Software para Calcular o Prazo de Desenvolvimento de Projetos de Software Utilizando FPA e Lógica Fuzzy

Beatriz Terezinha Borsoi[1], Kathya Silvia Kolazzos Linares[1], Rúbia Elisa de Oliveira Schultz Ascari[1], Luiz Fernando Toscan[1], Matheus Magnusson Bolo[1], Elisa Beatriz Silvério[1], Rafael André Ruas[1]

[1] Universidade Tecnológica Federal do Paraná (UTFPR) - Campus Pato Branco
Grupo de Estudos e Pesquisa em Tecnologias de Informação e Comunicação
Via do Conhecimento – Km 01. Bairro Fraron
Caixa Postal 571. CEP 85.503-390 – Pato Branco – PR

beatriz@utfpr.edu.br,kathya@utfpr.edu.br,rubia@utfpr.edu.br,
luiz.ziulfer@gmail.com, matheus_magnusson_2006@hotmail.com
el.btr@hotmail.com,raphaell-andre@hotmail.com

Resumo. *A estimativa de prazo de desenvolvimento de software possui aplicações diretas no planejamento do projeto de um software, seja para alocar recursos, planejar as atividades ou auxiliar na elaboração de contrato com o cliente. Assim, quanto mais precisa ela for, mais efetivos podem ser os resultados da sua aplicação. Contudo, definir uma estimativa precisa é difícil pela quantidade de fatores que podem estar envolvidos. Para minimizar essa dificuldade, este trabalho propõe implementação de um software para realizar essa estimativa com base nos requisitos do sistema e fatores de ajuste. O ajuste de prazo é obtido por meio do uso de lógica fuzzy que combina esses fatores em regras. Essas regras, em conjunto, determinam o percentual de ajuste ao prazo calculado tendo como base os requisitos do software.*

1. Introdução

Ter definido o prazo que será necessário para desenvolver um projeto de software é importante porque o mesmo pode ser utilizado para auxiliar na alocação de recursos, de pessoas, no planejamento e em definições de contrato com o cliente. Contudo, definir esse prazo, ainda que de forma estimada, é uma atividade bastante complexa.

A complexidade decorre da quantidade de fatores envolvidos. Alguns desses fatores estão relacionados aos requisitos do sistema, como, o entendimento do processo de negócio, a completeza na definição dos mesmos, as mudanças que o cliente faz durante e após a implementação do sistema e as limitações das tecnologias para implementá-los. E há, ainda, os fatores que são de outras naturezas, como, o conhecimento e experiência da equipe ou a possibilidade de ocorrência de mudanças na equipe.

A diversidade desses fatores dificulta a realização de estimativas por meio de sistemas computacionais, ainda que haja diversas técnicas que possam ser empregadas. Algumas dessas técnicas (abrangendo denominações como: procedimento, padrão, modelo,

metodologia) são utilizados para estimar prazo, custos e outros tendo como base o esforço necessário para desenvolver o software. Os custos também podem ser determinados a partir do prazo necessário para implementação.

A análise de pontos de função (APF) é uma técnica que pode ser utilizada no início do processo de implementação, baseada nos requisitos que são definidos pela equipe de desenvolvimento a partir dos interesses do usuário [1]. Essa técnica tem como base funções relacionadas a dados e transações. Além de 14 fatores de ajustes.

Neste trabalho é apresentado um sistema desenvolvido para o cálculo de estimativa de software utilizando os fatores definidores de prazo constantes em [2], baseados na APF, e fatores de ajuste de prazo que empregam lógica *fuzzy*. Dentre os fatores de ajuste estão a experiência e o conhecimento da equipe.

Este texto está organizado em seções e esta é a primeira e apresenta o contexto no qual o sistema desenvolvido se insere. A Seção 2 contém o referencial teórico sobre estimativa de software e na seção 3 estão conceitos de lógica *fuzzy*. Na seção 4 estão trabalhos relacionados. A apresentação do sistema desenvolvido está na Seção 5.

2. Estimativa de Software

Estimativas em projetos de desenvolvimento de software visam definir, basicamente a definir tempo, esforço e recursos. O tempo se refere ao prazo que é necessário para realizar as atividades. Os recursos estão relacionados ao valor financeiro, a quantidade de pessoas, equipamentos, tecnologias e outros para desenvolver o projeto. O esforço volta-se para o trabalho que é necessário para realizar as atividades. Existem diversas técnicas de estimativas. Nasir [3], citando Agarwal et al. [4] e Boehm [5], as agrupam em abordagens paramétricas e heurísticas:

a) Abordagens paramétricas – são modelos algorítmicos, como, por exemplo: linhas de código [6], pontos de função [7], [1], modelo de custo construtivo (COCOMO), modelo de estimativa de Putnam, pontos de particularidade, ciência do software de Halstead, número ciclomático de McCabe, casos de uso para sistemas orientados a objetos [8] e pontos de função para software em linguagem Java [9].

b) Abordagens heurísticas – incluem julgamento especialista ou parecer técnico com base na experiência pessoal. Delphi e Wideband-Delphi são duas dessas técnicas. A analogia é utilizada a partir do histórico de projetos com características semelhantes.

As estimativas são, geralmente, realizadas com base nas características do sistema. Contudo outros fatores podem ser considerados. A *API Points* [10], por exemplo, que é um método para estimar tamanho e esforço para desenvolvimento de uma API (*Application Programming Interface*), determina o tamanho do projeto com base na identificação das restrições e na escolha de percentual de complexidade.

Análise de Pontos de Função é uma técnica para medir projetos de software de uma perspectiva funcional [1]. APF é independente de métodos, tecnologias, equipamentos e outras condições necessárias para implementar o sistema. Essa técnica visa estabelecer uma medida de tamanho, considerando a funcionalidade implementada do ponto de vista do usuário [11] e pode ser usada para estimar o tamanho do projeto de

desenvolvimento ou de melhoria de sistemas de software. Pontos de função é a unidade de medida de tamanho dessa técnica.

A APF [1] determina os pontos de função não ajustados com base em dois tipos de função: dados e transações. Funções do tipo dado se referem às funcionalidades relacionadas a arquivo lógico interno e de interface externa. Funções do tipo transação representam as funcionalidades que realizam operações no sistema, executando ações (inclusão, alteração, exclusão, consulta) sobre as informações do banco de dados e são: entradas externas, saídas externas e consultas externas.

Dentre as formas de ajustar o prazo obtido a partir de pontos de função estão a NESMA [12], fatores de ajuste do próprio manual de contagem de FPA [1], Roetzheim [13], o modelo COCOMO [14] e Jones [15].

A NESMA propõe três tipos de contagem de pontos de função: a contagem indicativa, contagem estimada e a contagem detalhada. E o fator de ajuste é obtido a partir de 14 características definidas pela FPCPM [1]. Roetzheim [13] cita fatores ambientais de projeto que podem influenciar na definição de estimativas. O modelo COCOMO [14] intermediário considera como fatores de ajuste atributos agrupados em: produto, hardware, pessoal e projeto. Jones [15] inclui como fatores de ajuste: as entidades e os relacionamentos nos arquivos lógicos, tipo de consultas, algoritmos, cronograma, custo, nível de qualidade, plataformas de hardware e de software utilizadas, critérios de segurança e desempenho, treinamento e instalação.

O software desenvolvido como resultado deste trabalho está fundamentado em APF, embora tenham sido definidos outros agrupamentos para a estimativa de tempo. Esse tempo é ajustado por meio de fatores de ajuste. Um processo algorítmico utilizando Lógica Fuzzy é empregado para combinar esses fatores que consideram o contexto do processo de desenvolvimento de software, incluindo a equipe e o ambiente.

3. Lógica Fuzzy

A Lógica Fuzzy é baseada na teoria dos conjuntos *fuzzy*. Nessa lógica, também denominada nebulosa, os valores verdade são expressos linguisticamente (por exemplo: muito, pouco e bastante), em que cada termo linguístico é interpretado como um subconjunto *fuzzy* do intervalo unitário. A teoria dos conjuntos nebulosos foi desenvolvida a partir de 1965 por Lotfi Zadeh, para tratar do aspecto vago da informação [16]. Em decorrência do uso de termos lingüísticos, Zadeh denominou lógica *fuzzy* como equivalente à computação por meio de palavras [17].

Na teoria dos conjuntos *fuzzy*, a função de pertinência determina com que grau um elemento pertence a um conjunto no universo considerado. Na teoria dos conjuntos *fuzzy*, a transição entre pertencer e não pertencer a um determinado conjunto é gradual, ou seja, não é simplesmente pertence (1) e não pertence (0). E sim um valor no intervalo [0, 1], entre 0 e 1.

Uma relação *fuzzy* representa um conjunto *fuzzy* associando cada elemento do produto cartesiano ou par (x,y) por exemplo, a um grau de pertinência definido no intervalo unitário [0; 1].

No software desenvolvido como resultado deste trabalho, as relações fuzzy são

consideradas no formato matricial e utilizadas para ajustar o prazo estimado. Essas relações são compostas pela combinação de fatores que podem alterar o prazo estimado com base nos requisitos do sistema. Cada relação é uma combinação de dois desses fatores. E elas representam o grau de influência dos fatores modificadores em questão ao prazo previamente estimado para o desenvolvimento do projeto.

A ideia básica no uso de lógica *fuzzy* na implementação de sistemas computacionais é modelar as ações ou decisões a partir de conhecimento especialista. No caso da proposta deste trabalho, o conhecimento dos especialistas em estimativa de prazo de desenvolvimento de projetos de software é utilizado como base para o sistema calcular o prazo de projetos novos. Esses projetos podem ser muito diferentes daqueles nos quais os especialistas se basearam para definir os prazos que foram informados para o sistema. Porém, mesmo sendo sistemas distintos é indispensável que eles possam ser categorizados a partir dos mesmos requisitos.

4. Trabalhos Relacionados

Esta seção apresenta trabalhos considerados relacionados a este porque eles se referem ao uso de lógica *fuzzy* com o objetivo de aprimorar os resultados obtidos com técnicas de estimativas no desenvolvimento de software.

Yau e Tsoi [18] combinaram métricas e a Teoria Fuzzy. Lima Júnior, Farias e Belchior [19] propuseram uma forma de classificação contínua e gradual das funcionalidades de um sistema, usando números *fuzzy* para executar o papel da tradicional tabela de classificação. Braz e Silva [20] propõem uma métrica baseada em casos de uso, que usa conceitos da teoria de conjuntos *fuzzy* para criar uma classificação gradual das seções de um caso de uso. Fu et al. [21] propõem um método de análise de pontos de função que combina regras *fuzzy* e redes neurais do tipo *back propagation* com o objetivo de estimar tamanho de software. Chen et al. [22] combinam a teoria *fuzzy* e métodos de interpolação, a fim de apresentar uma análise de pontos de função com interpolação-*fuzzy*.

Um diferencial da proposta deste trabalho em relação aos considerados relacionados, são as regras *fuzzy* aplicadas aos fatores definidores de prazo. Assim, o contexto do ambiente, a estrutura organizacional e características da equipe são considerados na estimativa do prazo que está sendo calculado.

5. Aplicativo Desenvolvido para Estimativa de Prazo de Projetos de Software

O software desenvolvido utiliza os fatores definidores de prazo propostos por Borsoi et al. [2] para o cálculo do prazo não ajustado. Esse prazo é ajustado com base em fatores definidos como de ajuste. Para cada um desses fatores, o usuário indica um valor de influência, tendo como base uma escala. Essa escala determina a faixa de percentual de ajuste que o respectivo fator pode determinar no prazo calculado a partir dos requisitos do sistema. O percentual de ajuste é calculado por regras *fuzzy* definidas pela combinação desses fatores, gerando, assim, uma maneira mais adequada de determinar a influência no ajuste do prazo calculado a partir dos requisitos do sistema.

As Figuras 1 a 6 apresentam as telas do sistema desenvolvido. Nessas telas são indicadas a quantidade de cada um dos respectivos fatores e o tempo. As quantidades e

os respectivos tempos constantes nessas telas se referem a um sistema, com alguns cadastros e geração de nota fiscal eletrônica. É um módulo de um sistema maior. Esse módulo foi utilizado para exemplificar a contagem de prazo de desenvolvimento de sistemas baseado em pontos de função e lógica *fuzzy*.

No campo "Tempo" é indicado o tempo padrão que é utilizado como base para o cálculo. Esse tempo é definido pelo usuário e pode ter como base o histórico da equipe ou da empresa. Esses tempos padrão podem ser armazenados e utilizados posteriormente. Assim, ao invés de indicá-los, o usuário pode carregar um padrão de tempos já cadastrado. O usuário pode alterar os tempos informados como padrão, salvar como um novo padrão ou somente utilizá-los para o cálculo atual.

As quatro primeiras abas (Figura 1) apresentam os agrupamentos de fatores definidores de prazo descritos em Borsoi et al. [2], baseados nos requisitos do sistema. A manutenção de dados pode ser simples e/ou complexa, de acordo com a quantidade de tabelas manipuladas e o número de campos dessas tabelas.

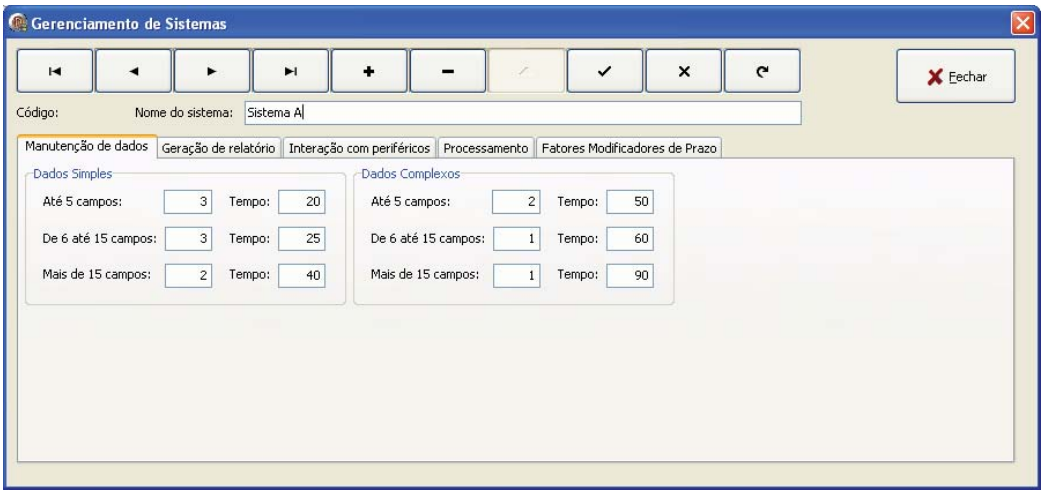

Figura 1. Fator manutenção de dados

A Figura 2 apresenta os requisitos relacionados à geração de relatórios. A complexidade é obtida por agrupamentos do número de tabelas manipuladas e a quantidade de campos dessas tabelas. E, ainda, se é um relatório (para ser impresso em formato *pdf*, por exemplo), ou apenas uma listagem em tela.

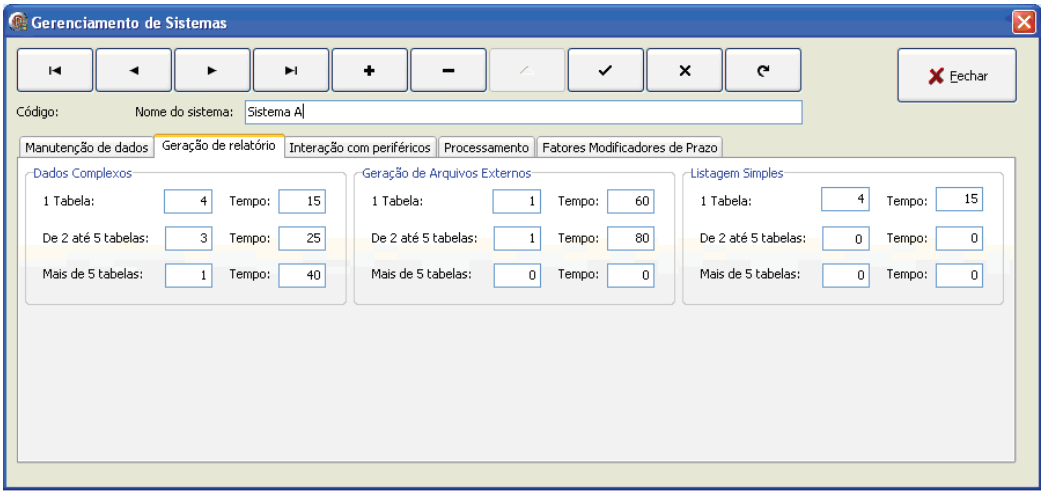

Figura 2. Fator geração de relatório

Na Figura 3 está a tela para a interação com periféricos, como leitores de código de barras, biometria, sensores e atuadores. No módulo do sistema considerado para exemplo não havia esse tipo de interação.

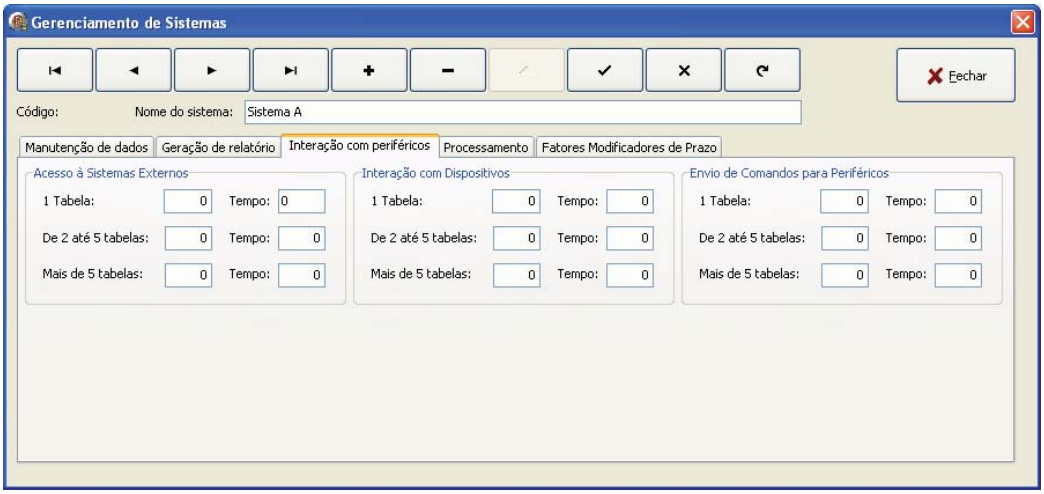

Figura 3. Fator integração com periféricos

A Figura 4 apresenta os requisitos relacionados ao processamento. São vários campos para facilitar uma representação mais real dos requisitos do sistema.

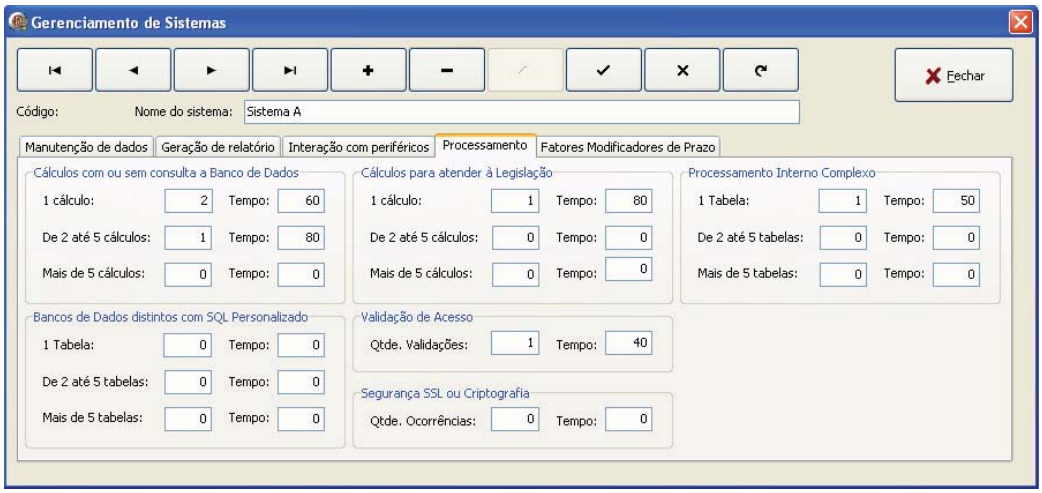

Figura 4. Fator processamento

A Figura 5 apresenta a tela para o usuário informar os fatores modificadores de prazo. O usuário indica um valor em uma escala preestabelecida. Esses fatores são combinados em regras utilizando lógica *fuzzy* e o resultado dessas combinações é o percentual de ajuste que eles definem no prazo obtido com os fatores indicados nas Figuras 1 a 4.

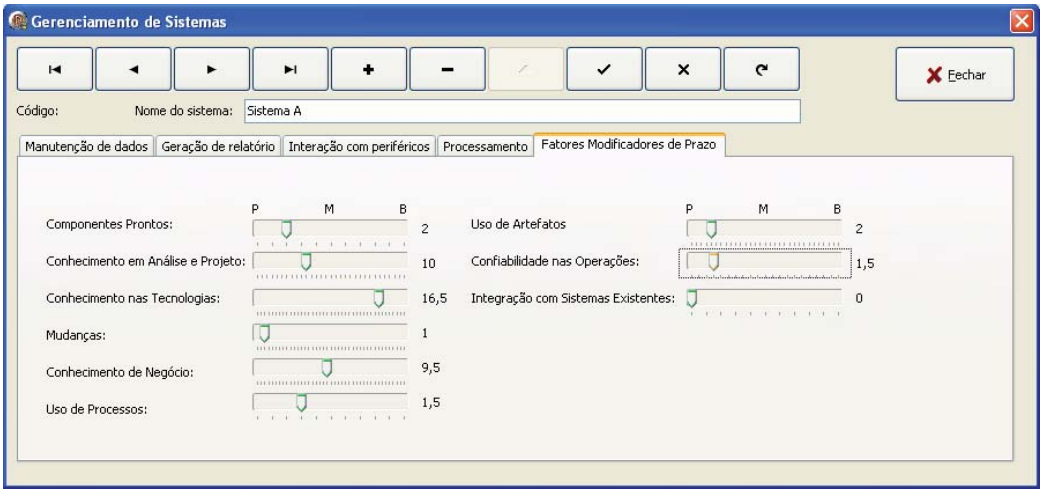

Figura 5. Fatores modificadores

A Figura 6 apresenta o prazo ajustado final obtido. Esse prazo é calculado a partir dos fatores definidos, dos fatores de ajustes e do tempo necessário para a modelagem do sistema. O tempo para modelagem é calculado automaticamente pelo sistema a partir dos fatores definidores e de ajuste.

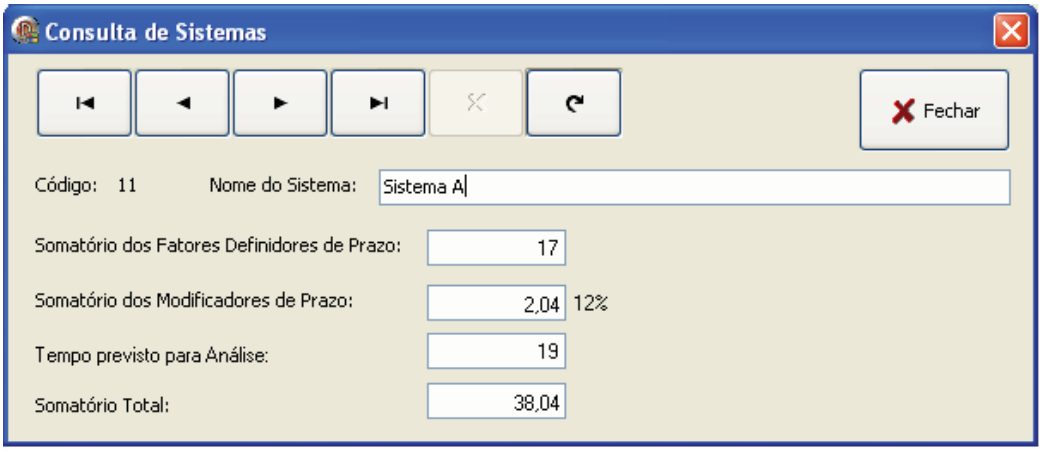

Figura 6. Prazo estimado

O prazo calculado pelo sistema, 38 horas como representado pela Figura 6, foi praticamente o que efetivamente utilizado. Esse módulo de sistema foi desenvolvido em uma semana, com jornada de 8 horas diárias.

6. Conclusão

O sistema implementado como resultado deste trabalho possibilita calcular o prazo estimado para desenvolver um projeto de software, independentemente de aspectos como tecnologias e de métodos empregados. A partir dos requisitos do sistema é obtido o prazo não ajustado. Esse prazo é ajustado por meio de fatores combinados em regras utilizando lógica *fuzzy*. Assim, um ajuste mais fino é obtido, considerando o contexto em que o software será desenvolvido.

Essa é uma primeira versão proposta do sistema. Testes com usuários estão sendo realizados para determinar a necessidade de ajustes nos fatores e também a inclusão de outros fatores. Como implementação futura no sistema está a possibilidade de o usuário indicar a escala de influência de cada fator e indicação de fatores para ajuste do prazo para modelagem (requisito, análise, projeto, arquitetura, testes) do sistema.

A efetividade do sistema proposto dependerá, também, da inserção correta dos valores utilizados como base para cálculo. Assim, mesmo utilizando um sistema computacional é preciso conhecer as habilidades e as dificuldades da equipe, o contexto de trabalho e, principalmente, que os requisitos do software a ser implementado estejam definidos o mais completamente possível. É relativamente difícil que os requisitos estejam definidos desta forma no início do projeto. Sugere-se, então, que o sistema proposto seja utilizado no início do projeto para uma primeira definição de prazo e outras vezes à medida que os requisitos e demais fatores estejam definidos.

Agradecimentos

A Fundação Araucária do Estado do Paraná, Brasil, pelo suporte financeiro ao projeto.

Referências

[1] FPCPM - Function Point Counting Practices Manual, Ver. 4.1.1, 1999, International Function Point Users Group. >http://www.ifpug.org>, maio 2011,15.

[2] Borsoi et al., "Fatores definidores e modificadores em estimativa de prazo de projeto de software", XI Semana de Inovações em Sistemas de Informação (XI SIS2INFO), 2010.

[3] Nasir, M., "A survey of software estimation techniques and project planning practices," ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD'06), 2006, p. 305-310.

[4] Agarwal, R., et al., "Estimating software projects," ACM SIGSOFT Software Engineering Notes, Vol. 26, No. 4, 2001, p. 60-67.

[5] Boehm, B. Abts, C. Chulani, S., "Software development cost estimation approaches–A survey," Software Engineering, Vol. 10, No. 1-4, 2000, p. 177-205.

[6] Park, R. E., "Software size measurement: a framework for counting source statements," 1992 <http://www.sei.cmu.edu/ publications/documents/ 92.reports/92.tr.020.html>, março 2011,03.

[7] Albrecht, A. J., "Measuring application development productivity," Joint Application Development Symposium, 1979, p. 83-92.

[8] Karner, G. Resource estimation for objectory projects. Objectory Systems, 1993.

[9] Kusumoto, S., Imagawa, M., Morimoto, S. "Function point measurement from java programs". 24 International Conference on Software Engineering, 2002.

[10] CESAR, "Centro de Estudos e Sistemas Avançados de Recife". <ttp://www.cesar.org.br>, maio 2011,03.

[11] Tavares, H. C. A. B, Carvalho, A. E. S. J. Castro, F. B., "Medição de pontos por função a partir da especificação de requisitos", Workshop em Engenharia de Requisitos (WER 2002), 2002, p. 278-298.

[12] Nesma, Netherlands Function Point Users Group. <http://www.nesma.nl/section/nesma/>, março 2011,05.

[13] Roetzheim, W., "Project cost adjustments," SD Magazine, November/2000. <http://www.sdmagazine.com/articles/2000/0011/0011g/0011g.htm>, março 2011,14.

[14] Boehm, B.W., Fairley, R.E., "Software estimation perspectives," IEEE Software, Vol 17, Nº 6, November 2000, p. 22-26.

[15] Jones, C., "Software sizing during requirements analysis", 2008. <http://www.modernanalyst.com/Resources/Articles/tabid/115/articleType/ArticleView /articleId/512/Software-Sizing-During-Requirements-Analysis.aspx>, fevereiro 2011, 5.

[16] Zadeh, L. A., "Fuzzy sets," Fuzzy Sets, Information and Control, Vol. 8, 1965, p. 338-353.

[17] Zadeh, L. A., "Fuzzy Logic = computing with words," IEEE Transactions On Fuzzy Systems, Vol. 4, No. 2, maio 1996, p. 103-111.

[18] Yau, C.; Tsoi, R., "Assessing the fuzziness of general," System Characteristics in Estimating Software Size. IEEE Transactions on Software Engineering, 1994, p. 189-193.

[19] Lima Júnior, O. S.; Farias, P. P. M.; Belchior, A. D., "A fuzzy model for function point analysis," Software Quality Journal, Vol. 11, 2003, p. 149-166.

[20] Braz, M. R.; Vergilio, S. R., "Using fuzzy theory for effort estimation of object-oriented software," 16th IEEE. International Conference on Tools with Artificial Intelligence (ICTAI'04), 2004, p. 196-201.

[21] Fu, Y.; Liu, X., Yang, R., Du, Y.; Li, Y., "A software size estimation method based on improved FPA," Second WRI World Congress on Software Engineering (WCSE), Vol. 2, 2010, p. 228-233.

[22] Chen, Q.; Cheng, R.; Fang, S.; Ou, Y., "Study of function points analysis based on fuzzy-interpolation," Journal of Computational Information Systems, Vol. 6, No. 5, 2010, p. 1369-1375.