# UMA ANÁLISE DE MÉTRICAS DE SOFTWARE ORIENTADAS À FUNÇÃO E SUA APLICAÇÃO AO DESENVOLVIMENTO ORIENTADO A OBJETOS

ARTIGO

**Everton Alves Miranda**
*Professor do CEFET Campos*
*Formando do Curso Superior de Tecnologia em Informática - CEFET Campos*

***Resumo***

***Este artigo visa a efetuar uma abordagem da mensuração de softwares através da metodologia de Pontos por Função, partindo de aspectos históricos, demonstrando seu funcionamento e analisando sua possível aplicação em softwares desenvolvidos dentro do paradigma da Orientação a Objetos.***

## 1 Introdução

As atividades de desenvolvimento de software, quando enfocadas com objetivos profissionais, trazem à tona uma série de dificuldades que merecem particular atenção por parte do desenvolvedor. Dentre essas dificuldades estão aquelas relacionadas a estimativas de cronograma e custo de desenvolvimento. Entretanto, antes de se fazerem futuras estimativas, é preciso efetuar medições dos parâmetros obtidos no presente e verificar os parâmetros do passado.

As bases das estimativas de software são as informações obtidas através de medições dos parâmetros dos softwares produzidos anteriormente, para finalidades semelhantes. "Projetos e estimativas baseados em dados históricos ajudam a analisar riscos e fazer relações de custo/benefício".(VALLE, 2000, p. 29). Para esta finalidade é que se desenvolveram as Métricas de Software.

## 2 Histórico

Na década de 60, iniciaram-se os primeiros esforços para mensuração de softwares. Essas medições eram feitas através da contagem direta das linhas de código (LOC). Com a evolução das linguagens de programação, tornou-se necessário considerar também o esforço associado à complexidade e tamanho funcional do software. Para isso, por volta dos anos 70, ocorreu o surgimento de um novo tipo de métrica que, através de conceitos subjetivos, efetuava a mensuração do software de uma forma indireta, a qual ficou conhecida como Pontos por Função (Function Point – FP). "Os Pontos por Função são derivados usando-se uma relação empírica baseada em medidas de informações e complexidades do software." (PRESSMAN, 1995, p. 67)

Essa métrica foi originalmente projetada para sistemas de aplicações comerciais e, por isso, necessitou evoluir devido ao rápido aumento da complexidade algorítmica das aplicações. Nasceram então os chamados Pontos Característicos ou Pontos de Particularidade (Feature Points). "Deve-se notar que os Pontos de particularidade e os Pontos por Função representam a mesma coisa – a 'funcionalidade' ou 'utilidade' do software." (PRESSMAN,1995, p. 67)

TABELA Nº 1
MODELO DE PLANILHA PARA COMPUTAÇÃO DOS PONTOS POR FUNÇÃO

| **Fatores de Ponderação**: | | |
|---|---|---|
| **Entradas do Usuário** | **S=3, M=4 e C=6** | **Resultados Ponderados** |
| Entrada 1 | | |
| Entrada 2 | | |
| **...** | | |
| Entrada n | | |
| *Entradas Ponderadas →* | | |
| **Saídas do Usuário** | **S=4, M=5 e C=7** | |
| Saída 1 | | |
| Saída 2 | | |
| **...** | | |
| Saída n | | |
| *Saídas Ponderadas →* | | |
| **Consultas do Usuário** | **S=3, M=4 e C=6** | |
| Consulta 1 | | |
| Consulta 2 | | |
| **...** | | |
| Consulta n | | |
| *Consultas Ponderadas →* | | |
| **Arquivos Lógic. Inter.** | **S=7, M=10 e C=15** | |
| Arquivo 1 | | |
| Arquivo 2 | | |
| **...** | | |
| Arquivo n | | |
| *Arquivos Ponderados →* | | |
| **Interfaces Externas** | **S=5, M=7 e C=10** | |
| Interface 1 | | |
| Interface 2 | | |
| **...** | | |
| Interface n | | |
| *Interfaces Ponderadas →* | | |
| **Contagem Total →** (Pontos-por-função Brutos) | | |

*Legenda da Ponderação:*
**S** = *Simples* **M** = *Média* **C** = *Complexa*

Posteriormente, foram integradas à dimensão dos dados de software dimensões funcionais de controle para obter-se uma nova métrica orientada à função: os Pontos por Função 3D. "Pontos por Função 3D é respectiva à aplicações que enfatizam capacidades de função e controle. Características de todas as três dimensões são 'contadas, quantificadas e transformadas' em uma medida que fornece uma indicação da funcionalidade fornecida pelo software." (GOMES, 1999, p. 51)

## 3 Pontos por Função

Os pontos por função são computados preenchendo-se uma planilha semelhante às apresentadas nas tabelas Nºs 1 e 2, onde são representados cinco Itens do Domínio de Informação, sendo cada um deles ponderado conforme seu grau de complexidade, através de um fator matemático.

Seguem, abaixo, definições dos Itens de Domínio de Informação:

Uma *Entrada do Usuário* (Entrada Externa) processa informações oriundas do mundo exterior à aplicação.

TABELA Nº 2
**TABELA BÁSICA PARA COMPUTAÇÃO DOS PONTOS POR FUNÇÃO**
(PRESSMAN, P. 67)

| **Contagem** | | **Ponderação** | | | | **Resultado** |
|---|---|---|---|---|---|---|
| Entradas do Usuário | | **X** | **S** 3 | **M** 4 | **C** 6 | |
| Saídas do Usuário | | **X** | 4 | 5 | 7 | |
| Consultas do Usuário | | **X** | 3 | 4 | 6 | |
| Arquivos | | **X** | 7 | 10 | 15 | |
| Interfaces Externas | | **X** | 5 | 7 | 10 | |
| | | | | | | |
| **Contagem Total =** | | | | | | |

Uma *Saída do Usuário* (Saída Externa) gera informações para fora da Aplicação.

Uma *Consulta do Usuário* (Consulta Externa) recupera informações através de uma combinação de entrada e saída.

Um *Arquivo Lógico Interno* (ALI) é um conjunto de informações relacionadas e mantidas pela aplicação.

Um *Arquivo de Interface Externa* (AIE) é

um grupo de informações relacionadas logicamente e mantidas fora da fronteira da aplicação.

Um *Dado Elementar Referenciado* (DER) é um Campo presente em um Arquivo lógico Interno ou em um Arquivo de Interface Externa.

Um *Arquivo Lógico Referenciado* (ALR) é um Arquivo de Interface Externa ou um Arquivo Lógico Interno, lido ou mantido por uma função.

TABELA Nº 3
**UMA TÉCNICA PARA A DEFINIÇÃO DO GRAU DE COMPLEXIDADE**
(PINHEIRO, P.18)

| **Entradas Externas** | | | |
|---|---|---|---|
| ALR | 1-4 DERs | 5-15 DERs | > 15 DERs |
| 0 - 1 | *Simples* | *Simples* | *Média* |
| 2 | *Simples* | *Média* | *Complexa* |
| > 2 | *Média* | *Complexa* | *Complexa* |
| **Saídas Externas** | | | |
| ALR | 1-5 DERs | 6-19 DERs | > 19 DERs |
| 0 - 1 | *Simples* | *Simples* | *Média* |
| 2 - 3 | *Simples* | *Média* | *Complexa* |
| > 3 | *Média* | *Complexa* | *Complexa* |
| **Arquivos Internos e Interfaces Externas** | | | |
| RLR | 1-19 DERs | 20-50 DERs | > 50 DERs |
| 1 | *Simples* | *Simples* | *Média* |
| 2 - 5 | *Simples* | *Média* | *Complexa* |
| > 5 | *Média* | *Complexa* | *Complexa* |

Quanto ao grau de complexidade, cabe a cada organização usuária efetuar suas próprias definições. Uma das técnicas utilizadas parte do relacionamento entre o número de Registros/ Arquivos Lógicos Referenciados (RLR/ARL) e o número de Dados Elementares Referenciados (DER), existentes nos Itens do Domínio de Informação, conforme o exemplo exposto na Tabela Nº 3.

A totalização dos pontos por função de Entradas, Saídas e Consultas do Usuário, juntamente com os de Arquivos lógicos Internos e Arquivos de Interface Externas, determinam os Pontos por Função Brutos do sistema de software em questão. O Próximo passo é determinar um fator de ajuste (*Fi*) para o mesmo, baseado em quatorze características gerais do sistema. (Tabela Nº 4)

"Cada característica está associada a descrições que auxiliam na determinação do nível de influência da mesma, que é graduado de 0 (zero) a 5 (cinco)."

Essas características influenciam em até ±35% (Trinta e Cinco por Cento) o tamanho do projeto ou aplicação. Conforme a fórmula (1) abaixo:

$$FP = CT \times [0{,}65 + (0{,}01 \times Fi)] \quad (1)$$

Onde:

FP = Pontos por Função (Function Point)

CT = Contagem total (Pontos por Função Brutos)

*Fi* = Fator de Ajuste

"O Projeto pode variar de 0,65 até 1,35." (PINHEIRO, p.18)

"As organizações que usam métodos de Pontos-por-Função desenvolvem critérios para determinar se uma entrada particular é simples, média ou complexa. Apesar disso, a complexidade é um tanto subjetiva". (PRESSMAN, 1995, p. 67)

TABELA Nº 4 - CARACTERÍSTICAS GERAIS DO SISTEMA

| **Pontuação de 0 a 5** | |
|---|---|
| **0 - Sem Influência 1 - Incidental**<br>**2 - Moderado 3 - Médio**<br>**4 - Significativo 5 - Essencial** | |
| a) Necessidade de Backup | |
| b) Necessidade de Comunicação de Dados | |
| c) Necessidade de Processamento Distribuído | |
| d) Necessidade de Alto Desempenho | |
| e) Necessidade de Utilização intensiva do Ambiente Operacional | |
| f) Necessidade de Entrada de Dados On-line | |
| g) Necessidade de Múltiplas Telas para Entradas On-line | |
| h) Necessidade de Atualização On-line de Arquivos-Mestres | |
| i) Necessidade de Reuso do Código | |
| j) Necessidade de Conversão e Instalação (Inclusas no Projeto) | |
| k) Necessidade de Multiplas Instalações | |
| l) Complexidade das Transações (Entradas, Saídas, Consultas, etc.) | |
| m) Complexidade do Processo Interno | |
| n) Facilidade de Utilização e Manutenção | |
| | |
| **Fator de Ajuste** (*Fi*) | |
| | |

Esse resultado pode ser utilizado para efetuar outras estimativas do software. Por exemplo:

| | |
|---|---|
| Produtividade = | FP/Pessoas x Mês |
| Qualidade = | Defeito/FP |
| Custo = | $/FP |
| Documentação = | Páginas de Documentos/FP |

## 4 Considerações

Estas métricas demonstraram sua validade para o desenvolvimento tradicional, porém apresentaram algumas falhas no Desenvolvimento Orientado a Objetos, pois existem atributos deste tipo de projeto que invalidam alguns Pontos por Função. Alguns fundamentos da orientação a objetos reduzem a validade da contagem de funções para avaliação de esforço e recursos necessários para a efetivação de um determinado projeto.

"Para pôr em prática a contagem de regras através de Pontos por Função para aplicativos de software desenvolvidos com Engenharia de Software Orientada a Objeto (ESOO), os conceitos e terminologias de ESOO e os de Pontos por Função têm de ser um conjunto dentro das relações do outro." (FETCKE, 1997, p. 03)

"Existem várias propostas para métricas OO que consideram as características e interações do sistema: número de classes, número de *cases*, número de métodos, médias de métodos por classe, média de linhas de código por métodos, profundidade máxima da hierarquia de classes, a relação existente entre métodos públicos e privados, entre outros." (GOMES, 1999, p. 52)

## 5 Conclusões

Para efetivação de uma Métrica Orientada a Funções para um Projeto Orientado a Objetos, é primordial a adição de um peso às características das classes, a qual produzirá uma medida de complexidade das mesmas, assim como se faz com os parâmetros de medidas utilizados nas medições de software não orientados a objetos (Características Gerais do Sistema). Deve-se dar uma maior importância às características de reusabilidade de código produzido, assim como, considerar a utilização de componentes pre-implementados, quando da definição das bases para o novo fator de ajuste de complexidade (*Fi*), que a partir de agora será aplicado à Orientação a Objetos e utilizado no cálculo, conforme a fórmula (2) abaixo:

$$FPOO = CT \times [0{,}65+(0{,}01 \times Fioo)] \quad (2)$$

Onde:

FPOO = Pontos por Função OO
CT = Contagem total (Pontos-por-Função Brutos)
*Fioo* = Fator de Ajuste para Orientação a Objeto

Para isso, será necessário efetuar mensurações em uma grande quantidade de softwares já desenvolvidos, para a obtenção dos dados históricos necessários ao embasamento das estimativas dos próximos softwares a serem desenvolvidos.

É possível que se tenha uma maior facilidade de efetuar esta coleta de dados, à medida em que vamos desenvolvendo os softwares atuais, do que tentando obtê-los de softwares desenvolvidos anteriormente. De qualquer forma, ambas as possibilidades demandarão uma grande quantidade de tempo para terem sua implementação concretizada.

## 6 Referências Bibliográficas

[1]FETCKE,T., ABRAM, A., NGUYEN, T-H. Mapping the OO-Jacobson Approach into Functions Points Analysis. Published in the Proceedings of TOOLS, 23, (97), Santa Barbara, CA, 28 July – 1 August 1997

[2]FURLAN, J. A. Modelagem de Objetos através da UML. Makron Books : São Paulo, 1998.

[3]GOMES, A. E. Métricas e Estimativas de Software: o Início de um Rallye. Developers Magazine, Rio de Janeiro, nov. 1999.

[4]Page-jones, M. O Que Todo

Programador Deveria Saber Sobre Projeto Orientado a Objetos. Makkron Books : São Paulo, 1997

[5]PINHEIRO, C. A. R., Análise de Pontos por Função Como Métrica de Desenvolvimento. Developers Magazine, Rio de Janeiro, nov. 1998.

[6]PRESSMAN, R. S., Engenharia de Software. Makron Books : São Paulo, 1995

[7]RUMBAUGH, J., PREMERLANI, W., EDDY, F., LORENSEN, W. Modelagem e Projetos Baseados em Objetos. Editora Campus : Rio de Janeiro, 1994.

[8]Simmervile, I., Software Engineering. Addison-Wesley Publishing Company Inc. Edinburgh Gate, England, 1995

[9]VALLE, A., MARCINIUK, M., MELHORETO, S.M., BURNETT, R. Um Roadmap Para Métricas de Software: Definições e Histórico. Developers Magazine, Rio de Janeiro, set. 2000.