UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Gustavo Bestetti Ibarra

FULL USE CASE SIZE (FUCS): ESTIMATIVA DE SOFTWARE COM BASE NO TAMANHO DE CASOS DE USO

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Santa Catarina para a obtenção do Grau Mestre em Ciência da Computação
Orientadora: Prof[a]. Dr[a]. Patrícia Vilain

Florianópolis

2011

Catalogação na fonte pela Biblioteca Universitária
da
Universidade Federal de Santa Catarina

I12f Ibarra, Gustavo Bestetti
Full use case size (FUCS) [dissertação] : estimativa de software com base no tamanho de casos de uso / Gustavo Bestetti Ibarra ; orientadora, Patrícia Vilain. – Florianópolis, SC, 2011.
193 p.: il., grafs., tabs.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Ciência da Computação.

Inclui referências

1. Informática. 2. Ciência da computação. 3. Software – Avaliação. 4. Software - Desenvolvimento. I. Vilain, Patrícia. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. III. Título.

CDU 681

Gustavo Bestetti Ibarra

FULL USE CASE SIZE (FUCS): ESTIMATIVA DE SOFTWARE COM BASE NO TAMANHO DE CASOS DE USO

Esta Dissertação foi julgada adequada para obtenção do Título de "Mestre em Ciência da Computação", e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Santa Catarina.

Local, 29 de agosto de 2011.

_____

Prof. Dr. Mário Antônio Ribeiro Dantas
Coordenador do Curso

Banca Examinadora:

_____

Profª., Drª. Patrícia Vilain
Orientadora
Universidade Federal de Santa Catarina

_____

Prof., Dr. Guilherme Horta Travassos
Universidade Federal do Rio de Janeiro

_____

Prof., Dr. Raul Sidnei Wazlawick
Universidade Federal de Santa Catarina

_____

Prof., Dr. Ricardo Pereira e Silva
Universidade Federal de Santa Catarina

Dedico este trabalho a Deus, minha esposa e meus filhos, meus pais, minha irmã, aos meus amigos e a minha orientadora por todo apoio, força, incentivo, companheirismo, paciência e amizade. Sem eles nada disso seria possível.

AGRADECIMENTOS

Gostaria de agradecer a todas as pessoas que ao longo do mestrado me apoiaram em vários momentos.

A minha esposa Sinara pela sua dedicação aos nossos filhos Ian e Kauã, pelas cobranças, incentivos e palavras de confiança. Meu amor, muito obrigado!

Ao meu filho Ian, que nos "altos" dos seus 7 anos, teve a maturidade de um adulto e toda a compreensão nos momentos que não pude me dedicar a ele. Que rezou por mim e que me incentivava a cada página escrita deste trabalho. Eu te amo meu filho! Você é demais!

Ao meu filho Kauã, que veio ao mundo no meio deste trabalho, me renovando e dando mais forças para completar mais esta etapa.

Aos meus pais Leo e Rejane que me ensinaram a seguir sempre estudando e que tiveram toda compreensão pela minha ausência.

Aos pais da minha esposa, Iara e Luiz, que me ajudaram demais, principalmente nos finais de semana, cuidando dos meus filhos para que eu pudesse me dedicar a este trabalho.

Aos padrinhos do Kauã, Zélia e José Lino, que também adotaram o Ian como afilhado, e que sempre estavam dispostos a me ajudar.

Aos meus amigos do grupo de casais Sagrada Família por todas as palavras de incentivo e por todas as orações.

Aos meus colegas de trabalho por acreditar no meu trabalho e me ajudarem a aplicá-lo no nosso dia a dia.

Aos professores Guilherme Horta Travassos, Raul Sidnei Wazlawick e Ricardo Pereira e Silva, membros da banca avaliadora deste trabalho, pelas contribuições a este trabalho.

De forma especial, agradeço a minha orientadora, Patrícia Vilain, por me aceitar como orientando, por acreditar na minha idéia e me ajudar a torná-la realidade, com toda sua dedicação e paciência.

Sem estas pessoas, certamente este trabalho não seria possível.

“Se você faz o que sempre fez, você obterá o que você sempre obteve”.

Anthony Robbins

RESUMO

Este trabalho propõe um novo método para medir e estimar software com base no tamanho de cada caso de uso chamado FUCS (*Full Use Case Size*). O principal objetivo do método é definir uma forma mais objetiva para medição e estimativa de software que seja baseado em casos de uso, e que possibilite sua aplicação em processos licitatórios e de contratação e terceirização de serviços de desenvolvimento de software em empresas públicas. O método proposto utiliza atributos técnicos que podem ser medidos a partir de uma breve descrição de casos de uso e requisitos de sistema relacionados. O tamanho de cada caso de uso (UC_S - *Use Case Size*) do sistema é derivado de três atributos técnicos: a quantidade de pontos por caso de uso (UC_{TP} - *Use Case Type Points*), o número de operações de sistema e regras de negócio (N_{SOBR} - *Number of System Operations and Business Rules*) e o número de requisitos de interface gráfica do usuário (N_{GUIR} – *Number of Graphical User Interface Requirement*). O tamanho total do sistema (S_{size}) é dado pela soma dos tamanhos de casos de uso (UC_S).

Palavras-chave: estimativa de software, medição de software, ponto de caso de uso, ponto de função, licitação e terceirização do desenvolvimento de software.

ABSTRACT

This work proposes a new method for measuring and estimating software based on the size of each use case called FUCS (Full Use Case Size). The main objective of the method is to define a more objective measurement and estimating software that is based on use cases, and which enables its application in bidding processes and contracting and outsourcing of software development in public companies. The proposed method uses technical attributes that can be measured from a brief description of use cases and related system requirements. The size of each use case ($\boldsymbol{UC_S}$ - *Use Case Size*) of the system is derived from three technical attributes: the amount of points per use case ($\boldsymbol{UC_{TP}}$ - *Use Case Points Type*), the number of system operations and business rules ($\boldsymbol{N_{SOBR}}$ - *Number of System Operations and Business Rules*) and the number of requirements for graphical user interface ($\boldsymbol{N_{GUIR}}$ - *Number of Graphical User Interface Requirement*). The total size of the system ($\boldsymbol{S_{size}}$) is the sum of the sizes of the use cases ($\boldsymbol{UC_S}$).

Keywords: estimating software, measuring software, use case point, function point, bidding and outsourcing of software development.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AIE – Arquivo de Interface Externa
ALI – Arquivo Lógico Interno
APF – Análise de Pontos de Função
AR – Arquivo Referenciado
ARP – Ata de Registro de Preços
BFPUG – Brazilian Function Point Users Group
CE – Consulta Externa
CFP – COSMIC Function Point
COSMIC – Common Software Measurement International Consortium
CP – Sistema de Controle de Ponto
CPM – Counting Pratices Manual
CRUD – Abreviatura dos termos *Create, Retrieve, Update, Delete*
EE – Entrada Externa
FFP – Full Function Point
FP – Function Points
FUCS – Full Use Case Size
FUSP – Fuzzy Use Case Size Point
IFPUG – International Function Point Users Group
IN – Instrução Normativa
MKII – Método Mark II
OS – Operação de Sistema
PCU – Pontos de Caso de Uso
PCUNA – Pontos de Caso de Uso Não Ajustados
PFNA – Pontos de Função Não Ajustados
PROAD – Sistema de Processos Administrativos do TRT/SC
RIGU – Requisito de Interface Gráfica do Usuário
RN – Regra de Negócio
RP – Registro de Preços
SE – Saída Externa
SISFAT – Sistema de Faturamento
SLTI – Secretaria de Logística e Tecnologia da Informação
SRP – Sistema de Registro de Preços
TCU – Tribunal de Contas da União
TD – Tipo de Dados
TI – Tecnologia da Informação
TNC – Empresa de Informática onde foi realizado um estudo de caso
TR – Tipo de Registro
TRT/SC – Tribunal Regional do Trabalho de Santa Catarina

TST – Tribunal Superior do Trabalho
UC – Use Case
UCS –Use Case Size (unidade de medida do método FUCS)
UI – User Interface
UKSMA – United Kingdon Software Metrics Association
USP – Use Case Size Point

LISTA DE SÍMBOLOS

UC_{TP}	Atributo técnico do método FUCS – Use Case Type Point
α	Variável de configuração do método FUCS – Peso das operações de sistema regras de negócio
β	Variável de configuração do método FUCS – Peso dos requisitos de interface gráfica do usuário
N_{SOBR}	Atributo técnico do método FUCS – *Number of System Operation and Business Rules*
N_{GUIR}	Atributo técnico do método FUCS – *Number of Graphical User Interface Requirement*
S_{size}	Resultado da medida do tamanho do sistema com o método FUCS – *System Size*
S_{effort}	Estimativa de esforço para desenvolvimento de um sistema com base no do tamanho do sistema medido com o método FUCS – *System Effort*

SUMÁRIO

1 INTRODUÇÃO

Os processos de medição e estimativa de software são atividades essenciais no gerenciamento e planejamento de projetos (PRESSMAN, 2009), pois através deles podem-se derivar estimativas de esforço, custos e prazos, auxiliando inclusive a análise de viabilidade de um projeto. Além disso, sua aplicação em processos de terceirização do desenvolvimento de software é de fundamental importância, pois, quando utilizado em conjunto com contratos, estabelece uma ferramenta com critérios objetivos para o controle dos custos dos projetos.

A terceirização é hoje um desafio para as empresas públicas e privadas brasileiras que buscam reduzir seus custos de produção e aumentar sua produtividade e lucros. Segundo Sérgio Pinto Martins:

> "No Brasil, a noção de terceirização foi trazida por multinacionais por volta de 1950, pelo interesse que tinham em se preocupar apenas com a essência do seu negócio. A indústria automobilística é exemplo de terceirização, ao contratar a prestação de serviços de terceiros para a produção de componentes do automóvel, reunindo peças fabricadas por aqueles e procedendo à montagem final do veículo". (MARTINS, 2009)

A exemplo da indústria automobilística, tanto as empresas públicas quanto as empresas privadas estão caminhando para a terceirização de serviços de desenvolvimento de software.

No caso das empresas privadas, esta medida justifica-se principalmente em função dos altos encargos incidentes sobre a folha de pagamento e na complexidade da legislação trabalhista. Já no caso das empresas públicas, a terceirização é quase que uma necessidade institucional, pois com as políticas de Estado Mínimo estas empresas estão sofrendo fortes pressões para reduzir gastos e pessoal (ALVES, 2002), e na maioria das vezes estas reduções afetam as áreas “meio” destas empresas, e consequentemente as áreas de Tecnologia da Informação (TI).

Considerando o cenário público, as terceirizações apresentam maior complexidade, pois só podem ser feitas através de processos licitatórios, os quais são regidos por lei (BRASIL, 1993) e são pouco flexíveis. Neste cenário, os processos de medição e estimativa tornam-se

ainda mais importantes em função das recomendações do TCU (Tribunal de Contas da União), que condenam a contratação por horas trabalhadas (TCU, 2006).

A necessidade de se estabelecer um processo de medição de software em processos licitatórios para uso em contratos com empresas públicas ficou ainda mais evidente depois da publicação da Instrução Normativa Nº 04/2010 da Secretaria de Logística e Tecnologia da Informação do Governo Federal (SLTI, 2008), que dispõe sobre o processo de contratação de serviços de Tecnologia da Informação pela Administração Pública Federal. Nesta regulamentação, ficou definido que um processo de contratação de serviços de TI deve explicitar, dentre outras informações, os procedimentos acerca dos métodos e métricas utilizados para mensuração dos serviços prestados.

Diversos métodos de medição e estimativa de software já foram propostos, como Linhas de Código, Análise de Pontos de Função (APF) (ALBRECHT, 1979), Pontos de Caso de Uso (PCU) (KARNER, 1993), Planning Poker, Story Points e Ideal Days (COHN, 2005). Linhas de Código foi um dos primeiros métodos, porém depende fortemente das características dos desenvolvedores e das linguagens de programação. Pontos de Função é um método que foi desenvolvido em 1979 por Allan Albrectht e é atualmente um dos métodos mais utilizados no processo de medição de projetos de software. No entanto, além de sua aplicação ser complexa (LAVAZZA, BIANCO e CARAVAGLIA, 2008), apresenta algumas limitações como a correlação entre os elementos envolvidos nos cálculos das estimativas (KITCHENHAM e KÄNSÄLÄ, 1993) (KITCHENHAM, 1997) (LOKAN, 1999). O método de medição de Pontos de Caso de Uso (KARNER, 1993) foi proposto por Gustav Karner como uma adaptação do método de Pontos de Função (VAZQUEZ, SIMÕES e ALBERT, 2010) para aplicação em projetos orientados a objetos a partir de casos de uso levantados na fase de requisitos (VIEIRA, 2007). No entanto, para a análise da complexidade dos casos de uso, as definições propostas por Karner são muito dependentes da forma como os casos de uso são escritos e detalhados (ANDA, DREIEM, *et al.*, 2001) (VIEIRA e WAZLAWICK, 2006) (VIEIRA, 2007). Já as técnicas utilizadas em métodos ágeis, como Planning Poker, Story Points e Ideal Days são muito subjetivas e dependem do “sentimento” da equipe de desenvolvimento do sistema em relação à funcionalidade a ser desenvolvida (OSTVOLD, HAUGEN e BENESTAD, 2008). Todos estes são fatores que dificultam a aplicação destes métodos em processos licitatórios e contratos de terceirização de serviços, e em especial em contratos com empresas

públicas, onde os procedimentos e critérios de medição devem estar explicitamente descritos nos contratos.

Este trabalho apresenta uma proposta de um novo método de estimativa de projetos de software baseado em casos de uso – o FUCS (*Full Use Case Size*). Para utilização do FUCS os casos de uso não precisam estar totalmente detalhados, permitindo assim sua aplicação nas fases iniciais dos projetos de software. Estes fatores facilitam sua utilização em processos licitatórios e em contratos de terceirização do desenvolvimento de sistemas, especialmente com empresas públicas, onde na grande maioria das vezes existe a necessidade de formalização de contratos em momentos em que os requisitos ainda não estão totalmente detalhados.

O FUCS foi desenvolvido, aplicado e validado ao longo de 7 projetos de software desenvolvidos no Tribunal Regional do Trabalho de Santa Catarina (TRT/SC). Desde então, este método tem sido utilizado em processos licitatórios e em contratação de serviços de desenvolvimento de software daquele Tribunal.

1.1 MOTIVAÇÃO

A terceirização do desenvolvimento de software atualmente é uma realidade para muitas empresas, tanto públicas quanto privadas, que desejam focar seus esforços na essência de seu negócio.

No Brasil os processos de terceirização em empresas públicas devem ser realizados seguindo a Lei 8666/93 (BRASIL, 1993) e outras regulamentações que instituem normas e procedimentos para licitações e contratos da Administração Pública (TCU, 2006) (SLTI, 2008).

O cuidado com licitações e contratos cujo objeto esteja relacionado com serviços de TI tem aumentado cada vez mais nas agências governamentais regulamentadoras, principalmente em função do alto grau de subjetividade[1] dos serviços relacionados a estas áreas. Como tentativa de regular as contratações nestas áreas, a Secretaria de Logística e Tecnologia da Informação do Governo publicou em 2010 a Instrução Normativa nº 04 (SLTI, 2008) que, dentre outras coisas, determina que um processo de contratação de serviços de TI deve explicitar os procedimentos acerca dos métodos e métricas utilizados para mensuração dos serviços prestados.

[1] A subjetividade a qual este texto faz referência está relacionada às diversas formas de execução dos serviços de TI, principalmente em relação à interpretação e implementação de requisitos de software.

Os principais métodos e técnicas de estimativas e medição de software apresentam algum tipo de limitação que dificultam sua aplicação em processos licitatórios e contratações, principalmente em função da grande variação dos resultados das medições quando realizadas por pessoas diferentes ou da complexidade que apresentam para sua aplicação.

No Brasil existe uma tendência de padronização da forma de contratação e medição dos produtos de software utilizando o método de Análise de Pontos de Função em virtude da variedade de normativas e regulamentações nacionais descritas no portal do governo eletrônico e também pelo fato do método de pontos de função contar com uma organização responsável pela sua padronização e evolução que é o IFPUG (*International Function Point Users Group*), que também é responsável por manter o Manual de Práticas de Contagem (*CPM – Counting Pratices Manual*).

O Roteiro de Métricas do Sistema de Administração dos Recursos de Informação e Informática (SISP), publicado no portal do governo eletrônico brasileiro (SLTI, 2010), apresenta uma série de regras complementares às estabelecidas no manual de contagem de pontos de função (IFPUG, 2010) com o propósito de cobrir algumas lacunas do método de pontos de função e possibilitar seu uso de uma forma mais objetiva, em contratos de prestação de serviços de desenvolvimento e manutenção de sistemas. Este roteiro é uma das iniciativas que contribuem para a padronização da forma de contratação do desenvolvimento de sistemas utilizando Pontos de Função no âmbito do serviço público. Entretanto, esta padronização deixa de levar em conta um fundamento essencial de um processo de medição, que é ser orientado ao seu objetivo.

Outra característica do processo de medição em pontos de função é seu foco voltado aos requisitos funcionais de um software sempre sob a ótica dos usuários. Com isso, características não funcionais de um projeto como padrões de desenvolvimento, requisitos de interface gráfica e regras de negócio não são levadas em consideração no processo de medição. Porém, na grande maioria das vezes o objetivo de um contrato para desenvolvimento de um software não considera como produto final somente um software instalável, mas também o código fonte juntamente com toda documentação e com todos os requisitos de qualidade e padrões estabelecidos pela empresa. Esta medida é comumente adotada por empresas como uma forma de manter o domínio de conhecimento sobre os produtos desenvolvidos, possibilitando assim sua evolução com independência dos fornecedores.

Outro ponto a ser considerado é que os processos de contratação de serviços de TI em empresas públicas são sempre realizados através de licitações que buscam classificar a empresa fornecedora que apresentar a proposta mais vantajosa, comumente expressa pelo menor preço. No entanto, a formação de custo para desenvolvimento de um projeto medido em pontos de função passa obrigatoriamente por uma estimativa de esforço para desenvolvimento de uma unidade de ponto de função, que, por sua vez, certamente deverá levar em consideração aspectos não funcionais dos projetos de desenvolvimento de software, que na grande maioria das vezes não estão disponíveis nos editais de licitações. Desta forma, os custos são formados considerando uma determinada margem de risco com o intuito de prevenir situações onde a complexidade de desenvolvimento seja alta. Por consequência, em algumas situações onde o software a ser desenvolvido apresente características menos complexas, o custo para seu desenvolvimento pode se tornar muito elevado, o que fere o princípio da economicidade, que nada mais traduz do que o dever de eficiência do administrador na gestão do dinheiro público.

A motivação do desenvolvimento do FUCS está na necessidade de estabelecer uma forma mais objetiva de mensuração dos projetos de software que levem em consideração aspectos funcionais e não-funcionais dos projetos, possibilitando assim que o esforço para desenvolvimento de um projeto possa ser estimado de uma forma mais direta e que o processo de formação de custo para desenvolvimento de um projeto tenha um menor fator de risco e, consequentemente, um custo mais apropriado e consistente. Desta forma, o uso do FUCS torna os processos licitatórios e contratações mais eficientes sob o ponto de vista do controle dos custos dos projetos.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

O objetivo principal deste trabalho consiste em definir um novo método de estimativa do tamanho de projetos de softwares baseados em casos de uso, que possibilite sua aplicação em processos de contratação e terceirização de desenvolvimento de software em empresas públicas.

1.2.2 Objetivos Específicos

Os principais objetivos específicos deste trabalho são:

1. Definir os atributos relacionados aos casos de uso que serão considerados no método de medição;
2. Definir um método de mapeamento e medição dos casos de uso com base nos atributos técnicos definidos;
3. Definir uma forma de customização do método para contextos diferentes;
4. Aplicar o método em projetos reais de desenvolvimento de software e avaliar os resultados.

1.3 MÉTODO DE TRABALHO

Inicialmente foram estudados os principais métodos de medição e estimativa de software utilizados pelo mercado atualmente. Em seguida, e com base em experiências do dia a dia, foi definida uma hipótese de pesquisa, a qual está definida na seção a seguir. Com base na hipótese, foi definido um método de medição, descrito em (IBARRA e VILAIN, 2010a) e (IBARRA e VILAIN, 2010b), que foi aplicado em alguns projetos desenvolvidos no Tribunal Regional do Trabalho da 12ª Região (TRT/SC). Após a aplicação do método foi realizado uma análise exploratória dos dados coletados ao longo da aplicação do método. Com base nos resultados obtidos, foi definida uma nova versão do método, o qual foi reaplicado aos projetos já desenvolvidos e a novos projetos com o intuito de verificar sua consistência. Esta nova versão do método foi apresentada em (IBARRA e VILAIN, 2010c) e está descrita em detalhes ao longo deste trabalho e em seus anexos.

1 4 HIPÓTESE

Este trabalho tem como hipótese que o esforço para desenvolvimento de um software pode ser derivado através da medição e estimativa dos tamanhos de cada caso de uso, sendo que o tamanho de cada caso de uso pode ser obtido através da análise e classificação dos seus requisitos. Desta forma, espera-se gerar estimativas menos discrepantes do que os métodos atuais de medição de software.

1.4 RESULTADOS ESPERADOS

O principal resultado esperado com a realização do estudo descrito neste trabalho é a definição de um método de estimativas e medição do tamanho de projetos de software que possa ser aplicado com maior eficácia em processos licitatórios e contratações em empresas públicas. O termo “eficácia” é aqui aplicado sob a ótica de um maior equilíbrio entre o esforço necessário para desenvolvimento de um sistema em relação ao seu tamanho medido através da aplicação do método proposto.

Além disso, é esperado que o método possa ser aplicado em empresas diferentes, permitindo seu ajuste em função dos processos e padrões de desenvolvimento definidos em cada empresa. É esperado que, uma vez ajustado aos processos e padrões das empresas, este método produza estimativas de esforço com margem de erro inferior a 20%.

1.5 LIMITAÇÕES

O escopo principal deste trabalho é de apresentar um método de estimativa e medição de projetos de software baseado em casos de uso. Desta forma, é aplicável somente a empresas que trabalham com casos de uso em seu processo de desenvolvimento, principalmente na fase de concepção do projeto. Apesar de o método proposto permitir seu ajuste em função dos processos e padrões de desenvolvimento de cada empresa, não é foco deste trabalho abordar extensivamente conteúdos referentes a processos e padrões de desenvolvimento.

O método descrito neste trabalho foi definido e refinado ao longo de três anos de sua aplicação no TRT/SC. Além deste Tribunal, foi aplicado a apenas um projeto no Tribunal Superior do Trabalho e a um projeto em uma empresa privada.

Além disso, o espaço amostral de dados sob o qual o método foi definido e refinado é relativamente pequeno em virtude da grande dificuldade na coleta de dados decorrente do tempo que projetos de software levam para serem concluídos.

1.6 ESTRUTURA DE TRABALHO

O capítulo 2 contempla informações relacionadas aos processos licitatórios, bem como conceitos, métodos e métricas relacionados a processos de medição e estimativa de software. O capítulo 3 apresenta trabalhos que também propõem métodos e métricas para medição e estimativa de software. O capítulo 4 apresenta em detalhes as principais características do método de medição e estimativa baseado em casos de uso proposto neste trabalho. O capítulo 5 descreve a aplicação do método em alguns projetos e seus resultados. No capítulo 6 estão expostas as conclusões finais deste trabalho juntamente com os possíveis trabalhos futuros a serem realizados.

2 REVISÃO BIBLIOGRÁFICA

Neste capítulo são apresentadas as principais informações relacionadas aos processos licitatórios, bem como os conceitos, métodos e métricas relacionadas a processos de medição e estimativa de software.

2.1 LICITAÇÕES E CONTRATOS

As contratações para execução de obras, de serviços, compras, alienações e locações no âmbito das instituições públicas brasileiras, sejam elas municipais, estaduais ou federais, são regulamentadas pela Lei 8666/93 (BRASIL, 1993) e obrigatoriamente devem ser precedidas de licitação (salvas exceções de baixo valor ou inegibilidade[2]).

A licitação é um procedimento previsto nos artigos 22, inciso XXVII, e 37, inciso XXI, ambos da Constituição Federal, que visa escolher a melhor proposta dentre as apresentadas por aqueles que desejam estabelecer contratos com o poder público.

Segundo o Tribunal de Contas da União, uma licitação pode ser definida como:

> "um procedimento administrativo formal em que a Administração Pública convoca, mediante condições estabelecidas em ato próprio (edital ou convite), empresas interessadas na apresentação de propostas para o oferecimento de bens e serviços." (TCU, 2011)

A licitação também objetiva garantir o cumprimento do princípio da isonomia, expresso na Constituição Federal Brasileira como a atuação do poder público de forma igualitária e sem distinção de pessoas, de forma objetiva e justa. Além disso, visa o princípio da economicidade, através da seleção da proposta mais vantajosa para a Administração Pública.

[2] Inegibilidade é um termo usado em processos licitatórios para fazer referência à contratação, aquisição de determinado serviço ou produto com a dispensa de procedimentos licitatórios.

A licitação, portanto, tem por objetivo permitir que a Administração Pública contrate aqueles que reúnam as condições necessárias para o atendimento do interesse público, considerando aspectos relacionados à capacidade técnica e econômico-financeira do licitante, à qualidade do produto e ao valor do objeto, selecionando, portanto, a alternativa mais vantajosa para a Administração Pública (WIKIBOOKS).

As licitações podem ser realizadas nas seguintes modalidades:

- Pregão, eletrônico ou presencial;
- Concorrência;
- Tomada de preços;
- Convite;
- Concurso;
- Leilão.

Como o foco deste trabalho está relacionado à contratação de serviços para desenvolvimento de sistemas em empresas públicas, esta seção apresentará apenas informações relacionadas às modalidades de pregão, pois a contratação destes tipos de serviços comumente é realizada por esta modalidade.

O pregão, seja presencial ou eletrônico, é a modalidade de licitação em que a disputa pelo fornecimento é feita em uma sessão pública, por meio de propostas e lances, para classificação e habilitação do fornecedor que apresentar a proposta de menor preço, desde que atendidas as condições estabelecidas no edital da licitação.

Comumente, principalmente na área de Tecnologia da Informação, as licitações são realizadas na modalidade de Pregão e utilizando um Sistema de Registro de Preços (SRP).

O SRP é o conjunto de procedimentos para registro formal de preços relativos à prestação de serviços e aquisição de bens, para contratações futuras, permitindo aquisições compartilhadas entre órgãos e entidades da Administração. Com o uso de um SRP o preço fica registrado em uma Ata de Registro de Preços (ARP) por um determinado tempo (definido no edital) e, enquanto a ARP estiver vigente, o órgão licitante (ou quem ele permitir) poderá aderir à ARP e estabelecer contratos com o fornecedor vencedor.

O pregão, acompanhado de SRP, tem sido muito utilizado nas licitações em função do dinamismo que oferece às empresas licitantes com a eliminação de licitações contínuas de bens e serviços semelhantes. Outra vantagem da utilização de um SRP está na facilidade

que ele traz para a gestão orçamentária dos recursos. Como não há compromisso de aquisição por parte da licitante, não há necessidade de reserva prévia dos recursos na fase da licitação, o qual só será necessário no momento da celebração do contrato. Assim, o administrador poderá optar pela alocação dos recursos em necessidades administrativas mais urgentes e alinhadas à estratégia do negócio.

Em relação aos processos licitatórios na área de Tecnologia da Informação (TI), várias regulamentações têm sido editadas. A Instrução Normativa nº 04 (SLTI, 2008), estabelece alguns critérios a serem seguidos nos processos licitatórios para contratação de bens ou serviços relacionados a TI. Um dos critérios estabelecidos nesta normativa, expressa que os editais para estes tipos de contratações deverão conter, dentre outras informações, os procedimentos acerca dos métodos e métricas utilizados para mensuração dos serviços prestados.

É com este foco que o método FUCS é apresentado. Seu principal objetivo é definir um conjunto de procedimentos (método) e métricas para mensuração dos serviços de desenvolvimento de softwares baseado em casos de uso.

2.2 TIPOS DE PROJETOS DE SOFTWARE

Observa-se no mundo real uma grande variedade de tipos de projetos de software e suas respectivas áreas de especificidades. Ocorre que para cada tipo de projeto de software podem ser necessárias atividades de gerenciamento diferenciadas, que levem em consideração características particulares de cada tipo. Isso também ocorre com os processos de medição e estimativa. Para uma melhor contextualização, a Tabela 1 classifica alguns tipos de projetos de software.

O método FUCS, proposto neste trabalho, foi avaliado apenas em projetos de sistema de informação. No entanto, seu uso não está limitado a estes tipos de projeto.

Tabela 1 - Categorias de software

Categoria	Descrição
Software Básico	Softwares de infraestrutura que fornecem apoio a outras aplicações. São caracterizados por forte interação com o hardware, acesso por múltiplos usuários concorrentes e compartilhamento de recursos do sistema. Exemplos: sistemas operacionais, compiladores, drivers e processadores de telecomunicações.
Software de Tempo Real	Softwares que monitoram e respondem a eventos do mundo real de forma instantânea. Exemplos: sistemas de controle aéreo, sistemas de controle de caldeiras, sistemas de telefonia etc.
Software de Sistemas de Informação	Softwares que manipulam dados estruturados, frequentemente acessados e mantidos em banco de dados para responder a atividades empresariais e governamentais. Exemplos: sistemas de folha de pagamento, contas a pagar e receber, controle de estoque etc.
Software Científicos e de Engenharia	Softwares caracterizados alta complexidade algorítmica e processamento de números. Exemplos: aplicações de astronomia, fadiga mecânica de automóveis, biologia molecular etc.
Software Embarcados	Softwares responsáveis por controlar produtos inteligentes tanto da área industrial quanto de consumo. Exemplos: controle de forno micro-ondas, funções digitais em automóveis etc.
Software de Escritório	Softwares que realizam processamento de textos, planilhas eletrônicas, apresentações etc.
Software de Inteligência Artificial	Softwares que fazem uso de algoritmos não numéricos para resolver problemas complexos que não são favoráveis à computação ou à análise direta. Exemplos: sistemas especialistas, reconhecimento de padrões, jogos, demonstração de teoremas etc.

Fonte: (PRESSMAN, 2009)

2.3 CASOS DE USO

O desenvolvimento de sistemas é um processo de construção de modelos. No desenvolvimento de sistemas orientados a objetos os modelos tipicamente iniciam com um modelo de requisitos (JACOBSON, 1995).

Os primeiros modelos do sistema devem ser compreensíveis às pessoas de dentro e de fora da organização e devem descrever o sistema sob o ponto de vista externo, como uma caixa-preta (JACOBSON, 1995). Os casos de uso são uma forma de estruturar e descrever esta caixa-preta. De uma maneira mais formal, um caso de uso pode ser descrito como:

> "um caso de uso é uma descrição narrativa de uma sequência de eventos que ocorre quando um ator (agente externo) usa um sistema para realizar uma tarefa". (JACOBSON, BOOCH e RUMBAUGH, 1999)

Kruchten (2004) complementa o conceito enfatizando o valor do resultado esperado de um caso de uso:

> "Um caso de uso é uma sequência de ações que um sistema executa e que produz um resultado de valor para um ou mais atores". (KRUCHTEN, 2004)

Em uma linguagem mais prática e direta, Cockburn define que:

> "Casos de uso são fundamentalmente uma forma textual, embora possam ser escritos usando fluxogramas, diagramas de sequência, redes de Petri ou linguagens de programação. Sob circunstâncias normais, eles servem como um meio de comunicação entre uma pessoa e outra, muitas vezes entre pessoas sem treinamento especial". (COCKBURN, 2005)

De forma geral, um caso de uso descreve uma sequência de passos ou operações que ocorrem durante a interação entre ator e sistema de forma a realizar uma tarefa em particular ou atingir um determinado objetivo. Neste cenário, um ator representa qualquer elemento externo que interage com o sistema.

Os casos de uso, quando corretamente escritos, descrevem os requisitos funcionais do sistema, ou seja, o que o sistema deve fazer. Porém não correspondem à totalidade dos requisitos de um sistema, pois eles não especificam (e nem devem) interfaces externas, formatos de dados, regras de negócio ou fórmulas complexas.

2.3.1 Modelos de Casos de Uso

Uma forma visual de se descrever um sistema pode ser feita através de diagramas de casos de uso que mostram as relações entre os atores e funcionalidades de um sistema. O termo modelo de casos de uso é aplicado ao conjunto do diagrama e a especificação dos casos de uso que o compõe.

Nos diagramas de casos de uso os atores são representados por bonecos esquemáticos da figura humana, enquanto os casos de uso são representados por elipses, como mostrado na Figura 1.

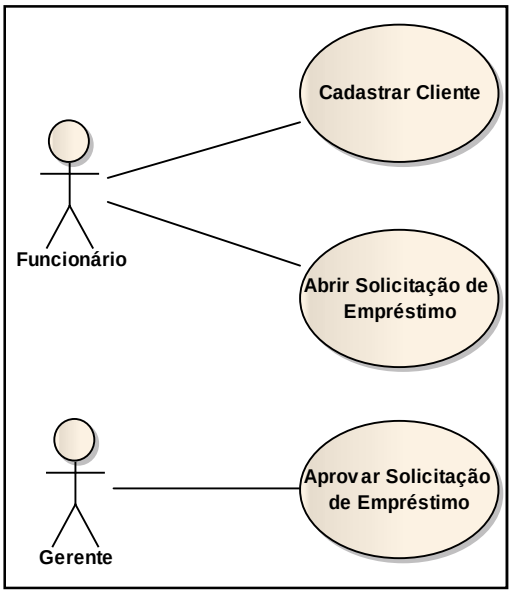

Figura 1 - Exemplo de Diagrama de Casos de Uso

2.3.2 Especificação de Casos de Uso

Os casos de uso podem ser especificados e detalhados em objetivos e interações, de uma forma mais refinada ou mais granular. Quanto ao nível de detalhamento, os casos de uso geralmente iniciam em um nível de detalhamento alto até chegar à sua forma mais detalhada, os casos de uso expandidos.

Os casos de uso de alto nível descrevem sucintamente um processo ou tarefa do negócio com foco no objetivo e no funcionamento do caso de uso. Já os casos de uso expandidos mostram mais detalhes do que os casos de uso de alto nível. Geralmente são detalhados em uma sequência de passos capaz de incluir todas as interações possíveis de serem identificadas pelos analistas (WAZLAWICK, 2004).

Cockburn (2005) aprofunda um pouco mais a granularidade do detalhamento dos casos de uso, definindo os seguintes níveis de detalhamento:

Nível de Resumo: são casos de uso de alto nível que envolvem múltiplos objetivos do usuário. A descrição destes tipos de casos de uso deve servir para três propósitos básicos: contextualizar os objetivos dos usuários, sequenciar os objetivos que possuem relação entre si e agrupar os casos de uso de nível mais baixo.

Nível de Usuário: são casos de uso expandidos que descrevem, em nível de objetivos do usuário, como uma tarefa ou um processo de negócio é realizado por um usuário ao interagir com o sistema, para atingir um objetivo do usuário.

Nível de Subfunção: são os casos de uso utilizados para representar um sub-objetivo ou passo em um cenário, abaixo do nível principal de interesse de usuários. De forma geral, este tipo de objetivo de caso de uso é normalmente utilizado por muitos outros casos de uso em nível de objetivo de usuário. Geralmente representam subfunções ou componentes.

Para a utilização do método descrito neste trabalho, os casos de uso não precisam estar detalhados, permitindo assim sua aplicação nas fases iniciais dos projetos. Desta forma, os casos de uso descritos em alto nível são suficientes para a utilização do método proposto neste trabalho.

2.3.3 Tipos de Caso de Uso

Apesar de ainda não haver um consenso entre os autores que escrevem sobre casos de uso, na prática existe uma série de casos de uso que são muito similares entre si. Cockburn (COCKBURN, 2005) apresenta dois tipos especiais de casos de casos de uso e que serão descritos a seguir.

2.3.3.1 Casos de Uso CRUD

Casos de uso do tipo CRUD representam aqueles pequenos casos de uso do tipo *Criar um <objeto>*, *Recuperar um <objeto>*, *Atualizar um <objeto>* e *Excluir um <objeto>* que, em última instância, compõem um caso de uso maior do tipo *Gerenciar um<objeto>* (COCKBURN, 2005). Estes tipos de casos de uso são chamados de CRUD em função da abreviação das iniciais dos termos em inglês *Create*, *Retrieve*, *Update* e *Delete*.

Como uma boa prática para não proliferação de uma série de documentos de especificação de casos de uso praticamente idênticos, pode-se definir uma especificação padrão para um caso de uso do tipo *Gerenciar um <objeto>*, e então utilizá-lo para todos os casos de uso que são similares (COCKBURN, 2005).

Quando não há qualquer alteração em relação à especificação padrão, basta referenciar o caso de uso da especificação padrão sem a necessidade de produzir novos documentos. Caso haja a necessidade de modificar algum comportamento específico, pode-se fazê-lo através da especificação por exceção, ou seja, basta referenciar o caso de uso modelo e especializar o(s) comportamento(s) que precisa(m) ser modificado(s).

2.3.3.2 Casos de Uso Parametrizados

De forma muito parecida com o que ocorre com os casos de uso do tipo CRUD, os casos de uso do tipo *Encontrar <objeto(s)>* são muito similares entre si. Estes tipos de caso de uso possivelmente contemplarão uma entrada do usuário e uma resposta do sistema que retorna uma lista de objetos que atendem aos critérios especificados pelos usuários. Estes casos de uso são então idênticos e diferem apenas no objeto que está sendo tratado e nos parâmetros (critérios) que são passados para realizar a pesquisa. Por este motivo são chamados de casos de uso parametrizados.

Provavelmente, durante a implementação do sistema, uma equipe de desenvolvimento criará um mecanismo de pesquisa genérico que atenderá a todos os casos de uso deste tipo, ou até mesmo será feito uso de algum framework que implemente estas operações de forma genérica, onde somente os pontos de extensão serão customizados.

2.4 MEDIÇÃO, MEDIDA E MÉTRICA

Os termos “medição”, “medida” e “métrica” estão presentes em muitas áreas e constituem parte do trabalho de muitos especialistas. Diariamente lidamos com estes termos, sem muitas vezes perceber: quando dirigimos, temos constantemente a velocidade do automóvel sendo medida; ao escutarmos o boletim do tempo, temos as estimativas das medidas de temperatura para o dia; quando compramos uma casa, perguntamos sempre qual seu tamanho em metros quadrados. As medidas realmente fazem parte de nossas vidas e na grande maioria das vezes lidamos com seus valores de forma muito natural.

Estes termos estão diretamente relacionados uns aos outros: **medição (ou cálculo)** trata dos processos de avaliação de um determinado objeto do mundo real com o intuito de descrevê-lo através de números ou símbolos. O resultado de um processo de medição é uma **medida**. Por sua vez, as medidas são apresentadas sobre uma determinada unidade, as quais são chamadas de **métrica**.

As medidas, em suas diversas áreas, servem de apoio para tomada de decisões. Na área médica, por exemplo, mede-se o PSA (antígeno específico da próstata) para diagnosticar o câncer de próstata. Na área de gerenciamento de projetos de software, utilizam-se processos de medição para derivar medidas e poder estimar o tamanho de um projeto e, consequentemente, derivar estimativas de custo, esforço e prazo.

2.4.1 Características de um Processo de Medição

Fenton e Pfleeger definem um processo de medição como:

> “Medição é um processo no qual números ou símbolos são atribuídos para atributos de entidades do mundo real de forma a descrevê-los com regras claras e bem definidas”. (FENTON e PFLEEGER, 1998)

De forma geral, as medições capturam informações sobre **atributos** de **entidades** (FENTON e PFLEEGER, 1998), que são representadas através de medidas. As **entidades** de um processo de medição podem ser de dois tipos:

- Um objeto: quando representa uma pessoa, um software, um prédio, etc.;
- Um evento: quando representa um comportamento como o deslocamento de um automóvel, a colisão de dois objetos, uma fase de um projeto de software, etc.

Os **atributos** representam características e propriedades de uma entidade. São exemplos de atributos:

- Temperatura: atributo das entidades Tempo, Pessoa ou Animal;
- Peso: atributo das entidades Pessoa, Animal ou Coisa;
- Tamanho: atributo das entidades Pessoas, Objetos (casas, prédios), etc.;
- Tamanho Funcional: atributo da entidade Software.

Os atributos das entidades podem ser representados e descritos através de **números e/ou símbolos** associados a uma **unidade**, que são as **medidas** e **métricas**, respectivamente. Por exemplo, ao dizer que a distância de um local a outro é de 10 metros, temos a representação dos dois conceitos, onde 10 é o valor da **medida** e metros é a **métrica** (unidade) utilizada para representar esta medida.

Estes tipos de medidas que normalmente envolvem um único atributo podem ser chamadas de medidas diretas. Logo, neste exemplo, o processo de medição é uma forma de medida direta de um atributo, como, por exemplo, a altura de um prédio.

Também existem alguns casos que não é possível medir de forma direta um atributo de uma entidade. Um exemplo disso é a avaliação de um automóvel usado. Para obtermos o valor de mercado (medida indireta) de um automóvel usado (entidade) levamos em consideração os atributos marca, modelo, ano, acessórios e estado de conservação. Todos estes são atributos que podem ser medidos de forma direta, e que quando avaliados em conjunto podem ser utilizados para obter uma medida indireta: o valor de mercado.

Ao identificar os atributos mais relevantes de uma entidade, sejam eles diretos ou indiretos, torna-se possível entender as dimensões de uma entidade e também sua comparação com outras entidades.

No projeto de desenvolvimento de um software é muito útil dispor de diversas medidas de atributos para ter um melhor entendimento das características do projeto, como seu tamanho funcional, sua complexidade e o tempo de duração do projeto. Muitas

vezes as medidas auxiliam inclusive nas tomadas de decisões sobre o projeto, principalmente em função de sua viabilidade, custos e prazos.

2.4.2 Tipos de Escalas de Medição

Para Panadian (PANADIAN, 2003) o processo de medição é composto por três fases coexistentes, sendo que uma influencia as demais. A primeira fase é a cognitiva, onde as entidades são medidas com a mente. Nesta fase, a medição inicia com percepção e posterior julgamento, tudo de uma forma muito subjetiva. A segunda fase é a semântica, onde são definidos "rótulos" para caracterização das entidades. Nesta fase, a aplicação dos rótulos é realizada através dos agrupamentos por similaridades. A terceira fase, que é a mais precisa e refinada, é a quantitativa. Nesta fase, os atributos e entidades são definidos por números, permitindo assim julgamento e comparações mais precisas. Nesta fase podem-se aplicar modelos matemáticos para análises mais refinadas.

Para realizações das medições é necessário conhecer e identificar qual a melhor escala para classificar ou valorar os atributos de uma entidade. Para Fenton e Pfleeger (FENTON e PFLEEGER, 1998), existem dois tipos de escalas: as subjetivas e as objetivas.

As escalas subjetivas consistem na classificação ou valoração de um atributo através de uma observação pessoal, e por esta razão são chamadas de subjetivas. Um dos grandes problemas destes tipos de escalas está nas possíveis variações das avaliações de um atributo quando realizadas por pessoas diferentes.

Como o foco deste trabalho está na definição de um método mais objetivo para medição de software, será dado maior enfoque às escalas objetivas, as quais se alinham melhor aos objetivos deste trabalho.

2.4.2.1 Escalas Objetivas

As escalas objetivas são assim denominadas pelo fato de medirem seus atributos de forma objetiva. Existem, pelo menos, seis tipos de escalas objetivas de medidas, que em ordem crescente de sofisticação, no que tange sua capacidade de representação e possibilidade de realização de operações matemáticas, estão definidas a seguir.

Nominal: atribui expressões que rotulam objetos avaliados, de forma que eles possam ser referenciados sem ambiguidade. Exemplo: descrever a nacionalidade de um indivíduo com os valores Brasileiro, Argentino, Chileno, etc.

Tipologia: classifica um atributo através de rótulos de classificação por categoria. Neste tipo de escala não existe a noção de ordem, relevância ou peso. Dessa forma não é necessário o estabelecimento de números para a distinção. Essa escala também é conhecida como escala nominal por categoria. Exemplo: classificar se determinado software é (C) comercial ou (A) acadêmico.

Ordinal: acrescenta noção de valorização às classificações originárias da escala de tipologia, ou seja, apresentam um sistema de classificação do tipo ranking. Operações matemáticas continuam sem fazer sentido neste tipo de escala, pois não há regra rígida para distribuição dos números. Exemplo: classificar o tamanho de um software, baseado no número de linhas de código, em comparação com a média de tamanho de sistemas desenvolvidos pela empresa, utilizando-se o ranking (1) menor, (2) igual e (3) maior.

Intervalo: acrescenta regras de limites entre os intervalos às classificações originárias da escala do tipo ordinal. Desta forma é possível avaliar o crescimento de uma classe inferior em relação a uma superior seguinte, possibilitando assim, a realização de operações de adição e subtração, entre os intervalos. Exemplo: definição do conceito de um aluno em uma disciplina, onde o conceito A é atribuído ao aluno que tiver nota superior a 9, o conceito B é atribuído ao aluno que tiver nota entre 8 e 9, o conceito C é atribuído ao aluno que tiver nota entre 7 e 8, e o aluno com nota abaixo de 7 fica com o conceito D.

Por taxa: acrescentam as possibilidades de operações de multiplicação e divisão, ausentes na escala por intervalo. Desta forma, as comparações tornam-se mais claras. Por exemplo, é possível com este tipo de escala concluir que um projeto levou a metade do tempo que outro ou que a taxa de erro foi duas vezes menor na comparação entre dois projetos. Há também o elemento de valor zero, significando total ausência de um atributo. O modelo matemático desta escala é dado por $M = aM'$, ou seja, a medida M é obtida através de uma variável escalar aplicada sobre outra medida, M'.

Razão: é utilizada quando a única forma de valorar um determinado atributo é através da contagem do número e ocorrências deste atributo em uma determinada entidade. Consequentemente, todas as operações matemáticas são aplicáveis para as contagens obtidas.

Existe também um tipo especial de escala que se encontra entre a escala ordinal e a intervalar, chamada Likert. Esta escala é muito utilizada para representar o grau de concordância em relação a uma determinada afirmação, onde os extremos são opiniões contraditórias e o termo central indica uma indiferença. Exemplo: um entrevistado ao ser perguntado sobre determinado tema poderia ter as seguintes opções de resposta: “Não concordo totalmente”, “Não concordo parcialmente”, “Indiferente”, “Concordo parcialmente” e “Concordo totalmente”. Neste tipo de escala, para cada opção de resposta é associado um valor numérico que representa o grau de concordância com a opção. Neste exemplo, uma possível escala de valores poderia ser -2, -1, 0, +1 e +2, respectivamente, onde os valores negativos indicam discordância e os positivos indicam concordância. O valor central (zero) indica uma indiferença do entrevistado em relação ao assunto.

2.5 MÉTODOS, MEDIDAS E MÉTRICAS DE MEDIÇÃO DE SOFTWARE

Os métodos de medição de software permitem a obtenção das medidas de um projeto de software, expressando, através de números, o tamanho ou as funcionalidades do projeto. A estes números são associadas as métricas, que permitem o entendimento completo da medida. Com um bom dimensionamento de um projeto de software, é possível estimar esforço, custos e prazos para sua execução.

É muito importante não confundir os termos “medição” e “medida”. Medição de software é definida por Peters e Pedrycz como:

> “Uma medição de software é uma técnica ou método que aplica medidas de software a uma classe de objetos de engenharia de software de forma a atingir um objetivo predefinido.” (PETERS e PEDRYCZ, 2001)

O termo “medida” de software é definido por (BASILI, CALDIERA e ROMBACH, 1994) como:

> “Uma medida de software é o mapeamento de um conjunto de objetos no mundo da engenharia de software em um conjunto de construções matemáticas, tais como número ou vetores de números.” (BASILI, CALDIERA e ROMBACH, 1994)

Um método de medição consiste na formalização de como um processo de medição deve ser executado a fim de obter uma medida consistente. Normalmente um método de medição define a sequência dos procedimentos, as regras e as diretrizes de como a medição deve ser realizada.

Em relação à medição de software, existem alguns métodos que definem processos de medição com base nas funcionalidades de um sistema (entidades) e através de alguns atributos específicos das funcionalidades, e que são chamados de métodos de medição funcional de software. Alguns exemplos de atributos de medidas de tamanho funcional são: número de cadastros, consultas, relatórios, classes, etc.

O uso destes métodos tem ganhado cada vez mais importância nas fases de planejamento dos projetos, principalmente em função da necessidade de se estabelecer previsões de esforço, custos e prazos para o desenvolvimento de um projeto. No entanto, estas previsões são melhores realizadas quando baseadas em dados históricos das empresas, porém, nem sempre isso é possível em função da ausência de informações históricas nas empresas.

Esta seção apresenta os principais métodos e métricas de medição e estimativa de software utilizados pelo mercado de software, bem como suas limitações.

2.5.1 Planning Poker

O Planning Poker (COHN, 2005) é uma forma de estimativa de software onde todos os integrantes da equipe do projeto participam de forma democrática e colaborativa no processo de estimativa do projeto ou de uma iteração do projeto. É um método de estimativas subjetivo baseado somente no sentimento da equipe. É muito utilizado em processos de desenvolvimento ágil, como o Scrum.

É uma forma divertida de se estimar projetos, pois toda a equipe participa de um jogo com cartas, onde, para cada rodada é jogado uma partida para estimar um item (por exemplo, uma funcionalidade) de um *backlog*[3]. As rodadas se repetem até que todos os itens do *backlog* estejam estimados.

As regras do jogo são simples: para cada rodada, o "coordenador" do jogo lê o item que será estimado e fornece os detalhes disponíveis até o momento. Todos os participantes da equipe dispõem de um conjunto de cartas com os números da sequência de fibonacci. Cada participante então escolhe um número que, de acordo com sua opinião, melhor representa o esforço necessário para desenvolvimento do item do *backlog* que está sendo estimado na rodada. Se houverem discrepâncias entre os valores apresentados por cada participante, a rodada é repetida até que haja um consenso. Caso as rodadas sejam repetidas por várias vezes sem haver o consenso, o "coordenador" do jogo poderá definir qual o valor da estimativa.

Este método não obriga o uso de nenhuma métrica, mas normalmente observa-se que as equipes derivam suas estimativas em *Story Point*[4] ou *Ideal Days*[5].

2.5.1.1 Pontos limitantes

Apesar de ser um método que vem sendo muito utilizado em empresas que adotam metodologias ágeis, o uso do *Planning Poker* em processos de terceirização do desenvolvimento torna-se inviável em função de seu alto grau de subjetividade e da dependência da experiência da equipe de desenvolvimento do projeto.

Com sua aplicação não é possível medir um projeto para compará-lo com outro, pois o foco do método é exclusivamente em estimativas. Somente é possível comparar um projeto com outro através do uso do Planning Poker (ainda que de forma subjetiva), se ambos os projetos tenham sido estimados pela mesma equipe.

---

[3] *Backlog* é um termo em inglês utilizado para se referir a uma lista de funcionalidade e/ou pendências de um projeto de software. Pode ser é utilizado para se referir ao conjunto de funcionalidades e/ou pendências a serem desenvolvidas em uma iteração, que no Scrum chama-se de *Sprint Backlog*; ou para se referir à totalidade de funcionalidades e/ou pendências de um projeto – *Produto Backlog.*

[4] *Story Point* é uma unidade de medida (métrica) para estimar o tamanho de uma Estória de Usuário.

[5] *Ideal Days* é uma unidade de medida de tempo (métrica) que leva em consideração o tempo útil (efetivo) em um dia.

2.5.2 Análise de Pontos de Função (APF)

A análise de pontos de função (APF) foi inicialmente proposta por Allan Albrecht, pesquisador da IBM, em 1979. Por se tratar de um método complexo e em função da necessidade de interpretação das regras para medir seus atributos, percebeu-se a demanda por treinamento e certificação de profissionais. Sob este enfoque, foi criado em 1986 o IFPUG- *International Function Point Users Group*[6], entidade sem fins lucrativos, composta por profissionais e empresas de diversos países do mundo, responsável pela definição das regras do método (ANDRADE, 2004). No Brasil, o BFPUG – *Brazilian Function Point Users Group*[7] é a entidade representante oficial associada ao IFPUG. Atualmente, o documento oficial do método APF é denominado *Function Point Counting Practices Manual (CPM)*, encontrando-se na versão 4.3.1 de 2010 e reconhecido pela norma ISO/IEC 14143:1998 – *Information Technology-Software Measurement – Functional Size Measurement*.

O APF mede o desenvolvimento do software do ponto de vista do usuário através da quantificação das funcionalidades fornecidas pelo software. As entidades e atributos utilizados na contagem podem ser obtidos através da especificação detalhada dos requisitos do software. A medida final do tamanho do projeto é dada sob sua própria métrica, chamada Pontos de Função (PF).

Uma visão geral do processo de medição através da contagem de pontos de função pode ser vista na Figura 2.

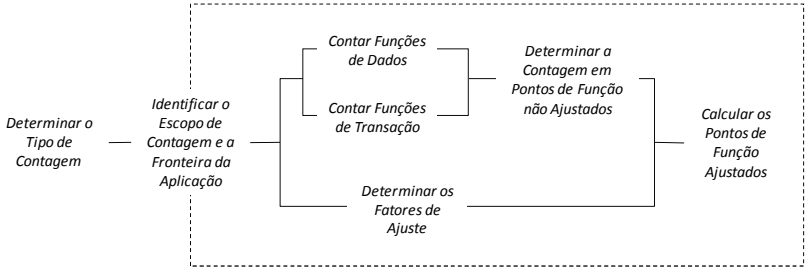

Figura 2 - Visão Geral do método APF.
Fonte: IFPUG

[6] www.ifpug.org
[7] www.bfpug.org

A aplicação do método segue algumas fases, as quais serão descritas a seguir.

2.5.2.1 Determinação do Tipo de Contagem

O primeiro passo da análise de pontos de função é determinar qual o tipo de contagem que será realizada. Os tipos de contagem previstos neste método estão definidos na Tabela 2.

Tabela 2 - Tipos de Contagem do método APF.

Projeto de Desenvolvimento	A contagem de pontos de função de um projeto de desenvolvimento mede as funções providas aos usuários na primeira instalação do software entregue quando o projeto é finalizado.
Projeto de Melhoria	A contagem de pontos de função de um projeto de melhorias mede as modificações feitas na aplicação existente que incluem, alteram, ou excluem funções de usuário entregues quando o projeto é finalizado.
Aplicação	A contagem de pontos de função da aplicação e a contagem de projeto de desenvolvimento são associadas a uma aplicação instalada. Esta contagem provê uma medição das funções atuais da aplicação entregue ao usuário. Esta contagem deve ser atualizada toda vez que um projeto de desenvolvimento ou melhoria é finalizado.

Fonte: IFPUG

2.5.2.2 Fronteira da Aplicação e o Escopo da Contagem

A fronteira da aplicação é a interface conceitual que delimita o software que será medido e o mundo exterior (seus usuários) (VAZQUEZ, SIMÕES e ALBERT, 2010). O escopo da contagem define qual a sua abrangência, ou seja, define se abrangerá um ou mais sistemas e quais módulos e funcionalidades serão incluídas na contagem.

2.5 2.3 Contagem das Funções de Dados

As funções de dados representam as funcionalidades fornecidas pelo sistema ao usuário sob a perspectiva de grupos lógicos de dados que são mantidos ou referenciados pela aplicação. Um cadastro de

clientes e um cadastro de produtos são exemplos típicos de grupos de dados. As funções de dados podem ser classificadas em:

Arquivo Lógico Interno (ALI): um grupo lógico de dados (entidade lógica e persistente) identificável pelo usuário e mantido dentro da fronteira da aplicação através de uma ou mais transações. Exemplo: tabelas de banco de dados ou arquivos de configuração que são mantidos pela aplicação.

Arquivo de Interface Externa (AIE): são similares aos ALIs, porém este grupo lógico de dados é mantido fora da fronteira da aplicação. São normalmente grupos de dados de outra aplicação que são lidos pela aplicação que está sendo contada. Exemplo: tabelas de banco de dados que são lidas pela aplicação contada, mas que são atualizadas por outra aplicação.

Os ALIs e AIEs são constituídos por tipos de registros (TR) e tipos de dados (TD). Os tipos de dados representam cada campo único de uma aplicação, que são reconhecidos pelo usuário. Os tipos de registros representam um subgrupo de tipos de dados, também reconhecidos pelos usuários. Fazendo uma analogia com tabelas de banco de dados, um tipo de dado seria equivalente a um campo da tabela (desde que visível ao usuário) e o tipo de registro seria equivalente ao conjunto dos campos, ou seja, à entidade ou tabela.

Na etapa de contagem das funções de dados, cada ALI e AIE deve ser classificado com relação a sua complexidade funcional. Esta classificação é realizada com base na contagem do número de tipos de dados e número de tipos de registros. Uma vez determinadas as quantidades de cada tipo, aplica-se a Tabela 3 para determinação da complexidade funcional dos ALIs e AIEs.

Tabela 3 - Complexidade funcional dos ALI e AIE.

		Quantidade de Tipos de Dados		
		<20	20 a 50	> 50
Quantidade de Tipos de Registros	1	Baixa	Baixa	Média
	2 a 5	Baixa	Média	Alta
	> 5	Média	Alta	Alta

Fonte: IFPUG

Uma vez obtidas as complexidades de cada ALI e AIE, é possível determinar uma primeira contagem funcional em termos de pontos de

função não ajustados, aplicando a conversão descrita na Tabela 4 para cada ALI e AIE.

Tabela 4 - Conversão da complexidade funcional dos ALI e AIE em PFNA.

Complexidade Funcional	Pontos de Função Não Ajustados (PFNA)
Baixa	7
Média	10
Alta	15

Fonte: IFPUG

2.5.2.4 Contagem das Funções de Transação

As funções de transação representam as funcionalidades de processamento de dados fornecidas pelo sistema aos usuários. Esta contagem juntamente com a de funções de dados, constituem o total de Pontos de Função não Ajustados (PFNA). Os tipos de funções de transação podem ser classificados em:

Entrada Externa (EE): uma entrada externa é um processo elementar que processa dados ou informação de controle que vêm de fora do limite de aplicação. A principal intenção de um EE é manter um ou mais ALIs e/ou alterar o comportamento do sistema. Exemplos típicos deste tipo de transação são as operações básicas sobre um determinado conceito, como incluir produto, alterar produto e excluir produto.

Saída Externa (SE): uma saída externa é uma transação que envia dados ou informações de controle do sistema para o usuário. O processamento deve envolver pelo menos um cálculo matemático ou expressões para produção de dados derivados, além de, quando necessário, atualizar um ou mais ALIs ou alterar o comportamento do sistema. Exemplo: relatórios de totais de faturamento por cliente.

Consulta Externa (CE): uma consulta externa é uma transação que envia dados ou informações do sistema para o usuário. Ao contrário das saídas externas (SE), as consultas externas são simples operações de recuperação de dados ou informações de controle dos ALIs e/ou AIEs. Exemplo: consulta no cadastro de produtos.

O processo de contagem das funções de transação passa pela identificação das quantidades de arquivos referenciados (AR) e dos tipos de dados envolvidos na transação. Um arquivo referenciado (AR) é um ALI ou AIE lido ou mantido pela função de transação. Os tipos de dados representam os campos que estão envolvidos na transação. As Tabelas 5 e 6 mostram como obter a complexidade funcional das transações em função das quantidades de arquivos referenciados e dos tipos de dados envolvidos na transação.

Tabela 5 - Critérios para avaliação de complexidade de EE.

		Quantidade de Tipos de Dados		
		<5	5 a 15	> 15
Quantidade de Arquivos Referenciados	< 2	Baixa	Baixa	Média
	2	Baixa	Média	Alta
	> 2	Média	Alta	Alta

Fonte: IFPUG

Tabela 6 - Critérios para avaliação de complexidade de SE e CE.

		Quantidade de Tipos de Dados		
		<6	6 a 19	> 19
Quantidade de Arquivos Referenciados	< 2	Baixa	Baixa	Média
	2 a 3	Baixa	Média	Alta
	> 3	Média	Alta	Alta

Fonte: IFPUG

A transformação da complexidade funcional de cada função de transação em PFNA é realizada de acordo com as Tabelas 7 e 8.

Tabela 7 - Transformação da complexidade de SE em PFNA.

Complexidade Funcional	**Pontos de Função Não Ajustados (PFNA)**
Baixa	4
Média	5
Alta	7

Fonte: IFPUG

Tabela 8 - Transformação da complexidade de EE e CE em PFNA.

Complexidade Funcional	Pontos de Função Não Ajustados (PFNA)
Baixa	3
Média	4
Alta	6

Fonte: IFPUG

2.5.2.5 Determinação do Fator de Ajuste

A determinação do fator de ajuste é o último passo para a contagem de pontos de função. Atualmente, para adequar-se ao padrão ISO/IEC 14143 de medição funcional, o IFPUG tornou o fator de ajuste opcional (VAZQUEZ, SIMÕES e ALBERT, 2010). Seu propósito é ajustar a medição de pontos de função não ajustados em +/-35% de acordo com a influência de 14 características gerais que estão listadas na Tabela 9.

Tabela 9 - Características gerais que influenciam nos fatores de ajuste.

1.	Comunicação de dados
2.	Processamento distribuído
3.	Performance
4.	Configuração altamente utilizada
5.	Volume de transações
6.	Entrada de dados *on-line*
7.	Eficiência do usuário final
8.	Atualização *on-line*
9.	Complexidade de processamento
10.	Reutilização
11.	Facilidade de instalação
12.	Facilidade de Operação
13.	Múltiplos locais
14.	Facilidade de mudanças

Fonte: (VAZQUEZ, SIMÕES e ALBERT, 2010)

Cada uma das características gerais possui um nível de influência sobre a aplicação que está sendo medida, e deve ser ponderada com base em um intervalo discreto de zero a cinco, conforme descrito na Tabela 10.

Tabela 10 - Níveis de influência de uma característica geral do sistema.

0.	Nenhuma influência
1.	Influência mínima
2.	Influência moderada
3.	Influência média
4.	Influência significativa
5.	Grande influência

Fonte: (VAZQUEZ, SIMÕES e ALBERT, 2010)

Uma vez determinado o nível de influência de cada uma das 14 características gerais do sistema, o fator de ajuste é calculado com base na Equação 1.

$$FA = \left(\sum_{i=1}^{14} NIC\right) 0,01 + 0,65 \qquad (1)$$

Onde:
NIC é o nível de influência de uma característica geral do sistema

2.5.2.6 Ajuste da Contagem

A última etapa para obtenção da contagem final em pontos de função é a aplicação do fator de ajuste na contagem geral do tamanho funcional em pontos de função não ajustados, obtidos através do somatório dos pontos de função não ajustados de cada função de dados e função de transação. A obtenção da contagem total de pontos de função não ajustados é obtida através da Equação 2.

$$PFNA = \left(\sum_{i=1}^{d} Complex(FD_d)\right) + \left(\sum_{i=1}^{t} Complex(FT_t)\right) \qquad (2)$$

Onde:
d - é o número de funções de dados
t – é o número de funções de transação
Complex(FD_d) é a complexidade da função de dados d
Complex(FT_t) é a complexidade da função de transação t

A contagem final de pontos de função ajustada pelas características gerais do sistema pode ser obtida através da Equação 3.

$$PFA = PFNA * FA \quad (3)$$

2.5.2.7 Pontos Limitantes

O método APF, apesar de ser um dos métodos mais utilizados atualmente, apresenta algumas falhas fundamentais na sua construção que o impede de produzir medições válidas (KITCHENHAM, 1997). Sua aplicação em um contexto controlado dentro de uma organização específica possivelmente apresentará poucos problemas. No entanto, se forem utilizados dados sobre os resultados de sua aplicação em diferentes empresas como base para se estabelecer contratos de desenvolvimento de sistemas ou para desenvolver modelos genéricos de estimativas, há um grande risco de se encontrar problemas (KITCHENHAM, 1997).

Em (KITCHENHAM, 1997), Barbara Kitchenham descreve sobre dois grandes problemas relacionados ao método de Análise de Pontos de Função: problemas de construção e problemas de instabilidade no modelo de predição.

O primeiro problema diz respeito aos tipos de escalas utilizados para a mensuração dos atributos das entidades alvo do método de medição. O uso de escalas ordinais do tipo Simples, Médio e Complexo associado aos seus atributos inviabilizam sua transformação em escalas de intervalos e razão. Medidas em escala ordinal não podem ser somadas formalmente porque seria um absurdo adicionar um rótulo de "simples" a um rótulo de "complexo", mesmo se for utilizado um rótulo 3 como um sinônimo para "Simples" e um rótulo 5 como um sinônimo de "complexo" (KITCHENHAM, 1997). Por consequência, se não é possível realizar operações matemáticas sobre ordinais, também não se pode definir a diferença entre os valores finais de uma contagem, ou seja, não é possível afirmar que o valor 4 obtido no final de uma medição é duas vezes maior que um valor 2. Da mesma forma, não é possível dizer que, se um projeto de tamanho 4 levou 40 horas, um projeto de tamanho 2 levará 20 horas, ou seja, não é possível derivar diretamente o esforço para desenvolvimento de um projeto.

O segundo grande problema relatado por Kitchenham está relacionado à correlação existente entre os atributos utilizados no método de medição, que podem ainda apresentar valores diferenciados para conjuntos de dados diferentes. A correlação entre os atributos

também foi identificada por Lokan (LOKAN, 1999) quando analisou um conjunto de 269 projetos.

A correlação entre as variáveis, bem como a variação de seus valores para conjuntos de dados diferentes, caracteriza uma instabilidade do modelo de predição, o que significa dizer que o processo de estimativa com base nas medições em pontos de função não são estáveis e podem variar em contextos diferentes.

Em um estudo anterior (KITCHENHAM e KÄNSÄLÄ, 1993), além destes problemas, também foi identificado que alguns atributos considerados no método de análise de pontos de função não possuem relação significativa com o esforço. Este é o caso do atributo “Consultas Externas”.

Outro ponto limitante do método de análise de pontos de função é que para aplicar o processo de medição são necessárias informações detalhadas sobre o software (KUSUMOTO, MATUKAWA, *et al.*, 2004), tornando a aplicação deste método dificultada em fases iniciais dos projetos.

Além destes problemas, a prática e utilização do método de análise de pontos de função no mercado de software têm mostrado que é necessário um analista experiente e treinado para a realização de uma boa contagem, pois sua aplicação é muito complexa e deve seguir muitas regras, as quais devem ser interpretadas para serem aplicadas. No entanto, o número deste tipo de profissional ainda é limitado no Brasil, e ainda mais nas empresas públicas.

Em virtude de todos estes problemas, o uso do método de análise de pontos de função em processos de terceirização do desenvolvimento de software em empresas públicas torna-se questionável, pois a instabilidade do modelo de predição pode acarretar em custos excessivos dos projetos.

2.5.3 Método Mark II ou Mk II

Este método é uma variação da Análise de Pontos de Função usado principalmente no Reino Unido. A Análise de Pontos de Função Mk II foi proposta por Charles Symons em 1991 como uma alternativa para produzir uma melhor contagem de complexidade de processamento interno (SYMONS, 1991). Segundo Symons, a classificação das funções de dados e funções de transações em Simples, Médio e Complexo é simples demais e pode acarretar em distorções na contagem final.

Inicialmente o método Mk II era de propriedade particular da empresa KPMG de consultoria em Tecnologia da Informação. Posteriormente, passou a ser de domínio público, mantido pela *United Kingdom Software Metrics Association* (UKSMA).

Uma das diferenças deste método para o APF é que nele o sistema é visto como uma coleção de transações lógicas, onde uma transação consiste de um componente de entrada, processamento e saída que é disparado por um único evento de interesse do usuário. As transações lógicas possuem um conceito muito similar ao conceito de casos de uso (RIBU, 2001). Alguns exemplos de transações lógicas são: criar um cliente, atualizar um produto, fazer uma operação de saque, consultar um pedido de compra e produzir um relatório.

Outra mudança significativa em relação à APF está no fato de que no método MK II, a lista de fatores de ajuste técnico (características gerais do sistema) é modificada, onde algumas características foram acrescentadas.

De maneira resumida, o método MKII consiste na avaliação das informações que são tratadas nas transações lógicas[8]. Para cada transação lógica, são contados o número de tipos de dados de entrada (N_i), o número de tipos de entidades referenciadas (N_e) e o número de tipos de dados de saída (N_o). Com base nesta contagem o tamanho funcional (*FPI – Function Point Index*) é dado através do somatório de todos os pesos das transações lógicas, dos tipos de dados de entrada (N_i), das entidades referenciadas (N_e) e dos tipos de dados de saída (N_o). O índice de pontos de função (*FPI*) pode então ser calculado através da Equação 4.

$$FPI = Wi \left(\sum_{i=1}^{n} Ni_i\right) + We \left(\sum_{i=1}^{n} Ne_i\right) + Wo \left(\sum_{i=1}^{n} No_i\right) \qquad (4)$$

Onde:
Wi = 0. 58
We = 1. 66
Wo = 0. 26
n – é o número de transações lógicas da aplicação
Ni – número de tipos de dados de uma transação lógica
Ne – número de entidades referenciadas em uma transação lógica
No – número de tipos de dados de saída de uma transação lógica

[8] No método MK II, uma transação lógica é o processo de menor nível composto de uma entrada-processamento-saída suportado pela aplicação.

Uma vez determinado o FPI, pode-se aplicar um ajuste em função dos fatores de complexidade técnica.

2.5.3.1 Pontos Limitantes

O método MKII é uma variação do método de pontos de função, substituindo o uso de escalas ordinais nas análises de complexidades das transações por escalas razão. Desta forma, é possível a realização de operações matemáticas entre os atributos envolvidos na contagem possibilitando uma comparação direta entre eles.

No entanto, o método mantém a complexidade da contagem, o que exige profissionais qualificados para sua realização, mantendo o mesmo problema do método de análise de pontos de função.

Além disso, para sua aplicação é necessário que os requisitos da aplicação estejam bem definidos, fato este que muitas vezes não ocorre nas fases iniciais dos projetos, onde é necessário o estabelecimento de contratos.

2.5.4 COSMIC (FFP - Full Function Point)

O método *Full Function Point* (FFP) foi proposto em 1997 com o objetivo de estender o método APF para permitir um melhor dimensionamento do tamanho de sistemas de tempo real e sistemas básicos (COSMIC, 2007). Em 1998, um grupo de especialistas em medição constituiu o *Common Software Measurement International Consortium* (COSMIC), organização responsável pelo método. Atualmente o método está na sua versão 3.0, de 2007, reconhecida pela norma ISO/IEC 14143:2007. Sua denominação inicialmente era COSMIC-FFP, que agora foi simplificada para somente COSMIC.

A Figura 3 mostra uma visão geral do método COSMIC.

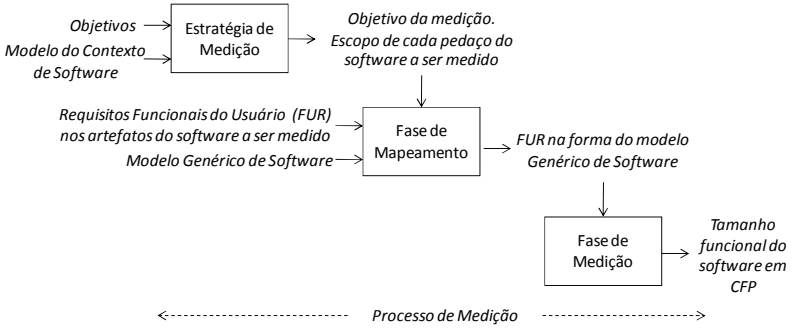

Figura 3 - Visão geral do método COSMIC.
Fonte: (COSMIC, 2007)

O método COSMIC define algumas regras e procedimentos que são aplicados ao software a ser avaliado para a derivação de um valor que quantifica o dimensionamento funcional da aplicação. Para sua aplicação o software deve estar disposto na forma de um modelo genérico de software.

De acordo com o modelo genérico de software do COSMIC, os requisitos funcionais da aplicação devem ser decompostos em um conjunto de processos funcionais. Cada um desses processos representa um único conjunto de sub-processos que executam movimentação ou manipulação de dados. Dessa forma, o software a ser medido deve ser constituído por entradas e deve produzir saídas para seus usuários. As informações manipuladas pelo software devem estar projetadas como grupos de dados, com seus respectivos atributos.

De forma geral, os principais passos para a mensuração de um projeto de software utilizando o método COSMIC são:

1. Identificação de movimento de dados: consiste na identificação de movimentos de Entrada (E), Saída (S), Leitura (R) e Escrita (W) dos dados, para cada processo funcional identificado na fase de mapeamento. Um movimento de dados move um ou mais atributos de ou para um único grupo de dados. Se um movimento envolver mais de um grupo de dados, deverá ser identificado uma E, S, R ou W para cada grupo de dados envolvido.

2. Aplicação da função de medição: para cada movimento de dados identificado (E, S, R ou W), em todos os processos funcionais, o valor de 1 COSMIC Function Point (CFP) deve ser definido como o tamanho elementar do movimento.

3. Agregação dos resultados da medição: consiste em agregar os resultados parciais dos movimentos de dados identificados, em um único valor de dimensionamento funcional. A Equação 5 define como obter o tamanho de cada processo.

$$Tam_{CFP}(ProcFunc_i) = \sum Tam(E_i) + \sum Tam(S_i) + \sum Tam(R_i) + \sum Tam(W_i) \quad (5)$$

Onde:
Tam – é utilizada como abreviação de Tamanho
ProcFunc – é utilizado como abreviação de Processo Funcional
E_i – é o número de entradas do processo funcional
S_i – é o número de saídas do processo funcional
R_i – é o número de leituras do processo funcional
W_i – é o número de escritas do processo funcional

Por fim, o tamanho funcional da aplicação é obtido pelo somatório do tamanho de todos os processos funcionais envolvidos na contagem. A Equação 6 define como obter o tamanho funcional da aplicação.

$$Tam_{CFP}(Software) = \sum Tam(ProcFunc_i) \quad (6)$$

Onde:
Tam – é utilizada como abreviação de Tamanho
ProcFunc – é utilizado como abreviação de Processo Funcional

2.5.4.1 Pontos Limitantes

O método COSMIC FFP consiste em medir todos os processos funcionais de uma aplicação em função dos seus movimentos de Entrada (E), Saída (S), Leitura (L) e Escrita (W) de dados, utilizando uma escala razão. Apesar de ser de simples utilização, a sua aplicação depende dos requisitos estarem detalhados em um estágio muito avançado, inviabilizando sua aplicação em estágios iniciais de um projeto.

Além disso, é necessário que o sistema seja especificado ou mapeado para um Modelo Genérico de Software definido no método, ou seja, para sua utilização é necessário que as empresas adaptem seus processos ao modelo definido no método.

2.5.5 Pontos de Caso de Uso (PCU)

O método de contagem de pontos de caso de uso foi inicialmente proposto por Gustav Karner (KARNER, 1993), como uma adaptação do método de contagem de pontos de função (VAZQUEZ, SIMÕES e ALBERT, 2010) voltada para sistemas orientados a objetos. Simplificando, o método de Karner consiste em contar a quantidade de atores e de transações dos casos de uso. Esta contagem é realizada através de um método conduzido em seis passos descritos a seguir.

2.5.5.1 Peso dos Atores do Sistema

Os atores dos modelos de casos de uso são categorizados em simples, médio e complexo. A classificação dos atores é realizada em função dos tipos de interface que eles derivam. A Tabela 11 define os pesos para cada tipo de ator, bem como uma descrição para auxiliar a classificação de um ator de acordo com seu tipo.

Tabela 11 - Pesos dos atores

Tipo	Descrição	Peso
Simples	Um ator é considerado simples se ele representa outro sistema com uma API (*Application Programming Interface*) definida.	1
Médio	Um ator é considerado médio se é uma interação com outro sistema através de um protocolo ou se é uma interação humana através de linha de comando.	2
Complexo	Um ator é considerado complexo se ele interage com o sistema através de uma interface gráfica.	3

Depois de classificados os atores do sistema, a quantidade de cada tipo de ator é multiplicada pelo respectivo peso, e ao final os valores são somados, derivando assim o peso total de atores (UAW-*Unadjusted Actors Weights*). A Equação 7 define como o UAW pode ser obtido.

$$UAW = NA_{Simples} + NA_{Médio} * 2 + NA_{Complexos} * 3 \qquad (7)$$

Onde:
NA – é o numero de atores

2.5.5.2 Peso dos Casos de Uso não Ajustados

Cada caso de uso é categorizado em simples, médio e complexo de acordo com seu número de transações (Tabela 12), incluindo as transações contidas nos fluxos alternativos dos casos de uso. Uma transação é um evento que ocorre entre a requisição do usuário e a resposta do sistema. Para cada tipo é atribuído um peso e a quantidade de cada tipo de caso de uso é multiplicada pelo respectivo peso, e ao final os valores são somados. O valor final é o tamanho dos casos de uso não ajustados UUCW (*Unadjusted Use Case Weights*).

Tabela 12 - Peso dos casos de uso

Tipo	Descrição	Peso
Simples	Até 3 transações	5
Médio	De 4 a 7 transações	10
Complexo	Mais que 7 transações	15

A Equação 8 define como o UUCW pode ser obtido.

$$UUCW = NUC_{Simples} * 5 + NUC_{Médio} * 10 + NUC_{Complexos} * 15 \qquad (8)$$

Onde:
NUC – é o numero de casos de uso

2.5.5.3 Pontos de Casos de Uso Não Ajustados

A quantidade de pontos de casos de uso não ajustados (*UUCP - Unadjusted Use Case Point*) é obtida através da soma dos pesos em função dos atores com o tamanho dos casos de uso não ajustados (UUCW). A Equação 9 define como obter o valor de UUCP.

$$UUCP = UUCW + UAW \qquad (9)$$

2.5.5.4 Fatores de Ajustes Técnicos

A quantidade de pontos de caso de uso obtida anteriormente é ajustada pelos fatores técnicos e ambientais, os quais estão definidos nas Tabelas 13 e 14. Para cada fator é atribuído um valor entre 0 e 5 de acordo com sua influência no projeto. Atribuindo 0 para um fator, significa que ele é irrelevante para o projeto, enquanto 5 indica que é um fator essencial. Escalas intermediárias devem ser usadas para indicar a real relevância do fator para o projeto.

Tabela 13 - Fatores técnicos que influenciam na complexidade

Fator	Descrição	PT
T1	Sistemas Distribuídos	2, 0
T2	Tempo de resposta/performance	1, 0
T3	Eficiência (on-line)	1, 0
T4	Processamento interno complexo	1, 0
T5	Código deve ser reutilizável	1, 0
T6	Facilidade de instalação	0, 5
T7	Usabilidade	0, 5
T8	Portabilidade	2, 0
T9	Facilidade de manutenção	1, 0
T10	Acessos simultâneos (concorrência)	1, 0
T11	Aspectos especiais de segurança	1, 0
T12	Acesso direto para terceiros	1, 0
T13	Facilidades especiais de treinamento	1, 0

O peso dos fatores técnicos ($T_{Factor} - Technical\ Factor$) é obtido através da multiplicação da pontuação atribuída para cada fator pelo seu respectivo peso (Tabela 13), somando-se então todos os valores ao final. A Equação 10 define como o T_{Factor}pode ser obtido.

$$T_{Factor} = \sum_{i=1}^{13} PT_i * NI_i \qquad (10)$$

Onde:
PT – representa o Peso do fator técnico T
NI – representa o Nível de Influência atribuído ao fator técnico

Uma vez obtido o T_{Factor}, calcula-se TCF (*Technical Complexity Factor*). A Equação 11 determina como o TCF pode ser obtido.

$$TCF = 0.6 + (0.01 * \mathrm{T_{Factor}}) \qquad (11)$$

Tabela 14 - Fatores ambientais que contribuem na eficiência

Fator	Descrição	PF
F1	Familiaridade com a Metodologia de Gestão e Desenvolvimento	1, 5
F2	Experiência na Aplicação	0, 5
F3	Expert na Técnica de Desenvolvimento	1, 0
F4	Experiência do Gerente de Projeto	0, 5
F5	Motivação	1, 0
F6	Requisitos estáveis	2, 0
F7	Trabalhadores part-time	-1, 0
F8	Dificuldade da Linguagem de Programação	-1, 0

O peso dos fatores ambientais (E_{Factor}) é obtido através da multiplicação da pontuação atribuída para cada fator pelo seu respectivo peso (Tabela 14), somando-se então todos os valores ao final. A Equação 12 define como o E_{Factor}pode ser obtido.

$$E_{Factor} = \sum_{i=1}^{8} PF_i * NI_i \qquad (12)$$

Onde:
PF – representa o Peso do fator ambiental F
NI – representa o Nível de Influência atribuído ao fator ambiental

Uma vez obtido o E_{Factor}, calcula-se EF (*Environmental Factor*). A Equação 13 determina como o EF pode ser obtido.

$$EF = 1.4 + (-0.03 * \mathrm{E_{Factor}}) \qquad (13)$$

2.5.5.5 Pontos de Caso de Uso Ajustados

O cálculo da quantidade de pontos de caso de uso ajustados (*UCP - Use Case Point*) é feito através da multiplicação dos pontos de casos de uso não ajustados pelo fator técnico e pelo fator ambiental. A Equação 14 define como obter o valor de UCP.

$$UCP = UUCP * TCF * EF \qquad (14)$$

2.5.5.6 Determinação do Esforço

O esforço é calculado através da multiplicação da quantidade de pontos de caso de uso ajustados (UCP) pelo valor de produtividade específico de Homem-Hora (HH) para cada ponto por caso de uso. Karner apresenta resultados que mostram que para cada UCP são necessários 20 horas-homem. A Equação 15 define como o Esforço pode ser estimado em função da quantidade de pontos de caso de uso.

$$Esforço = UCP * Produtividade_{/1UCP} \qquad (15)$$

Onde:
$Produtividade_{/1UCP}$ *- é o valor médio da quantidade de homem-hora para desenvolvimento de 1 UCP. Karner propõe um valor de 20 Homem-Hora por UCP.*

2.5.5.7 Pontos Limitantes

O método Pontos de Caso de Uso, apesar de ter uma abordagem simples, possui alguns pontos que dificultam sua utilização em processos de terceirização do desenvolvimento de software.

De forma geral, casos de uso são fundamentalmente uma forma textual (COCKBURN, 2005) de descrever requisitos funcionais de um projeto de software. No entanto, não existe uma forma padronizada de se escrever casos de uso. Diferentes analistas podem escrever um mesmo caso de uso de diferentes formas e com números diferentes de passos (transações).

Um ponto limitante deste método é o fato dele utilizar uma escala ordinal com base no número de passos (transações) para classificação da complexidade dos casos de uso. Desta forma, a classificação dos casos de uso torna-se muito dependente da forma como os casos de uso são escritos e detalhados (ANDA, DREIEM, *et al.*, 2001)(VIEIRA e WAZLAWICK, 2006) e podem então variar quando descritos por diferentes analistas, gerando medidas diferentes para um mesmo projeto de software (SMITH, 1999).

Esta dependência da forma como os casos de uso são escritos tornam o método de classificação das complexidades dos casos de uso muito frágil e inviabilizam a comparação de medidas quando feitas em contextos diferentes[9].

Outro ponto limitante no método de pontos de caso de uso é a necessidade do detalhamento dos casos de uso, fato que muitas vezes não ocorre nas fases iniciais dos projetos e também com empresas que optam por uma abordagem mais ágil no seu processo de desenvolvimento de software.

A fragilidade do método de avaliação de complexidade dos casos de uso e, consequentemente, da medição final de um projeto, bem com a necessidade de detalhamento dos casos de uso, dificultam a utilização do método de pontos de função em processos licitatórios e no estabelecimento de contratos, pois estes são mecanismos formais com regras bem definidas e não admitem ambiguidades (por princípio). Além disso, as licitações e contratos normalmente são estabelecidos nas fases iniciais dos projetos, onde não se tem ainda o detalhamento completo dos requisitos e casos de uso do projeto.

---

[9] Diferentes empresas, diferentes analistas e/ou diferentes processos e padrões de desenvolvimento de software.

3 TRABALHOS RELACIONADOS

Este capítulo apresenta algumas abordagens de diversos autores relacionadas ao uso de casos de uso para medição e estimativa de projetos de software. Os títulos das seções deste capítulo correspondem aos títulos das publicações relacionadas a estas abordagens. Como o intuito deste capítulo é apresentar trabalhos relacionados, o conteúdo de suas seções será restrito à essência dos métodos, sem levar em consideração os resultados obtidos nos estudos de casos de cada publicação. Para cada método apresentado serão abordados seus pontos limitantes.

3.1 ESTIMATIVAS PRECOCES COM BASE EM CASOS DE USO

Em (ROBIOLO e OROSCO, 2007), os autores introduzem dois tipos de medidas de software que podem ser obtidas através dos casos de uso: uma baseada nas entidades de dados, chamada de *Entity Objects*, e outra baseada nas transações dos sistemas, chamada de *Transactions*. Para a utilização das duas medidas é necessário uma análise sintática dos textos dos casos de uso.

Os *Entity Objects* são todos objetos definidos no modelo de dados de um sistema que possuem suas informações persistidas de alguma forma. Estes objetos podem ser identificados através da análise sintática dos textos dos casos de uso. Nesta análise sintática devem-se separar os substantivos encontrados que fazem parte do vocabulário do domínio do problema. Os substantivos repetidos ou similares também devem ser descartados.

Com o uso de *Entity Objects*, o tamanho da aplicação é dado através do somatório de todas os *Entity Objects* identificados em cada caso de uso (Equação 16).

$$TamanhoAplicação = \sum Número\ de\ Entity\ Objects \qquad (16)$$

Uma *Transaction* pode ser definida como um conjunto de ações executadas por um sistema como resposta a um estímulo de um ator. Essa resposta deve acrescentar algum valor ao ator.

As *transactions* também podem ser identificadas através da análise sintática dos casos de uso. Para isso, no processo de análise sintática, deve-se separar os sujeitos dos verbos. Os sujeitos correspondem aos atores, enquanto os verbos às transações.

Com o uso de *Transactions*, o tamanho da aplicação é dado através do somatório de todas as *transactions* identificadas em cada caso de uso (Equação 17).

$$TamanhoAplicação = \sum Número\ de\ Transactions\ dos\ Casos\ de\ Uso \quad (17)$$

3.1.1 Pontos Limitantes

A aplicação de um método baseado em análise sintática dos textos dos casos de uso traz riscos ao processo de medição, pois se torna muito dependente da forma como estão escritos os casos de uso. Além disso, a tarefa de análise sintática, se realizada manualmente, se torna muito trabalhosa, principalmente em sistemas que tenham muitos casos de uso. Em virtude destes fatores, muitas falhas poderiam ocorrer, as quais impactariam diretamente no resultado da medição.

Para uma utilização mais segura deste método é fundamental que seja utilizado um software para realizar o processamento da análise sintática para que, com base em um vocabulário de entrada definido pelo usuário, possa separar os sujeitos, verbos e substantivos. Além disso, o analisador sintático deveria ter suporte a vários idiomas para garantir sua aplicação em regiões diferentes.

Outro ponto importante é a necessidade do analisador suportar um dicionário de sinônimos para identificar quais substantivos e verbos são equivalentes.

3.2 ESTIMATIVA DE ESFORÇO BASEADA EM CASOS DE USO

Em (BRAZ e VERGILIO, 2006), os autores introduzem dois novos métodos de medição baseados no tamanho de cada caso de uso do sistema: (a) *Use Case Size Points (USP)* e (b) *Fuzzy Use Case Point (FUSP)*. Em ambos os métodos, o tamanho de um sistema é obtido em função do tamanho de seus casos de uso e é definido sob uma nova unidade de medida (métrica) também chamada de USP (*Use Case Size Point*)

3.2.1 Use Case Size Point (USP)

O método USP consiste em calcular o tamanho do caso de uso em função de 6 atributos dos casos de uso do sistema descritos a seguir:

1. *Total Points Actor (TPA)*: a complexidade dos atores é definida de acordo com a quantidade de dados fornecidos ou recebidos pelo ator no caso de uso que está sendo classificado (Tabela 15). A complexidade total dos atores é calculada através da equação 18.

$$TPA = \sum_{i=1}^{n} CA_i \quad (18)$$

2. *Total Points of the Precondition (TPPrC)*: cada pré-condição de um caso de uso tem sua complexidade (CPrC) determinada de acordo com o número de expressões lógicas testadas na condição (Tabela 16). A complexidade total das pré-condições é calculada através da equação 19.

$$TPPrC = \sum_{i=1}^{n} CPrC_i \quad (19)$$

3. *Principal Complexity Point (PCP)*: a complexidade do cenário principal é obtida através do somatório do número de entidades com o número de passos elementares necessários para sua conclusão, aplicando a Tabela 17.

4. *Total Points Complexity of the Alternative Scenarios (TPCA)*: a complexidade de cada cenário alternativo é obtida de forma similar ao cenário principal. Desta forma, cada cenário alternativo recebe uma quantidade de pontos (PCA) de acordo com a Tabela 17. A complexidade total dos cenários alternativos é calculada através da equação 20.

$$TPCA = \sum_{i=1}^{n} PCA_i \quad (20)$$

5. *Total Points Exceptions (TPE)*: cada exceção descrita no caso de uso tem sua complexidade (CE) determinada de acordo com o número de expressões lógicas testadas na

condição (Tabela 16). A complexidade total das exceções é calculada através da equação 21.

$$TPE = \sum_{i=1}^{n} CE_i \quad (21)$$

6. *Total Points of the Postconditions (TPPoC)*: cada pós-condição de um caso de uso tem sua complexidade (CPoC) determinada de acordo com o número de expressões lógicas testadas na condição (Tabela 16). A complexidade total das pós-condições é calculada através da equação 22.

$$TPPoC = \sum_{i=1}^{n} CPoC_i \quad (22)$$

Uma vez obtidas as variáveis descritas acima, o tamanho do caso de uso não ajustado (UUSP) é calculado através da equação 23.

$$UUSP = TPA + TPPrC + PCP + TPCA + TPE + TPPoC \quad (23)$$

Da mesma forma que nos métodos de pontos de função e pontos de caso de uso, o tamanho dos casos de uso pode ser ajustado em função dos fatores técnicos e ambientais. As equações 11 e 13 podem então ser aplicadas para determinação das variáveis TCF e EF. Uma vez definidos os fatores de ajuste o tamanho final do caso de uso pode calculado através da equação 24.

$$USP = UUSP * (TCF - EF) \quad (24)$$

O tamanho da aplicação é obtido através da equação (25).

$$Tamanho\ Aplicação = \sum_{i=1}^{n} USP_i \quad (25)$$

Tabela 15 - USP- Classificação dos Atores

Complexidade	Quantidade de Dados	UUSP
Simples	≤ 5	2
Médio	$6\ a\ 10$	4
Complexo	≥ 11	6

Tabela 16 - USP - Classificação das Pré e Pós-condições e Exceções

Complexidade	CPrC	CE	PCA	UUSP
	Expressão Avaliada		Entidades	
Simples	1 expressão lógica		≤ 3	1
Médio	2 ou 3 expressões lógicas		$4\ a\ 6$	2
Complexo	mais que 3 expressões lógicas		≥ 7	3

Tabela 17 - USP - Classificação dos Cenários

Complexidade	Quantidade de Dados	UUSP
Muito Simples	≤ 5	4
Simples	$6\ a\ 10$	6
Médio	$11\ a\ 15$	8
Complexo	$16\ a\ 20$	12
Muito Complexo	≥ 11	16

3.2.2 Fuzzy Use Case Size Point (FUSP)

Este método é uma variação do Use Case Size Point, onde as tabelas de classificação das complexidades dos atributos (variáveis) são transformadas em uma classificação contínua através de um processo de fuzificação e defuzificação.

O processo de fuzificação possibilita a geração de um número de fuzzy trapezoidal para cada categoria de complexidade encontrada nas tabelas de classificação (Figura 4). Com isso cada uma das tabelas é representada em um gráfico (Figura 5).

•

Table	m1	n1	a1	b1	m2	n2	a2	b2	m3	n3	a3	b3	m4	n4	a4	b4	m5	n5	a5	b5
1	1	3.5		6	6	8.5	3.5	11	11		8.5									
2	1	1.5		2	2	3	1.5	4	4		3									
3	1	3.5		6	6	8.5	3.5	11	11	13.5	8.5	16	16	18.5	13.5	21	21		18.5	
4	1	1.5		2	2	3	1.5	4	4		3									
5	1	2.5		4	4	5.5	2.5	7	7		5.5									

Figura 4 - Valores para fuzificação.
Fonte: (BRAZ e VERGILIO, 2006)

Depois do processo de fuzificação, é realizado o processo de defuzificação, onde os números de fuzzy são traduzidos novamente para valores do "mundo real". O processo de defuzificação envolve o cálculo da função de pertinência (x) (*membership function*) que representa a classificação de pertinência do elemento x na categoria de

complexidade. Este processo é construído sobre a aplicação de duas regras:

1. Se o número a ser classificado estiver entre o valor de m_ie $n_i$do número de fuzzy correspondente, o valor será o mesmo da categoria que este pertence. Isso ocorre quando o valor está na base superior do trapézio. Neste caso a função de pertinência (x) é igual a 1, gerando o mesmo valor que o método USP geraria.

2. Quando o número a ser classificado estiver entre os valores de n_ie $b_i$do número de fuzzy correspondente, significa que o elemento está classificado entre dois números de fuzzy, pois quando isso acontece o número também se encontra entre a_{i+1} e m_{i+1}. Quando isso ocorre é necessário calcular o grau de pertinência que o número tem com cada número de fuzzy correspondente, ou seja, com cada categoria representada pelos números. A equação 26 pode ser aplicada para determinação deste número.

$$dFUSP(x) = (x) * USP_i + (x) * USP_{i+1} (26)$$

$Onde$:
$dFUSP(x) = valor\ obtido\ no\ processo\ de\ defuzificação$
USP_i é $o\ valor\ de\ USP\ para\ a\ categoria\ i$
USP_{i+1} é $o\ valor\ de\ USO\ para\ a\ categoria\ i+1$

Este processo é repetido para a avaliação da complexidade de cada categoria. O valor obtido no processo de fuzificação é então utilizado como substituto ao valor absoluto correspondente à classificação da categoria que está sendo analisada. O restante do método é idêntico ao USP.

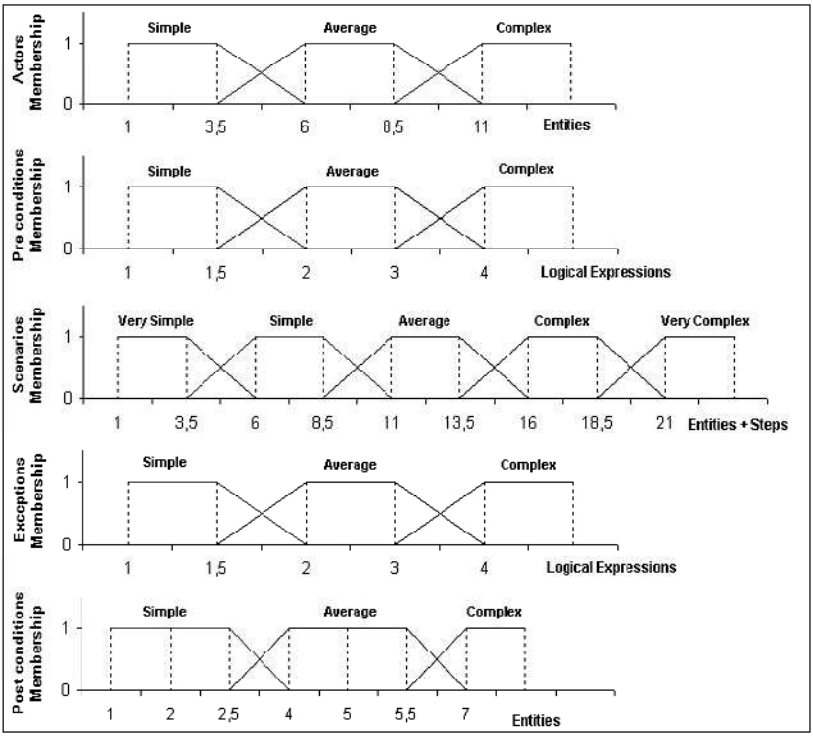

Figura 5 - Números Fuzzy das tabelas de classificação do método USP.
Fonte: (BRAZ e VERGILIO, 2006)

3.2.3 Pontos Limitantes

O método USP tem por principal objetivo eliminar a avaliação de complexidade dos casos de uso e tornar os casos de uso a entidade principal do processo de medição, atribuindo a estes valores absolutos de tamanho.

No entanto, para a determinação do tamanho de um caso de uso, o método depende da análise de uma série de variáveis, que só são obtidas quando os casos de uso estão no último nível de detalhe, o que muitas vezes não ocorre nas fases iniciais dos projetos.

Além disso, as variáveis utilizadas são classificadas em categorias de complexidade. Como visto anteriormente, o uso de escalas ordinais (classificações de complexidade) dificulta a obtenção de uma relação direta entre uma unidade de medida de tamanho derivada destas classificações com esforço relacionado para seu desenvolvimento.

O método FUSP visa cobrir esta lacuna, transformando as tabelas de classificações de complexidades de uma escala ordinal para uma escala contínua através das técnicas de fuzificação e defuzificação. Porém, acrescenta a complexidade da utilização de lógicas nebulosas (Fuzzy) para obtenção dos fatores de complexidades das variáveis.

3.3 ESTIMATIVAS DE CUSTO USANDO E-UCP

Em (PERIYASAMY e GHODE, 2009), é apresentada uma extensão do método de estimativas de pontos de caso de uso (KARNER, 1993) chamado e-UCP (*Extended Use Case Point*). O e-UCP utiliza todos os aspectos relacionados com os casos de uso, como atores, casos de uso, associações entre atores e casos de uso, relacionamento entre atores, relacionamento entre casos de uso e a narrativa detalhada de cada caso de uso. Para fins de aplicação do método e-UCP, os autores sugerem a utilização de um modelo de caso de uso (Tabela 18).

De forma resumida, o método e-UCP consiste em classificar os atores (Tabela 19), classificar os casos de uso (Tabela 20) e contabilizar os pesos de todas as narrativas para cada de uso (Tabela 21). O cálculo da quantidade de pontos de caso de uso não ajustados (UUCP) é obtido através do somatório dos pesos de cada ator, caso de uso e parâmetros das narrativas dos casos de uso, conforme equação 27.

$$UUCP_{total} = UUCP_{actor} + UUCP_{use\ case} + UUCP_{parametros} \quad (27)$$

Tabela 18- e-UCP - Modelo de Caso de Uso

Seção	Descrição da Seção
Nome do Caso de Uso	Um nome descritivo e representativo do caso de uso
Objetivo	Uma breve descrição do objetivo do caso de uso.
Parâmetros de Entrada	Lista dos parâmetros de entrada do caso de uso
Parâmetros de Saída	Lista dos parâmetros de saída do caso de uso
Atores Principais	Lista dos principais atores que utilização o caso de uso
Outros Atores	Lista de outros atores que utilizam o caso de uso
Precondições	Cláusulas das condições necessárias para entrada no caso de uso
Pós-condições	Condições que devem estar atendidas ao final da execução do caso de uso
Cenários de Sucesso	Sequência de instruções que explicam o caminho de sucesso do caso de uso
Exceções	Conjunto de exceções que podem acontecer ao longo da execução de um caso de uso
Inclusões	Lista de outros casos de uso que são incluídos no caso de uso descrito
Inclusões Por	Lista de casos de uso que incluem o caso de uso descrito
Extensões	Lista de casos de uso que o caso de uso descrito estende.
Extensões Por	Lista de casos de uso que estendem o caso de uso descrito
Observações adicionais	Observações diversas em relação ao caso de uso.

Da mesma forma que nos métodos de pontos de função e pontos de caso de uso, o valor do UUCP pode ser ajustado em função dos fatores técnicos e ambientais. As equações 11 e 13 podem então ser aplicadas para determinação das variáveis TCF e EF. Uma vez definidos os fatores de ajuste, o tamanho final do projeto pode ser obtido através da Equação 28.

$$e\text{-}UCP = UUCP_{total} * TCF * EF \quad (28)$$

Tabela 19 - e-UCP - Classificação dos Atores

Complexidade	Regra de Classificação	UUCP
Muito Simples	Ator especializado (primário ou secundário)	0, 5
Simples	Ator primário com $1 < número\ de\ associações \leq 3$	1
Menor que Médio	Ator primário com $3 < número\ de\ associações \leq 5$	1, 5
Médio	Ator primário com mais de 5 associações	2
	Ator secundário com apenas 1 associação	2
Complexo	Ator secundário com $1 < número\ de\ associações \leq 3$	2, 5
Muito Complexo	Ator secundário com $3 < número\ de\ associações \leq 5$	3
Muito Mais Complexo	Ator secundário com mais de 5 associações	3, 5

Tabela 20 - e-UCP - Classificação dos Casos de Uso

Complexidade	Regra de Classificação	UUCP
Simples	$número\ de\ associações \leq 2$	0, 5
Médio	$2 < número\ de\ associações \leq 4$	1
Complexo	$4 < número\ de\ associações \leq 6$	2
Muito Complexo	$número\ de\ associações > 6$	3

Tabela 21 - e-UCP - Pesos dos Parâmetros das Narrativas

Parâmetro da Narrativa do Caso de Uso	UUCP
Parâmetro de Entrada	0, 1
Parâmetro de Saída	0, 1
Um predicado na Precondição	0, 1
Um predicado na Pós-condição	0, 1
Uma ação no cenário principal	0, 2
Uma exceção	0, 1

3.3.1 Pontos Limitantes

Com o mesmo objetivo dos outros métodos descritos neste capítulo, o método e-UCP apresenta uma abordagem de estimativa baseada nos atores e casos de uso com seus relacionamentos e

associações. Ele leva também em consideração os parâmetros de entrada e de saída dos casos de uso, bem como suas pré e pós-condições, fluxos de exceção e ações do cenário principal do caso de uso.

No entanto, para a aplicação do método é necessário que os casos de uso estejam detalhados com quase todas as seções do modelo descrito na Tabela 18.

Além disso, a avaliação da quantidade de predicados das pré e pós-condições, bem como a avaliação das ações do cenário principal dos casos de uso, tornam a aplicação do método muito dependente da forma como os casos de uso estão escritos.

Esta limitação somada com a necessidade de um grande detalhamento dos casos de uso dificulta a aplicação deste método nas fases iniciais dos projetos, bem como no estabelecimento de contratos para terceirização de serviços de desenvolvimento de software.

3.4 PASSOS OBRIGATÓRIOS COMBINADO COM PCU

Em (VIEIRA, 2007), o autor apresenta 3 abordagens para estimativas de projetos (com algumas variantes) baseadas em casos de uso, que podem ser aplicadas em diferentes fases do projeto.

A primeira abordagem de estimativa visa sua aplicação nas fases iniciais do projeto onde existem poucas informações sobre os casos de uso. Para as estimativas nesta fase, o autor sugere que a estimativa de tempo seja realizada através da multiplicação do tempo médio de desenvolvimento de um caso de uso ($T_{médio_{uc}}$) pelo número de casos de uso (N) do sistema. Para isso, é necessário que a empresa disponha de uma base histórica de desenvolvimento que permita a obtenção de $T_{médio_{uc}}$. A Equação 29 mostra a fórmula deste cálculo.

$$Tempo_{estimado} = N * T_{médio_{uc}} \quad (29)$$

Ainda nesta fase, caso o analista consiga atribuir um valor de complexidade aos casos de uso ($complex(uc)$) variando de 0 a ∞, as estimativas poderiam ser obtidas através da multiplicação do $T_{médio_{uc}}$ pelo somatório das complexidades de cada caso de uso, como mostra a Equação 30.

$$Tempo_{estimado} = T_{médio_{uc}} * \sum_{i=1}^{N} complex(uc_i) \quad (30)$$

O autor ainda apresenta outra alternativa que leva em consideração uma estimativa de tempo para as atividades que independem dos casos de uso (TFI) e um fator de aumento (f) do número de casos de uso do projeto, como mostra a Equação 31.

$$Tempo_{estimado} = TFI + T_{médio_{uc}} * f * \sum_{i=1}^{N} complex(uc_i) \quad (31)$$

Em uma segunda abordagem, o autor sugere a redefinição da função $complex(uc)$ através da análise dos casos de uso expandidos. A técnica leva em consideração o detalhamento dos casos de uso classificando cada passo dos fluxos em passos obrigatórios e passos complementares, conforme definido por (WAZLAWICK, 2004), levando em consideração apenas as transações obrigatórias (to) dos fluxos do caso de uso (Equação 32).

$$complex(uc) = f_{uc} * \sum_{i=1}^{N_{uc}} complex_{uc}(to_i) \quad (32)$$

$Onde$:
$complex_{uc}(to)$ é $a\ função\ que\ estabelece\ a\ medida\ de\ complexidade\ de\ uma\ transação\ orbigatória$
N_{uc} é $o\ número\ de\ transações\ obrigatórias\ do\ caso\ de\ uso\ uc$
f_{uc} é $o\ fator\ de\ aumento\ esperado\ no\ número\ de\ transações\ obrigatórias\ do\ caso\ de\ uso\ uc\ nas\ etapas\ posteriores$

Aplicando a equação 32 em 31, obtemos a equação 33.

$$Tempo_{estimado} = TFI + T_{médio_{uc}} * f \sum_{i=1}^{N} \left(f_{uc} \sum_{j=1}^{N_{uc_j}} complex_{uc}(to_j) \right) \quad (33)$$

Se f_{uc} for definido como a média histórica, a equação 33 pode ser reescrita pela equação 34.

$$Tempo_{estimado} = TFI + T_{médio_{uc}} * f * f_{uc} \sum_{i=1}^{N} \sum_{j=1}^{N_{uc_j}} complex_{uc}(to_j) \quad (34)$$

Outra simplificação também pode ser realizada se for considerado que todos os passos obrigatórios são igualmente complexos e expressos por um valor médio α. Desta forma, a equação 34 poderia ser reduzida na equação 35.

$$Tempo_{estimado} = TFI + T_{médio_{uc}} * f * f_{uc} * \alpha * N_{uc} \quad (35)$$

$Onde$:
N_{uc} é $o\ número\ total\ de\ passos\ obrigatórios\ nos\ casos\ de\ uso\ de\ um\ sistema\ obtido\ por\ \sum_{i=1}^{N} N_{uc_i}$

Na terceira abordagem o autor considera a definição dos contratos de cada operação e consulta de sistema (transações obrigatórias) (LARMAN, 2004). Com os contratos definidos é possível detalhar mais a estimativa de complexidade individual ($complex_{uc}(to)$) de cada operação ou consulta de sistema.

Uma transação obrigatória corresponderá a uma operação ou consulta de sistema (LARMAN, 2004) com uma assinatura (onde são especificados os parâmetros). Cada operação ou consulta de sistema implica na existência de um ou mais contratos.

Cada contrato possui pré-condições e pós-condições (se for operação de sistema) ou resultados (se for consulta de sistema). Todos estes elementos poderão afetar a complexidade da transação.

Os atributos a serem considerados para determinação da complexidade de transação são:

- $n_{par} = número\ de\ parâmetros$
- $n_{pre} = número\ de\ pré - condições$
- $n_{pos} = número\ de\ pós - condições$
- $n_{res} = número\ de\ resultados$

Cada um destes atributos terá alguma influência na complexidade da transação de sistema. Porém o peso de cada valor pode ser diferente. É necessário, portanto, não apenas somar os valores, mas obter uma soma ponderada com pesos que devem ser determinados experimentalmente. A equação 36 apresenta como pode ser calculada a complexidade dos casos de uso em função das transações obrigatórias ($complex_{uc}(to)$) levando em consideração os atributos dos contratos das operações e consultas do sistema.

$$complex_{uc}(to) = p1 * n_{par}(to) + p2 * n_{pre}(to) + p3 * n_{pos}(to) + p4 * n_{res}(to) \quad (36)$$

Assim, na fase final da análise, quando os contratos já foram elaborados, a estimativa de esforço total (Equação 37) pode ser definida a partir da aplicação da equação 36 em 34.

$$Tempo_{estimado} = TFI + T_{médio_{uc}} * f * f_{uc} \sum_{i=1}^{N} \sum_{j=1}^{N_{uc_j}} [p1 * n_{par}(to) + p2 * n_{pre}(to) + p3 * n_{pos}(to) + p4 * n_{res}(to)] \quad (37)$$

3.4.1 Pontos Limitantes

As abordagens propostas pelo autor possuem foco na estimativa direta do tempo (em horas) para desenvolvimento de um sistema, e não na mensuração do tamanho dos casos de uso, ou do tamanho total do projeto. Para realização das estimativas é levado em consideração o tempo médio para desenvolvimento de um caso de uso.

A primeira abordagem proposta pelo autor (e suas variantes) exige uma base histórica para estimativa do tempo médio para desenvolvimento de um caso de uso. Além disso, depende de uma avaliação do analista para classificação da complexidade dos casos de uso.

Já a segunda abordagem exige que os casos de uso estejam expandidos, e que suas transações sejam classificadas em transações obrigatórias e complementares.

A terceira abordagem exige um detalhamento ainda maior dos casos de uso, fazendo com que as operações e consultas de sistema estejam expressas sob a forma de contratos (um ou mais contratos para cada transação obrigatória).

De maneira geral, as abordagens utilizadas pelo autor inviabilizam o uso deste método em processos licitatórios, pois exigiria que o objeto da licitação fosse medido em horas, o que é não é recomendado pelo Tribunal de Contas da União.

Além disso, o tempo médio de desenvolvimento de um caso de uso é muito dependente do contexto em que empresa, para qual a solução está sendo desenvolvida, está inserida. Isso dificulta ainda mais sua utilização em processos licitatórios, pois inviabilizaria a predição de valores e formação de custo pelas empresas participante da licitação.

Outro ponto limitante é a necessidade de detalhamento dos casos de uso (segunda e terceira abordagens). O detalhamento dos casos de uso muitas vezes não é realizado nas fases iniciais dos projetos onde há a necessidade de estabelecimentos dos contratos.

3.5 CONSIDERAÇÕES FINAIS SOBRE OS TRABALHOS RELACIONADOS

Este capítulo apresentou alguns trabalhos que apresentam abordagens para medição e estimativa de projetos de software com base em casos de uso. A maioria dos trabalhos relacionados apresentam uma forma de medição dos casos de uso que é baseada no detalhamento de cada caso de uso, ou seja, para sua aplicação é necessário que os casos de uso estejam muito detalhados e com todas suas seções, fato este que muitas vezes não ocorre nas fases iniciais dos projetos, quando os contratos de terceirização são estabelecidos.

De forma geral, as abordagens propostas pelos autores apresentam algum tipo de limitação que dificultam o seu uso em processos licitatórios e de terceirização do desenvolvimento de software.

Por este motivo, este trabalho apresenta um novo método de medição e estimativa de projetos de software que será descrito no capítulo a seguir.

4 FULL USE CASE SIZE (FUCS)

MEDIÇÃO E ESTIMATIVA DE SOFTWARE COM BASE NO TAMANHO DE CASOS DE USO

Este capítulo apresenta detalhes sobre o funcionamento do método proposto neste trabalho para medição e estimativa de software baseado no tamanho de seus casos de uso, seus conceitos fundamentais e suas fases e etapas de aplicação.

4.1 VISÃO GERAL DO FUCS

Este trabalho apresenta um novo método para medição e estimativa do tamanho de um projeto de software baseado no tamanho de cada caso de uso, chamado FUCS (*Full Use Case Size*). O termo "medição" é utilizado neste trabalho como sendo o processo definido pelo FUCS para obtenção da medida do tamanho do software. O termo "estimativa" é utilizado para fazer referência ao processo de previsão do esforço para desenvolvimento do software a partir das medidas obtidas com a aplicação do método FUCS.

No FUCS, o tamanho de um projeto de software é medido em função do tamanho de seus casos de uso, que por sua vez são medidos em função dos seus atributos: tipo do caso de uso, quantidade de requisitos de interface com o usuário e quantidade de operações básicas e regras de negócio associadas. As medidas dos tamanhos dos casos de uso e o tamanho do projeto são dadas sob uma nova métrica chamada UCS (*Use Case Size*).

Com o intuito de minimizar as discrepâncias nos resultados das avaliações dos atributos do método quando realizadas por pessoas diferentes, foram utilizadas escalas objetivas para medição dos atributos dos casos de uso (entidades), bem como algumas regras para medição destes atributos. Para a medição do atributo tipo do caso de uso é utilizado uma escala tipológica e, associada a esta, uma escala razão que determina um tamanho inicial de um caso de uso. Nos casos dos requisitos de interface do usuário e das operações básicas e regras de negócio, a escala utilizada é a razão.

A vantagem do uso de escalas objetivas no processo de medição está na possibilidade de realização de operações matemáticas de forma direta, sem a necessidade de transformações de dados, tornando assim o processo de medição mais objetivo e livre de interpretações variadas.

Para a utilização do FUCS não é necessário uma documentação detalhada do sistema. Casos de uso de alto nível e uma lista de requisitos do sistema são suficientes para medir e estimar um projeto com o FUCS. Isso faz com que o FUCS possa ser aplicado tanto nas fases iniciais do projeto, quando os casos de uso não estão ainda detalhados, quanto ao longo de todas as iterações do desenvolvimento e na medida em que novos requisitos são descobertos.

Diferentemente do método de pontos de função, o FUCS propõe a medição do software não somente pelo aspecto funcional (requisitos funcionais[10]), mas também leva em conta aspectos não funcionais (requisitos não funcionais[11]) inerentes à complexidade do negócio e ao processo de desenvolvimento da empresa. Por este motivo, não é possível classificá-lo como um método de medição funcional.

A Figura 6 apresenta um modelo genérico de medição funcional de software. Como pode ser visto, o processo de medição é composto basicamente de duas fases: de mapeamento e de avaliação. A fase de mapeamento consiste em aplicar os conceitos e definições do método aos artefatos do software a ser medido, extraindo os atributos a serem contados, resultando então em uma instância do processo. Na fase de avaliação, os atributos extraídos são medidos/contados de acordo com as regras e procedimentos estabelecidos no método, resultando então, na medida de software.

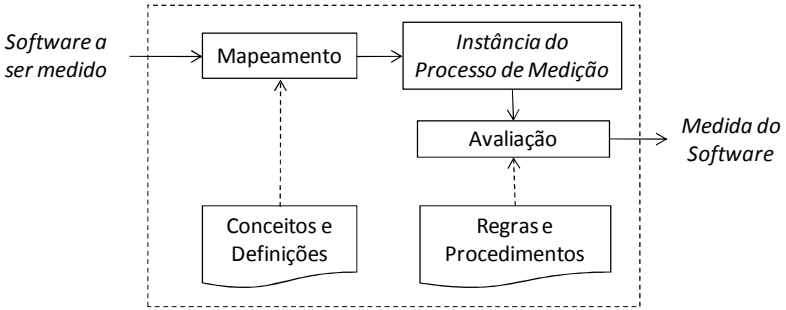

Figura 6 - Modelo Genérico de Medição Funcional.
Fonte: adaptado de (EBERT, DUMKE, et al. , 2005)

[10]Requisitos funcionais: são as declarações de serviços que o sistema fornecer, como sistema deve reagir a entradas específicas e como sistema deve se comportar em determinadas situações. (SOMMERVILLE, 2007)

[11]Requisitos não funcionais: são restrições sobre os serviços ou as funções oferecidas pelo sistema. Incluem restrições sobre o processo e padrões de desenvolvimento da empresa. (SOMMERVILLE, 2007)

Apesar do FUCS não poder ser classificado como um método de medição funcional, pelo fato de levar em consideração aspectos não-funcionais, seu processo segue o mesmo modelo genérico de medição funcional, apresentando também uma fase de mapeamento e uma de avaliação. A Figura 7 apresenta uma visão geral do processo de medição utilizado no FUCS.

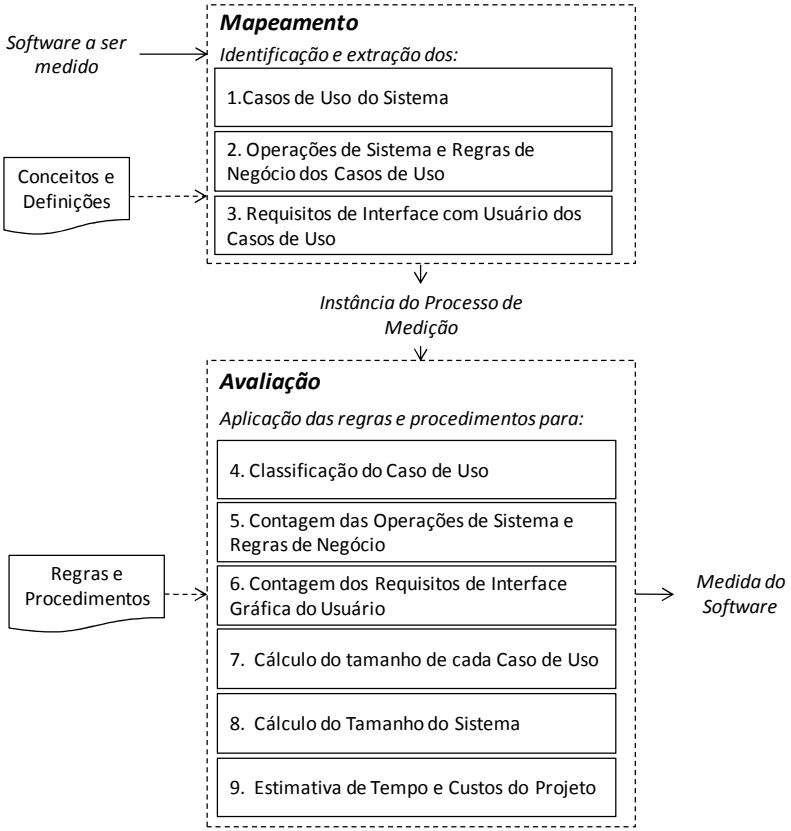

Figura 7 - Modelo do processo de medição do FUCS

A principal entidade envolvida no FUCS são os casos de uso. A fase de mapeamento consiste basicamente em analisar a documentação do projeto e identificar os casos de uso da aplicação juntamente com suas regras de negócio e requisitos de interface com o usuário. Uma vez levantados estes atributos podemos obter uma instância do processo de medição. A instanciação do processo de medição é composta pelo

conjunto de configurações do método juntamente com o conjunto dos atributos extraídos no processo de mapeamento.

A fase de avaliação consiste em aplicar as regras e procedimentos para medir/contar os atributos extraídos na fase de mapeamento, com base nas configurações definidas no processo de instanciação. No FUCS, o resultado do processo de avaliação são as medidas dos casos de uso e do projeto como um todo em UCS (*Use Case Size*).

Diferentemente do método de pontos de caso de uso (KARNER, 1993), o FUCS não aplica, ao final do processo de medição, o ajuste em função dos fatores técnicos e ambientais. Como o intuito do FUCS é permitir sua aplicação em processos de terceirização, as características ambientais são de responsabilidade das empresas contratadas. Já os fatores técnicos tornam-se desnecessários, pois já são considerados dentro do processo de medição definido pelo FUCS.

Nas subseções deste capítulo serão descritos todos os passos e conceitos envolvidos nas fases de mapeamento e de avaliação do processo de medição de software proposto.

4.2 CONCEITOS E DEFINIÇÕES DO FUCS

Diferentemente do método original de contagem de pontos de caso de uso (KARNER, 1993), onde o tamanho do sistema é baseado na classificação de complexidade de seus casos de uso, o FUCS define uma forma de medição do tamanho do projeto em função do tamanho de cada caso de uso. Esta medição é baseada na avaliação de três conceitos e informações relacionados a cada caso de uso:

- Tipo do Caso de Uso
- Número de Operações de Sistema e Regras de Negócio
- Número de Requisitos de Interface

A seguir serão definidos estes conceitos para posterior aplicação nas fases de mapeamento e avaliação do método proposto.

4.2.1 Tipos de Casos de Uso

O dinamismo dos negócios do mercado atual tem exigido respostas mais rápidas das áreas de TI, principalmente na criação de sistemas de informação para atender às necessidades de negócio das empresas.

A criação e utilização de frameworks e a padronização dos processos de desenvolvimento contribuem de forma significativa com este objetivo. Empresas e/ou áreas desenvolvedoras de software tem buscado cada vez mais padronizar seus processos de desenvolvimento de sistemas para suportarem este dinamismo e se manterem concorrentes no mercado.

A padronização dos tipos de casos de uso é uma das alternativas que as empresas tem encontrado para conseguir desenvolver suas aplicações de maneira mais rápida e consistente. Um caso de uso padronizado permite que a especificação, a construção e os testes do sistema sejam realizados de forma mais rápida e consistente, pois podem apoiar-se em ferramentas e frameworks que aumentam a produtividade e, consequentemente, diminuem o esforço de desenvolvimento.

Em (COCKBURN, 2005), Cockburn discute o que chama de “Casos de Uso CRUD” e também “Casos de Uso Parametrizados”. Segundo ele, estes tipos de casos de uso são muito comuns nos sistemas de informação e podem ser padronizados.

De forma geral, um tipo de caso de uso pode ser definido como sendo um padrão estabelecido para documentação e implementação de casos de uso que possuem forma e estrutura comuns entre si. Um tipo de caso de uso pode definir cenários principais padronizados e permitir a especialização através de pontos de extensão.

Os tipos de casos de uso definidos neste trabalho são extensões destes dois tipos de casos de uso especiais definidos por Cockburn. A definição de um conjunto de tipos de casos de uso tem por objetivo final permitir a avaliação da complexidade de cada tipo em função dos padrões, processos e ferramentas de cada empresa, e sua influência no tamanho do caso de uso. É importante destacar que o conjunto de tipos de casos de uso definidos neste trabalho pode (e deve) ser reavaliado para cada empresa que pretenda aplicar o FUCS para a medição e estimativa de seus projetos. É de fundamental importância que os tipos de casos de uso definidos reflitam de forma adequada aos padrões de casos de uso utilizados no contexto de aplicação do método.

A Figura 8 ilustra os tipos de caso de uso sugeridos no método FUCS, os quais estão descritos nas subseções seguintes.

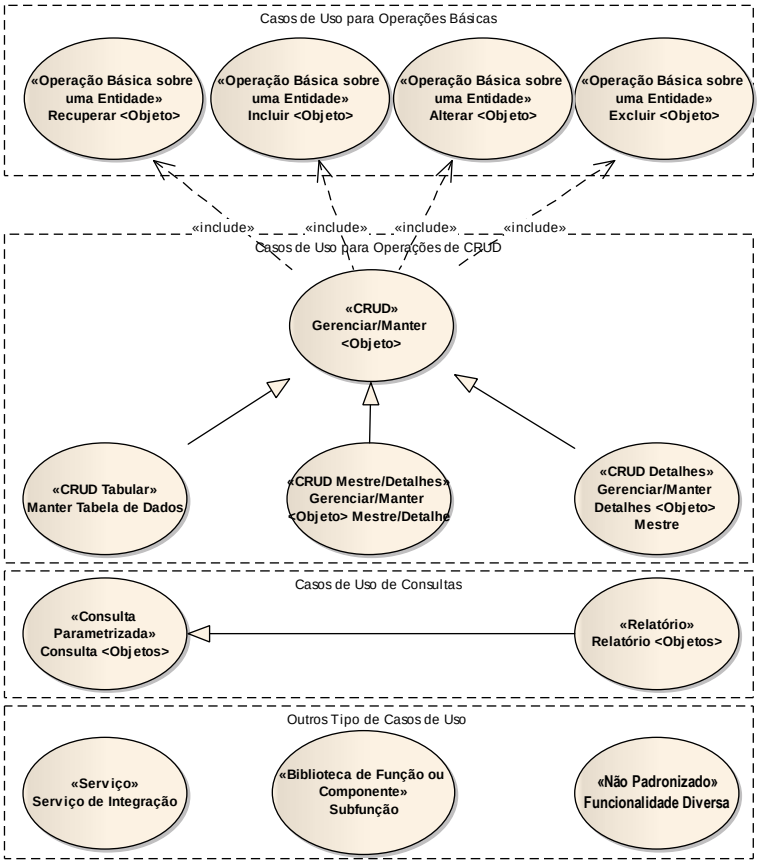

Figura 8 - Tipos de Casos de Uso

4.2.1.1 Tipo “Operação Básica sobre uma Entidade”

Estes tipos de casos de uso realizam apenas uma das operações básicas de CRUD (*Create*, *Retrieve*, *Update*, *Delete*) sobre um objeto. São operações do tipo “Incluir <objeto>”, “Excluir <objeto>”, “Alterar <objeto>” e “Recuperar <Objeto>”.

Estes tipos de caso de uso correspondem à menor unidade funcional de um sistema, que processa dados de fora para dentro ou de dentro para fora da fronteira da aplicação. Seu principal objetivo é manter (incluir, alterar, excluir) dados de um objeto ou recuperar um objeto do sistema. São exemplos deste tipo de caso de uso:

- Cadastrar cliente;
- Excluir cliente;
- Atualizar dados do cliente.

4.2.1.2 Tipo “Consulta Parametrizada”

Este tipo de caso de uso se refere às situações recorrentes de consultas diversas nos sistemas, e que levam a uma proliferação de cenários “quase idênticos”. Estes tipos de caso de uso possivelmente contemplarão uma entrada do usuário e uma resposta do sistema que retorna uma lista de objetos que atendem aos critérios especificados pelos usuários. Estes casos de uso são então idênticos e diferem apenas na entidade que está tratando e nos parâmetros (critérios) que são passados para realizar a pesquisa. Por este motivo são chamados de casos de uso parametrizados.

Este tipo de caso de uso difere do tipo “Operação Básica sobre uma Entidade – *Recuperar <objeto>*”, pois neste podem ser considerados um conjunto de parâmetros de pesquisa, enquanto no tipo “*Recuperar <objeto>*” um único parâmetro é considerado – o identificador do objeto no sistema de persistência da aplicação.

Provavelmente, durante a implementação do sistema, uma equipe de desenvolvimento criará um mecanismo de pesquisa genérico que atenderá a todos os casos de uso deste tipo, ou até mesmo será feito uso de algum framework que implemente estas operações de forma genérica, onde somente os pontos de extensão serão customizados. São exemplos deste tipo de caso de uso:

- Consultar cliente por nome;
- Consulta genérica de clientes: nome, endereço, profissão.

4.2.1.3 Tipo “Relatório”

Este tipo de caso de uso é uma especialização do tipo “Consulta Parametrizada”, onde o resultado da pesquisa é renderizado como um relatório, podendo incluir quebras de seção e totalizadores. Estes também são casos de uso muito comuns em sistemas de informação. São exemplos deste tipo de caso de uso:

- Relatório de Estoque de Produtos;
- Relatório de Clientes.

4.2.1.4 Tipo “CRUD”

Este tipo de caso de uso trata de uma colaboração de todos os casos de uso menores do tipo “Operação Básica sobre uma Entidade”. Estes tipos de casos de uso são muito comuns em sistemas de informação e devem ser utilizados no lugar de quatro operações básicas (uma para incluir, uma para alterar, uma para excluir e outra para recuperar) quando os atores que realizam as operações básicas são comuns a todas elas. São exemplos deste tipo de caso de uso:

- Manter cadastros de clientes;
- Manter catálogo de produtos.

4.2.1.5 Tipo “CRUD Tabular”

Este tipo de caso de uso é uma especialização do tipo “CRUD” onde todos os objetos de um mesmo tipo são gerenciados ao mesmo tempo. O gerenciamento destes objetos é feito de forma muito similar a uma tabela de dados. Normalmente é utilizado para cadastros mais simples, com poucos registros e poucos atributos. São exemplos de situações que podem ser aplicados este tipo de caso de uso:

- Gerenciamento da tabela de unidades da federação;
- Gerenciamento do catálogo de modelos de carros.

4.2.1.6 Tipo “CRUD Mestre / Detalhes”

Este tipo de caso de uso também é uma extensão do tipo “CRUD”. Porém, nestes tipos de casos de uso, as operações de CRUD são realizadas sobre um objeto principal (mestre) e seus objetos associados (detalhes), em uma única transação. Em outras palavras, este tipo de caso de uso possibilita o gerenciamento de um objeto principal juntamente com seus objetos associados. São exemplos deste tipo de caso de uso:

- Em um sistema de recursos humanos, o gerenciamento dos dados de um colaborador (objeto mestre) juntamente com suas especialidades (detalhes associados a colaborador);
- O gerenciamento de um catálogo de marcas de carros (objeto mestre) e seus modelos (detalhes associados às marcas).

4.2.1.7 Tipo “CRUD Detalhes”

Este tipo de caso de uso é uma especialização do tipo “CRUD Mestre/Detalhes”. A diferença é que neste tipo de caso de uso, o objeto principal (mestre) não é gerenciado, sendo possível somente o gerenciamento de seus detalhes. Estes casos de uso também são muito comuns em sistemas de informação e normalmente são utilizados em situações onde é necessário que determinados usuários não tenham acesso ao gerenciamento de um objeto principal, mas sim, ao gerenciamento de determinadas informações relacionadas ao objeto principal. Um exemplo deste tipo de caso de uso é o seguinte:

- Em um sistema de recursos humanos, o Gerenciamento do Histórico Profissional de um Colaborador, onde o objeto mestre (Colaborador) é somente-leitura, e é permitida somente a manutenção do histórico de cargos e salários do colaborador.

4.2.1.8 Tipo “Serviço”

Na medida em que a necessidade por integração entre sistemas cresce, torna-se necessário criar serviços que possibilitem esta integração de uma forma em que os sistemas mantenham-se desacoplados. Este tipo de caso de uso deve ser utilizado quando o ator do caso de uso é outro sistema. Nestes tipos de caso de uso, normalmente não existem interfaces gráficas. São exemplos destes tipos de casos de uso:

- *Webservices* de consulta de informações de um produto;
- Serviços *REST* [12] de atualização dos dados de um cliente;
- Geração e disponibilização de arquivo em diretório *FTP*[13] para integração com um sistema.

4.2.1.9 Biblioteca de Função ou Componente

De forma similar ao conceito de casos de uso de sub-funções, dado por Cockburn (COCKBURN, 2005), os casos de uso do tipo “Biblioteca de Função ou Componente” são aqueles que representam um sub-objetivo ou passo de um cenário, ou seja, aqueles casos de uso que podem ser utilizados por outros casos de uso. Normalmente eles não fazem parte do escopo funcional de um projeto, e aparecem ao longo da análise de requisitos, quando o analista verifica que um determinado sub-objetivo aparece em muitos casos de uso. Nestes casos, o analista pode extrair este sub-objetivo, gerando um caso de uso genérico e reutilizável do tipo “Biblioteca de Função ou Componente”.

Muitas vezes o desenvolvimento de um sistema exige a implementação de recursos específicos ao domínio do problema e que possam ser reutilizados por grande parte das funcionalidades do sistema. Estes recursos são costumeiramente desenvolvidos em bibliotecas de funções ou componentes, como forma de permitir seu reuso ao longo do desenvolvimento do sistema. São exemplos destes tipos de casos de uso:

- Biblioteca para cálculo das taxas de juros sobre débitos dos clientes;
- Componentes de conversão de moedas.

[12] Acrônimo de *Representational State Transfer*
[13] Acrônimo de *File Transfer Protocol*

4.2.1.10 Tipo “Não Padronizado”

Existem também os tipos de casos de uso que raramente aparecem em um sistema e que não há investimento em suas padronizações, uma vez que sua relação custo/benefício não é vantajosa. Normalmente são casos de uso mais complexos que tratam de informações variadas. Sugere-se que este tipo de caso de uso seja utilizado nas situações em que não é possível classificar um caso de uso em nenhum dos tipos definidos anteriormente.

No entanto, há de se ressaltar que o método descrito neste trabalho pode ser customizado às necessidades, ferramentas e padrões de cada empresa, sendo que uma das customizações possíveis é a configuração de um conjunto de tipos de caso de uso específicos para cada empresa, onde cada empresa pode criar seu próprio conjunto de tipos de caso de uso para atender as suas necessidades.

4.2.2 Operações de Sistema (OS) e Regras de Negócio (RN)

De forma geral, um caso de uso descreve uma sequência de passos que ocorrem durante a interação entre ator e sistema de forma a realizar uma tarefa em particular ou atingir um determinado objetivo. Na interação entre ator e sistema, o ator dispara alguma transação que resultam em uma resposta do sistema. Esta resposta do sistema é decorrente da ativação de uma operação de sistema.

Uma operação de sistema (OS) pode ser definida de forma similar ao conceito de Processo Elementar do método de Análise de Pontos de Função, como sendo:

> “a menor unidade de atividade que satisfaz todas as seguintes regras:
> - Tem significado para o usuário
> - Constitui uma transação completa
> - É autocontido
> - Mantém o negócio em um estado consistente”. (VAZQUEZ, SIMÕES e ALBERT, 2010)

Em (WAZLAWICK, 2004), são definidos os conceitos de operação e de consultas de sistema. Segundo o autor, uma operação ou

uma consulta de sistema indica um fluxo de informações de fora para dentro do sistema. A diferença entre operação e consulta de sistema está no fato de que as operações alteram as informações do sistema, enquanto as consultas correspondem a simples verificações de informações do sistema.

No contexto deste trabalho, não há a necessidade de se diferenciar as operações das consultas de sistemas. Desta forma, o conceito de operação de sistema utilizado neste trabalho abrange tanto os conceitos de operações quanto as consultas de sistemas.

Um tipo de caso de uso contém em sua definição uma ou mais operações de sistema pré-definidas. Os casos de uso do tipo "CRUD", por exemplo, possuem em sua definição 4 operações de sistema: incluir, alterar, excluir e recuperar um objeto.

Todo caso de uso pode ser classificado, segundo sua essência, em um tipo de caso de uso. No entanto, existem situações onde um caso de uso contém outras operações além daquelas normalmente definidas no seu tipo. Estas operações por sua vez podem impactar no tamanho do caso de uso, e por este motivo, devem ser consideradas em separado no processo de medição do software.

Um exemplo desta situação é um caso de uso do tipo "CRUD" para "Gerenciamento dos dados do usuário" que necessite de uma operação para "Gerar nova senha para o usuário". Esta operação não está contida em nenhuma operação pré-definida no tipo do caso de uso "CRUD" e, portanto, exigirá que seja definida e implementada, impactando no esforço de desenvolvimento do caso de uso.

Além disso, existem situações onde se torna necessária uma intervenção em uma operação pré-definida no tipo de caso de uso, para implementar alguma regra de negócio.

As regras de negócio (RN) são regras que especificam as características de software, como o seu comportamento, restrições e validações que restringem algum aspecto do negócio (AMBLER, 2004).

Considerando ainda como exemplo o caso de uso do tipo "CRUD" para "Gerenciamento dos dados do usuário", um requisito do tipo "*Ao incluir um novo usuário o sistema deverá gerar automaticamente uma senha e enviá-la para o email para o usuário*", é considerada uma regra de negócio.

As regras de negócio são tipicamente identificadas durante a análise do sistema e normalmente são descritas como requisitos não-funcionais. Outros exemplos de regras de negócios são:

- regras de cálculo: "Os juros são calculados a partir do 5 º dia útil após a data de validade, com uma aplicação da taxa diária de 1%";
- regras de processamento da informação: "Ao receber o catálogo de produtos de um fornecedor, o sistema irá iterar sobre os produtos para verificar se existem produtos com a tag 'Novo'. Se existirem, o sistema deve enviar um e-mail para o administrador informando sobre os novos produtos".

No contexto deste trabalho, e considerando que os casos de uso definem as funcionalidades sob o ponto de vista dos usuários, as regras de negócio são todos os requisitos solicitados pelos usuários que completam as operações de sistema dos casos de uso, tornando-as consistentes em relação ao negócio.

4.2.3 Requisitos de Interface Gráfica com o Usuário (RIGU)

Os requisitos de interface de usuário (RIGU) especificam as particularidades da interface gráfica do usuário de um determinado caso de uso. É importante ressaltar que estes não são requisitos gerais de um sistema, mas sim características e comportamentos específicos de uma interface gráfica de usuário para um determinado caso de uso.

Scott Ambler (2004) destaca a importância de se levar em consideração aspectos da interface gráfica do usuário quando diz:

> "Para os usuários, a interface de usuário é o sistema. Se você construir uma interface de usuário ineficaz para seu sistema, isso realmente não mostrará o quão bom é o resto do seu sistema: os usuários vão odiar o que você construiu para eles." (AMBLER, 2004).

Os requisitos de interface gráfica do usuário são fundamentais para o desenvolvimento de um software de qualidade. No entanto, muitas vezes sua implementação exige custos adicionais de desenvolvimento. De fato, quanto mais “rica” a interface gráfica do usuário, mais esforço exigirá para sua implementação.

Por este motivo, é muito importante que no processo de análise sejam identificadas as expectativas dos usuários em relação a como eles podem interagir com o sistema. Isso deve ser alcançado já no início do

projeto, facilitando a estimativa do esforço necessário para o desenvolvimento dos casos de uso.

Os requisitos de interface com o usuário podem ser especificados na medida em que os casos de uso são identificados. Uma possibilidade é utilizar técnica Essential UI (User Interface) de prototipagem (CONSTANTINE e LOCKWOOD, 1999).

São alguns exemplos de requisitos de interface:

- Carga automática de informações aninhadas: “Quando um usuário informar o CEP de cliente, o sistema deverá automaticamente buscar os dados do logradouro e exibi-los nos campos da tela”;
- Formatos de exibição de dados: “O usuário deseja que os produtos indisponíveis no estoque sejam destacados em vermelho”.

Apesar de terem grande impacto no esforço final para desenvolvimento de um software, nenhum dos métodos de medição e estimativa de software relacionados a este trabalho, apresentados nos capítulos 2 e 3, levam em consideração estes tipos de requisitos. Este é um dos diferenciais do método FUCS em relação aos demais, pois estes requisitos são considerados dentro do processo de medição estabelecido no FUCS e podem ter sua influência configurada para o contexto de cada empresa.

4.3 FASE DE MAPEAMENTO

O principal objetivo da fase de mapeamento do FUCS é identificar os casos de uso que compõem o escopo funcional do projeto a ser medido (ou estimado) e identificar os atributos necessários para medição do tamanho de cada caso de uso. Para identificação dos casos de uso e dos atributos, não é necessário uma documentação muito detalhada, bastando uma lista de requisitos e os casos de uso descritos em alto nível.

A fase de mapeamento pode ser subdividida em três passos, os quais serão descritos a seguir. Cada passo tem por finalidade identificar as informações de cada caso de uso e extrair os atributos a serem contados na fase de avaliação. As informações a serem identificadas na fase de mapeamento consistem na lista de casos de uso que compõem o escopo de medição juntamente com suas operações de sistema, regras de

negócio e requisitos de interface gráfica do usuário. As subseções seguintes mostram os passos para aplicação do método.

4.3.1 Passo 1: Identificar os Casos de Uso do Software

O primeiro passo consiste em identificar o escopo funcional do software a partir da documentação dos casos de uso da aplicação a ser medida. Um modelo de casos de uso é o artefato mais adequado para realização deste passo, visto que ele torna mais simples a visualização dos casos de uso do sistema. No entanto, esta atividade também pode ser realizada a partir de uma descrição geral do software, de uma lista de requisitos ou até mesmo de um documento de especificação de casos de uso. O resultado final da etapa de identificação dos casos de uso deve ser uma lista com todos os casos de uso da aplicação, vistos sob a ótica do usuário.

É muito importante nesta fase que o analista responsável pela medição esteja atento à identificação dos casos de uso dependentes. Na UML, existem diferentes tipos de relações de dependências entre casos de uso, em especial as de inclusão e extensão. Estes casos de uso também devem constar da listagem final dos casos de uso da aplicação.

4.3.2 Passo 2: Identificação das Operações de Sistema (OS) e Regras de Negócio (RN)

Na medida em que os casos de uso são identificados, é importante que o analista verifique quais as operações que compõem o caso de uso em questão e se existe alguma regra de negócio associada a cada operação.

Algumas perguntas, feitas em relação a cada caso de uso identificado, podem auxiliar nesta atividade:

1. Quais são a operações (tarefas) que o ator realiza neste caso de uso?

   Esta pergunta tem por objetivo identificar todas as operações de sistema realizadas no caso de uso avaliado. Um caso de uso do tipo “Manter cadastro de usuários do sistema” poderá incluir as seguintes operações:

- Incluir Usuário
- Alterar dados do Usuário
- Excluir Usuário
- Visualizar (recuperar) Usuário
- Gerar nova senha para o Usuário

2. Para cada uma das operações do sistema, existe alguma restrição de acesso ou regra de negócio associada?

   Com esta pergunta o analista poderá identificar eventuais restrições e regras de negócio associadas a cada caso de uso. Considerando ainda o exemplo anterior do caso de uso "Manter cadastro de usuários do sistema", poderiam surgir as seguintes restrições e regras de negócio:

   - Ao incluir um usuário o sistema deve gerar uma senha automaticamente e enviá-la para o email do usuário;
   - Um usuário só pode ser excluído pelo administrador do sistema;
   - Ao gerar uma nova senha, o sistema deverá enviá-la para o email do usuário.

O resultado desta etapa pode ser registrado em um artefato como o mostrado na Quadro 1.

Caso de Uso	**Operações de Sistema**	**Restrições e Regras de Negócio**
Manter cadastro de usuários do sistema	Incluir Usuário	RN1 - Ao incluir um usuário o sistema deve gerar uma senha automaticamente e enviá-la para o email do usuário.
	Alterar dados do Usuário	
	Excluir Usuário	RN2 - Um usuário só pode ser excluído pelo administrador do sistema.
	Recuperar Usuário	
	Gerar nova senha para o Usuário	RN3 - Ao gerar uma nova senha, o sistema deverá enviá-la para o email do usuário

Quadro 1 - Ex. de Registro das Atividades - Identificação das OSs e RNs

4.3.3 Passo 3: Identificação dos Requisitos de Interface Gráfica do Usuário (RIGU)

De maneira similar à identificação das operações de sistema e regras de negócio, o analista também deve identificar os requisitos de interface gráfica do usuário para cada caso de uso. É importante destacar que somente os requisitos específicos do caso de uso devem ser levantados. Requisitos gerais do sistema não devem ser considerados neste levantamento. Algumas perguntas, feitas em relação a cada caso de uso identificado, podem auxiliar nesta atividade:

1. Os atores do caso de uso desejam realizar alguma operação de maneira específica?

   O intuito desta pergunta é identificar se algum ator deseja operar o sistema de maneira específica para uma determinada operação. É comum aparecer requisitos específicos solicitados pelos usuários em relação à forma de operar o sistema. Estes tipos de requisitos devem ser registrados e associados ao caso de uso. São exemplos de requisitos que podem aparecer nesta situação:

   - No cadastramento de usuário, ao selecionar o estado do endereço, o sistema deve popular automaticamente a lista de cidades com as cidades do estado escolhido pelo usuário;
   - Ao digitar o número do CEP no cadastramento ou alteração dos dados do usuário, o sistema deverá popular automaticamente os demais campos do endereço do usuário.

2. O modo de apresentação gráfica das telas do caso de uso apresenta alguma especificidade?

   Com esta pergunta o analista pode identificar se existe algum requisito de apresentação específico do caso de uso, como ícones, recursos de separação dos dados em “abas” ou “guias” dinâmicas, dentre outras situações. Um exemplo deste tipo de requisitos é:

- Nas telas de alteração, exclusão e recuperação de usuário, o sistema deverá exibir ícone informativo de acordo com a situação do usuário.

O resultado desta etapa pode ser registrado em um artefato como o mostrado na Quadro 2.

Caso de Uso	**Requisitos de Interface Gráfica do Usuário**
Manter cadastro de usuários do sistema	RI1 - No cadastramento de usuário, ao selecionar o estado do endereço, o sistema deve popular automaticamente a lista de cidades com as cidades do estado escolhido pelo usuário.
	RI2 - Ao digitar o número do CEP no cadastramento ou alteração dos dados do usuário, o sistema deverá popular automaticamente os demais campos do endereço do usuário.
	RI3 - Nas telas de alteração, exclusão e recuperação de usuário, o sistema deverá exibir ícone informativo de acordo com a situação do usuário.

Quadro 2 - Ex. de Registro das Atividades- Identificação dos RIGUs

4.4 INSTANCIAÇÃO DO PROCESSO DE MEDIÇÃO

A instanciação do processo de medição deve ser realizada de forma independente das fases de mapeamento e avaliação. Para fins didáticos e para seguir a sequência das atividades descritas na Figura 7, foi optado por descrevê-la neste momento.

O processo de instanciação consiste na configuração dos parâmetros do método de medição para refletir da forma mais adequada o contexto da empresa onde o método será aplicado e do projeto a ser desenvolvido. De forma resumida, a instanciação do processo consiste em “calibrar” o método ao ambiente que será utilizado.

Assim, este processo de configuração do método deve ser feito considerando todo o processo de desenvolvimento da empresa, incluindo as ferramentas, padrões e frameworks utilizados pela empresa e no projeto.

Os parâmetros a serem configurados no método FUCS são:

- Tipos de caso de uso padrões e seus respectivos pesos (UC_{TP}–*Use Case Type Point*)
- Peso para as operações de sistema e regras de negócio (α)
- Peso para os requisitos de interface gráfica do usuário (β)

É importante destacar que para a utilização do método FUCS em processos licitatórios e de terceirização do desenvolvimento de software, é fundamental que os parâmetros definidos na instanciação do método estejam explicitamente definidos no edital de licitação ou no contrato de terceirização, ou seja, o processo de instanciação deve ser realizado antes de serem firmados os contratos.

Esta definição antecipada dos valores dos parâmetros do FUCS faz com que o processo de contagem seja realizado sob as mesmas condições e regras, minimizando assim as discrepâncias nas contagens, mesmo quando realizadas por pessoas diferentes.

4.4.1 Configurando os tipos de casos de uso e seus pesos (UC_{TP})

Este trabalho apresentou na seção “4.2.1 Tipos de Casos de Uso”, um conjunto de tipos de casos de uso que são comumente encontrados em sistemas de informação. No entanto, cada empresa pode definir seus próprios tipos em função de seus padrões de especificação e ferramentas de desenvolvimento. O importante é que os tipos estejam claramente definidos incluindo diretrizes e exemplos de quando utilizá-los.

Uma vez que os tipos de caso de uso estejam definidos, deve-se então definir o peso de cada tipo (UC_{TP} - *Use Case Type Point*) refletindo de forma mais adequada o esforço para desenvolvimento de cada tipo de acordo com os recursos e processos de desenvolvimento disponíveis, como por exemplo, padrões de especificação, ferramentas e frameworks.

O UC_{TP} deve receber um valor que pode variar de uma escala razão limitada de 1 a 5. Quanto maior o valor associado ao UC_{TP}, maior é a complexidade e o esforço necessário para desenvolvimento de um caso de uso do seu tipo.

Os estudos apresentados neste trabalho restringiram-se ao uso da escala limitada de 1 a 5 para permitir uma melhor comparação de complexidades entre os tipos de casos de uso. No entanto, não existem

impedimentos para que esta escala seja adaptada para outros limites, como por exemplo, de 1 a 10. O importante é que seja estabelecido um limite e que estas configurações estejam explicitamente descritas no edital de licitações e/ou no contrato de terceirização de serviços.

Para a configuração dos pesos de cada tipo de caso de uso (UC_{TP}), este trabalho apresenta algumas questões que podem ser utilizadas como um guia para a determinação destes pesos:

1. Existe um modelo de especificação disponível para o tipo de caso de uso?
2. Existe algum modelo de interface que pode ser utilizado para este tipo de caso de uso?
3. Existe algum framework que suporte a implementação deste tipo de caso de uso? Em média qual o percentual de redução de esforço obtido com a utilização deste framework?

Com base nas respostas às perguntas acima, um valor de UC_{TP} variando de 1 a 5 (ou outra escala definida pela empresa), deverá ser atribuído a cada tipo de caso de uso. Este valor deve ser diretamente proporcional ao esforço necessário para desenvolver um caso de uso desse tipo. A principal função da atribuição deste valor é definir um tamanho inicial para os casos de uso de cada tipo. Este tamanho inicial somado aos demais atributos do método FUCS irão compor o tamanho final do caso de uso.

Para atribuir um valor a UC_{TP}, é sugerido o seguinte algoritmo descrito na Figura 9.

$UC_{TP} = 5$
Se (resposta à pergunta 1 for SIM) Então
  $UC_{TP} = UC_{TP} - 1$
Se (resposta à pergunta 2 for SIM) Então
  $UC_{TP} = UC_{TP} - 1$
Se (resposta à pergunta 3 for SIM) Então
  Se (% de redução do esforço for de 20% a 40%) Então
    $UC_{TP} = UC_{TP} - 1$
  Se (% de redução do esforço for maior que 40%) Então
    $UC_{TP} = UC_{TP} - 2$

Figura 9 - Algoritmo para determinação do UC_{TP}

É importante destacar que, em relação à pergunta 3 (o uso de um framework e um percentual médio de redução do esforço previsto pela utilização desse framework), sugere-se a não redução do valor de UC_{TP}, para as taxas de redução do esforço para desenvolvimento inferiores a 20%.

A Tabela 22 mostra um exemplo de configuração dos tipos de casos de uso e seus respectivos pesos, baseado nos tipos definidos na seção "4.2.1 Tipos de Casos de Uso" e nos processos e padrões de desenvolvimento do TRT/SC.

Tabela 22 – Ex. de configuração dos tipos de casos de uso

Tipo de Caso de Uso	UC_{TP}
Operação Básica sobre uma Entidade	1
CRUD	2
CRUD Tabular	2
CRUD Mestre/Detalhe	4
CRUD Detalhes	3
Consulta Parametrizada	1
Relatório	1
Serviço	5
Biblioteca de Função ou Componente	5
Caso de uso "Não Padronizado"	5

Fonte: TRT/SC

Para tornar mais claro o entendimento da configuração destes pesos, a Tabela 23 mostra os detalhes sobre a configuração dos pesos de cada tipo de caso de uso de acordo com o contexto do TRT/SC. Este exemplo pode ser utilizado para auxiliar a configuração destes pesos em outros contextos de desenvolvimento.

A configuração descrita na Tabela 23 foi realizada de acordo com o guia sugerido no FUCS para a configuração dos tipos de caso de uso e seus respectivos pesos. Toda a configuração descrita foi realizada com base nas ferramentas, padrões e frameworks utilizados no TRT/SC.

Tabela 23–Configuração dos tipos de casos de uso e pesos no TRT/SC

Tipo de Caso de Uso	**Respostas às perguntas**			**Observações**	UC_{TP}
	1	**2**	**3**		
Operação Básica, Consulta Parametrizada e Relatórios	Sim	Sim	Sim	As operações básicas são geradas automaticamente pelo framework utilizado no TRT/SC e com auxílio de ferramentas gráficas. O percentual de redução de esforço pode chegar a 90%.	1
CRUD e CRUD Tabular	Sim	Sim	Sim	Estes tipos de caso de uso também podem ser gerados com auxílio de framework e ferramentas gráficas. No entanto, muitas vezes são necessárias algumas intervenções manuais. O percentual de redução do esforço fica entre 20% e 40%.	2
CRUD Detalhes	Sim	Sim	Não	O framework utilizado no TRT/SC disponibiliza um conjunto de ferramentas para desenvolvimento de forma gráfica destes tipos de casos de uso. No entanto, o padrão utilizado pelo TRT/SC requer uma série de customizações, que tornam o uso destas ferramentas improdutivo. Assim, são utilizadas apenas as classes base do framework que contam com pontos de extensão para implementação de comportamentos específicos.	3
CRUD Mestre/Detalhe	Sim	Sim	Não	As ferramentas disponíveis estão em um contexto parecido com o descrito para CRUD Detalhes. No entanto, a quantidade de pontos de extensão é mais reduzida, tornando-o mais complexo que o CRUD Detalhes.	4

Tipo de Caso de Uso	Respostas às perguntas			Observações	UC_{TP}
	1	2	3		
Serviço, Biblioteca de Função ou Componente e Caso de uso "Não Padronizado"	Não	Não	Não	Estes tipos de casos de uso são desenvolvidos manualmente sem auxílio do framework, portanto, não há qualquer recurso que aumenta a produtividade destes tipos de casos de uso.	5

É importante ressaltar que para a utilização do guia é necessário um conhecimento sobre o histórico de desenvolvimento da empresa, principalmente para a avaliação do percentual de redução de esforço proporcionado pelas ferramentas e frameworks utilizados pela empresa. Assim, o ideal é que a empresa possua informações históricas que permitam esta análise. No entanto, a ausência desta informação documentada não impede o uso do método, pois esta informação pode ser determinada por um analista experiente ou até mesmo em conjunto com toda a equipe da empresa.

4.4.2 Configurando o peso (α) para as Operações de Sistema e Regras de Negócio

O peso α corresponde ao grau de complexidade da implementação de operações de sistema e regras de negócio no contexto de aplicação do método. Este peso também deve ser ajustado em função dos padrões, frameworks e ferramentas utilizados no processo de desenvolvimento de cada empresa.

Para sua configuração, deve ser atribuído um valor que pode variar de uma escala razão limitada de 1 a 5. Quanto maior seu valor, maior é a influência das operações de sistema e regras de negócio no esforço total de desenvolvimento de um caso de uso. A configuração de α deve então ser feita refletindo-se sobre a seguinte questão:

- Em qual grau as ferramentas, padrões e frameworks facilitam a implementação de operações de sistema e regras de negócio?

Refletindo-se sobre a questão acima, pode-se então atribuir um valor de 1 a 5 à α. Em nível de sugestão são apresentadas as seguintes configurações:

- $\alpha = 5$: Quando não existem ferramentas, padrões e frameworks na empresa que facilitem a implementação de operações de sistema e regras de negócio;
- $\alpha = 3$: Quando a empresa conta com um conjunto de bibliotecas padrões de desenvolvimento e com pontos de extensões que facilitam a implementação destes tipos de requisitos;
- $\alpha = 1$: Quando, além de contar com um conjunto de bibliotecas padrões e pontos de extensões, a empresa possui ferramentas que possibilitam implementar estes tipos de requisitos de forma visual ou automatizada.

É importante destacar que valores intermediários também podem ser utilizados para representar situações que estejam entre uma situação e outra. O importante é que o valor atribuído à α seja relevante e correspondente aos processos, padrões e ferramentas utilizados na empresa.

Da mesma forma que a configuração dos tipos de casos de uso e seus respectivos pesos, a configuração de α deve constar no edital de licitação e/ou no contrato de terceirização, pois a definição deste parâmetro é fundamental para as empresas fornecedoras calcularem sua produtividade e consequentemente seus custos.

4.4.3 Configurando o peso (β) para os RIGUs

O peso β corresponde ao grau de complexidade da implementação de requisitos de interface gráfica do usuário (RIGU) no contexto de aplicação do método. Este peso também deve ser ajustado em função dos padrões, frameworks e ferramentas utilizados no processo de desenvolvimento de cada empresa.

Para sua configuração deve ser atribuído um valor que pode variar de uma escala razão limitada de 1 a 5. Quanto maior seu valor, maior é a influência dos requisitos de interface gráfica do usuário no esforço total de desenvolvimento de um caso de uso. A configuração de β deve então ser feita refletindo-se sobre a seguinte questão:

- Em qual grau as ferramentas, padrões e frameworks facilitam a implementação de requisitos de interface gráfica do usuário?

Refletindo-se sobre a questão acima, pode-se então atribuir um valor de 1 a 5 à β. Em nível de sugestão são apresentadas as seguintes configurações:

- $\beta = 5$: Quando não existem ferramentas, padrões e frameworks na empresa que facilitem a implementação de recursos da interface gráfica do usuário;
- $\beta = 3$: Quando a empresa conta com um conjunto de bibliotecas padrões de desenvolvimento e componentes de interface gráfica que facilitam a implementação destes tipos de requisitos;
- $\beta = 1$ – Quando, além de contar com um conjunto de bibliotecas padrões de desenvolvimento e componentes de interface gráfica, a empresa possui ferramentas que possibilitam implementar estes tipos de requisitos de forma visual ou automatizada.

É importante destacar que valores intermediários também podem ser utilizados para representar situações que estejam entre uma situação e outra. O importante é que o valor atribuído à β seja relevante e correspondente aos processos, padrões e ferramentas utilizados na empresa.

Da mesma forma que os outros parâmetros do método, a configuração de β também deve constar no edital de licitação e/ou no contrato de terceirização, pois consiste em um valor importante para as empresas fornecedoras calcularem sua produtividade e, consequentemente, seus custos.

4.4.4 Considerações gerais sobre a instanciação do processo

Conforme descrito ao longo desta seção, a instanciação do processo de medição deve ser realizada de forma independente das fases de mapeamento e avaliação, pois consiste na configuração do método ao contexto onde será aplicado o processo de medição.

Assim, para o uso do FUCS em processos licitatórios e contratos de terceirização de serviços de desenvolvimento de software, é fundamental que todas as configurações do método estejam estabelecidas no edital de licitação e/ou contrato de terceirização. Isso fará com que o processo de medição e contagem seja realizado sob as mesmas condições (parâmetros e valores), minimizando assim avaliações discrepantes quando realizadas por pessoas diferentes.

De forma resumida, para a utilização do FUCS em processos licitatórios e de terceirização, o edital e/ou contrato de terceirização deve explicitar as seguintes informações:

- Definição dos tipos de casos de uso utilizados na empresa contratante e seus respectivos pesos;
- Definição dos valores dos pesos α e β que serão utilizados no processo de medição.

4.5 AVALIAÇÃO (MEDIÇÃO)

A fase de avaliação consiste na contagem dos atributos extraídos na fase de mapeamento de acordo com as regras e procedimentos definidos neste método. A fase de avaliação pode ser realizada por etapas, iniciando com a classificação dos casos de uso de acordo com seus tipos, a fatoração de operações de sistema, regras de negócio e requisitos de interface em casos de uso do tipo “Biblioteca de Função ou Componente”, a contagem das operações de sistema, regras de negócio e requisitos de interface e por fim, o cálculo do tamanho de cada caso de uso e do projeto total. As medidas obtidas no método FUCS são expressas sob a métrica UCS (*Use Case Size*).

As seções a seguir descrevem cada etapa da fase de avaliação juntamente com as regras e procedimentos de contagem estabelecidos neste método. Para exemplificar as etapas será utilizado o exemplo descrito nos Quadros 1 e 2.

4.5.1 Classificação dos Casos de Uso: Determinando UC_{TP}

Uma vez que o processo está instanciado para o contexto de medição (seção 4.4), a fase de avaliação pode ser iniciada com a classificação de cada caso de uso do sistema de acordo com seu tipo. A Tabela 24 resume todos os tipos de casos de uso sugeridos neste trabalho juntamente com as diretrizes e orientações de como utilizá-los no processo de classificação dos casos de uso.

Tabela 24 - Diretrizes para classificação dos tipos dos casos de uso

Tipo do Caso de Uso	**Quando classificar**
Operação Básica sobre uma Entidade	Este tipo de caso de uso deve ser utilizado quando um determinado ator operar especificamente sobre uma Operação Básica sobre uma Entidade, ou seja, quando este ator não puder operar sobre o conjunto completo de operações sobre um objeto (CRUD). Exemplos: - Incluir Cliente - Excluir Ciente - Alterar dados do Cliente - Recuperar cliente (por identificador)
Consulta Parametrizada	Este tipo de caso de uso deve ser utilizado para as situações corriqueiras de pesquisa, onde os usuários informam um conjunto de parâmetros e o sistema retorna uma lista de ocorrências que atendem aos critérios especificados. Exemplo: - Pesquisa Clientes (por nome, por cidade, por estado)
Relatório	Este caso de uso deve ser utilizado nas situações onde o resultado de uma consulta parametrizada deva ser renderizada em formato de relatório, permitindo salvar o relatório no computador ou imprimi-lo. Os relatórios podem conter seções e totalizadores. Exemplos: - Relatório de Clientes da Região Sul; - Relatório de Vendas no mês.

Tipo do Caso de Uso	**Quando classificar**
CRUD	Este tipo de caso de uso deve ser utilizado nas situações onde os atores podem operar sobre o conjunto completo de operações sobre um objeto (CRUD). Pode ser utilizado também, desde que em conjunto com regras de negócio, para as situações onde algum ator não tenha permissão sobre alguma operação. Nestes casos, as restrições de acesso deverão ser classificadas como regras de negócio do sistema. Exemplos: - Gerenciar Usuários do Sistema - Gerenciar Clientes - Gerenciar Produtos
CRUD Tabular	Este tipo de caso de uso deve ser utilizado nas situações onde o gerenciamento dos objetos é feito de forma muito similar a uma tabela de dados. Neste tipo de caso de uso, todos os objetos são gerenciados em uma única transação. Exemplos: - Manter tabela de alíquotas - Manter Unidades da Federação - Manter tabela de Modelos de Veículos
CRUD Mestre/Detalhe	Este caso de uso deve ser utilizado quando há a necessidade de manutenção de um objeto principal (mestre) juntamente com um conjunto de objetos associados (detalhes). Exemplo: - Manter Dados dos Clientes (mestre), seus endereços e contatos (detalhes)
CRUD Detalhe	Este caso de uso deve ser utilizado quando há a necessidade de manter os registros de detalhes de um determinado objeto principal. Exemplo: - Em um sistema de recursos humanos, o Gerenciamento do Histórico Profissional de um Colaborador, onde o objeto mestre (Colaborador) é somente-leitura, e é permitido somente a manutenção do histórico de cargos e salários do colaborador.
Serviço	Este tipo de caso de uso deve ser utilizado quando o ator principal é outro sistema. Neste tipo não há necessidade de interfaces gráficas. Exemplos: - *Webservice* para consulta de clientes; - Geração de arquivo para download de catálogo de produtos.

Tipo do Caso de Uso	Quando classificar
Biblioteca de Função ou Componente	Este tipo de caso de uso deve ser usado quando o analista perceber que um determinado conjunto de funcionalidades pode ser reutilizado ao longo de todo o sistema. Exemplos: - Biblioteca para cálculo das taxas de juros sobre débitos dos clientes; - Componentes de conversão de moedas.
Não Padronizado	Os casos de uso que não possam ser classificados em nenhuma das categorias anteriores devem ser classificados como “Não Padronizado”.

É importante destacar que, caso o processo de instanciação resulte em uma tabela de tipos de casos de uso diferente da sugerida neste trabalho (Tabela 22), é de fundamental importância que sejam descritas as diretrizes para classificação dos casos de uso em função dos novos tipos, minimizando assim as divergências nas classificações dos casos de uso quando realizados por pessoas diferentes.

Considerando o exemplo do caso de uso descrito nos Quadros 1 e 2 e as orientações e diretrizes de classificação descritas na Tabela 24, o caso de uso “Manter cadastro de usuários do sistema” deve ser classificado como do tipo “CRUD”. Desta forma, considerando que a instanciação do processo resultou na configuração de tipos de caso de uso e seus respectivos pesos, descritos na Tabela 22, o peso deste caso de uso, em função de seu tipo é 2 ($UC_{TP} = 2$).

4.5.2 Contagem das Operações de Sistema e Regras de Negócio: Determinando N_{SOBR}

Antes de realizar a contagem das operações básicas e regras de negócio, é necessário garantir a unicidade delas fazendo uma análise em relação a todos os casos de uso que compõem o escopo de contagem.

Para determinar se uma operação de sistema ou uma regra de negócio são únicas no escopo de contagem, pode-se utilizar as seguintes diretrizes ao comparar um caso de uso a outro já identificado:

(a) Contam-se duas operações de sistema similares como uma só se elas executam a mesma função sobre o mesmo conceito (objeto);

(b) Contam-se duas regras de negócio similares como uma só se elas operam sobre os mesmos conceitos (objetos) e seus objetivos forem comuns entre si, independentemente das operações de sistema que estejam associadas.

Uma vez identificado que as operações de sistema e as regras de negócio associadas ao caso de uso são únicas no escopo de medição, a contagem deve ser realizada com base nas seguintes diretrizes:

(a) Para cada operação de sistema, verificar se ela não faz parte das operações pré-definidas no tipo em que está classificado o caso de uso. Caso não faça parte das operações pré-definidas, esta operação deve ser contada, caso contrário, ela não terá influência na contagem, pois já está sendo avaliada quando da classificação do caso de uso;

(b) Cada regra de negócio única no sistema deve ser contada.

Aplicando estas diretrizes de contagem podemos obter a variável N_{SOBR} (Nº de Operações de Sistema e Regras de Negócio) através da soma da quantidade de operações de sistema não definidas no tipo de caso de uso com a quantidade de regras de negócio associadas ao caso de uso. Considerando o exemplo descrito no Quadro 1, temos:

Operações de Sistema:

- Gerar nova senha para o Usuário **deve ser contada;**
- Incluir Usuário, Alterar dados do Usuário, Excluir Usuário e Recuperar Usuário **não devem ser contadas**, pois já estão pré-definidas no tipo do caso de uso – CRUD;
- Total de Operações do Sistema: 1.

Regras de Negócio:

- RN1 - Ao incluir um usuário o sistema deve gerar uma senha automaticamente e enviá-la para o email do usuário (**deve ser contada**);
- RN2 - Um usuário só pode ser excluído pelo administrador do sistema (**deve ser contada**);
- RN3 - Ao gerar uma nova senha, o sistema deverá enviá-la para o email do usuário (**não deve ser contada** pois possui o mesmo objetivo e opera sobre o mesmo objeto que a RN1**);**
- Total de Regras de Negócio: 2.

Logo, $N_{SOBR} = 1\ (OS) + \ 2\ (RN) = 3$.

4.5.3 Contagem dos Requisitos de Interface Gráfica do Usuário: Determinando N_{GUIR}

De forma similar a contagem de operações de sistema e das regras de negócio, a contagem dos requisitos de interface gráfica do usuário (RIGU) também requer que a unicidade dos requisitos esteja garantida fazendo uma análise em relação a todos os casos de uso que compõem o escopo de contagem.

Além disso, um RIGU, quando associado a mais de um caso de uso, pode transformar-se em um caso de uso do tipo "Biblioteca de Função ou Componente", permitindo assim sua reutilização ao longo do projeto.

Para determinar se um RIGU é único no escopo de contagem ou se ele pode ser transformado em um caso de uso, pode-se utilizar as seguintes diretrizes:

(a) Contam-se RIGUs similares como um só se eles oferecem a mesma funcionalidade para um mesmo conceito (objeto), mesmo que em casos de uso diferentes.

(b) Transforma-se um RIGU em um caso de uso do tipo "Biblioteca de Função ou Componente" quando sua funcionalidade puder ser parametrizável e aplicável a vários conceitos (objetos).

Ao transformar um RIGU em um caso de uso do tipo “Biblioteca de Função ou Componente” deve-se proceder toda a análise necessária para identificar as operações de sistema e as regras de negócio, bem como os requisitos de interface gráfica associados a ele. Este caso de uso deve então ser adicionado à listagem de casos de uso que compõem o escopo de contagem do sistema.

Um exemplo desta situação pode ser visto em um sistema de processos que possui caso de uso para “cadastramento de protocolo” e outro para “juntada de documentos em um processo”. Nos dois casos os usuários solicitaram a possibilidade de anexar arquivos usando o recurso de *drag and drop*. Porém, apesar da função ser a mesma, os conceitos (objetos) são diferentes: em um caso de uso o conceito é “processo” e em outro é “protocolo”.

Neste caso, poder-se-ia transformar o RIGU em um caso de uso genérico para permitir anexar arquivos com o recurso de *drag and drop*. Este caso de uso deveria ser então classificado como “Biblioteca de Função ou Componente”, sendo que tem associado a ele, pelo menos 1 requisito de interface gráfica – o recurso *drag and drop*.

Aplicando estas diretrizes de contagem podemos obter a variável N_{GUIR} (Nº de Requisitos de Interface Gráfica do Usuário) que recebe o valor obtido na contagem dos RIGUs. Considerando o exemplo descrito no Quadro 2, temos:

Requisitos de Interface Gráfica do Usuário:

- RI1 - No cadastramento de usuário, ao selecionar o estado do endereço, o sistema deve popular automaticamente a lista de cidades com as cidades do estado escolhido pelo usuário;
- RI2 - Ao digitar o número do CEP no cadastramento ou alteração dos dados do usuário, o sistema deverá popular automaticamente os demais campos do endereço do usuário;
- RI3 - Nas telas de alteração, exclusão e recuperação de usuário, o sistema deverá exibir ícone informativo de acordo com a situação do usuário;
- Total de RIGUs: 3

Logo, $N_{GUIR} = 3$.

4.5.4 Cálculo do tamanho de cada caso de uso (UC_S)

O tamanho de cada caso de uso ($UC_S - Use\ Case\ Size$) é obtido em função das variáveis UC_{TP}, N_{SOBR} e N_{GUIR}. A Equação 38 define como o tamanho de cada caso de uso pode ser obtido.

$$UC_S = (UC_{TP} + \alpha * N_{SOBR} + \beta * N_{GUIR}) \qquad (38)$$

Onde α e β são os pesos que representam o grau de complexidade associado à implementação de operações de sistema e regras de negócio, e requisitos de usuário gráfica do usuário, respectivamente. Estes pesos devem ser ajustados para cada empresa no processo de instanciação do método de medição (seção 4. 4).

Considerando, por exemplo, que na instanciação do processo foram definidos pesos para os tipos de caso de uso conforme a Tabela 22, e que os valores de α e β foram definidos como 2 e 3, respectivamente, o tamanho do caso de uso dos exemplos dos Quadros 1 e 2 seria calculado da seguinte forma:

Parâmetros:

$\mathrm{UC_{TP}} = 2$ *(Tabela 22 – Caso de Uso Tipo "CRUD")*
$\alpha = 2$
$\beta = 3$
$N_{OSNR} = 3$
$N_{GUIR} = 3$

Cálculo:

$\mathrm{UC'_S} = \mathrm{UC_{TP}} + \alpha * N_{OSNR} + \beta * N_{GUIR}$

$\mathrm{UC'_S} = 2 + 2 * 3 + 3 * 3$

$\mathrm{UC'_S} = 17$ UCS $(use\ case\ size)$

Ou seja, o caso de uso "Manter cadastro de usuários do sistema" possui um tamanho de 17 UCS (use case size).

É importante destacar que esta medição foi realizada com base na classificação do caso de uso e na contagem das operações de sistema, regras de negócio e requisitos de interface gráfica do usuário que estavam documentados nos exemplos deste trabalho. Assim, se o

processo de medição e estimativa do FUCS for aplicado por pessoas diferentes, possivelmente os resultados serão os mesmos.

Em virtude de α e β estarem definidos previamente a fase de medição, só há a possibilidade de ocorrer alguma diferença na medida se o caso de uso for classificado de forma diferente ou se ocorrer algum equívoco na contagem dos requisitos (OSs, RNs e RIGUs). No entanto, a classificação dos casos de uso é orientada por um conjunto de diretrizes (Tabela 24) e o processo de contagem dos requisitos é simples e objetivo. Isso faz com que o processo de medição não corra riscos de avaliações muito discrepantes e resultados muito variados, conferindo-lhe assim maior precisão nas medidas e estimativas do software.

O processo de medição deve então seguir com a determinação do tamanho de todos os casos de uso que compõem o escopo de contagem.

4.5.5 Cálculo do tamanho do sistema (S_{size})

Depois de calcular os tamanhos de todos os casos de uso do sistema, o tamanho do sistema ($S_{size} - System\,Size$) pode ser calculado pela soma dos casos tamanhos dos casos de uso (UC_S) tal como definido na Equação 39. A medida resultante é dada em UCS (*use case size*).

$$S_{size} = \left(\sum_{n=1}^{u} \mathrm{UC_S} \right) \qquad (39)$$

onde u é o número de casos de uso do sistema

O Anexo B, apresenta um exemplo didático da medição de um projeto, mostrando como todos os passos podem ser aplicados na medição e estimativa de um projeto real.

4.5.6 Estimando o esforço total e os custos do projeto

É importante destacar que o tamanho do sistema (S_{size}) calculado pelo FUCS é dado em função do tamanho de seus casos de uso, ou seja, esta medida só leva em consideração atividades relacionadas ao levantamento de requisitos, modelagem de negócio, análise e projeto do sistema, implementação e testes. As demais atividades de projeto, como gerenciamento, treinamento, implantação, transição, reuniões, viagens,

dentre outras, não são levadas em consideração neste método, pois dependem muito das características de cada projeto e dos usuários finais do sistema que está sendo desenvolvido. No entanto, para estimativas de esforço total e custos de um projeto, há de se ponderar estas outras atividades.

Apesar de não ser objetivo deste trabalho detalhar sobre estimativas gerais de esforço e custos de projetos, é sugerido uma forma simples de se obter estas estimativas.

O uso de uma tabela de distribuição do esforço por cada fase de projetos (que é variável de acordo com o contexto de cada empresa) é fundamental para derivar estimativas de esforço. A Tabela 25, mostra um exemplo da distribuição de esforço em um projeto de desenvolvimento de sistema no contexto do TRT/SC. O processo de desenvolvimento correspondente a esta tabela está descrito no estudo de caso 1 do capítulo 5.

Tabela 25 - Tabela de distribuição do esforço por fase

Fase/Atividade	% do esforço
Requisitos	8
Análise de Sistema	8
Projeto de Sistema	10
Implementação	30
Testes	10
Homologação	9
Treinamento	5
Implantação	5
Pós-implantação	5
Planejamento e Gestão	10

Fonte: TRT/SC

Utilizando a Tabela 25 como referência, podemos observar que as fases e atividades relacionadas ao levantamento de requisitos, análise e projeto do sistema, implementação e testes correspondem a 66% das atividades de todo o projeto. Estas são as atividades que correspondem ao desenvolvimento dos casos de uso de um sistema, e portanto, são consideradas no método proposto neste trabalho, ou seja, a estimativa de esforço para estas atividades pode ser obtida em função do tamanho do sistema (S_{size}). Considerando que S_{size} corresponde a 66% do esforço do projeto (para uma distribuição de esforço como a apresentada na Tabela 25), o esforço total (S_{effort} - *System Effort*) poderia então ser calculado pela equação 40.

$$S_{effort} = \frac{S_{size}}{0,66} * Produtividade \quad (40)$$

Os resultados descritos no capítulo 7, mostram que 1 UCS leva em média 7 horas para seu desenvolvimento. Assim, a variável $Produtividade$ poderia assumir o valor de 7 horas/UCS.

A formação do custo do projeto deve levar em consideração uma série de outros fatores, como custo por hora dos profissionais envolvidos em cada fase do projeto, plano de viagens com custos de deslocamentos e diárias, plano de cursos para a equipe do projeto, dentre outros fatores.

4.6 CONSIDERAÇÕES FINAIS SOBRE O FUCS

Conforme foi descrito ao longo deste capítulo, o processo de medição do FUCS segue o modelo genérico de medição funcional de software, apresentando as fases de mapeamento, instanciação e medição.

A fase de instanciação consiste em configurar o FUCS para o contexto de cada empresa. Isso é feito através da configuração dos tipos de casos de uso e seus respectivos pesos e dos parâmetros α e β, cujos valores devem ser definidos em escala razão refletindo da forma mais adequada aos padrões, ferramentas e frameworks utilizados em cada empresa.

Apesar de o FUCS apresentar algumas diretrizes e sugestões para a configuração de seus parâmetros, a efetiva configuração quando realizada por pessoas diferentes pode apresentar algumas diferenças nos seus valores. Por este motivo, para a utilização do FUCS em licitações e terceirização do desenvolvimento de software, é de fundamental importância que todas suas configurações estejam explicitamente descritas no edital de licitação e/ou contrato de terceirização. Esta medida deve ser tomada para garantir que os parâmetros do método estejam previamente definidos, evitando assim que pessoas diferentes utilizem valores diferentes nas suas medições e garantindo que todas as empresas tenham os mesmos valores de referência para formação dos seus custos.

As fases de mapeamento e medição do FUCS consistem na extração dos atributos a serem contados e na efetiva contagem deles, as quais devem ser feitas de acordo com as regras estabelecidas no método. O fato de o FUCS realizar seu processo de medição baseado na

contagem dos requisitos (regras de negócio, operações de sistema e requisitos de interface gráfica do usuário) e na classificação dos casos de uso de acordo com seu tipo, confere-lhe uma maior objetividade no processo de medição e evita avaliações discrepantes quando realizadas por pessoas diferentes.

Desta forma, com a definição prévia dos parâmetros de configuração do FUCS nos editais de licitações e nos contratos de terceirização juntamente com a objetividade do processo de contagem e medição do sistema, o FUCS apresenta uma forma mais objetiva de estimativas de software.

5 ESTUDO DE CASO

Este capítulo apresenta três estudos de casos onde o método de medição e estimativa proposto neste trabalho foi aplicado em projetos reais com o intuito de verificar a relação entre o esforço de desenvolvimento de cada caso de uso com seu tamanho obtido através do método, bem como o esforço total do projeto com o tamanho do projeto.

Os três estudos de casos foram realizados em empresas diferentes, sendo um projeto no TRT/SC, um no TST (Tribunal Superior do Trabalho) e um em uma empresa privada (TNC). Todos os projetos avaliados classificam-se na categoria dos sistemas de informações sendo que as informações são armazenadas em bancos de dados relacionais.

Para tornar a leitura mais dinâmica, não serão descritos todos os casos de uso dos sistemas dos estudos de caso, mas apenas um exemplo de cada estudo de caso para demonstrar como foram medidos com o método FUCS.

5.1 ESTUDO DE CASO 1

Este estudo de caso foi realizado em um projeto de um Sistema de Processos Administrativos do TRT/SC (PROAD-TRT/SC). Este projeto consistiu no desenvolvimento de um sistema web para gerenciamento eletrônico de todos os documentos e processos administrativos do TRT/SC. Seus módulos principais são: Módulo de Cadastros, Módulo de Pesquisas e Módulo de Gestão. Este estudo de caso foi desenvolvido utilizando um processo de software baseado no Processo Unificado (JACOBSON, BOOCH e RUMBAUGH, 1999). O processo de desenvolvimento de software adotado no TRT/SC é dividido nas seguintes fases: iniciação, concepção, elaboração, construção, testes, aceitação, implantação e estabilização (pós-implantação).

A fase de iniciação é caracterizada pela coleta de expectativas da administração do TRT/SC em relação aos custos e prazos. Além da iniciação, todas as outras fases são realizadas em um processo iterativo e incremental, onde a cada iteração é desenvolvido um conjunto de casos de uso.

Durante a fase de concepção, os membros da equipe de desenvolvimento participam de sessões de modelagem (AMBLER, 2004) juntamente com os usuários para realizarem o levantamento dos requisitos. Como apoio ao processo de levantamento de requisitos, o TRT/SC utiliza um método baseado no método descrito por Larman (LARMAN, 2004), o qual está descrito com mais detalhes no Anexo A. Ao final das sessões, os seguintes artefatos são produzidos:

- Diagrama de casos de uso ou uma lista de casos de uso
- Descrição geral dos casos de uso (opcional)
- Lista dos requisitos de interface de usuário e regras de negócio para cada caso de uso

Uma das principais diferenças deste processo de desenvolvimento com outros processos convencionais é a utilização de protótipos de interface de usuário substituindo uma descrição detalhada dos casos de uso.

A fase de elaboração produz os artefatos técnicos necessários para a fase de construção, tais como diagramas de classe, descrições de casos de uso e diagramas de sequência. A fase de construção é estritamente técnica, e produz o código fonte que implementa as funcionalidades necessárias. A fase de construção é seguida das fases de testes, aceitação, implantação e estabilização, respectivamente. Mudanças de requisitos podem surgir em qualquer fase do processo de desenvolvimento. Sempre que essas mudanças ocorrem, as estimativas são recalculadas, a fim de avaliar como essas mudanças impactam nos custos e prazos do projeto. No processo descrito acima, as fases de elaboração e construção foram terceirizadas, e o contrato de terceirização foi estabelecido através de um processo licitatório que especificava este método como forma de medição dos produtos entregues.

O processo de desenvolvimento do TRT/SC conta com o uso do framework JCompany[14], versão corporativa do framework *open-source* Jaguar[15]. Este framework apresenta soluções prontas para a maioria dos tipos de casos de uso, onde os comportamentos padrões dos casos de uso podem ser gerados através de ferramentas gráficas e com o auxílio de *wizards*. Neste framework, os comportamentos específicos de cada caso de uso (regras de negócio e requisitos de interface gráfica do usuário)

[14]http://www.powerlogic.com.br/powerlogic/ecp/comunidade.do?app=portal&pg=540&tax=0
[15]http://www.softwarepublico.gov.br/ver-comunidade?community_id=25913900

são implementados através de pontos de extensão como, por exemplo, métodos do tipo *antesDeIncluir*, *depoisDeIncluir*, *antesDeAlterar*, *depoisDeAlterar*, dentre outros.

O TRT/SC conta ainda com um conjunto de bibliotecas que implementam uma série de melhorias sobre o framework e agregam alguns componentes específicos do seu negócio. Além disso, o TRT/SC conta com um modelo de projeto que permite a geração inicial do sistema já configurado de acordo com os padrões de interface gráfica corporativos e com os métodos de autenticação e autorização padronizados.

Em relação à aplicação do método no estudo de caso, na fase de instanciação do processo de medição, foram definidos os valores das variáveis α e β de acordo com os padrões, ferramentas e frameworks que compõem o processo de desenvolvimento do TRT/SC. Dentro do contexto do TRT/SC e considerando seu processo de desenvolvimento, essas variáveis foram definidas para o valor 2, pois o processo de desenvolvimento conta com frameworks que possibilitam uma maior produtividade na implementação dos casos de uso, e oferecem pontos de extensão para inclusão de comportamentos específicos nos casos de uso padrões. A Tabela 22 mostra os tipos de casos de uso e seus respectivos pesos associados (UC_{TP}) utilizado no processo de medição deste estudo de caso.

As Figuras 10, 11 e 12 descrevem exemplos dos artefatos da concepção inicial do sistema produzidos no TRT dos quais os atributos necessários para a medição são extraídos e contados de acordo com os procedimentos e regras do FUCS. Estes artefatos são os mesmos enviados à fábrica de software para execução dos serviços de elaboração, construção e testes do sistema.

O Quadro 3, ilustra o resultado do mapeamento e da avaliação (medição) das informações dos artefatos de concepção do caso de uso para os conceitos do FUCS. De acordo com as informações mapeadas e descritas no Quadro 3, o tamanho do caso de uso “Petição Complementar” foi medido utilizando a equação 38, conforme descrito na última linha do Quadro 3.

Caso de Uso: Petição Complementar
Nível de Detalhamento: Baixo
Atores: Usuário comum
Objetivo do caso de uso: Permitir ao usuário entrar com Petição Complementar em processos que já estão em andamento.
Descrição: O usuário informa o número e ano do processo em que deseja entrar com a petição complementar. Informa uma justificativa para ingresso da petição complementar e anexa os arquivos que deseja juntar no processo.
Requisitos do caso de uso:

R1. Controle dos campos da tela: O sistema deverá manter habilitado somente o campo número do processo, até que este seja validado (R3).

R2. Formato e máscara do número do processo: O número e ano do processo devem ser informados no mesmo campo com a máscara NNNNN/AAAA, onde NNNNN representa o número do processo com 5 dígitos e AAAA representa o ano do processo com 4 dígitos. A máscara deve ser automática pelo sistema.

R3. Carregar dados do processo: Ao informar o número do processo o sistema deve buscar as informações do processo e exibi-las para o usuário, liberando demais campos da tela. Se o processo não existir ou estiver na situação JUNTADO, o sistema deverá exibir mensagem informativa ao usuário, não liberando acesso aos demais campos da tela.

R4. Deve-se permitir anexar um ou mais arquivos no pedido complementar, sendo que pelo menos 1 é obrigatório.

R5. Ao concluir o pedido complementar, o sistema deve gerar um arquivo com o conteúdo descrito na justificativa e anexá-lo automaticamente. O pedido e seus arquivos devem ficar na situação PENDENTE DE RECEBIMENTO, e o processo que recebeu o pedido complementar deve ficar na situação PEDIDO COMPLEMENTAR PENDENTE.

Figura 10 – Estudo de Caso 1: Descrição de um caso de uso

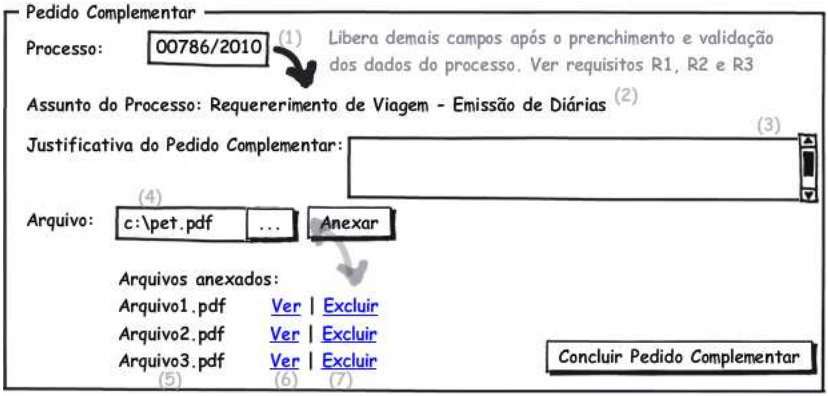

Figura 11 – Estudo de Caso 1: Protótipo de Tela do Caso de Uso

Figura 12 – Estudo de Caso 1: Classes de Domínio do Caso de Uso

	Caso de uso: Petição Complementar			
Mapeamento	Tipo de Caso de Uso:	CRUD Mestre/Detalhe	UC_{TP}:	4
	Justificativa da utilização do tipo:	Neste caso de uso **um mesmo ator** poderá incluir informações de um registro mestre (Petição) juntamente com vários detalhes (anexos).		
	Operações de Sistema e Regras de negócio:	R5	N_{OSNR}:	1
	Requisitos e Interface Gráfica do Usuário	R1, R2, R3 e R4	N_{GUIR}:	4
Avaliação	Tamanho do caso de uso (UC_S)	$UC_S = (UC_{TP} + \alpha * N_{OSNR} + \beta * N_{GUIR})$ $UC_S = (4 + 2 * 1 + 2 * 4) = 14\ UCS$		

Quadro 3 - Estudo de Caso 1: Mapeamento e Medição do Caso de Uso

O ANEXO C apresenta a relação de todos os casos de uso do sistema PROAD (representado na tabela como PROJ. E) juntamente com seus atributos e o seu tamanho, obtido através da aplicação do método FUCS.

Ao final da medição de todos os casos de uso, foi aplicada a equação 39 para obtenção do tamanho do sistema (S_{size}), totalizando 775 UCS. Os resultados descritos na seção 5.4, demonstram que em média, o esforço para desenvolvimento de 1 UCS pode ser estimado como sendo 7 horas. Assim, o esforço para desenvolvimento do PROAD poderia ser estimado em 5425 horas. No entanto, o esforço real de desenvolvimento foi de 4755 horas, dando uma margem de erro de 14,1%.

5.2 ESTUDO DE CASO 2

O segundo estudo de caso foi realizado em um projeto do TST (Tribunal Superior do Trabalho). Neste projeto, um sistema de publicação e disponibilização de diários oficiais eletrônicos (DEJT) foi totalmente reescrito, de uma plataforma ORACLE para uma plataforma JAVA/WEB. Neste estudo, nenhum processo formal e estabelecido foi utilizado. Os casos de uso, as regras de negócio e os requisitos de interface foram levantados com base no funcionamento do sistema existente (antigo). Na medida em que as informações dos casos de uso eram levantadas, seu desenvolvimento era iniciado. Neste processo também não foi produzido nenhum documento de especificação. O resultado final do projeto foi somente o software reescrito e funcionando. Neste estudo de caso, o desenvolvimento também foi terceirizado. E os procedimentos de medição foram estabelecidos com base no método FUCS.

Para o desenvolvimento deste sistema também foi utilizado o framework JCompany. No entanto, diferentemente do TRT/SC, o TST não conta com um conjunto de bibliotecas e todas as customizações do framework aos padrões da empresa. Desta forma, o desenvolvimento contou apenas com as funcionalidades padrões do framework.

Assim, em relação à aplicação do método neste estudo de caso, na fase de instanciação do processo de medição, foram definidos os valores das variáveis α e β de acordo com os padrões, ferramentas e frameworks utilizados no projeto. Apesar das tecnologias utilizadas neste projeto serem similares às utilizadas no estudo de caso 1, as variáveis α e β foram definidas para os valores 3 e 4, respectivamente, em virtude de que no contexto deste projeto, a empresa não contava com um mesmo[16] conjunto de bibliotecas, ferramentas e frameworks que aumentam a produtividade do desenvolvimento. Em relação à configuração dos tipos de casos de uso, foi optado por utilizar os mesmos tipos e pesos definidos na Tabela 22.

As Figuras 13 e 14 descrevem como exemplo o caso de uso “Manter Usuário” dos quais os atributos necessários para a medição foram extraídos e contados de acordo com os procedimentos e regras do FUCS.

[16] O estudo de caso 1 foi realizado no TRT/SC que conta com uma camada de bibliotecas de funções e um conjunto de ferramentas que dão maior produtividade ao desenvolvimento dos sistemas e na customização de regras de negócio e requisitos de interface, pois nestas bibliotecas já existem uma série de funcionalidades e componentes prontos.

Figura 13 - Estudo de Caso 2: Tela do Caso de Uso

Caso de Uso: Cadastro de Usuário
Requisitos do caso de uso:
R1. Não deve existir CPF e emails duplicados
R2. Enviar Email para usuário com link de ativação (com código gerado para o usuário)
R3. Ao escolher um perfil, o sistema deverá exibir somente as unidades publicadoras permitidas para aquele perfil
R4. Ao digitar o CPF, o sistema deverá recuperar automaticamente os dados do usuário com o CPF informado.

Figura 14 - Estudo de Caso 2: Requisitos do Caso de Uso

O Quadro 4 ilustra o resultado do mapeamento e da avaliação (medição) das informações dos artefatos de concepção do caso de uso para os conceitos do FUCS. De acordo com as informações mapeadas e descritas no Quadro 4, o tamanho do caso de uso "Manter Usuário" foi medido utilizando a equação 38, conforme descrito na última linha do Quadro 4.

	Caso de uso: Manter Usuário			
Mapeamento	Tipo de Caso de Uso:	CRUD Mestre/Detalhe	UC_{TP}:	4
	Justificativa da utilização do tipo:	Neste caso de uso **um mesmo ator** poderá incluir, alterar, excluir e recuperar informações de um registro mestre (Usuário) juntamente com vários detalhes (Unidades Publicadoras).		
	Operações de Sistema e Regras de negócio:	R1 e R2	N_{OSNR}:	2
	Requisitos e Interface Gráfica do Usuário	R3 e R4	N_{GUIR}:	2
Avaliação	Tamanho do caso de uso (UC_S)	$UC_S = (UC_{TP} + \alpha * N_{OSNR} + \beta * N_{GUIR})$ $UC_S = (4 + 3 * 2 + 4 * 2) = 18\ UCS$		

Quadro 4 - Estudo de Caso 2: Mapeamento e Medição do Caso de Uso

O ANEXO C apresenta a relação de todos os casos de uso do sistema DEJT (PROJ. H) juntamente com seus atributos e o seu tamanho, obtido através da aplicação do método FUCS.

Ao final da medição de todos os casos de uso, foi aplicada a equação 39 para obtenção do tamanho do sistema (S_{size}) totalizando 238 UCS. Usando uma produtividade de 7 Horas por UCS (conforme resultados descritos na seção 5.4 deste capítulo), o esforço de desenvolvimento do PROAD poderia ser estimado em 1666 horas. No entanto, o esforço real de desenvolvimento foi de 1470 horas, dando uma margem de erro de 13,3 %.

5.3 ESTUDO DE CASO 3

O terceiro estudo de caso foi realizado em um projeto de uma empresa privada, a TNC Informática. Neste projeto foi desenvolvido um sistema para avaliação de projetos financiados e em andamento para uma empresa estatal[17]. O sistema consiste em um módulo de administração, um módulo de avaliação de projetos e um módulo de acesso público. Este projeto foi totalmente desenvolvido em plataforma WEB usando tecnologias Microsoft.NET. Neste estudo, o desenvolvimento não foi terceirizado e a aplicação do método foi realizada depois do projeto ter iniciado seu desenvolvimento. O objetivo da aplicação do método neste projeto foi verificar se a estimativa obtida com sua aplicação aproximava-se do esforço real de desenvolvimento do projeto, e se a margem de erro ficava dentro do esperado.

Em relação ao seu processo de desenvolvimento, a TNC Informática não segue nenhum processo formal de desenvolvimento e não produz muita documentação de seus sistemas. O processo de desenvolvimento da TNC inicia com o desenho do diagrama de classes do projeto na ferramenta Enterprise Architect (EA)[18]. Uma vez que as classes estão desenhadas, um gerador de código é executado, o qual gera todos os casos de uso de operações básicas sobre cada classe do domínio, ou seja, as operações de inclusão, exclusão, alteração e consulta. O código fonte gerado oferece pontos de extensão onde os comportamentos padrões podem ser customizados.

[17] Por solicitação da TNC Informática, em virtude de restrições contratuais, o nome da empresa e demais dados do projeto (requisitos) do projeto não podem ser divulgados.
[18] Ferramenta de Modelagem UML. Disponível em: www.sparxsystems.com

Em virtude deste processo de desenvolvimento, foi atribuído o valor 1 às variáveis α e β, representando assim que as ferramentas e frameworks utilizados na TNC oferecem uma produtividade máxima para o desenvolvimento de software. Em relação à configuração dos tipos de caso de uso, foi optado por utilizar os mesmos tipos e pesos definidos na Tabela 22.

Em virtude de restrições contratuais e assinatura de termo de confidencialidade com a empresa contratante do projeto deste estudo de caso, a TNC Informática não autorizou a divulgação dos requisitos do projeto, permitindo tão somente a divulgação dos resultados das medições.

O ANEXO C apresenta a relação de todos os casos de uso deste sistema (PROJ. I) juntamente com seus atributos e o seu tamanho, obtido através da aplicação do método FUCS.

Ao final da medição de todos os casos de uso, foi aplicada a equação 39 para obtenção do tamanho do sistema (S_{size}), totalizando 64 UCS. Usando uma produtividade de 7 horas por UCS (conforme resultados descritos na seção 5.4), o esforço de desenvolvimento do PROAD poderia ser estimado em 448 horas. No entanto, o esforço real de desenvolvimento foi de 422 horas, dando uma margem de erro de 6,2%.

5.4 AVALIAÇÃO DOS RESULTADOS

Esta seção apresenta uma análise dos resultados obtidos com a aplicação do método FUCS na avaliação de 423 casos de uso de 9 projetos, sendo 7 destes projetos desenvolvidos no TRT/SC, 1 no TST (Tribunal Superior do Trabalho) e 1 em uma empresa privada (TNC Informática). Os três projetos apresentados nos estudos de caso são parte do conjunto destes nove projetos. Em cada empresa o processo foi instanciado de acordo com suas ferramentas, frameworks e padrões de desenvolvimento. A Tabela 26 descreve a configuração dos parâmetros utilizados no processo para nas 3 empresas.

Tabela 26 - Configuração do FUCS nas empresas TRT/SC, TST e TNC

Empresa	UC_{TP}	α	β
TRT/SC	De acordo com a Tabela 22	2	2
TST	De acordo com a Tabela 22	3	4
TNC	De acordo com a Tabela 22	1	1

A análise dos resultados passa por uma análise exploratória dos dados com o intuito de verificar as relações entre os atributos técnicos, a complexidade dos casos de uso e o esforço para seu desenvolvimento.

Antes de iniciar a análise exploratória dos dados é necessário saber se o tamanho da amostra obtida nos 9 projetos é suficiente para uma análise estatística considerando uma população infinita. O cálculo do tamanho mínimo da amostra foi realizado através do software estatístico G*Power[19] utilizando um teste t para um modelo fixo e um coeficiente de regressão único. Como parâmetros do teste foi utilizado um poder do teste[20] $(1-\beta)$ de 95%, um nível de significância (α) de 5% $(\alpha = 0,05)$ e um efeito (f^2) de 0, 15. O tamanho mínimo da amostra obtido foi de 89 (*Total sample size*), conforme pode ser visto na Figura 15.

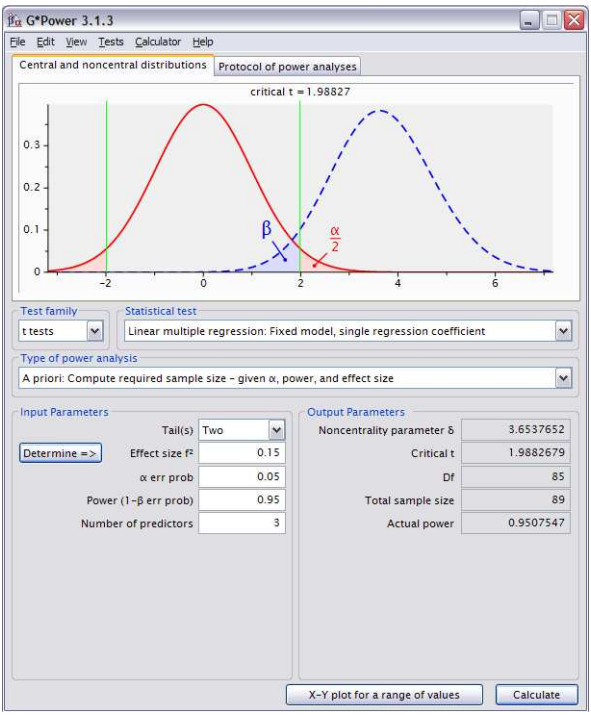

Figura 15 - Cálculo do tamanho da amostra

[19] Ferramenta para análise do poder de testes estatísticos. Disponível em: http://www.psycho.uni-duesseldorf.de/abteilungen/aap/gpower3

[20] O Poder de um teste é a probabilidade de o teste rejeitar H_0 quando H_0 é realmente falsa. (BARBETTA, REIS e BORNIA, 2009)

A amostra selecionada para o estudo foi de 423 casos de uso (ANEXO C). Para cada caso de uso da amostra foram registrados todos os atributos técnicos definidos no método FUCS, bem como o esforço total para seu desenvolvimento.

Todas as análises estatísticas demonstradas neste capítulo foram realizadas com o software estatístico MINITAB[21].

A primeira análise realizada tem por intuito verificar se os atributos técnicos definidos no método FUCS são capazes de explicar o esforço de desenvolvimento de cada caso de uso. Para isso, foi realizada uma análise de regressão considerando como variáveis independentes os atributos UC_{TP}, N_{OSNR} e N_{GUIR} e como variável dependente o esforço em horas. O resultado desta análise pode ser vista na Figura 16.

Análise de Regressão: Esforço em função de UC_{TP}, N_{SOBR}, N_{GUIR}

Equação de Regressão

Esforço $= 1,75 + 7,74\ \mathrm{UC_{TP}} + 13,8\ \mathrm{N_{SOBR}} + 10,7\ \mathrm{N_{GUIR}}$

Predictor	Coef	SE Coef	T	*p*
$Constante$	1,747	1,612	1,08	0,279
UC_{TP}	7,7448	0,5685	13,62	0,000
N_{SOBR}	13,8224	0,4961	27,86	0,000
N_{GUIR}	10,7216	0,5255	20,40	0,000

$S = 18,0517 \quad R_{sq} = 85,1\% \quad R_{sq}(adj) = 85,0\%$

$PRESS = 143201 \quad R_{sq}(pred) = 84,37\%$

Figura 16– Anal. de regressão: esforço em função de UC_{TP}, N_{OSNR}, N_{GUIR}

A análise de regressão descrita na Figura 16 mostra que os atributos técnicos utilizados no método de medição FUCS explicam 85,0% ($R_{sq}(adj)$) do esforço necessário para desenvolvimento dos casos de uso. O restante do esforço está relacionado com outras características que não estão consideradas no método. Além disso, a estatística $R_{sq}(pred)$ apresenta um valor muito significativo (84,37%), o que indica que a equação de predição obtida é eficiente para a estimativa de esforço.

[21] www.minitab.com

Esta conclusão é reforçada com o aparecimento do valor $p = 0,000$ $(p < 0,001)$ para todos os atributos técnicos definidos no método, onde *p*indica a probabilidade de o teste rejeitar a hipótese H_0 (não existe relação de *correlação* do atributo técnico em relação ao esforço) . Considerando a hipótese H_1 (hipótese alternativa), de que há uma relação de *correlação* do atributo técnico em relação ao esforço, e um nível de significância de 5% $(\alpha = 0,05)$, tem-se $(p < \alpha)$, o que significa que se deve rejeitar H_0 em favor de H_1. Conclui-se, portanto, com 95% de confiança que há uma relação de *correlação* entre cada atributo técnico definido no método em relação ao esforço.

O gráfico da Figura 17 mostra a análise dos resíduos considerando a equação de predição do esforço obtida na análise de regressão descrita na Figura 16. O histograma mostra também a existência de *outliers*, representando as situações que fogem à normalidade. No entanto, é possível observar que estas situações são poucas, e que na maioria dos casos, o esforço pode ser estimado com segurança.

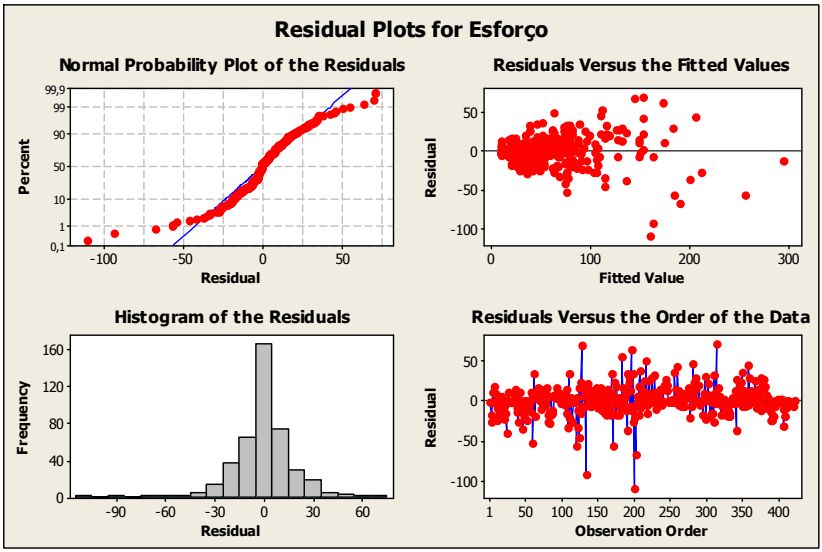

Figura 17 - resíduos da equação de predição do esforço da Figura 16

Por analogia, considerando que o tamanho dos casos de uso (UC_S) é obtido em função destes três atributos, podemos concluir que também é possível obter bons resultados de estimativas de esforço em função do tamanho do caso de uso. A Figura 18 confirma que o tamanho do caso

de uso (UC_S) pode ser utilizado como fator de predição do esforço, e que esta variável por si só explica 84,88% (estatística $R_{sq}(pred)$) do esforço para desenvolvimento de um caso de uso.

Análise de Regressão: Esforço em função de UC_S

Equação de Regressão Esforço $= 3,99 + 6,25\ UC_S$

Predictor	Coef	SE Coef	T	p
$Constante$	3,988	1,296	3,08	0,002
UC_S	6,2520	0,1268	49,31	0,000

$S = 17,9229$ $R_{sq} = 85,2\%$ $R_{sq}(adj) = 85,2\%$

$PRESS = 138500$ $R_{sq}(pred) = 84,88\%$

Figura 18- Análise de regressão: esforço em função de UC_S

Desta forma, a partir da análise da relação entre o esforço final de cada caso de uso com seu tamanho (UC_S), descrita na coluna HH/UCS da tabela do Anexo C, pode-se obter medidas descritivas que permitam a estimação do esforço em função do tamanho de cada caso de uso. A Figura 19 mostra um resumo das medidas descritivas sobre a relação esforço por tamanho de caso de uso (UCS – *use case size*).

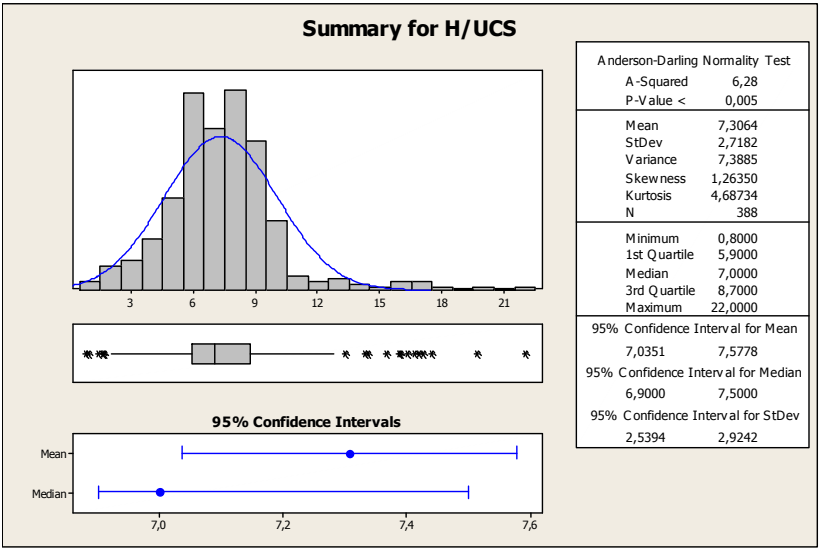

Figura 19 - Resumo estatístico da média de esforço (Horas) por UCS

Analisando a Figura 19 é possível perceber que a relação esforço/tamanho do caso de uso aproxima-se em uma distribuição normal (apesar da existência de alguns *outliers*), com uma média de 7,3 horas por UCS. A mediana oferece uma medida mais adequada, pois desconsidera os *outliers*, mostrando que nas situações normais, cada unidade de tamanho de caso de uso pode levar 7 horas para seu desenvolvimento. A Figura 19 mostra ainda que os valores concentram-se entre 5,9 (quartil inferior) e 8,7 (quartil superior). Empresas mais conservadoras poderiam utilizar o quartil superior como base para suas estimativas, chegando então a um valor de 8,7 horas/UCS.

A coluna Y da tabela do Anexo C, descreve a estimativa de esforço para o desenvolvimento de cada caso de uso considerando uma produtividade de 7 horas/UCS. A coluna ERRO da mesma tabela, mostra a diferença entre o esforço estimado e o esforço real. Os valores positivos indicam que a estimativa gerou um esforço maior do que o real, enquanto os valores negativos indicam que a estimativa gerou valores inferiores ao esforço real. A Figura 20 mostra um resumo das medidas descritivas em relação ao erro das estimativas. Como as estimativas foram geradas com base na mediana, naturalmente a mediana dos erros da estimativa é zero. O gráfico da Figura 20 também mostra que existe uma tendência de simetria em relação ao erro. Isso pode ser visto pelos quartis inferiores e superiores exibidos no gráfico. Além disso, as medidas *Minimum* e *Maximum* explicitam valores extremamente discrepantes das medidas. O valor mínimo (-62) indica que a estimativa realizada com base na mediana gerou um esforço de 62 horas a menos do que o esforço real. Algumas explicações possíveis para esta situação são:

- Erro na anotação do esforço real;
- Falta de habilidade de um desenvolvedor;
- Problemas de ambiente no desenvolvimento do caso de uso;
- Grande complexidade em alguma regra de negócio ou requisito de interface.

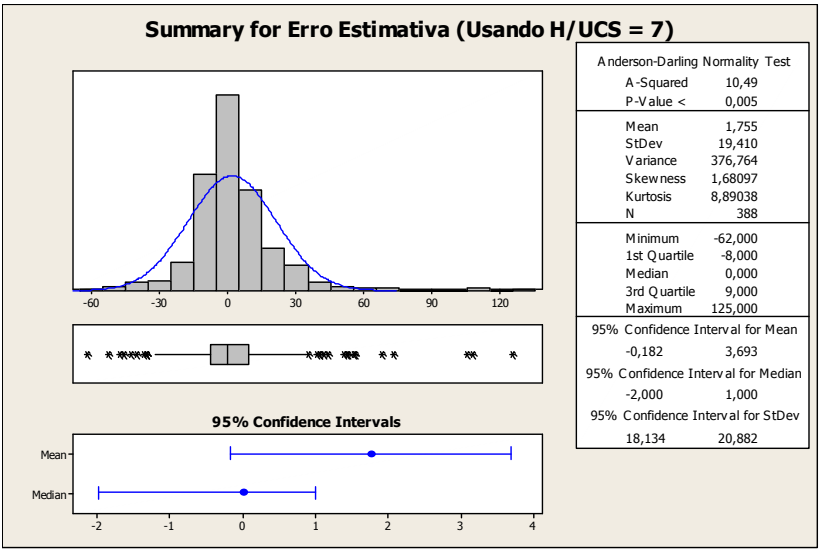

Figura 20 - Resumo das medidas descritivas do erro das estimativas

Em outro extremo, temos o valor máximo (125), que indica que a estimativa gerou um esforço de 125 horas a mais do que o realmente necessário. Esta caso também pode ser explicado em função e alguns fatores:

- Erro na anotação do esforço real;
- Extrema de habilidade de algum desenvolvedor;
- Grande número de regras de negócio ou operações com cenários parecidos, que permitiram o "reaproveitamento" de código.

No entanto, quando estimado um sistema inteiro, estes erros passam a ter menor expressão. A Tabela 27 mostra um resumo do tamanho de todos os projetos analisados, juntamente com o esforço e as estimativas. A coluna $\%_{erro}$mostra as diferenças entre o valor estimado em relação ao valor real. Valores negativos de $\%_{erro}$o indicam que a estimativa gerou um esforço menor do que o realmente necessário. Os valores positivos indicam estimativas maiores, ou seja, a produtividade no projeto foi maior do que 7 horas/UCS.

Tabela 27- Medição e Estimativa dos sistemas utilizando FUCS

Projeto	Empresa	Nº de Casos de Uso	S_{size}	Esforço Real	Estimativa Esforço	$\%_{erro}$
Projeto A	TRT/SC	32	198	1441	1386	-3,8
Projeto B	TRT/SC	17	191	1243	1337	7,6
Projeto C	TRT/SC	17	335	2232	2345	5,1
Projeto D	TRT/SC	18	149	996	1043	4,7
Projeto E	TRT/SC	120	775	4755	5425	14,1
Projeto F	TRT/SC	16	277	1772	1939	9,4
Projeto G	TRT/SC	139	973	7362	6811	-7,5
Projeto H	TST	29	238	1470	1666	13,3
Projeto I	TNC	35	64	422, 4	448	6,1

ATabela 27 mostra que todos os projetos apresentaram $\%_{erro}$ inferior a 20%, mostrando que o método FUCS atingiu seu objetivo de produzir medições e estimativas com margem de erros inferiores a 20%.

É importante destacar, que a instanciação de forma adequada do processo para as características de cada empresa permitiu a utilização de uma constante de produtividade (7 H/UCS), mesmo para empresas diferentes com processos diferentes, mantendo as estimativas dentro da margem de erro aceitável e definida nos objetivos deste trabalho.

Para finalizar a análise dos resultados, foi realizada uma análise de correlação entre cada par de atributos utilizados no processo de medição do método FUCS com o intuito de verificar se um atributo está correlacionado a outro, ou seja, se há uma relação de *correlação*, entre dois atributos (BARBETTA, REIS e BORNIA, 2009), como identificado por Kitchenham em relação ao método de análise de pontos de função (KITCHENHAM, 1997). A Tabela 28 mostra os valores das correlações entre cada atributo do método FUCS e o tamanho dos casos de uso.

Tabela 28 - Correlação (r) entre os atributos e medidas do método FUCS

	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	**Esforço**
UC_{TP}		0,144	0,260	0,452	0,458
N_{SOBR}	0,144		0,368	0,805	0,761
N_{GUIR}	0,260	0,368		0,793	0,72
UC_S	0,452	0,805	0,793		0,923
Esforço	0,458	0,761	0,702	0,923	

Os valores obtidos mostram correlações pouco significativas entre os atributos do método FUCS. Utilizando a tabela de Rugg (Tabela 29) conclui-se que a correlação entre UC_{TP} e N_{SOBR} é desprezível. A correlação entre UC_{TP} e N_{GUIR} é baixa, e não necessita preocupações. Já a correlação entre N_{SOBR} e N_{GUIR} é classificada como "apreciável". No entanto, seu valor está muito próximo do limite inferior do intervalo das correlações apreciáveis. Além disso, o atributo técnico N_{SOBR} e N_{GUIR} contabilizam tipos de requisitos distintos. Desta forma, é possível concluir que a correlação apresentada entre estes atributos não requer preocupações.

Tabela 29 - Tabela de Rugg para interpretação da correlação entre variáveis

Intervalo	**Interpretação da correlação**
$r < 0,15$	desprezível
$0,15 < r < 0,29$	baixa
$0,30 < r < 0,49$	apreciável
$r > 0,5$	acentuada

Fonte: (RODRIGUES, 2002)

Os valores começam a apresentar uma correlação mais significativa em relação aos atributos UC_{TP} , N_{SOBR} e N_{GUIR} com o tamanho docaso de uso (UC_S**)** e ao esforço para desenvolvimento. Estes valores indicam que os atributos utilizados no processo de medição FUCS não possuem correlação entre si, porém apresentam relação de *correlação* com o tamanho do caso de uso e o esforço de desenvolvimento.

6 CONCLUSÕES E TRABALHOS FUTUROS

6.1 CONCLUSÕES

Os principais métodos de medição e estimativa de software, apresentam limitações que dificultam sua aplicação em processos licitatórios e contratos de terceirização dos serviços de desenvolvimento de software.

Com o intuito de superar as limitações dos métodos atuais de medição de software, este trabalho apresentou um novo método para medição e estimativa de software, o método FUCS (*Full Use Case Size*).

O método FUCS é direcionado para processos de software onde os requisitos funcionais são especificados através de casos de uso. Ele avalia 3 atributos técnicos relacionados a cada caso de uso: tipo do caso de uso e seu respectivo peso (UC_{TP}), quantidade de operações de sistema e regras de negócio (N_{SOBR}) e quantidade de requisitos de interface gráfica do usuário (N_{GUIR}). Com base nestes 3 atributos foi definida uma fórmula que determina o tamanho de um caso de uso (UC_S). O tamanho do sistema (S_{size}) é então definido como o somatório dos tamanhos dos casos de uso que o compõe. Tanto o tamanho dos casos de uso quanto o tamanho do sistema são medidos sob uma nova métrica, chamada UCS (*Use Case Size*).

A principal vantagem do método FUCS em relação aos demais métodos de medição e estimativa de software está no fato de que os atributos utilizados no seu processo de medição podem ser facilmente obtidos nas fases iniciais do projeto, viabilizando assim seu uso em processos licitatórios e contratos de terceirização do desenvolvimento de software. Outra vantagem do método FUCS é ouso de escalas objetivas (tipológicas e razão) na mensuração dos atributos. Isso o torna um método mais objetivo e simples de ser aplicado, além de diminuir a necessidade de especialistas no processo de medição e as divergências em medições quando realizadas por analistas diferentes.

Em relação aos outros métodos baseados em casos de uso como os apresentados em (KARNER, 1993), (BRAZ e VERGILIO, 2006), (VIEIRA, 2007) e (PERIYASAMY e GHODE, 2009), o método FUCS apresenta a vantagem de analisar a complexidade dos casos de uso de forma mais objetiva, sem a necessidade de classificação da complexidade em escalas categóricas e sem a necessidade de ter os casos de uso expandidos. Esta abordagem minimiza conflitos e divergências nas classificações de casos de uso por pessoas diferentes,

além de permitir sua aplicação logo nas fases iniciais dos projetos, onde os casos de uso ainda não estão detalhados.

Em (ROBIOLO e OROSCO, 2007), são apresentados dois métodos de estimativas baseados em casos de uso que não exigem que os casos de uso estejam expandidos. No entanto, sua aplicação depende de uma análise sintática dos textos dos casos de uso que, se realizada manualmente, torna o processo de medição muito trabalhoso e sujeito a muitas falhas, principalmente em sistemas que tenham muitos casos de uso. Para automatizar o processo, é necessário contar com um analisador sintático com suporte a vários idiomas e a um dicionário de sinônimos que permita identificar quais substantivos e verbos são equivalentes.

Em relação ao método APF (ALBRECHT, 1979) e seus derivados, como os métodos MK II (SYMONS, 1991) e o COSMIC-FFP (COSMIC, 2007), o FUCS apresenta a vantagem de estabelecer uma medida com base em atributos que podem ser obtidos logo no início do projeto sem a necessidade dos requisitos do sistema estarem totalmente detalhados.

Outra vantagem do método FUCS em relação ao método APF, está no fato de que os atributos utilizados no processo de medição do FUCS não possuem correlação entre si, ou seja, não apresentam relações de *correlação.* Além disso, o método FUCS estabelece diretrizes de medição com foco na obtenção de uma medida que permita derivar o esforço para desenvolvimento de cada caso de uso, e por consequência o desenvolvimento do sistema. Já o método APF estabelece diretrizes com foco na medição das funcionalidades do sistema, independentemente do esforço necessário para desenvolvê-las. Por este motivo, o método APF é considerado um método de medição de tamanho funcional, enquanto o método FUCS não pode ser assim classificado.

Apesar de o método FUCS ter sido avaliado somente em projetos de sistemas de informação, isso não restringe sua aplicação somente a estes tipos de projetos. A aplicação em outros tipos de projetos pode ser realizada desde que seus parâmetros sejam devidamente configurados.

Em relação aos resultados da aplicação do método, os estudos de casos demonstram que no pior caso o método produziu uma estimativa com um erro inferior a 15% sendo que os resultados esperados com este trabalho limitam-se a uma margem de erro de até 20%. Além disso, a análise de correlação entre os atributos técnicos e o esforço final para o desenvolvimento de cada caso de uso, mostrado na Tabela 28, demonstra que os atributos técnicos definidos no método são adequados para a determinação do tamanho dos casos de uso e possuem forte influência no esforço total para desenvolvimento de cada caso de uso.

É importante destacar que o método FUCS deve ser utilizado no contexto de medição de um projeto inteiro e não somente de casos de uso isolados. Sua utilização em casos de uso de forma isolada pode resultar em discrepâncias muito grandes, como pode ser visto nos casos de uso UC195 e UC201 (Anexo C), ambos do Projeto C, que possuem o mesmo tamanho (25 UCS), porém um esforço para desenvolvimento muito distinto de 156 horas e 50 horas, respectivamente. Apesar de esta discrepância ocorrer no projeto C, quando aplicado o método FUCS para todo o conjunto de casos de uso do projeto, o percentual de erro em relação ao esforço de desenvolvimento e o esforço estimado com o método foi muito baixo, ficando em torno de 5,1%, como mostra a Tabela 27.

Outro ponto a ser enfatizado é que, para a utilização do método FUCS em processos licitatórios e contratos de terceirização de serviços, as configurações de seus parâmetros devem estar previamente estabelecidas nos editais e/ou contratos. Esta medida é fundamental para o cumprimento do objetivo principal do FUCS, que é oferecer um método mais objetivo para medição e estimativa de software.

Em relação à objetividade do método FUCS, é importante destacar que esta característica está relacionada à fase de medição do método, onde os atributos de software são medidos e contados. Já a fase de instanciação (configuração) do método apresenta alguns pontos de subjetividade, principalmente na avaliação dos pesos dos tipos de casos de uso (UC_{TP}) e na configuração dos valores dos parâmetros α e β. No entanto, esta subjetividade é minimizada com a definição prévia destas configurações.

Assim, a simplicidade como os atributos do software são medidos e contados, aliada à definição prévia das configurações do método, faz com que os resultados das medições, mesmo quando realizadas por pessoas diferentes, apresentem valores menos discrepantes do que os outros métodos.

Atualmente o método FUCS faz parte do processo de desenvolvimento do TRT/SC e tem sido utilizado nos processos licitatórios e contratos de terceirização dos serviços de desenvolvimento de software deste tribunal. Sua aplicação no TRT/SC tem mostrado que ele tem sido bastante preciso tanto nas estimativas de projetos quanto na negociação de mudanças nos requisitos dos projetos que ocorrem entre equipes de desenvolvimento e usuários e entre equipes de desenvolvimento e fábricas de software. As eventuais divergências entre equipes de desenvolvimento e empresas terceirizadas são resolvidas através de reuniões onde cada uma das partes apresenta o que foi

considerado na sua contagem e a justificativa de suas considerações com base nas diretrizes estabelecidas no método.

6.2 TRABALHOS FUTUROS

As diretrizes estabelecidas neste trabalho para o método FUCS foram avaliadas somente a projetos de desenvolvimento de sistemas de informações.

Como trabalhos futuros poderiam ser feitos novos estudos para a avaliação do método FUCS em outros tipos de projetos de software, como por exemplo, projetos de software de tempo real.

Outra linha de pesquisa que poderia ser desenvolvida é a definição de novas diretrizes para permitir a aplicação do FUCS em projetos de manutenção de sistemas de informação, ou seja, em projetos que exijam alterações em casos de uso já desenvolvidos.

Além disso, poderiam ser estudadas algumas transformações dos dados, como por exemplo o uso de uma transformação logarítmica, para tentar minimizar os efeitos dos *outliers* representados na Figura 17.

A possibilidade de configuração de α e β específicos para cada tipo de caso de uso, ao invés de serem utilizados para todo o projeto, também poderia ser avaliada em um trabalho futuro. Isso poderia trazer uma maior precisão na configuração do método para cada empresa e, consequentemente, nas medições e estimativas realizadas através do método.

Outro trabalho futuro que poderia ser desenvolvido é avaliar se existe alguma correlação entre a quantidade de casos de uso e o resultado final obtido com a aplicação do método. Para isso seria necessário aplicar o método em mais projetos, aumentando assim a amostra dos dados e possibilitando uma análise exploratória destas variáveis.

REFERÊNCIAS

ALBRECHT, A. **Measuring Application Development Productivity**. Proceedings of the IBM SHARE/GUIDE Applications Development Symposium, Monterey, CA, out. 1979. 83–92.

ALVES, A. M. **Contratação de Produtos e Serviços de Software**. Universidade Estadual de Campinas. Campinas, SP, Brasil. 2002.

AMBLER, Scott. **Modelagem Ágil** Porto Alegre: Boookman, 2004.

ANDA, B. et al. **Estimating Software Development Effort Based on Use Cases-Experiences from Industry**. Proceedings of the 4th International Conference on The Unified Modeling Language, Modeling Languages, Concepts, and Tools (UML'2001), London, UK , 2001.

ANDRADE, E. L. P. D. **Pontos de Casos de Uso e Pontos de Função na Gestão de Estimativa de Tamanho de Projetos de Software Orientados a Objetos**. Universidade Católica de Brasília. Brasília. 2004.

BARBETTA, Pedro Alberto; REIS, Marcelo Menezes; BORNIA, Antonio Cezar. **Estatística para cursos de e Engenharia e Informática** 2a Edição. ed. [S.l.]: Atlas, 2009.

BASILI, V. R. **Software development: a paradigm for the future**. Computer Software and Applications Conference (COMPSAC 89), 1989. 471-485.

BASILI, Victor ; CALDIERA, G. ; ROMBACH, H.D.. **Chapter Measurement. In Encyclopedia of Software Engineering** [S.l.]: pp. 646-661, John Wiley & Sons, Inc, 1994.

BRASIL. **Lei 8666/1993, de 21 de junho de 1993.**. **Regulamenta o art. 37, inciso XXI, da Constituição Federal, institui normas para**

licitações e contratos da Administração Pública e dá outras providências, 1993. Disponivel em: <http://www.planalto.gov.br/ccivil_03/Leis/L8666cons.htm>. Acesso em: 30 jun. 2011. url: http://www.planalto.gov.br/ccivil_03/Leis/L8666cons.htm.

BRAZ, M. R.; VERGILIO, S. R. **Software Effort Estimation Based on Use Cases**. Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC '06) , Washington, DC, USA, 2006.

CHRISTENSEN, Mark ; THAYER, Richard H. **The Project Manager's Guide to Software Engineering's Best Practices** [S.l.]: Wiley-IEEE Computer Society Press, 2002. ISBN ISBN: 978-0-7695-1199-3.

COCKBURN, Alistair. **Escrevendo Casos de Uso Eficazes:** Um guia prático para desenvolvedores de software Reimpressão 2008. ed. [S.l.]: Bookman, 2005. ISBN ISBN: 978-363-0457-1.

COHN, Mike. **Agile Estimating and Planning** [S.l.]: Prentice Hall, 2005. 1 Edição p. ISBN ISBN: 9780131479418.

CONSTANTINE, Larry L.; LOCKWOOD, Lucy AD. **Software for Use:** A Practical Guide to the Models and Methods of Usage-Centered Design". New York: ACM Press. 1999. New York, USA: Addison-Wesley, 1999. ISBN 978-02-019-2478-7.

COSMIC. **The COSMIC Functional Size Measurement Method: Measurement Manual. Version 3.0.1**. **Commom Software Measurement International Consortium**, 2007. Disponivel em: <http://www.cosmicon.com/portal/public/COSMIC%20Method%20v3.0.1%20Measurement%20Manual.pdf>. Acesso em: 30 jun. 2011.

EBERT, Christof et al. **Best Practices in Software Measurement** New York: Springer, 2005. ISBN ISBN 3-540-20867-4.

FAN, W. et al. **Extended Use Case Points Method for Software Cost Estimation**. Computational Intelligence and Software Engineering, dez. 2009. 1-5, 11-13. url: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5364706&isnumber=5362501.

FENTON, Norman E.; PFLEEGER, Shari Lawrence. **Software Metrics:** A Rigorous and Practical Approach, Revised [S.l.]: Course Technology, 1998. ISBN ISBN: 978-05-349-5425-3.

IBARRA, G. B.; VILAIN, P. **Estendendo a Contagem de Pontos de Caso de Uso para Melhorar a Estimativa do Tamanho de Projetos de Software**. Congresso Iberoamericano de Engenharia de Software (CIbSE'2010), Cuenca - Equador, 2010a.

IBARRA, G. B.; VILAIN, P. **Estendendo a Contagem de Pontos de Caso de Uso para Aplicação na Terceirização do Desenvolvimento de Software**. Simpósio Brasileiro de Sistemas de Informação, Marabá/PA, Brasil, 2010b.

IBARRA, G. B.; VILAIN, P. **Software Estimation Based on Use Case Size**. Brazilian Conference on Software: Theory and Practice (CBSoft"2010) on XXIV Brazilian Symposium on Software Engineering (SBES'2010), Salvador/BA, 2010c. Disponível em: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5629587.
Ultimo acesso em: 31/jul/2011.

IFPUG. **Function Point Counting Practices Manual 4.3.1**. **International Function Points User Group**, 2010. Disponivel em: <http://www.ifpug.org/publications/CPM%204.3.1%20TOC%20Excerpts.pdf>. Acesso em: 30 jun. 2011.

JACOBSON, Ivar. **The Use-Case Construct in Object-Oriented Software Engineering. In:** Scenario-Based Design: Envisioning Work and Technology in System Development [S.l.]: John Wiley & Sons, 1995. 309-337 p.

JACOBSON, Ivar ; BOOCH, Grady ; RUMBAUGH, James. **The Unified Software Development Process** 1a Edição. ed. [S.l.]: Addison-Wesley Professional, 1999.

KARNER, G. **Metrics for Objectory**. University of Linkopmg. Sweden. 1993.

KITCHENHAM, B.; KÄNSÄLÄ, K. **Inter-item correlations among function points**. Proceedings 15th International Conference on Software Engineering, maio 1993. 477-480.

KITCHENHAM, B. **Counterpoint: The Problem with Function Points**. IEEE Software, vol. 14, no. 2, 1997. 29,31.

KRUCHTEN, Philippe. **The Rational Unified Process:** An Introduction 3a Edição. ed. [S.l.]: Addison-Wesley, 2004. ISBN ISBN: 978-02-017-0710-6.

KUSUMOTO, S. et al. **Estimating Effort by Use Case Points: Method, Tool and Case Study**. In Proceedings of the Software Metrics, 10th International Symposium (METRICS '04), Washington, DC, USA, 2004. 292-299.

LARMAN, Craig. **Applying UML and patterns:** an introduction to object oriented 3a. Edição. ed. [S.l.]: Prentice Hall, 2004. ISBN ISBN: 978-01-314-8906-6.

LAVAZZA, L. A.; BIANCO, V. D.; CARAVAGLIA, C. **Model-based Functional Size Measurement**. Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement (ESEM '08), New York, NY, USA, 2008.

LOKAN, C. J. **An Empirical Study of the Correlations Between Function Point Elements**. Proceedings Sixth International Software Metrics Symposium, 1999. 200-206.

MARTINS, Sérgio Pinto. **A terceirização e o Direito do Trabalho** 9ª Edição. ed. [S.l.]: Atlas, 2009.

OSTVOLD, K. M.; HAUGEN, N. C.; BENESTAD, C. H. **Using planning poker for combining expert estimates in software projects**. Journal of Systems and Software, dez. 2008.

PANADIAN, C. Ravindranath. **Software Metrics:** A Guide to Planning, Analysis, and Application 1a Edição. ed. [S.l.]: Auerbach Publications, 2003. ISBN ISBN: 978-08-493-1661-6.

PERIYASAMY, K.; GHODE, A. **Cost Estimation using extended Use Case Point (e-UCP)**. Computational Intelligence and Software Engineering, dez. 2009. 1-5, 11-13. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5364515&isnumber=5362501.

PETERS, James ; PEDRYCZ, Witold. **Engenharia de Software:** Teoria e Prática 1a Edição. ed. [S.l.]: Campus, 2001.

PRESSMAN, Roger S. **Software Engineering:** A Practitioner's Approach 7ª Edição. ed. [S.l.]: MCGRAW-HILL, 2009. ISBN ISBN: 0073375977.

RIBU, K. **Estimating object oriented software projects with use cases**. University of Oslo. Oslo. 2001.

ROBIOLO, G.; OROSCO, R. **An Alternative Method Employing Uses Cases for Early Effort Estimation**. Software Engineering Workshop (SEW 2007), 2007. 89-98. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4402768&isnumber=4402752.

ROBIOLO, G.; BADANO, C.; OROSCO, R. **Transactions and paths: Two use case based metrics which improve the early effort estimation**. In Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement (ESEM '09),

Washington, DC, USA, 2009. 422-425. URL: http://dx.doi.org/10.1109/ESEM.2009.5316021.

RODRIGUES, Pedro Carvalho. **Bioestatística** Niterói/RJ: EdUFF, 2002. ISBN ISBN: 85-228-0341-2.

SILVEIRA, M. C.; VIDAL, R. M. **Software Reuse with Use Case Patterns**. SpringerLink, Advances in Object-Oriented Information Systems, Lecture Notes in Computer Science, 2002. 45-50.

SLTI. **INSTRUÇÃO NORMATIVA Nº 4, 19 de maio de 2008**. **Portal do Governo Eletrônico**, 2008. Disponivel em: <http://www.governoeletronico.gov.br/anexos/instrucao-normativa-n-04>. Acesso em: 30 jun. 2011.

SLTI. **Roteiro de Métricas**. **Portal do Governo Eletrônico**, 2010. Disponivel em: <http://www.governoeletronico.gov.br/biblioteca/arquivos/roteiro-de-metricas-de-software-do-sisp/download>. Acesso em: 30 jun. 2011.

SMITH, John. **The estimation of effort based on use cases** [S.l.]: Rational Software White Paper., 1999.

SOMMERVILLE, Ian. **Engenharia de Software** 8a. Edição. ed. São Paulo: Pearson Addison-Wesley, 2007. ISBN ISBN 978-85-88639-28-7.

SYMONS, Charles. **Software Sizing and Estimating:** Mk II Fpa (Function Point Analysis) [S.l.]: Wiley-Interscience, 1991. ISBN ISBN: 978-04-719-2985-7.

TCU. **Tribunal de Contas da União. Acórdão 786/2006** , 2006. Disponivel em: <http://contas.tcu.gov.br/portaltextual/MostraDocumento?lnk=(acordao+adj+786/2006+adj+plenario)[idtd][b001]>. Acesso em: 30 jun. 2011.

TCU. **Orientações sobre licitações, contratos e convênios**. **Portal do TCU**, 2011. Disponivel em: <http://portal2.tcu.gov.br/portal/page/portal/TCU/comunidades/licitacoes_contratos/LICITACOES_CONTRATOS_3AED.pdf>. Acesso em: 30 jun. 2011.

TRIBUNAL DE CONTAS DA UNIÃO. **Licitações: Conceitos e Princípios**. **Portal do TCU**. Disponivel em: <http://portal2.tcu.gov.br/portal/page/portal/TCU/comunidades/licitacoes_contratos/2%20Licita%C3%A7%C3%B5es-Conceitos%20e%20Princ%C3%ADpios.pdf>. Acesso em: 30 Junho 2011.

VAZQUEZ, Carlos Eduardo; SIMÕES, Guilherme Siqueira; ALBERT, Renato Machado. **Análise de Pontos de Função:** Medição, Estimativas e Gerenciamento de Projetos de Software 10a. Edição Revisada. ed. São Paulo: Érica, 2010. ISBN ISBN: 978-7194-899-0.

VIEIRA, E. L. **Uso do Conceito de Passos Obrigatórios para Aprimorar o Processo de Contagem do Método "Pontos de Caso de Uso"**. Universidade Federal de Santa Catarina. Florianópolis. 2007.

VIEIRA, E. L.; WAZLAWICK, R. S. **Teoria Explanatória para Estimativa Baseada em Casos de Uso no Desenvolvimento Orientado a Objetos**. XXXII CLEI - Conferência Latinoamericana de Informática, Santiago, Chile, 2006.

WAZLAWICK, Raul Sidnei. **Análise e Projeto de Sistemas de Informação Orientados a Objetos** Rio de Janeiro: Elsevier, 2004.

WIKIBOOKS. **Licitações e contratos públicos no Brasil/Licitações/Noções gerais**. **WIKIBOOKS**, 2011. Disponivel em: <http://pt.wikibooks.org/wiki/Licita%C3%A7%C3%B5es_e_contratos_p%C3%BAblicos_no_Brasil/Licita%C3%A7%C3%B5es/No%C3%A7%C3%B5es_gerais>. Acesso em: 30 jun. 2011.

WIKIBOOKS. **Noções Básicas sobre Licitações**. **WIKIBOOKS**. Disponivel em: <http://pt.wikibooks.org/wiki/Licita%C3%A7%C3%B5es_e_contratos_p%C3%BAblicos_no_Brasil/Licita%C3%A7%C3%B5es/No%C3%A7%C3%B5es_gerais>. Acesso em: 30 jun. 2011.

YANG, Y. et al. **Phase distribution of software development effort**. Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement (ESEM '08), 2008.

ANEXO A – Método para Identificação dos Casos de Uso

O método descrito neste anexo é uma adaptação do método descrito por Larman (LARMAN, 2004). Uma visão geral deste método é apresentada na Figura 21.

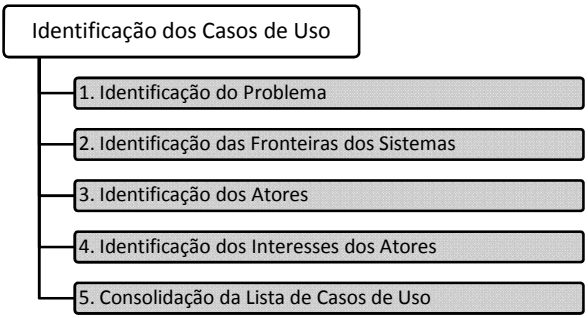

Figura 21- Visão Geral do Método para Identificação dos Casos de Uso

As seções seguintes demonstram com mais detalhes cada passo deste método.

A.1 IDENTIFICAÇÃO DO PROBLEMA

Como primeiro passo o analista deve identificar o problema para o qual o sistema pretende oferecer uma solução. A identificação do problema pode determinar o tipo do projeto: (i) desenvolvimento de um sistema novo, (ii) desenvolvimento de um novo módulo para um sistema existente, ou (iii) melhorias em um sistema existente. Existem casos em que a solução de um problema pode envolver mais de um sistema.

O principal objetivo da identificação do problema é ganhar uma melhor compreensão do quê precisa ser resolvido antes de iniciar o desenvolvimento de um projeto. Algumas diretrizes para alcançar estes objetivos são as seguintes:

- Atingir consenso em relação à definição do problema;
- Compreender as causas do “problema”, ou seja, o quê gera o problema;
- Identificar as premissas e restrições sobre a solução.

O artefato ilustrado na Figura 22 pode auxiliar no registro desta atividade.

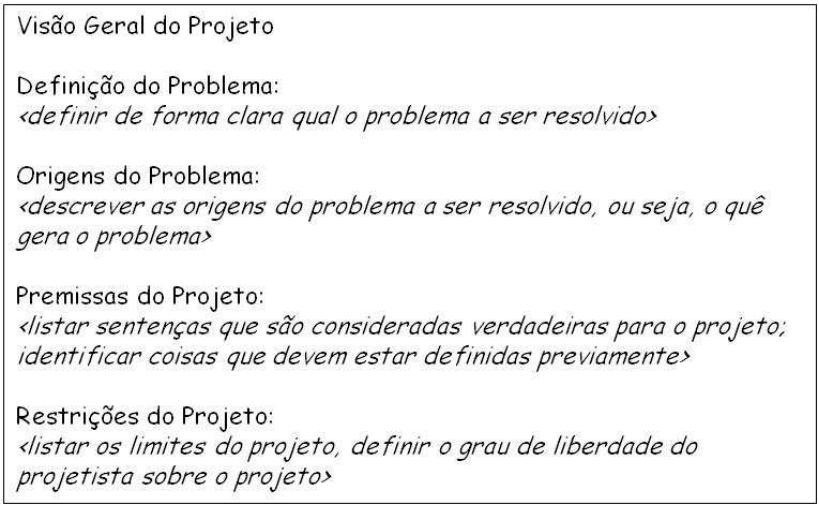

Figura 22 – Ex. de artefato para registro do problema a ser resolvido

A.2 IDENTIFICAÇÃO DAS FRONTEIRAS DOS SISTEMAS

A solução de um problema pode impactar em um ou mais sistemas, e de maneiras diferentes. A Figura 2223 mostra um exemplo de uma solução de problema que envolve mais do que um sistema sob a perspectiva das fronteiras dos sistemas envolvidos. Neste exemplo, a solução do problema é composta pelo desenvolvimento de um novo sistema e por melhorias em um sistema existente. Nos casos em que uma solução envolver mais do que um sistema, é muito importante "quebrar" o problema em problemas menores, relacionados especificamente a cada sistema, e medi-los separadamente. Isso se justifica pelo fato de que as medidas podem ser utilizadas na previsão do esforço do projeto e que cada projeto pode ser desenvolvido em plataformas e tecnologias diferentes, o que muda a taxa de produtividade por unidade de medida.

A identificação da fronteira de cada sistema envolvido na solução dos problemas é muito importante para a identificação dos casos de uso que compõem cada escopo de medição. A fronteira do sistema é uma interface conceitual que define os limites entre a solução do problema e o mundo real que a rodeia. Com a determinação da fronteira é possível

obter uma visão geral dos tipos de interação externa com o sistema. A Figura 23 ilustra uma primeira visão sob a perspectiva da solução de um problema.

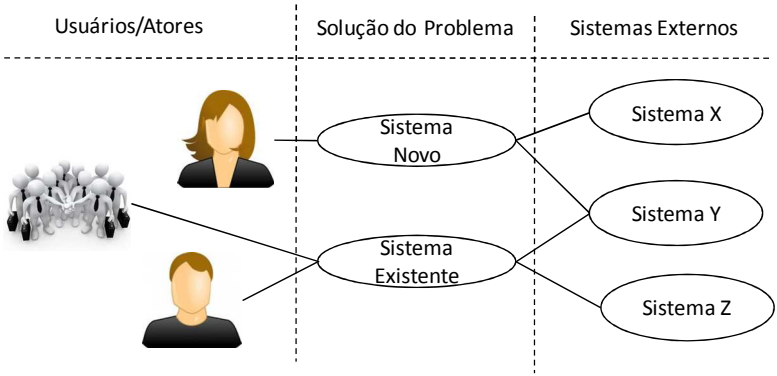

Figura 23 - Fronteira: 1ª visão sob a perspectiva da solução de um problema

O Quadro 5ilustra um exemplo de registro desta atividade.

Sistema envolvido na solução do problema	**Tipo de Projeto (descrição do impacto)**
Sistema A	Desenvolvimento de um novo sistema
Sistema B	Criação de um novo módulo no sistema B
Sistema C	Melhorias em funcionalidades existentes e criação de novas funcionalidades

Quadro 5 - Ex. de Registro das Atividades para Identificação dos UC

A.3 IDENTIFICAÇÃO DOS ATORES DO SISTEMA

Para entender melhor um sistema é muito importante identificar quem utilizará o sistema, ou seja, os atores. Um ator pode ser uma pessoa (Fulano), um cargo (Diretor Financeiro), um grupo de pessoas (Setor de Compras) ou até mesmo um sistema externo (Recursos Humanos) que interage com o sistema com um propósito definido.

A identificação dos atores auxilia na definição das interfaces externas e na determinação do escopo do sistema. Associados aos atores existem os casos de uso que descrevem o que cada ator espera do sistema. Sem identificar os atores, a avaliação da abrangência do conjunto de casos de uso que compõe o projeto torna-se uma atividade quase impossível. Além disso, a falta de um ator em um projeto de

desenvolvimento ou melhoria pode resultar em perdas de perspectivas importantes sob a ótica dos patrocinadores, resultando em uma solução que não satisfaz todas as necessidades do cliente.

Encontrar os atores envolvidos na solução de um problema não é uma tarefa trivial, porém, algumas perguntas podem ajudar a identificá-los:

1. Onde o sistema vai ser usado?
2. Quais pessoas/cargos irão operar o sistema?
3. Quem vai manter ou se beneficiar de informações do sistema?
4. Quem vai manter e configurar o sistema?
5. Quais outros sistemas podem interagir com o sistema?
6. Existe alguma rotina que precisa ser rodada automaticamente?

De maneira geral as perguntas de 1 a 4 auxiliam o analista a extrair os atores que utilizarão o sistema. As perguntas 5 e 6 auxiliam na identificação de outros sistemas que interagem com o sistema analisado, além de contribuir também no esclarecimento das fronteiras entre os sistemas.

O registro desta atividade pode ser consolidado complementando o Quadro 5. O resultado é mostrado no Quadro 6.

Sistema envolvido na solução do problema	**Tipo de Projeto (descrição do impacto)**	**Atores Identificados**
Sistema A	Desenvolvimento de um novo sistema	Diretor da Empresa, Setor Financeiro, Grupo Administrador, Sistema Operacional, Sistema Y
Sistema B	Criação de um novo módulo no sistema B	Agendador de Tarefas, Área Técnica
Sistema C	Melhorias em funcionalidades existentes e criação de novas funcionalidades	Setor de Recursos Humanos

Quadro 6 – Ex. de Registro das Atividades para Identificação dos UC

A.4 IDENTIFICAÇÃO DOS INTERESSES DOS ATORES

Depois de identificar os atores, o analista deve se preocupar em identificar quais são os interesses dos atores em cada sistema envolvido na solução do problema. Dos interesses dos atores é possível encontrar os principais casos de uso de um sistema. Algumas perguntas, feitas a cada ator identificado, podem auxiliar nesta atividade:

1. Quais são os objetivos pretendidos pelo ator com o sistema?
2. Os objetivos pretendidos podem ser realizados através de operações menores?
3. Quais são os principais conceitos de interesse do ator e quais operações básicas pretende realizar?
4. Existem outras operações diferentes das já observadas e que são pretendidas pelo ator?
5. O ator deseja que algumas regras possam ser customizadas através do sistema?
6. Quais os interesses dos sistemas externos que interagem com o sistema?
7. O ator deseja ser informado sobre alguma ocorrência externa ao sistema?

A.5 CONSOLIDAÇÃO DA LISTA DE CASOS DE USO

Depois de identificados os atores e seus interesses, o analista terá uma lista “bruta” de atores e casos de uso. O trabalho a ser feito neste momento é organizar esta lista de forma a garantir que cada caso de uso seja único no sistema. Algumas diretrizes devem ser observadas neste momento:

- Agrupe os casos de uso semelhantes;
- Crie ou agrupe os conjuntos completos de casos de uso das operações básicas (Inclusão, Alteração, Exclusão, Consulta) sobre um mesmo conceito em um caso de uso do tipo “Manter <Conceito>” mantendo os casos de uso menores associados ao caso de uso maior (extensão e inclusão);
- Verifique se os atores que realizam as operações básicas sobre um objeto são os mesmos para todas as operações. Neste caso, mude as associações de cada ator com cada operação básica para o caso de uso “Manter <conceito>”.

ANEXO B – Exemplo de Medição com o FUCS

Este anexo apresenta um exemplo de aplicação do método FUCS. Para este fim, será definido um projeto de um sistema fictício de faturamento para controle de emissão de notas fiscais (SISFAT).

Para tornar o exemplo mais didático, este anexo foi estruturado em 3 seções. A primeira seção mostra as características do sistema SISFAT e como ele foi concebido. A segunda seção demonstra como o FUCS foi instanciado para este contexto de medição. E, por fim, a terceira e última seção mostra o cálculo do tamanho do sistema de acordo com o método FUCS.

B.1 SISTEMA DE FATURAMENTO (SISFAT)

O SISFAT é uma típica aplicação desktop para controle de emissão de notas fiscais. A aplicação consiste basicamente em um cadastro de clientes e na geração de notas fiscais.

A demanda pelo desenvolvimento deste projeto foi formalmente descrita através de uma ata de reunião onde participaram a equipe de analistas de negócio da empresa e o demandante do projeto (cliente). A ata da reunião (Figura 24) foi assinada pelos envolvidos, dando origem, então, ao projeto de desenvolvimento.

Ata da Reunião

Objetivo: Abertura do Projeto – Sistema de Faturamento (SISFAT)
Participantes: Presidente da Empresa, Chefe do Setor de Faturamento e Equipe de Analistas de Negócio

Demanda/Resumo Executivo
O presidente abre a reunião falando da importância de melhorar os controles sobre o faturamento da empresa e solicita formalmente a abertura de um projeto de desenvolvimento de um sistema para cadastramento de clientes e emissão de notas fiscais. Após sua explanação, o presidente passou a palavra para o Chefe do Setor de Faturamento, que descreveu seu processo de negócio.
O chefe do setor de faturamento descreve que seu trabalho consiste basicamente em manter o cadastro de clientes da empresa e emitir as notas fiscais. Informa que é muito importante para a empresa que o cadastro dos clientes seja completo, inclusive com informações do endereço do cliente e contatos. Ele informa que gostaria muito que o CEP do endereço fosse válido segundo os Correios. Diz também que é muito importante que o sistema permita identificar a cidade do cliente, pois possivelmente a empresa abrirá novas filiais em breve, e assim, esta informação seria muito importante para futuras análises estatísticas e relatórios. No entanto, neste primeiro momento não há a necessidade destes relatórios, pois a empresa não tem ainda filiais. Informa ainda que para a geração de uma

nota, o sistema deverá levar em consideração a cidade do cliente pois para cada cidade existe uma alíquota de ISS diferente.
Em relação às notas fiscais, o chefe informa que uma nota só pode ser emitida para um cliente já cadastrado. Informa também que em uma mesma nota pode conter mais do que um serviço a ser faturado, e que o sistema deve ainda permitir que seja informado quais os impostos que devem ser deduzidos na nota fiscal. Esclarece ainda que os impostos são definidos em uma planilhaexcel obtida diretamente do site do governo.

Figura 24- Ata da reunião do sistema de faturamento

Após a reunião e com algumas outras informações esclarecidas por telefone com o chefe do setor de faturamento, os analistas de negócio se reuniram para desenhar os processos de negócio observados durante a reunião. Este trabalho resultou no diagrama de BPMN ilustrado na Figura 25.

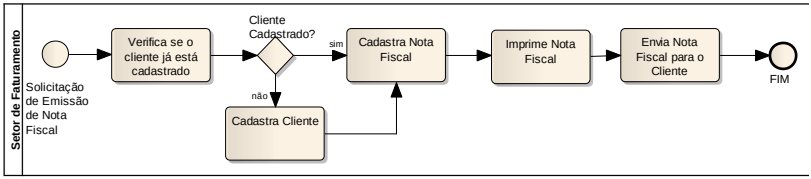

Figura 25 - Processo de Negócio: Emissão de Nota Fiscal

O processo foi então demonstrado aos envolvidos no projeto, que o aprovaram. A partir deste momento, a equipe de analistas de sistemas iniciou o levantamento de requisitos com o usuário (chefe do setor de faturamento). A primeira tarefa foi encontrar as principais funcionalidades do sistema, que resultou no diagrama de casos de uso da Figura 26.

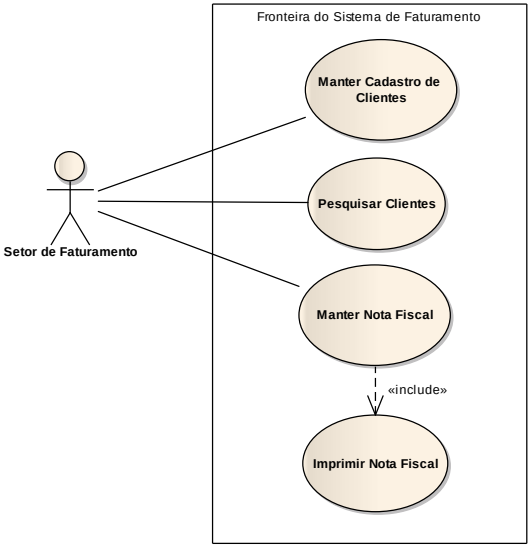

Figura 26 - Casos de uso do sistema de faturamento

Para melhor entendimento da aplicação e dos dados envolvidos no negócio, os analistas construíram uma primeira versão do diagrama de classes de domínio do negócio, que está ilustrado na Figura 27.

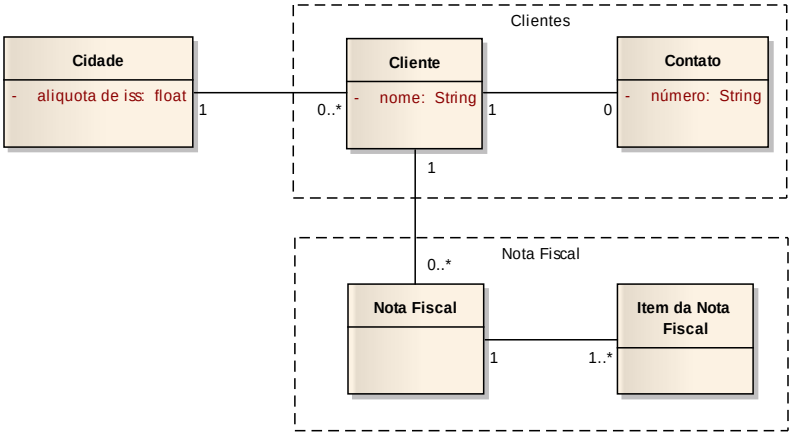

Figura 27 - Diagrama de classes conceituais do sistema de faturamento

Uma primeira descrição em alto nível dos casos de uso foi apresentada ao patrocinador, que então solicitou uma estimativa do projeto. A Figura 28 apresenta as descrições dos casos de uso.

Manter Cadastro de Clientes
Objetivo: permitir ao usuário (atores) manter o cadastro dos clientes da empresa.
Resumo: A atividade de manter pressupõe as funcionalidades de inclusão, alteração, exclusão e recuperação de clientes (CRUD). O cliente solicitaa validação do CEP do endereço do cliente e que, na medida em que o nome da cidade vai sendo digitado, o sistema mostre as opções disponíveis (autocomplete)

Manter Nota Fiscal
Objetivo: permitir ao usuário (atores) manter notas fiscais.
Resumo: O ator pode incluir, alterar ou excluir uma nota fiscal a qualquer momento. O registro de uma nota fiscal requer basicamente as seguintes informações:

- cliente
- serviços executados (itens da nota fiscal)
- impostos a deduzir

O cliente solicita que, quando da alteração ou exclusão de uma nota fiscal, seja enviado um email ao diretor informando a operação.

Imprimir Nota Fiscal
Objetivo: permitir ao usuário (atores) imprimir uma nota fiscal já cadastrada.
Resumo: depois de completar o cadastramento dos dados de uma nota fiscal, o usuário pode imprimí-la.

Pesquisar de Clientes
Objetivo: permitir ao usuário (atores) pesquisar seus clientes
Resumo: o usuário pode pesquisar seus clientes por vários critérios. Os principais identificados neste momento são:

- nome ou parte do nome,
- cidade
- UF

Figura 28 - Descrições de alto nível dos casos de uso do SISFAT

B.2 INSTANCIAÇÃO DO FUCS PARA O ESTUDO DE CASO

Conforme descrito na seção 4. 4, a medição de um sistema requer uma instanciação do processo de medição do FUCS. Nesta instanciação são definidos os tipos de caso de uso utilizados na empresa que

desenvolverá o sistema, juntamente com seus respectivos pesos. Além disso, as variáveis α e β, correspondentes aos graus de complexidade para desenvolvimento de novas operações de sistema e regras de negócio e requisitos de interface, também são definidas de acordo com os padrões, frameworks e ferramentas que compõem o processo de desenvolvimento de cada empresa.

Neste estudo de caso, a empresa vai desenvolver o projeto com a Ferramenta GENEXUS[22]. Considerando a produtividade de desenvolvimento dos casos de uso, optou-se por instanciar o método com os mesmos tipos de caso de uso e pesos definidos na Tabela 22. Às variáveis α e β foram atribuídos os valores 2 e 3, respectivamente, pois o GENEXUS apresenta uma série de facilidades, onde novas operações de sistema e regras de negócio podem ser realizadas com auxílio de ferramentas visuais. Já os requisitos de interface gráfica apresentam um pouco mais de dificuldade, e por este motivo foi atribuído o peso 3.

B.3 MEDIÇÃO E ESTIMATIVA DO SISTEMA DE FATURAMENTO

Para a medição do sistema é necessário mapear as informações do projeto para os conceitos e atributos utilizados no método FUCS. A seguir são descritos todos os mapeamentos dos casos de uso para os conceitos do FUCS.

Caso de uso:Manter Cadastro de Clientes			
Tipo de Caso de Uso:	CRUD	UC_{TP}:	2
Justificativa da utilização do tipo:	Neste caso de uso **um mesmo ator** poderá incluir, alterar, excluir e recuperar as informações dos clientes. Todas estas funcionalidades estão pré-definidas no tipo de caso de uso CRUD.		
Operações de Sistema e Regras de negócio:	RN – Validação do CEP na base dos correios	N_{OSNR}:	1
Requisitos e Interface Gráfica do Usuário	RI – *Autocomplete* de cidades	N_{GUIR}:	1

[22] http://www.genexus.com.br

Caso de uso:Manter Nota Fiscal			
Tipo de Caso de Uso:	CRUD Mestre/Detalhe	UC_{TP}:	4
Justificativa da utilização do tipo:	Neste caso de uso **um mesmo ator** poderá incluir, alterar, excluir e recuperar as informações das notas fiscais (mestre) juntamente com a relação de serviços executados (detalhe) e impostos a deduzir.		
Operações de Sistema e Regras de negócio:	RN – Cálculo da dedução de impostos de acordo com a alíquota da cidade RN – Envio de email para o diretor nos casos de alterações ou exclusão de notas fiscais	N_{OSNR}:	2
Requisitos e Interface Gráfica do Usuário	Não se aplica	N_{GUIR}:	0

Caso de uso: Imprimir Nota Fiscal			
Tipo de Caso de Uso:	Relatório	UC_{TP}	2
Justificativa da utilização do tipo:	Neste caso de uso, o ator imprime uma determinada nota fiscal. A nota fiscal inclui os dados do cliente, relação dos serviços e deduções dos impostos, e um total dos valores dos serviços.		
Operações de Sistema e Regras de negócio:	Não se aplica	N_{OSNR}:	0
Requisitos e Interface Gráfica do Usuário	Não se aplica	N_{GUIR}:	0

Caso de uso: Pesquisar Clientes			
Tipo de Caso de Uso:	Consulta Parametrizada	UC_{TP}:	1
Justificativa da utilização do tipo:	O ator pode pesquisar pelos critérios nome, cidade e UF.		
Operações de Sistema e Regras de negócio:	Não se aplica	N_{OSNR}:	0
Requisitos e Interface Gráfica do Usuário	Não se aplica	N_{GUIR}:	0

A Tabela 30 apresenta um resumo dos atributos definidos para cada caso de uso, bem como a medição do tamanho de cada um (UC_S) através da fórmula definida na equação 38, considerando as variáveis α e β com os valores 2 e 3, respectivamente, conforme definido na instanciação do processo.

Tabela 30 - Tamanhos dos casos de uso do sistema de faturamento

Caso de Uso	**Tipo do Caso de Uso**	UC_{TP}	N_{OSNR}	N_{GUIR}	UC_S
Manter Cadastro de Clientes	CRUD	2	1	1	7
Manter Nota Fiscal	CRUD Mestre/Detalhe	4	2		8
Imprimir Nota Fiscal	Relatório	2			2
Pesquisar Clientes	Consulta Parametrizada	1			1

O tamanho total do sistema obtido através da equação 39 totaliza 18 UCS (*use case size*). Os resultados descritos na seção 5.4, demonstram que em média, o esforço para desenvolvimento de 1 UCS pode ser estimado como sendo 7 horas. Assim, o esforço para desenvolvimento do sistema de faturamento deste estudo de caso poderia ser estimado em 126 horas.

ANEXO C – Lista de Casos de Uso dos Estudos de Casos

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. A	UC137	Serviço	5	0	1	7	48	6,9	49	1
PROJ. A	UC138	Serviço	5	0	1	7	60	8,6	49	-11
PROJ. A	UC139	Serviço	5	0	1	7	56	8,0	49	-7
PROJ. A	UC140	Serviço	5	0	1	7	44	6,3	49	5
PROJ. A	UC141	Serviço	5	0	1	7	44	6,3	49	5
PROJ. A	UC142	Consulta Parametrizada	1	0	0	1	15	15,0	7	-8
PROJ. A	UC143	Consulta Parametrizada	1	0	0	1	22	22,0	7	-15
PROJ. A	UC144	Consulta Parametrizada	1	0	0	1	17	17,0	7	-10
PROJ. A	UC145	Consulta Parametrizada	1	0	0	1	13	13,0	7	-6
PROJ. A	UC146	Consulta Parametrizada	1	0	1	3	25	8,3	21	-4
PROJ. A	UC147	Consulta Parametrizada	1	4	0	9	64	7,1	63	-1
PROJ. A	UC148	Consulta Parametrizada	1	0	0	1	6	6,0	7	1
PROJ. A	UC149	CRUD Mestre/Detalhe	4	3	5	20	143	7,2	140	-3
PROJ. A	UC150	CRUD Detalhe	3	0	1	5	26	5,2	35	9
PROJ. A	UC151	CRUD Detalhe	3	0	0	3	26	8,7	21	-5
PROJ. A	UC152	Consulta Parametrizada	1	1	0	3	36	12,0	21	-15
PROJ. A	UC153	CRUD Mestre/Detalhe	4	1	3	12	95	7,9	84	-11

ANEXO C – Lista de Casos de Uso dos Estudos de Casos (continuação)

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. A	UC154	Caso de Uso Não Padronizado	5	0	0	5	40	8,0	35	-5
PROJ. A	UC155	CRUD Mestre/Detalhe	4	1	1	8	60	7,5	56	-4
PROJ. A	UC156	Caso de Uso Não Padronizado	5	4	0	13	83	6,4	91	8
PROJ. A	UC157	Consulta Parametrizada	1	1	0	3	18	6,0	21	3
PROJ. A	UC158	CRUD Detalhe	3	3	1	11	77	7,0	77	0
PROJ. A	UC159	Serviço	5	1	0	7	66	9,4	49	-17
PROJ. A	UC160	Consulta Parametrizada	1	5	0	11	62	5,6	77	15
PROJ. A	UC161	CRUD Tabular	2	0	0	2	17	8,5	14	-3
PROJ. A	UC162	CRUD	2	2	0	6	27	4,5	42	15
PROJ. A	UC163	CRUD	2	0	0	2	32	16,0	14	-18
PROJ. A	UC164	CRUD Tabular	2	1	0	4	24	6,0	28	4
PROJ. A	UC165	Caso de Uso Não Padronizado	5	1	0	7	58	8,3	49	-9
PROJ. A	UC166	CRUD Mestre/Detalhe	4	3	0	10	60	6,0	70	10
PROJ. A	UC167	Consulta Parametrizada	1	5	0	11	61	5,5	77	16
PROJ. A	UC168	Consulta Parametrizada	1	1	0	3	16	5,3	21	5
PROJ. B	UC169	CRUD	2	4	4	18	149	8,3	126	-23
PROJ. B	UC170	Consulta Parametrizada	1	1	6	15	97	6,5	105	8
PROJ. B	UC171	CRUD	2	9	4	28	128	4,6	196	68

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. B	UC172	CRUD	2	2	4	14	72	5,1	98	26
PROJ. B	UC173	Relatório	3	1	4	13	104	8,0	91	-13
PROJ. B	UC174	Consulta Parametrizada	1	1	6	15	87	5,8	105	18
PROJ. B	UC175	Consulta Parametrizada	1	1	4	11	49	4,5	77	28
PROJ. B	UC176	Consulta Parametrizada	1	1	4	11	84	7,6	77	-7
PROJ. B	UC177	Consulta Parametrizada	1	1	4	11	62	5,6	77	15
PROJ. B	UC178	Consulta Parametrizada	1	0	0	1	9	9,0	7	-2
PROJ. B	UC179	Consulta Parametrizada	1	0	1	3	24	8,0	21	-3
PROJ. B	UC180	Caso de Uso Não Padronizado	5	4	1	15	88	5,9	105	17
PROJ. B	UC181	Caso de Uso Não Padronizado	5	0	2	9	61	6,8	63	2
PROJ. B	UC182	Consulta Parametrizada	1	3	0	7	42	6,0	49	7
PROJ. B	UC183	CRUD	2	0	0	2	16	8,0	14	-2
PROJ. B	UC184	Consulta Parametrizada	1	5	3	17	165	9,7	119	-46
PROJ. B	UC185	Consulta Parametrizada	1	0	0	1	6	6,0	7	1
PROJ. C	UC186	Caso de Uso Não Padronizado	5	4	5	23	165	7,2	161	-4
PROJ. C	UC187	Consulta Parametrizada	1	1	0	3	31	10,3	21	-10
PROJ. C	UC188	Caso de Uso Não Padronizado	5	5	4	23	155	6,7	161	6

* Y é o esforço de desenvolvimento estimado com base no tamanho dos casos de uso (UC_{TP}) utilizando um valor de referência de produtividade de 7 horas por UCS.

** E é o valor do Erro da estimativa calculado com base na diferença entre o valor estimado (Y) e o esforço real de desenvolvimento do caso de uso.

ANEXO C – Lista de Casos de Uso dos Estudos de Casos (continuação)

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. C	UC189	Consulta Parametrizada	1	1	0	3	32	10,7	21	-11
PROJ. C	UC190	Consulta Parametrizada	1	2	1	7	35	5,0	49	14
PROJ. C	UC191	Caso de Uso Não Padronizado	5	10	2	29	163	5,6	203	40
PROJ. C	UC192	Caso de Uso Não Padronizado	5	2	1	11	113	10,3	77	-36
PROJ. C	UC193	Caso de Uso Não Padronizado	5	3	5	21	159	7,6	147	-12
PROJ. C	UC194	Caso de Uso Não Padronizado	5	8	3	27	214	7,9	189	-25
PROJ. C	UC195	Caso de Uso Não Padronizado	5	5	5	25	156	6,2	175	19
PROJ. C	UC196	Caso de Uso Não Padronizado	5	5	6	27	186	6,9	189	3
PROJ. C	UC197	Consulta Parametrizada	1	8	5	27	237	8,8	189	-48
PROJ. C	UC198	Caso de Uso Não Padronizado	5	13	7	45	283	6,3	315	32
PROJ. C	UC199	Caso de Uso Não Padronizado	5	3	2	15	76	5,1	105	29
PROJ. C	UC200	Caso de Uso Não Padronizado	5	2	1	11	113	10,3	77	-36
PROJ. C	UC201	Caso de Uso Não Padronizado	5	4	6	25	50	2,0	175	125
PROJ. C	UC202	Consulta Parametrizada	1	4	2	13	64	4,9	91	27
PROJ. D	UC203	Caso de Uso Não Padronizado	5	0	14	33	123	3,7	231	108
PROJ. D	UC204	CRUD Mestre/Detalhe	4	0	2	8	48	6,0	56	8
PROJ. D	UC205	CRUD Tabular	2	1	2	8	49	6,1	56	7
PROJ. D	UC206	Consulta Parametrizada	1	0	1	3	26	8,7	21	-5

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. D	UC207	Consulta Parametrizada	1	1	0	3	5	1,7	21	16
PROJ. D	UC208	Consulta Parametrizada	1	1	0	3	40	13,3	21	-19
PROJ. D	UC209	Caso de Uso Não Padronizado	5	0	1	7	89	12,7	49	-40
PROJ. D	UC210	CRUD	2	0	1	4	19	4,8	28	9
PROJ. D	UC211	Consulta Parametrizada	1	0	1	3	23	7,7	21	-2
PROJ. D	UC212	CRUD Mestre/Detalhe	4	1	2	10	62	6,2	70	8
PROJ. D	UC213	CRUD	2	3	1	10	82	8,2	70	-12
PROJ. D	UC214	CRUD Tabular	2	1	4	12	73	6,1	84	11
PROJ. D	UC215	CRUD	2	1	3	10	87	8,7	70	-17
PROJ. D	UC216	Consulta Parametrizada	1	0	0	1	17	17,0	7	-10
PROJ. D	UC217	CRUD Tabular	2	1	3	10	113	11,3	70	-43
PROJ. D	UC218	Consulta Parametrizada	1	1	1	5	18	3,6	35	17
PROJ. D	UC219	CRUD	2	2	2	10	60	6,0	70	10
PROJ. D	UC220	Caso de Uso Não Padronizado	5	1	1	9	62	6,9	63	1
PROJ. E	UC1	Caso de Uso Não Padronizado	5	2	2	13	88	6,8	91	3
PROJ. E	UC10	Consulta Parametrizada	1	0	1	3	11	3,7	21	10
PROJ. E	UC100	Consulta Parametrizada	1	1	0	3	23	7,7	21	-2

* Y é o esforço de desenvolvimento estimado com base no tamanho dos casos de uso (UC_{TP}) utilizando um valor de referência de produtividade de 7 horas por UCS.

** E é o valor do Erro da estimativa calculado com base na diferença entre o valor estimado (Y) e o esforço real de desenvolvimento do caso de uso.

ANEXO C – Lista de Casos de Uso dos Estudos de Casos (continuação)

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. E	UC101	Consulta Parametrizada	1	1	0	3	31	10,3	21	-10
PROJ. E	UC102	Consulta Parametrizada	1	1	0	3	19	6,3	21	2
PROJ. E	UC103	Consulta Parametrizada	1	1	0	3	19	6,3	21	2
PROJ. E	UC104	Consulta Parametrizada	1	1	0	3	18	6,0	21	3
PROJ. E	UC105	Consulta Parametrizada	1	1	0	3	28	9,3	21	-7
PROJ. E	UC106	Consulta Parametrizada	1	1	1	5	18	3,6	35	17
PROJ. E	UC107	Consulta Parametrizada	1	1	2	7	58	8,3	49	-9
PROJ. E	UC108	Caso de Uso Não Padronizado	5	1	2	11	42	3,8	77	35
PROJ. E	UC109	Caso de Uso Não Padronizado	5	1	1	9	61	6,8	63	2
PROJ. E	UC11	Caso de Uso Não Padronizado	5	0	0	5	15	3,0	35	20
PROJ. E	UC110	Consulta Parametrizada	1	4	1	11	109	9,9	77	-32
PROJ. E	UC111	Consulta Parametrizada	1	3	1	9	40	4,4	63	23
PROJ. E	UC112	Serviço	5	0	0	5	40	8,0	35	-5
PROJ. E	UC113	Serviço	5	0	0	5	44	8,8	35	-9
PROJ. E	UC114	Consulta Parametrizada	1	7	0	15	80	5,3	105	25
PROJ. E	UC115	CRUD	2	1	3	10	36	3,6	70	34
PROJ. E	UC116	CRUD	2	0	2	6	41	6,8	42	1
PROJ. E	UC117	CRUD	2	0	3	8	47	5,9	56	9

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. E	UC118	CRUD	2	0	0	2	19	9,5	14	-5
PROJ. E	UC119	Caso de Uso Não Padronizado	5	0	0	5	38	7,6	35	-3
PROJ. E	UC12	Caso de Uso Não Padronizado	5	3	0	11	62	5,6	77	15
PROJ. E	UC120	CRUD	2	0	0	2	21	10,5	14	-7
PROJ. E	UC13	Caso de Uso Não Padronizado	5	1	2	11	76	6,9	77	1
PROJ. E	UC14	Consulta Parametrizada	1	2	0	5	32	6,4	35	3
PROJ. E	UC15	Consulta Parametrizada	1	1	0	3	10	3,3	21	11
PROJ. E	UC16	Consulta Parametrizada	1	2	2	9	57	6,3	63	6
PROJ. E	UC17	Consulta Parametrizada	1	2	1	7	53	7,6	49	-4
PROJ. E	UC18	Consulta Parametrizada	1	1	0	3	10	3,3	21	11
PROJ. E	UC19	Consulta Parametrizada	1	1	2	7	38	5,4	49	11
PROJ. E	UC2	Caso de Uso Não Padronizado	5	0	4	13	65	5,0	91	26
PROJ. E	UC20	Consulta Parametrizada	1	1	8	19	102	5,4	133	31
PROJ. E	UC21	Consulta Parametrizada	1	4	3	15	74	4,9	105	31
PROJ. E	UC22	Consulta Parametrizada	1	2	1	7	45	6,4	49	4
PROJ. E	UC23	CRUD	2	1	4	12	57	4,8	84	27
PROJ. E	UC24	CRUD	2	1	4	12	32	2,7	84	52

* Y é o esforço de desenvolvimento estimado com base no tamanho dos casos de uso (UC_{TP}) utilizando um valor de referência de produtividade de 7 horas por UCS.

** E é o valor do Erro da estimativa calculado com base na diferença entre o valor estimado (Y) e o esforço real de desenvolvimento do caso de uso.

ANEXO C – Lista de Casos de Uso dos Estudos de Casos (continuação)

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. E	UC25	Consulta Parametrizada	1	3	3	13	83	6,4	91	8
PROJ. E	UC26	Consulta Parametrizada	1	0	0	1	20	20,0	7	-13
PROJ. E	UC27	Consulta Parametrizada	1	0	0	1	7	7,0	7	0
PROJ. E	UC28	Consulta Parametrizada	1	0	0	1	7	7,0	7	0
PROJ. E	UC29	Consulta Parametrizada	1	0	0	1	8	8,0	7	-1
PROJ. E	UC3	Caso de Uso Não Padronizado	5	1	2	11	48	4,4	77	29
PROJ. E	UC30	Consulta Parametrizada	1	0	0	1	7	7,0	7	0
PROJ. E	UC31	Consulta Parametrizada	1	0	0	1	7	7,0	7	0
PROJ. E	UC32	Consulta Parametrizada	1	0	0	1	8	8,0	7	-1
PROJ. E	UC33	Consulta Parametrizada	1	0	0	1	14	14,0	7	-7
PROJ. E	UC34	Consulta Parametrizada	1	0	3	7	44	6,3	49	5
PROJ. E	UC35	Consulta Parametrizada	1	1	2	7	59	8,4	49	-10
PROJ. E	UC36	Consulta Parametrizada	1	0	0	1	16	16,0	7	-9
PROJ. E	UC37	Consulta Parametrizada	1	0	2	5	17	3,4	35	18
PROJ. E	UC38	CRUD Mestre/Detalhe	4	2	2	12	72	6,0	84	12
PROJ. E	UC39	CRUD Mestre/Detalhe	4	6	9	34	185	5,4	238	53
PROJ. E	UC4	Caso de Uso Não Padronizado	5	0	2	9	72	8,0	63	-9
PROJ. E	UC40	CRUD	2	0	3	8	46	5,8	56	10

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. E	UC41	CRUD	2	0	1	4	38	9,5	28	-10
PROJ. E	UC42	CRUD	2	0	2	6	33	5,5	42	9
PROJ. E	UC43	CRUD	2	1	1	6	28	4,7	42	14
PROJ. E	UC44	CRUD	2	0	1	4	27	6,8	28	1
PROJ. E	UC45	CRUD	2	0	1	4	22	5,5	28	6
PROJ. E	UC46	CRUD	2	2	3	12	42	3,5	84	42
PROJ. E	UC47	CRUD	2	1	2	8	32	4,0	56	24
PROJ. E	UC48	CRUD	2	0	1	4	17	4,3	28	11
PROJ. E	UC49	CRUD	2	1	7	18	96	5,3	126	30
PROJ. E	UC5	Caso de Uso Não Padronizado	5	1	9	25	149	6,0	175	26
PROJ. E	UC50	Consulta Parametrizada	1	1	0	3	23	7,7	21	-2
PROJ. E	UC51	Consulta Parametrizada	1	1	0	3	20	6,7	21	1
PROJ. E	UC52	CRUD	2	0	1	4	3	0,8	28	25
PROJ. E	UC53	CRUD	2	0	1	4	25	6,3	28	3
PROJ. E	UC54	CRUD	2	0	1	4	21	5,3	28	7
PROJ. E	UC55	CRUD	2	0	1	4	28	7,0	28	0
PROJ. E	UC56	Consulta Parametrizada	1	0	1	3	17	5,7	21	4

* Y é o esforço de desenvolvimento estimado com base no tamanho dos casos de uso (UC_{TP}) utilizando um valor de referência de produtividade de 7 horas por UCS.

** E é o valor do Erro da estimativa calculado com base na diferença entre o valor estimado (Y) e o esforço real de desenvolvimento do caso de uso.

ANEXO C – Lista de Casos de Uso dos Estudos de Casos (continuação)

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. E	UC57	Consulta Parametrizada	1	0	1	3	30	10,0	21	-9
PROJ. E	UC58	Consulta Parametrizada	1	1	2	7	31	4,4	49	18
PROJ. E	UC59	Caso de Uso Não Padronizado	5	1	0	7	61	8,7	49	-12
PROJ. E	UC6	Consulta Parametrizada	1	0	1	3	38	12,7	21	-17
PROJ. E	UC60	Caso de Uso Não Padronizado	5	1	2	11	22	2,0	77	55
PROJ. E	UC61	Serviço	5	0	2	9	72	8,0	63	-9
PROJ. E	UC62	Relatório	3	1	0	5	72	14,4	35	-37
PROJ. E	UC63	CRUD	2	2	2	10	47	4,7	70	23
PROJ. E	UC64	Consulta Parametrizada	1	0	1	3	24	8,0	21	-3
PROJ. E	UC65	Consulta Parametrizada	1	0	0	1	17	17,0	7	-10
PROJ. E	UC66	Consulta Parametrizada	1	0	0	1	16	16,0	7	-9
PROJ. E	UC67	CRUD Detalhe	3	0	0	3	28	9,3	21	-7
PROJ. E	UC68	CRUD Detalhe	3	0	2	7	62	8,9	49	-13
PROJ. E	UC69	Relatório	3	0	2	7	46	6,6	49	3
PROJ. E	UC7	Caso de Uso Não Padronizado	5	1	1	9	63	7,0	63	0
PROJ. E	UC70	Consulta Parametrizada	1	0	1	3	23	7,7	21	-2
PROJ. E	UC71	Consulta Parametrizada	1	0	1	3	29	9,7	21	-8
PROJ. E	UC72	Consulta Parametrizada	1	0	3	7	30	4,3	49	19

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. E	UC73	CRUD	2	0	2	6	42	7,0	42	0
PROJ. E	UC74	CRUD Mestre/Detalhe	4	1	4	14	86	6,1	98	12
PROJ. E	UC75	Consulta Parametrizada	1	1	2	7	59	8,4	49	-10
PROJ. E	UC76	CRUD	2	0	4	10	57	5,7	70	13
PROJ. E	UC77	Consulta Parametrizada	1	2	2	9	53	5,9	63	10
PROJ. E	UC78	Consulta Parametrizada	1	1	0	3	22	7,3	21	-1
PROJ. E	UC79	CRUD	2	3	1	10	85	8,5	70	-15
PROJ. E	UC8	Caso de Uso Não Padronizado	5	0	0	5	41	8,2	35	-6
PROJ. E	UC80	Consulta Parametrizada	1	0	1	3	27	9,0	21	-6
PROJ. E	UC81	Consulta Parametrizada	1	0	5	11	37	3,4	77	40
PROJ. E	UC82	CRUD	2	1	0	4	13	3,3	28	15
PROJ. E	UC83	Relatório	3	0	1	5	7	1,4	35	28
PROJ. E	UC84	Consulta Parametrizada	1	0	1	3	5	1,7	21	16
PROJ. E	UC85	Consulta Parametrizada	1	0	1	3	3	1,0	21	18
PROJ. E	UC86	CRUD	2	1	4	12	79	6,6	84	5
PROJ. E	UC87	CRUD	2	0	1	4	12	3,0	28	16
PROJ. E	UC88	CRUD	2	0	1	4	16	4,0	28	12

* Y é o esforço de desenvolvimento estimado com base no tamanho dos casos de uso (UC_{TP}) utilizando um valor de referência de produtividade de 7 horas por UCS.

** E é o valor do Erro da estimativa calculado com base na diferença entre o valor estimado (Y) e o esforço real de desenvolvimento do caso de uso.

ANEXO C – Lista de Casos de Uso dos Estudos de Casos (continuação)

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. E	UC89	Consulta Parametrizada	1	0	0	1	17	17,0	7	-10
PROJ. E	UC9	Caso de Uso Não Padronizado	5	0	6	17	114	6,7	119	5
PROJ. E	UC90	Consulta Parametrizada	1	0	1	3	23	7,7	21	-2
PROJ. E	UC91	Consulta Parametrizada	1	0	1	3	25	8,3	21	-4
PROJ. E	UC92	Consulta Parametrizada	1	0	1	3	36	12,0	21	-15
PROJ. E	UC93	Consulta Parametrizada	1	0	1	3	24	8,0	21	-3
PROJ. E	UC94	Consulta Parametrizada	1	0	1	3	30	10,0	21	-9
PROJ. E	UC95	Consulta Parametrizada	1	0	1	3	25	8,3	21	-4
PROJ. E	UC96	Consulta Parametrizada	1	0	1	3	29	9,7	21	-8
PROJ. E	UC97	Consulta Parametrizada	1	0	1	3	21	7,0	21	0
PROJ. E	UC98	CRUD	2	0	1	4	25	6,3	28	3
PROJ. E	UC99	Consulta Parametrizada	1	1	0	3	31	10,3	21	-10
PROJ. F	UC121	Caso de Uso Não Padronizado	5	11	6	39	200	5,1	273	73
PROJ. F	UC122	Caso de Uso Não Padronizado	5	2	2	13	77	5,9	91	14
PROJ. F	UC123	Consulta Parametrizada	1	6	2	17	80	4,7	119	39
PROJ. F	UC124	Consulta Parametrizada	1	6	2	17	68	4,0	119	51
PROJ. F	UC125	Caso de Uso Não Padronizado	5	1	1	9	80	8,9	63	-17
PROJ. F	UC126	Relatório	3	2	0	7	36	5,1	49	13

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. F	UC127	Caso de Uso Não Padronizado	5	5	4	23	175	7,6	161	-14
PROJ. F	UC128	Relatório	3	4	6	23	214	9,3	161	-53
PROJ. F	UC129	Caso de Uso Não Padronizado	5	4	3	19	121	6,4	133	12
PROJ. F	UC130	Consulta Parametrizada	1	4	4	17	112	6,6	119	7
PROJ. F	UC131	CRUD	2	4	0	10	92	9,2	70	-22
PROJ. F	UC132	CRUD	2	8	2	22	142	6,5	154	12
PROJ. F	UC133	CRUD	2	5	0	12	101	8,4	84	-17
PROJ. F	UC134	Consulta Parametrizada	1	8	4	25	70	2,8	175	105
PROJ. F	UC135	Caso de Uso Não Padronizado	5	3	3	17	132	7,8	119	-13
PROJ. F	UC136	Serviço	5	0	1	7	72	10,3	49	-23
PROJ. G	UC221	Caso de Uso Não Padronizado	5	2	1	11	104	9,5	77	-27
PROJ. G	UC222	Caso de Uso Não Padronizado	5	0	4	13	87	6,7	91	4
PROJ. G	UC223	Caso de Uso Não Padronizado	5	1	2	11	84	7,6	77	-7
PROJ. G	UC224	Caso de Uso Não Padronizado	5	0	0	5	48	9,6	35	-13
PROJ. G	UC225	Caso de Uso Não Padronizado	5	1	7	21	158	7,5	147	-11
PROJ. G	UC226	Consulta Parametrizada	1	0	1	3	29	9,7	21	-8
PROJ. G	UC227	Caso de Uso Não Padronizado	5	1	1	9	68	7,6	63	-5

* Y é o esforço de desenvolvimento estimado com base no tamanho dos casos de uso (UC_{TP}) utilizando um valor de referência de produtividade de 7 horas por UCS.

** E é o valor do Erro da estimativa calculado com base na diferença entre o valor estimado (Y) e o esforço real de desenvolvimento do caso de uso.

ANEXO C – Lista de Casos de Uso dos Estudos de Casos (continuação)

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. G	UC228	Caso de Uso Não Padronizado	5	0	0	5	50	10,0	35	-15
PROJ. G	UC229	Caso de Uso Não Padronizado	5	0	3	11	105	9,5	77	-28
PROJ. G	UC230	Consulta Parametrizada	1	0	0	1	6	6,0	7	1
PROJ. G	UC231	Caso de Uso Não Padronizado	5	0	0	5	29	5,8	35	6
PROJ. G	UC232	Caso de Uso Não Padronizado	5	3	0	11	89	8,1	77	-12
PROJ. G	UC233	Consulta Parametrizada	1	0	0	1	9	9,0	7	-2
PROJ. G	UC234	Consulta Parametrizada	1	0	3	7	51	7,3	49	-2
PROJ. G	UC235	CRUD Mestre/Detalhe	4	0	1	6	44	7,3	42	-2
PROJ. G	UC236	Caso de Uso Não Padronizado	5	1	0	7	68	9,7	49	-19
PROJ. G	UC237	Consulta Parametrizada	1	2	1	7	47	6,7	49	2
PROJ. G	UC238	Consulta Parametrizada	1	1	0	3	23	7,7	21	-2
PROJ. G	UC239	Consulta Parametrizada	1	2	1	7	43	6,1	49	6
PROJ. G	UC240	Consulta Parametrizada	1	2	1	7	61	8,7	49	-12
PROJ. G	UC241	Consulta Parametrizada	1	1	0	3	21	7,0	21	0
PROJ. G	UC242	Consulta Parametrizada	1	1	2	7	64	9,1	49	-15
PROJ. G	UC243	Consulta Parametrizada	1	1	3	9	72	8,0	63	-9
PROJ. G	UC244	Consulta Parametrizada	1	3	2	11	99	9,0	77	-22
PROJ. G	UC245	CRUD	2	1	4	12	91	7,6	84	-7

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. G	UC246	CRUD	2	1	4	12	86	7,2	84	-2
PROJ. G	UC247	Consulta Parametrizada	1	3	1	9	81	9,0	63	-18
PROJ. G	UC248	Consulta Parametrizada	1	0	0	1	8	8,0	7	-1
PROJ. G	UC249	Consulta Parametrizada	1	0	0	1	9	9,0	7	-2
PROJ. G	UC250	Consulta Parametrizada	1	0	0	1	9	9,0	7	-2
PROJ. G	UC251	Consulta Parametrizada	1	0	0	1	8	8,0	7	-1
PROJ. G	UC252	Consulta Parametrizada	1	0	0	1	8	8,0	7	-1
PROJ. G	UC253	Consulta Parametrizada	1	0	0	1	7	7,0	7	0
PROJ. G	UC254	Consulta Parametrizada	1	0	0	1	7	7,0	7	0
PROJ. G	UC255	Consulta Parametrizada	1	0	0	1	11	11,0	7	-4
PROJ. G	UC256	Consulta Parametrizada	1	0	0	1	9	9,0	7	-2
PROJ. G	UC257	Consulta Parametrizada	1	1	2	7	80	11,4	49	-31
PROJ. G	UC258	Consulta Parametrizada	1	0	0	1	10	10,0	7	-3
PROJ. G	UC259	Consulta Parametrizada	1	0	3	7	55	7,9	49	-6
PROJ. G	UC260	CRUD Mestre/Detalhe	4	2	3	14	96	6,9	98	2
PROJ. G	UC261	CRUD	2	6	5	24	196	8,2	168	-28
PROJ. G	UC262	CRUD	2	1	1	6	54	9,0	42	-12

* Y é o esforço de desenvolvimento estimado com base no tamanho dos casos de uso (UC_{TP}) utilizando um valor de referência de produtividade de 7 horas por UCS.

** E é o valor do Erro da estimativa calculado com base na diferença entre o valor estimado (Y) e o esforço real de desenvolvimento do caso de uso.

ANEXO C – Lista de Casos de Uso dos Estudos de Casos (continuação)

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. G	UC263	CRUD	2	0	0	2	12	6,0	14	2
PROJ. G	UC264	CRUD	2	0	1	4	37	9,3	28	-9
PROJ. G	UC265	CRUD	2	2	2	10	74	7,4	70	-4
PROJ. G	UC266	CRUD	2	1	0	4	24	6,0	28	4
PROJ. G	UC267	CRUD	2	0	1	4	30	7,5	28	-2
PROJ. G	UC268	CRUD	2	1	3	10	62	6,2	70	8
PROJ. G	UC269	CRUD	2	0	1	4	32	8,0	28	-4
PROJ. G	UC270	CRUD	2	0	1	4	36	9,0	28	-8
PROJ. G	UC271	CRUD	2	0	1	4	21	5,3	28	7
PROJ. G	UC272	CRUD	2	0	1	4	35	8,8	28	-7
PROJ. G	UC273	Consulta Parametrizada	1	0	1	3	24	8,0	21	-3
PROJ. G	UC274	Consulta Parametrizada	1	0	1	3	24	8,0	21	-3
PROJ. G	UC275	Consulta Parametrizada	1	1	1	5	41	8,2	35	-6
PROJ. G	UC276	Caso de Uso Não Padronizado	5	1	0	7	70	10,0	49	-21
PROJ. G	UC277	Caso de Uso Não Padronizado	5	1	0	7	63	9,0	49	-14
PROJ. G	UC278	Serviço	5	0	0	5	19	3,8	35	16
PROJ. G	UC279	Relatório	3	1	0	5	31	6,2	35	4
PROJ. G	UC280	CRUD	2	2	0	6	37	6,2	42	5

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. G	UC281	Consulta Parametrizada	1	2	0	5	40	8,0	35	-5
PROJ. G	UC282	CRUD Detalhe	3	3	4	17	155	9,1	119	-36
PROJ. G	UC283	Caso de Uso Não Padronizado	5	2	1	11	91	8,3	77	-14
PROJ. G	UC284	CRUD Tabular	2	4	4	18	138	7,7	126	-12
PROJ. G	UC285	Consulta Parametrizada	1	2	1	7	61	8,7	49	-12
PROJ. G	UC286	CRUD Detalhe	3	5	3	19	155	8,2	133	-22
PROJ. G	UC287	Consulta Parametrizada	1	0	0	1	10	10,0	7	-3
PROJ. G	UC288	Consulta Parametrizada	1	0	0	1	7	7,0	7	0
PROJ. G	UC289	Consulta Parametrizada	1	2	0	5	35	7,0	35	0
PROJ. G	UC290	CRUD Mestre/Detalhe	4	3	4	18	137	7,6	126	-11
PROJ. G	UC291	Consulta Parametrizada	1	0	0	1	4	4,0	7	3
PROJ. G	UC292	CRUD Mestre/Detalhe	4	0	3	10	64	6,4	70	6
PROJ. G	UC293	Consulta Parametrizada	1	0	0	1	7	7,0	7	0
PROJ. G	UC294	CRUD Tabular	2	4	3	16	103	6,4	112	9
PROJ. G	UC295	Consulta Parametrizada	1	0	0	1	9	9,0	7	-2
PROJ. G	UC296	Relatório	3	0	0	3	7	2,3	21	14
PROJ. G	UC297	Consulta Parametrizada	1	0	0	1	9	9,0	7	-2

* Y é o esforço de desenvolvimento estimado com base no tamanho dos casos de uso (UC_{TP}) utilizando um valor de referência de produtividade de 7 horas por UCS.

** E é o valor do Erro da estimativa calculado com base na diferença entre o valor estimado (Y) e o esforço real de desenvolvimento do caso de uso.

ANEXO C – Lista de Casos de Uso dos Estudos de Casos (continuação)

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. G	UC298	Consulta Parametrizada	1	0	1	3	28	9,3	21	-7
PROJ. G	UC299	CRUD Mestre/Detalhe	4	4	4	20	160	8,0	140	-20
PROJ. G	UC300	CRUD	2	3	3	14	68	4,9	98	30
PROJ. G	UC301	CRUD Mestre/Detalhe	4	0	0	4	39	9,8	28	-11
PROJ. G	UC302	Consulta Parametrizada	1	0	0	1	9	9,0	7	-2
PROJ. G	UC303	CRUD	2	5	3	18	140	7,8	126	-14
PROJ. G	UC304	CRUD Tabular	2	0	1	4	35	8,8	28	-7
PROJ. G	UC305	CRUD Mestre/Detalhe	4	1	2	10	75	7,5	70	-5
PROJ. G	UC306	Relatório	3	0	0	3	6	2,0	21	15
PROJ. G	UC307	Consulta Parametrizada	1	0	0	1	8	8,0	7	-1
PROJ. G	UC308	Relatório	3	0	0	3	9	3,0	21	12
PROJ. G	UC309	Consulta Parametrizada	1	1	0	3	23	7,7	21	-2
PROJ. G	UC310	Relatório	3	0	0	3	8	2,7	21	13
PROJ. G	UC311	Serviço	5	1	2	11	49	4,5	77	28
PROJ. G	UC312	CRUD	2	2	3	12	108	9,0	84	-24
PROJ. G	UC313	Caso de Uso Não Padronizado	5	4	5	23	160	7,0	161	1
PROJ. G	UC314	Consulta Parametrizada	1	1	0	3	20	6,7	21	1
PROJ. G	UC315	Caso de Uso Não Padronizado	5	5	4	23	223	9,7	161	-62

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. G	UC316	Consulta Parametrizada	1	1	0	3	21	7,0	21	0
PROJ. G	UC317	Consulta Parametrizada	1	2	1	7	63	9,0	49	-14
PROJ. G	UC318	Caso de Uso Não Padronizado	5	0	0	5	45	9,0	35	-10
PROJ. G	UC319	Caso de Uso Não Padronizado	5	0	0	5	44	8,8	35	-9
PROJ. G	UC320	Caso de Uso Não Padronizado	5	0	0	5	46	9,2	35	-11
PROJ. G	UC321	Relatório	3	0	0	3	7	2,3	21	14
PROJ. G	UC322	CRUD	2	0	0	2	12	6,0	14	2
PROJ. G	UC323	Caso de Uso Não Padronizado	5	0	0	5	36	7,2	35	-1
PROJ. G	UC324	CRUD	2	0	0	2	10	5,0	14	4
PROJ. G	UC325	Consulta Parametrizada	1	0	0	1	9	9,0	7	-2
PROJ. G	UC326	CRUD	2	0	0	2	18	9,0	14	-4
PROJ. G	UC327	Relatório	3	0	0	3	7	2,3	21	14
PROJ. G	UC328	Serviço	5	0	0	5	26	5,2	35	9
PROJ. G	UC329	Serviço	5	0	1	7	42	6,0	49	7
PROJ. G	UC330	Serviço	5	0	0	5	29	5,8	35	6
PROJ. G	UC331	Serviço	5	0	0	5	27	5,4	35	8
PROJ. G	UC332	Relatório	3	0	0	3	6	2,0	21	15

* Y é o esforço de desenvolvimento estimado com base no tamanho dos casos de uso (UC_{TP}) utilizando um valor de referência de produtividade de 7 horas por UCS.

** E é o valor do Erro da estimativa calculado com base na diferença entre o valor estimado (Y) e o esforço real de desenvolvimento do caso de uso.

ANEXO C – Lista de Casos de Uso dos Estudos de Casos (continuação)

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. G	UC333	Serviço	5	0	0	5	20	4,0	35	15
PROJ. G	UC334	Serviço	5	0	0	5	25	5,0	35	10
PROJ. G	UC335	Consulta Parametrizada	1	0	0	1	6	6,0	7	1
PROJ. G	UC336	Consulta Parametrizada	1	1	0	3	26	8,7	21	-5
PROJ. G	UC337	Consulta Parametrizada	1	0	0	1	8	8,0	7	-1
PROJ. G	UC338	Consulta Parametrizada	1	0	0	1	7	7,0	7	0
PROJ. G	UC339	Consulta Parametrizada	1	0	1	3	21	7,0	21	0
PROJ. G	UC340	Consulta Parametrizada	1	5	0	11	91	8,3	77	-14
PROJ. G	UC341	Consulta Parametrizada	1	2	0	5	34	6,8	35	1
PROJ. G	UC342	CRUD Mestre/Detalhe	4	2	7	22	98	4,5	154	56
PROJ. G	UC343	Caso de Uso Não Padronizado	5	0	5	15	120	8,0	105	-15
PROJ. G	UC344	Consulta Parametrizada	1	1	0	3	26	8,7	21	-5
PROJ. G	UC345	CRUD Mestre/Detalhe	4	4	4	20	145	7,3	140	-5
PROJ. G	UC346	Caso de Uso Não Padronizado	5	0	0	5	47	9,4	35	-12
PROJ. G	UC347	CRUD Mestre/Detalhe	4	4	1	14	120	8,6	98	-22
PROJ. G	UC348	Caso de Uso Não Padronizado	5	5	0	15	133	8,9	105	-28
PROJ. G	UC349	Consulta Parametrizada	1	1	0	3	25	8,3	21	-4
PROJ. G	UC350	CRUD Detalhe	3	3	2	13	123	9,5	91	-32

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. G	UC351	Consulta Parametrizada	1	5	0	11	72	6,5	77	5
PROJ. G	UC352	CRUD Tabular	2	0	0	2	15	7,5	14	-1
PROJ. G	UC353	CRUD	2	5	0	12	98	8,2	84	-14
PROJ. G	UC354	CRUD	2	0	0	2	17	8,5	14	-3
PROJ. G	UC355	Caso de Uso Não Padronizado	5	1	0	7	47	6,7	49	2
PROJ. G	UC356	CRUD Mestre/Detalhe	4	2	1	10	74	7,4	70	-4
PROJ. G	UC357	Consulta Parametrizada	1	5	0	11	77	7,0	77	0
PROJ. G	UC358	Consulta Parametrizada	1	2	0	5	32	6,4	35	3
PROJ. G	UC359	Caso de Uso Não Padronizado	5	5	9	33	251	7,6	231	-20
PROJ. H	UC360	CRUD Mestre/Detalhe	4	2	2	18	110	6,1	126	16
PROJ. H	UC361	Consulta Parametrizada	1	0	0	1	8	8,0	7	-1
PROJ. H	UC362	Consulta Parametrizada	1	1	0	4	35	8,8	28	-7
PROJ. H	UC363	CRUD	2	0	0	2	16	8,0	14	-2
PROJ. H	UC364	Consulta Parametrizada	1	1	0	4	25	6,3	28	3
PROJ. H	UC365	Consulta Parametrizada	1	7	0	22	110	5,0	154	44
PROJ. H	UC366	CRUD	2	0	0	2	18	9,0	14	-4
PROJ. H	UC367	Consulta Parametrizada	1	0	0	1	12	12,0	7	-5

* Y é o esforço de desenvolvimento estimado com base no tamanho dos casos de uso (UC_{TP}) utilizando um valor de referência de produtividade de 7 horas por UCS.

** E é o valor do Erro da estimativa calculado com base na diferença entre o valor estimado (Y) e o esforço real de desenvolvimento do caso de uso.

ANEXO C – Lista de Casos de Uso dos Estudos de Casos (continuação)

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. H	UC368	CRUD Mestre/Detalhe	4	3	0	13	80	6,2	91	11
PROJ. H	UC369	CRUD	2	0	2	10	64	6,4	70	6
PROJ. H	UC370	CRUD Mestre/Detalhe	4	0	0	4	24	6,0	28	4
PROJ. H	UC371	CRUD Tabular	2	0	0	2	16	8,0	14	-2
PROJ. H	UC372	CRUD	2	5	0	17	96	5,6	119	23
PROJ. H	UC373	Caso de Uso Não Padronizado	5	4	0	17	110	6,5	119	9
PROJ. H	UC374	Consulta Parametrizada	1	2	0	7	35	5,0	49	14
PROJ. H	UC375	Consulta Parametrizada	1	1	0	4	24	6,0	28	4
PROJ. H	UC376	Caso de Uso Não Padronizado	5	0	0	5	27	5,4	35	8
PROJ. H	UC377	Consulta Parametrizada	1	3	1	14	90	6,4	98	8
PROJ. H	UC378	Consulta Parametrizada	1	1	1	8	48	6,0	56	8
PROJ. H	UC379	Consulta Parametrizada	1	1	1	8	54	6,8	56	2
PROJ. H	UC380	Consulta Parametrizada	1	1	1	8	45	5,6	56	11
PROJ. H	UC381	Consulta Parametrizada	1	3	1	14	88	6,3	98	10
PROJ. H	UC382	Consulta Parametrizada	1	3	1	14	80	5,7	98	18
PROJ. H	UC383	Relatório	3	2	0	9	44	4,9	63	19
PROJ. H	UC384	Relatório	3	2	0	9	60	6,7	63	3
PROJ. H	UC385	Relatório	3	1	0	6	35	5,8	42	7

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
PROJ. H	UC386	Serviço	5	0	0	5	32	6,4	35	3
PROJ. H	UC387	Serviço	5	0	0	5	44	8,8	35	-9
PROJ. H	UC388	Serviço	5	0	0	5	40	8,0	35	-5
PROJ. I	UC388	Consulta Parametrizada	1			1	8,8	8,8	7	-1,8
PROJ. I	UC389	Operação Básica sobre uma Entidade	1	1	1	3	8,8	2,9	21	12,2
PROJ. I	UC390	Operação Básica sobre uma Entidade	1	1	1	3	19,8	6,6	21	1,2
PROJ. I	UC391	Consulta Parametrizada	1			1	6,6	6,6	7	0,4
PROJ. I	UC392	Consulta Parametrizada	1	2	1	4	22	5,5	28	6
PROJ. I	UC393	CRUD Detalhes	3		2	5	26,4	5,28	35	8,6
PROJ. I	UC394	Consulta Parametrizada	1		1	2	13,2	6,6	14	0,8
PROJ. I	UC395	Consulta Parametrizada	1		1	2	13,2	6,6	14	0,8
PROJ. I	UC396	Consulta Parametrizada	1			1	8,8	8,8	7	-1,8
PROJ. I	UC397	Operação Básica sobre uma Entidade	1			1	8,8	8,8	7	-1,8
PROJ. I	UC398	Operação Básica sobre uma Entidade	1			1	11	11	7	-4
PROJ. I	UC399	Operação Básica sobre uma	1			1	6,6	6,6	7	0,4

* Y é o esforço de desenvolvimento estimado com base no tamanho dos casos de uso (UC_{TP}) utilizando um valor de referência de produtividade de 7 horas por UCS.

** E é o valor do Erro da estimativa calculado com base na diferença entre o valor estimado (Y) e o esforço real de desenvolvimento do caso de uso.

ANEXO C – Lista de Casos de Uso dos Estudos de Casos (continuação)

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
		Entidade								
PROJ. I	UC400	Consulta Parametrizada	1			1	8,8	8,8	7	-1,8
PROJ. I	UC401	Operação Básica sobre uma Entidade	1			1	8,8	8,8	7	-1,8
PROJ. I	UC402	Operação Básica sobre uma Entidade	1			1	11	11	7	-4
PROJ. I	UC403	Operação Básica sobre uma Entidade	1			1	6,6	6,6	7	0,4
PROJ. I	UC404	Operação Básica sobre uma Entidade	1			1	8,8	8,8	7	-1,8
PROJ. I	UC405	Operação Básica sobre uma Entidade	1			1	11	11	7	-4
PROJ. I	UC406	Operação Básica sobre uma Entidade	1	2		3	26,4	8,8	21	-5,4
PROJ. I	UC407	CRUD Detalhes	3		4	7	35,2	5,0	49	13,8
PROJ. I	UC408	Operação Básica sobre uma Entidade	1	2		3	17,6	5,9	21	3,4
PROJ. I	UC409	Operação Básica sobre uma Entidade	1	2		3	17,6	5,9	21	3,4
PROJ. I	UC410	Operação Básica sobre uma Entidade	1		2	3	22	7,3	21	-1
PROJ. I	UC411	Operação Básica sobre uma Entidade	1			1	6,6	6,6	7	0,4
PROJ. I	UC412	Operação Básica sobre uma	1			1	8,8	8,8	7	-1,8

Projeto	Caso Uso	Tipo de Caso de Uso	UC_{TP}	N_{SOBR}	N_{GUIR}	UC_S	Esforço	$H/_{UCS}$	Y*	E**
		Entidade								
PROJ. I	UC413	Consulta Parametrizada	1			1	6,6	6,6	7	0,4
PROJ. I	UC414	Operação Básica sobre uma Entidade	1			1	6,6	6,6	7	0,4
PROJ. I	UC415	Operação Básica sobre uma Entidade	1			1	6,6	6,6	7	0,4
PROJ. I	UC416	Operação Básica sobre uma Entidade	1			1	6,6	6,6	7	0,4
PROJ. I	UC417	Consulta Parametrizada	1			1	6,6	6,6	7	0,4
PROJ. I	UC418	Consulta Parametrizada	1			1	6,6	6,6	7	0,4
PROJ. I	UC419	Operação Básica sobre uma Entidade	1			1	6,6	6,6	7	0,4
PROJ. I	UC420	Operação Básica sobre uma Entidade	1			1	6,6	6,6	7	0,4
PROJ. I	UC421	CRUD Detalhes	3			3	17,6	5,9	21	3,4
PROJ. I	UC422	Operação Básica sobre uma Entidade	1			1	8,8	8,8	7	-1,8

* Y é o esforço de desenvolvimento estimado com base no tamanho dos casos de uso (UC_{TP}) utilizando um valor de referência de produtividade de 7 horas por UCS.

** E é o valor do Erro da estimativa calculado com base na diferença entre o valor estimado (Y) e o esforço real de desenvolvimento do caso de uso.