Planejamento de Testes a partir de Casos de Uso

De que se trata o artigo:

O artigo apresenta uma estratégia indicando como testes podem ser obtidos a partir dos casos de uso especificados para um projeto.

Para que serve:

Durante o desenvolvimento de um software, diversas estratégias para teste podem ser aplicadas. Ao longo deste artigo iremos adotar uma estratégia de geração de testes baseada em especificação, representada pelos casos de uso de um sistema. Assim, partiremos desta informação para a geração de casos e procedimentos (roteiros) de teste. Desta forma, este artigo apresenta de forma prática como efetuar a geração de casos de teste.

Em que situação o tema é útil:

A cada dia as atividades de teste de software vêm tendo sua importância aumentada dentro das organizações de desenvolvimento de software. O tema deste artigo agrega conhecimento a este cenário através do apoio às atividades do analista de testes.

Se observarmos nos diferentes livros tradicionais de Engenharia de Software, veremos que sempre existe um capítulo ou seção destinado a Teste de software. Porém, eles normalmente apresentam apenas informações básicas sobre esta atividade, como por exemplo, os diferentes níveis de teste que podem ser aplicado (ex: unidade, integração ou sistema), as técnicas de teste que podem ser aplicadas (ex: técnica funcional ou estrutural) e os critérios para seleção dos testes associados a estas técnicas. Por exemplo, no artigo "Introdução a Teste de Software" publicado na edição 01 da Engenharia de Software Magazine discutimos sobre alguns desses critérios: Particionamento em classes de equivalência, Análise do Valor Limite e Grafo de Causa-Efeito. Para cada critério, vimos como aplicá-los e um exemplo da sua aplicação para a geração de casos de teste. Mais informações sobre esses critérios de seleção dos testes podem ser obtidas em ROCHA (2001).

Arilo Cláudio Dias Neto
ariloclaudio@gmail.com
É Bacharel em Ciência da Computação formado na Universidade Federal do Amazonas, Mestre em Engenharia de Sistemas e Computação formado na COPPE/UFRJ, e atualmente está cursando doutorado na área de Engenharia de Software da COPPE/UFRJ. Possui 5 anos de experiência em análise, desenvolvimento e teste de software. É editor técnico das Revistas SQL Magazine e WebMobile, gerenciadas pelo Grupo DevMedia.

No entanto, no desenvolvimento de um software real normalmente os problemas são bem mais complexos do que aqueles tradicionalmente usados quando estamos conhecendo esses critérios para seleção dos testes (ex: indicar qual o maior valor em um conjunto, indicar se um campo número só contém caracteres válidos, dentre outros). Normalmente os problemas a serem resolvidos são representados através de cenários, que podem ser facilmente representados por Diagramas de Casos de Uso da UML (www.uml.org) aliada a uma descrição do que cada caso de uso deve fazer.

Ao longo deste artigo iremos discutir uma possível estratégia indicando como testes podem ser obtidos a partir dos casos de uso especificados para um projeto. Entendemos que podem existir diferentes estratégias para isso, então iremos apresentar apenas uma possibilidade que pode ser facilmente aplicada para o teste de formulários de cadastro, normalmente existentes em sistemas de informação.

Estratégias de Teste de Software

Durante o desenvolvimento de um software, diversas estratégias para teste podem ser aplicadas. De acordo com PRESSMAN (2005), essas estratégias podem ser categorizadas da seguinte forma:

- **Baseadas em implementação:** utiliza o código como elemento para a geração dos testes. É uma atividade cara, sob o ponto de vista de recursos necessários para a sua realização, e bastante complexa quando o tamanho do código se torna bastante grande. É totalmente dependente de apoio ferramental.
- **Baseadas em especificação:** utiliza um documento de especificação como base para geração dos testes. Assim, tenta-se cobrir as imposições e restrições descritas nos requisitos estabelecidos para o sistema. A automação da geração dos testes nesse caso é mais complicada, caso não se tenha um formalismo para a elaboração da especificação do sistema.
- **Baseadas em modelos:** é uma sub-categoria de estratégias baseada em especificação. Utiliza modelos desenvolvidos ao longo do processo de desenvolvimento que representam o comportamento ou estrutura do software como base para a geração dos testes. Facilita a geração automática dos testes, porem é completamente dependente da facilidade para a construção do modelo adotado e de sua qualidade (corretude).

Cada estratégia apresentada possui sua aplicabilidade, vantagens e desvantagens. Não é propósito deste artigo discutir qual seria a estratégia mais adequada. Ao longo deste artigo iremos adotar uma estratégia de geração de testes baseada em especificação, representada pelos casos de uso de um sistema. Assim, partiremos desta informação para a geração de casos e procedimentos (roteiros) de teste. Relembrando os conceitos associados a esses elementos, conforme descrito no artigo "Introdução a Teste de Software" publicado na edição 01 da Engenharia de Software Magazine, temos:

- **Caso de Teste.** Descreve uma condição particular a ser testada e é composto por valores de entrada, restrições para a sua execução e um resultado ou comportamento esperado (CRAIG e JASKIEL, 2002).
- **Procedimento (Roteiro) de Teste.** É uma descrição dos passos necessários para executar um caso (ou um grupo de casos) de teste (CRAIG e JASKIEL, 2002).

Porém, antes de descrevermos como obter testes a partir da especificação de casos de uso, precisamos primeiramente entender melhor este documento que servirá como entrada para a geração dos testes: a Especificação de Casos de Uso. Este artefato do processo de desenvolvimento servirá como **oráculo** para os testes, ou seja, os resultados obtidos pelo sistema devem sempre estar de acordo com os resultados previstos neste documento. A próxima seção irá discutir sobre a estrutura e conteúdo deste artefato.

Especificação de Casos de Uso

Um Caso de Uso representa uma unidade discreta da interação entre um usuário (humano ou máquina) e o sistema. Ele representa as funcionalidades que um sistema deve prover e uma indicação que qual elemento, denominado de ator, pode acessar uma determinada funcionalidade. Um ator é um humano

Caso de Uso: *<nome>*
Ator: *<nome do ator>*
Pré-Condições: *<pré-condições>*
Fluxo:
1. *<descrição de cada passo>*
2. *<descrição de cada passo>*
3. ...

Pós-Condições: *<pós-condições>*
Fluxos Alternativos:
X.1 *<fluxo alternativo a partir de um determinado passo X>*
Regra de Negócio:
(1) *<regra de negócio associada ao caso de uso>*

Exceções: *<exceções de execução do caso de uso>*

Figura 1. Template para especificação de caso de uso

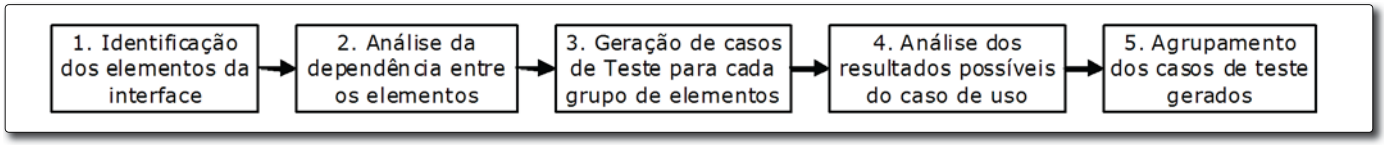

Figura 2. Processo de Geracao de Testes a partir de Casos de Uso

Quadro 1. Descrição do Caso de Uso

Caso de Uso: Cadastrar Funcionário
Ator: Administrador do Sistema
Pré-Condições: O administrador deve estar autenticado no sistema e deve ter acesso a este caso de uso
Fluxo:
1. Ator escolhe a opção Cadastrar Funcionário
2. Sistema abre um formulário a ser preenchido.
3. Ator preenche o formulário com os dados do funcionário a ser cadastrado.
4. Sistema valida os dados do funcionário e solicita confirmação.
5. Ator confirma o cadastro.
6. Sistema armazena os dados do novo funcionário.

Pós-Condições: O funcionário deve estar cadastrado no sistema.
Fluxos Alternativos:
5.1 Ator não confirma o cadastro.
5.2 Sistema volta ao passo 3.
Regra de Negócio:
(1) os campos a serem preenchidos para um formulário são: nome, data de nascimento, cargo (motorista, médico ou técnico em informática), CNH (caso seja motorista), CRM (caso seja médico), naturalidade (brasileira ou estrangeira), CPF (caso seja brasileiro), Passaporte (caso seja estrangeiro) e salário.
(2) os campos nome, data de nascimento, cargo, naturalidade e salário são obrigatórios.
(3) o campo CNH é obrigatório caso o cargo do funcionário seja "motorista" e o campo CREA é obrigatório caso o cargo seja "engenheiro".
(4) o campo CPF é obrigatório caso o funcionário seja "brasileiro" e o campo Passaporte é obrigatório caso seja "estrangeiro".
(5) cada tipo cargo possui uma faixa de salário possível que deve ser respeitada. As faixas são:
- Motorista: entre R$ 1000 e R$ 3000;
- Médico: entre R$ 3000 e R$ 10000;
- Técnico em Informática: entre R$ 1500 e R$ 7000;

Exceções: Dados não preenchidos corretamente ou salário fora da faixa de valores do cargo correspondente.

ou entidade máquina que interage com o sistema para executar um trabalho no contexto do sistema.

Este modelo foi incluído entre os diagramas disponíveis na UML. No entanto, o diagrama sozinho é totalmente limitado e não apresenta qualquer informação sobre o significado de tal funcionalidade. Sendo assim, cada Caso de Uso deve possuir uma descrição que deve descrever a funcionalidade que irá ser construída no sistema proposto.

É importante notar que o caso de uso não descreve como o software deverá ser construído, mas sim como ele deverá se comportar quando estiver pronto. Um software freqüentemente é um produto complexo, e sua descrição envolve a identificação e documentação de vários casos de uso, cada um deles descrevendo uma "fatia" do que o software ou uma de suas partes deverá oferecer.

Uma descrição de um caso de uso deve ser formada pelos seguintes elementos:

– **Nome:** Identificador inequívoco do caso de uso, deve ser escrito em formato de verbo/substantivo e ser suficiente para o utilizador perceber a que se refere o caso de uso.

– **Atores:** perfis de usuários que executam o caso de uso.

– **Pré-condições:** restrições para iniciar a execução de um caso de uso.

– **Seqüência de Ações (Fluxo principal):** passos ordenados para execução de um caso de uso.

– **Pós-condições:** condição final a ser estabelecida ao final da execução do caso de uso.

– **Fluxos Alternativos:** fluxos de ações que ocorrem paralelamente ao fluxo principal, dada uma ação específica.

Regras de negócio: restrições (regras) para execução de um ou mais passos do fluxo principal ou alternativo.

– **Exceções:** estados inválidos para o sistema.

A **Figura 1** descreve um exemplo de *template* para a especificação de um caso de uso.

Gerando Testes a partir de Casos de Uso

Uma vez garantida a qualidade da Especificação de Casos de Uso, artefato de entrada para a geração dos testes, devemos seguir alguns passos visando a obtenção de casos e procedimentos de teste para avaliação de cada funcionalidade do sistema, representada pelos casos de uso. A **Figura 2** representa os passos que compõem esta estratégia.

Para entendermos melhor esses passos, seguiremos um estudo de caso referente a um caso de uso bastante comum no nosso dia-a-dia de desenvolvedores: um formulário de cadastro.

Estudo de Caso para Geração de Testes: Cadastrar Funcionário

O caso de uso a ser testado está descrito no **Quadro 1**.

Além dessas informações, algumas suposições devem ser feitas para viabilizar o planejamento dos testes:

a) Na interface do sistema, o preenchimento incorreto de um campo será mostrado item a item, ou seja, se todos os campos forem preenchidos incorretamente, o sistema apresentará mensagem de dados inválidos para todos eles.

b) Existem campos que são obrigatórios a partir de certas condições (ex: CNH para motoristas), mas o sistema não impede que esse campo seja preenchido em outras situações.

c) Iremos assumir que os campos CNH e CRM não possuem regra de formação, como sabemos que existe para CPF. Assim, se qualquer valor for preenchido, ele será considerado válido.

d) Os campos numéricos só aceitarão valores numéricos (não preciso testar o contrário) e o campo data só aceitará datas (válidas ou inválidas).

e) Os campos cargo e naturalidade serão opções a serem escolhidas (com alguma das opções já selecionada previamente). Assim, estes campos nunca serão deixados em branco.

A seguir iremos passo a passo gerar os testes para este formulário.

Passo 1: Identificação do Elementos de Interface

Os elementos de interface que compõem um caso de uso são, por exemplo, campos, menus, links ou botões. Precisamos identificá-los para podermos iniciar o processo de geração dos testes para o caso de uso.

No nosso estudo de caso, partiremos da idéia de que o caso de uso só é executado após escolhermos a opção CADASTRAR FUNCIONÁRIO no menu principal da aplicação. Assim, os elementos de interface que fazem parte dessa interface são:

- Nome (String);
- Data de Nascimento (tipo Data);
- Cargo (lista de opções);
- CNH (String);
- CRM (String);
- Salário (Numérico);
- Nacionalidade (lista de opções);
- CPF (String);
- Passaporte (String).

Passo 2: Análise da Dependência entre os Elementos

Em seguida, precisamos observar as dependências entre os elementos de interface, como por exemplo, uma regra indicando que um campo só pode ser preenchido caso um outro campo tenha sido preenchido previamente.

No nosso estudo de caso, olhando as regras de negócio observamos as seguintes dependências:

- Os campos CNH, CRM e Salário dependem do Cargo selecionado.
- Os campos CPF e Passaporte dependem da Naturalidade selecionada.

Assim, este caso de uso possui 4 grupos de elementos independentes:

- Nome;
- Data de Nascimento;
- Cargo, CNH, CRM e Salário;
- Naturalidade, CPF e Passaporte.

Passo 3: Geração dos Casos de Teste para cada Grupo de Elementos

Identificados os grupos de elementos, devemos aplicar algum dos critérios de

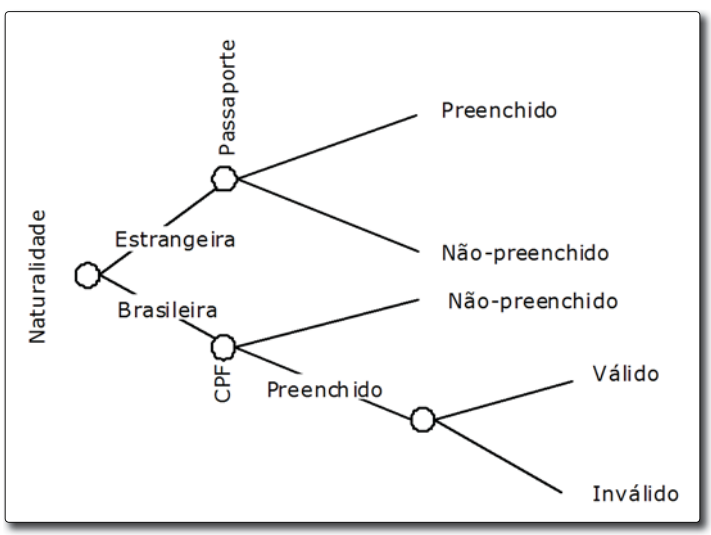

Figura 3. Grafo de Causa-Efeito para o Grupo (Naturalidade, CPF, Passaporte)

seleção de testes funcionais que vimos no artigo "Introdução a Teste de Software" para a geração de casos de teste para cada grupo. Essa seria uma forma de dividir o problema em partes menores para simplificar o processo de geração dos testes.

Os casos de teste gerados para cada grupo são os seguintes:

• (Nome): aplicamos o critério de Particionamento em Classes de Equivalência e obtivemos dois casos de teste, uma para a classe válida e outro para classe inválida (não preenchido).

T_{nome} = {(" ", INVÁLIDO); ("Arilo",VÁLIDO)}

• (Data de Nascimento): aplicamos o critério de Particionamento em Classes de Equivalência e obtivemos três casos de teste, uma para a classe válida, outro para data inválida (classe inválida) e outro para data não preenchida (classe inválida).

T_{data}={("",Inválido);("30/02/1982",Inválido); ("13/09/1982",Válido)}

• (Nacionalidade,CPF,Passaporte): aplicamos o critério de Grafo de Causa-Efeito e obtivemos cinco casos de teste, dois para o caso de nacionalidade estrangeira (um com passaporte preenchido e um com passaporte em branco) e três para o caso da nacionalidade brasileira (um com CPF não preenchido, um com CPF inválido [dígito verificador incorreto] e outro com CPF preenchido e válido), de acordo com a **Figura 3**.

$T_{nacionalidade}$ = {("Brasileira", "", --, [INVÁLIDO]); ("Brasileira", "782622652-97", --; [INVÁLIDO]); ("Brasileira", "636.112.337-57", ""; [VÁLIDO]), ("Estrangeira", --, "", [INVÁLIDO]); ("Estrangeira", "", "23243"; [VÁLIDO])}

• (Cargo,CNH,CRM,Salário): aplicamos o critério de Grafo de Causa-Efeito, sendo que para o campo Salário aplicamos ainda o critério de Análise do Valor

#	Caso de Teste (N)	Caso de Teste (D)	Caso de Teste (C)	Caso de Teste (N)	Resultado Esperado
1	""	""	("Motorista", "", --, --)	("Brasileira", "", --)	DADOS INVÁLIDOS
2	"Arilo"	30/02/1980	("Motorista","123456334", --, 999.99)	("Brasileira", "782622652-97", --)	DADOS INVÁLIDOS
3	""	14/05/2007	("Motorista","123456334", --, 3000.01)	("Brasileira", "636.112.337-57","")	DADOS INVÁLIDOS
4	"Arilo"	""	("Médico", "", --, --)	("Estrangeira", --, "")	DADOS INVÁLIDOS
5	""	30/02/1980	("Médico", --, "7625-2", 2999.99)	("Estrangeira", "", "23243")	DADOS INVÁLIDOS
6	"Arilo"	14/05/2007	("Médico", --,"7625-2"; 10000.01)	("Brasileira", --, "")	DADOS INVÁLIDOS
7	""	""	("Técnico", "", "", 1499.99)	("Brasileira", "782622652-97", --)	DADOS INVÁLIDOS
8	"Arilo"	30/02/1980	("Técnico", "", "", 7000.01)	("Estrangeira", --, "")	DADOS INVÁLIDOS
9	"Arilo"	14/05/2007	("Motorista","123456334",--, 1000)	("Brasileira", "636.112.337-57","")	CADASTRO REALIZADO
10	"Arilo"	14/05/2007	("Motorista","123456334", --, 3000)	("Estrangeira", "", "23243")	CADASTRO REALIZADO
11	"Arilo"	14/05/2007	("Médico", --, "7625-2", 3000)	("Brasileira", "636.112.337-57","")	CADASTRO REALIZADO
12	"Arilo"	14/05/2007	("Médico", --,"7625-2"; 10000)	("Estrangeira", "", "23243")	CADASTRO REALIZADO
13	"Arilo"	14/05/2007	("Técnico", "", "", 1500)	("Brasileira", "636.112.337-57","")	CADASTRO REALIZADO
14	"Arilo"	14/05/2007	("Técnico", "", "", 7000)	("Estrangeira", "", "23243")	CADASTRO REALIZADO

Tabela 1. Conjunto final de casos de teste para o caso de uso CADASTRAR FUNCIONÁRIO

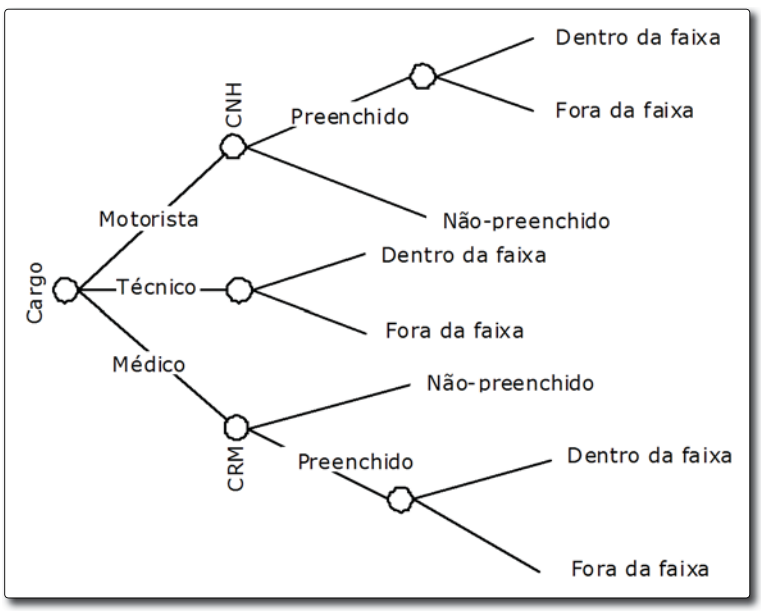

Figura 4. Grafo de Causa-Efeito para o Grupo (Cargo,CNH,CRM,Salário)

Limite. Com isso, obtivemos catorze casos de teste, cinco para o caso de cargo ser motorista, cinco para o caso do cargo ser médico e quatro para o cargo técnico em informática (de acordo com a **Figura 4**).

T_{cargo} = {("Motorista", "", --, --; [INVÁLIDO]); ("Motorista", "123456334", --, 999.99; [INVÁLIDO]); ("Motorista", "123456334", --, 1000; [VÁLIDO]); ("Motorista", "123456334", --, 3000; [VÁLIDO]) ; ("Motorista", "123456334"; --, 3000.01; [INVÁLIDO]); ("Médico", "", --, --; [INVÁLIDO]); ("Médico", --, "7625-2", 2999.99; [INVÁLIDO]); ("Médico", --,"7625-2"; 3000; [VÁLIDO]); ("Médico", --,"7625-2"; 10000; [VÁLIDO]) ; ("Médico", --,"7625-2"; 10000.01; [INVÁLIDO]); ("Técnico", "", "", 1499.99; [INVÁLIDO]); ("Técnico", "", "", 1500; [VÁLIDO]); ("Técnico", "", "", 7000; [VÁLIDO]); ("Técnico", "", "", 7000.01; [INVÁLIDO])}

Passo 4: Análise dos resultados possíveis do caso de uso

Gerados os casos de teste para cada grupo de elementos, o passo seguinte consiste na análise dos possíveis resultados providos pelo caso de uso. Para a geração dos casos de teste para um caso de uso, devemos atender às duas regras seguintes:

1. Deve cobrir todos os casos de teste gerados para cada grupo de elemento.
2. Deve existir ao menos 1 caso de teste para cada resultado possível gerado pelo caso de uso.

No nosso estudo de caso, os resultados possíveis da execução do cadastro de funcionários são apenas dois: CADASTRO REALIZADO COM SUCESSO ou DADOS INVÁLIDOS NO FORMULÁRIO a partir da violação de alguma regra de negócio.

Passo 5: Agrupamento dos Casos de Teste Gerados

O último passo é o agrupamento dos casos de teste gerados no passo 3. O agrupamento entre os casos de teste não precisa seguir uma regra, desde que atenda às duas anteriores descritas no passo 4. No entanto, precisamos considerar o seguinte cenário:

- Integrar dois casos de teste de resultados inválidos irá gerar um novo caso de teste também de resultado inválido.
- Integrar dois casos de teste de resultados válidos irá gerar um novo caso de teste também de resultado válido.
- Integrar um caso de teste de resultado inválido com um de resultado válido irá gerar um novo caso de teste de resultado inválido.

Sendo assim, o conjunto completo e mínimo de casos de teste para avaliação deste caso de uso está descrito na **Tabela 1**.

Conclusões

Este artigo apresentou uma possível estratégia para geração de casos de teste a partir de casos de uso, mas é preciso destacar mais uma vez que outras estratégias podem ser adotadas. Ao longo das nossas atividades do dia-a-dia, podemos nos deparar com situações que requeiram uma estratégia diferente da estratégia aqui apresentada. Cabe ao responsável pelos testes tomar a decisão de quais passos irá adotar para a geração dos testes, desde que mantenha em mente o objetivo de gerar testes de qualidade e que realmente possibilitam a avaliação de uma funcionalidade de um software.

Como passo seguinte do processo de teste, devem ser gerados os procedimentos de teste a fim de definir o roteiro/ordem de execução dos casos de teste gerados. As diretrizes para definição dos procedimentos de teste podem ser tema de um próximo artigo em uma edição futura. ●

Referências

CRAIG, R.D., JASKIEL, S. P., "Systematic Software Testing", Artech House Publishers, Boston, 2002.
PRESSMAN, R. S., "Software Engineering: A Practitioner's Approach", McGraw-Hill, 6th ed, Nova York, NY, 2005.
ROCHA, A. R. C., MALDONADO, J. C., WEBER, K. C. et al., "Qualidade de software – Teoria e prática", Prentice Hall, São Paulo, 2001.

Dê seu feedback sobre esta edição!

A Engenharia de Software Magazine tem que ser feita ao seu gosto. Para isso, precisamos saber o que você, leitor, acha da revista!

Dê seu voto sobre este artigo, através do link:

www.devmedia.com.br/esmag/feedback