REQUISITOS NÃO-FUNCIONAIS

Enquanto que os requisitos funcionais se preocupam em “O que” o sistema deve conter para funcionar, que são suas funcionalidades, os requisitos não-funcionais especificam “Como” o sistema deve se comportar, que são seus comportamentos. São características mais difíceis de ser medidas do que os requisitos funcionais, no entanto, elas ainda podem ser verificadas. A seguir serão listados os requisitos não-funcionais do software:

- **Manutenibilidade:** Este software foi escrito pensando na facilidade em dar manutenção em tempos futuros. O software “Vencer Sempre” prioriza a extensibilidade, códigos corretivos e métodos evolutivos. Cada método ou classe contém um nome específico condizente com a realidade daquele método/classe, dividindo as informações e funcionalidades em hierarquias. Um exemplo é classe Emprestimo que pode realizar empréstimos, e seu método mais evidente para isto é o Agendar() que utiliza do método Registrar(), da mesma forma, para cancelar um empréstimo, é utilizado o método Cancelar() que utiliza o método Excluir(), isto é, apenas lendo o nome do método é possível saber o que ele faz. Além de toda herança, cada início de método foi comentado com “Sumários” que podem ser documentados, descrevendo o que a função faz e quais são seus parâmetros. O código está implementado de uma forma que novas extensões podem ser criadas facilmente, assim como adaptações/melhorias, incluindo correções. Um exemplo que é passivo de correção, são classes distintas das interfaces gráficas que executam quase a mesma lógica em um mesmo método, contendo diferenças bem sutis, isto significa que há uma possibilidade de ser dividido em uma classe nova que trata destas questões similares. Outro exemplo são a variedade de métodos sobrecarregados, que seguindo a mesma lógica, novas funcionalidades podem ser implementadas usando uma combinação destes métodos.

- **Usabilidade:** A usabilidade é um conceito amplo e muito complexo de medir ou verificar, muitas vezes dependendo do usuário para avaliar como o software “Reage” ou “Ensina” ele a utilizar. Alguns destes conceitos foram usados na engenharia desse software, que é a questão de ter elementos posicionados e agrupados de uma maneira que o usuário rapidamente pode ver e identificar, assim como saber manipular estes elementos. Além de todas as imagens chamativas porém sutis e leves que combinam com as nuances de cores em azuis da interface, o software pode agir como “professor” do usuário, mostrando pra ele quais são as funcionalidades do sistema, suas semelhanças, diferenças e utilidades que elas apresentam durante o ciclo de atividade. Cada botão tem seu espaço, seu nome e sua janela. Cada janela compartilha do

mesmo pensamento, mas agindo em prol de objetivos diferentes. Será que o modelo mental apresentado aqui é o mesmo modelo transmitido ao usuário neste momento? É um caso a se pensar. Já na questão de menus em campos de texto, isso evita que o usuário possa errar ou esquecer dados relativo aos nomes dos professores ou nomes exatos dos equipamentos. Com a atribuição de menus, essa possibilidade é eliminada, devendo o usuário simplesmente selecionar o dado e visualizar os filtros na tabela correspondentes ao mesmo dado. Os formatos de fontes de texto de toda a interface também priorizaram a leitura, usando Arial tamanho 12.

- **Acessibilidade:** Por último, mas não menos importante, em cada elemento construído da interface, existem as Strings de acessibilidade, que são atribuídas juntamente com seu nome de acessibilidade e tipo de elemento. Estes dados são úteis para softwares que auxiliam deficientes visuais, no qual é necessário “ouvir” narrações destes softwares de acordo com as Strings especificadas. A String de descrição contém uma maneira rica, porém objetiva de descrever o componente, enquanto que a String de nome da acessibilidade é uma maneira direta de identificar o componente, usando um linguajar similar ao contexto de surdos-mudos ao se comunicar em libras. Exemplo: “JanelaListaEntregues”, “BotãoCadastroProfessor”, etc.

Cada um destes requisitos não-funcionais pode ser verificado e inspecionado no código-fonte do software, disponível em uma das pastas deste trabalho. O código pode ser manutenível visando uma maior rentabilidade econômica na criação de novas funcionalidades de forma rápida. O diretor ou secretário da escola facilmente poderá aprender a utilizar a interface para agendar um empréstimo no primeiro dia, além de ter respostas rápidas na utilização. E se caso um destes funcionários tiver um tipo de deficiência auditiva ou visual, o software Vencer Sempre não vai restringi-lo, possibilitando usar novos softwares para auxiliá-lo a identificar os elementos da interface e utilizar normalmente sem dificuldades.