III. Requisitos são Frases

Entendemos requisitos de software como sentenças que expressam as necessidades dos clientes e que condicionam a qualidade do software. Em função disso classificamos requisitos como requisitos funcionais, requisitos não funcionais e requisitos inversos. O primeiro está diretamente ligado a funcionalidade do software, enquanto o segundo reflete os requisitos que expressam restrições que o software deve atender ou qualidade específicas que o software deve ter, o terceiro trata de situações que não podem ocorrer.

III.I Representação

Requisitos Funcionais

Para os requisitos funcionais existem três classes de sentenças: entrada, saída e mudança de estado. Cada requisito tem que seguir uma das duas estruturas abaixo.

O sistema deve + [verbo + objeto | frase verbal]
+ [complemento de agente | null] + [condição | null].

O sistema deve + [verbo + objeto | frase verbal]
+ [complemento de agente | null] + { a) condição-1,

b) condição-2, ... condição-n}.

Definições

- **Verbo** é um verbo simples que expresse a funcionalidade daquele requisito. Dependendo do tipo do verbo, objeto pode ser um objeto direto ou um objeto direto seguido de um objeto indireto.
- **Frase verbal** é uma frase que expressa a funcionalidade do requisito.
- **Complemento de agente** é a identificação de um agente relacionado com o requisito. Algumas vezes esse complemento pode ser descrito pelo objeto indireto. Um agente pode ser uma pessoa, uma instituição, um grupo ou um dispositivo físico externo ao software.
- Uma **condição** é uma sub-sentença que reflete uma situação específica.

Uma forma de adicionar mais estrutura, por exemplo no que diz respeito a conectivos é apresentada abaixo.

O sistema deve +verbo + objeto + "para"
+ complemento + ["quando" | "se"] + condição.

Requisitos Não-Funcionais

Os requisitos não-funcionais podem ser expressos de duas maneiras: independentes ou associados a um requisito não funcional. No caso de um requisito não-funcional associado temse a seguinte estrutura:

O sistema deve + [verbo + objeto | frase verbal]
+ [complemento de agente | null] + [condição | null] + [restrição | null].

Nesse caso **restrição** é uma frase que qualifica a funcionalidade restringindo-a. É uma frase não verbal onde vezes o núcleo da frase é um adjetivo.

Os requisitos não-funcionais independentes têm geralmente a seguintes estruturas:

O sistema deve + [“ter” | “manter” | “possuir” | “atender” | “fazer” + objeto]
+ frase-adjetiva.

O sistema deve + [“ter” | “manter” | “possuir” | “atender” | “fazer” + objeto]
+ [frase-adjetiva | null].

Condição + o sistema deve + [“ter” | “manter” | “possuir” | “atender” + “fazer” + objeto]
+ [complemento de agente | null] + [frase-adjetiva | null].

Requisitos Inversos

Os requisitos inversos são situações que não podem ocorrer em situação alguma. São de certa maneira restrições de alcance geral.

O sistema não pode + [frase verbal].

III.II Organização

Os requisitos são um conjunto de sentenças numeradas e classificadas segundo seu tipo (RF e NFR e RF 1 –) e sua classe (entrada, saída e mudança de estado) no caso de requisitos funcionais. Em caso de listas muito grandes, uma boa política é agrupar os requisitos com menor distância conceitual.

Abaixo listamos alguns exemplos.

Loja de Vídeo

Requisitos Funcionais:

1. O sistema deve cadastrar o cliente. (entrada)
2. O sistema deve emitir um recibo para o cliente. (saída)
3. O sistema deve transformar uma fita disponível em fita emprestada, quando a fita for alugada pelo cliente. (mudança de estado)

Requisitos Não Funcionais:

4. O sistema deve cadastrar o cliente rapidamente, em menos de 2 minutos.
5. O sistema deve emitir um recibo para o cliente, com o tempo máximo de 8 segundos após a transação .
6. O sistema deve atender as normas do padrão IEEE.

Requisitos Inversos:

7. O sistema não pode perder dados do cliente.

Biblioteca

Requisitos Funcionais:

1. O sistema deve cadastrar bibliotecários. (entrada)
2. O sistema deve cadastrar os usuários. (entrada)
3. O sistema deve achar para os bibliotecários, qual o usuário que está com um determinado livro. (saída)
4. O sistema deve tornar um livro em livro emprestado, quando um usuário pegar este livro emprestado. (mudança de estado)

Requisitos Não-Funcionais:

5. Dependendo do tipo de usuário o sistema deve atender a completa revogação da multa.
6. O sistema deve cadastrar os usuários de maneira amigável, por intermédio de uma interface fácil de usar.
7. O sistema deve fazer o cadastramento rapidamente, em menos de 3 minutos.
8. O sistema deve ser portável para plataformas Linux.

Requisitos Inversos:

9. O sistema não pode cobrar multa de professores em tempo integral.

Além da organização básica vista acima, é possível que as listas de requisitos possam estar ligadas a outros modelos de apoio ao esclarecimento ou a elicitação dos requisitos. Abaixo, falamos sobre uma ligação desse tipo com um glossário.

III.III Armazenamento

Como vimos anteriormente o armazenamento pressupõe que tenhamos políticas para **classificação**, **indexaçã**o e **apresentação** dos requisitos .

Adotando-se a lista de requisitos, estes estarão classificados segundo o que vimos acima: por tipo e por classe. Eventualmente, caso seja necessário, um esquema de agrupamento segundo conhecimento explícito do domínio pode juntar requisitos em grupos afins.

A indexação dos requisitos é feita segundo o esquema de armazenamento, podendo eventualmente estar disponível acesso por palavras chave. Essas palavras chave ficam são melhor definidas quando de utiliza um esquema de glossário em conjunto com a lista de requisitos [Leite 93]. Referências cruzadas entre requisitos podem ser feitas por meio do glossário ou por ligações diretas a numeração dos requisitos. Existe também a possibilidade de ligações com outros modelos, neste caso a organização da lista deve prever explicitamente essas ligações

dentro de sua política de rastreabilidade.
A apresentação dos requisitos é principalmente em forma de lista. Eventualmente pode-se apresentá-los em forma de hipertexto, caso haja uma implementação das relações com o glossário ou com outros requisitos da mesma lista, ou ainda com outros artefatos.
Além dos aspectos de armazenamento sob a ótica de reutilização, como vimos acima, há que se ressaltar o aspecto de **evolução** dos requisitos. Nesse caso deve-se ter disponível um esquema de gerência de configuração que torne possível que os requisitos sejam uma *baseline* para o processo de software e que além disso esteja em constante evolução. O conceito de *requirements baseline* foi formulado por nós [Leite 97] e aplica-se a qualquer modelo de requisitos, obviamente incluindo o modelo que enfatizamos aqui, o de sentenças de requisitos. A definição da *baseline* de requisitos explica que essa evolução se dá em dois eixos, um no que diz respeito a pontos de referência do modelo de processo de software e outro eixo no que diz respeito a progressão do processo de software no que se refere a mudança de nível de abstração.
Ou seja o *baseline* de requisitos muda tanto num mesmo nível de abstração como entre níveis de abstração. Portanto o armazenamento do modelo de requisitos deve levar em consideração que esses requisitos evoluíram e precisam ter um controle sobre possíveis configurações e versões.
É claro que a complexidade de controlar essa *baseline* é não trivial, no entanto é uma maneira integrada de tentarmos garantir que a alocação dos requisitos, tanto funcionais como não funcionais seja registrada e rastreável. Assim, é possível efetivamente gerenciar os requisitos, principalmente num contexto volátil.