TUCP: Uma Extensão da Técnica UCP

Tatiana Cavalcanti Monteiro[1, 2], Carlo Giovano S. Pires[1], Arnaldo Dias Belchior[2]

[1]Instituto Atlântico
R. Chico Lemos, 946, CEP 60822-780 - Fortaleza – CE – Brasil

[2]Mestrado em Informática Aplicada - Universidade de Fortaleza (UNIFOR)
Av. Washington Soares, 1321, CEP 60811-341 - Fortaleza - CE – Brasil

{tatiana,cgiovano}@atlantico.com.br, belchior@unifor.br

Abstract. *The software engineering argues the implantation of activities of size estimates, effort, period and cost, as forms of improving the software projects planning and tracking. In spite of there being several techniques of estimates, the use of the same ones in software companies is not still such a common practice. The technique UCP (Use Case Points), for instance, it is adherent to object-oriented software products and based on use cases. However, they have been finding some situations where there are difficulties of obtaining resulted fully satisfactory when using the UCP. This work presents an extension of the technique UCP – that is TUCP – being looked for a calculation more refined for the projects effort. Besides, TUCP allows a more detailed vision of estimates in the main stages of the software cycle life, making possible the accomplishment of refinements of those estimates for an accompaniment more effective of the project.*

Resumo. *A engenharia de software recomenda a implantação de atividades de estimativas de tamanho, esforço, prazo e custo, como formas de melhorar o planejamento e o acompanhamento de projetos de software. Apesar de haver várias técnicas de estimativas, a utilização das mesmas em empresas de software ainda não é uma prática tão comum. A técnica UCP (Pontos por Caso de Uso), por exemplo, é aderente a produtos de software orientados a objetos e baseados em casos de uso. No entanto, têm-se encontrado algumas situações, onde há dificuldades de se obter resultados plenamente satisfatórios ao se utilizar a UCP. Este trabalho apresenta uma extensão da técnica UCP – a TUCP – buscando-se um cálculo mais acurado para o esforço de projetos. Além disso, a TUCP permite uma visão mais detalhada de estimativas nas principais etapas do ciclo de vida do software, possibilitando a realização de refinamentos dessas estimativas para um acompanhamento mais efetivo do projeto.*

1. Introdução

As estimativas apóiam essencialmente as atividades de planejamento e acompanhamento de projetos de software. Estimativas eficientes permitem a verificação da viabilidade do projeto, a elaboração de propostas técnicas e comerciais, a confecção de planos e cronogramas detalhados, e o acompanhamento efetivo de projetos. Dentre as diversas técnicas para se estimar projetos de software, a UCP (*Use Case Points*), que surgiu nos anos noventa, tem sido aprimorada e tem apresentado crescente utilização. Esta técnica é adequada a projetos, que descrevem seus requisitos de software através de casos de uso.

Este trabalho apresenta uma extensão da técnica UCP, a TUPC (*Technical Use Case Points*), propondo um cálculo mais acurado para o cálculo do esforço de projetos, e permitindo uma visão mais detalhada das estimativas por etapa do ciclo de vida do software, possibilitando a realização de refinamentos dessas estimativas para um acompanhamento mais efetivo do projeto. Além disso, a TUCP propõe calibrações nos fatores de produtividade nessas etapas, permitindo assim a obtenção de estimativas mais acuradas.

A organização deste trabalho é feita em cinco seções. Na seção 2, descreve-se sobre os conceitos de casos de uso. Na seção 3, delineia-se sobre estimativas de projetos de software. Na seção 4, discorre-se sobre a extensão da UCP proposta, a TUCP. Na seção 5, apresenta-se um estudo de caso, utilizando-se a TUCP. Finalmente, na seção 6, são mostradas as conclusões.

2. Casos de Uso

Os casos de uso são utilizados para descrever requisitos visíveis de um sistema, na fase de análise de requisitos de um projeto e contribuírem para o plano de teste e manuais de usuários. Podem auxiliar na elaboração e validação de uma proposta de projeto e para a garantia de que todos os requisitos tenham sido contemplados. Os casos de uso também são utilizados para a elaboração do cronograma do projeto, descrevendo no plano de projeto o que será disponibilizado para o cliente [Schneider *et al.* 2001].

Os casos de uso podem ser modelados a partir dos cenários obtidos para o sistema. Um cenário mostra uma seqüência de passos, que descreve as interações entre um ator e o sistema, isto é, uma alternativa do que pode acontecer. Cada passo deve ser uma declaração simples, evidenciando claramente quem o está executando. O passo deve apresentar a intenção do ator e não os mecanismos daquilo que o ator faz. Portanto, a interface com o caso de uso não deve ser descrita [Fowler 2005].

Um caso de uso descreve o comportamento de um sistema, que produz um resultado mensurável de valor para um ator. Este expõe aquilo que os atores querem que o sistema execute para eles. Isso deve ser uma tarefa completa na perspectiva de um ator. Portanto, cada caso de uso identificado deverá estar associado à pelo menos um ator [Schneider *et al.* 2001].

Para iniciar realmente a construção de um sistema, necessita-se de mais detalhes técnicos. O fluxo de eventos poderá documentar o fluxo lógico de execução de um caso de uso. É uma forma de detalhar como um caso de uso será iniciado e executado [OMG 2003].

A seguir, serão apresentados os principais conceitos sobre casos de uso, juntamente com um guia de utilização dos mesmos, proposto neste trabalho.

2.1. Ator

Um ator representa um papel que uma pessoa, um dispositivo de hardware ou um outro sistema pode desempenhar em relação ao sistema. Os nomes dos atores devem denotar claramente o papel que estes desempenham [RUP 2003] e são representados por um *stickman*.

Para se identificar os atores de um sistema, deve-se pensar primeiramente nos possíveis usurários que o utilizarão, e também como esses atores podem ser categorizados. Normalmente, é tida como uma boa prática pensar em alguns usuários (dois ou três) e verificar se os atores identificados vão cobrir as necessidades desses usuários.

A melhor forma de identificar casos de uso é considerar o que cada ator exige do sistema. Isto porque o sistema existe apenas para seus usuários e deve estar baseado nas necessidades deles. Muitas das necessidades dos atores serão reconhecidas através dos requisitos funcionais especificados no sistema. Essas necessidades podem ser modeladas através de fluxos de eventos [Kruchten 2001].

2.2. Fluxo de Eventos

As duas principais partes do fluxo de eventos são: o fluxo de eventos básico (ou *fluxo básico*) e os fluxos de eventos alternativos (ou *fluxos alternativos*). Opcionalmente, podem-se ter também os fluxos de eventos de exceção (ou fluxos de exceções). Esses fluxos podem ser capturados em seções de uma especificação de casos de uso.

O fluxo básico deve abordar o que "geralmente" ocorre quando o caso de uso é executado, enfatizando o cenário de sucesso do início ao fim (cenário *happy day*). É recomendável que ele seja relativamente pequeno e de fácil leitura, semelhante a uma pequena história, apresentando os passos necessários para alcançar o objetivo principal do caso de uso.

Os fluxos alternativos abordam o comportamento de caráter opcional ou excepcional em relação ao comportamento normal e também as variações do comportamento normal. Pode-se pensar nos fluxos alternativos como "desvios" ou variantes do fluxo básico, alguns dos quais voltarão ao fluxo básico e alguns finalizarão a execução do caso de uso [Kruchten 2001].

Uma seqüência específica de ações que mapeiam comportamentos em fluxos de eventos pode ser obtida através de cenários. Um cenário é um caminho através do qual fluxos de eventos de caso de uso iniciam em um ponto e terminam em outro. Ele pode envolver o fluxo básico e um número qualquer de combinações de fluxos alternativos.

2.3. Tipos de Relacionamentos

Um caso de uso pode apresentar três tipos relacionamentos: inclusão, extensão, e generalização. Nesses relacionamentos, o caso de uso original, que é modificado, é chamado de *caso de uso base*. Todos os casos de uso abstratos são do tipo de inclusão, extensão, ou generalização. Portanto, não é necessário ter um ator associado a esses casos de uso.

O relacionamento de inclusão (*include*) conecta um *caso de uso base* a um *caso de uso de inclusão*. Um caso de uso de inclusão descreve um segmento de comportamento que é inserido em uma instância de caso de uso ao ser executado o caso de uso base. O caso de uso base explicitamente insere o caso de uso de inclusão, mas nenhum deles pode acessar os atributos um do outro [Kruchten 2001].

O relacionamento de extensão (*extende*) estabelece a conexão entre um *caso de uso de extensão* e um *caso de uso base* [Schneider *et al.* 2001]. Esse relacionamento pode ser definido em que parte do caso de uso base. A extensão deve ser inserida fazendo referência a *pontos de extensão* no caso de uso base [RUP 2003].

Uma generalização de casos de uso é um relacionamento de um caso de uso filho com um caso de uso pai, especificando como um filho pode adotar todo o comportamento e as características descritas para o pai [OMG 2003]. Deve-se usar o relacionamento de generalização se os casos de uso pai e filhos compartilharem estruturas e propósitos similares.

A seguir serão apresentados outros conceitos importantes que poderão compor uma especificação de casos de uso: precondição, pós-condição, requisitos especiais, e regras de negócio.

2.4. Outros Conceitos Importantes

A *precondição* e a *pós-condição* são utilizadas para esclarecer como o fluxo de eventos inicia e termina respectivamente. Todavia, estes conceitos somente deverão ser utilizados se vierem agregar algum valor ao público alvo do caso de uso.

A precondição corresponde ao estado do sistema e da sua vizinhança, que é exigido antes do início da execução de um caso de uso. Os estados descritos na precondição devem ser

observados pelo usuário. No entanto, nem todos os casos de uso irão precisar de precondição. A pós-condição é o estado que o sistema pode apresentar após o término do caso de uso. Os estados descritos devem ser observados também pelo usuário [Kruchten 2001].

Um requisito especial é um requisito não funcional pertencente a um caso de uso, que não é coberto no texto do fluxo de eventos, por ter havido dificuldades na definição do mesmo, como: requisitos legais, padrões de aplicativo, restrições de projeto, etc. [RUP 2003].

As regras de negócio relevantes para o sistema devem ser descritas fora do fluxo de eventos, como um item à parte na especificação de casos de uso. Caso se justifique, seja pela complexidade ou relevância do sistema, seja por orientação da organização, as regras de negócio poderão ser alocadas em um documento próprio para este fim.

2.5. Especificação dos Casos de Uso

Várias têm sido as propostas de como se construir casos de uso efetivos, através de formulários e regras para a descrição dos mesmos [Anda *et al*. 2001; Ribu 2001; Anda 2002; OMG 2003; Cockburn 2001]. No entanto, ainda não há uma padronização para a descrição de casos de uso [Fowler 2005], especialmente por esta ser de natureza subjetiva e depender do grau de detalhamento e conhecimento das informações disponibilizadas.

Neste trabalho, é proposto um modelo de *Especificação de Casos de Uso* (Apêndice A), a partir de pesquisas na literatura sobre o tema [RUP 2003; OMG 2003; Ribu 2001; Cockburn 2001, Schneider *et al*. 2001]. Isto porque a forma de especificar casos de uso é um fator preponderante no cálculo da estimativa de tamanho do software. Outro fator para que a estimativa de tamanho seja suficientemente precisa e passível de ser calculada, é que com relação ao *conceito de transação* em um caso de uso.

Seguem-se algumas instruções a serem seguidas para identificar melhor o que é ou não uma transação em um caso de uso, a partir de conceitos de Anda (2002), Schneider e Winters (2001) e Ribu (2001).

O que pode ser considerado uma transação são passos do fluxo de eventos do caso de uso que: (*i*) contém campos de entrada com valores passíveis de escolha; (*ii*) apresentem retorno de consultas com filtros preenchidos por buscas em bancos de dados; (*iii*) proporcionem validações complexas de negócio; (*iv*) contém geração de relatório; (*v*) apresentem funcionalidades de consultas auxiliares como casos de uso de extensão; (*vi*) existam validações simples de campo de entrada de dados.

O que não deve ser considerado uma transação são passos do fluxo de eventos que: (*i*) descrevem o início e o fim do caso de uso; (*ii*) detalham a interação entre o sistema e o ator; (*iii*) solicitem escolhas com valores fixos (sem leitura de dados); (*iv*) façam leituras auxiliares de dados que já tenham sido feitas em outros fluxos do mesmo caso de uso; e (*v*) usem fluxos alternativos do caso de uso de validações simples e que contenham mensagens de erro. A seção abordará a importância de estimativas de projeto de software.

3. Estimativas de Projetos de Software

Um dos principais riscos que atinge o processo de estimativas é a falta de credibilidade nas estimativas pelas equipes de desenvolvimento [Boehm 2000]. Isto ocorre quando as estimativas são irreais, isto é, os projetos são freqüentemente subestimados ou superestimados. A precisão das estimativas de tamanho torna-se fundamental para a elaboração de cronograma e orçamento realistas, pois essas estimativas constituem-se na base para a derivação das estimativas de esforço, prazo, e custo [SEI 2002].

Segundo Vasquez (2003), o processo de estimativa de um projeto de software envolve quatro atividades básicas: (*i*) estimar o tamanho do produto a ser desenvolvido; (*ii*) estimar o esforço empregado na execução do projeto; (*iii*) estimar o prazo do projeto; e (*iv*) estimar o custo do projeto.

O cálculo do esforço do projeto pode ser obtido a partir da estimativa de tamanho do produto. O esforço total é obtido baseado no processo de desenvolvimento, que envolve muito mais do que a simples atividade de codificação do software – o fato é que a codificação é freqüentemente a menor parte do esforço real de todo o projeto. Elaboração de documentos, implementação de protótipos, projeto do produto a ser entregue, revisão e teste do código levam uma grande fatia de todo o esforço do projeto [Peters 1999].

Com as estimativas de tamanho e esforço calculadas para um projeto de software pode-se, então, determinar as estimativas de prazo. Isto geralmente envolve estimar quantas pessoas estarão envolvidas no projeto, que atividades serão executadas (*WBS*), e quando essas atividades iniciarão e serão finalizadas.

O custo é comumente proporcional ao esforço despendido para a construção do mesmo, onde o trabalho humano é o principal recurso a ser consumido. Conseqüentemente, o custo é com freqüência associado a homens-mês (h.m) ou homens-hora (h.h).

Atualmente, existem diversas métricas de estimativa de tamanho de projetos de software. Entretanto, não é trivial a seleção de uma métrica que seja a mais apropriada para uma organização. Podem-se citar algumas das métricas mais conhecidas e utilizadas atualmente: *Function Point Analysis* (Análise por Pontos de Função); *Bang*; *Features Points*; *Boeing´s ED Function Point Analysis*, MK II *Function Point Analysis*, COSMIC *Full Function Point*, *Internet Points*, *Domino Points*, *Class-Method Points*, *Use Case Points* (Pontos de Caso de Uso) [Garmus and Herron 2000; Mcphee 1999; Roetzheim 2000].

3.2. Pontos de Casos de Uso

A UCP (*Use Case Points*) é uma técnica de estimativa de tamanho de projeto de software orientado a objetos, criada por Karner em 1993, com base na FPA, MK II FPA e no processo "*Objectory*", onde foi desenvolvida a técnica de diagramação para o conceito de casos de uso [Ribu 2001].

A UCP é simples, rápida e fácil de usar [Damodaran; Washington s.d]. O processo de contagem dessa métrica é apresentado no

Quadro 1, onde são contados atores, e casos de uso. Com base nessa contagem são calculados os UCP não ajustados. Posteriormente, são determinadas as complexidades dos *fatores técnicos* e dos *fatores ambientais*.

A primeira etapa é classificar os atores envolvidos no sistema, e efetuar o somatório dos produtos do número de atores de cada tipo pelo respectivo peso (Tabela 1). Com isso obtém-se o peso total dos atores do sistema – UAW (*Unadjusted Actor Weight*).

Quadro 1. Etapas do processo de contagem de UCP [Ribu 2001]

1. Contar os atores e atribuir o grau de complexidade.
2. Contar os casos de uso e atribuir o grau de complexidade.
3. Somar o total de atores com o total de casos de uso para obter o UCP não-ajustado.

4. Determinar a complexidade do fator técnico.
5. Determinar a complexidade do fator ambiental.
6. Calcular o UCP ajustado.

Tabela 1. Classificação dos Tipos de Atores

Tipo de Ator	Descrição	Peso
Simples	Aplicação com API definida	1
Intermediário	Outro sistema interagindo através de um protocolo de comunicação, como TCP/IP ou FTP	2
Complexo	Um usuário interagindo através de uma interface gráfica (stand-alone ou Web)	3

Uma vez calculado o UAW, a etapa seguinte é classificar os casos de uso e calcular o peso bruto dos mesmos – UUCW (*Unadjusted Use Case Weight*).

Para fins de cálculo, dividem-se os casos de uso em três níveis de complexidade, de acordo com o número de transações (ver seção 2) envolvidas em seu processamento (Tabela 2).

Tabela 2. Classificação dos Tipos de Casos de Uso [Karner 1993]

Tipo de Caso de Uso	Número de Transações	Peso
Simples	Até 3 transações	5
Intermediário	De 4 a 7 transações	10
Complexo	Acima de 7 transações	15

Para se calcular os pontos de caso de uso não ajustados – UUCP (*Unadjusted Use Case Points*) – basta efetuar o somatório entre a complexidade de atores (etapa 1) e o cálculo da complexidade dos casos de uso (etapa 2), conforme a Eq. 1.

$$UUCP = \sum UAW + \sum UUCW \quad \text{(Eq. 1)}$$

O cálculo da complexidade do fator técnico do projeto – TCF (*Technical Complexity Factor*) – é mostrado na Eq. 2. O TFactor é obtido através da soma dos níveis de influência atribuídos a cada fator técnico multiplicado por seu peso correspondente [Ribu 2001].

$$TCF = 0,6 + (0,01 * TFator) \quad \text{(Eq. 2)}$$

O cálculo dos fatores ambientais – EF (*Environmental Factor*) – diz respeito à experiência da equipe, à estabilidade do projeto, e à motivação dos programadores [Ribu 2001]. O EFactor é o somatório de todos os produtos entre o peso de cada fator ambiental e seu grau de influência atribuído. Esse fator é calculado na Eq. 3.

$$TCF = 1,4 + (-0,03 * EFactor) \quad \text{(Eq. 3)}$$

Finalmente, o cálculo do tamanho do sistema em pontos por caso de uso (UCP) é apresentado na Eq. 4.

$$UCP = UUCP + TCF + TCF \quad \text{(Eq. 4)}$$

Na seção seguinte, será apresentada a TUCP que é uma extensão da UCP, buscando um cálculo mais acurado para o esforço despendido em projetos.

4. TUCP

A precisão da UCP é dependente de uma padronização nas especificações de casos de uso e no entendimento claro do que seja o conceito de transação. Estas questões influenciam diretamente no cálculo do tamanho do sistema.

No cálculo do tamanho do software na UCP, os fatores de ambiente (EF) são considerados. Assim sendo, o uso desses fatores pode levar a diferentes tamanhos, dependendo da equipe ou empresa que desenvolver um dado projeto. Assim, entende-se que apenas os fatores técnicos e não-técnicos deveriam ser tidos na estimativa de tamanho, evitando gerar dependência do tamanho com características da equipe de desenvolvimento e da empresa. Todavia, os fatores ambientais têm sua influência efetiva no cálculo das estimativas de esforço.

A UCP estima o software como um produto único, não havendo um detalhamento das estimativas de tamanho durante as etapas do ciclo de vida (requisitos, análise e projeto, codificação e testes, por exemplo). Este trabalho, baseado em [Meneses 2001], fornece uma visão mais detalhada das estimativas (tamanho, esforço, prazo e custos) por etapa do processo de desenvolvimento, possibilitando realizar refinamentos e acompanhamentos dessas estimativas ao longo desse processo.

A TUCP envolve os seguintes pontos, em seu processo de extensão da UCP:

- Propõe um modelo para a *Especificação de Casos de Uso* (Apêndice A), pois isto influencia fortemente no cálculo do tamanho desses casos de uso.
- Detalha o conceito de transação no contexto de casos de uso, por afetar diretamente o cálculo dos UUCPs (seção 2.5).
- Refina a contagem de pontos por casos de uso complexos (Tabela 3), para evitar valores subestimatidos, quando o número de transações é grande.
- Desatrela os Fatores de ambiente (EFs) do cálculo do tamanho, por entender que o tamanho é uma grandeza física, e não deveria ter seu valor alterado em função desses fatores.
- Considera os Fatores de ambiente (EFs) apenas no cálculo do Esforço, juntamente com o Fator de produtividade (de forma semelhante à UCP).
- Granulariza o cálculo do tamanho do projeto por cada etapa do processo de desenvolvimento. O somatório de todos esses tamanhos será o tamanho final de todo o projeto.

4.1. Cálculo do Tamanho

O cálculo do tamanho da TUCP é mostrado na Eq. 5, onde TUUCP (*Technical Unadjusted Use Case Points*) são os *pontos técnicos de caso de uso não ajustados*.

$$TUCP = TUUCP * TCF \quad \text{(Eq. 5)}$$

O cálculo de TUUCP é dado na Eq. 6.

$$TUUCP = \sum UAW + \sum TUUCW \quad \text{(Eq. 6)}$$

O cálculo de UAW é semelhante ao da UCP. Já o cálculo de TUUCW foi estendido para casos de uso complexos, que tinham um grande número de transações.

Através de experiências em dezenas de projetos de software utilizando-se a técnica UCP, em uma organização certificada SW-CMM, nível 2, constataram-se atrasos na entrega do produto final para o cliente. Esses atrasos, em geral, estavam relacionados a projetos que continham casos de uso complexos, especialmente aqueles com um grande número de transações (maior que 12 transações).

Com base nesta observação, uma adaptação no cálculo dos pesos para os casos de uso complexos foi elaborada. Uma nova tabela de valores para a classificação do peso foi montada para refletir essa questão apresentada (Tabela 3).

Tabela 3. Classificação dos Tipos de Casos de Uso na TUCP

Tipo de Caso de Uso	Descrição	Peso (TUUCW)
simples	até 3 transações incluindo os passos alternativos	5
médio	de 4 a 7 transações incluindo os passos alternativos	10
complexo	de 8 a *t* transações incluindo os passos alternativos	15
n-complexo	acima de *t* transações	P_X

Os casos de uso com até *t* transações será calculado da mesma maneira apresentada por Karner (1993). Acima de *t* transações, o tipo de caso de uso será denominado de *n*-complexo. O cálculo do peso dos casos de usos *n*-complexos é exibido na Eq. 7.

$$\text{TUUCW} = 15n + p \quad \text{(Eq. 7)}$$

Onde: $n = T / t$; e T = número de transações do caso de uso; p = é o peso obtido, quando o resto da divisão (r) de T / t é aplicado ao peso original (simples, médio, e complexo), (Tabela 3). Neste contexto, se $r = 0$, então $p = 0$; se $r \in [1, 3]$, então $p = 5$; se $r \in [4, 7]$, então $p = 10$; e se $r \in [8, t\text{-}1]$, então $p = 15$.

Por exemplo: um caso de uso com 13 transações, e com $t = 11$, temos $T = 13$, $n = 13 / 11$, como $r \in [1, 3]$, então $p = 5$, assim:

$$\text{TUUCW} = 15 \text{ x } 1 + 5 = 20$$

A definição do valor de *t* poderá depender das características da organização, ou até mesmo das características de um dado tipo de projeto. Recomenda-se a calibração do valor de *t*, após ter sido levantado um histórico de (tipos) projetos concluídos, que possuam casos de uso do tipo *n*-complexo (Tabela 3). Essa calibração deverá ser realizada baseada no esforço real de implementação do conjunto de projetos considerados.

Após a definição do valor de *t*, o peso de um caso de uso *n*-complexo vai sendo incrementado em um ciclo do tipo (3; 4; *k*). Isto significa dizer que a cada ($k + 3$), ($k + 4$), ($k + k$), ($2k + 3$) ... transações, e assim por diante, o valor do peso é modificado. O valor de *k* é dado pela Eq. 8.

$$k = t - 7 \quad \text{(Eq. 8)}$$

No exemplo da Eq. 8, o valor utilizado foi $k = 4$. Esse valor mostrou-se mais apropriado para o conjunto de projetos desenvolvidos na organização SW-CMM, nível 2, onde foram realizados os experimentos. No entanto, percebeu-se que alguns dos projetos do experimento tinham características peculiares (e.g.: utilização de *frameworks* de desenvolvimento, forte grau de *reuso*). Para estes casos, um valor de $k > 4$ indicava resultados mais próximos do esforço real despendido nesses projetos.

4.2. TUCP por Etapa do Ciclo de Vida do Projeto

Como o esforço de um projeto é diretamente proporcional a seu tamanho, então o tamanho de um caso de uso por etapa do ciclo de vida pode ser baseado no percentual de esforço empregado para seu desenvolvimento. Assim sendo, e baseado no trabalho de [Meneses 2001], definiu-se a estimativa de tamanho do caso de uso por etapa do ciclo de vida na Eq.9.

$$\text{TUCP}_{(\text{UC\_etapa})} = \left(\left(\frac{\text{TUCP}}{\sum \text{TUUCW}} \right) * \text{TUUCW}_{(\text{UC})} \right) * \text{Percentual de Esforço}_{(\text{etapa})} \quad \text{(Eq. 9)}$$

O $TUCP_{(UC\_etapa)}$ é tamanho do caso de uso por etapa do ciclo de vida, sendo definido proporcionalmente a seu fator de complexidade e ao percentual de distribuição de esforço para a etapa do ciclo de vida considerada. O $TUUCW_{(UC)}$ é o peso do caso de uso que está sendo calculado.

O tamanho TUCP do caso de uso como um todo ao longo do projeto é mostrado na Eq. 10.

$$TUCP_{(caso\ de\ uso)} = \left(\frac{TUCP}{\sum TUUCW} \right) * TUUCW_{(UC)} \quad \text{(Eq. 10)}$$

O tamanho TUCP de uma etapa do projeto é dado pela Eq. 11.

$$TUCP_{(etapa)} = TUCP * \text{Percentual de Esforço}_{(etapa)} \quad \text{(Eq. 11)}$$

4.3. Estimativa de Esforço do Projeto

Para a obtenção da estimativa de esforço total do projeto, a TUCP utiliza um fator de produtividade (PROD), o fator de ambiente (EF), e o tamanho TUCP. O fator de produtividade pode ser calibrado de acordo com a produtividade da equipe de um projeto.

Karner (1993) propôs um fator de produtividade de 20 h.h (homens-hora) por UCP. Schneider e Winters, citado por Ribu (2001), analisando a proposta de Karner (1993), propuseram um fator de produtividade de 20 h.h por UCP, para projetos com uma equipe estável e experiente, e 28 h.h por UCP para projetos com requisitos não estáveis e uma equipe não experiente. Spaks, citado por Anda *et al*. (2001), mostrou que esse fator pode variar entre 15 h.h e 30 h.h por UCP.

Na TUCP, o cálculo do esforço total do projeto é semelhante à UCP, isto é, o fator de ambiente (EF) é considerado com mostra a Eq. 12.

$$Esforço = TUCP * EF * PROD \quad \text{(Eq. 12)}$$

4.3.1. Estimativa de Esforço por etapa do ciclo de vida do projeto

Na UCP [Karner 1993], o fator produtividade é único para todo o projeto. No entanto, constatou-se que o fator de produtividade pode variar dependendo da etapa do ciclo de vida do projeto. Isto porque a experiência da equipe de desenvolvimento pode variar de uma etapa para outra, já que, normalmente, as pessoas envolvidas em uma etapa não são as mesmas de uma outra etapa. Além disso, outros fatores podem influenciar na produtividade como: tipo de equipe, linguagem, experiência, utilização de *frameworks*, etc.

A calibragem para a produtividade para cada etapa do ciclo de vida deve ser obtida da base histórica organizacional. Assim, o esforço por etapa do ciclo de vida do projeto é calculado na Eq. 13.

$$Esforço_{(etapa)} = TUCP_{(etapa)} * EF * PROD_{(etapa)} \quad \text{(Eq. 13)}$$

A seguir, será apresentado um estudo de caso para um melhor entendimento das estimativas propostas. Os resultados expostos foram obtidos em projetos reais que utilizaram a TUCP.

5. Estudo de Caso

O estudo de caso foi realizado em uma organização de pesquisa e desenvolvimento certificada como SW-CMM nível 2. Essa organização utiliza o RUP (*Rational Unified Process*) [Kruchten 2000; Rational 2003] em projetos de P&D em diversas áreas de tecnologia da informação e telecomunicações, e utiliza diversas plataformas como J2EE, J2ME e .NET.

Neste trabalho, serão apresentados os resultados de três projetos. No entanto, apenas um projeto será detalhado para fins de demonstração de cálculo da TUCP. Na organização, foi considerado t = 11, após vários refinamentos do valor de k (Eq. 8).

O projeto A foi desenvolvido na plataforma J2EE com uma equipe de projeto composta por seis pessoas consideradas experientes. Esse projeto possuía vinte e quatro casos de uso, sendo sete simples, dez intermediários, um complexo e seis *n*-complexo (Tabela 3). Este sistema tinha apenas um ator complexo (Tabela 1). O TUUCP deste sistema será apresentado a seguir.

$$TUUCP = \sum UAW + \sum TUUCW = 3 + 310 = 313$$

O tamanho TUCP deste sistema foi calculado com base nas características do projeto e é apresentado abaixo.

$$TUCP = TUUCP * TCF = 313 * 1{,}125 \cong 352{,}1$$

A partir do cálculo do fator de complexidade ambiental (EF), e considerando-se a produtividade da equipe de 20 h.h, obteve-se o esforço do projeto a seguir.

$$Esforço = TUCP * EF * PROD = 352{,}1 * 0{,}815 * 20 \cong 5739{,}6 \text{ h.h}$$

Com a TUCP é possível se ter estimativas de tamanho e esforço por etapa do ciclo de vida para cada caso de uso. Como o Projeto A possui vinte e quatro casos de uso, será apresentado apenas o cálculo para um único caso de uso (UC_01), detalhado por etapa do ciclo de vida. A Tabela 4 contém um exemplo de um caso de uso do sistema com a sua distribuição de tamanho e esforço por etapa do ciclo de vida (Requisitos (REQ), Análise e Projeto (A&P), Codificação (COD) e Teste (TST)).

Tabela 4. Detalhamento do UC_1 do Projeto A por etapa do ciclo de vida

Caso de Uso	REQ (TUCP)	REQ (h.h)	A&P (TUCP)	A&P (h.h)	COD (TUCP)	COD (h.h)	TST (TUCP)	TST (h.h)	Tamanho (TUCP)	TUCP (h.h)
UC_01	1,1	18,5	1,4	23,1	2,6	41,7	0,6	9,3	5,7	92,6

A partir da Eq. 11, tem-se o cálculo do tamanho do UC_01, abaixo.

$$TUCP_{(UC\_01)} = \left(\left(\frac{352{,}1}{310} \right) * 5 \right) \cong 5{,}7$$

Tanto a produtividade da equipe, quanto o percentual de esforço foram obtidos a partir da base histórica organizacional (Tabela 5).

Tabela 5. Percentual de esforço por etapa do ciclo de vida

Etapa	Percentual de Esforço
Requisitos	20%
Análise e Projeto	25%
Codificação	45%
Testes	10%

Para o cálculo do tamanho do caso de uso UC_01 nas etapas (requisitos, análise e projeto, codificação e testes) foi utilizada a Eq. 9, como se segue.

1. $TUCP_{(UC\_01\_REQ)} = \left(\left(\frac{352,1}{310}\right)*5\right)*0,20 \cong 1,1$

2. $TUCP_{(UC\_01\_A\ \&\ P)} = \left(\left(\frac{352,1}{310}\right)*5\right)*0,25 \cong 1,4$

3. $TUCP_{(UC\_01\_COD)} = \left(\left(\frac{352,1}{310}\right)*5\right)*0,45 \cong 2,6$

4. $TUCP_{(UC\_01\_TST)} = \left(\left(\frac{352,1}{310}\right)*5\right)*0,10 \cong 0,6$

O cálculo do tamanho TUCP de cada etapa (requisitos, análise e projeto, codificação e testes) do ciclo de vida do projeto foi realizado a partir da Eq. 11, como se segue.

1. $TUCP_{(REQ)} = 352,1 \text{ x } 0,20 \cong 70,4$

2. $TUCP_{(A\ \&\ P)} = 352,1 \text{ x } 0,25 \cong 88,0$

3. $TUCP_{(COD)} = 352,1 \text{ x } 0,45 \cong 158,5$

4. $TUCP_{(TST)} = 352,1 \text{ x } 0,10 \cong 35,2$

A partir do tamanho TUCP de cada etapa do ciclo de vida do projeto, o esforço por etapa foi calculado abaixo pela Eq. 13.

1. $Esforço_{(REQ)} = 70,4*0,815*20 \cong 1147,9$

2. $Esforço_{(A\ \&\ P)} = 88,03*0,815*20 \cong 1434,9$

3. $Esforço_{(COD)} = 158,5*0,815*20 \cong 2582,8$

4. $Esforço_{(TST)} = 35,2*0,815*20 \cong 573,9$

Na Tabela 6, são mostrados os dados do Projeto A: (*i*) etapas do ciclo de vida; (*ii*) o tamanho em TUCP; (*iii*) o esforço estimado; (*iv*) o esforço real; (*v*) a produtividade estimada; (*vi*) a produtividade real; (*vii*) o percentual de esforço realizado de cada etapa do ciclo de vida do projeto; e (*viii*) o percentual de erro em relação ao esforço estimado. Pode ser percebido que a equipe foi muito produtiva, pois sua média de produtividade foi de quase 15 h.h.

Para este projeto não foi preciso o ajuste no fator de produtividade, já a produtividade é de 20 h.h. Entretanto, se um projeto tivesse uma equipe não experiente com as atividades de codificação, poderia ter sido feito um ajuste no fator de produtividade do esforço ($PROD_{(etapa)}$) para 28 h.h, por exemplo.

Tabela 6. Dados do Projeto A calculados com base na TUCP

Etapas do Ciclo de Vida	Tamanho (TUCP)	Esforço Estimado (H.H)	Esforço Real (H.H)	Produtividade Estimada (H.H/TUCP)	Produtividade Real (H.H/TUCP)	% Esforço Real	% Erro Estimado
Requisitos	70,4	1147,9	956,2	20	13,58	18,19	-20
Análise e Projeto	88,0	1434,9	1007,8	20	11,45	19,17	-42
Codificação	158,5	2582,8	2734,5	20	17,26	52,02	6
Testes	35,2	574,0	558,0	20	15,85	10,62	-3
Total	**352,1**	**5739,6**	**5256,5**	**20**	**14,93**	**100**	**-9**

O cálculo para o percentual de erro estimado, considerando o esforço real e esforço estimado, para cada etapa utilizou a métrica SER (*Symmetric Relative Error*), proposta por M. Jorgensen e D. Sjobeg (Ribu, 2001):

- SER = (Real - Estimado) / Real $\Leftrightarrow$ Real $\leq$ Estimado
- SER = (Real - Estimado) / Estimado $\Leftrightarrow$ Real $\geq$ Estimado

Na Tabela 7 e Tabela 8, são apresentados os dados de três projetos calculados utilizando-se a técnica TUCP e a UCP respectivamente. Esses dados foram comparados com os valores reais obtidos nos projetos. Pode ser percebido que o percentual de erro estimado em TUCP foi menor do que o percentual de erro em UCP para dois dos projetos, e no Projeto C o percentual foi o mesmo. Neste caso, o Projeto não apresentava casos de uso do tipo *n*-complexo (Tabela 9).

Tabela 7. Tamanho e esforço dos projetos A, B e C pela TUCP

Projetos	Tamanho (TUCP)	Esforço Estimado (H.H)	Esforço Real (H.H)	Produtividade Estimada (H.H/TUCP)	Produtividade Real (H.H/TUCP)	% Erro Estimado (TUCP)
Projeto A	352,1	5739,6	5256,5	20	14,93	-9
Projeto B	74,7	1352,5	1477,0	20	19,77	9
Projeto C	103,3	1833,8	1871,0	24	18,12	2

Tabela 8. Tamanho e esforço dos projetos A, B e C pela UCP

Projetos	Tamanho (UCP)	Esforço Estimado (H.H)	Esforço Real (H.H)	Produtividade Estimada (H.H/UCP)	Produtividade Real (H.H/UCP)	% Erro Estimado (UCP)
Projeto A	222,8	4457,0	5256,5	20	23,59	18
Projeto B	62,08	1241,7	1477,0	20	23,79	19
Projeto C	76,4	1833,8	1871,0	24	24,49	2

Tabela 9. Tipos e quantidades de casos de uso nos projetos A, B e C

Projetos	Tipo de Caso de Uso				Total
	Simples	Intermediário	Complexo	N-Complexo	
Projeto A	7	10	1	6	24
Projeto B	3	2	0	1	6
Projeto C	1	3	4	0	8

Vale salientar que a comparação da eficácia de uma técnica (UCP) em relação à outra (TUCP) dá-se apenas através do cálculo do esforço, e quando os projetos possuem casos de uso *n*-complexo. Não se podem comparar os tamanhos da UCP e da TUCP, por terem os atributos de cálculo distintos.

5. Conclusões

Este trabalho apresentou a técnica TUCP, que é uma extensão da UCP. Nos projetos experimentados, a TUCP apresentou um cálculo mais acurado para o esforço de projetos. Além disso, a TUCP permite uma visão mais detalhada de estimativas nas principais etapas do ciclo de vida do software, possibilitando um acompanhamento mais efetivo do projeto.

Podem-se destacar como principais contribuições deste trabalho:

- Uma proposta de um *Modelo de Especificação de Casos de Uso*, pois isto influencia fortemente no cálculo do tamanho desses casos de uso, juntamente com a conceituação mais apurada de transação no contexto de casos de uso.

- Criação do tipo de caso de uso denominado *n*-complexo, para casos de uso complexos com um número elevado de transações.
- Granularização do cálculo do tamanho do projeto por cada etapa do processo de desenvolvimento, permitindo uma visão mais detalhada das estimativas ao longo do projeto.
- O fator de produtividade de todo o projeto e por etapa do ciclo de vida gerou um percentual de erro menor em relação ao valor real, possibilitando assim o planejamento e o acompanhamento mais efetivos do projeto.

Como conclusões mais importantes deste trabalho, podem-se destacar:

- Uma base histórica de estimativas da organização deve ser implementada (como recomenda a literatura de métricas de software), para que sirvam de fundamentação nas calibrações, e nos fatores de produtividade para projetos futuros.
- A especificação de caso de uso deve ser descrita em um nível de detalhamento adequado, para que a estimativa baseada em UCP / TUCP seja eficaz.
- Os Fatores de ambiente (EF) passaram a estar relacionado apenas com o esforço; assim, o tamanho passa a depender apenas de requisitos funcionais e não-funcionais.
- A inexistência de padrões aceitos universalmente para a especificação de casos de uso dificulta a comparação entre projetos de diferentes organizações. Portanto, não há como garantir que os valores em UCP ou TUCP estarão medindo a mesma coisa, se os critérios utilizados para a construção de casos de uso forem muito diversificados. Daí a sugestão de especificação do Apêndice A, que deverá ser seguida de uma orientação de uso.

Referências Bibliográficas

OMG. (2003) "Unified Modeling Language Specification", version 1.5. March 2003. Disponível em: <http://www.omg.org>. Acesso em: 17/09/2004.

Anda, B; et al. (2001) "Estimating software development effort based on use cases: experiences from industry", In: International Conference on UML2001, 4. Proceedings. Toronto, Oct. 1 – 5.

ANDA, B. (2002) "Comparing effort estimates based on Use Case Points with expert Estimates", Empirical Assessment in Software Engineering (EASE 2002). Keele, UK, p. 8 – 10.

Boehm, B., (2000) "Software Cost Estimation With COCOMO II", Prentice Hall, New Jersey.

Cockburn, A. (2001) Writing effective: use cases. Addison-Wesley Boston.

Damodaran, M; Washington A. (s.d.) "Estimation Using Use Case Points", Computer Science Program. Texas – Victoria; University of Houston. Acesso em 11/04/2004: http://bfpug.com.br/Artigos/UCP/Damodaran-Estimation_Using_Use_Case_Points.pdf.

Fowler, M;Scott, K. (2005) "UML Essencial - Um Breve Guia para a Linguagem-PAD", Bookman.

Garmus, D., Herron, D. (2000) "Function Point Analysis: Measurement practices for successful software projects", Addison-Wesley: EUA.

Karner, G. (1993) "Metrics for Objectory", Diploma thesis, University of Linköping, Sweden. No. LiTH-IDA-Ex-9344:21.

Kruchten, P. (2001) "The Rational Unified Process: an introduction", Addison-Wesley.

McPhee, C. (1999) "SENG 621: Software process management: software size estimation", University of Calgary. 11p. Acesso em 18/08/04: http://sern.ucalgary.ca/~cmcphee/SENG621/Software_Size_Estimation.html.

Meneses, J. B. (2001) Inspector: Um Processo de Avaliação de Progresso para Projetos de Software. Dissertação de Mestrado da UFPE. Recife

Peters, K. (1999) "Software Project Estimation", Software Productivity Centre Inc. (SPC) in Vancouver, British Columbia, Canada. Disponível em: http://www.spd.ca/downloads/resources/estimates/estbasics.pdf.

Rational Corporation (2003) "DEV112: Principles of Analysis I. Web Courses", Acesso em 23/04/2003: http://www.rational.net.

Ribu, K. (2001) "Estimating Object-Oriented Software Projects with Use Cases", Master of Sciense Thesis, University of Oslo, United States.

Roetzheim, W. H. (2000) "Estimating Internet Development", Software Development Magazine.

ROSS, M. (s.d.) "Size does Matter: Continuous Size Estimating and Tracking", Quantitative Software Management. Acesso em 18/08/04: http://www.qsm.com.

RUP (2003) Rational Software Corporation, Rational Unified Process, Version 2003.06.00.65, CD-ROM, Rational Software, Cupertino, California, 2003.

Schneider, G., Winters, J. (2001) "Applying Use Case: A Practical Guide", 2nd ed. Addison-Wesley.

SEI (2002). "CMMI-SW for Systems Engineering/Software Engineering ,Version 1.1. CMU/SEI-2002-TR-012", Acesso em 27/04/2004: http://www.sei.cmu.edu/publications/documents/02.reports/02tr002.html.

Vasquez, C. E. (2003) "Análise de ponto de função: medição, estimativas e gerenciamento de projetos de software", 1ed. São Paulo: Érica.

Apêndice A – *Modelo de Especificação de Caso de Uso*

1 CASO DE USO

<Nome do caso de uso>

1.1 Descrição

<Descrever o comportamento do caso de uso>

1.2 Atores envolvidos

<Elencar os atores associados ao caso de uso>

1.3 Fluxo de Eventos

1.3.1 Fluxo Básico (*Happy Day)*

<Listar os passos numerados do fluxo básico>
<Passo 1>
<Passo 2>
. . .

1.3.2 Fluxos Alternativos

<Listar os fluxos alternativos (ou de exceção) em relação ao fluxo básico.>
< Fluxo Alternativo 1>
<Listar os passos numerados do fluxo alternativo>
< Fluxo Alternativo 2>
<Lista de passos numerados do fluxo alternativo>
. . .

1.4 Precondições

<Apresentar as precondições que devem ser satisfeitas>

1.5 Pós-condições

<Relacionar as pós-condições que devem ser satisfeitas>

1.6 Relacionamento com Outros Casos de Uso

<Exibir os relacionamentos (dependência de dados) com outros casos de uso quando se aplicar>

1.7 Pontos de Extensão

<Apresentar os pontos de extensões do caso de uso>

1.8 Requisitos Especiais

<Listar os requisitos especiais (não-funcionais) específicos do caso de uso. Exemplo: tempo de resposta>

1.9 Regras de Negócio

<Listar as regras de negócio que dizem respeito ao caso de uso: Regra 01, Regra 02, ... Regra *n*>

Regra 01:
Nome da Regra: *<Incluir o nome da regra>*
Descrição da Regra:
<Descrever as regras de negócio do caso de uso>

2 ATORES

2.1 *<Nome do Ator>*

2.2 Descrição do papel

<Descrever o papel do ator>