Domain Driven Design with LLMs

An informal comparative study

Ana Paula Mota, Natali Rocha,
João Paulo Mota, Gabriel Dadalto
MAC5913 - IME USP
motaanapaulasantos@gmail.com
natalirocha76@gmail.com
joaopauloster@gmail.com
gabrieldadalto@usp.br

Abstract

Software development begins long before the first line of code is written. Software engineers must interpret business rules, often expressed in natural language during meetings or in documentation, and translate them into a software model that forms the basis of the system architecture. This phase can be time-consuming, as it depends not only on gathering requirements from business experts but also on the engineers' ability to accurately translate them into a technical model—a process heavily influenced by their experience.

In startups and small companies, teams are often limited in size and may lack expertise in this critical phase. We propose the question if Large Language Models (LLMs) and Domain Driven Design principles can be leveraged to build solid and useful abstractions, significantly accelerating the modeling phase and improving quality of the software delivered.

Using identical business requirements, we contrast the models generated by each LLM with those produced by an expert human modeler. We conclude with insights into the use of LLMs as support tools for software modeling, the current state of automatic model generation, and the potential of these tools to enhance software development processes.

Introduction

0.1 What is DDD and why does it matter?

Domain-Driven Design (DDD) refers to a set of techniques and practices that assist software teams in constructing models based on concepts used by domain experts—those who deal daily with the specific problems the software aims to address. This approach to modeling offers several advantages over other software development paradigms, as highlighted in the literature.

One key benefit is the early establishment of a common language between software developers and stakeholders, which facilitates communication and improves the maintainability of the application. This is because the terminology used in a specific domain is unlikely to change drastically in a short period. Another advantage is the ease with which a model built using DDD can be translated into software that emphasizes separation of concerns, high cohesion, and low coupling—qualities widely recognized as essential for robust software design.

In his book Domain-Driven Design, Eric Evans[6] emphasizes that the most significant complexity in many applications is not technical but lies in the domain itself. If this domain complexity is not adequately addressed in the design, even well-conceived infrastructural technology will fall short.

0.2 DDD and AI: Enter LLMs

One of the core principles of DDD is the creation of a ubiquitous language - a common vocabulary shared by business experts and engineering teams. This vocabulary is expressed in natural language, making it a natural fit for tools which process human language.

Large Language Models (LLMs) have emerged as powerful tools for natural language processing, capable of understanding human language. This capability positions LLMs as potential assistants in the domain model process. For example, LLMs can generate models or diagrams based on conversations with domain experts or user stories, reducing the time and effort required to transform domain knowledge into technical models.

Model	Parameters (Billions)	Context Window (Tokens)
OpenAI ChatGPT		
GPT-1 (2018)	0.12	Unknown
GPT-3 (2020)	175	2048
GPT-3.5 (2022)	Not disclosed	4096
GPT-4 (2023)	1800	32,000
GPT-4o (2024)	Not disclosed	128,000
Meta LLaMA		
LLaMA 1 (2023)	7, 13, 65	Unknown
LLaMA 2 (2023)	7, 13, 70	4,000
LLaMA 3 (2024)	Up to 405	128,000
LLaMA 3.1 (2024)	405	128,000
LLaMA 3.2 (2024)	1, 3, 11, 90 (Vision models)	128,000

Table 1: Evolution of ChatGPT and LLaMA models.[9][3][7]

AI Generated Domains: An Experiment

0.3 Can AI-generated domain models match the quality of those designed by human experts?

We evaluated the outputs based on the following criteria when compared to human expert output:

- Accuracy: the capture of domain's rules, relationships
- Completeness: include all necessary elements without omitting critical details
- Clarity: model is easy to understand

To explore this question, we selected 3 LLMs (Claude Sonnet[4], DeepSeek[2] and ChatGPT[8]) tools and provided them with the same input: simple description of user stories on how to create a basic bank application.

Using a zero shot approach - where no examples or tags are given - we tasked the LLMs with generating domain model specifications, including bounded context, aggregations, entities and value objects. We then compare those results with the model generated by a human senior software engineer.

1 Results

Table 1 displays scores given to each model following the simple evaluation criteria of how well do we think the values generated by the LLMs match those of the human expert.

The final score represents the average of performance across four categories: bounded contexts, aggregates, entities, and value objects. Each category was evaluated by comparing it to a human expert's analysis, with ratings ranging from 1 (low) to 3 (high). These individual ratings were then averaged to produce the final score.

Table 2 displays the scores given to each category.

	Claude Sonnet	DeepSeek	Chat GPT
Accuracy	🟢	🟢	🔴
Completeness	🟢	🟢	🟡
Clarity	🟢	🟢	🟢

Table 2: Average score for each LLM

	Claude Sonnet	DeepSeek	Chat GPT
Context	🟢	🟢	🔴
Aggregate	🟢	🟢	🟡
Entities	🟢	🟢	🟢
Value Objects	🟢	🟢	🟢

Table 3: score for each LLM in 4 categories

2 Conclusion

Although domain modeling is not an exact science, the results generated by LLMs demonstrate potential, often coming closer to human-level outputs.

Through the experimentation, it was observed that LLMs can assist in the domain modeling process by leveraging their natural language interpretation capabilities. However, their effectiveness heavily depends on the clarity and completeness of the input provided. Unlike humans, LLMs lack the ability to enrich the input with real-word context or domain-specific insights.

A key limitation of LLMs is their strict interpretation of the input without the capacity to question or validate it based on deeper knowledge of real-world systems, such as how banking systems operate. On the other hand, humans can draw on their experience and contextual understanding to refine and improve the model.

Despite these limitations, LLMs can still provide value in DDD processes and practices, particularly in environments where resources are limited, such as startups or small teams. They can generate preliminary specifications and diagrams that serve as a starting point for event storming sessions with business experts. These initial artifacts can accelerate the process of business understanding and the development of ubiquitous language, ultimately saving time and effort in the early stages of domain modeling.

In conclusion, while LLMs are not a replacement for human expertise, they are a good tool that can complement human efforts, especially in resource-constrained environments providing a foundation for collaboration between technical teams and business experts. process of business understanding and the build of ubiquitous language.

A Appendix

The full prompt that was given to the LLMs for performing this study can be found in the full article.

References

[1]

[2] Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. 2025.

[3] CommunityOpenAI. Chatgpt - token context window information, 2024.

[4] Anthropic Company. Anthropic claude llm tool, 2025.

[5] Mckinsey Consulting. The economic potential of generative ai: The next productivity frontier, 2023.

[6] Eric Evans. *Domain-Driven Design - Tackling complexity in the heart of software*. Addison-Wesley Professional, 1th edition, 2003.

[7] LifeWire. Llama 2 vs llama 3 - what is new, 2024.

[8] OpenAI. Chatgpt llm tool, 2025.

[9] TheVerge. Meta's llama first ai vision model, 2024.