PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

PUC Minas Virtual

Pós-graduação *Lato Sensu* em Engenharia de *Software*

Trabalho de Conclusão de Curso

GESPROJ

Palloma Stéphanne Silva Brito

Belo Horizonte
Abril/2022

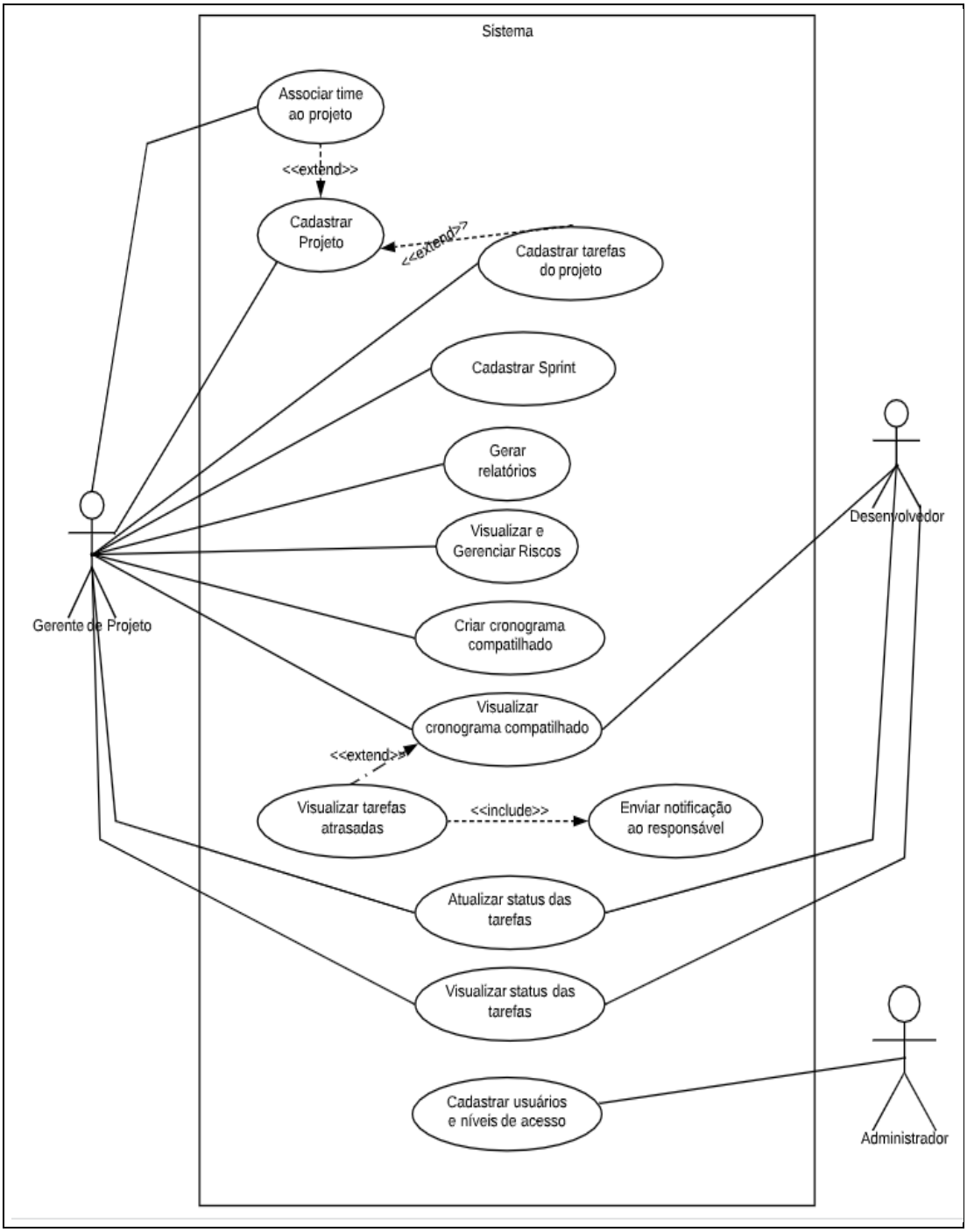
Trabalho de Conclusão de Curso

Sumário

1. Cronograma de trabalho

Datas		Atividade / Tarefa	Produto / Resultado
De	**Até**		
28/02/2022	28/02/2022	1. Assistir à videoaula “Visão Geral do TCC”.	Visão geral sobre o escopo do trabalho.
01/03/2022	01/03/2022	2. Leitura do roteiro do trabalho de conclusão de curso.	Entendimento de todo o passo para elaboração do TCC.
02/03/2022	03/03/2022	3. Leitura das duas opções de escopo fornecidas e escolha de uma delas.	Entendimento das duas opções e escolha do “*Sistema de gerenciamento de projetos*”.
07/03/2022	08/03/2022	4. Elaborar o cronograma de atividades do TCC	Cronograma de trabalho, contendo as datas previstas para cada atividade.
09/03/2022	11/03/2022	5. Elaborar o diagrama de casos de uso, contendo todos os atores e casos de uso identificados para a aplicação.	Diagrama de caso de uso.
14/03/2022	15/03/2022	6. Descrever os requisitos não-funcionais previstos para a aplicação.	Lista de requisitos não-funcionais.
16/03/2022	24/03/2022	7. Criação do protótipo navegável do sistema, mostrando a tela inicial e as de três casos de uso principais.	Protótipo navegável do sistema.
25/03/2022	25/03/2022	8. Visualização dos materiais de apoio para desenvolver as próximas etapas.	Entendimento melhor sobre os Diagramas UML.
28/03/2022	28/03/2022	9. Elaborar Diagrama de classes de domínio de todo o sistema.	Diagrama de classes de domínio do sistema.
29/03/2022	30/03/2022	10. Escolher o padrão arquitetural da aplicação e as tecnologias que serão utilizadas.	Escolha do padrão MVC e definição de tecnologias.
31/03/2022	31/03/2022	11. Elaborar Diagrama de componentes de todo sistema.	Diagrama de componentes.
01/04/2022	01/04/2022	12. Descrição dos componentes indicados para a arquitetura do sistema.	Detalhamento dos componentes na tabela.
04/04/2022	04/04/2022	13. Elaborar Plano de Testes para pelo menos três casos de uso do sistema.	Tabela com descrição dos testes.
05/04/2022	06/04/2022	14. Elaborar Planilha de contagem de pontos de função, considerando todo o sistema.	Planilha de contagem de pontos de função.
07/04/2022	07/04/2022	15. Gravar vídeo de apresentação do protótipo navegável desenvolvido.	Vídeo do protótipo.
08/04/2022	08/04/2022	16. Ajustes na formatação do documento e revisões gerais.	Trabalho revisado.
13/04/2022	13/04/2022	17. Armazenamento e disponibilização do trabalho no Github.	Link do repositório.
13/04/2022	13/04/2022	18. Submissão do trabalho no Ambiente Virtual de Aprendizagem.	Entrega final do trabalho.

2. Diagrama de casos de uso



3. Requisitos não-funcionais

A seguir, encontram-se descritos os requisitos não funcionais da aplicação:

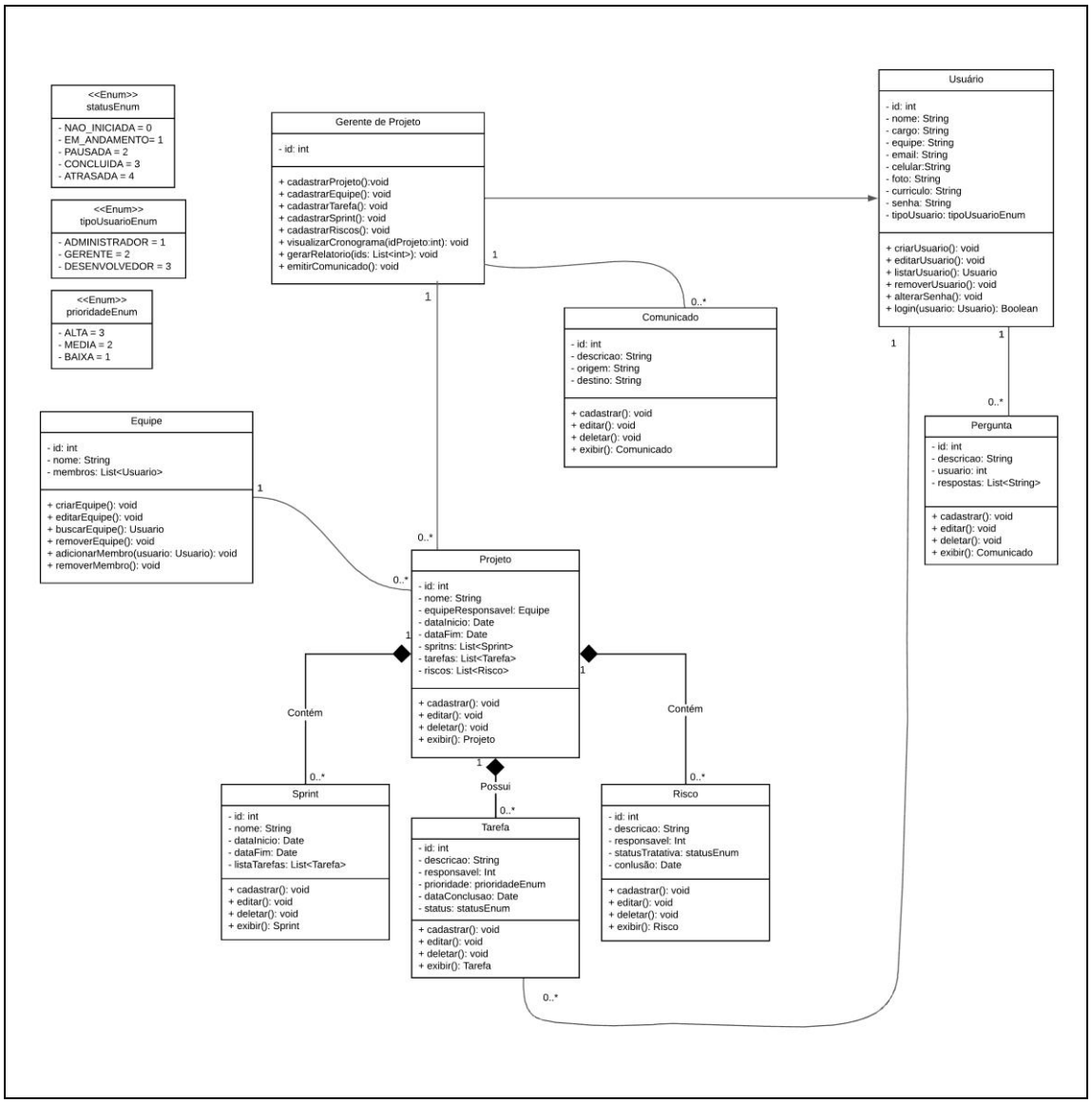
- O sistema deve poder ser acessado por meio da Web e em qualquer dispositivo móvel, tais como tablets e smartphones;
- O sistema deve se comunicar com serviços através do consumo de APIs;
- O sistema deve proteger dados e funcionalidades com um mecanismo de segurança confiável, criptografando senhas antes de inseri-las no banco;
- O sistema deve estar disponível 24/7;
- O sistema deve processar requisições do usuário em, no máximo, 500 milissegundos.
- O sistema deve suportar pelo menos 100 usuários conectados ao mesmo tempo;
- O Gerente de Projetos deve ser capaz de usar todas as funções do sistema após quatro horas de treinamento. Após esse treinamento, o número médio de erros cometidos não deve exceder a dois por hora de uso do sistema.

4. Protótipo navegável do sistema

O protótipo navegável do sistema foi desenvolvido utilizando o *Figma* e pode ser acessado clicando neste link. Nele, encontram-se diversos casos de usos da aplicação.

O vídeo contendo uma apresentação do sistema encontra-se neste link e também no repositório do Github.

5. Diagrama de classes de domínio



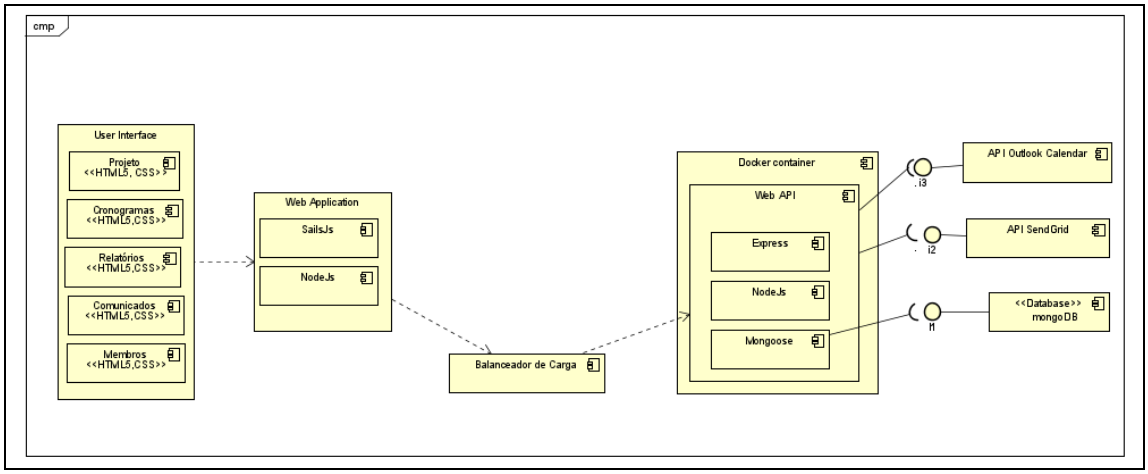
6. Modelo de componentes

6.1. Padrão arquitetural

Para este projeto, optou-se por utilizar o padrão arquitetural MVC, possibilitando a divisão do projeto em camadas muito bem definidas. Em outras palavras, a comunicação entre interfaces do sistema e regras de negócios serão definidas através de um controlador. Quando um evento for executado na interface do sistema, como por exemplo a criação de um novo projeto, a interface irá se comunicar com o controlador que por sua vez se comunicará com as regras de negócios do sistema.

Pensando nisso, será utilizado para implementação do sistema o *Sails.js*. O *Sails.js* (ou simplesmente Sails) é um framework de aplicação web *model–view–controller* (MVC) desenvolvido sobre o ambiente *Node.js*. Ele foi projetado para facilitar a criação de aplicativos Web e APIs personalizadas em *Node.js.* O *frontend* do sistema será desenvolvido utilizando HMTL5, CSS e JavaScript. O MongoDB será utilizado como banco de dados e a utilizaremos a biblioteca Mongoose como interface. O backend da aplicação executará através do Docker. A Web API será desenvolvida utilizando o NodeJs e o Express. O sistema estará integrado à API REST de Calendário do Outlook e à API do SendGrid para envio de e-mails. Jest para testes de unidade e integração.

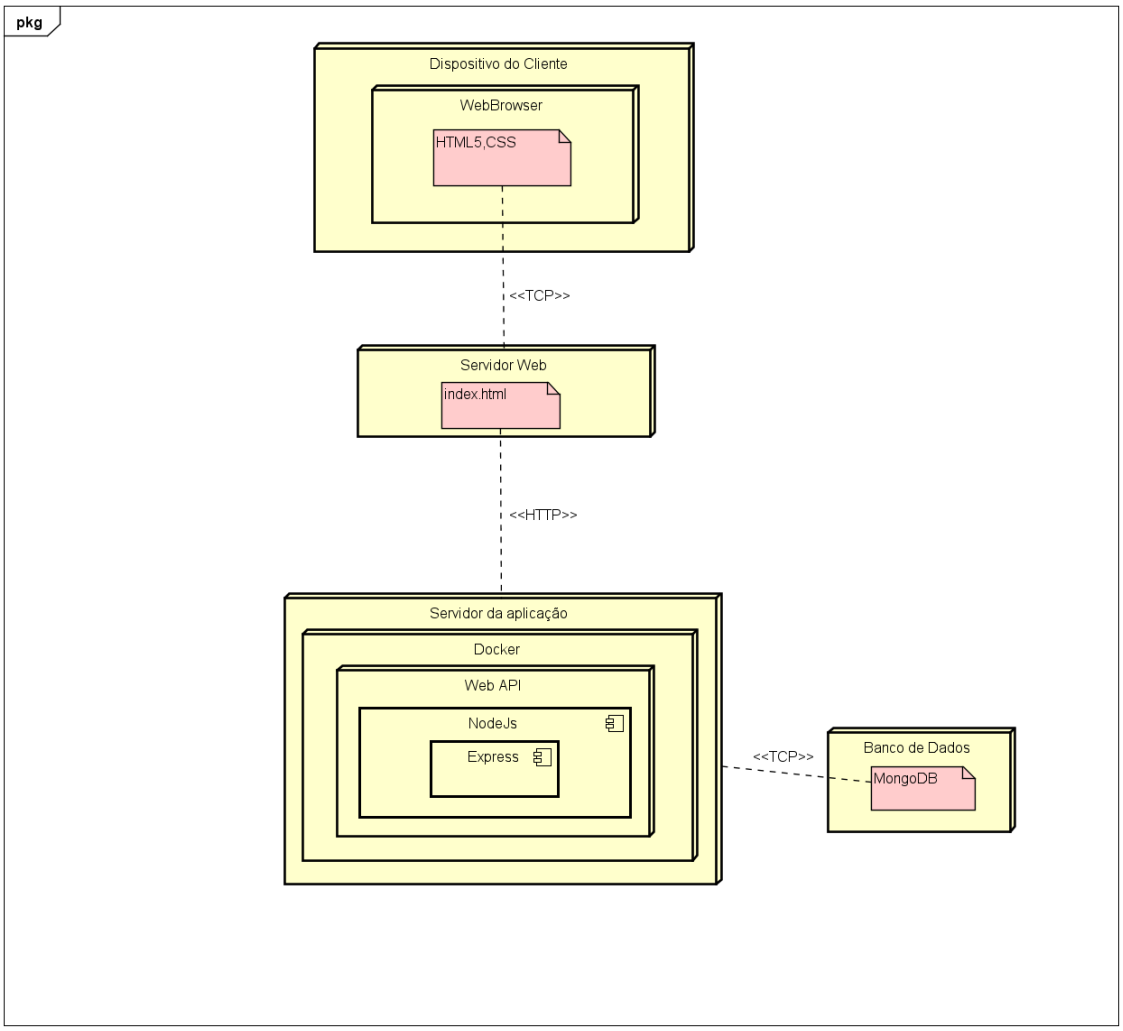
6.2. Diagrama de componentes



6.3. Descrição dos componentes

Número	Componente	Descrição
1	User Interface	A *User Inferface* representa nossa View e será desenvolvida em Javascript, HTML5 e CSS. É através deste componente que o usuário irá interagir para acessar os recursos da nossa aplicação.
2	Web App	A Web App simboliza a aplicação Web em si, que contará com os frameworks SailsJs e NodeJs no seu desenvolvimento.
3	Balanceador de carga	O Balanceador de carga será utilizado para distribuir uniformemente a carga de trabalho da API, a fim de evitar sobrecarga, otimizar a utilização de recursos, maximizar o desempenho e minimizar o tempo de resposta.
4	Web API	A Web API, ou Interface de Programação de Aplicativos, conterá a maior parte dos serviços que serão disponibilizados através do sistema. Ela será desenvolvida utilizando o padrão REST e o Express e NodeJs como frameworks. Os principais serviços oferecidos pelo sistema, como CRUD de Usuários e Projetos por exemplo, serão acessados através de endpoints dessa API. Ela rodará em Docker.
5	API Outlook Calendar	A Web API se comunicará com a API do Outlook Calendar para facilitar a organização e exibição do Calendário de Tarefas para cada usuário e, também, para os projetos como um todo.
6	API SendGrid	A WEB API também se comunicará com a API SendGrid para envio de e-mails automáticos e personalizados aos usuários.
7	mongoDB	O mongoDB será o banco de dados utilizado como um componente no sistema para persistência de dados, por serem mais flexíveis e escaláveis.

7. Diagrama de implantação



8. Plano de Testes

Número	Caso de uso	Objetivo do caso de teste	Entradas	Resultados esperados
1	Cadastrar novo usuário	Validar alertas de restrições dos campos de preenchimento.	- Clicar no menu “Membros” e depois em “+”. - Inserir um nome com número de caracteres além do permitido (máximo 70)e deixar o campo e-mail em branco. Preencher demais campos. - Clicar no botão “Salvar”.	Deve ser exibido na tela um alerta de erro, informando os campos que não foram preenchidos corretamente.
2	Cadastrar novo usuário	Validar se o registro dos campos preenchidos foi feito corretamente no banco de dados.	- Clicar no menu “Membros” e depois em “+”. - Deve-se inserir os dados de um novo usuário, preenchendo todos os campos corretamente. - Clicar no botão “Salvar”.	O novo usuário deve aparecer listado no no menu “Membros” com seus respectivos dados cadastrados.
3	Visualizar relatório de projeto	Validar que todos as atividades vinculadas ao projeto sejam apresentadas.	- Clicar em “Relatório de todas as atividades, por projeto” e selecionar um dos projetos na lista suspensa.	Deve ser exibido o gráfico de “Desenvolvimento planejado x real”, constando toas as atividades vinculadas ao projeto.
4	Visualizar relatório de projeto	Validar que, caso não haja nenhuma atividade vinculada ao projeto, nenhum gráfico seja gerado.	- Clicar em “Relatório de todas as atividades, por projeto” e selecionar um dos projetos que não possua atividade vinculada.	Deve ser exibida uma mensagem dizendo que não há nenhuma atividade cadastrada para aquele projeto.
5	Recuperar senha de acesso ao sistema	Validar que as informações para recuperação de senha sejam enviadas ao e-mail cadastrado.	- Acessar o link “Esqueceu sua senha” na página inicial. - Digitar um e-mail vinculado a um usuário do sistema.	- Deve ser exibida uma mensagem de que as informações para a recuperação foram enviadas para o e-mail informado.

			- Clicar em “Confirmar”,	- O usuário deve receber as informações no e-mail.
6	Recuperar senha de acesso ao sistema	Validar que, caso o e-mail informado não esteja cadastrado na base de dados, nenhum e-mail seja enviado.	- Acessar o link “Esqueceu sua senha” na página inicial. - Digitar um e-mail não vinculado a nenhum usuário do sistema. - Clicar em “Confirmar”,	- Deve ser exibido na tela um alerta de erro, informando que não possui nenhum usuário vinculado ao e-mail informado.

9. Estimativa de pontos de função

A planilha de estimativa de pontos de função está disponível neste link.

10. Informações da implementação

Todo o conteúdo deste trabalho encontra-se disponível neste repositório do Github.

11. Referências

SOMMERVILLE, Ian. **Engenharia de software**. São Paulo: Pearson Addison-Wesley, 9a. Edição, 2011.

PRESSMAN, ROGER S. **Engenharia de Software, Uma Abordagem Profissional**, 8ª. Edição,McGraw Hill – Artmed, 2016

LARMAN, Graig. **Utilizando UML e Padrões: Uma introdução a análise e ao projeto orientados a objetos**. Porto Alegre: Bookman, 3a Edição, 2011

DELAMARO, Márcio; Maldonado, José Carlos; Jino, Mário. Introdução ao Teste de Software. 1ª edição. Elsevier. 2007