# Um Modelo de Avaliação da Produtividade de Projetos de Software baseado em uma Abordagem Multicritério

**José Adson O. G. da Cunha[1,2], João Pedro C. Fernandes Thomaz [3,4], Hermano Perrelli de Moura[1]**

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

[2]Unidade de Desenvolvimento Paraíba, DATAPREV – Empresa de Tecnologia e Informações da Previdência Social
58.013-240 – João Pessoa – PB – Brasil

[3]Faculdade Boa Viagem (FBV)
51.200-060 – Recife – PE – Brasil

[4]Centro de Estudos de Gestão do Instituto Superior Técnico (CEG-IST)
1069-001 Lisboa – Portugal

jaogc@cin.ufpe.br, joaothomaz@netcabo.pt, hermano@cin.ufpe.br

***Abstract.*** *In Software Engineering, productivity is determined by the interaction of many factors, so that none factor in particular is capable of ensuring high performance of a software project. Nevertheless, it is measured on a single way by the division between the quantities produced by the effort required. Therefore, it is necessary a way of measuring the real productivity of a software project taking into consideration the circumstances in which the software was developed. Thus, this article proposes a model for evaluating the productivity of software projects through a multicriteria approach.*

***Resumo.*** *No âmbito da Engenharia de Software, a produtividade é determinada pela interação de muitos fatores, de modo que nenhum fator em especial é capaz de garantir o alto desempenho em um projeto de software. Apesar disso, ela é medida de forma única, através da divisão entre a quantidade produzida pelo esforço necessário. Portanto, torna-se necessária uma forma de medição da real produtividade de um projeto de software que reflita as circunstâncias nas quais o software foi desenvolvido. Dessa forma, este artigo propõe um modelo de avaliação da produtividade de projetos de software através de uma abordagem multicritério.*

## 1. Introdução

Na Economia, a produtividade, de um modo geral, é a relação entre a quantidade de bens ou serviços produzidos e a despesa ou trabalho necessário para produzi-los (Jones, 1996). Conseqüentemente, a produtividade de software é a relação entre a quantidade

de software produzida e a despesa ou trabalho para produzi-la. Apesar de simples na teoria, na prática tal definição se torna um problema não muito trivial.

Diversos estudos ao longo da história da Engenharia de Software procuraram identificar que fatores influenciam na produtividade e qualidade no desenvolvimento de software. Scacchi (1995) realizou um estudo, no qual discute as várias abordagens de medição da produtividade, bem como os fatores que afetam a produtividade, sugerindo a construção de um sistema de modelagem e simulação da produtividade em projetos de software baseado em conhecimento. Em seu estudo, Scacchi identificou dezoito fatores, dividindo-os em três grupos: os relacionados com o ambiente de desenvolvimento, com o produto a ser desenvolvido e com a equipe envolvida. Jones (1986) aponta mais de quarenta variáveis que podem influenciar na produtividade do desenvolvimento de software. O modelo de estimativa de custo COCOMO II (Boehm, 1996) divide os fatores em quatro grupos: relacionados com o produto, com os computadores utilizados no projeto, com a equipe de pessoas envolvida e com o projeto (que inclui práticas de desenvolvimento). White (1999) divide os fatores em quatro grupos: relacionados com o produto, com a equipe de pessoas envolvidas, com a tecnologia utilizada e com o processo utilizado. De acordo com Chiang (2004), a melhoria na produtividade no desenvolvimento de software se dá através do balanceamento de três pilares do gerenciamento de software: tecnologia, pessoas e processo. Clincy (2003) realizou um estudo com líderes de projeto de diferentes organizações e concluiu que as áreas que mais impactam na habilidade da organização de aumentar sua produtividade são: estrutura e clima organizacional, sistemas de recompensa, processos de desenvolvimento de software e uso de ferramentas. Taylor (2005) concluiu que, além dos demais fatores, o foco na melhoria da cultura organizacional pode trazer grandes benefícios na produtividade da organização.

Outros trabalhos, além de realizar análises dos vários fatores que afetam a produtividade e qualidade, apontam padrões de comportamento e organização de projetos de software que possibilitam alta produtividade. Bailey et al. (1981) concluíram que a alta produtividade está diretamente relacionada com o uso de metodologia de desenvolvimento de software disciplinada. Já Behrens (1983) concluiu em sua análise realizada em vinte e cinco projetos de desenvolvimento de software que o tamanho do projeto, o ambiente de desenvolvimento e linguagem de programação impactam na produtividade. Em particular, o autor concluiu que pequenos times produzem mais código do que grandes times. Boehm (1981) reporta que a produtividade em um projeto de desenvolvimento de software é mais fortemente afetada por aqueles que desenvolvem o sistema e pela forma como estes estão organizados e gerenciados como time. Scacchi (1984) complementa esta evidência revisando alguns relatórios sobre gerenciamento de grandes projetos. Nesta revisão, ele confirma que quando projetos são mal gerenciados ou organizados, a produtividade é substancialmente mais baixa do que o normal.

DeMarco (1999) em seu trabalho que envolveu a análise de aproximadamente 600 desenvolvedores de 92 organizações, defende que os principais problemas de produtividade estão relacionados a fatores humanos e organizacionais e não a fatores técnicos como muitos defendem. O autor apresenta, inclusive, uma série de

experimentos realizados com o objetivo de identificar tais fatores e em seus resultados fica explícita tal afirmação. Nestes experimentos, fatores como linguagem de programação adotada, faixa salarial e experiência na função não influenciam significativamente na produtividade. Já outros fatores, como espaço físico individual e nível de ruído no ambiente de trabalho afetam de um modo muito intenso.

De fato, a produtividade é determinada pela interação de muitos fatores, de modo que nenhum fator em especial é capaz de garantir a alta produtividade em um projeto de software. Altos níveis de um determinado fator podem beneficiar o aumento da produtividade, mas um fator isoladamente não é suficiente para a obtenção de um alto nível de produtividade (Scacchi, 1995).

De modo a estimar o esforço, bem como planejar o desenvolvimento de um software, gerentes de projetos precisam muitas vezes se basear nos resultados de bases históricas. Para tanto, é necessário que as circunstâncias nas quais os projetos foram desenvolvidos sejam documentadas de forma padronizada. No entanto, atualmente, a produtividade é medida de forma única, através da divisão entre a quantidade produzida (geralmente medida em linhas de código ou pontos de função) pelo esforço necessário (geralmente medido em horas ou homem-mês).

A tarefa de avaliar um projeto de software segundo os vários fatores que afetam a produtividade constitui-se um problema multicritério. Problemas dessa natureza são comumente solucionados através de abordagens multicritério, que, segundo Bouyssou (1990), caracterizam-se pela construção de vários critérios através da utilização de vários pontos de vista. Estes pontos de vista representam os eixos através dos quais os diversos atores de um processo de decisão justificam, transformam e questionam as suas preferências, realizando comparações com base na avaliação das alternativas de acordo com estes pontos de vista e estabelecendo, então, as preferências parciais.

Um dos principais objetivos dos modelos multicritério é a agregação de diferentes e conflitantes critérios em um indicador de desempenho (Santana, 2002). Os métodos de agregação a um critério único de síntese são abordagens multicritério onde a avaliação das alternativas é feita segundo uma função global de valor, que agrega todos os pontos de vista considerados fundamentais.

Dessa forma, este artigo propõe um modelo de avaliação da produtividade que incorpore os fatores que influenciam no desenvolvimento de um projeto de software, através de uma abordagem multicritério.

O restante do artigo está organizado da seguinte forma: a Seção 2 apresenta a metodologia utilizada para elaboração do modelo; a Seção 3 apresenta como o modelo foi estruturado; a Seção 4 apresenta a fase de avaliação do modelo; a Seção 5 apresenta as recomendações a partir de projetos fictícios; e por fim, a Seção 6 apresenta as conclusões.

## 2. Metodologia Utilizada

De modo a construir o modelo proposto, foi utilizada a Metodologia de Conferências de Decisão (Phillips, 1982), que consiste em uma série de reuniões de trabalho intensivo,

chamadas de conferências de decisão, sem uma agenda pré-definida, tampouco apresentações preparadas, sendo compostas por grupos de pessoas que representam as várias dimensões da situação problemática, podendo durar de um a três dias, dependendo da complexidade do problema. Uma característica exclusiva é a criação, no local, de um modelo que incorpora dados e julgamentos dos participantes (Enterprise LSE, 2008).

A integração da Teoria da Decisão (processo técnico de modelagem do contexto e situação decisional), das Tecnologias de Informação (processo tecnológico de apoio computacional especializado) e dos Processos de Grupo (processo social de facilitação para planejar e dirigir uma reunião bem sucedida, mantendo-a imparcial e focada nas necessidades do grupo), na Metodologia de Conferências de Decisão, conforme mostrado na Figura 1, permite criar sinergias que irão tornar o produto final das sessões de trabalho (valor) maior do que a soma das suas partes (Reagan-Cirincione, 1994).

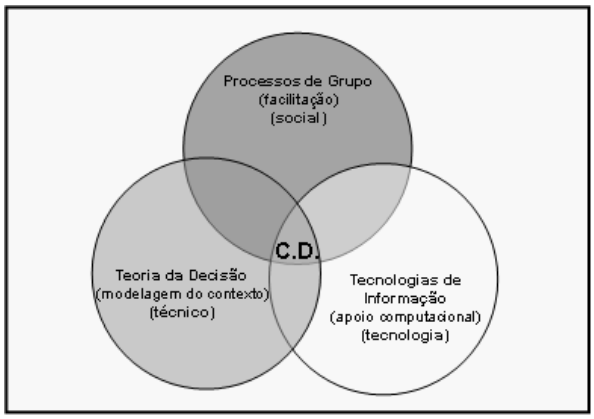


**Figura 1. Metodologia de Conferências de Decisão (Thomaz, 2005)**

De modo a apoiar a decisão através da determinação dos fatores a serem considerados para a construção de um modelo multicritério, os participantes tiveram que ser rigorosamente escolhidos para representar as várias dimensões do problema em questão.

As sessões foram compostas por seis participantes, além do facilitador, sendo três engenheiros de qualidade, com experiência em análise de requisitos, um consultor em métricas, um consultor em testes, com experiência em codificação e um gerente de projetos. Dos seis participantes, três são mestres e dois são mestrandos, além de um possuir certificação PMP. Dessa forma, constituiu-se uma equipe heterogênea, com membros experientes em áreas específicas da Engenharia de Software.

Ao todo, foram realizadas quatro sessões de conferências de decisão, respeitando a restrição de tempo imposta pelos participantes, o que de certa forma não comprometeu o resultado final. As sessões foram realizadas em uma empresa de desenvolvimento e pesquisa reconhecida internacionalmente, avaliada como CMMI nível 3. Maiores detalhes sobre as fases de elaboração do modelo podem ser obtidas em Cunha (2008).

## 3. Estruturação do Modelo

A fase de estruturação deu-se início com um pequeno *brainstorming*, auxiliado pela técnica de *Oval Mapping Technique* (Ovalmap, 2008) com a utilização de *post-its*, através da qual foram levantados alguns conceitos que serviram de pontos de partida para a determinação dos conceitos fundamentais para a resolução do problema. Foram fornecidos três *post-its* a cada participante para que escrevessem em cada um deles um conceito, aspecto ou uma pequena descrição que indicasse um fator considerado importante para a avaliação da produtividade de projetos de software.

Uma vez preenchidos, tais *post-its* foram recolhidos e, um a um, lidos e colocados em um quadro, sendo levantadas questões sobre os conceitos expressos, de modo a esclarecer o seu significado e manter o grupo concentrado na compreensão dos aspectos referidos. A Figura 2 apresenta os *post-its* preenchidos pelos participantes.

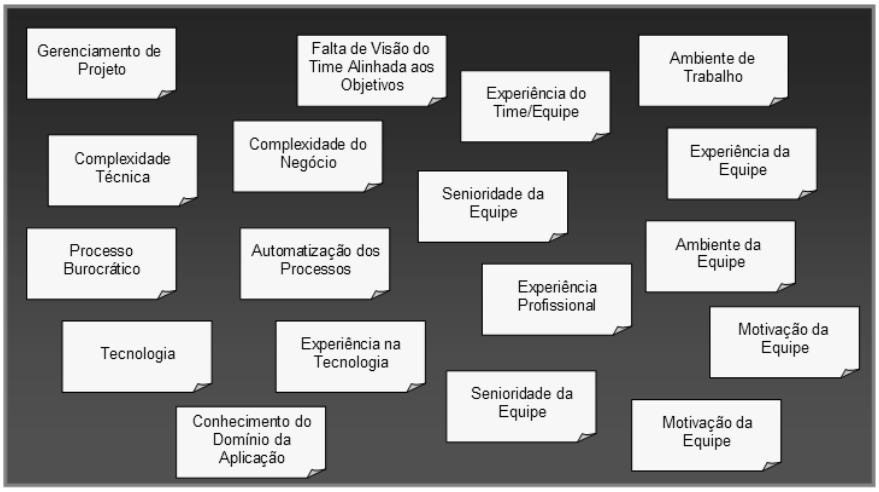


**Figura 2. Sessão de *post-its***

A partir das discussões entre os participantes, foi ficando evidente que muitos dos elementos do grupo tinham preocupações semelhantes sobre o problema, o que permitiu ao facilitador agrupar estes aspectos no quadro por "áreas de preocupação", ou seja, em subconjuntos de aspectos relacionados ou similares, conforme a Figura 3. Toda esta atividade do grupo foi desenvolvida em um ambiente de diálogo e com a utilização de questões de diagnóstico (Schein, 1999) sobre o "porquê?" e o "para quê?" de cada um dos aspectos considerados importantes, fazendo uso dos princípios e técnicas da consultoria de processos de grupo (Thomaz, 2005). Nesta fase, o facilitador teve que muitas vezes "parar" a discussão para procurar esclarecer o significado de conceitos e obter definições que ajudassem a clarificar o conceito que se pretendia transmitir no *post-it* (ou durante a discussão).

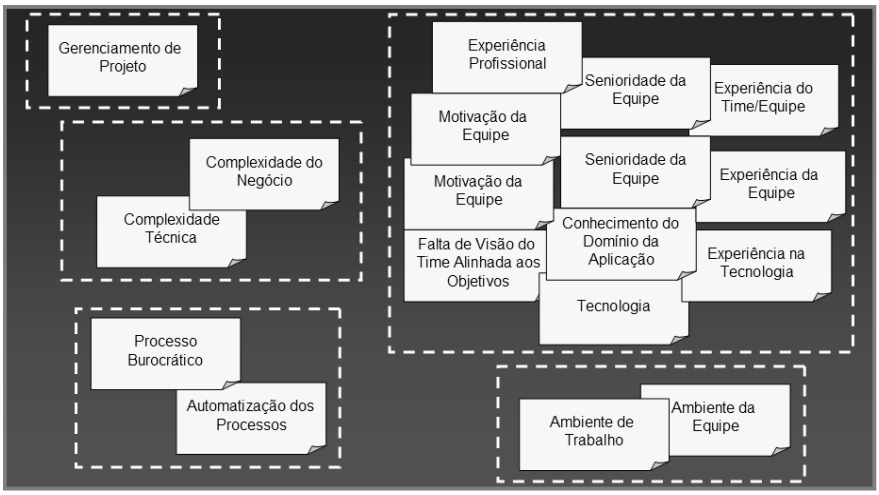


**Figura 3. *Post-its* agrupados**

Através da discussão realizada, foi possível reavaliar a importância de cada aspecto para a avaliação da produtividade de projetos de software e identificar novos aspectos e relações entre as várias dimensões do problema, o que permitiu descartar aspectos e renomear outros que não traduziam o conceito ou a idéia pretendida, sendo vários deles reescritos, desagregados ou agregados.

Uma vez definido o mapa cognitivo, foi necessário construir a árvore de pontos de vista. Um ponto de vista (PV) é a representação de um valor julgado suficientemente importante pelos atores para ser considerado de uma forma explícita no processo de avaliação das ações ou alternativas, podendo ser classificados em pontos de vista fundamentais (PVF) e pontos de vista elementares (PVE) (Bana e Costa, 1992 in Thomaz, 2005). Um ponto de vista fundamental reflete um valor relevante no contexto do problema, enquanto que os pontos de vista elementares são meios para se alcançar os pontos de vista fundamentais.

A Figura 4 mostra a árvore de pontos de vista construída para este problema. Conforme já referido, a árvore foi elaborada pelo facilitador e discutida com os atores. A árvore apresentada representa a estrutura definitiva do problema, não sendo a primeira proposta do facilitador e incorpora as modificações julgadas convenientes pelos atores não alterando, no entanto, a estrutura geral inicialmente obtida.


**Figura 4. Árvore de Pontos de Vista**

Para viabilizar um consenso entre os atores, e evitar dúvidas quanto ao que buscam representar os diversos elementos de avaliação, tornando os PVF's inteligíveis, os conceitos dos vários PVF's apresentados na Figura 4, a partir dos seus respectivos PVE's, foram definidos, conforme abaixo:

- **PVF 1 – Experiência da Equipe:** Corresponde às características da equipe em relação à capacidade de aplicar os conhecimentos teórico-práticos, tempo de uso desta capacidade para fins práticos e a aprendizagem demonstrada, conforme os PVE´s abaixo.
    - **PVE 1.1 – Tempo:** Tempo de experiência na área de atuação.
    - **PVE 1.2 – Conhecimento teórico-prático:** Nível de conhecimento na área de atuação.
    - **PVE 1.3 – Capacidade de Aprendizagem:** Aprendizado conquistado pelos membros ao final do projeto, em relação ao conhecimento dos mesmos no início do projeto.
- **PVF 2 – Ambiente de Trabalho:** Corresponde ao ambiente proporcionado aos membros do projeto, em termos de estrutura física e cultura organizacional, conforme os PVE´s abaixo.

  - **PVE 2.1 – Estrutura Física:** Adequação da empresa aos requisitos ergonômicos, de higiene, de segurança e de equipamentos, além do nível de barulho existente.
  - **PVE 2.2 – Cultura Organizacional:** Cultura da empresa quanto à existência de procedimentos normalizados, de um ambiente sócio-cultural, assim como de projeção internacional.

- **PVF 3 – Capacidade do Projeto:** Corresponde às características dos membros da equipe em termos de organização, liderança e trabalho em equipe, conforme os PVE´s abaixo.
  - **PVE 3.1 – Organização:** Existência e atualização do planejamento por parte do gerente de projeto, com a definição do cronograma, pontos críticos, metas, alinhando à equipe com os resultados.
  - **PVE 3.2 – Liderança:** Capacidade do líder (gerente de projeto) em conduzir a equipe ao sucesso.
  - **PVE 3.3 – Trabalho em Equipe:** Nível de envolvimento, integração e compromisso mútuo dos membros do projeto para com os resultados.
- **PVF 4 – Processo:** Corresponde à adaptabilidade do processo, bem como o grau de automação disponível para o mesmo, conforme os PVE´s abaixo.
  - **PVE 4.1 – Adaptabilidade:** Capacidade do processo de ser modificado de modo a torná-lo mais produtivo, diante das características do projeto.
  - **PVE 4.2 – Automação:** Nível de utilização de ferramentas para automatizar as atividades do processo.
- **PVF 5 – Complexidade do Projeto:** Corresponde à complexidade do projeto em termos da complexidade técnica e do negócio, conforme os PVE´s abaixo.
  - **PVE 5.1 – Complexidade Técnica:** Variedade de tecnologias utilizadas no projeto, bem como ao nível de complexidade dos algoritmos utilizados.
  - **PVE 5.2 – Complexidade do Negócio:** Complexidade oriunda do domínio da aplicação.
- **PVF 6 – Cliente:** Corresponde à participação do cliente ao longo do projeto, em termos da volatilidade dos requisitos e do envolvimento do mesmo ao longo do projeto, conforme os PVE´s abaixo.
  - **PVE 6.1 – Volatilidade dos Requisitos:** Nível com que os requisitos são modificados ao longo do projeto.
  - **PVE 6.2 – Envolvimento do Cliente:** Nível de participação do cliente ao longo do projeto.

Uma vez definidos os pontos de vista fundamentais, foi necessário operacionalizá-los, ou seja, construir seus descritores. De acordo com Bana e Costa

(1992), um descritor é definido como um conjunto ordenado de níveis de impacto plausíveis associado a um ponto de vista fundamental **j**, denotado por $\mathbf{N_j}$, onde cada nível de impacto deste descritor é denotado por $\mathbf{N_{k,j}}$, e corresponde à representação do impacto de uma ação ideal, de tal forma que da comparação de quaisquer dois níveis do descritor resulte sempre uma diferenciação clara, aos olhos dos atores, no que se refere aos elementos primários de avaliação que formam este ponto de vista fundamental.

Uma vez que foi constatada a não existência de descritores diretos (ou naturais) para os PVF´s, os mesmos foram operacionalizados através de descritores construídos, através de combinações entre os níveis dos descritores dos seus respectivos PVE´s. Para cada PVF, foram definidos 3 a 5 níveis, dentre os quais os níveis "Bom" e "Neutro", a serem utilizados posteriormente na ponderação dos pontos de vista fundamentais. A indicação de tais níveis é importante de modo a eliminar a influência de níveis de impacto considerados muito negativos, segundo o avaliador, de modo a não prejudicar a determinação dos pesos.

De modo a exemplificar a construção dos descritores, será considerado o **PVF 4 – Processo**, o qual foi operacionalizado através de um descritor qualitativo, construído e discreto, formado pelos PVE´s Adaptabilidade e Automação.

O **PVE 4.1 – Adaptabilidade** foi operacionalizado através de um descritor qualitativo, construído e discreto, correspondendo à capacidade do processo de ser modificado de modo a torná-lo mais produtivo, em se tratando da necessidade de artefatos a serem produzidos. De modo a construir o descritor do PVF, foram definidos os níveis deste PVE, conforme a Tabela 1.

**Tabela 1. Descritor do PVE 4.1 – Adaptabilidade**

| Nível | Descrição | B | N |
|---|---|---|---|
| **N3** | Processo altamente adaptável, com a utilização de artefatos realizada conforme a necessidade do projeto. | | |
| **N2** | Apesar da pouca quantidade de artefatos exigidos pelo processo, o mesmo indica que todos os artefatos sejam gerados, independente da necessidade do projeto. | | |
| **N1** | Processo burocrático, com a necessidade da geração de grande quantidade de artefatos, sendo muitos deles pouco utilizados ao longo do projeto. | | |

O **PVE 4.2 – Automação** foi operacionalizado através de um descritor qualitativo, construído e discreto, correspondendo ao nível de cobertura das ferramentas em relação às atividades do processo. De modo a construir o descritor do PVF, foram definidos os níveis deste PVE, conforme a Tabela 2.

**Tabela 2. Descritor do PVE 4.2 – Automação**

| Nível | Descrição | B | N |
|---|---|---|---|
| **N3** | Ambiente CASE integrado, cobrindo todo o ciclo de vida do projeto. | | |
| **N2** | Ferramentas de suporte à maior parte das fases do ciclo de vida do projeto. | | |
| **N1** | Mínima quantidade de ferramentas, englobando apenas editores e compiladores. | | |

O **PVF 4 - Processo** foi operacionalizado através da combinação dos níveis dos descritores dos PVE´s que o formam, conforme definido na Tabela 3, onde o nível N4 foi considerado “Bom” e o nível N2 “Neutro”.

**Tabela 3. Descritor do PVF 4 – Processo**

| Nível | Descrição | B | N |
|---|---|---|---|
| **N4** | Processo altamente adaptável, com a utilização de artefatos realizada conforme a necessidade do projeto, além de apresentar um ambiente CASE integrado, cobrindo todo o ciclo de vida do projeto. | | |
| **N3** | Apesar da pouca quantidade de artefatos exigidos pelo processo, o mesmo indica que todos os artefatos sejam gerados, independente da necessidade do projeto, apresentando um ambiente CASE integrado, cobrindo todo o ciclo de vida do projeto. | | |
| **N2** | Apesar da pouca quantidade de artefatos exigidos pelo processo, o mesmo indica que todos os artefatos sejam gerados, independente da necessidade do projeto, apresentando ferramentas de suporte à maior parte das fases do ciclo de vida do projeto. | | |
| **N1** | Processo burocrático, com a necessidade da geração de grande quantidade de artefatos, sendo muitos deles pouco utilizados ao longo do projeto, apresentando ferramentas de suporte à maior parte das fases do ciclo de vida do projeto. | | |

O mesmo procedimento foi realizado para a construção dos descritores dos demais pontos de vista fundamentais, respeitando a verificação da não-ambiguidade, de forma a não haver perda de informação na associação de um determinado nível de impacto a um projeto (entre a pessoa que o associou e a outra que o interpreta).

## 4. Avaliação do Modelo

A atividade de construção de escalas foi realizada com o uso da metodologia MACBETH (*Measuring Attractiveness by a Categorical Based Evaluation Technique*), uma técnica interativa de apoio à construção, sobre um conjunto S de estímulos ou ações potenciais, de escalas numéricas de intervalos que quantifiquem a atratividade dos elementos de S na opinião do(s) ator(es), baseada em juízos semânticos de diferença de atratividade entre duas ações (Bana e Costa et al., 1994). É útil tanto para a construção de uma função de valor cardinal, quanto como técnica de ponderação, para a determinação de constantes de escala (taxas de substituição, pesos) em um modelo de agregação aditiva.

A idéia básica da abordagem MACBETH para a obtenção de escalas de valor cardinal é fazer um conjunto de questões sobre a diferença de atratividade entre dois estímulos e através de respostas semânticas permitir obter informação sobre cada PVF

(intra-critério), testando a consistência das respostas do decisor para obter uma escala cardinal compatível com os julgamentos semânticos (respostas) dados. Assim, o procedimento de questionamento consiste em solicitar ao decisor um julgamento verbal (qualitativo) sobre a diferença de atratividade entre cada duas ações "x" e "y" do conjunto S de estímulos ou ações (com "x" mais atrativo que "y"), escolhendo uma das seguintes categorias semânticas de diferença de atratividade ($C_k$):

$C_1$ – Diferença de atratividade ***muito fraca***
$C_2$ – Diferença de atratividade ***fraca***
$C_3$ – Diferença de atratividade ***moderada***
$C_4$ – Diferença de atratividade ***forte***
$C_5$ – Diferença de atratividade ***muito forte***
$C_6$ – Diferença de atratividade ***extrema***

Dessa forma, partiu-se para a construção das matrizes triangulares superiores de juízos de valor (ou de julgamentos absolutos de diferença de atratividade) sobre cada um dos descritores, obtendo escalas de valor cardinal que vão possibilitar a avaliação local (parcial) dos projetos de software em cada ponto de vista fundamental. Durante o processo de construção das matrizes ocorreram alguns casos de inconsistência cardinal que foram resolvidos através de discussões entre o facilitador e os atores, com a conseqüente modificação de alguns julgamentos.

Os seis pontos de vista fundamentais foram operacionalizados através de descritores qualitativos, construídos e discretos, embora alguns deles possuam pontos de vistas elementares avaliados com descritores quantitativos.

De modo a refinar as funções de valor e aumentar a inteligibilidade da escala obtida, foram criados três projetos fictícios, onde "Proj 1" corresponde ao nível mais elevado em todos os PVF´s, "Proj 2" corresponde a um projeto entre "Bom" e "Neutro" em todos os PVF´s, e "Proj 3" corresponde a um projeto abaixo de "Neutro" em todos os PVF´s.

Assim, as matrizes ficaram com cinco níveis, compostos pelas 3 opções (projetos fictícios) e 2 níveis de referência ("Bom" e "Neutro"). Como se pode observar na Figura 5, nas matrizes de julgamento não houve unanimidade na atribuição de uma categoria semântica MACBETH para as diferenças de atratividade entre os alguns dos níveis de impacto, como no caso da diferença de atratividade entre o nível "Proj 1" e o nível "Neutro" (Forte – Muito Forte), do PVF 1. No entanto, tal hesitação foi considerada e aceita pela metodologia MACBETH obtendo-se assim, a escala de valor cardinal, à direita da figura, para este ponto de vista. É, obviamente, aconselhável que não haja hesitações que correspondam a grandes intervalos de diferença de atratividade para cada par, como por exemplo "Fraco – Muito Forte", uma vez que demonstra pouco conhecimento do problema por parte dos atores ou uma situação pouco esclarecida.

A escala obtida na Figura 5 representa os valores cardinais gerados pelo MACBETH para cada um dos quatro níveis de impacto do descritor do PVF 4 – Processo, através dos quais os projetos de software serão avaliados, segundo este PVF.

| | Proj 1 | BOM | Proj 2 | NEUTRO | Proj 3 | Escala atual |
|---|---|---|---|---|---|---|
| Proj 1 | nula | nula | fraca | fort-mfort | extrema | 100.00 |
| BOM | nula | nula | fraca | fort-mfort | extrema | 100.00 |
| Proj 2 | | | nula | mod-fort | mt. forte | 60.00 |
| NEUTRO | | | | nula | fraca | 0.00 |
| Proj 3 | | | | | nula | -40.00 |

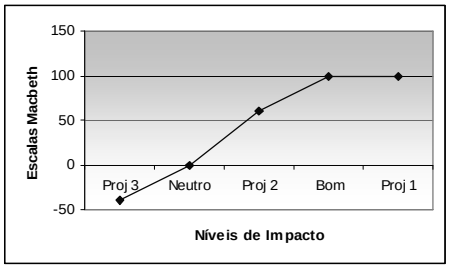


**Figura 5. Matriz de Juízos de Valor e Escala do PVF 4 – Processo**

Uma vez definidas as funções de valor cardinal de cada PVF, foi necessário definir os pesos (fatores de harmonização de escala). Antes de proceder com os julgamentos de diferença de atratividade, os participantes tiveram que ordenar os PVF´s em ordem decrescente de atratividade (condição para a utilização de uma matriz triangular superior). Concluída a ordenação dos PVF's, a matriz de julgamentos foi preenchida. Uma vez que não existia uma noção clara da diferença de atratividade entre os PVF´s, optou-se por julgar a maioria das diferenças como "positiva", como mostrado na Figura 6, verificando de seguida se a escala produzida correspondia à sensibilidade e intuição dos participantes.

| | [ PVF1-ExpEq ] | [ PVF 3 - CPro ] | [ PVF 5-CoProj ] | [ PVF 4-Proc ] | [ PVF 2-AmTrab ] | [ PVF 6-Client ] | Neutro | Escala atual |
|---|---|---|---|---|---|---|---|---|
| [ PVF1-ExpEq ] | nula | fraca | moderada | frac-mod | moderada | forte | positiva | 29.00 |
| [ PVF 3 - CPro ] | | nula | positiva | positiva | positiva | positiva | positiva | 24.00 |
| [ PVF 5-CoProj ] | | | nula | positiva | positiva | positiva | positiva | 19.00 |
| [ PVF 4-Proc ] | | | | nula | positiva | positiva | positiva | 13.00 |
| [ PVF 2-AmTrab ] | | | | | nula | positiva | positiva | 9.00 |
| [ PVF 6-Client ] | | | | | | nula | positiva | 6.00 |
| Neutro | | | | | | | nula | **0.00** |

**Figura 6. Matriz de Juízos de Valor para Obtenção das Taxas de Substituição**

A Figura 7 apresenta o histograma dos pesos atribuídos aos PVF's, onde o **PVF 1 - Experiência da Equipe** foi julgado como sendo o mais relevante dentro dos pontos de vista importantes para a avaliação da produtividade e o **PVF 6 - Cliente** como sendo o menos relevante nessa avaliação.

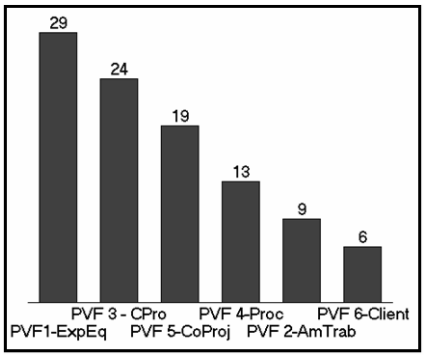


**Figura 7. Ponderação dos PVF's**

## 5. Elaboração das Recomendações

Uma vez concluído o processo, foram efetuadas as recomendações baseado no modelo de avaliação proposto. A Tabela 1 apresenta o resultado final do processo, com as avaliações globais das ações (ou projetos) fictícios diante dos seis pontos de vista fundamentais, através da agregação dos mesmos possibilitada pelos pesos, representando assim os índices de produtividade dos projetos.

Sendo as escalas fixadas em dois pontos, "Neutro" com 0 pontos e "Bom" com 100 pontos, a ação "Proj 1", com 119.50 pontos, apresentou um rendimento superior à ação globalmente boa ("Bom" – 100.00), enquanto a ação "Proj 3", com -47.35 pontos apresentou um rendimento inferior à ação globalmente neutra ("Neutro" – 0.00).

**Tabela 1. Resultado do modelo**

| Opções | Global | PVF1-ExpEq | PVF 2-AmTrab | PVF 3 - CPro | PVF 4-Proc | PVF 5-CoProj | PVF 6-Client |
|---|---|---|---|---|---|---|---|
| Proj 1 | 119.50 | 130.00 | 100.00 | 140.00 | 100.00 | 100.00 | 120.00 |
| Bom | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 |
| Proj 2 | 58.60 | 70.00 | 40.00 | 60.00 | 60.00 | 50.00 | 50.00 |
| Neutro | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| Proj 3 | -47.35 | -35.00 | -50.00 | -60.00 | -40.00 | -50.00 | -60.00 |
| Pesos : | | 0.2900 | 0.0900 | 0.2400 | 0.1300 | 0.1900 | 0.0600 |

Dessa forma, caso o modelo fosse utilizado para fins de *benchmarking*, o resultado provido pelo índice poderia fornecer informações aos gerentes de projetos que queiram utilizar os dados daquele projeto para estimar o esforço de seu projeto. Dessa forma, além de possuir a métrica padrão de produtividade, o gerente de projetos poderá contar com as circunstâncias nas quais o software foi desenvolvido, representado pelo índice.

Diante das restrições de tempo, o modelo não pôde ser discutido mais a fundo com os participantes das sessões. Desse modo, através da prática, o modelo apresentado pode servir de base para a elaboração de um modelo mais completo, com a participação de mais pessoas especialistas e com tempo disponível para as discussões que norteiam a elaboração de qualquer modelo.

## 6. Conclusões

A integração da Metodologia de Conferências de Decisão com a Metodologia Multicritério de Apoio à Decisão, das técnicas de mapeamento cognitivo e da abordagem MACBETH, permite construir um modelo capaz de representar tanto os vários objetivos conflitantes dos participantes e sua subjetividade inerente, quanto a inevitável incerteza em relação às conseqüências futuras, constituindo assim uma "ferramenta para pensar" que permite aos participantes a visualização das conseqüências lógicas das suas decisões, sob diversos pontos de vista.

A incorporação dos juízos valores do decisor, os múltiplos critérios, a interação e aprendizagem dos intervenientes, a abordagem de estruturação por pontos de vista e a utilização do software M-MACBETH, como gerador de funções de valor cardinal e das importâncias relativas dos pontos de vista fundamentais, se apresentaram como ferramentas bastante eficazes na apreciação de problemas multicritério.

Quando ao modelo proposto, uma vez que a simples medição da produtividade através da divisão da quantidade produzida pelo esforço gasto não retrata fielmente as circunstâncias nas quais um projeto foi desenvolvido, procurou-se através de uma abordagem multicritério elaborar um modelo que reflita a produtividade, levando em consideração os fatores elencados por especialistas nas diversas áreas da Engenharia de Software, em sessões de Conferências de Decisão.

Dessa forma, além da utilização da medição padrão da produtividade (tamanho / esforço), gerentes de projetos poderiam utilizar o modelo proposto para enriquecer as bases históricas de projetos de software, de modo a tornar possível a realização de *benchmarking* permitindo a estimativa de esforço baseado em projetos similares levando em consideração as avaliações de cada projeto segundo o modelo proposto, refletindo assim as circunstâncias nas quais o software foi desenvolvido.

O modelo proposto vem, assim, preencher o *gap* existente na avaliação da produtividade de projetos de software, o qual até então é medido através da quantidade produzida pelo esforço necessário. O índice de produtividade provido através do modelo proporcionará uma maior riqueza de informações quando a produtividade de projetos for utilizado para estimativa de esforço de projetos futuros.

## Referências

Bailey, J. and Basili, V. (1981) A Meta-Model for Software Development Resource Expenditures. In: Proc. 5th. Intern. Conf. Soft. Engr., IEEE Computer Society, pp 107-116.

Bana e Costa. C.A., (1992) “Structuration, Construction et Exploitation d'un Modèle Multicritère d'Aide à la Décision”, Tese de Doutoramento em Engenharia de Sistemas, Universidade Técnica de Lisboa, IST, Lisboa.

Bana e Costa, C.A, Vansnick, J.C. (1994) MACBETH – Na interactive path towards the construction of cardinal value functions, International Transactions in Operational Research, 1, 4, (489- 500).

Behrens, C. A. (1983) Measuring Software Productivity of Computer System Development Activities with Point Functions. IEEE Trans. Soft. Engr. SE-9(6), pp. 648-652.

Boehm, B. (1981) Software Engineering Economics. Prentice-Hall, Englewood Cliffs, NJ.

Boehm, B. W. et al. (1996) The COCOMO 2.0 Software Cost Estimation Model. American Programmer, July, pp.2-17.

Bouyssou, D. (1990) Building Criteria: A prerequisite for MCDA, in: Bana e Costa, C. A. (Ed.), Readings in Multiple Criteria Decision Aid, Springer-Verlag, Berlin, (58-80).

Chiang, R. and Mookerjee, S. (2004) Improving Software Team Productivity. Communications of the ACM. Volume 47 , Issue 5. Pages: 89 – 93.

Clincy, V. A. (2003) Software Development Productivity and Cycle Time Reduction. Journal of Computing Sciences in Colleges. Volume 19, Issue 2. Pages: 278-287.

Cunha, José Adson O. G. (2008) Decisius: Um Processo de Apoio à Decisão e sua Aplicação na Definição de um Índice de Produtividade para Projetos de Software. Dissertação de Mestrado. Universidade Federal de Pernambuco.

DeMarco, T. (1999) Peopleware: Productive Projects and Teams, 2nd Ed. Dorset House Publishing.

Enterprise LSE (2008), Decision Conferencing. Disponível em: http://www.lse.ac.uk/collections/decisionConferencing/. Consulta realizada em 20/01/2008.

Jones, C. (1986) Programming Productivity, McGraw-Hill, New York.

Jones, C. (1996) Applied Software Measurement: Assuring Productivity and Quality. 2 ed. McGraw-Hill.

Ovalmap (2008) http://www.ovalmap.com . Acessado em 26/01/2008.

Phillips, L. D. (1982) Requisite decision modelling: A case study, Journal of the Operational Research Society, 33, 4 (pp. 303–311).

Reagan-Cirincione, P. (1994), “Improving the accuracy of group judgment: A process intervention combining group facilitation, social judgment analysis, and information technology”, Organizational Behavior and Human Decision Processes, 58 (pp. 246–270).

Santana, E. A. (2002) Contrato satisfatório multidimensional e a teoria do incentivo, Revista Brasileira de Economia, 56, 4 (pp. 661–694).

Scacchi, W. (1984) Managing Software Engineering Projects: A social Analysis. IEEE Trans. Soft. Engr., SE-10(1), pp. 49-59.

Scacchi, W. (1995) Understanding Software Productivity. Appears in Advances in Software Engineering and Knowledge Engineering, D. Hurley (ed.), Volume 4, pp. 37-70.

Schein, E. H. (1999), Process Consultation Revisited: Building the Helping Relationship, Addison-Wesley, Reading, MA.

Taylor, B. (2005) Organizational Culture is Important in Software Productivity. http://www.workinginunison.com/papers/cultureandproductivity.pdf.

Thomaz, João Pedro da Cruz Fernandes (2005) O Apoio à Tomada de Decisão na Avaliação do Desempenho de Pessoas: Contributos para o Processo de Decisão Militar em Tempo de Paz, Tese de Doutorado, Instituto Superior Técnico, Universidade Técnica de Lisboa.

White, K. S. (1999) Software Engineering Management For Productivity And Quality. In: International Conference on Accelerator and Large Experimental Physics Control Systems, Trieste, Italy.