Easy Effort Estimation – 3E: Método para Estimativa de Projetos de Software para Pequenas Empresas

Starch Souza[1,3], Ana Roullier[2], Silvio Lemos Meira[1], Jeane Santos[1,3], Tayanna Sotero[1,3], Heron Aguiar[1,4]

[1]Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Caixa Postal 7851 – 50.732-970 – Recife – PE – Brazil

[2]DEINFO – Universidade Federal Rural de Pernambuco (UFRPE)

CEP: 52171-900 – Recife – PE - Brazil

[3]Grupo TCI – Tecnologia do Conhecimento e da Informação

CEP: 50100-030 - Recife – PE – Brazil

[4]SWQualiti Consultoria e Sistemas LTDA

Rua Padre Cabral, 63 Sala 301 – Recife – PE - Brazil

{sms2, srlm, jmss2, tcs4, hva}@ cin.ufpe.br, anarouiller@gmail.com

Abstract*. This paper presents an effort estimation method for software projects destinated to small and medium-sized business companies. The 3E method uses the organization historical database and best practices of existing methods in literature, like Function Points, Use Case Points, CoCoMo II and Wideband Delphi. This paper presents a comparative study between the existing methods and a case study with its achieved results to the present moment, having low deviation between planned and actual effort, compared to Use Case Points*

Resumo. *Este artigo apresenta um método de estimativa de esforço de projetos de software destinado a empresas de pequeno e médio porte. O método 3E utiliza a base em dados históricos da organização e melhores práticas de métodos existentes na literatura como Pontos de Função, Pontos de Caso de Uso, CoCoMo II e Wildband Delphi. Neste artigo é apresentado um estudo comparativo entre os métodos existentes e foi realizado um estudo de caso com os resultados alcançados até o momento apresentando baixo desvio entre o esforço planejado e realizado, quando comparado com Pontos de Caso de Uso.*

1. Introdução

Realizar estimativas de esforço para o desenvolvimento de um projeto de software envolve o uso de boas práticas. Mehwish Nasir [Nasir, 2006] recomenda que esta estimativa utilize diversos métodos e que sejam considerados de preferência dados coletados de projetos passados, além disso, também ressalta a importância de estar reestimando este esforço durante o progresso do projeto.

Segundo Petri Vesterinen [Vesterinen, 1999], estimar esforço é um tema que deve ser considerado no desenvolvimento de projetos de software. Diversos modelos e métodos podem ser usados de forma satisfatória, contudo, um modelo que é adequado para uma determinada organização, geralmente, pode não ser para outra.

Uma das formas de obter as estimativas de esforço é por meio da determinação da medida funcional de tamanho de software[1] e posteriormente a derivação para horas, considerando o perfil da equipe que irá desenvolver o produto de software. A medida funcional de tamanho de software é bastante utilizada e trata-se de uma medida externa, pois mede somente aquilo que é percebido pelos usuários do produto de software, independentemente da forma de implementação escolhida.

Empresas de desenvolvimento de software de pequeno e médio porte geralmente não possuem especialistas com conhecimento suficiente nas técnicas de estimativas apresentadas pela literatura.

O objetivo principal deste trabalho é apresentar um método de estimativa de esforço que leva em consideração: (1) as informações de projetos passados e (2) as boas práticas de modelos e técnicas de estimativa publicadas na literatura que determinam a medida funcional de tamanho de software. O método também considera as características de empresas de pequeno e médio porte, que segundo o Ministério da Ciência e Tecnologia [MCT] geralmente não possuem especialistas em estimativas.

Além desta seção introdutória, este artigo está organizado em mais 4 seções. A Seção 2 descreve e compara alguns dos principais métodos de estimativas de tamanho: Análise de Pontos por Função, Pontos de Caso de Uso, CoCoMo II e Wideband Delphi. A Seção 3 apresenta o método proposto por este trabalho, cujos resultados de sua aplicação por meio da realização de um experimento está descrito na Seção 4 (incluindo uma comparação com a técnica de Pontos de Caso de Uso). Por fim, a Seção 5 apresenta algumas conclusões do trabalho realizado.

2. Técnicas de estimativas

O planejamento e controle efetivos de projetos não são possíveis sem uma estimativa confiável, porém, de uma forma geral a indústria de software não utiliza estimativas de forma apropriada. Isto torna a estimativa de software efetiva uma das atividades mais desafiadoras de projetos de software [Peters, 1999]. Nesta seção serão apresentadas as principais técnicas publicadas.

2.1 Análise de Pontos por Função

A técnica de Análise de Pontos por Função (FPA - Function Point Analysis) é uma técnica para estimar tamanho de um projeto de software através da quantidade de Pontos de Função de um software. Pontos de função é uma medida funcional de tamanho de software, introduzida em 1979 por Alan Albrecht da IBM, [Albrecht, 1979] e cuja contagem é realizada com base em cinco tipos de componentes de software: arquivos internos, arquivos externos, entradas, saídas e consultas. A contagem de Pontos de Função envolve a identificação e classificação dos componentes e exige

[1] Medida funcional de tamanho de software é um conceito definido pelo padrão ISO/IEC 14143-1:1998 e refere-se à medição do tamanho do software considerando-se apenas a funcionalidade solicitada e recebida pelos respectivos usuários

conhecimento especializado. Após a identificação destes, é atribuída a complexidade baixa, média ou alta. Com base na complexidade e em tabelas específicas, a cada componente é atribuída uma quantidade de pontos de função, a soma das contribuições de todos os componentes resulta na quantidade de pontos de função não ajustados. Estes pontos podem ser ajustados por meio da aplicação de 14 fatores de ajustes, conforme sugere a técnica. Estes fatores alteram a contagem do valor inicial não ajustado.

O IFPUG- International Function Point Users Group (IFPUG) é uma organização internacional sem fins lucrativos sediada nos Estados Unidos da América que regulamenta a técnica de FPA.

Os Pontos de Função não medem diretamente o esforço de desenvolvimento. Outros fatores podem ser considerados pela técnica como: a confiabilidade desejada para o software, a metodologia de desenvolvimento utilizada, o nível de testes requerido e a complexidade dos algoritmos.

2.2 Pontos de Caso de Uso

A técnica de estimativa baseada em Pontos de Caso de Uso (UCP - Use Case Points) trata-se de método para estimar o tamanho de projetos de software orientado a objetos e foi proposto por Gustav Karner [Karner,1993]. Esta técnica é uma adaptação do método de Análise de Pontos por Função que explora o diagrama e descrição de casos de uso. Este método foi criado para permitir que seja possível estimar o tamanho do sistema ainda em fases iniciais do projeto de software utilizando-se os artefatos gerados.

O processo de estimativa de UCP é composto por seis etapas: contar os atores e atribuir o grau de complexidade, contar os casos de uso e atribuir o grau de complexidade, calcular os pontos de caso de uso não ajustados, determinar o fator de complexidade técnica, determinar a eficiência do fator ambiental e calcular os pontos de caso de uso ajustados.

Os UCP dão a estimativa do tamanho para desenvolver o sistema, o qual pode ser mapeado para homens/hora para realizar as várias fases do ciclo de vida de desenvolvimento do software. Gustav Karner [Karner,1993] propôs a produtividade de 20 homens/hora por UCP.

.2.3 CoCoMo II

O Constructive Cost Mode II, (CoCoMo II), segundo Barry Boehm [Boehm, 2000], é um método que busca medir esforço, prazo, tamanho de equipe e custo necessário para o desenvolvimento do software, desde que se tenha a dimensão do mesmo, por meio de um método de estimativa de tamanho de software, como pontos de função.

O método é uma evolução do CoCoMo 81 que apresentava deficiências como a idade dos projetos que embasaram, a incapacidade de lidar com ciclos iterativos e com a utilização de componentes Commercial-Off-The-Shelf (COTS) - são componentes de software prontos para utilização, de terceiros, comercialmente disponíveis, e que se tornam importantes durante a criação do novo software, devendo ser utilizados preferencialmente durante a fase de pré-desenvolvimento do produto (software) [Boehm, 2000].

As estimativas são calculadas a partir do número de linhas de instruções do código-fonte, pontos de caso de uso ou ainda pontos de função; e sobre os fatores de

escala (grau de precedência, flexibilidade de desenvolvimento, arquitetura/resolução do risco, coesão da equipe e maturidade do processo) e custo (divididas entre os contextos de produto, plataforma, pessoal e projeto).

2.4 Wideband Delphi

O método de estimativa Wideband Delphi é uma técnica para estimativa de esforço baseada em consenso de especialistas, diferente das técnicas apresentadas nas seções anteriores. Foi desenvolvida por Barry Boehm [Boehm, 1981] como uma melhoria no modelo Standard Delphi [Helmer, 1966].

O método consiste em 6 etapas que são: planejamento, reunião inicial, preparação da estimativa individual, reunião de estimativa, apresentação de divergências das estimativas e revisão dos resultados.

Este método deve ser aplicado para estimar esforço dos itens de uma lista de atividades de um projeto ou de uma iteração [Boehm, 1981]. A etapa de planejamento consiste em definir e gerar o escopo do projeto e/ou iteração. A etapa seguinte, reunião inicial, tem como principal objetivo apresentar a todos os envolvidos o projeto e o escopo, além de uma explicação geral sobre o método de estimativa.

Na etapa de preparação da estimativa individual, cada colaborador realiza a estimativa individual, para nas etapas seguintes serem apresentadas, revisadas e reunidas às demais estimativas, sendo obtido o consenso para a estimativa do esforço para a execução das atividades.

2.5 Estudo comparativo

Os critérios de avaliação da cobertura de atividades e fatores apresentados na Tabela 1 foram extraídos de um estudo comparativo entre modelos de estimativa de custo [Chulani, 1998].

Os mesmos critérios foram utilizados para comparar os métodos apresentados anteriormente como Pontos de Caso de Uso, Pontos de Função, CoCoMo II e Wideband Delphi. O estudo comparativo é apresentado na tabela a seguir:

Tabela 1. Critérios para avaliação de métodos de estimativas

Grupo	Critérios	FPA	UCP	CoCoMo II	Wideband Delphi
Atributos de tamanho	Instruções de código	-	-	✓	-
	Pontos de Função	✓	-	✓	-
	Experiências anteriores	✓	✓	✓	-
	Experiência de especialistas	-	-	-	✓
	Pontos de caso de uso	-	✓	✓	-
Atributos do programa	Tipo/Domínio	✓	✓	✓	-
	Complexidade	✓	✓	✓	-
	Linguagem	-	✓	✓	-
	Reuso	✓	✓	✓	-
	Confiabilidade requerida	✓	✓	✓	-
Atributos do computador	Restrições de recursos	✓	✓	✓	-
	Volatilidade da plataforma	✓	✓	✓	-
Atributos dos recursos	Capacidade	-	✓	✓	-
	Continuidade	-	✓	✓	-

humanos	Experiência	-	✓	✓	-
Atributos do projeto	Técnicas e Ferramentas	-	✓	✓	-
	Restrições do cronograma	-	-	✓	-
	Funcionalidades	✓	✓	-	-
	Maturidade do processo	-	✓	✓	-
	Coesão da equipe	-	✓	✓	-
	Itens de Segurança	-	✓	✓	-
	Desenvolvimento multisite	✓	✓	✓	-
Cobertura das atividades	Iniciação	-	✓	✓	-
	Elaboração	-	✓	✓	-
	Construção	-	✓	✓	-
	Transição e Manutenção	-	✓	✓	-

Desta tabela conclui-se que existem pontos fortes e fracos nos métodos apresentados, sendo possível um estudo detalhado de combinação dos critérios mais adequados para a apresentação de um método de estimativa em conformidade com a realidade de uma empresa de pequeno e médio porte.

O método de estimativa proposto por este trabalho será apresentado em detalhes na seção a seguir e irá se adequar aos critérios de experiências anteriores, experiência de especialistas, tipo e domínio do projeto, experiência da equipe envolvida e maturidade do processo.

3. Easy Effort Estimation

O método apresentado a seguir, denominado Easy Effort Estimation – 3E, foi desenvolvido para atender às necessidades de empresas de pequeno e médio porte que atuam na área de desenvolvimento de software.

O método 3E está baseado nas técnicas de estimativas existentes no mercado, já descritas na seção anterior, e estimativa por analogia que compara o projeto a ser estimado com projetos concluídos do mesmo tipo, por meio dos dados históricos [Pandian, 2004]. As principais características do deste método são:

- Utiliza dados históricos de projetos já concluídos;
- É aplicado para estimar esforço do desenvolvimento de um software em um estágio muito inicial do processo de desenvolvimento, quando o projeto possui somente requisitos funcionais em um nível mais abstrato;
- Tem como base os métodos FPA e UCP;

As entradas, etapas e saídas que podem ser visualizadas na figura a seguir representa de forma resumida o método 3E:

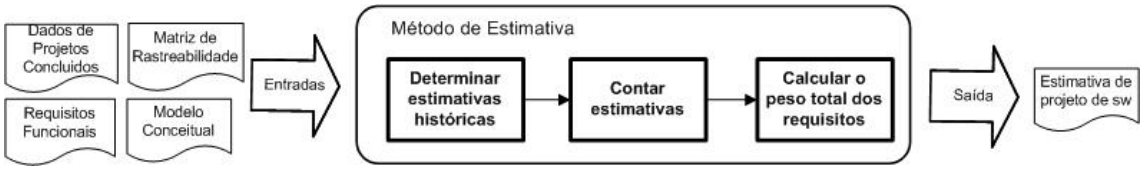

Figura 1. Método 3E

As entradas são obrigatórias para utilização correta do método 3E e são compostas por:

1. **Requisitos funcionais**: após a produção do documento de especificação de requisitos, é escolhido um conjunto de requisitos funcionais que irá compor o escopo do projeto a ser estimado;
2. **Dados dos projetos concluídos**: são os dados históricos de projetos de mesmo tipo já concluídos na organização serão utilizados como ponto de partida da estimativa, e pode ainda ser calibrado por especialistas experientes;
3. **Matriz de rastreabilidade**: com o documento de requisitos é elaborada a matriz de rastreabilidade, que oferece a visualização de dependência dos requisitos funcionais e casos de uso;
4. **Modelo conceitual ou entidade-relacionamento**: o modelo conceitual pode ser representado por um conjunto de tabelas e atributos relacionados. Este modelo é amplamente divulgado na literatura e facilmente construído a partir de informações abstratas, sendo mais um atrativo para a utilização do método 3E.

A saída do método 3E é o resultado da execução de todos os passos, sendo, portanto, a estimativa de esforço do projeto de software.

O detalhamento dos passos da técnica está detalhado nas subseções abaixo.

3.1 Determinar estimativas históricas

Durante o desenvolvimento de projetos de software é formada a base de informações históricas dos projetos. Sugere-se que sejam registrados os ***esforços realizados*** associados aos ***requisitos funcionais***, contudo, é possível apontar horas por atividades se existir a rastreabilidade entre a atividade e o requisito funcional associado.

Este apontamento de horas será utilizado posteriormente como base para estimar projetos similares. Desta forma, as estimativas históricas para os projetos são organizadas conforme apresentado na Tabela 2.

Para determinar as estimativas históricas que serão utilizadas para o projeto, os requisitos são agrupados, de forma subjetiva, por especialistas da organização que estão envolvidos com o projeto de acordo com a complexidade (baixa, média, alta e altíssima). As suas estimativas de esforço são coletadas e o esforço médio em horas é atribuído a cada categoria de complexidade. O esforço pode ser calibrado por especialistas experientes.

Para categorização dos requisitos funcionais, o método apresenta o peso total de referência embasado na combinação dos pesos referentes às características: interface, transações, entidades, atributos e dependências. Inicialmente os valores do peso total, apresentados na Tabela 2, foram aplicados com o objetivo de validar a técnica e poderão sofrer calibração.

O esforço, em horas, está relacionado à complexidade e é obtido a partir de base de dados históricos da organização para o tipo do projeto. Além disso, o esforço poderá ser reavaliado por meio de revisão de especialistas.

Desta forma, as estimativas de esforço baseadas na experiência da organização e no tipo de projeto são determinadas e classificadas em complexidade baixa, média, alta e altíssima. Na tabela a seguir é apresentado um exemplo da determinação das estimativas históricas.

Tabela 2. Estimativa de esforço de requisitos funcionais baseada na estimativa histórica de um projeto

	Baixa	Média	Alta	Altíssima
Peso Total	10	20	30	40
Esforço (h)	16	25	38	62

3.2 Contar Estimativas

Para contar as estimativas de tamanho de software, foram utilizadas características de interfaces, transações, entidades, atributos e dependências. Cada requisito do sistema deve ser classificado segundo tais características, descritas a seguir:

a) Interface: Interface consiste no conjunto de telas que o software utiliza para executar o requisito de forma adequada. A interface pode ser classificada em simples, média, difícil ou complexa, conforme o critério apresentado:

- Simples: quando possui apenas componentes do tipo campos de texto e botões;
- Média: exibe campos de texto, botões, combos e listas de opções;
- Difícil: é aquela que possui campos de texto, botões, combos, listas e validações;
- Complexa: se apresenta campos de texto, botões, combos, listas, validações, estruturas de árvore, integração com subsistemas, entre outros componentes complexos.

Na Tabela 3 são exibidos os pesos definidos para a característica "interface".

Tabela 3. Pesos para a característica Interface

Interface	Simples	Média	Difícil	Complexa
Peso	1	2	3	4

b) Transações: Transações dizem respeito a casos de uso derivados de um requisito funcional a fim de alcançar os objetivos do mesmo, ou seja, requisitos menores contidos no requisito que está sendo estimado.

Na Tabela 4 são exibidos os pesos definidos para a característica "transações".

Tabela 4. Pesos para a característica Transações

Transações	Até 3	Entre 4 e 7	Acima de 7
Peso	5	10	15

c) Entidades: As entidades aqui citadas são representadas por meio de um modelo conceitual ou entidade-relacionamento. O modelo conceitual consiste em uma das entradas para o método de estimativa, usado para a obtenção do quantitativo de entidades relacionadas aos requisitos funcionais.

Na tabela a seguir são apresentados os pesos definidos para a característica "entidades".

Tabela 5. Pesos para a característica Entidades

Entidades	Até 1	Entre 2 e 5	Acima de 5
Peso	5	10	15

d) Atributos: São considerados atributos todas as características que descrevem uma entidade a ser utilizada por 1 ou mais requisitos funcionais.

Na Tabela 6 são exibidos os pesos definidos para a característica "atributos".

Tabela 6. Pesos para a característica Atributos

Atributos	Até 3	Entre 4 e 10
Peso	5	10

e) Dependências: Os requisitos funcionais podem ter um ou mais requisitos funcionais associados. E cada requisito funcional é detalhado em um ou mais casos de uso. Dependência representa a relação entre os requisitos e os casos de uso dos requisitos relacionados. A quantidade de dependências é obtida a partir da matriz de rastreabilidade de requisitos funcionais.

Na Tabela 7 podem ser visualizados os pesos definidos para a característica "dependências".

Tabela 7. Pesos para a característica Dependências

Dependências	Ausente	Até 2	Entre 3 e 5	Acima de 5
Peso	0	3	6	9

3.3 Calcular o peso total dos requisitos

Após a contagem das estimativas de tamanho de cada requisito funcional, o peso total é a soma do peso individual associado à interface, transações, entidades, atributos e dependências, conforme é apresentado na Tabela 8.

A partir do peso total, cada requisito tem a complexidade determinada e associada a um esforço em horas necessárias ao seu desenvolvimento.

Na Tabela 8 é apresentado um exemplo da distribuição dos pesos para cada característica resultando na complexidade do desenvolvimento do requisito funcional. A partir da complexidade, o esforço em horas para o desenvolvimento do requisito é calculado. O resultado da estimativa será o somatório dos esforços de desenvolvimento de todos os requisitos funcionais, conforme fórmula apresentada na Figura 2. É necessário destacar que os requisitos precisam estar identificados e numerados seqüencialmente para que a fórmula possa ser aplicada.

$$\text{Estimativa} = \sum_{i=1}^{n} \text{Esforço}_i \left(\text{pesoInterface}_i + \text{pesoTransações}_i + \text{pesoEntidades}_i + \text{pesoAtributos}_i + \text{pesoDependência}_i \right)$$

i :: é o código do requisito funcional

Figura 2. Fórmula da estimativa

Abaixo é apresentada uma visão parcial da planilha, que faz uso da fórmula apresentada na Figura 2, com uma lista requisitos funcionais, cálculo da complexidade dos requisitos e atribuição de esforço em horas.

Tabela 8. Visão parcial da planilha de estimativa de esforço de requisitos funcionais de um projeto

Requisito	Interface	Transações	Entidades	Atributos	Dependências	Complexidade	Esforço (hs)
RF-001	Simples	1	2	2	0	Baixa	16
RF-002	Média	5	2	7	1	Alta	38
RF-003	Média	5	1	3	1	Média	25
RF-004	Média	5	2	2	1	Média	25
RF-005	Simples	5	2	2	2	Média	25
RF-006	Média	5	3	3	4	Alta	38
						TOTAL	**167**

O esforço estimado para cada requisito deve ser distribuído de acordo com o processo de software adotado, características do produto em desenvolvimento e da organização.

4. Estudo de Caso

Nesta seção, apresentaremos os resultados obtidos da aplicação do método de estimativas definido versus o método de pontos de casos de uso a dois projetos de uma mesma organização, caracterizados a seguir.

Estes resultados servirão como base para fundamentar os objetivos do presente trabalho e como uma fonte de dados históricos para refinar as estimativas de projetos similares a serem realizados no futuro.

A Empresa na qual o experimento foi realizado trata-se de uma organização de médio porte que desenvolve software por encomenda. Os projetos de software desenvolvidos seguem um processo de desenvolvimento de software aderente ao modelo de capacidade CMMI Nível 2 [Chrissis, 2003], à norma ISO 9001:2000 [ISO 2000] e ao padrão de gerenciamento de projetos PMBOK [PMBOK Guide, 2004] atuando em diversas plataformas de desenvolvimento com projetos de software e manutenção evolutiva de produtos de software.

4.1 Caracterização dos Projetos

Os projetos analisados neste trabalho estão caracterizados na Tabela 9.

Tabela 9. Caracterização dos projetos estimados

Nome do Projeto	Descrição	Equipe envolvida	Tecnologia
Projeto A	Desenvolvimento de um sistema web para controle das atividades de pessoas alocadas em projetos e fornecimento de relatórios gerenciais.	8 membros	JAVA J2EE, SQL Server, Framework proprietário
Projeto B	Projeto de manutenção e reestruturação de um portal de negócios na web com integração entre subsistemas.	10 membros	ASP.NET, SQL Server

Os projetos A e B contemplam o desenvolvimento de aplicações e foram estimados de duas formas:

- Utilizando o método definido no processo de planejamento, antes da especificação dos casos de uso, quando o projeto possuía requisitos funcionais em um nível ainda abstrato;

- Utilizando a técnica de pontos de caso de uso, que foi aplicada após a especificação dos casos de uso, ou seja, mais tardiamente no ciclo de vida do projeto.

A respeito das equipes envolvidas nos projetos, vale destacar que parte das equipes consiste em membros de áreas de apoio ao desenvolvimento (como qualidade, infra-estrutura, gerência de configuração e equipe independente de testes) e, portanto, estes membros são compartilhados entre outros projetos da empresa, sendo o esforço destes considerado parcial para o projeto.

4.2. Determinação de estimativas históricas

A organização em estudo utiliza um sistema, denominado TimeSheet, para registro de atividades de projetos relacionadas aos requisitos funcionais e apontamento de esforço realizado por recurso alocado ao projeto para cada atividade.

A partir dos dados do TimeSheet, a organização formou uma base de dados históricos, categorizados por tipo de projeto e complexidade dos requisitos. Na Tabela 10 são exibidos os tipos de projetos realizados pela empresa, similares aos analisados, e o esforço médio para realização de requisitos de complexidade baixa, média, alta e altíssima. Esta tabela foi construída conforme sugestão apresentada na Seção 3 deste artigo.

Tabela 10. Determinação do esforço por categoria de complexidade de requisito para os tipos de projeto da organização

Categoria	Tecnologia	Complexidade			
		Baixa	***Média***	***Alta***	***Altíssima***
JAVA	JAVA J2EE, SQL Server	34	50	87	120
ASP.NET	ASP.NET, SQL Server	52	87	186	260

Esta base histórica foi construída a partir da análise de 12 projetos de software realizados pela organização, sendo 7 destes projetos enquadrados na categoria JAVA e 5 na categoria ASP.NET.

A contagem de estimativas foi realizada e os dados de complexidade e esforço associado aos requisitos exibidos nas Tabelas 11 e 12 foram obtidos em relação aos projetos A e B, respectivamente.

Tabela 11. Contagem de estimativas para o Projeto A

Requisito	Interface	Transações	Entidades	Atributos	Dependências	Complexidade	Esforço (hs)
RF001	Simples	3	2	2	0	Baixa	34
RF002	Média	4	2	7	1	Alta	87
RF003	Média	4	1	3	1	Média	50
RF004	Média	4	2	2	1	Média	50
RF005	Simples	4	2	2	2	Média	50
RF006	Média	7	3	3	4	Alta	87
RF007	Média	4	2	3	1	Média	50
RF008	Simples	3	2	3	4	Média	50
RF009	Média	3	4	3	5	Média	50
RF010	Média	3	3	2	4	Média	50
RF011	Média	5	2	3	0	Média	50
RF012	Simples	4	10	50	10	Altíssima	120

RF013	Média	3	2	4	2	Média	50
						TOTAL	778

Tabela 12. Contagem de estimativas para o Projeto B

Requisito	Interface	Transações	Entidades	Atributos	Dependências	Complexidade	Esforço (hs)
RF001	Complexa	**3**	6	11	3	Alta	186
RF002	Simples	3	3	7	0	Média	87
RF003	Difícil	5	7	20	5	Altíssima	260
RF004	Simples	1	2	3	1	Média	87
RF005	Simples	1	1	2	0	Baixa	52
RF006	Média	1	1	2	0	Baixa	52
RF007	Média	1	1	2	0	Baixa	52
RF008	Complexa	1	6	12	3	Alta	186
RF009	Média	1	3	12	2	Média	87
RF010	Difícil	2	4	7	3	Média	87
						TOTAL	**1136**

4.3. Resultados obtidos

Para a aplicação do método de UCP o fator de produtividade, que representa a produtividade média da equipe, adotado para ambos os projetos foi: 20 homem-hora / UCP, por se tratar de sugestão de Gustav Karner [Karner, 1993].

Os resultados das estimativas de esforço em cada um dos projetos descritos são apresentados na Tabela 13.

Tabela 13. Resultados da aplicação de pontos de casos de uso e método proposto

Nome do Projeto	*Percentual de conclusão*	Pontos de Casos de Uso		Método Definido	
		Planejado	***Desvio***	***Planejado***	***Desvio***
Projeto A	**30% concluído** 248 Homem/Hora	1776,43 (30% = 532,8)	114,83% de acréscimo	778 (30% = 233,4)	5,88% de redução
Projeto B	**100% concluído** 1017,5 Homem/Hora	1239,84	17,93% de acréscimo	1136,00	10,43% de acréscimo

As informações sobre esforço planejado e realizado são apresentadas em Homem/Hora. Os desvios são calculados da seguinte forma: (Realizado - Planejado) / Planejado, onde é indicado o percentual de redução ou acréscimo em relação ao esforço planejado.

O projeto A ainda está em andamento, portanto os dados de esforço total realizado não foram ainda coletados. Este projeto encontra-se na fase de codificação e foi realizado 30% do cronograma estimado do projeto e consumidas 248 Homem/Hora, valor próximo ao estimado com método definido: 233,4 Homem/Hora. Considerando o esforço estimado com a técnica de pontos de casos de uso 30% do projeto equivaleria a 532,8 Homem/Hora.

5. Conclusão

Muitos métodos de estimativas, modelos e metodologias existem e são aplicáveis em diferentes naturezas de projetos, contudo, nenhuma das técnicas garante 100% de acuracidade [Nasir, 2006].

O método de estimativa proposto por este trabalho se encontra em fase de institucionalização e já apresenta, como visto no estudo de caso, um baixo desvio entre o esforço planejado e realizado se comparado com a técnica de estimativa baseada em UCP. O baixo desvio pode ser resultado da utilização da base de dados históricos de projetos similares dentro da organização e ainda o uso das melhores práticas dos métodos apresentados na Tabela 1.

Embora a construção de uma base de dados histórica que reflita o desempenho da organização de software seja de grande valia para a aplicação do método, a não existência desta base não inviabiliza a aplicação. A organização que não possuir uma base histórica pode fazer uso de dados de produtividade externos ou realizar reuniões de *brainstorming* entre seus especialistas (pode ser utilizado o método de Wideband Delphi) para definição de estimativas iniciais a respeito de seus projetos e a partir daí construir sua própria base.

Outro fator a ser destacado diz respeito às entradas para o método. O método deve receber como entradas, além dos requisitos do sistema, um modelo E-R também conhecido como modelo conceitual, a matriz de rastreabilidade que associa os requisitos do projeto e os dados a respeito de sua interface gráfica. Estas entradas contêm as informações base para classificação das características do método, a indisponibilidade destas terá impacto na precisão das estimativas.

Contudo, melhorias serão propostas de acordo com o amadurecimento e uso do método de estimativa, sendo então possível, a sua utilização em outras organizações que ainda não fazem uso de um método de estimativa.

Existem alguns trabalhos que poderão ser feitos no futuro, dentre eles podemos destacar a inserção de fatores de ajuste ao método, pois hoje cada requisito recebe um determinado valor exato em homem-hora de acordo com sua complexidade, não sendo possível a atribuição de um valor intermediário. Outro trabalho a ser destacado consiste na distribuição do esforço estimado dentre as fases de desenvolvimento de acordo com o processo de desenvolvimento de software implantado e distribuição deste esforço entre os perfis envolvidos no processo.

6. Referências

Karner, Gustav (1993) “Resource Estimation for Objectory Projects”. In: International Conference on Software Engeneering, St. Louis, MO, USA. Session: Empirical Software Engeneering, pp: 303 – 311.

Albrecht A. J. (1979) “Measuring application development productivity”. In: Proceedings of Joint SHARE, GUIDE, and IBM Application Development Symposium, pp. 83-92, Monterey, Calif., USA.

Boehm, Barry et al. (1980); “Software Cost Estimation With COCOMO II”. Prentice Hall, 2000, pp.544.

Pressman, Roger S. (1995); “Engenharia de Software”. São Paulo, Atlas 1995, pp 1056.

Medeiros,Ernani (2004). “Desenvolvendo Software com UML 2.0”. São Paulo, Makron Books, 2004.

Macedo, Marcus V. L. R. (2003) “Uma proposta de aplicação da métrica de pontos de função em aplicações de dispositivos portáteis” Dissertação de mestrado, UNICAMP, 2003.

Chulani S., Boehm B., Abts C. (1998) “Software Development Cost Estimation Approaches – A Survey”. Qualifying Exam report in partial fulfillment of requirements of the Ph.D. program of the Computer Science department at USC.

Chrissis M. B., Konrad M., Shrum S. (2003) “CMMI: Guidelines for Process Integration and Product Improvement”, Addison Wesley, 2003.

ISO/IEC TR 9001 (2000). “Sistemas de Gestão da Qualidade - Requisitos”.

American Nacional Standard (2004). “A Guide to the Project Management Body of Knowledge”. Third Edition.

Nasir M. (2006) “A Survey of Software Estimation Techniques and Project Planning Practices”, In: Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2006. SNPD 2006. Seventh ACIS International Conference on, pp: 305-310.

IFPUG – International Function Point Users Group – www.ifpug.org, acessado dia 15/03/2007.

MCT – Ministério da Ciência e Tecnologia do Brasil – www.mct.gov.br, acessado dia 15/03/2007.

Peters K. (1999) “Software Project Estimation” Software Productivity Center Inc.

Pandian C.(2004) “Software Metrics a Guide do Planning, Analysis and Aplication” In: ACM SIGSOFT Software Engineering Notes, volume 5, Issue 5, pp.: 38-39.