Kaggle "Drawing with LLMs" Competition

Systems Analysis

Edward Julian García Gaitán Jaider Camilo Carvajal Marin
Nelson David Posso Suárez

Course: Systems Analysis and Design
Professor: Eng. Carlos Andrés Sierra, M.Sc.

Systems Engineering Program
Faculty of Engineering
Universidad Distrital Francisco José de Caldas

May 2025

1. Introduction & The Kaggle Challenge

- Kaggle's **"Drawing with LLMs"** competition challenges us to generate SVG images from text (prompts) using Large Language Models (LLMs).
- It involves the confluence of: Natural Language Understanding (NLU), automatic code generation, and graphics synthesis.
- It presents notable **systemic complexity** and is considered an open system.
- Previous analyses (Workshop 1) revealed critical sensitivities and the potential manifestation of chaotic behaviors ("butterfly effect" in prompts).

Central Problem:

Designing a robust system to translate prompts into valid and coherent SVGs, managing LLM variability.

2. Key Objectives & Core Requirements

The system design aims to achieve the following macro-objectives:

- **Functional Effectiveness:** Semantically coherent and syntactically valid SVGs.
- **Robustness and Reliability:** Predictable behavior and proper error/anomaly handling.
- **Operational Efficiency:** Meet performance constraints (latency, capacity).
- **Scalability and Maintainability:** Facilitate system adaptation and evolution.

2. Key Objectives & Core Requirements

The system design aims to achieve the following macro-objectives:

- **Functional Effectiveness:** Semantically coherent and syntactically valid SVGs.
- **Robustness and Reliability:** Predictable behavior and proper error/anomaly handling.
- **Operational Efficiency:** Meet performance constraints (latency, capacity).
- **Scalability and Maintainability:** Facilitate system adaptation and evolution.

Main Requirements (Summary):

- **Crucial FRs:** Prompt ingestion and parsing, advanced semantic interpretation, valid SVG generation, flexible configuration, results persistence.
- **Key NFRs:** Performance (latency ¡1s is an ambitious goal), reliability (100% valid SVGs), scalability (up to 10k prompts), modular maintainability.

3. Proposed System Architecture

A **modular and layered architecture** with a well-defined data processing flow is proposed, promoting separation of concerns and low coupling.

3. Proposed System Architecture

A **modular and layered architecture** with a well-defined data processing flow is proposed, promoting separation of concerns and low coupling. **Applied Design Principles:**

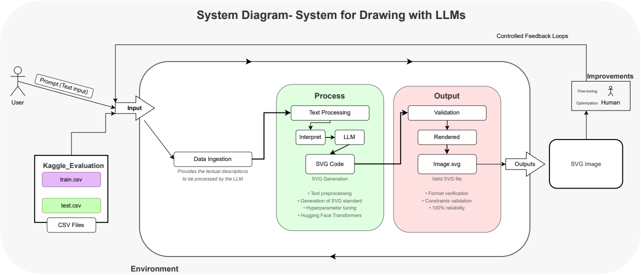
- Modularity, Separation of Concerns (SoC), Interface Abstraction, High Cohesion, Low Coupling, Dataflow-Oriented Design, Testability.

3. Proposed System Architecture

A **modular and layered architecture** with a well-defined data processing flow is proposed, promoting separation of concerns and low coupling. **Applied Design Principles:**

- Modularity, Separation of Concerns (SoC), Interface Abstraction, High Cohesion, Low Coupling, Dataflow-Oriented Design, Testability.

Conceptual System Diagram:



4. Key Strategies (Part 1): Prompts and LLM

To manage system sensitivity and mitigate chaotic behaviors:

4. Key Strategies (Part 1): Prompts and LLM

To manage system sensitivity and mitigate chaotic behaviors: **1. Robust and Adaptive Prompt Engineering (Module M2):**

- *Normalization and Standardization:* Cleaning prompts.
- *Prompt Templating:* Guiding the LLM.
- *Dynamic Few-Shot Learning Techniques:* For complex prompts.

4. Key Strategies (Part 1): Prompts and LLM

To manage system sensitivity and mitigate chaotic behaviors: **1. Robust and Adaptive Prompt Engineering (Module M2):**

- *Normalization and Standardization:* Cleaning prompts.
- *Prompt Templating:* Guiding the LLM.
- *Dynamic Few-Shot Learning Techniques:* For complex prompts.

2. Control and Stabilization of LLM Generation (Module M3):

- *Fine-tuning of Inference Parameters:* Temperature, top-p, etc.
- *Multiple Sampling and Selection:* Generating several SVG candidates.
- *Intelligent Retry Mechanisms.*
- *Stop Sequences and Length Limits.*

5. Key Strategies (Part 2): Validation and Monitoring

Continuing strategies to make the system more robust:

5. Key Strategies (Part 2): Validation and Monitoring

Continuing strategies to make the system more robust: **3. Rigorous Validation and Defensive Postprocessing (Module M4):**

- *Multi-level Validation:* XML syntax, SVG profiles, attributes.
- *Heuristic Correction and Sanitization Rules.*
- *Complexity Limits (Circuit Breakers).*

5. Key Strategies (Part 2): Validation and Monitoring

Continuing strategies to make the system more robust: **3. Rigorous Validation and Defensive Postprocessing (Module M4):**

- *Multi-level Validation:* XML syntax, SVG profiles, attributes.
- *Heuristic Correction and Sanitization Rules.*
- *Complexity Limits (Circuit Breakers).*

4. Continuous Monitoring, Detailed Logging, and Feedback Loops:

- *Structured Logging (JSON):* For analysis and debugging.
- *Process and Quality Metrics.*
- *Root Cause Analysis.*
- *Human-in-the-Loop (HITL) Evaluation.*

6. Technology Stack and Implementation Plan

Recommended Technology Stack (Main):

- *Main Language:* Python 3.9+ (robust AI/ML ecosystem).
- *NLP/LLMs:* Hugging Face Transformers, Sentence Transformers.
- *SVG Generation/Manipulation:* lxml, svgwrite.
- *Data Handling:* Pandas. *Testing:* PyTest.

6. Technology Stack and Implementation Plan

Recommended Technology Stack (Main):

- *Main Language:* Python 3.9+ (robust AI/ML ecosystem).
- *NLP/LLMs:* Hugging Face Transformers, Sentence Transformers.
- *SVG Generation/Manipulation:* lxml, svgwrite.
- *Data Handling:* Pandas. *Testing:* PyTest.

Summarized Implementation Plan (8 Weeks):

1. *Weeks 11:* Initialization and Core Prototyping.
2. *Weeks 12:* Development of Central Modules.
3. *Weeks 13-14:* Implementation of Robustness and Quality Strategies.
4. *Weeks 15-16:* Optimization and Documentation.
5. *Weeks 17-18:* Final Testing, Refinement, and Packaging.

7. Conclusion

Significant progress was made in defining the conceptual framework, system architecture, and strategies, setting a solid foundation for implementation. Workshops helped clarify requirements and develop methods to handle the sensitivity and chaotic behavior of LLMs in creative tasks.

7. Conclusion

Significant progress was made in defining the conceptual framework, system architecture, and strategies, setting a solid foundation for implementation. Workshops helped clarify requirements and develop methods to handle the sensitivity and chaotic behavior of LLMs in creative tasks.

A systemic approach and systems engineering principles were essential to structure the problem and design modular components. Chaos theory provided insight into output variability, guiding adaptive strategy design.

7. Conclusion

Significant progress was made in defining the conceptual framework, system architecture, and strategies, setting a solid foundation for implementation. Workshops helped clarify requirements and develop methods to handle the sensitivity and chaotic behavior of LLMs in creative tasks.

A systemic approach and systems engineering principles were essential to structure the problem and design modular components. Chaos theory provided insight into output variability, guiding adaptive strategy design.

This comprehensive document reflects strong collaborative learning, supported by academic resources and professor guidance, positioning us confidently for the implementation phase.

Acknowledgements

- We thank Eng. Carlos Andrés Sierra, M.Sc., for his guidance and the resources provided for the Systems Analysis and Design course.

Q&A and Feedback

We are ready for your questions and ideas!