Group 2 Project Phase 1 Introduction

Jay Casey & Emily Nover

February 6, 2026

I. Introduction

Working with LLM's can greatly benefit the speed and productivity of software engineers. Programs now have the ability to directly implement AI solutions into their code without needing to write thousands of lines to achieve the same result. The current standard for this integration is prompt engineering. This process involves connecting to an LLM and writing specific instructions in the code to explain exactly what should be done. This is a lengthy process and leaves much room for error when it comes to clarity. As prompt engineering is still a newer topic, many engineers are still illiterate when it comes to speaking the LLM's language. ByLLM is a solution to this problem. By using Meaning-Typed Programming (MTP), byLLM completely removes the need to write these lengthy prompts. MTP uses minimal written code in order to interpret what functionality the user wants the LLM to contribute to the program [1]. This takes the responsibility of crafting the prompt out of the user's hands, but leaves room for another large issue, misinterpretation. Since the user does not directly inform the LLM of the action it wants to happen, there may be preventable errors. This project seeks to understand the accuracy, reliability, and maintenance needed when substituting traditional methods for byLLM's MTP driven approach.

II. Problem Statement

With MTP, byLLM simplifies the process of utilizing AI-assisted integrations in programming tasks down to simply using the by operator with a <model> name. Within that process, byLLM reduces overhead by delegating a variety of tasks to an user-specified Large

Language Model (LLM) provider such as OpenAI, Google, Anthropic, or even a local model, disguising the complexity of using AI integrations in coding tasks. Code is generated based on the user's intent without the excessive prompting that comes with traditional prompt-based programming. Despite the promising nature of MTP and byLLM, there is an omission in the research with the capability of these tools to use the "by" operator to detect and repair dirty data automatically compared to traditional data cleaning methods such as regex-based scripting [1]. Traditional approaches in data quality tasks can be prone to error due to the natural restrained nature of the data cleaning methods to specific domains without consideration for the ability to adapt across domains, datasets, or prevent the influence of real-world noise. MTP and byLLM might be a potential solution to this, but there are challenges as MTP is sensitive to poor coding practices, struggling with cryptic and bad namings. Many datasets are considered to be "dirty" due to these problems, and it can appear across similar datasets (e.g., government datasets); byLLM also relies on API token usage and/or local models for access to AI, the number of API calls might be less than a funded project due to billing costs, limiting this project to the free tiers of API access. Regardless, this project will aim to address the omission of data quality tasks by developing a large-scale application, a data cleaning pipeline, to be tested with known "dirty" datasets in various domains in three ways: 1) using byLLM and its "by" operator, 2) using traditional prompt-based programming, and 3) using traditional cleaning approaches (e.g., regex) to create manually cleaned data, to find out if MTP and its tools are capable of performing as well or better than traditional methods.

References:

[1] Jayanaka L. Dantanarayana, Yiping Kang, Kugesan Sivasothynathan, Christopher Clarke, Baichuan Li, Savini Kashmira, Krisztian Flautner, Lingjia Tang, and Jason Mars. 2025. MTP: A Meaning-Typed Language Abstraction for AI-Integrated Programming. Proc. ACM Program. Lang. 9, OOPSLA2, Article 314 (October 2025), 29 pages. https://doi.org/10.1145/3763092