and is in charge of orchestrating concurrency and/or sequencing between the LLMs and writing the .tex file with the different content fragments, which are ultimately delivered by a "good translator" and/or "purist stylist" in case a final touch is needed.

At the end, the user —made of flesh and bone— receives from the hand of the good shepherd —made of Python code— a .tex document. They shake hands and part ways.
