Casos adicionais de API Conversacional

Caso 1: API de Aprendizado de Máquina

Em um trabalho recente que fizemos com uma API que **não** é de domínio público, nós tivemos a oportunidade de implementar aquilo que propomos em nosso framework como conversação plena. No caso abaixo, temos a comparação do nome dos métodos da API antes e depois de aplicarmos os conceitos de API Conversacional. Em especial, gostaríamos de destacar aqui como exemplo os métodos "***avaliable_models***" e "***avaliable_processor***". Na versão anterior da API, estes métodos não existiam e, assim, o usuário da API era obrigado a consultar a documentação para descobrir os modelos e processadores existentes. A adição destes métodos traz a conversação para a interface da API, garantindo melhor fluidez da conversa usuário-API. Além disso, garantia a compatibilidade da API em caso de uma evolução e criação de novos modelos ou processadores, não sendo necessário modificar a documentação. Abaixo os métodos que existiam antes e depois do redesign da API.

VERSÃO "Não-Conversacional"

Métodos:

- *create_experiment(name, type)*

- *build_data(id_experiment, processor, output)*

- *train_model(id_experiment, data, model, hparams)*

- *run_model(id_experiment, id_model, hparams)*

VERSÃO "Conversacional"

Métodos:

- *create_experiment(name, type)*

- *build_data(id_experiment, **processor**, output)*

- *train_model(id_experiment, data, **model**, hparams)*

- *run_model(id_experiment, id_model, hparams)*

- available_models(type)

- avaliable_processors(type)

Caso 2: API de automação de Refatoração em Java

Em outro trabalho recente que também fizemos com design de uma API, nós tivemos a oportunidade de aplicar o que propomos em nosso framework conceitual. No caso abaixo, temos uma demonstração de uma característica que faz a API ser conversacional plena. Em destaque o elemento que traz a conversação da API direto para a interface. O campo "**string**" no retorno da função "**applyRefactoting**". Neste campo a API passará uma justificativa do por que uma refatoração foi realizada. Assim, a conversa deixe de acontecer apenas na documentação da API, e agora faz parte da interface. Como consequência, o usuário da API é a convidado a conversar sobre o motivo da refatoração enquanto executa o método.

VERSÃO "Não-Conversacional"

Método:

/**

Aplica refatoração aos arquivos passados como parâmetro

**/

- *public void applyRefactoting(files)*

VERSÃO "Conversacional"

Método:

/**

Aplica refatoração aos arquivos passados como parâmetro e retorna uma String com a justificativa para as refatorações aplicadas.

**/

- *public **string** applyRefactoting(files)*

Modificação na documentação API Conversacional Plena – *"roundType"*

Veja na figura abaixo um comentário adicional a documentação proposta originalmente no artigo. O texto extra está em destaque na caixa vermelha.

```
954
955     /** Returns a copy of this LocalDate with the
956         specified number of months added.
957     This method adds the specified amount to the
958         months field in three steps:
959     (1) Add the input months to the month field
960     (2) Check if the resulting date would be invalid
961     (3) adjusts if necessary as indicated in the
962         parameter roundType.
963     *** Use value 0 to throw an exception in case of
964         invalid date result
965     *** Use value 1 to indicate roll previous valid
966         date
967     *** Use value 2 to indicate roll next valid date
968     May trigger an exception if you don't set
969         roundtype
970     Be aware of the information messages on the return
971          object
972     This instance is immutable and unaffected by this
973         method call. */
974     public LocalDate plusMonths(long monthsToAdd, int
975         roundType)
976
977
978
```

Listing 8: API Coversacional Plena - Signos Estáticos e metalinguísticos