Patient Monitoring Service (PMS)

Part 2: Architecture evaluation

Context. We have obtained a first design (and implementation) of some of the PMS functionality. However, the main architect behind this initial design and its principal design decisions has left the company, and now you are hired to (i) establish a solid understanding of this initial system (part 2 of the project assignment, *this part*), and (ii) extend the initial system (part 3).

1 Part 2 – analysis and evaluation

The aim of this assignment is to establish a solid understanding of the initial PMS architecture. To streamline this, we have compiled a list of targeted questions (labeled 'Q') and extension exercises (labeled 'E') that will help you to achieve this goal.

The questions allow you to examine the system from the perspectives of Modifiability, Interoperability, Availability, Performance and Scalability. Use the architectural description document and the VPP model to formulate an answer to these questions. Make sure you understand the notation and meaning (semantics) of the different architectural views.

Some pointers and guidelines:

- Don't just answer a question with yes/no or with a component name, but share your insights and opinion, and discuss how you would address the issue at hand (if any).
- Write down the reasoning you followed and highlight the evidence found in the initial architecture. Don't just echo or copy relevant diagrams that you've found in the description document; you can just refer to these with their number (section number, page number, figure number, etc).
- Focus on assessing whether the architectural description is clear, complete and correct, or whether it lack important aspects and/or has inconsistencies with other parts.
- If you cannot fully answer a question, state what information is missing, and how you could obtain it. It is OK to make assumptions that cannot be backed directly with evidence from the architectural description, but make sure to state them explicitly.
- You are expected to include your opinion on the design choices that have been made into your answers. Would you have done it differently? Did you reckon a good decision has been made, or do you foresee problems as we continue with this initial system?

1.1 Clinical models

The initial system uses clinical assessment models used to assess the risk level of a patient.
Q1. What is the distinction between `ClinicalModel`s and `MLModel`s (if any)?
Q2. Where are the `ClinicalModel`s stored? Who creates clinical models in the system? Will a single `ClinicalModel` be used for different patients?
Q3. The system uses machine learning based models to perform risk assessment. Are different types of models (non-ML) also supported and if not, would adding support be impactful? Where are these supported `MLModel`s stored?
Q4. When they are needed for a risk estimation job, how does the `RiskEstimationProcessor` get the applicable models (both `ClinicalModel`s and `MLModel`s)? Is it fetching or loading these for every job that is executed, and why (not)?
Q5. Could the clinical models be executed on the patient gateway (smartphone)? Which architectural considerations play a role in this decision? Can 'parts of clinical models' be performed on the gateway (i.e., could the required computational load be distributed over the involved devices in some way)?
Q6. Does the developer need to provide the functionality for executing or applying the applicable models (both `ClinicalModel`s and `MLModel`s) from scratch? If not, who develops this?
Q7. Which non-functional considerations have been taken into account in the initial design of the `RiskEstimatorProcessor`? Per discussed non-functional aspect, indicate which specific view and diagram you used to find out.
Q8. Would it be possible to change to a different `MLaaS`? What is the architectural impact of such a change?
Q9. Concrete algorithms have been abstracted. Is the selection of ML algorithms architecturally significant? If so, which non-functional aspects play a role?

1.2 System context

Correctly understanding the relationship between the system under design (SUD) and external elements is crucial. When answering the following questions pay careful attention to the correct interpretation of system context.
Q10. What is the position of the `MLaasLibrary` in the context diagram of the client-server view, and in the context diagram of the decomposition view, and explain why. What does this model element represent?
Q11. What is the position of the `MLLibrary` in the context diagram of the client-server view, and in the context diagram of the decomposition view, and explain why. What does this model element represent?
Q12. What is the position of the `MLaaS` and its *node* in the context diagram of the *deployment* view, and explain why. What do these model element represent?

1.3 Performing risk assessments

Q13. How does the system ensure an adequate throughput of risk calculation jobs?
Q14. How does the system react if there are too many processing jobs?
Q15. What if the system resources (CPU, memory, storage) for performing the risk calculations prove to be insufficient?

Q16. What happens when risk calculation jobs fail (e.g., due to crashes or bugs)? Does it affect other jobs?

1.4 Making changes to clinical models

Q17. The system uses machine learning based models to perform risk assessment. Are different types of models (non-ML) also supported and if not, would adding support be impactful?
Q18. Do changes made to clinical models affect ongoing calculations? Why (not)? When will changes to clinical models affect new risk estimation jobs?
Q19. Which subsystems will be affected by modifications to the clinical risk models?
Q20. Where is the `ClinicalModelCache` deployed? Which other component(s) should be deployed together with the cache?

1.5 Design extension

E1. Extend this initial design with a dedicated dashboard that allows PMS Telemedicine operators to overview the ongoing risk estimation processes, monitor the performance of the subsystem in terms of relevant system metrics (e.g., average job execution time, system status). Make sure that this is a web front-end (e.g., based on web frameworks such as Kibana, Zabbix) has read-only access for inspecting the system with no side-effects (e.g., no configuration actions or changes to the operational system).

- Document your decisions and communicate your solution in the appropriate architecture views, using Visual Paradigm.
- ⚠ Pay specific attention to updating the context diagrams.
- Ensure that your extended does not introduce any new issues as reported by the SAPlugin.
- Include the changed/newly-introduced elements of the architecture (diagrams, interface specifications) in the report. You can select these elements from the export generated by the SAPlugin for Visual Paradigm.
- Describe and discuss your main decisions (e.g. design, deployment), and pay attention to the overall architectural impact of your changes.

Submit the VPP file along with the part 2 report.

1.6 Final reflections

Q21. Which architectural tactics or patterns did you recognize? For which quality attributes do you think they were introduced?
Q22. How would you summarize the strengths and weaknesses of this architecture? Are there other aspects of the architecture that strike you as either genius or odd? If so, list these below and briefly explain.
Q23. Which follow-up questions would you ask the software architect that has created this initial architecture (if any)? Which assumptions or interpretations made during the evaluation exercise would you like to double-check?

2 Instructions

For part 2, you are asked to create a report with motivated answers to the questions above.

Formatting rules You can use the tool of your choice to author the report, but you must deliver a single PDF file. No other digital formats are accepted. The file must be self-contained (e.g., no extra figures in attachment) and readable (e.g., font size in pictures is adequate). Pages must be numbered. The cover page must mention the course name and the team member names (including their student id numbers).

Answer each question with the following template:

> **Q123** (you may copy the question text, but do not have to)
>
> - **Answer:** Your answer to the question.
> - **Evidence and assumptions:** Pointers to the evidence that you have used to answer the question (figure numbers, section numbers, etc.), as well as assumptions that you have made but that are not in the architectural description.
> - **Opinion and improvements:** Your opinion on how the aspect you have investigated is handled in the current architecture, and possible improvements you would propose.

In case you decide to use LaTeX, you can use the provided template (on Toledo). In that case, you can select the relevant elements from the extended architecture from the LaTeX report that can be generated by the SAPlugin in Visual Paradigm.

Delivery The deadline to turn in your report for part 2 is March 31 (noon). You are expected to upload both the PDF report and the extended VPP file on Toledo.

The Software Architecture team.