Architecture Design & Consideration

Our team has done intensive research regarding the architecture of Nurse Rotation planner System.

Before we chosen our framework, we have studied 2 types of architecture.

1. Java Swing Based desktop executable applications. (opta planner example code)
   Building desktop executable applications are not recommended in the current industry practice.
2. Apache ISIS Web Application (from github, with opta planner version 6.4)

   Very complex code base and inflexible to expand or decouple services.

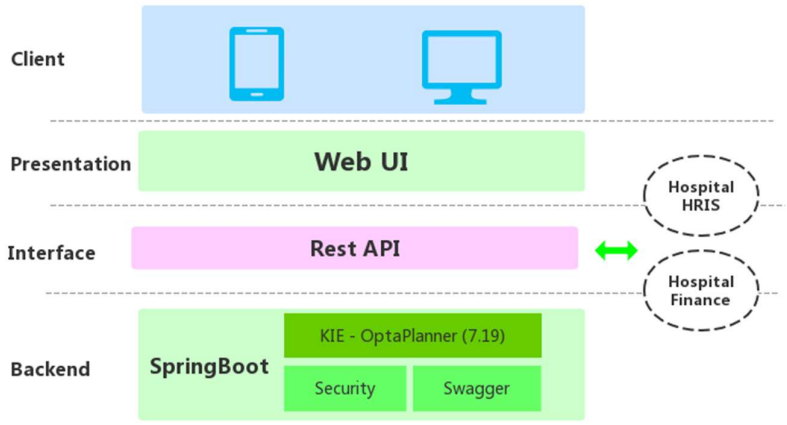
In recently years, vast number of companies are adopting micro-service architecture, which allows large systems to be built up from a number of collaborating components. In DBS, almost exclusively, new systems we built are microservice using Spring Boot.

We further researched and understood that Red Hat Process Automation Manager, from version 7.1 release, it supports the deployment of the process automation manager runtime as a "**capability**" within Spring Boot applications. (**1st Nov 2018**)

https://developers.redhat.com/blog/2018/11/01/spring-boot-enabled-business-process-automation-with-red-hat-process-automation-manager/

Considering the above, we have decided to adopt this architecture as the Pivot project for future business value-adding projects, with the experience gained in integrating Spring Boot and Opta Planner, it will be much easier for our individual firms to land on real-life optimization systems based on OptaPlanner.

Architecture Diagram



https://www.processon.com/view/link/5cb71759e4b08b66b9af536c

Business Model Design

How is the business problem translated into OptaPlanner Model Design.

Business Problem: (mei rong to add)

Our Design:

1. Model our planning problem as a class that with annotation @PlanningSolution. (NurseRoastering.class)

   Problem Facts: (To be added from the source code)

Skill	
Shift Type	

   Planning Entities: (This returns the final optimized solution)

   Shift Assignment

   Planning Score:

   HardSoft Score based on the rules.

2. Configure NurseRoastering Solver to achieve optimized performance.

Heuristic Type	Weakest Fit	
MoveThreadCount	Auto (Use the maximum CPU cores)	
Entity Tabu Size	7	
Further info could be added in possible		

3. Data Set & UI

   Due to time constraint, we did not prepare the UI so that it allows users to fully customize the data on the screen, instead, we have created multiple data sets based on the current ward nurse rotation problems in Singapore.

   For the user friendly consideration, we let users to use dropdown to test the different combinations. The best result will be displayed to the web UI which contains the roastering planning for the next month (could be extended for longer dates too)