Integração do Sorting Hat com usVision

Autor: Vinicius Pereira Ximenes Frota (IME-USP)
Orientador: Prof. Dr. Eduardo Guerra (unibz)
Co-orientador: MSc. João Francisco Lino Daniel (unibz)

Introdução

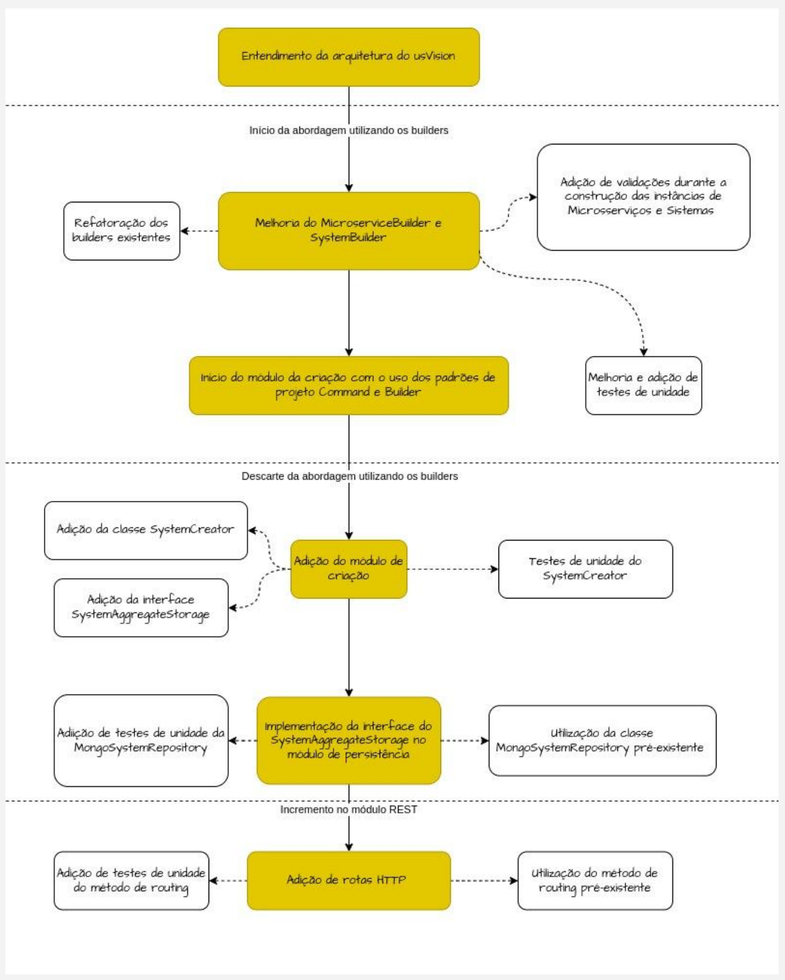
A usVision é uma ferramenta responsável por modelar a comunicação entre microsserviços de um sistema de software e gerar relatórios dos padrões e anti-padrões (más práticas) de arquitetura de microsserviços. Através do seu módulo de persistência, ele lê os dados do sistema do mongoDB para poder instanciar os modelos dos sistemas de software e gerar esses relatórios.

Antes da realização deste trabalho, o usVision não possuía um módulo de criação desses microsserviços para inserção em seu banco de dados. Portanto, era necessário popular seu banco de dados via query diretamente para utilizá-lo.

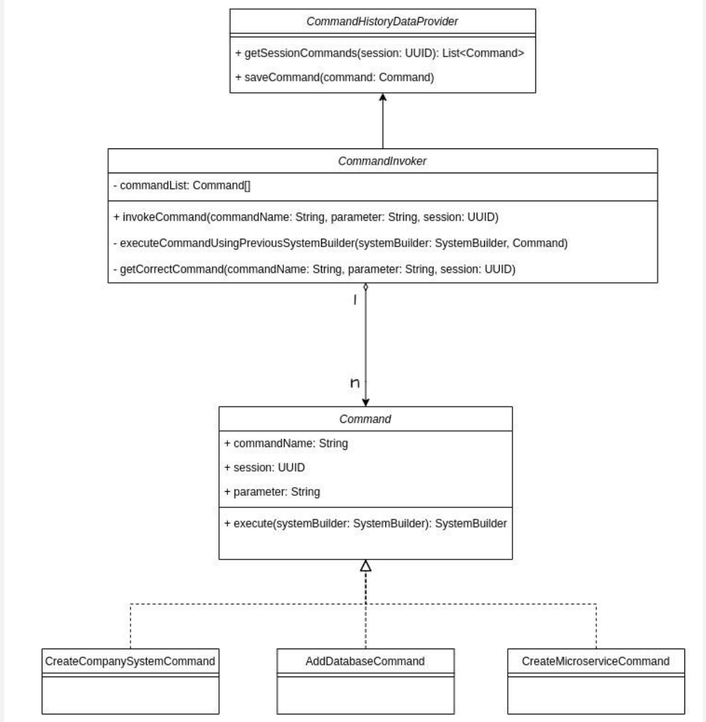
O Sorting Hat, por outro lado, é uma ferramenta que coleta as informações dos microsserviços de um sistema por meio de repositórios no GitHub a partir de heurísticas e análises do Docker.

Inspirado pelo Sorting Hat, este trabalho criou um módulo de criação e uma interface REST para este módulo no usVision, permitindo que ferramentas como o Sorting Hat ou outra com a proposta parecida, insira no banco de dados as informações de um sistema de software que utiliza a arquitetura de microsserviços.

Roadmap



Abordagem descartada

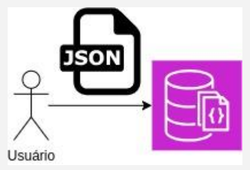


1. Para cada comando novo, o CommandInvoker executa novamente os comandos anteriores e em seguida invoca o comando atual.
2. Cada comando chama o construtor ou métodos de um SystemBuilder/CompanySystemBuilder

Essa abordagem foi descontinuada, pois:

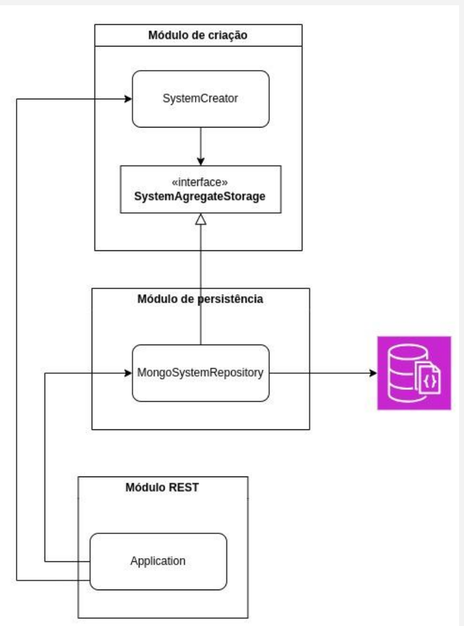
1. Builders são frequentemente utilizados em Java para classes que possuem construtores grandes e deixar claro qual parâmetro está sendo passado.
   a. Como o usVision foi construído em Kotlin, os construtores têm parâmetros nomeados e opcionais, portanto, não segue o nosso caso de uso.
   b. Os construtores e demais métodos de construção das classes de sistemas e de microsserviços são simples. Portanto, os builders não eram necessários.
2. A repetição da execução dos comandos é desperdício de processamento, de memória e até de código.

Abordagem utilizada

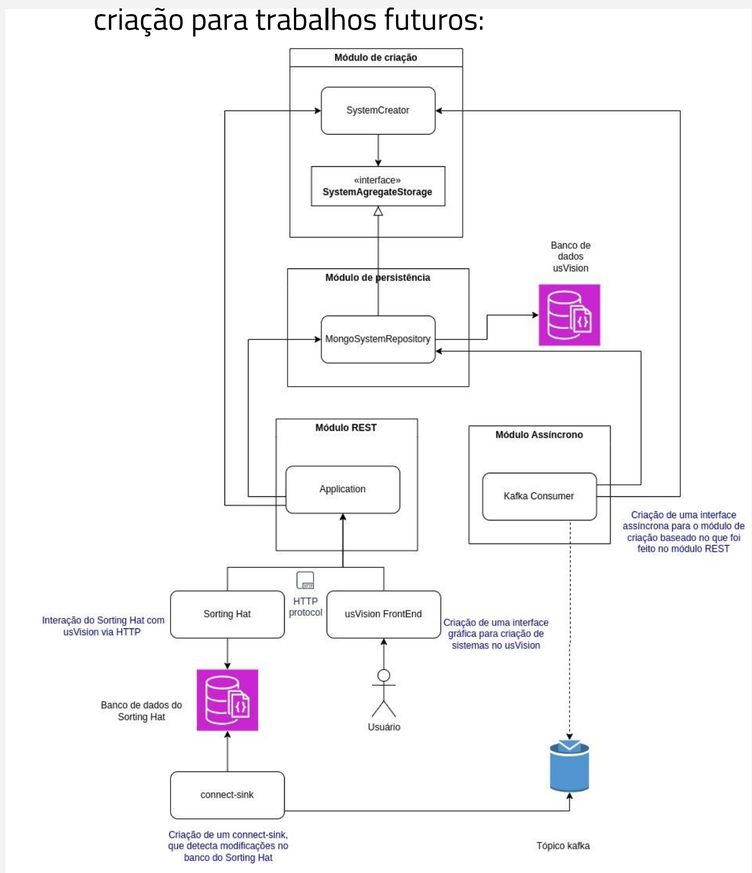


Antes, a inserção era feita diretamente no banco de dados, conforme ilustrado na figura ao lado

- Agora, é possível popular o banco de dados do usVision através do módulo REST:



- Também é possível utilizar o módulo de criação para trabalhos futuros:



Conclusão

- Ao utilizar um padrão de projeto, é preciso considerar se ele serve para o caso de uso e até se a linguagem de programação utilizada o torna desnecessário.
- A criação de um módulo específico para a criação de sistemas e microsserviços permite a adição de módulos de interface para a criação, como foi feito com a interface REST e poderá ser feito com uma interface assíncrona de mensageria, por exemplo.
- A criação de uma interface REST para criação de microsserviços permite que o Sorting Hat ou outra ferramenta se comunique com o usVision.

Referências

- Gamma, Erich, et al. Design Patterns : Elements of Reusable Object-Oriented Software. Boston, Addison-Wesley, 1994.
- Evans, Eric. Domain-Driven Design : Tackling Complexity in the Heart of Software. Boston, Mass. ; Munich, Addison-Wesley, 2014.

Página

https://vfrota-tcc-landing.vercel.app/