Casos de testes sobre a Modelagem Informacional de Requisitos

Tiago Araújo Vignoli

Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Bacharelado em Ciência da Computação

Orientador: Prof. Michel Heluey Fortuna

Juiz de Fora, MG
Julho de 2009

Casos de testes sobre a Modelagem Informacional de Requisitos

Tiago Araújo Vignoli

Monografia submetida ao corpo docente do Departamento de Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora, como parte integrante dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Aprovada pela banca constituída pelos seguintes professores:

_____

Prof. Michel Heluey Fortuna – orientador
DSc. Engenharia de Sistemas e Computação, COPPE, 2008

_____

Ciro de Barros Barbosa
Phd. Ciência da Computação, University of Twente Holanda, 2001

_____

Fernanda Claudia Alves Campos
DSc. Engenharia de Sistemas e Computação, COPPE, 1999

Juiz de Fora, MG
Julho de 2009

Agradecimentos

A todos que colaboraram na realização deste estudo e a meu orientador que esteve sempre presente na condução deste trabalho.

Sumário

Lista de Reduções

II	Interface Informacional (de um UC)
MIR	Modelagem Informacional de Requisitos
MOD	Modelo de (classes de) Objetos de Domínio
UC	*Use Case*
UML	Unified Modeling Language

Lista de Figuras

Lista de Tabelas

Resumo

O presente trabalho apresenta a elaboração de casos de testes a partir da especificação técnica de um sistema, expondo as vantagens desta atividade e efetuando uma comparação entre elaborar os casos a partir de um sistema especificado pela Modelagem Informacional de Requisitos e elaborar a partir da Lista de Requisitos e Diagrama de Estados e Transições. Algumas técnicas de testes puderam ser diretamente associadas aos documentos produzidos pela MIR, permitindo concluir que tal modelagem facilita a identificação e planejamento dos testes a serem aplicados sobre um sistema, cuja implementação ainda não foi iniciada.

Palavras-Chave

Casos de Uso, Modelo de Classes, Modelagem Informacional de Requisitos, Casos de testes.

Capítulo 1

Introdução

Ao longo dos anos, diversas técnicas e estratégias de testes têm sido propostas de modo a orientar a atividade de controle de qualidade e aumentar sua eficiência.

A experiência do autor deste trabalho permite dizer que o testador ao longo de sua atividade, adquire um conhecimento sobre o sistema, superior ao próprio analista que o modelou, devido ao fato de conhecer profundamente a especificação, a forma como o sistema produzido funciona e a metodologia de trabalho adotada durante o desenvolvimento. Tal conhecimento permite desenvolver uma visão crítica que, segundo GRAHAM (2002), deve ser explorada antes mesmo da construção do sistema ser iniciada, ou seja, a partir da especificação técnica, conforme também proposto por GELPERIN & HETZEL (1988).

Info cases é uma adaptação da técnica de casos de uso, especialmente projetada para sistemas de informação. Ela privilegia a especificação dos requisitos de informação do sistema, aliviando o modelador de parte da especificação de comportamento normalmente elaborada nos casos de uso. A motivação deste trabalho é prover um procedimento de elaboração de casos de teste voltado para a técnica de info cases, além de efetuar uma comparação entre os casos de testes propostos e os produzidos para um sistema que não foi especificado por info cases.

Para atender a estes objetivos a presente monografia foi dividida da seguinte forma: o capítulo 2 apresenta técnicas de testes; o capítulo 3 introduz a modelagem de requisitos com info cases; o capítulo 4 descreve a proposta construída neste trabalho para a elaboração de casos de testes a partir de info cases, exemplificando-a para um sistema simples de gerência de restaurante, e efetua uma comparação com os casos de teste produzidos para um sistema não modelado através de info cases. O capítulo 5 conclui o trabalho resumindo os resultados obtidos e suas principais limitações, e aponta sugestões para trabalhos futuros.

Capítulo 2

Testes de software

Este capítulo tem por objetivo apresentar algumas técnicas de construção de casos de teste, estabelecendo assim o contexto do presente trabalho. A seção 2.1 faz uma breve introdução do assunto. Em seguida são apresentadas algumas das principais técnicas de construção de casos de teste, segundo a categorização usual: testes caixa-branca (seção 2.2) e testes caixa-preta (seção 2.3).

2.1. Introdução

A atividade de testes é a última revisão de especificação, projeto e codificação, sendo, portanto, uma atividade crítica da garantia de qualidade de software. Segundo PRESSMAN (2006), em vista dos custos envolvidos associados a erros de software, não é incomum que o emprego de esforços relacionados aos testes possa chegar a 40% do total de um projeto.

Para sistemas de alta confiabilidade como sistemas de controle de vôo ou refrigeração de reatores em uma usina nuclear, por exemplo, as atividades de testes podem custar de três a cinco vezes mais do que todas as outras etapas da engenharia de software somadas.

Com relação à atividade de garantia de qualidade é importante deixar claro alguns pontos básicos.

Teste é o processo de executar um programa com a intenção de encontrar erros, sejam por resultados errados ou por inadequação aos requisitos. Com base neste conceito, define-se que um teste é bem sucedido quando algum erro ainda não conhecido é encontrado (MYERS, 1978).

Levando em consideração que o processo de desenvolvimento envolve várias pessoas e cada uma destas possui um modo próprio de se expressar, não é difícil imaginar que a introdução de erros pode ocorrer em quaisquer das etapas de um projeto. Portanto, considerar que a atividade de testes deve gerar culpa seria um tanto quanto injusto, segundo BEIZER (1990). O que pode ser feito é uma coleta de dados para avaliação do processo de desenvolvimento em busca de melhorias, seja para modificação da metodologia utilizada, seja para aperfeiçoamento dos profissionais envolvidos, ou ainda para o remanejamento/ substituição de pessoas.

Embora o objetivo da atividade de testes seja revelar a presença de erros, ela não pode garantir a ausência destes. Esta é uma limitação inerente ao controle de qualidade e se torna ainda mais evidente quando recursos como tempo e investimento são reduzidos.

Vale ressaltar que como benefício das atividades de testes tem-se um aumento da confiança no software, atribuindo a este um nível de qualidade ao constatar que as funções e desempenho estão, aparentemente, dentro do especificado.

Após a aplicação dos testes, os resultados obtidos são analisados e comparados com o que era esperado. Caso exista alguma discrepância, dá-se início à depuração.

Entende-se por depuração a busca pela causa do erro. Infelizmente a duração desta etapa é muito imprevisível podendo levar de minutos a dias, dificultando o planejamento das atividades de testes.

Na medida em que os resultados são avaliados, é possível atribuir valor à qualidade e confiabilidade do software em desenvolvimento. Caso muitos erros graves e de difícil solução sejam encontrados, temos um indicativo da má qualidade do software, fazendo com que mais testes sejam executados. Por outro lado, se os resultados dos testes indicam a presença de poucos erros ou apenas erros fáceis e de rápida solução, duas situações são plausíveis: ou o software possui uma qualidade e confiabilidade elevadas, ou os testes foram incapazes de encontrar erros graves.

A inexistência de erros quando da aplicação de testes, segundo PRESSMAN (2006), seria um indicativo de que a configuração dos testes não foi capaz de detectar erros, ou seja, eles serão percebidos pelos usuários durante a etapa de implantação do sistema, momento em que o custo para os reparos são de 60 a 100 vezes maiores do que na etapa de desenvolvimento.

Normalmente, os testes de software são organizados como um conjunto de casos de teste. Cada caso de teste define as condições de execução do software e os resultados esperados dessa execução. Existem diversas técnicas para a elaboração de casos de teste. Elas estão subdivididas em duas categorias: testes caixa-branca e testes caixa-preta. As próximas duas seções apresentam algumas das técnicas mais usuais em cada categoria.

2.2. Testes Caixa Branca

Esta técnica utiliza da estrutura de controle do projeto procedimental para derivar os casos de testes. Aplicada diretamente sobre o código do software desenvolvido, sua intenção é (PRESSMAN, 2006):

- Garantir que todos os caminhos independentes dentro de um módulo tenham sido exercitados pelo menos uma vez
- Exercitar todas as decisões lógicas para valores falsos ou verdadeiros
- Executar todos os laços em suas fronteiras e dentro de seus limites operacionais
- Exercitar as estruturas de dados internas para garantir a sua validade

É inerente a esta técnica o acesso ao código fonte por parte do testador, a ponto de poder realizar alterações e assim conseguir verificar as partes integrantes do componente que esta sendo avaliado. Desta forma é possível exercitar todas as estruturas internas da aplicação, tais como: condições, fluxo de dados, caminhos lógicos e ciclos.

Um dos problemas associados com as técnicas de teste tipo caixa-branca é o excessivo número de casos de teste necessários. Uma técnica possível é procurar exercitar todos os caminhos lógicos de execução possíveis. Entretanto, isso pode demandar a elaboração de milhares de casos de teste, o que, em geral, não é viável. PRESSMAN (2006) exemplifica essa situação para um programa com fluxo de controle mostrado na figura 2.1, que contém um pequeno laço de 20 repetições, com condições lógicas aninhadas. O número de caminhos de execução possíveis chega a aproximadamente 10^{14}.

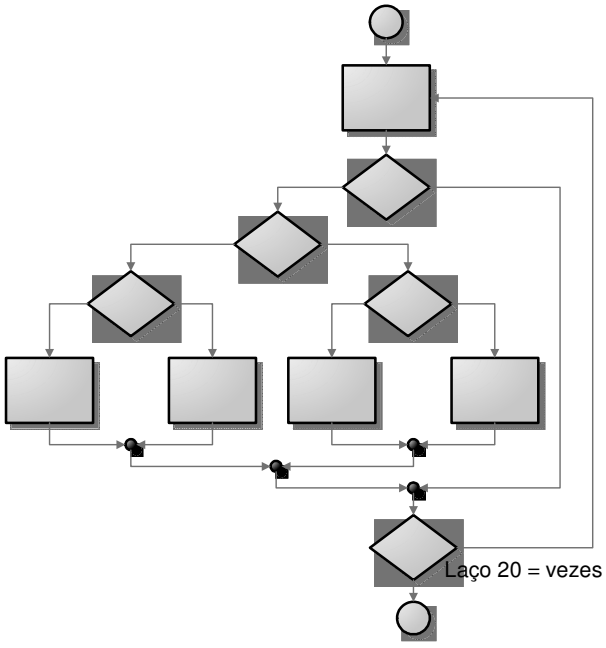


Figura 2.1. Laço com 20 repetições (adaptado de (PRESSMAN, 2006))

Apesar da desvantagem acima mencionada, a técnica de caixa branca é importante por oferecer cobertura para aquelas funções que, a primeira vista, não são tão executadas, ou fogem do caminho lógico principal, os chamados casos especiais. Também, erros

tipográficos (erros de digitação) que podem passar despercebidos para o desenvolvedor acostumado com seu próprio código, costumam ser detectados nesse tipo de teste.

A seguir, serão apresentadas algumas técnicas de teste caixa-branca. O uso de mais de uma técnica costuma aumentar as chances de descoberta de erros.

2.2.1 Teste de Caminho Básico

O teste de caminho básico utiliza uma medida de complexidade lógica do programa como guia para definir um conjunto básico de caminhos de execução (PRESSMAN, 2006). Os casos de teste derivados para exercitar o conjunto básico têm a garantia de executar cada instrução do programa pelo menos uma vez durante a atividade de teste.

O método do caminho básico utiliza uma notação simples denominada *grafo de fluxo* para representar o fluxo de controle do programa (PRESSMAN, 2006). A figura 2.2 exemplifica o grafo de fluxo para algumas estruturas de controle comumente presentes nos programas.

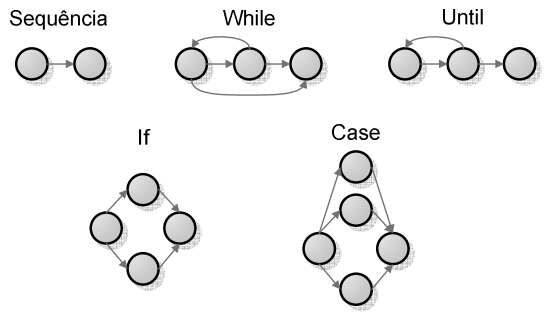


Figura 2.2. Grafos de fluxo de estruturas de controle (extraído de (PRESSMAN, 2006))

Cada círculo da figura acima é um nó que pode conter mais de uma instrução e/ou condição, enquanto os arcos representam as transferências de controle. A figura 2.3 ajuda a esclarecer o significado de um grafo de fluxo. Ela mostra, à esquerda, o fluxo de controle de um programa e, à direita, o correspondente grafo de fluxo. A numeração exibida em ambas as partes da figura evidencia a correspondência entre seus elementos.

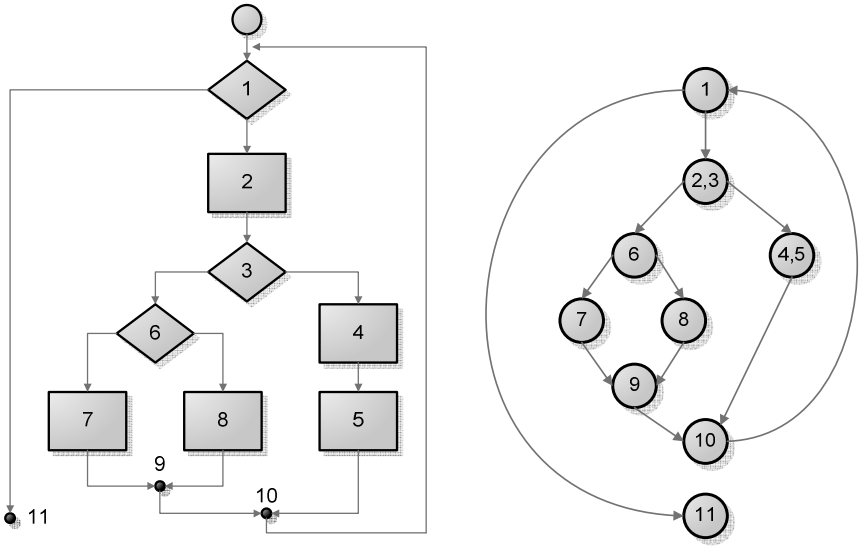


Figura 2.3. Fluxo de controle e sua representação em grafo de fluxo (extraído de (PRESSMAN, 2006))

Segundo PRESSMAN (2006), qualquer representação de projeto procedimental pode ser traduzida em um grafo de fluxo.

No teste de caminho básico é importante definir os caminhos (de execução) independentes de um programa, pois o número desses caminhos será exatamente o número mínimo de casos de testes a aplicar para garantir que todos os caminhos sejam percorridos. Um caminho independente é qualquer caminho através do programa que introduza pelo menos um novo conjunto de instruções de processamento ou uma nova condição. Quando estabelecido em termos de um grafo de fluxo, um caminho independente deve incluir pelo menos um ramo que não tenha sido atravessado antes que o caminho seja definido (PRESSMAN, 2006).

Uma métrica de software interessante que pode ser utilizada neste contexto é a complexidade ciclomática (PRESSMAN, 2006). Esta métrica visa mensurar a complexidade lógica de um programa e, quando aplicada ao contexto de teste de caminho básico, traz como resultado a quantidade de caminhos independentes dentro do conjunto básico do programa a ser avaliado, ou seja, a quantidade de casos de testes a aplicar para que todos os caminhos sejam percorridos pelo menos uma vez.

Segundo PRESSMAN (2006), uma das formas de se calcular a complexidade ciclomática de um procedimento é dada pela fórmula:

1. V(G) = E - N + 2, onde:

   E - número de ramos do grafo de fluxo

N - Número de nós do grafo

Por exemplo, aplicando a fórmula acima para o grafo de fluxo da figura 2.3 (acima) obtém-se uma complexidade ciclomática igual a 4 (E – N + 2 = 11 – 9 + 2 = 4). Portanto, o teste de caminho básico, aplicado a esse procedimento, deve exercitar pelo menos 4 caminhos independentes, cobertos em, pelo menos, 4 casos de teste. Embora o teste de caminho básico seja simples e efetivo, ele não é suficiente por si só. Nas próximas três seções serão apresentados outros testes caixa-branca cuja utilização conjunta tende a ampliar a cobertura dos testes e melhorar a qualidade final do programa.

2.2.2. Teste de Condição

Este tipo de teste visa colocar à prova as condições lógicas existentes em um módulo de um programa.

Uma *condição (lógica) simples* é uma variável booleana (V ou F) ou uma expressão relacional, possivelmente precedida pelo operador NOT. Uma *expressão relacional* assume a forma: Expressão1 <OperadorRelacional> Expressão2, onde E_1 e E_2 são expressões aritméticas e OperadorRelacional é um operador relacional (>>, ≥, <<, ≤, = ou ≠). Uma *condição composta* é constituída de duas ou mais condições simples, operadores booleanos (AND, OR ou NOT) e parênteses. Uma condição sem expressões relacionais é chamada *expressão booleana*.

Quando uma condição esta incorreta é porque um dos componentes da condição esta errado, incorrendo em um dos itens abaixo:

- Erro de operador booleano (uso inadequado, excesso, ausência)
- Erro de variável booleana
- Erro de parênteses, precedência entre componentes
- Erro de operador relacional
- Erro de expressão aritmética

Existem diversas estratégias de testes de condição. O teste de (todos os) ramos é, provavelmente, a mais simples. Visa executar, pelo menos uma vez, os ramos verdadeiro e falso de uma condição C qualquer.

O teste de domínio (WHITE & COHEN, 1980) requer que se definam casos de testes para verificação das condições de modo que, em uma expressão lógica simples do tipo Expressão1 <OperadorRelacional> Expressão2, a Expressão1 obtenha valores maiores,

iguais e inferiores a Expressão2, totalizando três casos de testes. Este tipo de teste revelará a existência de erros de operador relacional. Para verificar se existem erros nas expressões, é necessário tornar seus valores bem próximos (PRESSMAN, 2006).

Para o teste de condições, a estratégia chamada de *teste de ramos e de operadores relacionais* auxilia na descoberta de erros. Em sua definição, uma condição lógica é desmembrada em condições simples, e conjuntos de valores para cada operando são determinados de maneira a verificar as possibilidades existentes. Caso exista um erro de operador, um par de valores fatalmente conduzira à falha da condição (PRESSMAN, 2006).

Para deixar mais claro, tomemos como exemplo uma condição C_1 do tipo B_1 AND B_2. Assumindo valores booleanos (V ou F) para cada termo, definimos o conjunto de pares de valores {(V, V) (V, F) (F, V)} que devem ser aplicados ao par (B_1, B_2), para que a condição possa ser exercitada

Caso a condição seja do tipo B_1 AND ($E_1 = E_2$), onde B_1 é uma expressão booleana e E_i (i = 1, 2) são expressões aritméticas, então o valor verdadeiro ou falso para o segundo termo ($E_1 = E_2$) da condição deverá ser transformado para operadores relacionais, ou seja, será verdadeiro o segundo termo se $E_1 = E_2$ e será falso se $E_1 > E_2$ ou $E_1 < E_2$. Substituindo no conjunto de pares mostrado anteriormente para a verificação da condição teremos: {(V,=) (F,=) (V,<) (V, >)}, garantindo a detecção de erros de operadores relacionais e booleanos para a condição.

Segundo PRESSMAN (2006), as diversas variantes de teste de condição geralmente apresentam duas vantagens. Primeiro, a medição da cobertura do teste de uma condição é simples. Segundo, a cobertura de teste das condições de um programa oferece orientação para a geração de testes adicionais para o programa.

2.2.3. Teste de fluxo de dados

Essa categoria de testes caixa-branca define a escolha do caminho de teste com base nas definições e nos usos de variáveis. Em (MALDONADO, 1991) tem-se a prova de que esse tipo de teste pode atingir complexidade exponencial com o número de comandos de decisão de um programa. No entanto, estudos empíricos (WEYUKER, 1990) revelam que, para programas reais, o número de casos de teste necessários pode ser bem menor (ordem linear).

Para ilustrar um teste de fluxo de dados, suponhamos que a cada instrução de um programa seja atribuído um número de instrução único, e que cada função não modifique

seus parâmetros ou variáveis globais. Sendo assim, para uma instrução S qualquer se define:

DEF(S) = {X | a instrução S contém uma definição de X}

USE(S) = {X | a instrução S contém um uso de X}

Uma *associação definição-uso* (ou *associação DU*) de X envolve duas instruções S e S' e pode ser representada pela tripla (X, S, S'), significando que X pertence à DEF(S) e a USE(S'), e a definição de X em S, está viva em S', ou seja, a instrução S possui caminho para a instrução S', sem que exista nesse caminho uma nova declaração de X. Uma estratégia de teste de fluxo de dados, baseada em associações definição-uso, denominada *todas as definições*, exige que pelo menos uma associação definição-uso seja exercitada para cada definição de variável. Uma desvantagem deste tipo de teste é o fato de ele não englobar o critério todos os ramos e, portanto, ele não provê cobertura para caminhos não-executáveis[1].

Outras variantes de teste de fluxo de dados existem. Por exemplo, têm-se um teste denominado *todos os usos*, onde para cada definição de variável, todas as associações definição-uso são exercitadas. No teste *todos os du-caminhos*, para cada definição de variável e para cada associação definição-uso, todos os caminhos livres de definição e livres de laço (*du-caminhos*), que cobrem essa associação, são exercitados (MALDONADO, 1991).

2.2.4. Teste de laços

Laços estão presentes na grande maioria dos algoritmos implementados em software, o que torna importante o teste dos mesmos. Esse tipo de teste caixa-branca se concentra na validade da construção dos laços, e casos de testes são propostos de acordo com o tipo de laço: simples, concatenados, aninhados, e estruturados. A figura 2.4 exibe esses tipos de laços.

[1] Um caminho é executável se existir um conjunto de valores para as variáveis de entrada do programa que causa a execução deste caminho.

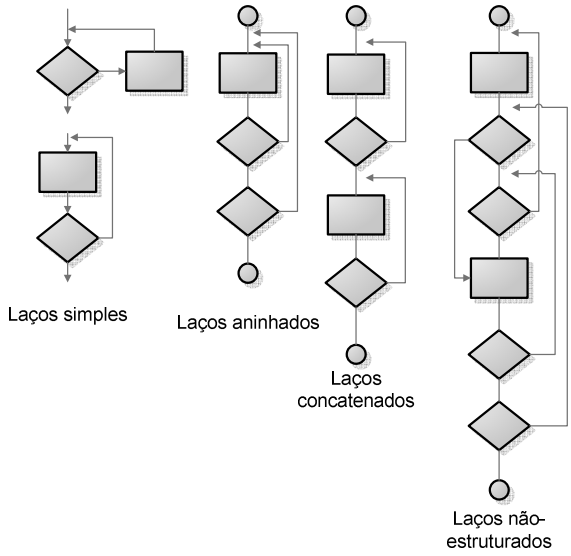


Figura 2.4. Tipos de laços (extraído de (PRESSMAN, 2006))

Para laços simples, os seguintes casos de testes são propostos (PRESSMAN, 2006):

- Pule o laço inteiramente
- Somente uma passagem através do laço
- Duas passagens através do laço
- m Passagens pelo laço, sendo m <n. (onde n é o número máximo de passagens pelo laço)
- n-1, n e n+ 1 passagens pelo laço.

No caso de laços aninhados, o número de casos de teste cresce geometricamente com o número de aninhamentos. Para evitar um número muito grande de casos de teste BEIZER (1990) propõe os seguintes casos de testes para laços aninhados:

- Iniciar no laço mais interno, fixando os laços a valores mínimos.
- Aplicar testes de laços simples no laço mais interno mantendo os contadores dos laços externos a valores mínimos; depois adicionar testes para valores fora da faixa ou excluídos.
- Proceder com a aplicação dos testes de dentro para fora, mantendo os laços mais externos com valores mínimos, e os laços internos com valores “típicos”, enquanto se faz a verificação de um laço intermediário.
- Repetir o procedimento até que todos os laços sejam verificados.

Para laços concatenados, os casos de testes dependem de como o laço está estruturado:

- Se os laços foram independentes, os mesmos casos de teses para laços simples podem ser utilizados.
- Se os laços não forem independentes (o contador do segundo laço depende do contador do primeiro), então devem ser utilizados os mesmos casos de testes propostos para laços aninhados.

No caso de laços não-estruturados, é recomendável substituí-los por construções da programação estruturada.

2.3. Testes Caixa Preta

Também conhecido como teste funcional, esta técnica não leva em consideração o código fonte, ou seja, o componente ou software é exercitado apenas através dos dados de entrada e de saída, ignorando-se a estrutura interna (lógica de execução). Como os requisitos funcionais são o foco, casos de testes são derivados para que todos sejam exercitados.

Normalmente, neste tipo de teste, os dados de entrada são escolhidos e os de saída são previamente calculados para posterior comparação com os resultados esperados.

Os testes caixa-preta visam complementar a prática dos testes de caixa-branca, pois tendem a descobrir uma classe de erros distinta daquela dos testes caixa-branca (PRESSMAN, 2006).

As categorias em que o teste caixa-preta procura encontrar erros são:

- Funções incorretas ou ausentes
- Erros de interface
- Erro nas estruturas de dados ou no acesso a banco de dados de externos
- Erros de desempenho
- Erros de inicialização ou término

Pelas categorias listadas acima, é possível perceber que a aplicação deste tipo de teste não pode ser feita logo no início da atividade de teste, como normalmente acontece com os testes caixa-branca; a tendência é que eles sejam aplicados nas etapas finais dos testes.

Para o projeto deste tipo de teste, as perguntas abaixo devem ser respondidas:

- Como a validade funcional é testada?
- Quais classes de entrada constituirão bons casos de testes?
- O sistema é particularmente sensível a certos dados de entrada?
- Como os valores da fronteira de uma classe de dados são isolados?

- Quais índices de dados ou valores e volumes o sistema pode tolerar?
- Que efeito terá combinações específicas de dados sobre a operação do sistema?

Para responder a estas perguntas, diversas técnicas podem ser utilizadas, como as que serão explicadas na seqüência.

2.3.1. Particionamento de Equivalência

A técnica *particionamento de equivalência* divide o domínio de dados de entrada de um programa em classes de dados, de acordo com as classes de equivalência para uma condição de entrada. As condições podem ser um valor numérico, um intervalo, um valor booleano ou um conjunto de valores relacionados, de modo que a classe de equivalência seja um conjunto de estados válidos ou inválidos para as condições de entrada.

No caso de uma condição booleana, é possível definir uma classe válida e uma inválida para a referida condição. Se a condição de entrada especificar um valor ou um intervalo de valores, as classes de equivalência são três: uma classe válida (valores dentro do intervalo ou igual ao valor esperado) e duas classes inválidas (valores superiores ou inferiores ao esperado).

O exemplo dado por PRESSMAN (2006) utiliza um software para conexão discada em que é necessário informar um *login* e uma senha para proceder com a discagem e conexão remota ao provedor.

Os requisitos são:

- O campo de *login* é no formato de email; pode conter letras, dígitos e caracteres especiais e, caso o usuário não forneça o domínio, a aplicação completa automaticamente.
- O campo de senha é alfanumérico de no mínimo seis dígitos e máximo de doze.

A especificação de classes de equivalência fica da seguinte forma:

Login:

- Condição de entrada, intervalo - string com requisitos (a string não pode iniciar por dígito)
- Condição de entrada, booleano - o domínio pode estar presente na informação do usuário ou não.

Senha:

- Condição de entrada, booleano - a senha pode estar presente ou não.

- Condição de entrada, intervalo - string com restrições (deve conter de seis a doze caracteres).

Apesar do campo *login* e senha não serem do tipo lógico, a técnica de particionamento de equivalência utiliza a nomenclatura 'Condição de entrada, booleano', para se referir ao caso de teste em que é feita a verificação da entrada ou não de dados nos referidos campos.

Esta técnica visa definir casos de teste em que cada um descobre uma classe de erros, reduzindo a quantidade total de casos de testes a aplicar sobre o sistema.

2.3.2. Análise do Valor Limite

Complementar à técnica do particionamento de equivalência, esta técnica utiliza as classes de equivalência para definir valores próximos de seus limites, visto que comumente os erros são mais presentes nas extremidades (limites) do que nos valores intermediários.

As diretrizes para se definir os casos de testes pela técnica de análise do valor limite, com base nas condições de entrada, são:

- Quando a condição especifica um intervalo delimitado, os casos de testes devem ser projetados para valores logo acima e logo abaixo destes limites.
- Quando se tem uma série de valores, o máximo e o mínimo devem ser verificados, bem como valores logo abaixo e logo acima destes.
- Devem ser aplicadas as diretrizes citadas acima para as condições de saída do programa, ou seja, devem ser projetados casos de testes que forcem que a saída seja o valor mínimo/máximo permitido.
- As estruturas de dados que possuem limites claros e definidos devem ser verificadas em suas fronteiras.

2.3.3. Tabelas de decisão

Estas tabelas são criadas para facilitar a identificação de qual decisão deve ser tomada, quando uma combinação complexa de condições deve ser avaliada.

Uma tabela de decisões é dividida em quatro regiões:

- 1ª: Lista de condições
- 2ª: Lista de ações
- 3ª: Colunas onde serão marcadas as condições a serem satisfeitas
- 4°: Colunas onde será marcada a decisão de acordo com as condições

A figura 2.5 (a seguir) exemplifica as regiões da Tabela de Decisões

1ª	3ª
2ª	4ª

Figura 2.5. Esquema de uma Tabela de Decisão (adaptado de (PRESSMAN, 2006))

PRESSMAN (2006) apresenta o seguinte exemplo para ilustrar a aplicação de uma tabela de decisão:

Se a conta de luz de um cliente for cobrada usando-se um método de taxas fixas, uma cobrança mensal mínima é avaliada para quem consome menos de 100kWh. Caso contrário, a cobrança computadorizada aplicará uma estrutura de taxas contida num Plano A. Porém, se a conta for cobrada usando-se um método de taxas variáveis, uma estrutura de taxas do Plano A será aplicada a quem consome menos de 100kWh, sendo o consumo adicional cobrado de acordo com um plano B.

Tabela 2.1. Exemplo de Tabela de Decisão

	1	2	3	4	5
Conta com taxa fixa	V	V	F	F	F
Conta com taxa variável	F	F	V	V	F
Consumo < 100kWh	V	F	V	F	F
Consumo 100kWh	F	V		V	F
Cobrança mensal mínima	X				
Cobrança do Plano A		X	X		
Cobrança do Plano B				X	
Outro tratamento					X

Definida a tabela de decisão, cada coluna será uma regra, e para cada regra se define um caso de testes.

2.3.4. Testes de comparação

Este tipo de testes é aplicado em sistemas considerados críticos, ou seja, sistemas onde se exige alta confiabilidade para evitar perdas materiais significativas ou mesmo de vidas humanas.

Os testes de comparação costumam ser executados de forma mais automatizada que os demais já vistos, e consiste em aplicar testes em que os resultados de softwares diferentes são metodicamente comparados. Diferentes equipes de desenvolvimento constroem, de forma independente uma da outra, um software que atenda a uma mesma especificação. No final, apenas um desses softwares será utilizado.

Durante a realização de testes (elaborados segundo as demais técnicas caixa-branca e caixa-preta), sempre que uma diferença de resultado para dados de entradas iguais for detectada, os testes são interrompidos e todos os programas são investigados para se descobrir a razão da discrepância de valores.

Este tipo de teste possui suas falhas: se a especificação estiver incorreta, então todos os sistemas estarão errados e, ainda, se forem obtidos resultados idênticos, mas incorretos, o teste deixará de detectar o erro.

Capítulo 3

Modelagem Informacional de Requisitos

O objetivo deste capítulo é apresentar uma técnica semi-formal para a especificação de requisitos de sistemas de informação, denominada *Modelagem Informacional de Requisitos* - MIR (FORTUNA, BORGES, 2005), como base para cumprir o principal objetivo deste trabalho, que é a proposição de técnicas de elaboração de casos de testes adaptadas à MIR.

A MIR se baseia na modelagem com casos de uso (JACOBSON *et al.,* 1992); de fato, os casos de uso da MIR (denominado *infocases*) constituem uma especialização dos casos de uso tradicionais

3.1. Casos de Uso

Casos de uso (ou do inglês *Use Cases*- UC) representam a interação do sistema com seus usuários. O conceito foi elaborado por JACOBSON *et al.*(1992), e posteriormente incorporado à UML (*Unified Modeling Language*) (OMG, 2009) em 1996. Desde então passou a ser amplamente utilizado na modelagem de requisitos de software.

Casos de uso são utilizados para explicitar as funcionalidades de um sistema. De forma geral, é o primeiro documento a ser produzido, preferencialmente com o auxílio dos *stakeholders*, para especificar os requisitos funcionais do sistema. É comum criar diversos casos de uso para se abranger todas as necessidades dos *stakeholders*. Entende-se por *stakeholder* uma pessoa ou organização que tenha interesse no sistema a ser construído e, portanto, deve ser consultado durante a fase de levantamento de requisitos.

Na elaboração dos casos de uso utiliza-se, na maioria das vezes, a linguagem natural para descrever a interação entre o sistema e seus atores. Os atores representam entidades externas ao sistema (ambiente) e podem ser usuários ou outros sistemas (componentes de hardware ou ainda outros programas) que tenham interação com o sistema sendo modelado. Em outras palavras, um ator especifica um papel desempenhado por um usuário ou sistema e, desta forma, vários atores podem representar uma mesma entidade em funcionalidades distintas, dentro de uma especificação de casos de uso.

De acordo com as necessidades e objetivos dos atores, são construídos os *diagramas de casos de uso* (DUC). A figura 3.1 é um exemplo de diagrama de casos de uso.

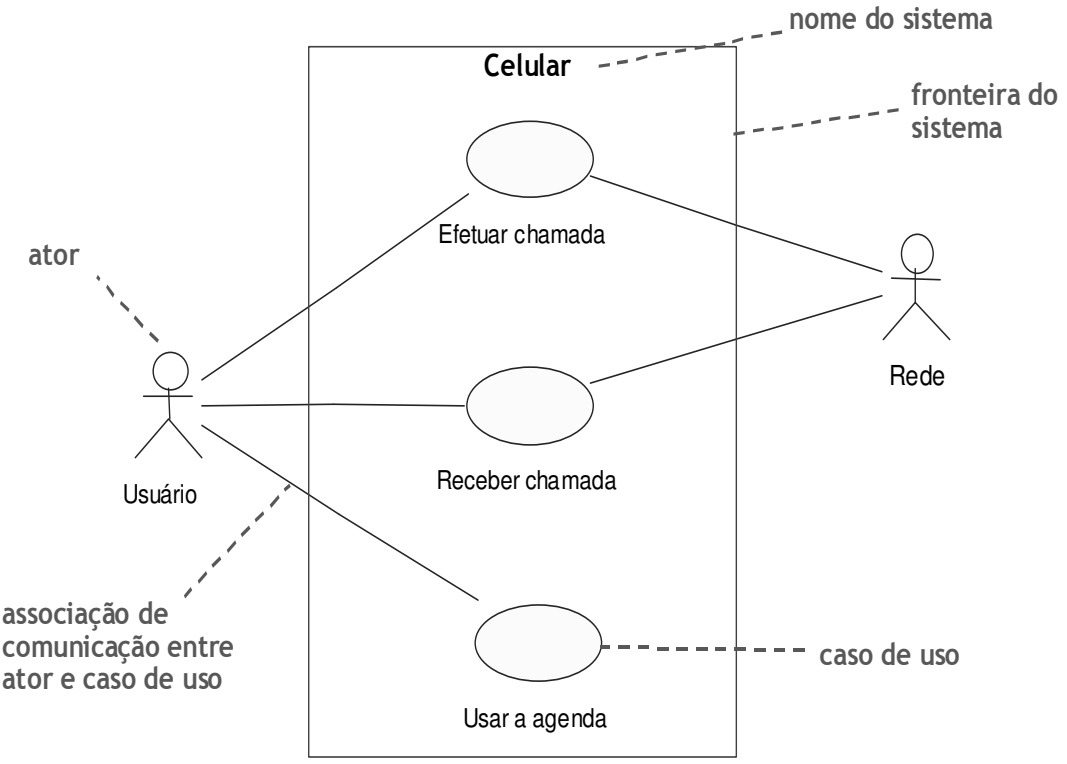


Figura 3.1. Diagrama de Casos de Uso

Para definir os atores, basta identificar as pessoas e sistemas que irão interagir com o sistema a ser desenvolvido, e o papel que desempenharão.

3.2. Modelagem Informacional de Requisitos

Na Modelagem Informacional de Requisitos – MIR (FORTUNA, BORGES, 2005) os casos de uso devem atender a alguns requisitos para serem validados:

- Sua ativação e completa realização devem depender apenas de uma única intervenção de iniciativa de um ator, o chamado *evento autônomo*;
- Deve permite ao ator alcançar um objetivo, obtendo um sentimento de missão cumprida;
- Caso a realização do UC cause alteração do estado do sistema, o novo estado resultante deve ser um *estado estável* (ou *estado firme*).

Entende-se por estado estável, um estado no qual não existe a necessidade de retorno a um estado anterior (*rollback*), mesmo que nenhum outro caso de uso seja posteriormente ativado (FORTUNA *et al*, 2008).

O conceito de *estado estável* pode ser melhor esclarecido através de exemplos. Em um sistema de gerência de restaurante, após o sistema registrar o pedido de refeição de um cliente, ele se encontra em um estado estável, pois se nada mais ocorrer no sistema (por

exemplo, o pagamento da refeição não ocorrer), o estado atual deve ser mantido[2]. Já em um sistema de ponto-de-venda (por exemplo, o caixa de um supermercado), após o sistema registrar os itens que serão comprados, ele não se encontra em um estado estável. Para isso, é necessário que ocorra o pagamento da compra. Em outras palavras, caso esse pagamento não ocorra, será necessário retornar o sistema para o estado anterior àquele do registro dos itens a serem comprados.

Eventos autônomos são os eventos gerados pelo ambiente externo ao sistema (atores), para os quais nenhuma suposição pode ser embutida no sistema sobre o evento externo que o antecede. Todo evento autônomo dispara um processo responsável pelo alcance de um objetivo de algum ator.

No caso do sistema de gerência de restaurante, os eventos externos "Registrar pedido de cliente" e "Pagar a conta" são autônomos, pois nenhuma suposição pode ser embutida no sistema sobre que evento antecede cada um. Pagar a conta pode suceder Registrar pedido, mas não necessariamente. Entre eles, outros eventos podem ocorrer, como por exemplo, "Emitir cardápio". Já no caso do sistema ponto-de-venda, o evento externo "Pagar a compra" não é autônomo, pois o sistema é codificado com a suposição que cada registro de compra deve ser sucedido pelo pagamento da compra. Diz-se que o evento Pagar compra é *subassumido* pelo evento autônomo Registrar compra.

3.2.1 Info Case

Um info case é um caso de uso que satisfaz a dois critérios:

- O critério do evento autônomo: a ativação e completa execução do info case deve depender da ocorrência de apenas um evento autônomo; e
- O critério do evento autônomo: após a realização do info case, o sistema deve ficar em um estado estável.

A figura 3.2 lista os info cases de um possível sistema de gerência de restaurante.

Objetivos Informacionais do **ator *Cliente***	Objetivos Informacionais do **ator *Gerente***
1 - Abrir pedido	6 - Atualizar o cardápio
2 - Cancelar pedido	7 - Solicitar consumo diário
3 - Pedir a conta	8 - Solicitar receita
4 - Pagar a conta	9 - Solicitar cópia do cardápio
5 - Pendurar a conta	10 - Cadastrar cliente habitual

Figura 3.2. Info cases – Sistema Restaurante (FORTUNA, BORGES, 2005)

[2] Supondo ser necessário ter um controle dos itens consumidos no pedido, para a reposição do estoque na cozinha do restaurante.

Assim como os casos de uso, info cases devem possuir uma descrição. Entretanto, diferentemente dos casos de uso que empregam, normalmente, apenas linguagem natural, a descrição de um info case utiliza uma notação baseada em BNF (KNUTH, 1964) para especificar os fluxos de informação trocados entre o sistema e seus atores. A essa descrição dá-se o nome de interface informacional dos info cases.

3.2.2 Interface Informacional

A interface Informacional de um info case é a especificação semi-formal dos fluxos de informação (entrada e saída) trocados entre o sistema e seus atores, durante a realização do info case. Tal especificação é elaborada utilizando uma notação similar a BNF, a qual é composta dos seguintes símbolos:

- ➔: fluxo de entrada
- ➔: fluxo de saída
- 🗐: itens compostos ou pacotes
- +: composição
- $_{n}\{x\}_{m}$: de n a m ocorrências de x. Pode ser omitido o "m" indicando apenas a ocorrencia mínima "n".
- ( ): usado para agrupar itens
- |: ou lógico
- [ ]: representa itens condicionais

A figura 3.3 é um exemplo de aplicação da interface informacional na modelagem de um sistema para restaurantes:

ATOR: Cliente **UC 1: Abrir pedido**
➔ pedido = dt_pedido + id_mesa + itens_ped
🗐 itens_ped = $_{1}$ {id_item + quant_item}
➔ id_pedido
ATOR: Cliente **UC 2: Cancelar pedido**
➔ cancela_ped = id_pedido
ATOR: Cliente **UC 3: Pedir a nota**
➔ solicitacao_nota = id_pedido + [id_cliente]
➔ nota = id_pedido + nr_mesa + dt_pedido + itens_nota + vl_nota + [nome_cliente + tel_cliente]

Figura 3.3. Interface Informacional (FORTUNA & BORGES, 2005)

Itens elementares de informação são itens de informação presentes na interface informacional dos info cases que representam informação indivisível do ponto de vista do sistema. Por exemplo, no info case da figura 3.3., dt_pedido (data do pedido) e nr_mesa

(número da mesa) são exemplos de itens elementares de informação. A descrição dos info cases inclui também um dicionário de itens elementares de informação, na forma descrita na próxima seção.

3.2.3 Dicionário de Itens Elementares

Esse dicionário traz informações mais detalhadas sobre cada item elementar de informação, presente na interface informacional dos info cases. A figura 3.4 mostra a parte do dicionário correspondente ao info case "Pedir a nota".

Ator: Cliente	**UC 3: Pedir a nota**		
Nome	**Descrição**	**Tipo**	**Domínio**
id_pedido	Identificador do pedido da nota a emitir.	Nr. Natural	
quant_item	Quantidade consumida de um item.	Nr. Natural	
vl_item	Valor do consumo de um item. Preço unitário × quantidade consumida do item.	Moeda	
cat_item	Categoria do item de consumo	Texto	{prato, bebida}

Figura 3.4. Dicionário de itens (FORTUNA, BORGES, 2005)

3.2.4 Glossário de Termos

Alguns termos do domínio do sistema são utilizados em vários info cases. Eles representam conceitos utilizados no dia-a-dia de trabalho dos *stakeholders*. Para evitar que tais termos sejam definidos repetidamente, o modelo de info cases prevê um glossário para registrá-los. A figura 3.5 ilustra parte do glossário de termos do sistema de gerência de restaurante.

Termo	**Descrição**	**Informação adicional**
Cartão Bancário	Dispositivo físico de identificação, com informações gravadas magneticamente	Inclui o numero do cliente
Cliente	Uma pessoa que detêm contas em uma instituição financeira, participante da rede interbancos de terminais de auto-atendimento, e que possua um cartão bancário	

Figura 3.5. Glossário de Termos (FORTUNA & BORGES, 2005)

3.2.5 Objetivos Organizacionais

O modelo de info case é complementado com a especificação dos *objetivos organizacionais* dos atores. Um objetivo organizacional é um conjunto de "seqüências admissíveis" de info cases, em cada uma que representa uma "linha de trabalho" ou um "objetivo de negócio" relevante no domínio da aplicação, para um ou mais atores. Por exemplo, no caso da aplicação de gerência de restaurante, os conjuntos Tomar refeição = {(1-Abrir pedido, 3-Pedir a conta, 5-Pendurar a conta), (1-Abrir pedido, 3-Pedir a conta, 5-Pendurar a conta, 4-Pagar a conta)} e Cancelar refeição = {(1-Abrir pedido, 2-Cancelar pedido)}, são objetivos organizacionais do ator Cliente.

A obtenção das seqüencias admissíveis requer conhecimentos dos info cases e das relações de dependência temporal e de incompatibilidade entre eles. Por exemplo, o cancelamento de um pedido só pode ser realizado se, em algum momento anterior, um pedido tiver sido registrado no sistema (relação de dependência temporal **dt**). Por outro lado, o info case de Pendurar a conta não pode ser ativado caso a conta tenha sido paga (relação de incompatibilidade **ic**). A figura 3.6 exibe um grafo que apresenta as duas relações (**dt** e **ic**) entre os info cases do sistema de gerência de restaurante.

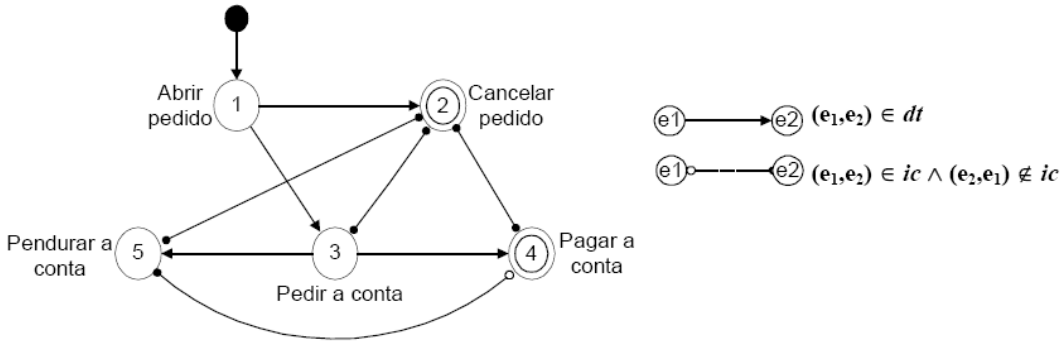


Figura 3.6. Grafo de admissibilidade para o sistema Restaurante (FORTUNA, BORGES, 2005)

Capítulo 4

Testes no contexto da MIR

Este capítulo tem por objetivo apresentar uma proposta para a elaboração de casos de teste a partir dos info cases de um sistema, obtidos através da modelagem informacional de requisitos (MIR). Para isso, explorou-se a aplicabilidade das técnicas de teste apresentadas no capítulo 2, para o contexto da MIR.

4.1. Fontes de Informação

Durante a especificação de requisitos de um sistema, ainda não se tem o código-fonte do mesmo. Portanto, em princípio, a estratégia de teste aplicável nesse contexto é a estratégia caixa-preta. Essa constatação foi o ponto de partida para a proposta de elaboração de casos de teste a partir dos info cases. Outra decisão natural foi organizar os casos de teste por info case, visto que a especificação informacional de requisitos é modularizada com base nos info cases. Além disso, conforme pode ser inferido da discussão no capítulo anterior, sobre as características dos info cases que os diferenciam de um caso de uso tradicional, um info case pode ser visto como um processo do sistema bastante coeso e delimitado. Em geral, cada info case dá lugar, em fases posteriores do desenvolvimento, a uma rotina do sistema.

O primeiro passo no sentido de se chegar aos casos de teste a partir dos info cases foi verificar que informações estão disponíveis nos info cases, e como elas poderiam auxiliar na escolha e elaboração dos casos de teste. A seguir são apresentadas as fontes de informações úteis para esse objetivo, presentes na especificação dos info cases do sistema.

Composição dos fluxos

A composição dos fluxos de cada info case, e dos pacotes neles contidos, serve para mostrar que informações de entrada o sistema espera receber durante a realização do info case, e que informações de saída o sistema irá produzir como resultado. Essa informação é primordial para a elaboração dos casos de teste: as informações presentes no fluxo de entrada constituem as entradas a serem fornecidas pelo testador na ativação do caso de teste, enquanto que as informações presentes no fluxo de saída constituem as saídas que devem ser esperadas como resultado do caso de teste.

Dicionário de itens

Essa parte da especificação dos info case foi primeiramente apresentada no capítulo anterior. Ela contém detalhes sobre cada item elementar de informação presente nos fluxos de entrada e de saída de informações, que constituem a interação entre o sistema e seus atores, durante a realização de um info case. Para cada item elementar de informação, o dicionário contém, entre outras coisas, a especificação do tipo de dado do domínio de valores do item, o domínio propriamente dito, e uma descrição do item. Essas informações fornecem um importante subsídio para a elaboração de casos de teste por classes de equivalência e por análise de valor limite

Objetivos organizacionais

O conceito de objetivo organizacional foi apresentado no capítulo anterior e também tem o seu papel na definição dos casos de teste para um sistema especificado através de info cases. No entanto, diferentemente dos casos de teste elaborados a partir da composição dos fluxos de informação, os casos de teste elaborados a partir dos objetivos organizacionais englobam mais de um info case, situando-se assim na categoria de teste de integração. As relações de dependência temporal e de incompatibilidade entre info cases, representadas no grafo de admissibilidade construído a partir dos objetivos organizacionais de um sistema, permite a definição de caminhos de execução (seqüências de info cases) viáveis e não-viáveis, orientando assim o testador nos testes de integração do sistema. Essa modalidade de teste lembra a técnica caixa-branca de caminhos básicos (seção 2.2), embora ocorra em um nível de granularidade menor (o nível dos info cases).

4.2. Casos de testes

Conforme indicado na seção anterior, a composição dos fluxos, o dicionário de itens elementares e os objetivos organizacionais, que juntos compõem a especificação dos info cases do sistema, possuem informações úteis à elaboração dos casos de testes e devem ser analisados em conjunto para isso.

No contexto deste trabalho foram desenvolvidos casos de teste para dois sistemas: um de gerência de restaurante e outro de fábrica de software. O sistema Restaurante teve seus casos de testes derivados dos info cases, enquanto que o sistema Fábrica de Software teve casos de testes derivados tanto de info cases quanto de uma especificação constituída de uma lista de requisitos e de um diagrama de estados. A derivação de casos de teste para o sistema Fábrica de Software, a partir de duas especificações de requisitos distintas, teve

por objetivo comparar o conjunto de casos de teste obtidos em cada situação. Tanto a especificação de requisitos dos sistemas quanto os casos de teste delas derivados estão apresentados nos anexos (sistema Restaurante - Anexos A e B; sistema Fábrica de Software – Anexos C, D e E).

A seguir é exemplificado como pode ser feita a construção dos casos de testes, considerando o tipo de teste a ser aplicado e de onde foram retiradas as informações necessárias à construção dos casos de teste. O sistema utilizado no exemplo é o de gerência de restaurante (já apresentado no capítulo anterior). É considerado apenas um info case, pois isso é suficiente para demonstrar a proposta deste trabalho. A especificação completa do sistema, bem como dos casos de teste construídos para ele estão apresentados nos Anexos A e B, respectivamente.

ATOR: Garçom	**IC 1: Abrir pedido**

➔ **pedido = dt_pedido + id_mesa + itens_ped**

Descrição: Informações de um pedido de refeição. A mesa identificada por **nr_mesa** não pode ter pedido em aberto (ainda não pago ou pendurado).
Propósito: Informar ao sistema o início de uma nova refeição e o que será nela consumido.
Freqüência: 250/dia.

🗐 **itens_ped = $_1${id_item + quant_item}**

Descrição: Informações sobre os itens (pratos e bebidas) a serem consumidos.

🡐 **id_pedido**

Casos de testes para dt_pedido

Seguindo a ordem de ocorrência dos itens, analisaremos primeiramente no fluxo de entrada, o item **dt_pedido.**

Com base no dicionário de itens, o item **dt_pedido** é do tipo *data* e seu valor *default* é a data corrente no sistema. Estas informações são suficientes para se definir, pelo tipo de teste particionamento de equivalência, os grupos de valores válidos e inválidos, para se verificar se este requisito será atendido pelo sistema ou não.

Ao considerar o tipo *data*, os seguintes casos de testes podem ser planejados pela aplicação do Particionamento de equivalência.

Condição de entrada, intervalo:

- Inserir uma data posterior a data corrente: sistema deve bloquear a abertura de pedido

O caso de teste acima é composto pela ação a ser realizada, seguido do resultado esperado. Neste caso de teste está explicito um intervalo inválido para a data; para proceder com a verificação basta utilizar uma data qualquer que seja posterior à data em que o pedido esta sendo aberto.

Condição de entrada, intervalo:

- Inserir data igual ou inferior à data corrente: pedido deve ser aberto com sucesso

Este caso de testes evidencia um intervalo de valores válidos para a data. Pode-se, inclusive, aplicar dois testes: um em que **dt_pedido** seja igual, e outro em seja inferior à data vigente no sistema. Ambos estarão dentro do intervalo válido pela definição do teste de particionamento de equivalência.

O resultado esperado é a abertura do pedido; caso não seja possível realizar o objetivo do IC, então um erro foi detectado.

Ainda com relação ao tipo do item elementar, é utilizada uma técnica complementar ao particionamento de equivalência **-** o teste chamado Análise do valor limite, conforme exemplificado abaixo:

- Inserir datas que estejam fora do intervalo aceitável para dias: ‘00’ e ‘32’: sistema não deve aceitar a data inserida.

  Neste caso de teste foram definidos dois valores que estão além do limite válido para dias, resultando em dois testes a serem aplicados.

  É esperado que o sistema não aceite datas inválidas
- Inserir datas que estejam fora do intervalo aceitável para meses: ‘00’ e ‘13’: sistema não deve aceitar a data inserida.

Pela descrição do item no dicionário de itens, foi mencionado um valor default. Aplicando o particionamento de equivalência teremos:

Condição de entrada, booleano

- Sistema deve preencher automaticamente com a data corrente.

  Este caso de testes classificado como booleano não implica que o campo seja do tipo booleano, mas sim que seu preenchimento atende a condição de ser preenchido ou não.

  Portanto, deverá ser observado se o sistema irá preencher automaticamente o campo e se será feito com a data atual do sistema.

- Apagar a data do campo: sistema deve exigir a presença de uma data para abertura do pedido

  Uma vez que **dt_pedido** pode conter valores diversos identificados pelo particionamento em intervalos, subentende-se que o campo permite edição para o usuário e desta forma seu conteúdo poderá ser apagado. Analisando os demais info cases, é possível notar que **dt_pedido** é necessário para emissão de relatórios, por exemplo, (info case 11). Portanto, não pode ser criado um pedido sem que tal campo seja preenchido.

  De forma resumida, os casos de testes para **dt_pedido** são:

  dt_pedido

  (Particionamento de equivalência)

  Condição de entrada, booleano
- Sistema deve preencher automaticamente com a data corrente.
- Apagar a data do campo: sistema deve exigir a presença de uma data para abertura do pedido

  Condição de entrada, intervalo
- Inserir uma data posterior a data corrente: sistema deve bloquear a abertura de pedido
- Inserir data igual ou inferior à data corrente: pedido deve ser aberto com sucesso

  (Análise do valor limite)
- Inserir datas que estejam fora do intervalo aceitável para dias: ‘00’ e ‘32’: sistema deve bloquear a ação.
- Inserir datas que estejam fora do intervalo aceitável para meses: ‘00’ e ‘13’: sistema deve bloquear a ação.

Casos de testes para id_mesa

No dicionário de itens, o item **id_mesa** é do tipo *natural*, gerado a partir do cadastramento de mesas, conforme consta no info case (IC) 10. Observando o grafo de admissibilidade, nota-se uma relação de dependência entre o IC 1 e o IC 10, ou seja, pelo menos uma mesa deve estar cadastrada para que um pedido seja aberto. Feito o cadastramento, a ordem de execução destes IC não mais importa.

Um primeiro caso de teste seria solicitar um pedido sem que nenhuma mesa tenha sido cadastrada no sistema ainda:

- Solicitar pedido sem que nenhuma mesa tenha sido cadastrada: sistema deve impedir abertura do pedido.

Prosseguindo na elaboração dos casos de testes, como um pedido deve ser associado a uma mesa, pelo particionamento de equivalência dois casos de testes são identificados:

Condição de entrada, booleano: número da mesa deve ser inserido ao abrir pedido.

- Abrir pedido sem o número da mesa: sistema deve validar a presença do número
- Abrir pedido com o número válido de mesa: pedido deve ser aberto com sucesso

Levando em consideração os números das mesas e o domínio definido no dicionário de itens, a condição de entrada para intervalos do particionamento de equivalência possibilita vislumbrar os seguintes testes:

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens

- Solicitar pedido para uma mesa que tenha um pedido em aberto (não pago, ou pendurado): sistema deve bloquear a ação.
- Inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.
- Solicitar pedido inserindo número de mesa que pertença ao intervalo [1, 50]: pedido deve ser realizado com sucesso

Para aprofundar ainda mais os testes dentro dos intervalos já definidos, a técnica de análise do valor limite sugere exercitar tanto os extremos destes conjuntos, quanto uma unidade além deles:

- Solicitar pedido em que o número da mesa seja igual a zero: sistema deve bloquear a ação
- Solicitar pedido com número da mesa maior do que a quantidade de mesas do estabelecimento: sistema deve bloquear a ação
- Solicitar pedidos em que os números das mesas sejam iguais aos extremos da quantidade de mesas do estabelecimento: pedido deve ser aberto com sucesso

Casos de testes para itens_ped

O item **itens_ped** é uma composição a partir dos itens elementares **id_item** e **quant_item**, ou seja, os dois são necessários para o sucesso da execução do info case 1. Além disto, a ocorrência mínima para este pacote é igual a "1"

Pela técnica de particionamento de equivalência, considerar uma condição de entrada lógica para **itens_ped** implica em apenas duas condições: estar presente ou não, equivalente a dizer verdadeiro (V) ou falso (F) para sua ocorrência. Como **itens_ped** é um pacote, os itens que o compõem devem ser verificados quanto a suas ocorrências como se fosse um teste de condição lógica. Ao considerar apenas as ocorrências ou não, teremos os conjuntos para realizar as verificações: **itens_ped**: {(V), (F)}

Ao considerar os itens que o compõem: **itens_ped**: {(V, V), (F, V), (V, F)}

Para os conjuntos definidos acima, os seguintes casos de testes são propostos:

- Realizar pedido informando apenas um item e sua quantidade: caso de sucesso

  Verificação em que as ocorrências dos itens elementares são (V, V)

- Realizar pedido inserindo apenas o item: sistema deve bloquear a ação

  Verificação em que as ocorrências dos itens são (V, F)

- Realizar pedido inserindo apenas a quantidade: sistema deve bloquear a ação

  Verificação em que as ocorrências dos itens são (F, V)

Além dos testes acima, é necessário realizar a verificação de ocorrência mínima, conforme indicado no IC:

- Realizar pedido sem informar nenhum item nem sua quantidade: sistema deve impedir a ação

Como a composição do pacote já foi verificada, não é necessário realizar o teste de condição de entrada lógica para os itens que o compõem.

Casos de testes para id_item

Observando o grafo de admissibilidade, nota-se uma relação de dependência entre o IC 1 e o IC 7, ou seja, pelo menos um item deve estar cadastrado no cardápio para que um pedido seja solicitado. Feito o cadastramento, a ordem de execução destes IC não mais importa.

Um primeiro caso de testes seria solicitar um pedido sem que nenhum item exista no cardápio:

- Solicitar pedido sem que nenhum item tenha sido adicionado ao cardápio: sistema deve impedir abertura do pedido.

Os testes para verificação do comportamento do sistema quando um item não é identificado já foram verificados durante os testes de **itens_ped**. Resta verificar com relação aos grupos de valores aceitáveis, conforme sugerido pelo particionamento de equivalência:

- Solicitar um item cujo **id_item** não esteja no cardápio: sistema deve bloquear a ação
- Solicitar um item cujo **id_item** esteja no cardápio: pedido deve ser aberto com sucesso

Casos de testes para quant_item

Para **quant_item**, a verificação quanto à sua inserção ou não durante realização de um pedido também já foi verificada.

No dicionário de itens consta que quant_item é do tipo *natural*. Sendo assim, os seguintes casos de testes podem ser criados:

Condição de entrada, intervalo: número Natural

- Informar a quantidade com um número fracionário: sistema deve bloquear a ação
- Informar a quantidade com um número negativo: sistema deve bloquear a ação
- Informar um número inteiro positivo: sistema deve aceitar quantidade informada

Verificados os conjuntos de valores aceitáveis e não aceitáveis, aplica-se a análise do valor limite:

- Informar quantidade igual a zero: sistema deve aceitar quantidade informada

  Esse caso de testes levanta um questionamento: o sistema deve aceitar pedidos com quantidade zero? Pela especificação técnica sim. Como o sistema ainda não foi construído, um simples questionamento aos *stakeholders* permitiria verificar essa questão. Em tópico mais a frente serão abordados todos os pontos da especificação passíveis de questionamento.

Como não foi definido o limite superior para o item **quant_item**, entende-se que qualquer quantidade deve ser aceita pelo sistema:

- Informar quantidade igual a 1000 unidades para um item qualquer: sistema deve aceitar a quantidade informada

  Este caso de testes sugere outro questionamento: qualquer quantidade é permitida?

Casos de testes para id_pedido

Não somente as informações de entrada devem ser verificadas, mas também as saídas que o sistema produz. Pela especificação do info case 1, **id_pedido** é um número gerado automaticamente pelo sistema. Subentende-se que este número deve ser único. Os demais aspectos considerados na verificação, como seqüência crescente e sem saltos, foi produzido considerando a lógica das seqüências usadas por bancos de dados.

A verificação ficaria da forma:

- Sistema deve gerar um número correspondente ao pedido aberto, tal número deve seguir uma seqüência sempre crescente sem coincidência e sem saltos.

A verificação deve ser aplicada após a abertura de vários pedidos, o que será uma conseqüência da aplicação dos casos de testes de sucesso apresentados neste capítulo.

Casos de testes para os Objetivos Organizacionais

Com base na especificação dos objetivos organizacionais do sistema restaurante (Anexo A), é possível elaborar uma estratégia de teste cuja lógica de aplicação se assemelha ao teste de caminho básico (caixa-branca). Conforme explicado em capítulo anterior, a estratégia deste teste é representar cada instrução procedimental como um nó em um grafo. No caso específico da técnica de info cases, cada nó será um info case constante no grafo de admissibilidade, conforme mostra a figura 4.1 abaixo:

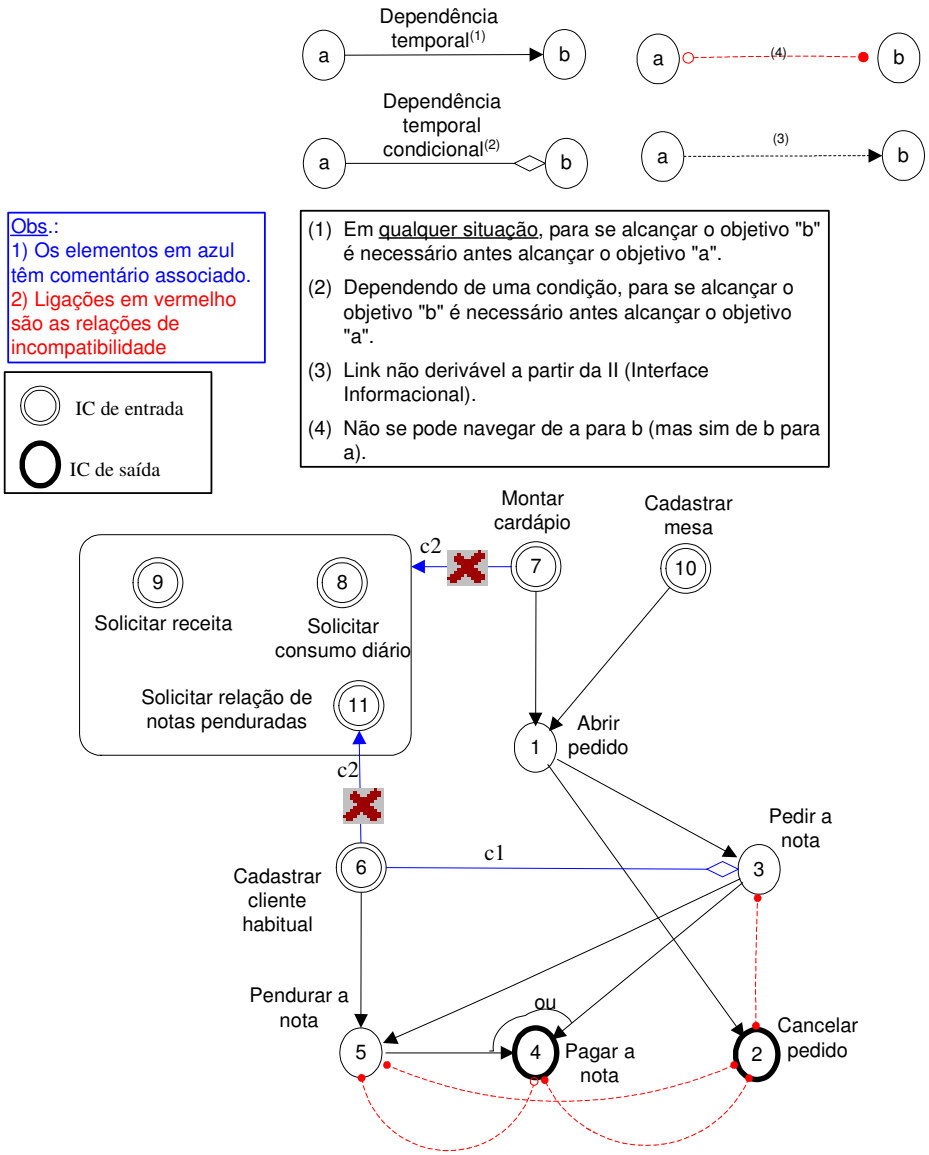


Figura 4.1. Grafo de admissibilidade – sistema Restaurante (adaptado de FORTUNA & BORGES, 2005)

Os caminhos básicos definidos para o modelo acima são:

- 7-10-1-3-4
- 7-10-1-2
- 7-10-1-3-6-5

É possível também definir as seqüências que não podem ser aceitas pelo sistema, por exemplo:

- 1-3-2
- 1-3-5-2
- 1-3-4-5

- 1-2-3
- 1-2-4
- 1-2-5

Cada seqüência listada acima, tanto para caminhos aceitáveis quanto para caminhos não permitidos, deve ser considerada para a elaboração de um caso de teste, a ser aplicado durante os testes de integração.

Os IC 8, 9 e 11 ficaram de fora dos casos de testes acima, pois não dependem da execução de nenhum outro IC, e podem ser realizados a qualquer momento, ou seja, podem ser inseridos em qualquer uma das seqüências. Como eles tratam da emissão de relatórios, e conforme definido no roteiro de testes para estes IC, é sempre importante executar teste de relatório com a base de dados ainda vazia para a prevenção de erros em consultas e campos desalinhados, caso o sistema possua uma interface para visualização dos relatórios.

4.3. Considerações sobre os casos de teste

Durante a elaboração dos casos de testes para o sistema Restaurante a partir dos info cases, alguns pontos foram observados, os quais são explicitados a seguir.

Na descrição do IC 1: "Abrir pedido", consta a seguinte informação: "(...) A mesa identificada por **nr_mesa** não pode ter pedido em aberto (ainda não pago ou pendurado).". Existe uma ambigüidade simples neste trecho: devem-se levar em consideração os pedidos pendurados ou os não pendurados? Faz sentido para o sistema que, ao se pendurar nota, a mesa seja liberada para uso por outro cliente; portanto, "em aberto" deveria ser definido como ainda não pago ou não pendurado.

Apesar deste ajuste, ao definir o que é um pedido "aberto" estaríamos criando um problema de entendimento para o IC 3 (Pedir a nota), cujo propósito é: "Identificar o pedido (em aberto) para a emissão da nota correspondente. (...)". Neste IC um pedido em aberto deve considerar os pedidos não cancelados e pendurados (ainda não pagos).

A melhor solução neste caso seria ao invés de usar o termo "pedido em aberto", usar os termos associados à identificação de estado de um pedido: cancelado, não cancelado, pendurado e não pendurado.

Ainda no IC 1 o item elementar **quant_item** foi definido como do tipo número natural, sem especificar no dicionário de itens se o valor nulo é permitido ou não. Desta forma, um caso de teste pode especificar um pedido com quantidade zero para ser aberto (tanto intencionalmente quanto acidentalmente - tentando digitar "10", sai apenas "0").

Outro ponto de questionamento para **quant_item** está relacionado à existência de limite superior. Pela especificação, qualquer quantidade pode ser informada ao solicitar um pedido, mesmo que seja superior à quantidade de cervejas existentes no estoque do restaurante, por exemplo. Como não existe, na lista de requisitos, um controle de estoques, entende-se que, de fato, a quantidade máxima não deve sofrer nenhuma limitação, cabendo ao analista propor uma melhoria futura no sistema após a integração com o controle de estoques. Estes pontos foram percebidos durante a elaboração dos casos de testes na analise do valor limite para a classe de valores aceitáveis para o item **quant_item**, momento este em que os extremos desta classe devem ser verificados.

No IC 3, um provável erro de digitação foi percebido: **pç_unit** ao invés de **pc_unit** como consta no IC 7 "Atualizar cardápio". Poderia ser um item elementar diferente, mas como não existe definição no dicionário de itens para **pç_unit** entende-se que foi apenas um erro tipográfico.

No IC 8: "Solicitar consumo diário", o **quant_item** do fluxo de saída se refere à quantidade total consumida de um item na data escolhida, segundo o dicionário de itens, enquanto o mesmo **quant_item**, presente no IC 1, se refere à quantidade solicitada de um item em apenas um pedido. Neste ponto existe um conflito na definição do domínio para **quant_item**. Uma proposta a ser feita é utilizar a idéia do teste de fluxo de dados *todas as definições todos os usos* (DU), sobre os itens elementares dos info cases, de modo a mapear a ocorrência de cada item, identificando onde são declarados (conjunto DEF(S)), onde são usados (conjunto USE(S)), além de especificar seu domínio. Construir uma tabela com tais informações torna visível onde os itens foram declarados, onde são usados e se o domínio especificado se ajusta a todas as suas ocorrências.

O IC 10 "Cadastrar mesa" não diz se várias mesas podem ser cadastradas com um mesmo número. Um caso de testes foi proposto para esta situação, e seu resultado (independente de qual for) deve ser questionado para se definir que comportamento esperar do sistema. Este caso de testes foi imaginado após a definição da classe de valores válidos para **nr_mesa**. Já que o intervalo é limitado surgiu à dúvida: pode haver repetição? Caso seja possível a repetição, outras perguntas podem ser feitas com relação ao funcionamento do sistema: como o sistema vai distinguir para qual mesa o pedido esta sendo aberto? Como os relatórios deverão se comportar? Ao pendurar uma nota, existe o risco de pendurar para um pedido de outra mesa com o mesmo número? É difícil mensurar rapidamente todos os impactos de se permitir cadastrar mais de uma mesa com o mesmo número, sendo necessário sanar tal dúvida o quanto antes.

O IC 11 “Solicitar relação de notas penduradas”, apresenta um item que não consta no dicionário de itens: **pç_item**. Pelo objetivo do IC, facilmente se percebe que se trata do preço unitário do item do cardápio no momento em foi realizado o pedido.

Uma sugestão seria substituir a ocorrência de **pç_item** por (**vl_item / quant_item**), deixando claro que **pç_item** não sofrerá alterações mesmo que o cardápio seja atualizado.

A título de conclusão mais geral, pode-se dizer que os questionamentos acima, a respeito da especificação de info cases do sistema Restaurante, trazem benefícios ao projeto quando levados ao conhecimento daquele que elaborou a especificação, visto que modificações nesta fase são rápidas de serem feitas e seu custo é baixo se comparado ao custo de correções introduzidas após a codificação ou a entrega do sistema aos usuários.

Uma das vantagens de se elaborar casos de testes diretamente sobre a especificação de requisitos é que a equipe de controle de qualidade passa a conhecer profundamente o sistema, possibilitando que sejam consultados pelos desenvolvedores quando alguma dúvida surgir. Quando chegar o momento de aplicar os testes, todos os passos estarão devidamente registrados para que cada regra de negócio seja verificada. Além disto, os casos de testes podem ser passados aos próprios desenvolvedores para que eles o utilizem em seus testes, antes de liberar seus trabalhos para a atividade de controle de qualidade.

4.4. Restaurante x Fábrica de Software – Estudo Comparativo

Neste trabalho foram elaborados casos de testes para o sistema Restaurante a partir da especificação de info cases do sistema, e casos de testes para o sistema Fábrica de Software, a partir de uma especificação composta de uma lista de requisitos e um diagrama de estados e transições (Anexos C e D). Para complementar o trabalho, o sistema Fábrica de software também foi especificado através de info cases, a partir dos quais foram derivados casos de teste (Anexo E). Esta seção descreve os resultados da comparação entre os diversos conjuntos de casos de testes produzidos.

Os info cases contém informações que permitem identificar de imediato alguns tipos de testes que podem ser aplicados no sistema. O formalismo utilizado nos info cases evidencia o fluxo de dados necessários para que um objetivo de ator possa ser atingido, sua notação torna intuitivo o entendimento causando a impressão de que o leitor esta diante de uma interface do sistema. Para um testador, esta visão de um sistema que ainda não existe, em conjunto com a definição proposta para cada item elementar constante no dicionário de

itens, é suficiente para que estratégias de testes como particionamento de equivalência e análise do valor limite possam ser elaborados.

O grafo de admissibilidade completa a visão das ações e seu encadeamento, evidenciando os requisitos necessários para a conclusão de um objetivo de um ator, ou seja, durante elaboração dos roteiros o testador terá o entendimento necessário para exigir que o sistema se comporte conforme esperado.

Os casos de testes elaborados a partir dos info cases para o sistema Restaurante ficaram mais detalhados dos que os elaborados para o sistema Fábrica de Software com base apenas na lista de requisitos e no diagrama de estados e transições. A razão para esta diferença é que a especificação por info cases traz de forma completa as informações técnicas necessárias à construção do sistema, enquanto a lista de requisitos e o diagrama de estados fornecem uma orientação do como deve ser realizado o desenvolvimento. Quanto mais precisa for a especificação, mais claro fica para o testador como verificar o sistema.

É importante salientar que o estudo da MIR influenciou no momento de elaborar os casos de testes para o sistema Fábrica de software, mesmo antes de ter os infocases definidos. Isto porque foi percebida pelo autor deste trabalho uma semelhança entre a realização de uma transição em um diagrama de estados com a realização de um IC. Desta forma, foram criados casos de testes limitados a cada transição do diagrama assim como foram criados casos de testes limitados a cada IC.

Tal semelhança foge do escopo deste trabalho e pode ser explorada em trabalhos futuros: propor uma relação entre info case e transições em um diagrama de estados, e relacionar os estados estáveis da MIR aos estados do diagrama, ou seja, uma equivalência entre um workflow e o modelo de info case.

Uma dificuldade encontrada durante a elaboração dos testes para os infocases foi o nível de abstração utilizado para os identificadores e para a interface sistema/usuário. Segundo o dicionário de itens, quando um id participa do fluxo de entrada de um IC qualquer, ele é tratado como um número natural. No entanto, nenhum sistema irá exigir que seus usuários memorizem números para que possam referenciar os objetos instanciados pelo sistema.

O id é utilizado internamente pelo sistema, sendo que na interface este número será convertido em um dos atributos do objeto correspondente. Esta abstração não deixa claro qual atributo será usado; por causa disto o teste para o atributo não poderá ser idealizado e também não deve ser elaborado um teste para exercitar o número natural correspondente ao id em questão, pois o usuário não terá um controle direto sobre sua inserção.

Com relação à interface, os casos de testes devem ser elaborados tendo em mente a forma como os sistemas usualmente são desenvolvidos dentro do processo de desenvolvimento. Por exemplo, a implementação de um IC com entrada de **id_item** para abertura de um pedido (sistema Restaurante) pode ser feita de modo que o Garçom digite manualmente o nome de um item, ou então os itens do cardápio serão listados em uma tabela para que ele escolha. A escolha do item poderá ainda ser filtrada pela categoria (comida ou bebida) caso este dado seja inserido antes do item, ou a categoria será automaticamente inserida depois de definido o item.

Tais diferenças causam alterações nos casos de testes que não puderam ser previstas pela MIR, sugerindo que os casos de testes produzidos serão influenciados pela forma de trabalho da empresa na qual a técnica venha a ser utilizada. Apesar das dificuldades relacionadas ao id, ainda foi possível definir algumas classes de equivalência como, por exemplo, a condição de entrada booleana.

Elaborar casos de testes diretamente a partir de uma lista de requisitos é uma tarefa difícil e delicada de ser desempenhada, pois a lista nem sempre condensa todas as informações em um único tópico, além de não definir com exatidão o fluxo de dados.

Por esta razão, elaborar casos de testes com base em uma especificação que apresenta formalismos como é o caso da MIR, tornou mais rápida e objetiva a tarefa de elaborar os casos de testes, apesar das limitações relacionadas à forma de trabalho e de poderem ser previstos apenas casos de testes tipo caixa-preta.

Capítulo 5
Conclusão

Este trabalho investigou as técnicas de elaboração de casos de teste e como elas poderiam ser aplicadas quando o sistema é especificado por info cases. Foram apresentadas diversas diretrizes para isso, exemplificando-as através de seu uso em um sistema. Além disso, procurou-se comparar os casos de teste delas resultantes, com os casos de teste obtidos quando o sistema não é especificado através de info cases.

A principal contribuição deste trabalho foi prover uma orientação para o teste de sistemas especificados através de info cases. Ou seja, o testador passa a ter algumas diretrizes para a elaboração de casos de teste a partir de info cases. Espera-se, como conseqüência, uma maior agilidade, uniformidade e qualidade dos casos de uso construídos, pois os mesmos deixam de ser feitos de forma puramente *ad hoc*.

Algumas limitações estão presentes no trabalho, decorrentes principalmente do tempo que se teve para sua realização. Por exemplo, a contribuição dos testes caixa-branca foi pouco explorada. Acredita-se que o teste baseado em fluxo de dados (seção 2.2.3) possa ter um papel mais relevante na derivação de casos de teste para info cases. Outra limitação está na avaliação das diretrizes produzidas. Além de estar calcada apenas em estudos individuais, os resultados do estudo comparativo empreendido podem ter sofrido influência do aprendizado que o experimentador obteve por conta dos testes com info cases, quando ele foi elaborar os casos de teste para o outro tipo de especificação (lista de requisitos e diagrama de estados).

Por isso, uma sugestão de trabalho futuro é a realização de estudos experimentais mais planejados para uma melhor avaliação da cobertura e qualidade dos casos de teste obtidos de info cases. Outro trabalho futuro relevante seria tentar aprofundar a compreensão sobre a semelhança entre a ativação de uma transição em um diagrama de estados com a ativação de um info case. Um dos pontos a investigar seria a relação entre os estados estáveis da modelagem com info cases e os estados no diagrama de transições, ou em outras palavras, a relação entre um workflow e o modelo de info cases. Por fim, sugere-se também o desenvolvimento de uma ferramenta que permita apoiar a elaboração dos casos de testes, adaptada à modelagem de requisitos com info cases.

Referências Bibliográficas

BEIZER, B., **Software Testing Techniques**, 2° Ed., Van Nostrand Reinhold, 1990.

FORTUNA, M., BORGES, M.R.S. **Modelagem Informacional de Requisitos.** *VIII Workshop on Requirements Engineering*, Porto, Portugal, 2005.

FORTUNA, M; WERNER, C. M.; BORGES, M.R.S. **Um Modelo Integrado de Requisitos com Casos de Uso.** In: Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes de Software, Isla Margarita, Venezuela, 2007.

FORTUNA, M. H., 2008, "Um Modelo Integrado de Requisitos com Casos de Uso", Tese de Doutorado, Universidade Federal do Rio de Janeiro (UFRJ), Rio de Janeiro, Brasil, Dezembro.

GELPERIN, D., HETZEL, B., **The Growth of Software Testing**. - Communications of the ACM nº 31, junho de 1988, pp. 687-95.

GRAHAM, D., **Requirements and Testing: Seven Missing-Link Myths**, IEEE Software Magazine, outubro de 2002. pp 15-17.

JACOBSON, I. et al. **Object-Oriented software Engineering – A User Casen Driven Approach.** Addison-Wesley, 1992.

KNUTH, D., 1964, "Backus Normal Form vs. Backus Naur Form", Communications of the ACM, v.7, n.12, pp. 735-736.

MALDONADO, J. C., **Critérios Potenciais usos: Uma contribuição ao Teste Estrutural de Software**, Teste de doutorado – UNICAMP – 1991

MYERS, G., J**. The Art of Software Testing**, Wiley & Sons, 1978.

OBJECT MANAGEMENT GROUPG (OMG) UML 2.0 Spacification. Disponível em VISUAL PARADIGM INTERNATIONAL. Visual Paradigm. Disponível em: http://www.uml.org/. Acesso em março, 2009.

PRESSMAN, R., S. **Engenharia de software**. MAKRON Books, São Paulo, 2006.

WHITE, L. J., COHEN, E. I., "**A Domain Strategy for Program Testing**", IEEE Trans. Software Engineering, vol. SE-6, n° 5, maio de 1980, pp. 247-257.

WEYUKER, E. J., **The Cost of Data Flow Testting: An Empirical Study**, IEEE Trans. On Software Engieering, vol.SE-16, n° 2, 1990.

Apêndice A

Especificação Técnica Sistema Restaurante

SUMÁRIO

LISTA DE REQUISITOS

Um restaurante deseja um sistema para o controle do seu funcionamento. As seguintes funções deverão ser desempenhadas pelo sistema:

1. Registrar o pedido de pratos e bebidas de cada cliente. Este registro servirá para, entre outras coisas, a emissão do ticket (nota), no encerramento da refeição.

2. Registrar o cancelamento de um pedido (refeição não realizada).

3. Emitir a nota com as seguintes informações: número da mesa, número do pedido, data do pedido, nome de cada prato e bebida consumidos juntamente com seu preço unitário, quantidade consumida e preço total. No final do tícket deverá aparecer o valor total do pedido, a ser pago pelo cliente (esse valor é calculado a partir do que foi consumido e acrescentando 10% a título de taxa de serviço). No caso de clientes habituais, (conhecidos do restaurante) pode aparecer o nome e o telefone do cliente.

4. Manter um registro dos clientes habituais do restaurante, aos quais é facultada a pendura de notas.

5. Registrar a pendura da nota pelo cliente. A pendura da nota só é facultada aos clientes habituais do restaurante.

6. Registrar o pagamento de uma nota aberta, ou de uma ou mais notas penduradas, de um cliente. O sistema deverá auxiliar o caixa do restaurante, computando o valor do troco a partir do montante entregue pelo cliente para o pagamento da(s) nota(s).

7. Apresentar o consumo (diário) de pratos e bebidas (para fins de reposição de estoque), emitindo, ao final do dia, um relatório indicando o total de unidades consumidas, de cada prato ou bebida. O relatório deverá listar o nome dos pratos e bebidas separadamente, e em ordem decrescente da quantidade consumida.

8. Apresentar a receita do restaurante, realizada e a realizar (pelo pagamento das notas penduradas), considerando um período de tempo passado, compreendido entre duas datas fornecidas.

9. Apresentar a relação de notas penduradas dentro de um período de tempo passado, compreendido entre duas datas. O relatório deverá exibir a identificação de cada cliente, seu telefone, os detalhes das notas penduradas e o valor total pendurado pelo cliente. Ao final, apresentar também o crédito total do restaurante representado pelas notas penduradas. A principal finalidade é a cobrança de notas penduradas.

10. Manter atualizado o cardápio, pela inclusão, alteração ou eliminação de ítens de consumo (prato ou bebida).

O restaurante serve, em média, 250 refeições por dia. O restaurante possui, atualmente, 30 mesas, mas esse número poderá variar.

INFO CASES

ATOR: Cliente IC 1: Abrir pedido

➔ pedido = dt_pedido + id_mesa + itens_ped

Descrição: Informações de um pedido de refeição. A mesa identificada por **nr_mesa** não pode ter pedido em aberto (ainda não pago ou pendurado).
Propósito: Informar ao sistema o início de uma nova refeição e o que será nela consumido.
Freqüência: 250/dia.

itens_ped = $_1${id_item + quant_item}

Descrição: Informações sobre os itens (pratos e bebidas) a serem consumidos.

➔ id_pedido

ATOR: Cliente IC 2: Cancelar pedido

➔ cancela_ped = id_pedido

Propósito: Identificar o pedido a ser cancelado (pedido aberto na mesa indicada).

ATOR: Cliente IC 3: Pedir a nota

➔ solicitacao_nota = id_pedido + [id_cliente]

Propósito: Identificar o pedido (em aberto) para a emissão da nota correspondente. Opcionalmente, também pode identificar o cliente, se ele for um cliente habitual.

➔ nota = id_pedido + nr_mesa + dt_pedido + itens_nota + vl_nota + [nome_cliente + tel_cliente]

Descrição: Ticket impresso contendo informações sobre o consumo e o valor a pagar. Deve ser emitido mesmo no caso de pedido a ser pendurado, caso em que pode já incluir o nome e o telefone do cliente.
Propósito: Permitir a conferência, pelo cliente, do que ele consumiu e do valor a pagar.

itens_nota = $_1${id_item + cat_item + nome_item + pç_unit + quant_item + vl_item}

Descrição: Informações sobre os itens (pratos e bebidas) consumidos.
Ordenação: Crescente de **cat_item** ++ **nome_item**.

ATOR: Cliente IC 4: Pagar a nota

➔ pagamento = id_pedido + vl_entregue + dt_pagto

Propósito: Pagar uma nota (em aberto ou pendurada).

➔ troco

Propósito: Informar ao caixa do restaurante o valor monetário a ser devolvido ao cliente (troco).

ATOR: Cliente IC 5: Pendurar a nota

➔ **pendura = id_pedido + id_cliente**

Propósito: Identificar a nota (em aberto, de um cliente habitual) a ser pendurada.

ATOR: Gerente IC 6: Cadastrar cliente habitual

➔ **cliente = nome_cliente + tel_cliente**

Descrição: Informações sobre um cliente habitual do restaurante. Apenas clientes habituais têm direito a pendurar notas.

🡐 **id_cliente**

ATOR: Gerente IC 7: Atualizar o cardápio

➔ **item_consumo = nome_item + pc_unit + cat_item + descr_item**

Descrição: Informações sobre um item de consumo (prato ou bebida) a ser incluído no cardápio do restaurante.

🡐 **id_item**

ATOR: Gerente IC 8: Solicitar consumo diário

➔ **solic_consumo = dt_emissao + dt_consumo**

Propósito: Selecionar a data para a qual se deseja o relatório de consumo.

🡐 **consumo_dia = dt_emissao + dt_consumo + consumo_itens**

Descrição: Informações sobre o consumo na data escolhida. Se a data de emissão for igual à data do consumo, e a consulta for realizada antes do fechamento do dia de funcionamento do restaurante, o resultado da consulta poderá apresentar valores apenas parciais.
Propósito: Facilitar a reposição do estoque de insumos do restaurante.

🗐 **consumo_itens = $_0${cat_item + {id_item + nome_item + quant_item} }$_2$**

Descrição: Informações sobre o consumo de itens, para cada categoria (pratos e bebidas).
Ordenação: Crescente de **cat_item**; decrescente de **quant_item**.

ATOR: Gerente IC 9: Solicitar receita

➔ **solic_receita = periodo_apur + dt_emissao**

Propósito: Selecionar o período para o qual se deseja apurar a receita do restaurante.

🗐 **periodo_apur = dt_inicio + dt_fim**

Descrição: Datas iniciais e finais (inclusive) do período de apuração da receita.

🡐 receita = dt_emissao + periodo_apur + vl_consumo + receita_realiz + receita_pend + receita_txServ + receita_total

Descrição: Informações sobre a receita oriunda dos pedidos abertos e não cancelados, dentro do período de apuração. Se a data de emissão for igual à data do final do período de apuração, e a consulta for realizada antes do fechamento do dia de funcionamento do restaurante, o resultado da consulta poderá apresentar valores apenas parciais.
Propósito: Facilitar o acompanhamento, pela gerência, dos resultados financeiros do restaurante (realizados e a realizar), e permitir a distribuição aos garçons da receita proveniente da taxa de serviço.

ATOR: Gerente IC 10: Cadastrar mesa

➔ mesa = nr_mesa

Descrição: Número de uma mesa do restaurante.
Propósito: Serve como identificador de uma mesa.

🡐 id_mesa

ATOR: Gerente IC 11: Solicitar relação de notas penduradas

➔ solic_penduras = periodo_apur + dt_emissao

Propósito: Selecionar o período para o qual se deseja apurar as notas penduradas.

periodo_apur = dt_inicio + dt_fim

Descrição: Período de apuração das notas penduradas.

🡐 penduras = dt_emissao + periodo_apur + {id_cliente + nome_cliente + tel_cliente + notas_pend + vl_pendCli} + vl_pend

Descrição: Relação de notas penduradas no período de apuração. Se a data de emissão for igual à data do final do período de apuração, e a consulta for realizada antes do fechamento do dia de funcionamento do restaurante, o resultado da consulta poderá apresentar valores apenas parciais.
Propósito: 1) Identificar as notas penduradas de um cliente, quando o mesmo deseja quitá-las mas não está com a cópia do ticket que lhe foi entregue na data do consumo; 2) Apoiar o gerente no processo de cobrança das notas penduradas.
Ordenação: Crescente de **nome_cliente**.

notas_pend = $_1${id_pedido + dt_pedido + nr_mesa + itens_nota + vl_nota}

Descrição: Relação das notas penduradas do cliente em questão (**id_cliente**), dentro do período de apuração escolhido.
Ordenação: Crescente de **dt_pedido**.

itens_nota = $_1${id_item + cat_item + nome_item + pç_item + quant_item + vl_item}

Descrição: Informações sobre os itens (pratos e bebidas) consumidos, que compõem a nota em questão.
Ordenação: Crescente de **cat_item** ++ **nome_item**.

DICIONÁRIO DE ITENS

IC 1 Abrir pedido

E/S	Nome	Descrição	Tipo Base	Domínio	Unidad	Precisão
E	dt_pedido	Data em que o pedido foi efetuado. Default: data corrente no momento do registro do pedido no sistema.	Data	Menor ou igual à data corrente.		
E	id_mesa	Identificador de uma mesa do restaurante.	Nr.Natural	Um dos números gerados no cadastramento de mesa, e que corresponda a uma mesa sem pedido em aberto (ainda não pago ou pendurado).		
E	id_item	Identificador de um item pedido (prato ou bebida do cardápio).	Nr.Natural	Um dos números gerados na atualização do cardápio.		
E	quant_item	Quantidade pedida do item.	Nr.Natural			
S	id_pedido	Identificador de um pedido.	Nr.Automát[3]			

IC 2 Cancelar pedido

E/S	Nome	Descrição	Tipo Base	Domínio	Unidad	Precisão
E	id_pedido	Identificador do pedido em aberto a ser cancelado.	Nr.Natural	Um dos números gerados na abertura de um pedido, e que corresponda a um pedido em aberto (ainda não pago ou pendurado).		

IC 3 Pedir a nota

E/S	Nome	Descrição	Tipo Base	Domínio	Unidad	Precisão
E	id_pedido	Identificador de um pedido em aberto.	Nr.Natural	Um dos números gerados na abertura de pedido, e que corresponda a um pedido em aberto (ainda não pago ou pendurado).		
E	id_cliente	Identificador do cliente habitual do restaurante. Só no caso do cliente ser freqüentador habitual do restaurante.	Nr.Natural	Um dos números gerados no evento de cadastramento de cliente habitual.		
S	nr_mesa	Número da mesa onde se realiza a refeição correspondente ao pedido cuja nota foi solicitada.	Nr.Natural			
S	quant_item	Quantidade consumida de um item, no pedido em questão.	Nr.Natural			

[3] Número gerado automaticamente pelo sistema, para ser o identificador de um pedido.

IC 3 Pedir a nota (continuação...)

E/S	Nome	Descrição	Tipo Base	Domínio	Unidad	Precisão
S	vl_item	Valor do consumo de um item. É calculada a partir do preço unitário do item e da quantidade dele consumida.	Moeda		real	Centavo
S	vl_nota	Valor total da nota. Corresponde ao somatório do valor de cada item consumido, mais um acréscimo de 10% a título de taxa de serviço.	Moeda		real	Centavo

IC 4 Pagar a nota

E/S	Nome	Descrição	Tipo Base	Domínio	Unidad	Precisão
E	id_pedido	Identificador do pedido a ser pago.	Nr.Natural	Um dos identificadores gerados na abertura de pedido, correspondente a um pedido ainda não pago (em aberto ou pendurado)..		
E	dt_pagto	Data do pagamento. Valor default: data corrente no momento do pagamento.	Data	Maior ou igual à data do pedido que esteja sendo pago.		
E	vl_entregue	Valor entregue pelo cliente para pagamento da nota.	Moeda	Maior ou igual ao valor a ser pago (valor da nota).	real	Centavo
S	troco	Diferença entre o valor entregue pelo cliente para pagamento da nota e o valor da nota.	Moeda		real	Centavo

IC 5 Pendurar a nota

E/S	Nome	Descrição	Tipo Base	Domínio	Unidad	Precisão
E	id_pedido	Número (identificador) do pedido cuja nota se deseja pendurar.	Nr.Natural	Um dos números gerados na abertura de pedido, correspondente a um pedido em aberto.		
E	id_cliente	Identificador do cliente habitual responsável pelo pedido a pendurar. Default: identificador de cliente informado no evento de solicitação da nota (caso tal identificador tenha sido lá informado).	Nr.Natural	Um dos números gerados no evento de cadastramento de clientes habituais.		

IC 6 Cadastrar cliente habitual

E/S	Nome	Descrição	Tipo Base	Domínio	Unidad	Precisão
E	nome_cliente	Nome de um cliente habitual do restaurante.	Texto			
E	tel_cliente	Telefone do cliente habitual.	Telefone			
S	id_cliente	Identificador do cliente habitual.	Nr.Automát			

IC 7 Atualizar o cardápio

E/S	Nome	Descrição	Tipo Base	Domínio	Unidad	Precisão
E	nome_item	Nome de um item de consumo (do cardápio).	Texto			
E	cat_item	Categoria do item de consumo	Texto	{bebida, prato}		
E	pc_unit	Preço cobrado (no cardápio) por uma unidade do item.	Moeda		real	Centavo
E	descr_item	Descrição do item de consumo. Em geral, só para itens mais sofisticados, como por exemplo, coquetéis.	Texto			
S	id_item	Identificador do item de consumo.	Nr.Automát			

IC 8 Solicitar consumo diário

E/S	Nome	Descrição	Tipo Base	Domínio	Unidad	Precisão
E	dt_consumo	Data do dia do qual se deseja o relatório de consumo. Valor default: data corrente.	Data	Igual ou menor do que a data corrente.		
S	id_item	Identificador de um item consumido na data escolhida (dt_consumo).	Nr.Natural			
S	quant_item	Quantidade total consumida do item, na data escolhida.	Nr.Natural			

IC 9 Solicitar receita

E/S	Nome	Descrição	Tipo Base	Domínio	Unidad	Precisão
E	dt_inicio	Data inicial do período de apuração da receita.	Data	Menor ou igual à data corrente.		
E	dt_fim	Data final do período de apuração da receita.	Data	Menor ou igual à data corrente e maior ou igual à data inicial do período de apuração.		
S	vl_consumo	Valor total do consumo, no período de apuração. Calculado a partir das quantidades consumidas e do preço unitário de cada item consumido.	Moeda		real	Centavo
S	receita_realiz	Valor total da receita efetivamente auferida (realizada), proveniente das notas pagas no período de apuração. Não inclui a parte relativa à taxa de serviço.	Moeda		real	Centavo
S	receita_pend	Valor total da receita a realizar, referente a notas penduradas e ainda não pagas, dentro do período de apuração. Inclui a parte relativa a taxa de serviço.	Moeda		real	Centavo
S	receita_txServ	Valor total da receita efetivamente auferida (realizada) a título de taxa de serviço, proveniente das notas pagas dentro do período de apuração.	Moeda		real	Centavo
S	receita_total	Valor total da receita, dentro do período de apuração. Resulta do somatório das receitas realizada, pendurada e referente à taxa de serviço.	Moeda		real	Centavo

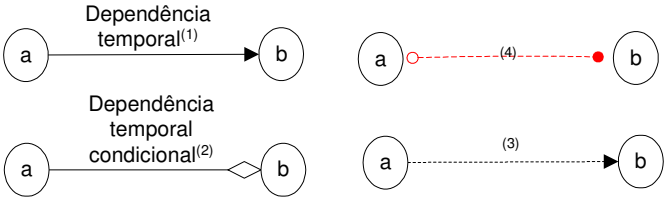
IC 10 Cadastrar mesa

E/S	Nome	Descrição	Tipo Base	Domínio	Unidad	Precisão
E	nr_mesa	Número identificador de uma mesa, atribuído pela gerência do restaurante.	Nr.Natural	{1..50}		
S	id_mesa	Número identificador de uma mesa, gerado pelo sistema.	Nr.Automát			

IC 11 Solicitar relação de notas penduradas

E/S	Nome	Descrição	Tipo Base	Domínio	Unidad	Precisão
E	dt_inicio	Data inicial do período de apuração das notas penduradas.	Data	Menor ou igual a data corrente.		
E	dt_fim	Data final do período de apuração das notas penduradas.	Data	Menor ou igual a data corrente e igual ou maior que a data inicial do período de apuração (dt_inicio).		
S	id_cliente	Identificador de um cliente com nota pendurada no período de apuração escolhido.	Nr.Natural			
S	id_pedido	Identificador de um pedido correspondente a uma nota pendurada do cliente, dentro do período de apuração escolhido.	Nr.Natural			
S	vl_pendCli	Valor total pendurado do cliente. Resulta do somatório do valor de cada nota pendurada do cliente, no período de apuração.	Moeda		real	Centavo
S	vl_pend	Valor total pendurado. Resulta do somatório de todas as notas penduradas, de todos os clientes, no período de apuração.	Moeda		real	Centavo

GRAFO DE ADMISSIBILIDADE



Obs.:
1) Os elementos em azul têm comentário associado.
2) Ligações em vermelho são as relações de incompatibilidade

IC de entrada
IC de saída

(1) Em qualquer situação, para se alcançar o objetivo "b" é necessário antes alcançar o objetivo "a".
(2) Dependendo de uma condição, para se alcançar o objetivo "b" é necessário antes alcançar o objetivo "a".
(3) Link não derivável a partir da II (Interface Informacional).
(4) Não se pode navegar de a para b (mas sim de b para a).

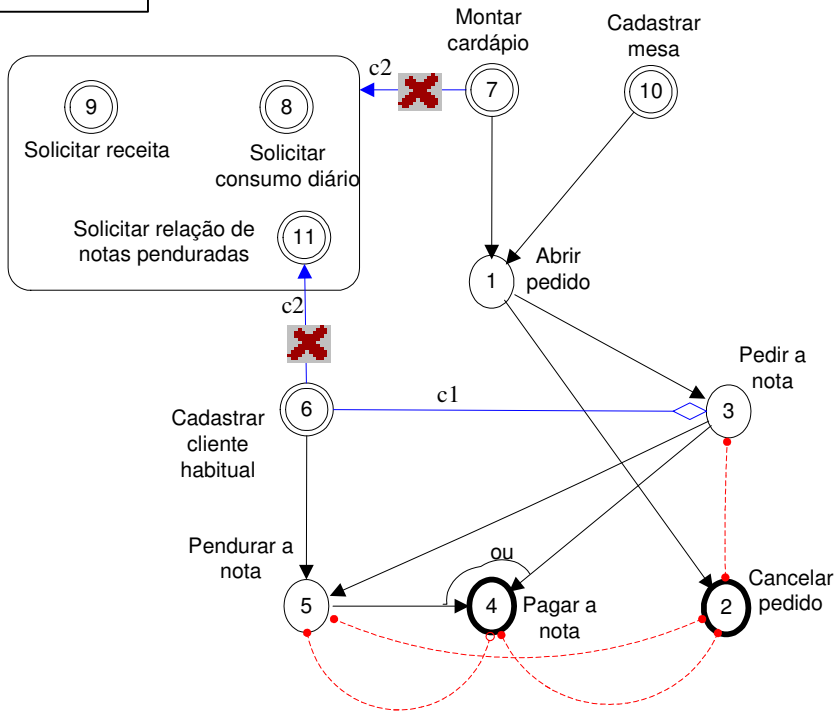
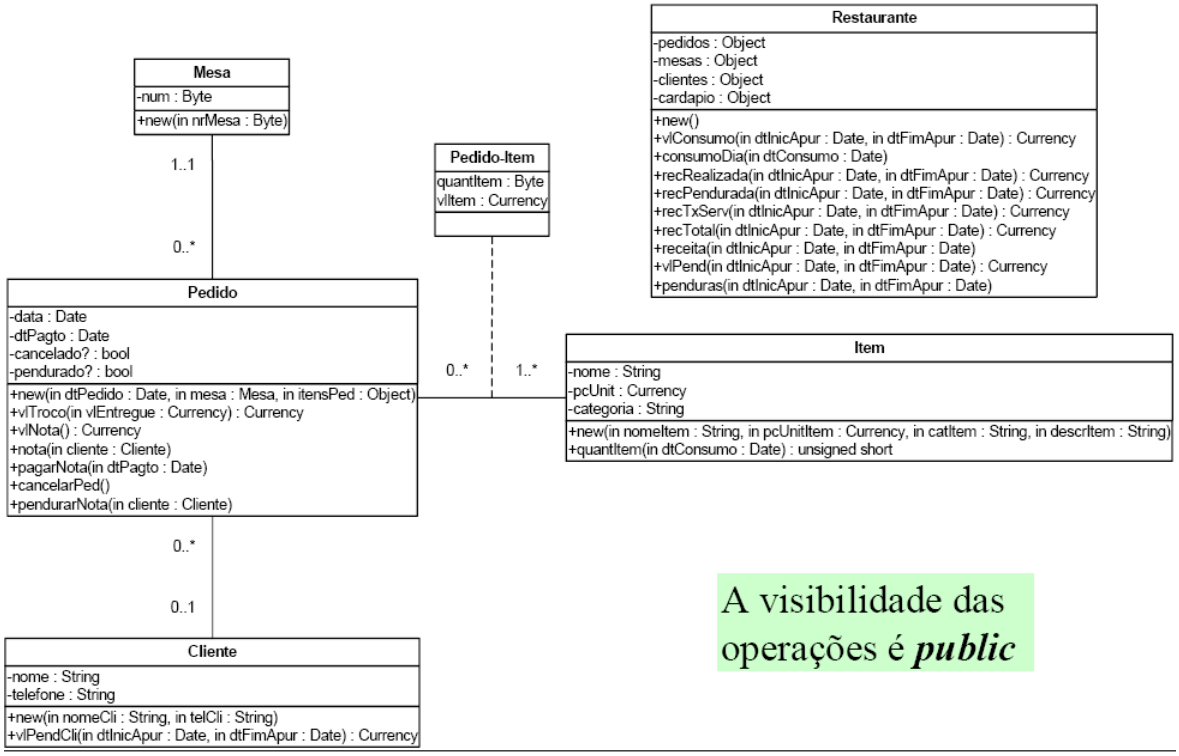


DIAGRAMA DE CLASSES



A visibilidade das operações é ***public***

Apêndice B

Casos de testes – Sistema Restaurante

Sumário

ATOR: Garçom IC 1: Abrir pedido

➔ pedido = dt_pedido + id_mesa + itens_ped

Descrição: Informações de um pedido de refeição. A mesa identificada por **nr_mesa** não pode ter pedido em aberto (ainda não pago ou pendurado).
Propósito: Informar ao sistema o início de uma nova refeição e o que será nela consumido.
Freqüência: 250/dia.

itens_ped = $_1${id_item + quant_item}

Descrição: Informações sobre os itens (pratos e bebidas) a serem consumidos.

➔ id_pedido

Fluxo de entrada
dt_pedido
(Particionamento de equivalência)
Condição de entrada, booleano
- sistema deve preencher automaticamente com a data corrente.
- apagar a data do campo: sistema deve exigir a presença de uma data para abertura do pedido

Condição de entrada, intervalo
- inserir uma data posterior a data corrente: sistema deve bloquear a abertura de pedido
- inserir data igual ou inferior à data corrente: pedido deve ser aberto com sucesso

(Análise do valor limite)
- inserir datas que estejam fora do intervalo aceitável para dias: ‘00’ e ‘32’: sistema deve bloquear a ação.
- inserir datas que estejam fora do intervalo aceitável para meses: ‘00’ e ‘13’: sistema deve bloquear a ação.

id_mesa
(Particionamento de equivalência)
Condição de entrada, booleano: número da mesa deve ser inserido ao abrir pedido.
- abrir pedido sem o número da mesa: sistema deve validar a presença do número
- abrir pedido com o número da mesa: pedido deve ser aberto com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens
- solicitar pedido para uma mesa que tenha um pedido em aberto (não pago, ou pendurado): sistema deve bloquear a ação.
- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.
- solicitar pedido inserindo número de mesa que pertença ao intervalo [1, 50]: pedido deve ser realizado com sucesso

(Análise do valor limite)
- solicitar pedido em que o número da mesa seja igual a zero: sistema deve bloquear a ação
- solicitar pedido com número da mesa maior do que a quantidade de mesas do estabelecimento: sistema deve bloquear a ação
- solicitar pedidos em que os números das mesas sejam iguais aos extremos da quantidade de mesas do estabelecimento: pedido deve ser aberto com sucesso

itens_ped
(Particionamento de equivalência)
Condição de entrada, booleano:

- realizar pedido informando apenas um item e sua quantidade: caso de sucesso
- realizar pedido inserindo apenas o item: sistema deve bloquear a ação
- realizar pedido inserindo apenas a quantidade: sistema deve bloquear a ação

Além dos testes acima, é necessário realizar a verificação de ocorrência mínima, conforme indicado no IC:

- realizar pedido sem informar nenhum item nem sua quantidade: sistema deve impedir a ação

Como a composição do pacote já foi verificada, não é necessário realizar o teste de condição de entrada lógica para os itens que o compõem.

id_item:

Condição de entrada, intervalo: itens solicitados devem constar no cardápio

- solicitar um item cujo **id_item** não esteja no cardápio: sistema deve bloquear a ação
- solicitar um item cujo **id_item** esteja no cardápio: pedido deve ser aberto com sucesso

quant_item:

Condição de entrada, intervalo: número Natural

- informar quantidade igual a zero: sistema deve aceitar quantidade informada
- informar a quantidade com um número fracionário: sistema deve bloquear a ação

(Análise do valor limite)

- informar quantidade igual a um: caso de sucesso, quantidade informada deve ser aceita

Fluxo de saída

id_pedido

(Particionamento de equivalência)

Condição de saída, intervalo

- sistema gera um número correspondente ao pedido aberto, tal número deve seguir uma seqüência sempre crescente sem coincidência e sem saltos.

ATOR: Garçom IC 2: Cancelar pedido

➔ cancela_ped = id_pedido

Propósito: Identificar o pedido a ser cancelado (pedido aberto na mesa indicada).

Fluxo de entrada

id_pedido

(Particionamento de equivalência)

Condição de entrada, booleano: pedido deve ser identificado

- solicitar cancelamento sem passar id_pedido: sistema deve impedir a ação

Condição de entrada, intervalo: IC aplicável somente a pedidos em aberto

- solicitar cancelamento para um pedido já cancelado: sistema deve bloquear a ação
- solicitar cancelamento para um pedido pendurado: sistema deve bloquear a ação
- solicitar cancelamento para um pedido em aberto: cancelamento deve ser realizado com sucesso
- informar um número de pedido em formato diferente do número automático gerado pelo sistema: sistema deve impedir a ação

(Análise do valor limite)

- informar um número de pedido igual a zero: sistema deve impedir a ação
- informar um número de pedido uma unidade acima do maior número de pedido existente: sistema deve impedir a ação

ATOR: Garçom IC 3: Pedir a nota

➔ **solicitacao_nota = id_pedido + [id_cliente]**

Propósito: Identificar o pedido (em aberto) para a emissão da nota correspondente. Opcionalmente, também pode identificar o cliente, se ele for um cliente habitual.

🡐 **nota = id_pedido + nr_mesa + dt_pedido + itens_nota + vl_nota + [nome_cliente + tel_cliente]**

Descrição: Ticket impresso contendo informações sobre o consumo e o valor a pagar. Deve ser emitido mesmo no caso de pedido a ser pendurado, caso em que pode já incluir o nome e o telefone do cliente.
Propósito: Permitir a conferência, pelo cliente, do que ele consumiu e do valor a pagar.

🗐 **itens_nota = $_1${id_item + cat_item + nome_item + pc_unit + quant_item + vl_item}**

Descrição: Informações sobre os itens (pratos e bebidas) consumidos.
Ordenação: Crescente de **cat_item** ++ **nome_item**.

Fluxo de entrada

id_pedido

(Particionamento de equivalência)
Condição de entrada, booleano: pedido deve ser identificado ao pedir a nota
- solicitar nota sem passar **id_pedido**: sistema deve impedir a ação
- solicitar nota passando **id_pedido** válido: ação executada.

Condição de entrada, intervalo: um dos pedidos em aberto
- informar pedido que esteja cancelado: sistema deve impedir a ação
- informar pedido que esteja pendurado: nota deve ser emitida
- informar um número de pedido em formato diferente do número automático gerado

(Análise do valor limite)
- informar um número de pedido igual a zero: sistema deve impedir a ação
- informar um número de pedido uma unidade acima do maior número de pedido existente: sistema deve impedir a ação

id_cliente

Condição de entrada, booleano: cliente pode ou não ser informado
- solicitar nota informando o cliente: sistema deve emitir a nota
- solicitar nota sem informar o cliente: sistema deve emitir a nota

Condição de entrada, intervalo: somente clientes cadastrados no sistema
- informar **id_cliente** em um formato diferente do gerado automaticamente: sistema deve impedir ação
- informar **id_cliente** superior ao maior id cadastrado: sistema deve impedir ação
- informar **id_cliente** cadastrado: sistema deve permitir ação

(Análise do valor limite)
- informar número igual a zero: sistema deve impedir a ação
- informar número superior ao maior número de cadastro existente: sistema deve impedir a ação
- informar número em formato diferente do gerado pelo sistema : sistema deve impedir a ação

Fluxo de saída

nota

- todos os itens listados na nota devem seguir ordenação por **cat_item** e em seguida por **nome_item**
- itens listados na nota devem ser apenas os constantes do pedido identificado por id_pedido

id_pedido

- deve ser o mesmo número passado ao solicitar a nota

nr_mesa
- numero da mesa correspondente ao pedido realizado

dt_pedido
- deve ser a data do pedido

id_item
- identificação do item

cat_item
- categoria relativa ao item identificado por **id_item**

nome_item
- nome relativo ao item identificado por **id_item**

pc_unit
- preço unitário relativo ao item identificado por **id_item** exibido em reais

quant_item
- quantidade do item identificado por **id_item** de acordo com o pedido do qual a nota foi solicitada

vl_item
- valor obtido pelo produto entre **quant_item** e **pc_unit** exibido em reais

vl_nota
- valor obtido pela soma entre todos os **vl_item** e acrescido de 10%. Exibido em reais

nome_cliente
- nome do cliente caso tenha sido identificado o cliente no momento de pedir a nota

tel_cliente
- telefone de contado do cliente caso tenha sido identificado o cliente no momento de pedir a nota

ATOR: Garçom IC 4: Pagar a nota

➔ pagamento = id_pedido + vl_entregue + dt_pagto

Propósito: Pagar uma nota (em aberto ou pendurada).

← troco

Propósito: Informar ao caixa do restaurante o valor monetário a ser devolvido ao cliente (troco).

Fluxo de entrada

pagamento

(Particionamento de equivalência)

Condição de entrada, intervalo: pagamento somente pode ser realizado para pedidos em aberto ou pendurados

- solicitar pagamento para pedido em aberto e ainda não pago: caso de sucesso
- solicitar pagamento para pedido pendurado: caso de sucesso
- solicitar pagamento para pedido cancelado: sistema deve bloquear ação

id_pedido

(Particionamento de equivalência)

Condição de entrada, booleano: pedido deve ser identificado ao efetuar pagamento da nota

- solicitar pagamento sem passar o número do pedido: sistema deve impedir ação
- solicitar pagamento passando o número do pedido: ação deve ser executada

Condição de entrada, intervalo: um dos pedidos em aberto ou pendurado

- informar pedido que esteja cancelado: sistema deve impedir a ação

- informar pedido que esteja pendurado: sistema deve permitir ação
- informar um número de pedido em formato diferente do número gerado automaticamente: sistema deve impedir ação

(Análise do valor limite)

- informar um número de pedido igual a zero: sistema deve impedir a ação
- informar um número de pedido uma unidade acima do maior número de pedido existente: sistema deve impedir a ação

vl_entregue

- valor inserido deve ser no formato tipo moeda

(Particionamento de equivalência)

- informar valor inferior ao valor da nota **vl_nota**: sistema deve impedir ação
- informar valor igual ao valor da nota **vl_nota**: sistema deve permitir ação
- informar valor superior ao valor da nota **vl_nota**: sistema deve permitir ação

dt_pagto

(Particionamento de equivalência)

Condição de entrada, booleano: data pode ou não ser inserida

- sistema deve inserir automaticamente a data corrente do sistema
- apagar a data e solicitar pagamento: sistema deve impedir a ação

Condição de entrada, intervalo: data superior ou igual à data do pedido **dt_pedido**

- pagar nota inserindo data inferior à data do pedido: sistema deve bloquear a ação
- pagar nota com data maior do que a data do pedido: ação deve ser realizada
- pagar nota com data igual à data do pedido: ação deve ser realizada

Fluxo de saída

troco

- diferença entre **vl_nota** e **vl_entregue**, exibido em formato tipo moeda
- troco deve ser sempre um valor positivo

ATOR: Garçom IC 5: Pendurar a nota

➔ **pendura = id_pedido + id_cliente**

Propósito: Identificar a nota (em aberto, de um cliente habitual) a ser pendurada.

Fluxo de entrada

pendura

(Particionamento de equivalência)

- somente pode ser solicitada pendura de pedidos em aberto, pedidos pagos, cancelados ou pendurados não participam do conjunto de pedidos que possam ser pendurados.

id_pedido

(Particionamento de equivalência)

Condição de entrada, booleano: pedido deve ser identificado

- não informar o número do pedido: sistema deve impedir ação
- informar o número do pedido: ação deve ser executada

Condição de entrada, intervalo: um dos pedidos em aberto e ainda não pago

- informar pedido que esteja cancelado: sistema deve impedir a ação
- informar pedido que esteja pendurado: sistema deve impedir ação
- informar pedido que esteja em aberto ainda não pago: ação deve ser realizada
- informar um número de pedido em formato diferente do número gerado automaticamente: sistema deve impedir ação

(Análise do valor limite)

- informar um número de pedido igual a zero: sistema deve impedir a ação
- informar um número de pedido uma unidade acima do maior número de pedido existente: sistema deve impedir a ação

id_cliente
(Particionamento de equivalência)
Condição de entrada, booleano: cliente deve ser informado
- realizar IC 3 informando um cliente válido, mesmo cliente informado no IC 3 deve ser retornado automaticamente pelo sistema: ação deverá ser realizada
- realizar IC 3 informando um cliente válido, apagar a identificação do cliente: ação deve ser impedida
- realizar IC 3 sem informar um cliente: ação deve ser impedida
- realizar IC 3 sem informar um cliente e inserir a identificação de um cliente válido: ação deve ser realizada.

Condição de entrada, intervalo: somente clientes cadastrados no sistema
- informar **id_cliente** em um formato diferente do gerado automaticamente: sistema deve impedir ação
- informar **id_cliente** superior ao maior id cadastrado: sistema deve impedir ação
- informar **id_cliente** cadastrado: sistema deve permitir ação
- realizar IC 3 passando um **id_cliente**, realizar IC 5 alterando o cliente para outro válido: sistema deve pendurar nota para o **id_cliente** passado neste IC 5.

(Análise do valor limite)
- informar número igual a zero: sistema deve impedir a ação
- informar número superior ao maior número de cadastro existente: sistema deve impedir a ação

ATOR: Gerente IC 6: Cadastrar cliente habitual

➔ cliente = nome_cliente + tel_cliente

Descrição: Informações sobre um cliente habitual do restaurante. Apenas clientes habituais têm direito a pendurar notas.

← id_cliente

Fluxo de entrada
nome_cliente
(Particionamento de equivalência)
Condição de entrada, intervalo
- campo textual sem restrições

tel_cliente
- campo tipo telefone, sem restrições explícitas pela especificação

Fluxo de saída
id_cliente
(Particionamento de equivalência)
Condição de saída, intervalo
- sistema gera um número correspondente ao cliente cadastrado, tal número deve seguir uma seqüência sempre crescente sem coincidência nem saltos.

ATOR: Gerente IC 7: Atualizar o cardápio

➔ item_consumo = nome_item + pc_unit + cat_item + descr_item

Descrição: Informações sobre um item de consumo (prato ou bebida) a ser incluído no cardápio do restaurante.

← id_item

Fluxo de entrada

nome_item

(Particionamento de equivalência)

Condição de entrada, intervalo

- campo textual sem restrições

pc_unit

(Particionamento de equivalência)

Condição de entrada, intervalo

- campo em formato tipo moeda
- exibido em reais com precisão de centavos: duas casas decimais.

cat_item

(Particionamento de equivalência)

Condição de entrada, intervalo

- apenas um dos elementos do conjunto {'prato', 'bebida'}: sistema não deve permitir valores diferentes.

descr_item

(Particionamento de equivalência)

Condição de entrada, intervalo

- campo textual sem restrições

Fluxo de saída

id_item

(Particionamento de equivalência)

Condição de saída, intervalo

- sistema gera um número correspondente ao item cadastrado, tal número deve seguir uma seqüência sempre crescente sem coincidência e sem saltos.

ATOR: Gerente IC 8: Solicitar consumo diário

➔ solic_consumo = dt_emissao + dt_consumo

Propósito: Selecionar a data para a qual se deseja o relatório de consumo.

← consumo_dia = dt_emissao + dt_consumo + consumo_itens

Descrição: Informações sobre o consumo na data escolhida. Se a data de emissão for igual à data do consumo, e a consulta for realizada antes do fechamento do dia de funcionamento do restaurante, o resultado da consulta poderá apresentar valores apenas parciais.
Propósito: Facilitar a reposição do estoque de insumos do restaurante.

consumo_itens = $_0${cat_item + {id_item + nome_item + quant_item}$_2$

Descrição: Informações sobre o consumo de itens, para cada categoria (pratos e bebidas).
Ordenação: Crescente de **cat_item**; decrescente de **quant_item**.

Fluxo de entrada

dt_emissão

- data vigente do sistema

dt_consumo

(Particionamento de equivalência)

Condição de entrada, booleano

- se não for inserido nenhum valor a data corrente deve ser automaticamente inserida

- apagar a data e solicitar o consumo: sistema deve impedir ação

Condição de entrada, intervalo

- informar uma data posterior à data atual: sistema deve impedir ação

- informar uma data igual à data atual: sistema deve executar ação

- informar uma data anterior à data atual: sistema deve executar ação

Fluxo de saída

dt_emissão

- data da emissão do relatório, é a data vigente.

dt_consumo

- data escolhida para emissão do relatório, esta data deve corresponder a todos os pedidos constantes no relatório.

cat_item

(Particionamento de equivalência)

Condição de saída, intervalo

- apenas um dos elementos do conjunto {'bebida', 'prato'}, sem repetição e em ordem alfabética de ocorrência

id_item

- identificação de um item

nome_item

- campo textual com nome do item conforme consta no cardápio relativo ao **id_item**

descr_item

- campo textual com descrição do item conforme consta no cardápio relativo ao **id_item**

quant_item

- valor numérico relativa à quantidade total pedida de um determinado item na data **dt_consumo, quant_item** deve ser o somatório de todos os **quant_item** dos pedidos em que **dt_pedido = dt_consumo**

consumo_itens

Considerar a formatação de saída especificada no IC8

- agrupado em ordem alfabética por cat_item: apenas duas ocorrências

- agrupado por item: tantas ocorrências quantas forem necessárias até completar todos os itens constantes nos pedidos em que **dt_pedido = dt_consumo,** ordenado por ordem decrescente de quantidade.

A elaboração do relatório sugere uma estrutura de repetição até que sejam exauridos os dados a exibir. Os casos de testes para relatórios podem ser construídos usando a estratégia de testes de laço, para verificação das consultas realizadas e da forma de exibição em tela, conforme abaixo:

- solicitar relatório para uma data em que nenhum pedido tenha sido feito: sistema deve gerar relatório mesmo que em branco.

- solicitar relatório para uma data em que apenas um pedido com um item de uma categoria tenha sido realizado: sistema deve gerar relatório respeitando ordenação e formatação especificada.

- solicitar relatório para uma data em que vários pedidos de uma mesma categoria de item tenham sido realizados: sistema deve respeitar agrupamento e ordenação
- solicitar relatório para uma data de modo que tenham sido realizados diversos pedidos das duas categorias existentes: sistema deve respeitar agrupamento e ordenação
- solicitar relatórios para a data atual em que pedidos estejam em andamento: sistema deve gerar relatório com base nos pedidos já realizados

ATOR: Gerente IC 9: Solicitar receita

➔ **solic_receita = periodo_apur + dt_emissao**

Propósito: Selecionar o período para o qual se deseja apurar a receita do restaurante.

periodo_apur = dt_inicio + dt_fim

Descrição: Data inicial e final (inclusive) do período de apuração da receita.

🡐 **receita = dt_emissao + periodo_apur + vl_consumo + receita_realiz + receita_pend + receita_txServ + receita_total**

Descrição: Informações sobre a receita oriunda dos pedidos abertos e não cancelados, dentro do período de apuração. Se a data de emissão for igual à data do final do período de apuração, e a consulta for realizada antes do fechamento do dia de funcionamento do restaurante, o resultado da consulta poderá apresentar valores apenas parciais.
Propósito: Facilitar o acompanhamento, pela gerência, dos resultados financeiros do restaurante (realizados e a realizar), e permitir a distribuição aos garçons da receita proveniente da taxa de serviço.

Fluxo de entrada

dt_inicio

(Particionamento de equivalência)

Condição de entrada, booleano

- não informar valor: sistema deve impedir ação
- informar valor: sistema deve executar ação

Condição de entrada, intervalo

- informar data posterior à data vigente: sistema deve impedir ação
- informar data igual à vigente: sistema deve executar ação
- informar data anterior à vigente: sistema deve executar ação
- informar data posterior à **dt_fim**: sistema deve impedir ação

dt_fim

(Particionamento de equivalência)

Condição de entrada, booleano

- não informar valor: sistema deve impedir ação
- informar valor: sistema deve executar ação

Condição de entrada, intervalo

- informar data posterior à data vigente: sistema deve impedir ação
- informar data igual à vigente: sistema deve executar ação
- informar data anterior à vigente: sistema deve executar ação
- informar data anterior à **dt_inicio**: sistema deve impedir ação
- informar data igual à **dt_inicio**: sistema deve executar ação
- informar data posterior à **dt_inicio**: sistema deve executar ação

dt_emissão

- campo inserido automaticamente com a data vigente do sistema

Fluxo de saída

receita

- relatório gerado sem levar em consideração os pedidos cancelados para nenhum de seus valores retornados ou calculados.

- solicitar receita para um período de apuração em que não houve nenhum pedido: relatório deve ser gerado mesmo que sem dados a exibir

- solicitar receita para um período de apuração correspondente a um único dia em que ouve apenas um pedido: sistema deve gerar relatório respeitando formatação e cujos dados correspondam ao pedido

- solicitar receita para um período de apuração correspondente a vários dias, de forma que: não exista nenhum pedido, exista apenas um pedido e existam vários pedidos, respectivamente: sistema deve gerar relatório respeitando a formatação especificada

dt_emissão

- data vigente do sistema.

periodo_apur

- traz as datas **dt_inicio e dt_fim** inseridas ao Solicitar receita

vl_consumo

- valor calculado a partir da soma de todos os **vl_item** das notas emitidas dentro do período de apuração **periodo_apur**, desde que a nota não tenha sido cancelada.

receita_realiz

- valor calculado pela soma de todos os **vl_item** das notas pagas dentro do período de apuração **periodo_apur.** Desconsidera os 10%

receita_pend

- valor calculado pela soma de todos os **vl_nota** das notas penduradas dentro do período de apuração **periodo_apur.** Considera os 10%

receita_txServ

- valor calculado pela soma de todos os **vl_nota** das notas pagas dentro do período d de apuração **periodo_apur.** Considera os 10%

receita_total

- valor calculado pela soma de todos os **vl_nota** das notas dentro do período de apuração **periodo_apur,** indiferente com relação a notas pagas ou penduradas e considerando os 10%

ATOR: Gerente IC 10: Cadastrar mesa

➔ mesa = nr_mesa

Descrição: Número de uma mesa do restaurante.
Propósito: Serve como identificador de uma mesa.

← id_mesa

Fluxo de entrada
nr_mesa
(Particionamento de equivalência)
Condição de saída, intervalo: subconjunto dos números naturais definido entre os limites [1,50]
- inserir número igual a zero: sistema deve impedir ação
- inserir número igual 51: sistema deve impedir ação
- inserir número fracionário: sistema deve impedir ação
- executar este IC duas vezes tentando inserir duas mesas com um mesmo número válido: comportamento do sistema não especificado

(Análise do valor limite)
- inserir número igual 1: sistema deve executar ação
- inserir número igual 50: sistema deve executar ação

Fluxo de saída
id_mesa
(Particionamento de equivalência)
Condição de saída, intervalo
- sistema gera um número correspondente à mesa cadastrada, tal número deve seguir uma seqüência sempre crescente sem coincidência nem saltos.

ATOR: Gerente IC 11: Solicitar relação de notas penduradas

➔ solic_penduras = periodo_apur + dt_emissao

Propósito: Selecionar o período para o qual se deseja apurar as notas penduradas.

periodo_apur = dt_inicio + dt_fim

Descrição: Período de apuração das notas penduradas.

← penduras = dt_emissao + periodo_apur + {id_cliente + nome_cliente + tel_cliente + notas_pend + vl_pendCli} + vl_pend

Descrição: Relação de notas penduradas no período de apuração. Se a data de emissão for igual à data do final do período de apuração, e a consulta for realizada antes do fechamento do dia de funcionamento do restaurante, o resultado da consulta poderá apresentar valores apenas parciais.
Propósito: 1) Identificar as notas penduradas de um cliente, quando o mesmo deseja quitá-las e não está com a cópia do ticket que lhe foi entregue na data do consumo; 2) Apoiar o gerente no processo de cobrança das notas penduradas.
Ordenação: Crescente de **nome_cliente**.

notas_pend = $_1${id_pedido + dt_pedido + nr_mesa + itens_nota + vl_nota}

Descrição: Relação das notas penduradas do cliente em questão (**id_cliente**), dentro do período de apuração escolhido.

Ordenação: Crescente de **dt_pedido**.

itens_nota = $_1${id_item + cat_item + nome_item + pç_item+ quant_item + vl_item}

Descrição: Informações sobre os itens (pratos e bebidas) consumidos, que compõem a nota em questão.
Ordenação: Crescente de **cat_item** ++ **nome_item**.

Fluxo de entrada
solic_penduras

-solicitação passível de ser feita somente pelo ator 'Gerente'

dt_inicio
(Particionamento de equivalência)
Condição de entrada, booleano

- não informar valor: sistema deve impedir ação
- informar valor: sistema deve executar ação

Condição de entrada, intervalo

- informar data posterior à data vigente: sistema deve impedir ação
- informar data igual à vigente: sistema deve executar ação
- informar data anterior à vigente: sistema deve executar ação
- informar data posterior à **dt_fim**: sistema deve impedir ação

dt_fim
(Particionamento de equivalência)
Condição de entrada, booleano

- não informar valor: sistema deve impedir ação
- informar valor: sistema deve executar ação

Condição de entrada, intervalo

- informar data posterior à data vigente: sistema deve impedir ação
- informar data igual à vigente: sistema deve executar ação
- informar data anterior à vigente: sistema deve executar ação
- informar data anterior à **dt_inicio**: sistema deve impedir ação
- informar data igual à **dt_inicio**: sistema deve executar ação
- informar data posterior à **dt_inicio**: sistema deve executar ação

dt_emissão

- campo inserido automaticamente com a data vigente do sistema

Fluxo de saída
penduras

- relatório gerado considerando o intervalo definido no período de apuração
- relatório deve agrupar os resultados por cliente em ordem alfabética
- relatório deve agrupar as notas penduradas de um cliente exibindo em ordem crescente de data
- relatório deve exibir os itens constantes em uma nota agrupando pela categoria do item e em seguida pelo nome do item, ambos em ordem alfabética
- solicitar relatório para um período em que não houve nenhum pedido pendurado: sistema deve gerar relatório respeitando formatação
- solicitar relatório em um período em que houve apenas um pedido pendurado: sistema deve gerar relatório respeitando formatação
- solicitar relatório para um período em que houve vários pedidos pendurados: sistema deve gerar relatório respeitando formatação
- solicitar relatório para o dia corrente: sistema deve gerar relatório respeitando formatação

dt_emissão

- data vigente do sistema.

periodo_apur
- traz as datas **dt_inicio e dt_fim** inseridas ao Solicitar relação de notas penduradas
id_cliente
- identificação de um cliente que possua notas penduradas
nome_cliente
- nome do cliente identificado por **id_cliente**
tel_cliente
- telefone do cliente identificado por **id_cliente**
id_pedido
- identificação de um pedido em aberto e pendurado
dt_pedido
- data em que foi realizado o pedido identificado por **id_pedido**
nr_mesa
- número da mesa relativa ao pedido identificado por **id_pedido**
itens nota
id_item
- identificação de um item constante no pedido identificado por **id_pedido**
cat_item
- categoria dos itens registrados no pedido identificado por **id_pedido**, deve ser limitada às categorias existentes no sistema: {'bebida', 'prato'}
nome_item
- nome do item identificado por **id_item**
pç_item
- preço do item identificado por **id_item**, no momento em que foi realizado o pedido. Preço não deve sofrer alterações mesmo que o cardápio seja atualizado em data posterior à pendura do pedido.
quant_item
-quantidade solicitada do item identificado por **id_item** em relação à nota pendurada identificada por **id_pedido**
vl_item
- valor calculado pelo produto entre **pc_unit** e **quant-item** no momento de realizar o pedido
vl_nota
- valor total da nota relativo ao pedido identificado por **id_pedido**
vl_pendCli
- valor calculado a partir da soma de todos os **vl_nota** para cada nota pendurada em relação a um mesmo cliente identificado por **id_cliente**
vl_pend
- valor calculado pela soma de todas as notas penduradas sem distinção entre clientes dentro do período de apuração definido entre as datas **dt_inicio** e **dt_fim**

Apêndice C

Especificação técnica sistema fábrica de software

Sumário

LISTA DE REQUISITOS

Uma empresa de TI deseja um sistema para dar suporte ao seu processo de desenvolvimento. As seguintes funções devem ser desempenhadas pelo sistema:

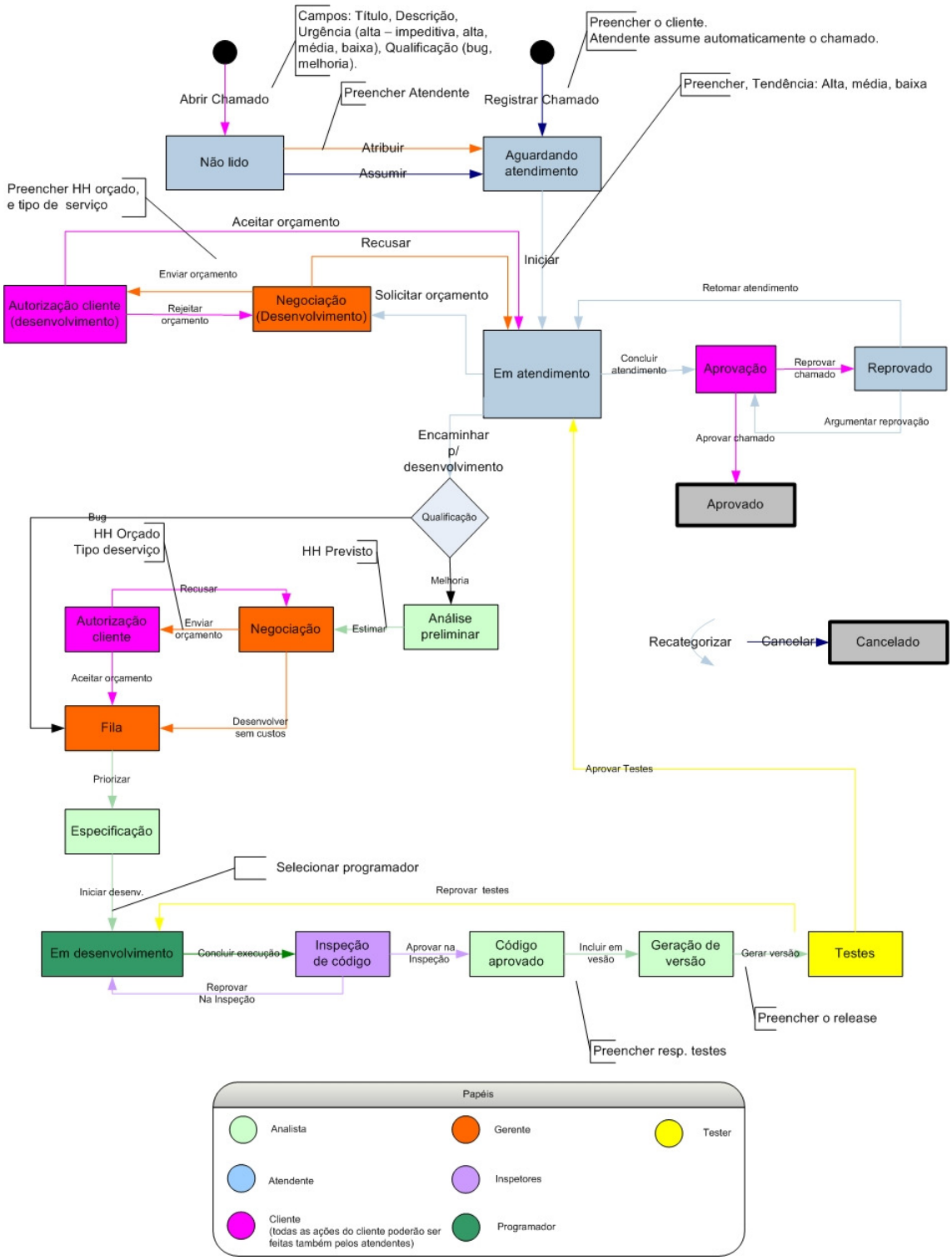
1. Registrar uma solicitação (chamado) de um cliente ou de uma pessoa interna à própria empresa
   1.1. Quando o chamado é aberto por um cliente, ele deve ser marcado como 'Não lido' até que um atendente possa realizar o tratamento adequado.
      1.1.1. No momento da abertura o cliente informa a urgência ("alta-impeditivo", "alta", "média" ou "baixa") e também a qualificação ("bug" ou "melhoria") para o chamado.
   1.2. Quando for aberto por um atendente, automaticamente será tratado como em atendimento pelo próprio atendente que fez o registro.
      1.2.1. O atendente terá a opção de informar o cliente, urgência e qualificação para o chamado.
   1.3. Apenas clientes e atendentes podem Abrir Chamados.
   1.4. Para chamados abertos por clientes:
      1.4.1. Permanecerão como não lidos até que sejam assumidos por um atendente
      1.4.2. Gerência poderá atribuir o chamado a um atendente.
2. Toda ação que o cliente puder realizar sobre um chamado, o atendente também poderá realizar
3. Um chamado poderá ser cancelado a qualquer momento, seja pelo cliente, atendente ou gerente.
4. Uma vez iniciado o atendimento, o atendente será responsável por encaminhar o chamado, conforme conveniência, para as seguintes ações:
   4.1. Orçamento de desenvolvimento
   4.2. Enviar para desenvolvimento
   4.3. Concluir Atendimento
5. O orçamento será realizado pela gerencia.
   5.1. O gerente poderá também se recusar a realizar um orçamento, caso julgue não ter informações suficientes para sua realização
6. Após realização do orçamento o cliente poderá validar ou não
   6.1. Caso seja feita a validação, o chamado volta para a responsabilidade do atendente que o encaminhará novamente, conforme conveniência
   6.2. Caso não seja validado o chamado retorna para novo orçamento por parte da gerencia.
7. Ao enviar para desenvolvimento deve ser avaliado se o chamado é um bug ou uma melhoria, conforme indicação do cliente quando da abertura do chamado.
   7.1. Bug: o chamado será encaminhado automaticamente para a fila para que seja priorizado pelo analista.
   7.2. Melhoria: o chamado passará por uma análise preliminar para então decidir se será necessária uma negociação seguida de autorização do cliente ou se pode ser encaminhado para a fila.
   7.3. Atendente pode requalificar um chamado a qualquer momento, alterando entre "bug" ou "melhoria"

8. Após priorização, o chamado será encaminhado para a especificação, momento em que um analista irá especificar o que deve ser realizado. Sua responsabilidade é disponibilizar uma especificação técnica para a equipe interna e uma especificação funcional para o cliente. Feito isto poderá ser iniciado o desenvolvimento com criação de tarefas, indicação dos desenvolvedores e custo esperado em horas para atendimento do chamado.
9. Concluída a fase de desenvolvimento será iniciada a fase de análise de código por parte do inspetor de código.
    9.1. Caso ocorra reprovação o mesmo desenvolvedor que realizou a implementação deve ser responsabilizado pela correção
    9.2. Caso seja aprovado o código, o chamado será disponibilizado para geração de versão por parte do analista
10. Com o código aprovado, uma versão será gerada pelo analista para ser enviada para testes
    10.1. Ao enviar para testes, um tester deve ser designado
11. Na etapa de testes duas possibilidades são possíveis
    11.1. Aprovação em testes: chamado retornará para responsabilidade do atendente para que tome as ações que forem convenientes, o status do chamado deverá ser 'Em atendimento'
    11.2. Reprovação em testes: chamado retorna para desenvolvimento
12. Ao Concluir Atendimento, o chamado ficará a disposição do cliente para que o aprove ou reprove
    12.1. Aprovação: Chamado será finalizado
    12.2. Reprovação: chamado poderá ser reiniciado ou atendente poderá argumentar a reprovação
        12.2.1. Ao argumentar, o cliente poderá novamente aprovar/reprovar
        12.2.2. Ao Reiniciar o chamado poderá ser novamente enviado para orçamento ou para desenvolvimento.

<u>Regras gerais</u>

13. Cliente pode cancelar a qualquer momento um chamado
14. Tarefas podem ser criadas e associadas aos chamados a qualquer momento
    14.1. Gerente pode criar tarefas para qualquer pessoa e cancelar ou concluir elas
    14.2. Cliente jamais terá tarefa sob sua responsabilidade
    14.3. Atendente e analista podem criar tarefas para os desenvolvedores, inspetores e testers. Podem também cancelar e concluir todas as tarefas do chamado pra o qual são responsáveis, exceto as tarefas do gerente.
    14.4. Quaisquer usuários, exceto o cliente, podem criar tarefas para si próprios.
    14.5. Responsável pela tarefa pode cancelar ou concluir a própria tarefa

Diagrama de estados e transições



INFO CASES

ATOR: Cliente **IC 1: Abrir chamado**

➔ **chamado = dt_chamado + titulo_chamado + descricao + urgencia + qualificacao_chamado**

Descrição: Informação de uma solicitação de serviço por parte de um cliente. A data será automaticamente inserida pelo sistema armazenando, inclusive, informação de horas.
Propósito: Informar ao sistema a abertura de um chamado marcando seu status como "Não lido"

➔ **id_chamado**

ATOR: Atendente **IC 2: Registrar chamado**

➔ **chamado = dt_chamado + titulo_chamado +descricao + urgencia + qualificacao_chamado + id_pessoa**

Descrição: registro de chamado realizado diretamente por um atendente, o cliente deve ser informado podendo ser qualquer pessoa cadastrada no sistema.
O atendente para este chamado será automaticamente inserido com o nome de quem fez o registro, sem possibilidade de alteração.
Propósito: Informar ao sistema a abertura de um chamado marcando seu status como "Aguardando atendimento"

➔ **id_chamado**

ATOR: Cliente/Atendente/Gerente **IC 3: Cancelar chamado**

➔ **cancelar_chamado = id_chamado + justificativa**

Descrição: IC pode ser realizado a qualquer momento tanto pelo cliente quanto pelo atendente responsável.
Propósito: realizar cancelamento de um chamado, independente do status dele e alterando seu status para "Cancelado".

ATOR: Gerente **IC 4: Atribuir**

➔ **atribuir_chamado = id_chamado + id_pessoa + comentario**

Propósito: Atribuir um chamado à responsabilidade de um atendente específico.
Chamado terá seu status alterado para "Aguardando atendimento"

ATOR: Atendente	**IC 5:**	**Assumir**

➔ **assumir_chamado = id_chamado + comentario**

Propósito: Um atendente assume um chamado indicando que será ele o responsável pelo seu atendimento. O status do chamado será atualizado para "Aguardando atendimento"

ATOR: Atendente	**IC 6:**	**Iniciar**

➔ **iniciar = id_chamado + comentario**

Propósito: Indicar que o chamado esta sendo atendido. O status do chamado será atualizado para "Em atendimento"

ATOR: Atendente	**IC 7:**	**Solicitar orçamento**

➔ **solicitar_orc = id_chamado + comentario**

Descrição: Envia chamado para responsabilidade da gerencia de desenvolvimento para elaboração de orçamento.
Propósito: O status do chamado será atualizado para "Negociação desenvolvimento"

ATOR: Gerente	**IC 8:**	**Não orçar**

➔ **nao_orc = id_chamado + justificativa**

Descrição: Gerente pode se recusar a realizar um orçamento quando considerar que as informações são insuficientes.
Propósito: O status do chamado será atualizado para "Em atendimento"

ATOR: Gerente	**IC 9:**	**Enviar Orçamento**

➔ **enviar_orc = id_chamado + tipo_servico + HH_orcado + justificativa**

Descrição: Gerente disponibiliza o orçamento informando o tipo de serviço a ser realizado.

Propósito: Caso o status do chamado seja "Negociação desenvolvimento" então ele será atualizado para "Autorização cliente (Negociação desenvolvimento)".
Se o status for "Negociação" então ele será atualizado para "Autorização cliente"

ATOR: Cliente **IC 10: Decidir orçamento**

➔ **decidir_orc = id_chamado + decisao_orcamento + justificativa**

Descrição: Cliente pode recusar um orçamento elaborado, informando a razão da recusa, para que o Gerente possa prosseguir com a negociação.
Propósito: chamado terá seu status definido de acordo com o valor para decisao_orcamento status no momento em que o cliente confirmar a ação
Caso decisão seja "Recusar orçamento"
Se status_chamado = "Autorização cliente (Negociação desenvolvimento)", então o novo status será "Negociação desenvolvimento"
Se status_chamado = "Autorização cliente (Negociação)", então o novo status será "Negociação"
Caso decisão seja "Aceitar orçamento"
Se status_chamado = "Autorização cliente (Negociação desenvolvimento)", então o novo status será "Em atendimento"
Se status_chamado = "Autorização cliente (Negociação)", então o novo status será "Fila"

ATOR: Atendente **IC 11: Enviar para desenvolvimento**

➔ **enviar_desenv = id_chamado + comentario**

Descrição: O atendente envia um chamado para desenvolvimento.
Propósito: Dependendo da qualificação do chamado seu status será diferente. Caso seja "Bug" status = "fila", se for "Melhoria" status = "Análise preliminar"

ATOR: Analista **IC 12: Estimar**

➔ **estimar = id_chamado + tipo_servico + HH_estimado + comentario**

Descrição: o analista ao verificar o chamado pode identificar a necessidade de serviços não orçados anteriormente. Neste momento ele envia o chamado para a Gerente informando o tipo de serviço que será realizado e quanto de horas estimou ser necessário para sua conclusão.
Propósito: O status do chamado será atualizado para "Negociação"

ATOR: Gerente **IC 13: Desenvolver sem custos**

➔ **desenv_scusto = id_chamado + comentario**

Descrição: durante a fase de negociação, a Gerente pode decidir por enfileirar o chamado sem considerar custos para o cliente.
Propósito: O status do chamado será atualizado para "Fila"

ATOR: Gerente **IC 14: Priorizar**

➔ **priorizar = id_chamado + comentario**

Descrição: Ao priorizar o chamado é passado para a responsabilidade do Analista da equipe responsável definida anteriormente, neste momento o analista definirá a especificação técnica necessária para que o trabalho seja desenvolvido.
Propósito: O status do chamado será atualizado para "Especificação"

ATOR: Analista **IC 15: Iniciar desenvolvimento**

➔ **iniciar_desenv = id_chamado + id_pessoa + comentario**

Propósito: permitir ao analista que envie o chamado para desenvolvimento, o status do chamado será atualizado para "Em desenvolvimento"

ATOR: Desenvolvedor **IC 16: Concluir desenvolvimento**

➔ **concluir_desenv = id_chamado + comentario**

Descrição: somente poderá concluir um chamado o desenvolvedor definido para o ele.
Propósito: o status do chamado será atualizado para "Inspeção de código"

ATOR: Inspetores **IC 17: Concluir inspeção**

➔ **concluir_inspecao = id_chamado + result_inspec + justificativa**

Descrição: caso sejam encontrados trechos de código que contenham erros ou que possam ser melhorados, o inspetor retorna o chamado para desenvolvimento para que os acertos necessários sejam realizados.
Propósito: o status do chamado será atualizado de acordo com a opção escolhida para **result_inspec**.
Se for "Aprovado" então o status será atualizado para "Geração de versão"

Se for “Reprovado” então o status será atualizado para “Em desenvolvimento”

ATOR: Analista	**IC 18: Gerar versão**

➔ **gerar_versao = id_chamado + comentario + id_pessoa + nr_release**

Descrição: o analista disponibiliza o chamado juntamente com todos os arquivos e procedimentos necessários à entrega para que um tester definido por ele possa realizar a verificação, tendo como base a especificação técnica elabora anteriormente.
Deve ser passada a identificação de uma pessoa pelo **id_pessoa**, desde que seu tipo seja “Tester”.
Propósito: o status do chamado será atualizado para “Testes”

ATOR: Tester	**IC 19: Concluir testes**

➔ **concluir_testes = id_chamado + result_testes + justificativa**

Descrição: se julgar necessário o tester reprova o chamado indicando a razão da reprovação.
O chamado retornará para a o desenvolvedor definido como responsável pelo chamado.
Propósito: o status do chamado será atualizado de acordo com a opção escolhida para **result_testes**.
Se for “Aprovado” então o status será atualizado para “Em atendimento”
Se for “Reprovado” então o status será atualizado para “Em desenvolvimento”

ATOR: Atendente	**IC 20: Concluir atendimento**

➔ **concluir_atend = id_chamado + justificativa**

Descrição: ao concluir um chamado o atendente o envia para análise do cliente. Neste momento ocorre a entrega do chamado com todos os arquivos e passos aprovados pelo Tester, caso o chamado tenha passado por desenvolvimento.
Propósito: o status do chamado será atualizado para “Aprovação”

ATOR: Atendente / Cliente **IC 21: Avaliar atendimento**

➔ **avaliar_atendimento = id_chamado + avaliacao_atendimento + justificativa**

Descrição: cliente pode aprovar ou reprovar um chamado informando a razão para que o atendente avalie a situação.
Propósito: o status do chamado será atualizado de acordo com a opção escolhida para **avaliacao_atendimento**.
Se for "Aprovado" então o status será atualizado para "Aprovado"
Se for "Reprovado" então o status será atualizado para "Reprovado"

ATOR: Atendente **IC 22: Avaliar reprovação**

➔ **avaliar_reprovacao = id_chamado + decisao + justificativa**

Descrição: atendente decide entre argumentar a reprovação do cliente enviando o chamado novamente para "Aprovação" ou se vai retomar o atendimento enviando o chamado para "Em atendimento"
Propósito: o status do chamado será atualizado de acordo com a opção escolhida para **decisao**.
Se for "Argumentar reprovação" então o status será atualizado para "Aprovação"
Se for "Retomar atendimento" então o status será atualizado para "Em atendimento"

ATOR: Todos os atores exceto Cliente **IC 23: Criar tarefa**

➔ **criar_t = id_chamado + id_pessoa + hh_tarefa**

Descrição: criar tarefa definindo quem será a pessoa responsável por realizá-la, quanto de tempo é necessário e o tipo de ação esperada.
A pessoa pode ser qualquer um dos atores existentes no sistema, exceto o cliente.
Gerente pode criar tarefa para todos
Analista e Atendente podem criar tarefas para todos nos chamados em que são responsáveis, exceto para o Gerente
Todos podem criar tarefas para si

➔ **id_tarefa**

ATOR: Todos os atores exceto Cliente **IC 24: Concluir tarefa**

➔ **concluir_t = id_tarefa + hh_gasto_tarefa + comentario**

Descrição: Tanto a pessoa que foi definida para execução da tarefa quanto o Gerente, o Atendente e o Analista do chamado em que são responsáveis podem cancelar tarefas.

ATOR: Todos os atores exceto Cliente **IC 25: Cancelar tarefa**

➔ **cancelar_t = id_ tarefa + justificativa**

Descrição: as tarefas podem ser canceladas pela pessoa responsável por elas
Gerente pode cancelar a tarefa de qualquer ator
Analista e atendente de um chamado podem cancelar tarefas do chamado em questão, exceto se a tarefa for de um Gerente

ATOR: Gerente **IC 26: Cadastrar pessoa**

➔ **cadastrar_pessoa = nome + tipo_pessoa**

Descrição: Gerente poderá cadastrar pessoas para que atuem nos chamados de acordo com o tipo definido.

➔ **id_pessoa**

ATOR: Atendente **IC 27: Requalificar chamado**

➔ **chamado = id_chamado + urgencia + qualificacao_chamado**

Descrição: Em caso de necessidade o atendente poderá requalificar chamado a qualquer momento.
Propósito: Permite realizar a requalificação do chamado a qualquer momento por parte do atendente.

DICIONÁRIO DE ITENS

IC 1: Abrir chamado

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	dt_chamado	Data em que foi aberto o chamado, traz como default a data corrente no sistema sem permitir alteração	Data	Igual à data corrente		
E	titulo_chamado	Campo para inserção de texto relativo ao objetivo do chamado	Texto	Limite de 100 caracteres		
E	descricao	Campo para descrição detalhada do que motivou abertura do chamado	Texto			
E	urgencia	A urgência deve ser uma dentre as opções disponíveis	Texto	{“Alta - Impeditivo”, “Alta”, “Media”, “Baixa”}		
E	qualificacao_chamado	Qualifica o chamado com relação à sua necessidade, se foi aberto por motivo de erro no sistema ou se é uma expansão/melhoria a ser encomendada	Texto	{“Correção”, “Melhoria”}		
S	id_chamado	Identificador de um chamado	Nr. Automático[4]	Numeração inserida automaticamente pelo sistema		

IC 2: Registrar chamado

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	dt_chamado	Data em que foi aberto o chamado, traz como default a data corrente no sistema sem permitir alteração	Data	Igual à data corrente		
E	titulo_chamado	Campo para inserção de texto relativo ao objetivo do chamado	Texto	Limite de 100 caracteres		
E	descricao	Campo para descrição detalhada do que motivou abertura do chamado	Texto			
E	urgencia	A urgência deve ser uma dentre as opções disponíveis	Texto	{“Alta - Impeditivo”, “Alta”, “Media”, “Baixa”}		
E	qualificacao_chamado	Qualifica o chamado com relação à sua necessidade, se foi aberto por motivo de erro no sistema ou se é uma expansão/melhoria a ser encomendada	Texto	{“Correção”, “Melhoria”}		
E	id_pessoa	Identificador de uma pessoa para ser o cliente do chamado	Nr. Natural			
S	id_chamado	Identificador de um chamado	Nr. Automático	Numeração automaticamente inserida pelo sistema		

[4] Número gerado automaticamente pelo sistema, para ser o identificador de um chamado.

IC 3: Cancelar chamado

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	justificativa	Campo de preenchimento obrigatório para inserção de texto com uma justificativa para o cancelamento.	Texto			

IC 4: Atribuir

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	id_pessoa	Identificador de uma pessoa cujo tipo seja atendente	Nr. Natural	Maior do que zero		
E	comentario	Campo de preenchimento opcional para que informações explicativas possam ser inserias pelo usuário	Texto			

IC 5: Assumir

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	comentario	Campo de preenchimento opcional para que informações explicativas possam ser inserias pelo usuário	Texto			

IC 6: Iniciar

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	comentario	Campo de preenchimento opcional para que informações explicativas possam ser inserias pelo usuário	Texto			

IC 7: Solicitar orçamento

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	comentario	Campo de preenchimento opcional para que informações explicativas possam ser inserias pelo usuário	Texto			

IC 8: Não orçar

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	justificativa	Campo de preenchimento obrigatório para inserção de texto com uma justificativa para a ação realizada.	Texto			

IC 9: Enviar orçamento

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	tipo_servico	Tipo de serviço que será executado para atendimento do chamado	Texto	{“Desenv. web”, “Desenv. Java”, “Desenv. C#”, “Banco de dados”}		
E	hh_orcado	Valor relativo ao HH necessário à realização do serviço necessário	Moeda		R$	Centavos
E	justificativa	Campo de preenchimento obrigatório para inserção de texto com uma justificativa para a ação realizada.	Texto			

IC 10: Decidir orçamento

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	decisao_orcamento	Opção para realização de ação que decidirá o status do chamado	Texto	{“Aceitar orçamento”, “Recusar orçamento”}		
E	justificativa	Campo de preenchimento obrigatório para inserção de texto com uma justificativa para a ação realizada.	Texto			

IC 11: Enviar para desenvolvimento

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	comentario	Campo de preenchimento opcional para que informações explicativas possam ser inserias pelo usuário	Texto			

IC 12: Estimar

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	tipo_servico	Tipo de serviço que será executado para atendimento do chamado	Texto	{“Desenv. web”, “Desenv. Java”, “Desenv. C#”, “Banco de dados”}		
E	HH_estimado	Quantidade de horas necessárias para realizar o desenvolvimento necessário ao CDS	Nr. Real	Maior do que zero		1
E	comentario	Campo de preenchimento opcional para que informações explicativas possam ser inserias pelo usuário	Texto			

IC 13: Desenvolver sem custos

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	comentario	Campo de preenchimento opcional para que informações explicativas possam ser inserias pelo usuário	Texto			

IC 14: Priorizar

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	comentario	Campo de preenchimento opcional para que informações explicativas possam ser inserias pelo usuário	Texto			

IC 15: Iniciar desenvolvimento

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	id_pessoa	id_pessoa cujo tipo seja desenvolvedor para assumir papel de desenvolvedor responsável pelo chamado	Nr. Natural	Maior do que zero		
E	comentario	Campo de preenchimento opcional para que informações explicativas possam ser inserias pelo usuário	Texto			

IC 16: Concluir desenvolvimento

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	comentario	Campo de preenchimento opcional para que informações explicativas possam ser inserias pelo usuário	Texto			

IC 17: Concluir inspeção

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	result_inspec	Opção para realização de ação que decidirá o status do chamado	Texto	{“Aprovado em inspeção”, “Reprovado em inspeção”}		
E	justificativa	Campo de preenchimento obrigatório para inserção de texto com uma justificativa para a ação realizada.	Texto			

IC 18: Gerar versão

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	comentario	Campo de preenchimento opcional para que informações explicativas possam ser inserias pelo usuário	Texto			
E	id_pessoa	id_pessoa cujo tipo seja tester, será a pessoa a testar o chamado	Nr. Natural	Maior do que zero		
E	nr_release	Número identificar da versão para o pacote de arquivos necessários à entrega do chamado	Nr. Natural	Maior do que zero		

IC 19: Concluir testes

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	result_testes	Opção para realização de ação que decidirá o status do chamado	Texto	{“Aprovado em testes”, “Reprovado em testes”}		
E	justificativa	Campo de preenchimento obrigatório para inserção de texto com uma justificativa para a ação realizada.	Texto			

IC 20: Concluir atendimento

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	justificativa	Campo de preenchimento obrigatório para inserção de texto com uma justificativa para a ação realizada.	Texto			

IC 21: Avaliar atendimento

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	avaliacao_atendimento	Opção para realização de ação que decidirá o status do chamado	Texto	{“Aprovado”, “Reprovado”}		
E	Justificativa	Campo de preenchimento obrigatório para inserção de texto com uma justificativa para a ação realizada.	Texto			

IC 22: Avaliar reprovação

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	decisao	Opção para realização de ação que decidirá o status do chamado	Texto	{“Argumentar reprovação”, “Retomar atendimento”}		
E	justificativa	Campo de preenchimento obrigatório para inserção de texto com uma justificativa para a ação realizada.	Texto			

IC 23: Criar tarefa

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	id_pessoa	Identificador da pessoa que será a responsável por executar a tarefa criada. Não pode ser relativa a uma pessoa cujo tipo seja cliente	Nr. Natural	Maior do que zero		
E	hh_tarefa	Quantidade de horas estimadas para a conclusão da tarefa	Nr. Real	Maior do que zero		1

IC 24: Concluir tarefa

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	id_pessoa	Identificador da pessoa que será a responsável por executar a tarefa criada. Não pode ser relativa a uma pessoa cujo tipo seja cliente	Nr. Natural	Maior do que zero		
E	hh_gasto_tarefa	Quantidade de horas investidas na tarefa até sua conclusão	Nr. Real	Maior do que zero		1
E	comentario	Campo de preenchimento opcional para que informações explicativas possam ser inserias pelo usuário	Texto			

IC 25: Cancelar tarefa

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_tarefa	Identificador de uma tarefa	Nr. Natural	Maior do que zero		
E	justificativa	Campo de preenchimento obrigatório para inserção de texto com uma justificativa para a ação realizada.	Texto			

IC 26: Cadastrar pessoa

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	Nome	Nome da pessoa a ser incluída no sistema	Texto			
E	tipo_pessoa	Tipo corresponde ao papel que será executado pela pessoa	Texto	{“Analista”, “Atendente”, “Cliente”, “Desenvolvedor”,		

IC 26: Cadastrar pessoa

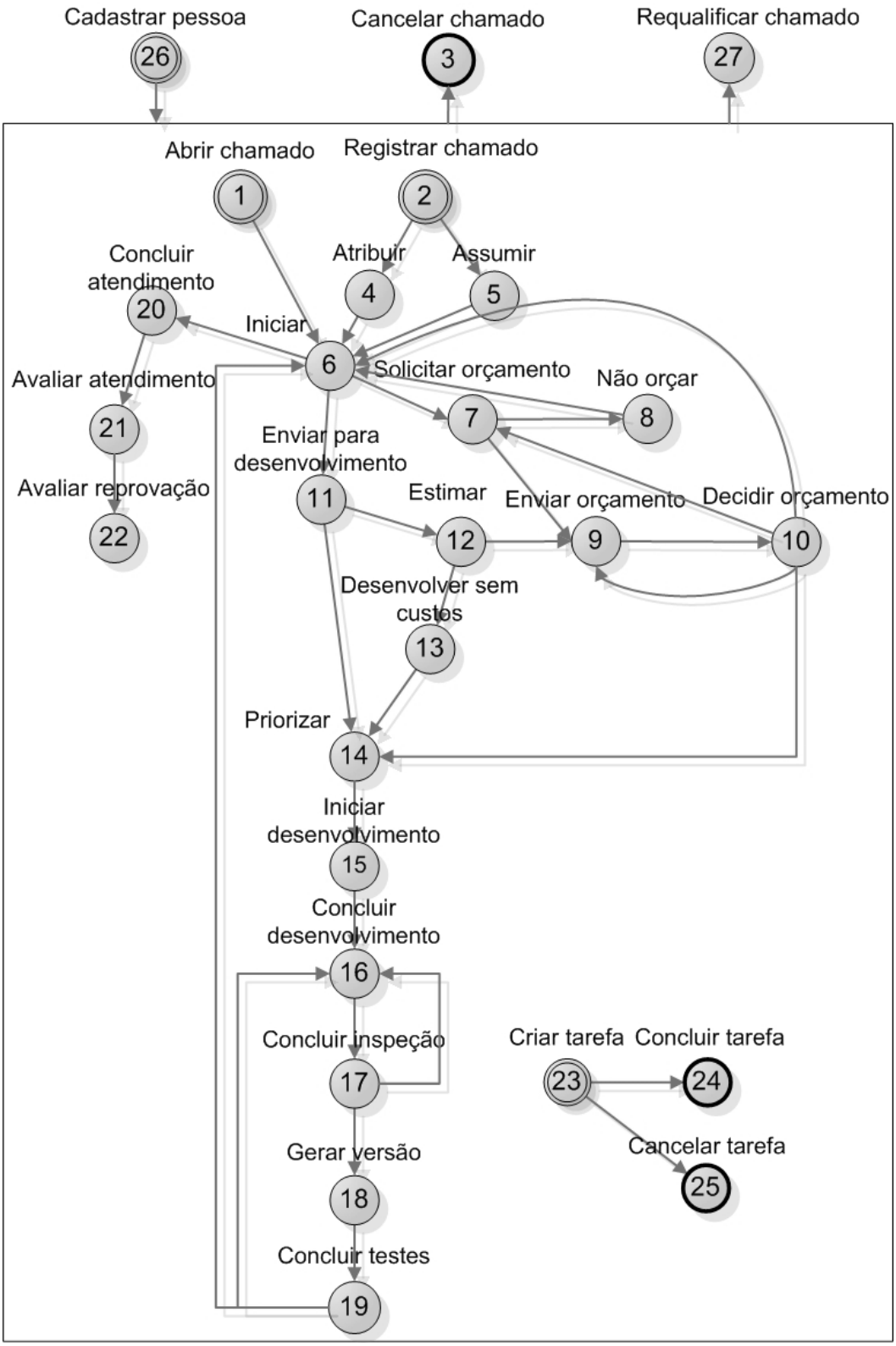
				“Gerente”, “Tester”}
S	id_pessoa	Identificador de uma pessoa	Nr. Automático[5]	Numeração inserida automaticamente pelo sistema

IC 27: Requalificar chamado

E/S	Nome	Descrição	Tipo Base	Domínio	Unidade	Precisão
E	id_chamado	Identificador de um chamado	Nr. Natural	Maior do que zero		
E	urgencia	A urgência deve ser uma dentre as opções disponíveis	Texto	{“Alta - Impeditivo”, “Alta”, “Media”, “Baixa”}		
E	qualificacao_chamado	Qualifica o chamado com relação à sua necessidade, se foi aberto por motivo de erro no sistema ou se é uma expansão/melhoria a ser encomendada	Texto	{“Correção”, “Melhoria”}		

[5] Número gerado automaticamente pelo sistema, para ser o identificador de uma pessoa.

GRAFO DE ADMISSIBILIDADE



Apêndice D

Casos de Testes Sistema Fábrica de Software

Casos de testes para ciclo completo no sistema Fábrica de software

Atores para os testes
Analista, Atendente, Cliente, Desenvolvedor, Gerente, Inspetor, Tester.

Cadastrar ator

Descrição: proceder com o cadastro de pessoas definindo seu papel para utilização do sistema
Mapa do teste:

Passo	Entrar no sistema com login default de administrador e acessar a parte de cadastro de pessoas. Proceder com a criação de um usuário da seguinte forma: Inserir um nome Escolher um tipo dentre as opções disponíveis
Verificação	Nome deve aceitar letras e números Tipo disponibilizado pelo sistema deve ser um dos seguintes: "Analista", "Atendente", "Cliente", "Desenvolvedor", "Gerente", "Inspetor" ou "Tester"
Passo	Efetuar o cadastro de pelo menos duas pessoas de cada tipo existente, para que os casos de testes seguintes possam ser executados.

Abrir chamado

Descrição: Criação de chamado por parte de um cliente
Mapa do teste

Passo	Logar no sistema com uma pessoa cujo tipo seja cliente e solicitar a abertura de um chamado
Verificação	Tela de abertura de chamado deve ser apresentada contendo os seguintes campos: Data, Título, Descrição, Urgência e Qualificação Campo data deve vir automaticamente preenchido com data atual
Passo	Confirmar a criação do chamado
Verificação	Sistema deve solicitar preenchimento do campo título
Passo	Inserir um texto qualquer no campo título e confirmar
Verificação	Caracteres especiais devem ser respeitados Sistema deve solicitar preenchimento do campo Descrição
Passo	Inserir um texto qualquer no campo Descrição e confirmar
Verificação	Caracteres especiais devem ser respeitados Sistema deve solicitar preenchimento do campo Urgência
Passo	Escolher uma urgência qualquer dentre as disponíveis e confirmar
Verificação	Sistema deve disponibilizar as seguintes opções para urgência: "Alta – impeditiva", "Alta", "Média" e "Baixa" Sistema deve solicitar preenchimento do campo Qualificação
Passo	Escolher uma dentre as opções disponíveis para a Qualificação
Verificação	Sistema deve disponibilizar as seguintes opções: "Bug" e "Melhoria"
Passo	Confirmar criação do chamado
Verificação	Chamado deve ser aberto de forma que o cliente dele seja a pessoa que realizou a ação "Abrir chamado" Status do chamado deve ser "Não lido"

Registrar chamado

Descrição: Criação de chamado por parte de um Atendente

Mapa do teste

Passo	Logar no sistema com uma pessoa cujo tipo seja Atendente e realizar a ação "Registrar chamado"
Verificação	Tela de abertura de chamado deve ser apresentada contendo os seguintes campos: Data, Título, Descrição, Urgência, Qualificação, Cliente e Atendente Campo Data deve vir automaticamente preenchido com data atual Campo Atendente deve vir preenchido com nome de quem registra o chamado
Passo	Confirmar a criação do chamado
Verificação	Sistema deve solicitar preenchimento do campo título
Passo	Inserir um texto qualquer no campo título e confirmar
Verificação	Caracteres especiais devem ser respeitados Sistema deve solicitar preenchimento do campo Descrição
Passo	Inserir um texto qualquer no campo Descrição e confirmar
Verificação	Caracteres especiais devem ser respeitados Sistema deve solicitar preenchimento do campo Urgência
Passo	Escolher uma urgência qualquer dentre as disponíveis e confirmar
Verificação	Sistema deve disponibilizar as seguintes opções para urgência: "Alta – impeditiva", "Alta", "Média" e "Baixa" Sistema deve solicitar preenchimento do campo Qualificação
Passo	Escolher um dentre as opções disponíveis para a Qualificação e confirmar
Verificação	Sistema deve disponibilizar as seguintes opções: "Bug" e "Melhoria" Sistema deve solicitar o preenchimento do campo Cliente
Passo	Preencher o campo Cliente com a identificação de uma pessoa qualquer já cadastrada no sistema
Verificação	Qualquer pessoa pode ser escolhida para ser cliente de um chamado aberto por um atendente. Pessoa que abre o chamado será o atendente deste.
Passo	Confirmar registro do chamado
Verificação	Chamado deve ser registrado e seu status será: "Aguarda atendimento"

Cancelar chamado

Descrição: Cancelamento de chamados, independente de seu status original
Este caso de testes deve poder ser executado por pessoas do tipo Cliente, Atendente ou Gerente.
Mapa do teste

Passo	Verificar no sistema se ação Cancelar esta disponível para todos os status que um chamado possa assumir
Passo	Este caso de testes deve ser possível de ser executado pelos atores Atendente, Cliente e Gerente. Repetir o procedimento abaixo para cada ator
Passo	Logar no sistema com um dos atores com permissão para cancelar e escolher uma chamado que esteja em um dos status que permita cancelamento
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação “Cancelar”
Verificação	Tela para cancelar chamado deve ser apresentada contendo o campo: Justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa
Passo	Inserir um texto qualquer no campo Justificativa e confirmar
Verificação	Status do chamado passará a ser “Cancelado”

Atribuir chamado

Descrição: Gerente atribui um chamado a um atendente
Mapa do teste

Passo	Logar no sistema com uma pessoa cujo tipo seja Gerente, escolher um chamado que esteja no status “Não lido”
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação “Atribuir”
Verificação	Tela para atribuir chamado deve ser apresentada contendo os campos: Atendente e Comentário.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo para identificação do Atendente Sistema deve exibir apenas as pessoas cujo tipo seja Atendente no campo correspondente
Passo	Inserir identificação para um atendente e confirmar
Verificação	Status do chamado será alterado de “Não lido” para “Aguarda atendimento” Campo comentário não deve passar por validação, pois seu preenchimento não é obrigatório.

Assumir chamado

Descrição: Atendente executa ação assumir chamado tornando-se o atendente responsável por ele.
Mapa do teste

Passo	Logar no sistema com uma pessoa cujo tipo seja Atendente, escolher um chamado que esteja no status "Não lido"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Assumir"
Verificação	Tela para assumir chamado deve ser apresentada contendo o campo Comentário.
Passo	Confirmar
Verificação	Atendente que realizar a ação será definido como o atendente responsável pelo chamado escolhido Status do chamado passará a ser "Aguardando atendimento" Campo comentário não deve passar por validação, pois seu preenchimento não é obrigatório.

Iniciar

Descrição: Atendente inicia o atendimento de um chamado
Mapa do teste

Passo	Logar no sistema com uma pessoa cujo tipo seja Atendente, escolher um chamado que esteja no status "Aguardando atendimento"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Iniciar"
Verificação	Tela para iniciar chamado deve ser apresentada contendo o campo Comentário.
Passo	Confirmar
Verificação	Status do chamado passará a ser "Em atendimento" Campo comentário não deve passar por validação, pois seu preenchimento não é obrigatório.

Solicitar orçamento

Descrição: Atendente solicita orçamento por parte da gerencia para um chamado específico
Mapa do teste

Passo	Logar no sistema com uma pessoa cujo tipo seja Atendente, escolher um chamado que esteja no status "Em atendimento"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Solicitar orçamento"
Verificação	Tela para solicitar orçamento deve ser apresentada contendo o campo Comentário.
Passo	Confirmar
Verificação	Status do chamado passará a ser "Negociação desenvolvimento" Campo comentário não deve passar por validação, pois seu preenchimento não é obrigatório.

Não orçar

Descrição: Gerente decide não elaborar orçamento solicitado pelo atendente, incluindo sua justificativa para tal

Mapa do teste

Passo	Logar no sistema com uma pessoa cujo tipo seja Gerente, escolher um chamado que esteja no status "Negociação (desenvolvimento)"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Não orçar"
Verificação	Tela para recusa será apresentada contendo o campo Justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa
Passo	Inserir um texto qualquer no campo Justificativa e confirmar
Verificação	Status do chamado passará a ser "Em atendimento"

Enviar orçamento

Descrição: Gerente envia orçamento elaborado para aceitação do Cliente

Mapa do teste

Informação	Enviar orçamento se aplica a chamados que estejam no status "Negociação (desenvolvimento)" e "Negociação", este roteiro deve ser aplicado para cada um dos status citados
Passo	Logar no sistema com uma pessoa cujo tipo seja Gerente, escolher um chamado que esteja em um dos status citados acima.
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Enviar orçamento"
Verificação	Tela para envio de orçamento será apresentada contendo os campos: HH orçado, Tipo de serviço e justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo HH orçado
Passo	Inserir texto no campo tanto digitando quanto colando da área de transferência do SO
Verificação	Sistema deve impedir a inserção de valores não numéricos
Passo	Inserir 6 dígitos no campo e confirmar
Verificação	Sistema deve formatar números inseridos de modo a manter duas casas decimais Sistema deve solicitar o tipo de serviço
Passo	Escolher uma dentre as opções de tipo de serviço e confirmar
Verificação	Sistema deve solicitar que o campo Comentário seja preenchido
Passo	Inserir um texto qualquer no campo comentário e confirmar
Verificação	Status do chamado deve ser alterado de acordo com status de anterior: Status anterior: "Negociação (desenvolvimento)" Status posterior: "Autorização cliente (Negociação desenv.)" Status anterior: "Negociação" Status posterior: "Autorização cliente"

Decidir orçamento

Descrição: Cliente decide entre recusar e aceitar um orçamento enviado pelo Gerente. Dois casos de testes são necessários: um para recusar e outro para aceitar um orçamento.
Para cada teste duas possibilidades existem conforme status do chamado, totalizando 4 testes em dois mapas.

Mapa do 1° teste – Recusar orçamento

Informação	Decidir orçamento se aplica a chamados que estejam no status "Autorização cliente (desenvolvimento)" e "Autorização cliente", este roteiro deve ser aplicado para cada um dos status citados
Passo	Logar no sistema com uma pessoa cujo tipo seja Cliente, escolher um chamado que esteja em um dos status citados acima.
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Decidir orçamento"
Verificação	Tela será apresentada contendo os campos Avaliação e Justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Avaliação
Passo	Dentre as opções "Aceitar orçamento" e "Recusar orçamento" escolher "Recusar orçamento" e confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa
Passo	Preencher o campo Justificativa com um texto qualquer e confirmar
Verificação	Status do chamado deve ser alterado de acordo com status anterior: Status anterior: "Autorização cliente (Negociação desenv.)" Status posterior: "Negociação (desenvolvimento)" Status anterior: "Autorização cliente" Status posterior: "Negociação"

Mapa do 2° teste – Aceitar orçamento

Informação	Decidir orçamento se aplica a chamados que estejam no status "Autorização cliente (desenvolvimento)" e "Autorização cliente", este roteiro deve ser aplicado para cada um dos status citados
Passo	Logar no sistema com uma pessoa cujo tipo seja Cliente, escolher um chamado que esteja em um dos status citados acima.
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Decidir orçamento"
Verificação	Tela será apresentada contendo os campos Avaliação e Justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Avaliação
Passo	Dentre as opções "Aceitar orçamento" e "Recusar orçamento" escolher "Aceitar orçamento" e confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa
Passo	Preencher o campo Justificativa com um texto qualquer e confirmar
Verificação	Status do chamado deve ser alterado de acordo com status anterior: Status anterior: "Autorização cliente (Negociação desenv.)" Status posterior: "Em atendimento" Status anterior: "Autorização cliente" Status posterior: "Fila"

Enviar para desenvolvimento

Descrição: O atendente envia o chamado para desenvolvimento, dependendo da qualificação do chamado seu destino será diferente.
Mapa do teste

Passo	Logar no sistema com uma pessoa cujo tipo seja Atendente, escolher um chamado que esteja no status "Em atendimento"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Enviar para desenvolvimento"
Verificação	Tela será apresentada contendo o campo Analista e Comentário.
Passo	Confirmar
Verificação	Sistema deve solicitar que seja escolhido um analista Sistema deve disponibilizar para escolha, somente pessoas cujo tipo seja "Analista"
Passo	Escolher um analista e confirmar a ação
Verificação	Sistema deve alterar o status do chamado de acordo com a qualificação dele: Se qualificação for "Bug" então o status passará a ser "Fila" Se a qualificação for "Melhoria" então o status será "Análise preliminar"

Estimar

Descrição: Analista realiza estimativa para que a gerência possa realizar um orçamento.
Mapa do teste

Passo	Logar no sistema com o Analista definido para um chamado que esteja no status "Análise preliminar"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Estimar"
Verificação	Tela para estimativa será apresentada contendo os campos: Tipo de serviço, HH estimado e Comentário.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Tipo de serviço Sistema deverá disponibilizar as seguintes opções: Desenv. web, Desenv. Java, Desenv. C# e Banco de dados
Passo	Escolher uma dentre as opções acima e confirmar
Verificação	Sistema deverá solicitar preenchimento do campo HH estimado
Passo	Preencher campo HH estimado com valores não numéricos
Verificação	Sistema deve impedir inserção
Passo	Inserir valor 0 (zero)
Verificação	Sistema deve informar que não é necessário informa um HH, apagará o conteúdo do campo e retornará o foco para o campo HH estimado
Passo	Informar um valor numérico para o HH
Verificação	Sistema deverá formatar a entrada do número de modo a sempre manter uma casa decimal
Passo	Inserir um texto qualquer no campo comentário
Verificação	Qualquer tipo de caractere deve ser aceito
Passo	Apagar conteúdo do campo comentário e confirmar a ação
Verificação	Sistema não deve exigir o preenchimento do campo comentário Status do chamado deve ser alterado para "Negociação"

Desenvolver sem custos

Descrição: Gerente opta por enviar chamado para Fila, independente de ter realizado ou não negociação com o cliente.

Mapa do teste

Passo	Logar no sistema com uma pessoa cujo tipo seja Gerente, escolher um chamado que esteja no status "Negociação (desenvolvimento)"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Não orçar"
Verificação	Tela para recusa será apresentada contendo o campo Justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa
Passo	Inserir um texto qualquer no campo Justificativa e confirmar
Verificação	Status do chamado passará a ser "Em atendimento"

Priorizar

Descrição: Gerente ou o Analista do chamado podem iniciar o trabalho de desenvolvimento, retirando chamado da fila.

Mapa do teste

Informação	Esta ação pode ser executada por um Gerente ou pelo Analista de um chamado que esteja na fila. Verificar caso de testes para os dois papéis
Passo	Logar no sistema com um dos papeis acima e escolher um chamado que esteja no status "Fila"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Priorizar"
Verificação	Tela será apresentada contendo o campo Comentário.
Passo	Confirmar
Verificação	Status do chamado passará a ser "Especificação"

Iniciar desenvolvimento

Descrição: Após o Analista de um chamado ter realizado a especificação necessária, ele pode enviar o chamado para desenvolvimento definindo tarefas e seus respectivos desenvolvedores.

Mapa do teste

Passo	Logar no sistema com o Analista de um chamado que esteja no status "Especificação".
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Enviar para desenvolvimento"
Verificação	Tela contendo os campos: Desenvolvedor responsável, Comentário e uma tabela para exibição de tarefas.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Desenvolvedor responsável Sistema deve exibir apenas as pessoas cadastradas no sistema que sejam desenvolvedores
Passo	Escolher uma das pessoas pra ser o desenvolvedor do chamado e confirmar
Verificação	Status do chamado passará a ser "Em desenvolvimento"

Concluir desenvolvimento

Descrição: Desenvolvedor responsável conclui suas tarefas e passa o chamado para a etapa de inspeção de código

Mapa do teste

Passo	Logar no sistema com o desenvolvedor responsável por um chamado no status "Em desenvolvimento".
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Concluir desenvolvimento" e confirmar
Verificação	Status do chamado passará a ser "Inspeção de código"

Concluir inspeção

Descrição: Inspetor conclui suas atividades decidindo entre aprovar o código avaliado ou reprová-lo.

Concluir inspeção necessita de dois casos de testes, uma para a reprovação, outra para a aprovação

Mapa do 1° teste – Reprovação na inspeção

Passo	Logar no sistema com uma pessoa cujo tipo seja Inspetor, escolher um chamado que esteja no status "Inspeção de código"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação, Desenvolvedor e Histórico de ações.
Passo	Escolher ação "Reprovar na inspeção"
Verificação	Tela para reprovação será apresentada contendo o campo Justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa
Passo	Inserir um texto qualquer no campo Justificativa e confirmar
Verificação	Status do chamado passará a ser "Em desenvolvimento"

Mapa do 2° teste – Aprovar na inspeção

Passo	Logar no sistema com uma pessoa cujo tipo seja Inspetor, escolher um chamado que esteja no status "Inspeção de código"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação, Desenvolvedor e Histórico de ações.
Passo	Escolher ação "Aprovar na inspeção"
Verificação	Tela para reprovação será apresentada contendo o campo Comentário.
Passo	Confirmar
Verificação	Sistema não deve exigir o preenchimento do campo comentário Status do chamado passará a ser "Geração de versão"

Gerar versão

Descrição: Analista após a aprovação na inspeção, deve gerar uma versão a ser entregue. Concluída esta etapa ele pode realizar a ação "Gerar versão", disponibilizando o chamado para que passe pelo controle de qualidade por um tester por ele designado.

Mapa do teste

Passo	Logar no sistema com uma pessoa cujo tipo seja Analista, escolher um chamado que esteja no status "Geração de versão"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Gerar versão" e confirmar
Verificação	Tela para gerar versão será apresentada contendo os campos: Tester, Versão e Comentário
Passo	Confirmar conclusão da ação
Verificação	Sistema deve solicitar preenchimento do campo Tester Sistema deve disponibilizar no campo Tester apenas as pessoas cujo tipo seja Tester
Passo	Escolher uma das pessoas listadas e confirmar
Verificação	Sistema deve solicitar preenchimento do campo Versão
Passo	Tentar inserir no campo versão caracteres não numéricos ou o número zero
Verificação	Sistema deve impedir a inserção de valores não numéricos ou valor nulo
Passo	Inserir um valor numérico no campo Versão e confirmar
Verificação	Status do chamado passará a ser "Testes" Sistema não deve solicitar preenchimento do campo comentário, pois seu preenchimento não é obrigatório

Concluir testes

Descrição: Tester após realizar suas atividades no chamado poderá concluir suas atividades escolhendo entre reprovar ou provar o chamado. Esta caso de testes exige duas situações para sua verificação

Mapa do 1° teste – Reprovar em testes

Passo	Logar no sistema com uma pessoa cujo tipo seja Tester, escolher um chamado que esteja no status "Testes"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação, Desenvolvedor, Inspetor, Analista e Histórico de ações.
Passo	Escolher ação "Reprovar em testes"
Verificação	Tela para recusa será apresentada contendo o campo Justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa
Passo	Inserir um texto qualquer no campo Justificativa e confirmar
Verificação	Status do chamado passará a ser "Em desenvolvimento"

Mapa do 2° teste – Aprovar em testes

Passo	Logar no sistema com uma pessoa cujo tipo seja Tester, escolher um chamado que esteja no status "Testes"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação, Desenvolvedor, Inspetor, Analista e Histórico de ações.
Passo	Escolher ação "Aprovar em testes"
Verificação	Tela para recusa será apresentada contendo o campo Justificativa.

Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa O objetivo deste campo é que sejam inseridos os procedimentos de testes aplicados
Passo	Inserir um texto qualquer no campo Justificativa e confirmar
Verificação	Status do chamado passará a ser "Em atendimento"

Concluir atendimento

Descrição: Atendente ao considerar o chamado atendido, pode realizar a ação de concluir atendimento enviando o chamado para avaliação por parte do cliente.
A conclusão do chamado independe de ter havido ou não desenvolvimento ou orçamento.
Mapa do teste

Passo	Logar no sistema com uma pessoa cujo tipo seja Atendente, escolher um chamado que esteja no status "Em atendimento"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Concluir atendimento"
Verificação	Tela para Conclusão será apresentada contendo o campo Justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa
Passo	Inserir um texto qualquer no campo Justificativa e confirmar
Verificação	Status do chamado passará a ser "Aprovação"

Avaliar atendimento

Descrição: Cliente tem as opções de reprovar um chamado ou aprová-lo finalizando seu atendimento.
Dois casos de testes são necessários
Mapa do 1° teste – Reprovar chamado

Passo	Logar no sistema com uma pessoa cujo tipo seja Cliente, escolher um chamado que esteja no status "Aprovação"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Reprovar chamado"
Verificação	Tela para reprovação será apresentada contendo o campo Justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa
Passo	Inserir um texto qualquer no campo Justificativa e confirmar
Verificação	Status do chamado passará a ser "Reprovado"

Mapa do 2° teste – Aprovar chamado

Passo	Logar no sistema com uma pessoa cujo tipo seja Cliente, escolher um chamado que esteja no status "Aprovação"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Aprovar chamado"
Verificação	Tela para aprovação será apresentada contendo o campo Justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa
Passo	Inserir um texto qualquer no campo Justificativa e confirmar
Verificação	Status do chamado passará a ser "Concluído"

Avaliar reprovação

Descrição: Atendente ao acessar um chamado reprovado pelo seu cliente terá as opções de argumentar a reprovação do cliente, enviando-o novamente para aprovação, ou então poderá retomar o atendimento.
Dois casos de testes são necessários
Mapa do 1° teste - Argumentar

Passo	Logar no sistema com uma pessoa cujo tipo seja Atendente, escolher um chamado que esteja no status "Reprovado"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Argumentar"
Verificação	Tela para argumentação será apresentada contendo o campo Justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa
Passo	Inserir um texto qualquer no campo Justificativa e confirmar
Verificação	Status do chamado passará a ser "Aprovação"

Mapa do 2° teste – Retomar atendimento

Passo	Logar no sistema com uma pessoa cujo tipo seja Atendente, escolher um chamado que esteja no status "Reprovado"
Verificação	Tela de detalhe do chamado deve ser apresentada contendo os seguintes campos: Número chamado, Título, Descrição, Urgência, Qualificação e Histórico de ações.
Passo	Escolher ação "Retomar atendimento"
Verificação	Tela para argumentação será apresentada contendo o campo Justificativa.
Passo	Confirmar
Verificação	Sistema deve solicitar preenchimento do campo Justificativa
Passo	Inserir um texto qualquer no campo Justificativa e confirmar
Verificação	Status do chamado passará a ser "Em atendimento"

Criar tarefa

Descrição: As regras para a criação de tarefas dependem do ator que a realizada
Mapa do teste para o Gerente

Passo	Logar no sistema com todos os atores do sistema, escolher a opção de criar tarefas ou entrar no detalhe de um chamado qualquer e escolher a ação criar tarefas
Verificação	Tela de criação de tarefas será exibida contendo os seguintes campos: • Número chamado: se a criação da tarefa for acionada de dentro do chamado este campo virá preenchido automaticamente • Pessoa responsável: Cliente nunca será disponibilizado neste campo para escolha nem poderá criar tarefas, demais pessoas serão listadas de acordo com o ator que solicita a criação da tarefa. Ator Gerente pode criar tarefa para qualquer pessoa Atores Atendente e Analista só não podem criar tarefas para Gerente Atores Desenvolvedor, Inspetor e Tester somente podem criar tarefas para si próprios • HH tarefa: este campo deve aceitar somente valores reais positivos com uma casa decimal
Passo	Confirmar criação de tarefa
Verificação	Sistema deve solicitar preenchimento do campo Pessoa responsável

Passo	Escolher uma pessoa e confirmar
Verificação	Sistema deve solicitar preenchimento do campo HH tarefa
Passo	Tentar inserir caracteres não numéricos ou apenas o dígito zero
Verificação	Sistema deve permitir inserção de caracteres numéricos apenas ou numéricos desde que o valor do HH seja superior a zero, formatando automaticamente o campo para exibir uma casa decimal
Passo	Inserir um valor numérico qualquer no campo HH estimado e confirmar a criação da tarefa
Verificação	Tarefa deverá ser criada e associada ao chamado desejado

Cancelar e concluir tarefa

Descrição: O cancelamento e a conclusão de uma tarefa seguem regras apenas com relação ao ator que pode realizar.
O Gerente pode cancelar e concluir tarefas de qualquer pessoa,
O atendente e o analista podem cancelar ou concluir tarefas do chamado sobre o qual possuem responsabilidade desde que não seja uma tarefa de Gerente.
Desenvolvedor, Inspetor e Tester podem cancelar e concluir apenas suas próprias tarefas
Utilizar o mapa abaixo considerando os cenários descritos a seguir
Mapa do teste

Passo	Logar no sistema com um dos atores descritos no cenário a ser verificado Entrar no detalhe de um chamado ou acessar no sistema a lista de tarefas Escolher a tarefa a realizar o teste
Verificação	Tela de detalhe da tarefa deve ser exibida contendo os seguintes campos: Número chamado, Número tarefa, Título (da tarefa), Pessoa responsável, HH tarefa.
Passo	Escolher ação designada no cenário e confirmar
Verificação	Sistema deve alterar status da tarefa para "Cancelada" caso tenha sido realizada a ação Cancelar, ou para "Concluída" caso tenha sido realizada a ação Concluir. Campo Comentário não é de preenchimento obrigatório, portanto não deve ser exigido seu preenchimento.

Cenários

1. Com o Gerente acessar uma tarefa para todos os atores e verificar se está disponível a opção para cancelar. Em apenas uma tarefa executar a ação cancelar.
2. Com o Gerente, acessar uma tarefa de cada ator do sistema e verificar se esta disponível a opção de concluir. Em apenas uma destas tarefas, executar a ação Concluir.
3. Com o atendente de um chamado entrar no detalhe deste chamado e verificar se esta disponível a opção de cancelar as tarefas em andamento dos atores Desenvolvedor, Inspetor, Tester e do próprio Atendente
4. Com o atendente de um chamado entrar no detalhe deste chamado e verificar se esta disponível a opção de Concluir as tarefas em andamento dos atores Desenvolvedor, Inspetor, Tester e do próprio Atendente
5. Com um atendente entrar no detalhe de um chamado em que ele não tenha responsabilidade (não tenha assumido ou não tenha sido atribuído a ele). Dentre as tarefas existentes neste chamado, verificar se não esta disponível a opção para concluí-las ou cancelá-las. Somente ficarão disponível estas opções caso alguma tarefa esteja sob sua responsabilidade
6. Para todos os atores, as opções de cancelar e de concluir deverão estar disponíveis para as tarefas sob a responsabilidade do próprio ator.

Reclassificar

Descrição: Chamado pode ser reclassificado pelo Atendente a qualquer momento.
Mapa do teste

Passo	Logar no sistema com uma pessoa cujo tipo seja Atendente, escolher um chamado sob sua responsabilidade, independente do status em que se encontrar
Verificação	Tela de detalhe do chamado deve ser apresentada
Passo	Escolher ação “Reclassificar”
Verificação	Tela para reclassificação será apresentada contendo os campos: Título, Descrição, Urgência e Classificação Campos virão preenchidos com os dados originais Apenas o Título não poderá ser alterado Urgência deve exibir as seguintes opções para escolha: “Alta – impeditiva”, “Alta”, “Média” e “Baixa” Classificação deve exibir as seguintes opções para escolha: “Bug” e “Melhoria”
Passo	Apagar o conteúdo de todos os campos e confirmar
Verificação	Sistema deve solicitar preenchimento de todos os campos
Passo	Inserir dados nos campos de forma a deixá-los diferente do que estava anteriormente e confirmar
Verificação	Sistema deve submeter os dados sem alterar o status do chamado
Passo	Acessar novamente o chamado e confirmar se as alterações realizadas foram de fato submetidas.

Apêndice E

Casos de Testes Fabrica de Software - Info Cases

Índice

ATOR: Cliente IC 28: Abrir chamado

➔ chamado = dt_chamado + titulo_chamado + descricao + urgencia + qualificacao_chamado

Descrição: Informação de uma solicitação de serviço por parte de um cliente. A data será automaticamente inserida pelo sistema armazenando, inclusive, informação de horas.
Propósito: Informar ao sistema a abertura de um chamado marcando seu status como "Não lido"

➔ id_chamado

Aplicação de testes

Apenas o ator cliente pode executar este IC

Fluxo de entrada

dt_chamado

(Particionamento de equivalência)
Condição de entrada, booleano

- sistema deve preencher automaticamente com a data corrente.
- apagar a data do campo: sistema não deve permitir a edição deste campo

titulo_chamado

(Particionamento de equivalência)
Condição de entrada, booleano

- abrir chamado sem inserir o título: sistema deve impedir a criação do chamado, exigindo que o campo seja preenchido
- preencher com um valor válido e proceder com a criação do chamado: chamado deve ser aberto com sucesso

(Particionamento de equivalência)
Condição de entrada, intervalo

- inserir texto livre com número de caracteres inferior a 100 unidades: chamado deve ser aberto com sucesso
- inserir texto com mais do que 100 caracteres: chamado deve ser aberto com sucesso

(análise do valor limite)

- inserir texto com exatamente 100 caracteres: chamado deve ser aberto com sucesso
- inserir texto com exatamente 101 caracteres: sistema deve impedir a inserção de mais do que 100 caracteres para que o usuário saiba que o limite foi atingido
- inserir texto com exatamente 1 caractere: chamado deve ser aberto com sucesso

descricao

(Particionamento de equivalência)
Condição de entrada, booleano

- abrir chamado sem inserir nenhuma descrição: sistema deve impedir a criação do chamado, exigindo que o campo seja preenchido
- preencher com um valor qualquer e proceder com a criação do chamado: chamado deve ser aberto com sucesso

(Particionamento de equivalência)
Condição de entrada, intervalo: como não foi determinado limite ou regras para este campo, o conjunto inválido é o conjunto vazio e o válido qualquer conjunto de caracteres.

- solicitar chamado inserindo apenas 1 caractere no campo: sistema deve permitir a criação do chamado
- solicitar chamado com um texto que contenha tipos variados de caracteres: sistema deve permitir a criação do chamado

urgencia

(Particionamento de equivalência)
Condição de entrada, booleano
- abrir chamado sem preencher opção de urgência: sistema deve impedir a criação do chamado, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer e proceder com a criação do chamado: chamado deve ser aberto com sucesso

Condição de entrada, intervalo: sistema deve disponibilizar os valores válidos: "Alta - Impeditiva", "Alta", "Media" e "Baixa"
- solicitar chamado escolhendo uma das opções oferecidas pelo sistema: chamado deve ser aberto som sucesso.
- solicitar chamado inserindo valores diferentes dos oferecidos pelo sistema (seja por digitação ou colando texto copiado de outro local via área de transferência do sistema operacional): sistema deve impedir inserção de valores diferentes dos apresentados

qualificacao_chamado
(Particionamento de equivalência)
Condição de entrada, booleano
- abrir chamado sem preencher opção de qualificação: sistema deve impedir a criação do chamado, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer e proceder com a criação do chamado: chamado deve ser aberto com sucesso

Condição de entrada, intervalo: sistema deve disponibilizar os valores válidos: "Bug" e "Melhoria"
- solicitar chamado escolhendo uma das opções oferecidas pelo sistema: chamado deve ser aberto som sucesso.
- solicitar chamado inserindo valores diferentes dos oferecidos pelo sistema (seja por digitação ou colando texto copiado de outro local via área de transferência do sistema operacional): sistema deve impedir inserção de valores diferentes dos apresentados

Fluxo de saída
id_chamado
- deve ser gerado um número natural maior do que zero para cada um dos chamados abertos durante execução dos casos de testes deste roteiro
- os números gerados devem ser sempre crescentes, sem coincidência nem saltos na seqüência.

Resultado esperado
Após realização deste IC, o status do chamado deve ser "Não lido"

ATOR: Atendente IC 29: Registrar chamado

➔ chamado = dt_chamado + titulo_chamado +descricao + urgencia + qualificacao_chamado + id_pessoa

Descrição: registro de chamado realizado diretamente por um atendente, o cliente deve ser informado podendo ser qualquer pessoa cadastrada no sistema.
O atendente para este chamado será automaticamente inserido com o nome de quem fez o registro, sem possibilidade de alteração.
Propósito: Informar ao sistema a abertura de um chamado marcando seu status como "Aguardando atendimento"

← id_chamado

Aplicação de testes
Apenas o ator Atendente pode executar este IC

Fluxo de entrada

dt_chamado

(Particionamento de equivalência)
Condição de entrada, booleano

- sistema deve preencher automaticamente com a data corrente.
- apagar a data do campo: sistema não deve permitir a edição deste campo

titulo_chamado

(Particionamento de equivalência)
Condição de entrada, booleano

- abrir chamado sem inserir o título: sistema deve impedir a criação do chamado, exigindo que o campo seja preenchido
- preencher com um valor válido e proceder com a criação do chamado: chamado deve ser aberto com sucesso

(Particionamento de equivalência)
Condição de entrada, intervalo

- inserir texto livre com número de caracteres inferior a 100 unidades: chamado deve ser aberto com sucesso
- inserir texto com mais do que 100 caracteres: chamado deve ser aberto com sucesso

(análise do valor limite)

- inserir texto com exatamente 100 caracteres: chamado deve ser aberto com sucesso
- inserir texto com exatamente 101 caracteres: sistema deve impedir a inserção de mais do que 100 caracteres para que o usuário saiba que o limite foi atingido
- inserir texto com exatamente 1 caractere: chamado deve ser aberto com sucesso

descricao

(Particionamento de equivalência)
Condição de entrada, booleano

- abrir chamado sem inserir nenhuma descrição: sistema deve impedir a criação do chamado, exigindo que o campo seja preenchido
- preencher com um valor qualquer e proceder com a criação do chamado: chamado deve ser aberto com sucesso

(Particionamento de equivalência)
Condição de entrada, intervalo: como não foi determinado limite ou regras para este campo, o conjunto inválido é o conjunto vazio e o válido qualquer conjunto de caracteres.

- solicitar chamado inserindo apenas 1 caractere no campo: sistema deve permitir a criação do chamado
- solicitar chamado com um texto que contenha tipos variados de caracteres: sistema deve permitir a criação do chamado

urgencia

(Particionamento de equivalência)
Condição de entrada, booleano

- abrir chamado sem preencher opção de urgência: sistema deve impedir a criação do chamado, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer e proceder com a criação do chamado: chamado deve ser aberto com sucesso

Condição de entrada, intervalo: sistema deve disponibilizar os valores válidos: “Alta - Impeditiva”, “Alta”, “Media” e “Baixa”

- solicitar chamado escolhendo uma das opções oferecidas pelo sistema: chamado deve ser aberto som sucesso.
- solicitar chamado inserindo valores diferentes dos oferecidos pelo sistema (seja por digitação ou colando texto copiado de outro local via área de transferência do sistema operacional): sistema deve impedir inserção de valores diferentes dos apresentados

qualificacao_chamado

(Particionamento de equivalência)
Condição de entrada, booleano

- abrir chamado sem preencher opção de qualificação: sistema deve impedir a criação do chamado, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer e proceder com a criação do chamado: chamado deve ser aberto com sucesso

Condição de entrada, intervalo: sistema deve disponibilizar os valores válidos: "Bug" e "Melhoria"

- solicitar chamado escolhendo uma das opções oferecidas pelo sistema: chamado deve ser aberto som sucesso.
- solicitar chamado inserindo valores diferentes dos oferecidos pelo sistema (seja por digitação ou colando texto copiado de outro local via área de transferência do sistema operacional): sistema deve impedir inserção de valores diferentes dos apresentados

id_pessoa

(Particionamento de equivalência)

Condição de entrada, booleano: cliente deve ser informado.

- registrar chamado sem identificar um cliente: sistema deve impedir o registro do chamado
- registrar chamado identificando um cliente: registro do chamado deve ser realizado com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens

- solicitar pedido inserindo identificação de cliente existente na base: chamado deve ser registrado com sucesso

Observação: como não foi especificada nenhuma limitação com relação ao tipo de pessoa que pode ser inserida como cliente para um chamado, entende-se que qualquer pessoa pode ser o cliente.

Fluxo de saída

id_chamado

- deve ser gerado um número natural maior do que zero para cada um dos chamados abertos durante execução dos casos de testes deste roteiro
- os números gerados devem ser sempre crescentes, sem coincidência nem saltos na seqüência.

Resultado esperado

Após realização deste IC, o status do chamado deve ser "Aguardando atendimento"

ATOR: Cliente/Atendente/Gerente **IC 30: Cancelar chamado**

➔ cancela_chamado = id_chamado + justificativa

Descrição: IC pode ser realizado a qualquer momento tanto pelo cliente quanto pelo atendente responsável.

Propósito: realizar cancelamento de um chamado, independente do status dele e alterando seu status para "Cancelado".

Aplicação de testes

Usando o ator Atendente, executar os casos de testes abaixo e em seguida conferir se os atores Cliente e Gerente podem realizar este IC

Fluxo de entrada

id_chamado

(Particionamento de equivalência)

Condição de entrada, booleano: chamado deve ser informado.

- solicitar cancelamento sem identificar o chamado que se deseja cancelar: sistema deve impedir o registro do chamado

- solicitar cancelamento identificando chamado: cancelamento deve ser realizado com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens

- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.

justificativa

(Particionamento de equivalência)

Condição de entrada, booleano: justificativa deve ser informada.

- solicitar cancelamento sem preencher a justificativa: sistema deve solicitar preenchimento do campo
- solicitar informando a justificativa: cancelamento deve ser realizado com sucesso

Condição de entrada, intervalo: como não foi determinado limite ou regras para este campo, o conjunto inválido é o conjunto vazio e o válido qualquer conjunto de caracteres.

- realizar ação inserindo apenas 1 caractere no campo: sistema deve permitir a ação
- realizar ação com um texto que contenha tipos variados de caracteres: sistema deve permitir a ação respeitando os caracteres inseridos

Resultado esperado

Após realização deste IC, o status do chamado deve ser "Cancelado"

ATOR: Gerente IC 31: Atribuir

➔ atribuir_chamado = id_chamado + id_pessoa + comentario

Propósito: Atribuir um chamado à responsabilidade de um atendente específico. Chamado terá seu status alterado para "Aguardando atendimento"

Aplicação de testes

Apenas o perfil de Gerente deve poder executar este IC

Fluxo de entrada

id_chamado

(Particionamento de equivalência)

Condição de entrada, booleano: chamado deve ser informado.

- solicitar cancelamento sem identificar o chamado que se deseja cancelar: sistema deve impedir o registro do chamado
- solicitar cancelamento identificando chamado: cancelamento deve ser realizado com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens

- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.

id_pessoa

(Particionamento de equivalência)

Condição de entrada, booleano

- sistema deve preencher automaticamente com o atendente que executa o registro do chamado.
- apagar valor deste campo: sistema não deve permitir a edição deste campo

Condição de entrada, intervalo

- somente pessoas do tipo atendente devem ser listadas

comentario

(Particionamento de equivalência)

Condição de entrada, booleano: comentário é de preenchimento opcional.

- realizar ação sem preencher o comentário: sistema deve realizar a ação desejada

- realizar ação preenchendo o comentário: sistema deve realizar a ação desejada

Condição de entrada, intervalo: como não foi determinado limite ou regras para este campo, qualquer entrada deve ser aceita, inclusive o conjunto vazio.

- realizar ação inserindo apenas 1 caractere no campo: sistema deve permitir a ação
- realizar ação com um texto que contenha tipos variados de caracteres: sistema deve permitir a ação respeitando os caracteres inseridos

Resultado esperado

Após realização deste IC, o status do chamado deve ser "Aguardando atendimento"

ATOR: Atendente	**IC 32: Assumir**

➔ **assumir_chamado = id_chamado + comentario**

Propósito: Um atendente assume um chamado indicando que será ele o responsável pelo seu atendimento. O status do chamado será atualizado para "Aguardando atendimento"

Aplicação de testes

Apenas o perfil de Atendente deve poder executar este IC

Fluxo de entrada

id_chamado

(Particionamento de equivalência)

Condição de entrada, booleano: chamado deve ser informado.

- realizar ação sem identificar o chamado: sistema deve impedir a ação
- realizar a ação identificando o chamado com numero válido: ação deve ser realizada com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens

- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.
- inserir números de chamados que sejam do conjunto dos naturais não nulos: ação deve ser realizada com sucesso.

comentario

(Particionamento de equivalência)

Condição de entrada, booleano: comentário é de preenchimento opcional.

- realizar ação sem preencher o comentário: sistema deve realizar a ação desejada
- realizar ação preenchendo o comentário: sistema deve realizar a ação desejada

Condição de entrada, intervalo: como não foi determinado limite ou regras para este campo, qualquer entrada deve ser aceita, inclusive o conjunto vazio.

- realizar ação inserindo apenas 1 caractere no campo: sistema deve permitir a ação
- realizar ação com um texto que contenha tipos variados de caracteres: sistema deve permitir a ação respeitando os caracteres inseridos

Resultado esperado

Após realização deste IC, o status do chamado deve ser "Aguardando atendimento"

ATOR: Atendente **IC 33: Iniciar**

➔ **iniciar = id_chamado + comentario**

Propósito: Indicar que o chamado esta sendo atendido. O status do chamado será atualizado para "Em atendimento"

Aplicação de testes

Apenas o perfil de Atendente deve poder executar este IC

Fluxo de entrada

(Idêntico ao IC 5)

Resultado esperado

Após realização deste IC, o status do chamado deve ser "Em atendimento"

ATOR: Atendente **IC 34: Solicitar orçamento**

➔ **solicitar_orc = id_chamado + comentario**

Descrição: Envia chamado para responsabilidade da gerencia de desenvolvimento para elaboração de orçamento.

Propósito: O status do chamado será atualizado para "Negociação desenvolvimento"

Aplicação de testes

Apenas o perfil de Atendente deve poder executar este IC

Fluxo de entrada

(Idêntico ao IC 5)

Resultado esperado

Após realização deste IC, o status do chamado deve ser "Negociação desenvolvimento"

ATOR: Gerente **IC 35: Não orçar**

➔ **nao_orc = id_chamado + justificativa**

Descrição: Gerente pode se recusar a realizar um orçamento quando considerar que as informações são insuficientes.

Propósito: O status do chamado será atualizado para "Em atendimento"

Aplicação de testes

Apenas o perfil de Atendente deve poder executar este IC

Fluxo de entrada

(Idêntico ao IC 3)

Resultado esperado

Após realização deste IC, o status do chamado deve ser "Em atendimento"

ATOR: Gerente **IC 36: Enviar Orçamento**

➔ **enviar_orc = id_chamado + tipo_servico + HH_orcado + justificativa**

Descrição: Gerente disponibiliza o orçamento informando o tipo de serviço a ser realizado.

Propósito: Caso o status do chamado seja "Negociação desenvolvimento" então ele será atualizado para "Autorização cliente (Negociação desenvolvimento)".
Se o status for "Negociação" então ele será atualizado para "Autorização cliente"

Aplicação de testes

Apenas as pessoas com tipo Gerência podem executar este IC

Fluxo de entrada

id_chamado

(Particionamento de equivalência)

Condição de entrada, booleano: chamado deve ser informado.

- realizar ação sem identificar o chamado: sistema deve impedir a ação
- realizar a ação identificando o chamado com numero válido: ação deve ser realizada com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens

- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.
- inserir números de chamados que sejam do conjunto dos naturais não nulos: ação deve ser realizada com sucesso.

tipo_servico

Condição de entrada, booleano

- enviar orçamento sem definir o tipo de serviço: sistema deve impedir a ação, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer e enviar o orçamento: ação deve ser realizada com sucesso

(Particionamento de equivalência)

Condição de entrada, intervalo: sistema deve disponibilizar os valores válidos: "Desenv. web", "Desenv. Java", "Desenv. C#" e "Banco de dados"

- Enviar orçamento escolhendo uma das opções oferecidas pelo sistema: ação deve ser realizada com sucesso.
- solicitar chamado inserindo valores diferentes dos oferecidos pelo sistema (seja por digitação ou colando texto copiado de outro local via área de transferência do sistema operacional): sistema deve impedir inserção de valores diferentes dos apresentados

HH_orcado

Condição de entrada, booleano

- enviar orçamento sem passar o valor: sistema deve impedir a ação, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer e enviar o orçamento: ação deve ser realizada com sucesso

(Particionamento de equivalência)

- inserir caracteres não numéricos no campo: sistema não deve aceitar caracteres
- inserir caracteres numéricos: sistema deve aceitar, formatando a vírgula para apenas duas casas

justificativa

(Particionamento de equivalência)

Condição de entrada, booleano: justificativa deve ser informada.

- solicitar cancelamento sem preencher a justificativa: sistema deve solicitar preenchimento do campo
- solicitar informando a justificativa: cancelamento deve ser realizado com sucesso

Condição de entrada, intervalo: como não foi determinado limite ou regras para este campo, o conjunto inválido é o conjunto vazio e o válido qualquer conjunto de caracteres.

- realizar ação inserindo apenas 1 caractere no campo: sistema deve permitir a ação
- realizar ação com um texto que contenha tipos variados de caracteres: sistema deve permitir a ação respeitando os caracteres inseridos

Para verificar status do chamado após execução deste IC, proceder com o envio do orçamento par chamados nos status: "Negociação desenvolvimento" e "Negociação" o status resultante será "Autorização cliente (Negociação desenvolvimento)" e "Autorização cliente", respectivamente.

ATOR: Cliente **IC 37: Decidir orçamento**

➔ **decidir_orc = id_chamado + decisao_orcamento + justificativa**

Descrição: Cliente pode recusar um orçamento elaborado, informando a razão da recusa, para que o Gerente possa prosseguir com a negociação.

Propósito: chamado terá seu status definido de acordo com o status no momento em que o cliente recusar o orçamento:

Se status_chamado = "Autorização cliente (Negociação desenvolvimento)", então o novo status será "Negociação desenvolvimento"

Se status_chamado = "Autorização cliente (Negociação)", então o novo status será "Negociação"

Aplicação de testes

Os perfis de Cliente e Atendente devem poder executar este IC

Observação**: e**ste IC ocorre em duas situações distintas, para a verificação proceder de acordo com o grafo de admissibilidade para cobrir as duas possibilidades existentes:

Testes para aceitação de orçamento
Primeiro teste (caminho 2-5-6-7-9-11)
Fluxo de entrada
(Idêntico ao IC 3)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Autorização cliente (Negociação desenvolvimento)" para "Em atendimento"

Segundo teste (caminho 2-5-6-12-13-14-11)
Fluxo de entrada
(Idêntico ao IC 3)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Autorização cliente (Negociação)" para "Fila"

Testes para recusa de orçamento
Primeiro teste (caminho 2-5-6-7-9-10)
Fluxo de entrada
(Idêntico ao IC 3)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Autorização cliente (Negociação desenvolvimento)" para "Negociação desenvolvimento"

Segundo teste (caminho 2-5-6-12-13-14-10)
Fluxo de entrada
(Idêntico ao IC 3)

Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Autorização cliente (Negociação) para "Negociação"

ATOR: Atendente IC 38: Enviar para desenvolvimento

➔ enviar_desenv = id_chamado + comentario

Descrição: O atendente envia um chamado para desenvolvimento.
Propósito: Dependendo da qualificação do chamado seu status será diferente. Caso seja "Bug" status = "fila", se for "Melhoria" status = "Análise preliminar"

Aplicação de testes
Este IC deve ser executado apenas pelo ator Atendente
Observação**:** este IC possui duas saídas distintas de acordo com a qualificação do chamado, para verificar a saída (teste de condição) serão necessários dois chamados, um com qualificação = "bug" e outro com qualificação = "Melhoria"

Primeiro teste (qualificacao = "bug")
Fluxo de entrada
(Idêntico ao IC 5)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Autorização Em atendimento" para "Fila"

Segundo teste (qualificacao = "melhoria")
Fluxo de entrada
(Idêntico ao IC 5)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Em atendimento" para "Análise preliminar"

ATOR: Analista IC 39: Estimar

➔ estimar = id_chamado + tipo_servico + HH_estimado + comentario

Descrição: o analista ao verificar o chamado pode identificar a necessidade de serviços não orçados anteriormente. Neste momento ele envia o chamado para a Gerente informando o tipo de serviço que será realizado e quanto de horas estimou ser necessário para sua conclusão.
Propósito: O status do chamado será atualizado para "Negociação"

Aplicação de testes
Apenas o ator Analista pode executar este IC
Fluxo de entrada
id_chamado
(Particionamento de equivalência)
Condição de entrada, booleano: chamado deve ser informado.
- realizar ação sem identificar o chamado: sistema deve impedir a ação
- realizar a ação identificando o chamado com numero válido: ação deve ser realizada com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens
- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.

- inserir números de chamados que sejam do conjunto dos naturais não nulos: ação deve ser realizada com sucesso.

tipo_servico

Condição de entrada, booleano

- enviar orçamento sem definir o tipo de serviço: sistema deve impedir a ação, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer e enviar o orçamento: ação deve ser realizada com sucesso

(Particionamento de equivalência)

Condição de entrada, intervalo: sistema deve disponibilizar os valores válidos: "Desenv. web", "Desenv. Java", "Desenv. C#" e "Banco de dados"

- Enviar orçamento escolhendo uma das opções oferecidas pelo sistema: ação deve ser realizada com sucesso.
- solicitar chamado inserindo valores diferentes dos oferecidos pelo sistema (seja por digitação ou colando texto copiado de outro local via área de transferência do sistema operacional): sistema deve impedir inserção de valores diferentes dos apresentados

HH_estimado

Condição de entrada, booleano

- enviar orçamento sem definir o hh_estimado: sistema deve impedir a ação, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer e enviar: ação deve ser realizada com sucesso

(Particionamento de equivalência)

Condição de entrada, intervalo: valores reais com uma casa decimal apenas

- inserir valores não numéricos: sistema deve impedir ação
- inserir valores numéricos: sistema deve aceitar formatando para uma casa decimal

(Análise do valor limite)

- informar valor zero: sistema deve bloquear a ação
- informar "0,1": sistema deve aceitar

comentario

(Particionamento de equivalência)

Condição de entrada, booleano: comentário é de preenchimento opcional.

- realizar ação sem preencher o comentário: sistema deve realizar a ação desejada
- realizar ação preenchendo o comentário: sistema deve realizar a ação desejada

Condição de entrada, intervalo: como não foi determinado limite ou regras para este campo, qualquer entrada deve ser aceita, inclusive o conjunto vazio.

- realizar ação inserindo apenas 1 caractere no campo: sistema deve permitir a ação
- realizar ação com um texto que contenha tipos variados de caracteres: sistema deve permitir a ação respeitando os caracteres inseridos

Resultado esperado

Após execução do IC o status_chamado deve ser alterado de "Análise preliminar" para "Negociação"

ATOR: Gerente IC 40: Desenvolver sem custos

➔ desenv_scusto = id_chamado + comentario

Descrição: durante a fase de negociação, a Gerente pode decidir por enfileirar o chamado sem considerar custos para o cliente.

Propósito: O status do chamado será atualizado para "Fila"

Aplicação de testes

Apenas o ator Gerência deve poder executar este IC

Fluxo de entrada

(Idêntico ao IC 5)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Negociação" para "Fila"

ATOR: Gerente IC 41: Priorizar

➔ priorizar = id_chamado + comentario

Descrição: Ao priorizar o chamado é passado para a responsabilidade do Analista da equipe responsável definida anteriormente, neste momento o analista definirá a especificação técnica necessária para que o trabalho seja desenvolvido.
Propósito: O status do chamado será atualizado para "Especificação"

Aplicação de testes
Apenas o ator Gerência deve poder executar este IC
Fluxo de entrada
(Idêntico ao IC 5)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Fila" para "Especificação"

ATOR: Analista IC 42: Iniciar desenvolvimento

➔ iniciar_desenv = id_chamado + id_pessoa + comentario

Propósito: permitir ao analista que enviei o chamado para desenvolvimento, o status do chamado será atualizado para "Em desenvolvimento"

Aplicação de testes
Apenas o ator Analista deve poder executar este IC
Fluxo de entrada
id_chamado
(Particionamento de equivalência)
Condição de entrada, booleano: chamado deve ser informado.
- realizar ação sem identificar o chamado: sistema deve impedir a ação
- realizar a ação identificando o chamado com numero válido: ação deve ser realizada com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens
- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.
- inserir números de chamados que sejam do conjunto dos naturais não nulos: ação deve ser realizada com sucesso.

id_pessoa
(Particionamento de equivalência)
Condição de entrada, booleano: desenvolvedor deve ser informado.
- enviar para desenvolvimento sem identificar um desenvolvedor: sistema deve impedir a ação
- enviar para desenvolvimento identificando um desenvolvedor: ação deve ser realizada com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens
- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.
- enviar para desenvolvimento sem que exista uma pessoa cadastrada no sistema cujo tipo seja "Desenvolvedor": sistema deve informar a necessidade de cadastrar uma pessoa

- apenas as pessoas com tipo "Desenvolvedor" devem ser aceitas pelo sistema, todas as demais não poderão ser designadas como desenvolvedor responsável por um chamado.

comentario
(Particionamento de equivalência)
Condição de entrada, booleano: comentário é de preenchimento opcional.
- realizar ação sem preencher o comentário: sistema deve realizar a ação desejada
- realizar ação preenchendo o comentário: sistema deve realizar a ação desejada

Condição de entrada, intervalo: como não foi determinado limite ou regras para este campo, qualquer entrada deve ser aceita, inclusive o conjunto vazio.
- realizar ação inserindo apenas 1 caractere no campo: sistema deve permitir a ação
- realizar ação com um texto que contenha tipos variados de caracteres: sistema deve permitir a ação respeitando os caracteres inseridos

Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Especificação" para "Em desenvolvimento"

ATOR: Desenvolvedor IC 43: Concluir desenvolvimento

➔ **concluir_desenv = id_chamado + comentario**

Descrição: somente poderá concluir um chamado o desenvolvedor definido para o ele.
Propósito: o status do chamado será atualizado para "Inspeção de código"

Aplicação de testes
Apenas o ator Desenvolvedor deve poder executar este IC
Fluxo de entrada
(Idêntico ao IC 5)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Em desenvolvimento" para "Inspeção de código"

ATOR: Inspetores IC 44: Concluir inspeção

➔ **concluir_inspecao = id_chamado + result_inspec + justificativa**

Descrição: caso sejam encontrados trechos de código que contenham erros ou que possam ser melhorados, o inspetor retorna o chamado para desenvolvimento para que os acertos necessários sejam realizados.
Propósito: o status do chamado será atualizado de acordo com a opção escolhida para **result_inspec**.
Se for "Aprovado" então o status será atualizado para "Geração de versão"
Se for "Reprovado" então o status será atualizado para "Em desenvolvimento"

Aplicação de testes para Reprovação em inspeção
Requisitos
Apenas o ator Inspetor deve poder executar este IC
Para a reprovação do chamado o **result_inspec** deve ser "Reprovado em inspeção**"**
Fluxo de entrada
(Idêntico ao IC 3)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Inspeção de código" para "Em desenvolvimento"

Aplicação de testes para Aprovação em inspeção

Requisitos

Apenas o ator Inspetor deve poder executar este IC

Para a reprovação do chamado o **result_inspec** deve ser "Aprovado em inspeção**"**

Fluxo de entrada

(Idêntico ao IC 3)

Resultado esperado

Após execução do IC o status_chamado deve ser alterado de "Inspeção de código" para "Geração de versão"

ATOR: Analista IC 45: Gerar versão

➔ **gerar_versao = id_chamado + comentario + id_pessoa + nr_release**

Descrição: o analista disponibiliza o chamado juntamente com todos os arquivos e procedimentos necessários à entrega para que um tester definido por ele possa realizar a verificação, tendo como base a especificação técnica elabora anteriormente.

Deve ser passada a identificação de uma pessoa pelo **id_pessoa**, desde que seu tipo seja "Tester".

Propósito: o status do chamado será atualizado para "Testes"

Aplicação de testes

Fluxo de entrada

id_chamado

(Particionamento de equivalência)

Condição de entrada, booleano: chamado deve ser informado.

- realizar ação sem identificar o chamado: sistema deve impedir a ação
- realizar a ação identificando o chamado com numero válido: ação deve ser realizada com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens

- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.
- inserir números de chamados que sejam do conjunto dos naturais não nulos: ação deve ser realizada com sucesso.

comentario

(Particionamento de equivalência)

Condição de entrada, booleano: comentário é de preenchimento opcional.

- realizar ação sem preencher o comentário: sistema deve realizar a ação desejada
- realizar ação preenchendo o comentário: sistema deve realizar a ação desejada

Condição de entrada, intervalo: como não foi determinado limite ou regras para este campo, qualquer entrada deve ser aceita, inclusive o conjunto vazio.

- realizar ação inserindo apenas 1 caractere no campo: sistema deve permitir a ação
- realizar ação com um texto que contenha tipos variados de caracteres: sistema deve permitir a ação respeitando os caracteres inseridos

id_pessoa

(Particionamento de equivalência)

Condição de entrada, booleano: tester deve ser informado.

- enviar para testes sem identificar um tester: sistema deve impedir a ação
- enviar para testes identificando um tester: ação deve ser realizada com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens

- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.

- enviar para testes sem que exista uma pessoa cadastrada no sistema cujo tipo seja "Tester": sistema deve informar a necessidade de cadastrar uma pessoa
- apenas as pessoas com tipo "Tester" devem ser aceitas pelo sistema, todas as demais não poderão ser designadas como tester para um chamado.

nr_release
(Particionamento de equivalência)
Condição de entrada, booleano: release deve ser informado.
- enviar para testes sem informar release: sistema deve impedir a ação
- enviar para testes identificando o release: ação deve ser realizada com sucesso

Condição de entrada, intervalo:
- inserir caracteres não numéricos: sistema deve impedir inserção
- inserir caracteres numéricos: sistema deve aceitar inserção

Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Geração de versão" para "Testes"

ATOR: Tester **IC 46: Concluir testes**

➔ **concluir_testes = id_chamado + result_testes + justificativa**

Descrição: se julgar necessário o tester reprova o chamado indicando a razão da reprovação.
O chamado retornará para a o desenvolvedor definido como responsável pelo chamado.
Propósito: o status do chamado será atualizado de acordo com a opção escolhida para **result_testes**.
Se for "Aprovado" então o status será atualizado para "Em atendimento"
Se for "Reprovado" então o status será atualizado para "Em desenvolvimento"

Aplicação de testes para Reprovação em testes
Requisitos
Apenas o ator Tester deve poder executar este IC
Para a reprovação do chamado o **result_testes** deve ser "Reprovado em testes**"**
Fluxo de entrada
(Idêntico ao IC 3)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Testes" para "Em desenvolvimento"

Aplicação de testes para Aprovação em testes
Requisitos
Apenas o ator Tester deve poder executar este IC
Para a reprovação do chamado o **result_testes** deve ser "Aprovado em testes**"**
Fluxo de entrada
(Idêntico ao IC 3)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Testes" para "Em atendimento"

ATOR: Atendente **IC 47: Concluir atendimento**

➔ **concluir_atend = id_chamado + justificativa**

Descrição: ao concluir um chamado o atendente o envia para análise do cliente. Neste momento ocorre a entrega do chamado com todos os arquivos e passos aprovados pelo Tester, caso o chamado tenha passado por desenvolvimento.
Propósito: o status do chamado será atualizado para "Aprovação"

Aplicação de testes
Apenas o ator Atendente deve poder executar este IC
Fluxo de entrada
(Idêntico ao IC 3)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Em atendimento" para "Aprovação"

ATOR: Atendente / Cliente **IC 48: Avaliar atendimento**

➔ **avaliar_atendimento = id_chamado + avaliacao_atendimento + justificativa**

Descrição: cliente pode aprovar ou reprovar um chamado informando a razão para que o atendente avalie a situação.
Propósito: o status do chamado será atualizado de acordo com a opção escolhida para **avaliacao_atendimento**.
Se for "Aprovado" então o status será atualizado para "Aprovado"
Se for "Reprovado" então o status será atualizado para "Reprovado"

Aplicação de testes para Reprovação de chamado
Requisitos
Apenas os atores Cliente e Atendente podem executar este IC
Para a reprovação do chamado o **avaliacao_atendimento** deve ser "Reprovado**"**
Fluxo de entrada
(Idêntico ao IC 3)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Aprovação" para "Reprovado"

Aplicação de testes para Aprovação de chamado
Requisitos
Apenas os atores Cliente e Atendente podem executar este IC
Para a reprovação do chamado o **avaliacao_atendimento** deve ser "Aprovado**"**
Fluxo de entrada
(Idêntico ao IC 3)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Aprovação" para "Aprovado"

ATOR: Atendente **IC 49: Avaliar reprovação**

➔ **avaliar_reprovacao = id_chamado + decisao + justificativa**

Descrição: atendente decide entre argumentar a reprovação do cliente enviando o chamado novamente para "Aprovação" ou se vai retomar o atendimento enviando o chamado para "Em atendimento"
Propósito: o status do chamado será atualizado de acordo com a opção escolhida para **decisao**.
Se for "Argumentar" então o status será atualizado para "Aprovação"
Se for "Retomar atendimento" então o status será atualizado para "Em atendimento"

Aplicação de testes para Argumentação da reprovação
Requisitos
Apenas o ator Atendente pode executar este IC

Para a argumentação do chamado o campo **decisao** deve ser "Argumentar reprovação**"**
Fluxo de entrada
(Idêntico ao IC 3)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Reprovado" para "Aprovação"

Aplicação de testes para Retomada de atendimento
Requisitos
Apenas o ator Atendente pode executar este IC
Para a argumentação do chamado o campo **decisao** deve ser "Retomar atendimento**"**
Fluxo de entrada
(Idêntico ao IC 3)
Resultado esperado
Após execução do IC o status_chamado deve ser alterado de "Reprovado" para "Em atendimento"

ATOR: Todos os atores exceto Cliente **IC 50: Criar tarefa**

➔ criar_t = id_chamado + id_pessoa + hh_tarefa

Descrição: criar tarefa definindo quem será a pessoa responsável por realizá-la, quanto de tempo é necessário e o tipo de ação esperada.
A pessoa pode ser qualquer um dos atores existentes no sistema, exceto o cliente.
Gerente pode criar tarefa para todos
Analista e Atendente podem criar tarefas para todos nos chamados em que são responsáveis, exceto para o Gerente
Todos podem criar tarefas para si

← id_tarefa

Aplicação de testes
Apenas os atores Cliente e Atendente devem poder executar este IC
Fluxo de entrada
id_chamado
(Particionamento de equivalência)
Condição de entrada, booleano: chamado deve ser informado.
- realizar ação sem identificar o chamado: sistema deve impedir a ação
- realizar a ação identificando o chamado com numero válido: ação deve ser realizada com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens
- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.
- inserir números de chamados que sejam do conjunto dos naturais não nulos: ação deve ser realizada com sucesso.

id_pessoa
(Particionamento de equivalência)
Condição de entrada, booleano: pessoa deve ser informada.
- criar tarefa sem identificar uma pessoa: sistema deve impedir a ação
- criar tarefa identificando uma pessoa: ação deve ser realizada com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens
- criar tarefa para tipo de pessoa "Cliente": sistema deve impedir a ação
- com ator Gerente deve ser possível criar tarefas para todas as pessoas exceto cliente

- com os atores: desenvolvedor, tester e inspetor, o sistema deve aceitar que sejam criadas tarefas somente para eles próprios
- com os atores Atendente e Analista deve ser possível criar tarefas para todas as pessoas dentro do chamado ao qual são responsáveis, exceto para o gerente
- atores podem criar tarefas para si

hh_tarefa

Condição de entrada, booleano

- criar tarefa sem definir seu hh: sistema deve impedir a ação, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer e enviar: ação deve ser realizada com sucesso

(Particionamento de equivalência)

Condição de entrada, intervalo: valores reais com uma casa decimal apenas

- inserir valores não numéricos: sistema deve impedir ação
- inserir valores numéricos: sistema deve aceitar formatando para uma casa decimal

(Análise do valor limite)

- informar valor zero: sistema deve bloquear a ação
- informar "0,1": sistema deve aceitar

Fluxo de saída

id_tarefa

- deve ser gerado um número natural maior do que zero para cada uma das tarefas criadas durante execução dos casos de testes acima
- os números gerados devem ser sempre crescentes, sem coincidência nem saltos na seqüência.

ATOR: Todos os atores exceto Cliente **IC 51: Concluir tarefa**

➔ concluir_t = id_tarefa + hh_gasto_tarefa + comentario

Descrição: Tanto a pessoa que foi definida para execução da tarefa quanto o Gerente, o Atendente e o Analista do chamado em que são responsáveis podem cancelar tarefas.

Aplicação de testes

Apenas os atores Gerência e a pessoa que atua como responsável pela tarefa podem executar este IC

Fluxo de entrada

id_tarefa

(Particionamento de equivalência)

Condição de entrada, booleano:

- realizar ação sem identificar a tarefa: sistema deve impedir a ação
- realizar a ação identificando a tarefa com numero válido: ação deve ser realizada com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens

- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.
- inserir números de tarefas que sejam do conjunto dos naturais não nulos: ação deve ser realizada com sucesso.

hh_gasto_tarefa

Condição de entrada, booleano

- Concluir tarefa sem informar hh gasto: sistema deve impedir a ação, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer: ação deve ser realizada com sucesso

(Particionamento de equivalência)
Condição de entrada, intervalo: valores reais com uma casa decimal apenas
- inserir valores não numéricos: sistema deve impedir ação
- inserir valores numéricos: sistema deve aceitar formatando para uma casa decimal

(Análise do valor limite)
- informar valor zero: sistema deve bloquear a ação
- informar "0,1": sistema deve aceitar

comentario
(Particionamento de equivalência)
Condição de entrada, booleano: comentário é de preenchimento opcional.
- realizar ação sem preencher o comentário: sistema deve realizar a ação desejada
- realizar ação preenchendo o comentário: sistema deve realizar a ação desejada

Condição de entrada, intervalo: como não foi determinado limite ou regras para este campo, qualquer entrada deve ser aceita, inclusive o conjunto vazio.
- realizar ação inserindo apenas 1 caractere no campo: sistema deve permitir a ação
- realizar ação com um texto que contenha tipos variados de caracteres: sistema deve permitir a ação respeitando os caracteres inseridos

ATOR: Todos os atores exceto Cliente **IC 52: Cancelar tarefa**

➔ **cancelar_t = id_ tarefa + comentario**

Descrição: as tarefas podem ser canceladas pela pessoa responsável por elas
Gerente pode cancelar a tarefa de qualquer ator
Analista e atendente de um chamado podem cancelar tarefas do chamado em questão, exceto se a tarefa for de um Gerente

Aplicação de testes
Todos os atores exceto cliente podem executar este IC

Fluxo de entrada

id_tarefa
(Particionamento de equivalência)
Condição de entrada, booleano:
- realizar ação sem identificar a tarefa: sistema deve impedir a ação
- realizar a ação identificando a tarefa com numero válido: ação deve ser realizada com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens
- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.
- inserir números de tarefas que sejam do conjunto dos naturais não nulos: ação deve ser realizada com sucesso.

justificativa
(Particionamento de equivalência)
Condição de entrada, booleano: justificativa deve ser informada.
- solicitar cancelamento sem preencher a justificativa: sistema deve solicitar preenchimento do campo
- solicitar informando a justificativa: cancelamento deve ser realizado com sucesso

Condição de entrada, intervalo: como não foi determinado limite ou regras para este campo, o conjunto inválido é o conjunto vazio e o válido qualquer conjunto de caracteres.
- realizar ação inserindo apenas 1 caractere no campo: sistema deve permitir a ação
- realizar ação com um texto que contenha tipos variados de caracteres: sistema deve permitir a ação respeitando os caracteres inseridos

ATOR: Gerente IC 53: Cadastrar pessoa

➔ cadastrar_pessoa = nome + tipo_pessoa

Descrição: Gerente poderá cadastrar pessoas para que atuem nos chamados de acordo com o tipo definido.

← id_pessoa

Aplicação de testes

Apenas o ato Gerência pode executar este IC

Fluxo de entrada

nome

(Particionamento de equivalência)

Condição de entrada, booleano:

- realizar ação sem preencher o nome: sistema deve realizar a ação desejada
- realizar ação preenchendo o nome: sistema deve realizar a ação desejada

Condição de entrada, intervalo: como não foi determinado limite ou regras para este campo, qualquer entrada deve ser aceita

- realizar ação inserindo apenas 1 caractere no campo: sistema deve permitir a ação
- realizar ação com um texto que contenha tipos variados de caracteres: sistema deve permitir a ação respeitando os caracteres inseridos

tipo_pessoa

(Particionamento de equivalência)

Condição de entrada, booleano

- cadastrar pessoa sem preencher o tipo: sistema deve impedir a ação, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer e proceder com o cadastro: ação deve ser realizada com sucesso

Condição de entrada, intervalo: sistema deve disponibilizar os valores válidos: “Analista”, “Atendente”, “Cliente”, “Desenvolvedor”, “Gerente” e “Tester”

- cadastrar pessoa escolhendo uma das opções oferecidas pelo sistema: ação deve ser realizada com sucesso.
- cadastrar pessoa inserindo valores diferentes dos oferecidos pelo sistema (seja por digitação ou colando texto copiado de outro local via área de transferência do sistema operacional): sistema deve impedir inserção de valores diferentes dos apresentados

Fluxo de saída

id_pessoa

- deve ser gerado um número natural maior do que zero para cada pessoa cadastrada
- os números gerados devem ser sempre crescentes, sem coincidência nem saltos na seqüência.

ATOR: Atendente **IC 54: Requalificar chamado**

➔ **chamado = id_chamado + urgencia + qualificacao_chamado**

Descrição: Em caso de necessidade o atendente poderá requalificar chamado a qualquer momento.

Propósito: Permite realizar a requalificação do chamado a qualquer momento por parte do atendente.

Aplicação de testes

Apenas o ato Atendente pode executar este IC

Fluxo de entrada

id_chamado

(Particionamento de equivalência)

Condição de entrada, booleano: chamado deve ser informado.

- realizar ação sem identificar o chamado: sistema deve impedir a ação
- realizar a ação identificando o chamado com numero válido: ação deve ser realizada com sucesso

Condição de entrada, intervalo: definir os intervalos com base nos seguintes itens

- inserir valores numéricos que não sejam do conjunto dos naturais: sistema deve impedir a ação.
- inserir números de chamados que sejam do conjunto dos naturais não nulos: ação deve ser realizada com sucesso.

urgencia

(Particionamento de equivalência)

Condição de entrada, booleano

- abrir chamado sem preencher opção de urgência: sistema deve impedir a criação do chamado, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer e proceder com a criação do chamado: chamado deve ser aberto com sucesso

Condição de entrada, intervalo: sistema deve disponibilizar os valores válidos: “Alta - Impeditiva”, “Alta”, “Media” e “Baixa”

- solicitar chamado escolhendo uma das opções oferecidas pelo sistema: chamado deve ser aberto som sucesso.
- solicitar chamado inserindo valores diferentes dos oferecidos pelo sistema (seja por digitação ou colando texto copiado de outro local via área de transferência do sistema operacional): sistema deve impedir inserção de valores diferentes dos apresentados

qualificacao_chamado

(Particionamento de equivalência)

Condição de entrada, booleano

- abrir chamado sem preencher opção de qualificação: sistema deve impedir a criação do chamado, exigindo que o campo seja preenchido
- preencher com um valor válido qualquer e proceder com a criação do chamado: chamado deve ser aberto com sucesso

Condição de entrada, intervalo: sistema deve disponibilizar os valores válidos: “Bug” e “Melhoria”

- solicitar chamado escolhendo uma das opções oferecidas pelo sistema: chamado deve ser aberto som sucesso.
- solicitar chamado inserindo valores diferentes dos oferecidos pelo sistema (seja por digitação ou colando texto copiado de outro local via área de transferência do sistema operacional): sistema deve impedir inserção de valores diferentes dos apresentados