

Business Intelligence

Exercise 6

Sana Afreen & Yordanos Girma

Goethe University

January 27, 2026

Exercise 1 – Overview

- Exercise 1 focuses on building and chaining **LLM-based pipelines**
- Each subtask performs a specific transformation or validation step
- Pipelines operate on:
 - BPMN models
 - Natural language descriptions
 - Extracted process elements
 - Detected modeling mistakes

Exercise 1: Overview

- Design and evaluate an LLM-based pipeline for BPMN validation
- Each part performs a single well-defined task
- Outputs are passed to subsequent pipeline stages

Exercise 1a: Prompt (Generator)

Prompt used:

You are given a BPMN model of a process. Generate a detailed natural language description of the process. Use the task and gateway names exactly as they appear in the BPMN model.

Input:

- BPMN XML file

Exercise 1a: Output (Generator)

Generated Process Description:

- The process starts when the passenger enters the terminal.
- The passenger goes to the machine.
- The passenger identifies themselves and chooses a seat.
- The machine prints the boarding pass.
- If the passenger has baggage, they go to the baggage counter.
- Finally, the passenger proceeds to the security check.

Observation:

- Fluent and readable description
- BPMN names are paraphrased
- Some required elements are missing

Exercise 1b: Prompt (Extractor)

Prompt used:

Extract all process elements mentioned in the following process description. Return only a list of element names without explanations.

Input:

- Generated process description (Exercise 1a)

Exercise 1b: Output (Extractor)

Extracted Elements:

- Passenger enters terminal
- Passenger goes to the machine
- Passenger identifies themselves
- Passenger chooses seat
- Passenger prints boarding pass
- Passenger goes to baggage counter
- Passenger goes to security check

Observation:

- Semantic understanding is correct
- Exact BPMN labels are not preserved

Exercise 1c: Prompt (Mistake Detector)

Prompt used:

Compare the following list of extracted elements with the BPMN model. Identify missing, extra, or incorrectly named elements.

Inputs:

- Extracted element list
- Original BPMN model

Exercise 1c: Output (Detected Mistakes)

Detected Issues:

- Missing BPMN element: Passenger gets Boarding Pass
- Missing BPMN element: Passenger goes to security check
- Name mismatch:
 - “Passenger prints boarding pass”
 - Expected: “Passenger gets Boarding Pass”
- Extra element not in BPMN:
 - Passenger identifies themselves

Result:

- Structured error list forwarded to corrector

Exercise 1d: Prompt (Corrector)

Prompt used:

*Correct the process description based on the detected mistakes.
Use the exact BPMN element names and do not introduce new elements.*

Input:

- List of detected mistakes

Exercise 1d: Output (Corrected Description)

Corrected Process Description:

- The passenger enters the terminal.
- The passenger goes to the machine.
- The passenger chooses seat.
- The passenger gets Boarding Pass.
- If baggage is present, the passenger goes to baggage claim counter.
- The passenger goes to security check.

Observation:

- Missing elements were added
- BPMN names are closer to original labels
- Minor paraphrasing still possible

Exercise 1: Key Takeaways

- Prompt design directly affects output quality
- LLM understands semantics but struggles with exact BPMN labels
- Feedback loop improves results but does not guarantee correctness
- Sets foundation for Exercises 2 and 3

Exercise 2: Passenger Check-In Machine

Objective:

- Apply the pipeline from Exercise 1 to a real BPMN use case
- Evaluate generator, extractor, detector, and corrector behavior

Case Study:

- Passenger Check-In Machine BPMN process
- Focus on alignment between BPMN model and generated description

Exercise 2: Initial Generator Output

Observed Issues:

- Multiple hallucinated BPMN elements:
 - Arrival at Terminal
 - Flight Check-in Begins
 - Parallel Gateway
 - Successful Completion
- Generator introduced non-existent start and end events
- Gateway logic frequently misrepresented

Exercise 2: Missing BPMN Elements

Required elements not present in initial output:

- Passenger goes to the machine
- Passenger chooses seat
- Passenger gets boarding pass
- Passenger enters terminal
- Passenger goes to security check
- Passenger goes to baggage claim counter

Impact:

- Low semantic alignment
- Incorrect process flow interpretation

Exercise 2: Iterative Correction Results

After 5 iterations:

- Some missing elements were gradually added
- Persistent hallucinations repeatedly reappeared
- Exclusive gateway was incorrectly modeled as a task
- Inconsistent element naming caused repeated mismatches

Exercise 2: Final Outcome & Assessment

- System did not fully converge after 5 iterations
- Feedback loop architecture works as intended
- LLM struggled with strict BPMN semantics
- Missing elements still detected in final iteration

Exercise 3: Extended BPMN Evaluation

Objective:

- Evaluate pipeline robustness on additional BPMN processes
- Analyze limitations of LLM-based correction

Processes Evaluated:

- Passenger Check-In Counter
- Baggage Departure BPMN

Exercise 3: Passenger Check-In Counter

Key Issues:

- BPMN process only partially reflected
- Exact BPMN element names not preserved
- Luggage decision gateway incorrectly modeled
- Generic hallucinated elements introduced:
 - Start Event
 - Exclusive Gateway

Exercise 3: Iterative Correction (Counter)

Correction Behavior:

- Missing elements added incrementally
- Persistent paraphrasing of BPMN labels:
 - “choose a seat” instead of “Passenger chooses seat”
 - “take luggage if necessary” instead of “Take luggage”
- Continued hallucination of control-flow elements

Initial Generator Output:

- Generic BPMN description
- No domain-specific baggage activities
- Hallucinated generic elements:
 - Activity A / Activity B
 - Decision Point (XOR)
 - Start Event / End Event
- 0% alignment with real BPMN task names

Exercise 3: Iterative Correction (Baggage)

Observed Improvements:

- Gradual addition of domain-specific tasks:
 - X-ray screening
 - Manual baggage inspection
 - Labeling and sorting activities
- Partial integration of security decision points
- Inconsistent retention of required elements

Exercise 3: Final Conclusion

- Correction pipeline improves coverage but lacks stability
- Hallucinated BPMN constructs persist
- Exact BPMN name compliance is difficult for LLMs
- Demonstrates limitations of LLMs for strict BPMN modeling

Overall Conclusion

- Pipeline architecture is conceptually sound
- LLMs perform well for semantic understanding
- Exact BPMN compliance remains a major challenge
- Iterative feedback improves results but does not guarantee convergence

Thank You!

Business Intelligence – LLM-based BPMN Pipelines