

Robust System Design and Project Management

Daniel David Cuellar, Henry Jhonmarcos Osorio Sánchez,
Juan Andrés Jimenez Palomino, Miguel Alejandro Chavez Porras

Universidad Distrital Francisco Jose de Caldas

Abstract. In the following document we develop the previous analysis of the system from the previous documents, adding a management plan from problems, making a risk analysis and verifying with the use of quality guidelines and standards.

Keywords: Quality · System Analysis · Risks · Management Plan

1 Previous Workshop Summary

In the previous workshop, the requirements of the system were proposed, including both functional and non-functional. Functional requirements can be synthesized in the process of designing, training, and evaluating the model, as well as cleaning datasets. Non-functional specified time between responses, data pipelines, as well as maintainability and scalability to ensure system efficiency and effectiveness and finally requiring traceability for its process steps.

Other requirements specified how the stochastic variables of the system should be handled to ensure stability for system sensitivity and chaos. This theme was treated deeply in its associated section, where each step had its considerations for chaos, and which ones a bigger effect on the system. And the last requirements treated were centered around the users, which described security of the data, transparency and reliability, and how its interfaces should present the interpretable predictions (system outputs).

Finally, some diagrams were proposed showing how the system is modeled based on its initial analysis, and how the data pipeline is managed. A little section of the technical stack was added as well.

2 Robust Design Principles

System Quality and Reliability

The system architecture follows robust design principles to ensure consistency, resilience, and adaptability. It adopts a layered modular approach, where each module operates independently but communicates through well-defined interfaces. This minimizes propagation of faults and allows isolated updates.

Fault tolerance is achieved through automated exception handling and recovery routines. This way a failure on a component can be restarted autonomously

without affecting the rest of the system. Feedback monitoring ensures as well, that any instability detected, can be corrected prematurely, maintaining the model reliability.

3 Quality Guidelines

Six Sigma
ISO 27001

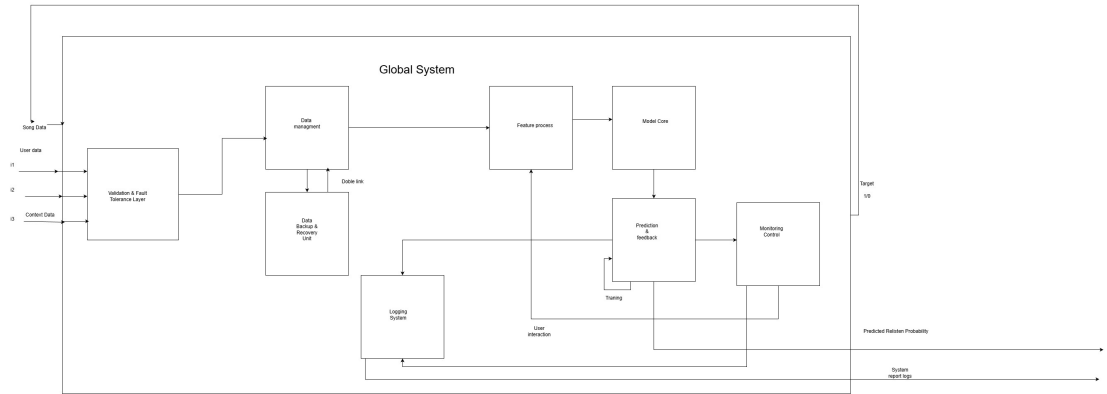


Fig. 1. Global System.

Validation& Fault Tolerance Layer

Function: Ensures data integrity, format compliance, and availability by validating incoming data streams before they reach the Data Management module. If inconsistencies or errors are detected, it triggers a fault signal to the Monitoring & Control system.

Data Backup & Recovery Unit

Function: Maintains secure backups and historical versions of all managed data. Provides bidirectional synchronization with the Data Management module to ensure data consistency and support recovery in case of failures.

Logging System Function: Extends the existing Monitoring & Control functionality to capture system events, performance metrics, and errors. Provides external outputs for detailed system reports and logs, facilitating auditing and operational oversight.

Data ingestion

Considering the nature of the problem and the proposed datasets, it is encouraged to use batch-based processing data supported by an ETL pipeline (extract-transform-load). The following diagram shows how data ingestion will be executed.

4 Risk Analysis

The system depends on files that describe users, songs, and listening habits. If any of these become corrupted, the system could produce poor results (when making predictions) or fail to function properly.

We can prevent this as follows:

- Perform data checks before use to verify authenticity and consistency.
- Keep backup copies both in the cloud and locally to restore data in case of any issue.
- Use version control to track all changes made.
- Carry out gradual maintenance on the data to detect errors, duplicates, or missing information.

At the same time, the model may lose performance due to the constant change in people's preferences, as a result of new musical trends and songs. This leads to a loss of accuracy if old data continues to be used.

We can prevent this as follows:

- Retrain the model with more recent data whenever performance decreases.
- Search for relevant inputs to evaluate user behavior, providing feedback based on their listening habits.

Finally, our action plan to keep the system running is as follows:

- We will monitor real-time performance to detect problems early; this can be done through a control panel.
- Based on any issues that may arise, we will provide feedback as needed to periodically improve what has been learned from monitoring.

5 Management Plan

Project Manager / Analyst :

- Define project scope, milestones, and deliverables.
- Analyze data requirements and coordinate team efforts.
- Monitor progress and manage communication.

Data Engineer / Developer :

- Clean, preprocess, and transform raw data for modeling.
- Develop data pipelines and ensure data integrity.
- Integrate validation and backup modules in the system design.

Machine Learning Engineer / Modeler :

- Design and train recommendation models.
- Optimize algorithms and evaluate performance metrics.
- Implement feedback loop for model improvement.

Tester / Quality Assurance :

- Test data pipelines and model outputs for accuracy and consistency.
- Validate system logging, fault tolerance, and backup recovery.
- Generate reports and document experimental results.

Milestone	Deliverable	Deadline (example)
Data Understanding and Preprocessing	Cleaned dataset with validation checks	Week 1
Feature Engineering and Modeling	Initial recommendation model	Week 2
Model Evaluation and Optimization	Performance metrics and tuned model	Week 3
System Integration and Testing	Integrated system with monitoring, backup, and validation	Week 4
Final Report and Presentation	Documentation, charts, and summary of results	Week 5

Table 1. Gantt chart

Kanban Board :

Track tasks for each team member, categorize as “To Do / In Progress / Done.”

Gantt Chart (Optional) :

Visualize timeline and dependencies between milestones.

Scrum Meetings (Weekly) :

Short stand-ups to monitor progress, identify blockers, and adjust tasks.

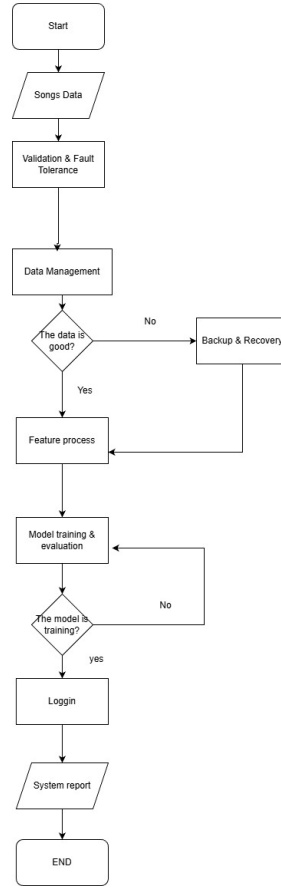


Fig. 2. Workflow.

6 Incremental Improvements

The analysis and design of this system revealed the complexity, sensitivity, and chaos that the problem has but besides that, show how the interaction of all the parts between them and how that is different than view each one separated. We also learn about the requirements of a system and how develop an understanding of the system in a design, Data flow, Risks way. Taking that in mind, now we have a deeper view of an overall system, seeing how to implement that system and now how to make it better with quality guidelines, having a Risk analysis for some possible troubles and take the corresponding plan.

In conclusion we can make the system more viable and sustainable.

References

1. <https://www.kaggle.com/competitions/kkbox-music-recommendation-challenge/overview>
2. <https://github.com/EngAndres/ud-public/tree/main/courses/systems-analysis>
3. Systems Analysis and Design, by Alan Dennis, Barbara Haley Wixom, and Roberta M. Roth
4. <https://www.guadalupanolasalle.edu.co/sgc/ISO9000-2015-Fundamentos-y-vocabulario.pdf>
5. <https://www.normaiso27001.es/>