INF1413 Teste de Software

Período: 2017-2

Prof. Arndt von Staa

1o. Trabalho

Data de divulgação: 29 de agosto (terça-feira)

Data de entrega: 13 de setembro (quarta-feira)

OBS. Mudei a data de 11/9 para 13/9.

1. Conceitos

No contexto de software, um sistema corresponde a um grafo conexo, em que os vértices são os elementos do sistema e as arestas as interdependências entre eles. Sistemas de software são descritos por vários documentos (ex. especificação em português, projetos em UML, documentação textual de módulos e componentes, código, script de teste, entre outros). Evidentemente esses documentos interdependem, formando um sistema de documentos. Por sua vez, cada documento é formado por uma ou mais representações, e que também interdependem, formando, por sua vez, um sistema de representações. Cada representação é redigida em uma linguagem de representação. Por exemplo, o projeto de uma estrutura de dados complexa pode ser representado por um diagrama que modela a estrutura (linguagem de representação A), um conjunto de diagramas que exibem exemplos de instâncias dessa estrutura (linguagem de representação B), um conjunto de assertivas estruturais que estabelecem as regras que os elementos do modelo da estrutura devem satisfazer (linguagem de representação C e, provavelmente, um texto livre que descreve a essência da estrutura e a sua finalidade (linguagem de representação D). Portanto, para descrever um sistema de software utiliza-se um sistema de linguagens de representação.

As diversas linguagens de representação utilizadas necessariamente interdependem. O correspondente grafo tem por vértices as linguagens e por arestas as interdependências entre elas e é necessariamente conexo. As interdependências das linguagens implicam as interdependências das correspondentes representações. Consequentemente, ao modificar uma dada representação, as representações com que interdepende, ou seja, as *representações vizinhas*, terão possivelmente que ser alteradas também, produzindo, assim, uma onda (*ripple*) de alterações que permeia o conjunto de todas as representações existentes no momento em que a alteração é realizada. Quanto mais abrangente for essa onda, maior será o volume de retrabalho. Se, como ocorre frequentemente, as alterações não forem devidamente propagadas, teremos ao final um conjunto de representações incoerentes entre si e também incoerentes com o sistema implementado. Essas incoerências são conhecidas por *dívidas técnicas*. Em geral, o efeito das dívidas técnicas é uma dificuldade de manter e/ou evoluir o sistema. Essa dificuldade tende a ser proporcional ao "tamanho da incoerência". Tal como dívidas financeiras, dívidas técnicas tendem a tornar mais trabalhosa a manutenção à medida que o tempo passa, a menos que se tome alguma iniciativa para "pagar a dívida", ou seja, remover as causas da dívida técnica.

2. Problema a resolver

Suponha que você queira desenvolver um *componente genérico* capaz de realizar a reserva de assentos. Esse componente deve poder atender a cinemas, ou teatros, ou restaurantes. Evidentemente a generalidade é alcançada por intermédio de uma variedade de tabelas de configuração que precisarão ser identificadas e especificadas. Essas tabelas configuram cada local em que será possível reservar

assentos. Um exemplo de um sistema de reserva de assentos pode ser observado no sistema **`ingresso.com`**. Uma das páginas desse sistema exibe um desenho mostrando a localização dos assentos existentes, marcando aqueles que estão livres e os que já foram reservados.

O componente de reserva de assentos é ativado após realizar-se a seleção do local e a data e hora. Por exemplo, para filmes: que sala, que data e que horário. O componente interage com o usuário com o objetivo de reservar lugares em concordância com os interesses do usuário *reservador*. O componente deve poder produzir: o estado inicial (atual) do espaço total e o estado final do espaço total, informando o que já está ocupado e o que ainda está disponível, e também uma lista dos lugares reservados vinculados aos clientes (ex. expectadores).

3. Condições

Cada participante do grupo, deve registrar cuidadosamente na *sua planilha* EXCEL cada atividade (ex. produzir documento X) e as tarefas dessas atividades (ex. produzir uma das representações do documento, alterar a representação, controlar a qualidade da representação). Deve ainda indicar se a tarefa corresponde a retrabalho inútil ou não. Lembre-se, retrabalho inútil tem a ver com correção de trabalho já realizado. Cabe salientar que são raríssimos os casos em que não ocorra algum retrabalho inútil. De maneira geral, ajustar ou corrigir uma representação é retrabalho inútil, adicionar coisas a ela não é.

4. Ações

Siga cuidadosamente a ordem das ações a seguir.

1. Defina o dicionário de dados (um tipo de representação). Possivelmente será necessário corrigir, ou ajustar, e/ou complementá-lo à medida que o trabalho for sendo elaborado.
2. Projete a organização das tabelas de configuração.
3. Especifique os requisitos funcionais e pelo menos 4 requisitos não funcionais ou inversos do componente.
4. Produza 4 exemplos de uso (ver ATDD) para algumas das características do componente.
5. Usando casos de uso tabulares projete o comportamento do componente.
6. Quais foram as linguagens de representação que você utilizou? Referencie onde estão definidas as linguagens não naturais, ou caso sejam *ad hoc*, descreva-as.
7. Membros do grupo atuarão como revisores. Eles deverão registrar os critérios de controle da qualidade que irão guiar a verificação e devem produzir *laudos detalhados*. As revisões devem utilizar os seguintes pontos de vista:
    - sistema que utilizará o componente;
    - reservador que interagirá com o componente;
8. Após as revisões faça as alterações para assegurar que o conjunto de representações forme um todo coerente. Não devem remanescer dívidas técnicas. Justifique por que você acredita que não haja dívidas técnicas remanescentes.
9. Agora que você tem uma especificação aparentemente completa e correta, o cliente solicita uma alteração. O componente agora deverá ser capaz de realizar as reservas de lugar tendo em vista características das pessoas. Por exemplo: usa cadeira de rodas, tem elevado sobre peso, possui dificuldade de locomoção, etc. Realize as alterações necessárias para satisfazer o pedido do cliente. Registre tudo o que foi modificado e especificamente o tempo que levou fazê-lo.

5. Condições de entrega

Leia os critérios de correção de trabalhos que se encontram ao final do documento que descreve a disciplina.

Os trabalhos devem ser realizados em grupos de 2 ou 3 alunos. Menos ou mais perde 1 ponto.

O *string* de *assunto da mensagem* de envio do resultado do trabalho deve obedecer ao padrão **`INF1413_Trabalho1_<grupo>`** em que **`<grupo>`** é formado pelo conjunto de identificadores de cada membro do grupo. O identificador de um aluno é uma sequência de 2 ou 3 letras (usualmente as iniciais do nome).

Todos os arquivos de texto entregues devem ser Microsoft Word (**`.doc`**, ou **`.docx`**) e as planilhas devem ser EXCEL. O conjunto de arquivos a serem entregues deve estar "zipado" em um único arquivo **`.zip`**. O nome do arquivo deve obedecer ao padrão: **`INF1413_Trab1_<grupo>.zip`**.

Cada dia útil de atraso perde 1 ponto. Lembre-se que sábado e dia enforcado é dia útil.

6. Critérios de correção complementares

- Qualidade do texto – o texto deve estar (quase) correto do ponto de vista ortográfico e sintático (uso da língua portuguesa).
- O texto deve ser sucinto e fácil de ler e de compreender.
- Qualidade da verificação – o texto deve demonstrar que a verificação de sua qualidade foi feita com esmero, procurando identificar todos os defeitos reais.