INSTITUTO BRASILIENSE DE DIREITO PÚBLICO – IDP
ESCOLA DE DIREITO DE BRASÍLIA – EDB
ESPECIALIZAÇÃO EM CONTROLE EXTERNO E GOVERNANÇA PÚBLICA

MARCELO AUGUSTO PEDREIRA XAVIER

MÉTRICAS DE *SOFTWARE* NAS CONTRATAÇÕES PÚBLICAS FRENTE ÀS RECOMENDAÇÕES DO TCU: ESTUDO DE CASO NO TRIBUNAL DE CONTAS DO ESTADO DE GOIÁS SOBRE O USO DA MEDIDA HOMEM-HORA

GOIÂNIA,
MARÇO 2017

MARCELO AUGUSTO PEDREIRA XAVIER

MÉTRICAS DE *SOFTWARE* NAS CONTRATAÇÕES PÚBLICAS FRENTE ÀS RECOMENDAÇÕES DO TCU: ESTUDO DE CASO NO TRIBUNAL DE CONTAS DO ESTADO DE GOIÁS SOBRE O USO DA MEDIDA HOMEM-HORA

Trabalho de Dissertação apresentado ao Curso de Pós-Graduação *Lato Sensu* como requisito parcial para obtenção título de Especialista em Controle Externo e Governança Pública.

Orientadora: Polliana Cristina Oliveira de Carvalho

GOIÂNIA,
MARÇO 2017

Marcelo Augusto Pedreira Xavier

MÉTRICAS DE *SOFTWARE* NAS CONTRATAÇÕES PÚBLICAS FRENTE ÀS RECOMENDAÇÕES DO TCU: ESTUDO DE CASO NO TRIBUNAL DE CONTAS DO ESTADO DE GOIÁS SOBRE O USO DA MEDIDA HOMEM-HORA

Trabalho de Dissertação apresentado ao Curso de Pós-Graduação Lato Sensu como requisito parcial para obtenção título de Especialista em Controle Externo e Governança Pública.

Goiânia-GO, 22 de Março de 2017.

_____

Profª Polliana Cristina de Oliveira Carvalho
Professora Orientadora

_____

Prof. Thiago Moreira de Carvalho
Membro da Banca Examinadora

_____

Prof.....
Membro da Banca Examinadora

GOIÂNIA,
MARÇO 2017

À minha companheira e aos meus filhos, fontes de inspiração, a quem dedico este trabalho e a minha vida. Uma dedicação especial à memória do meu amigo Pedro.

AGRADECIMENTOS

Agradeço a Deus sobre todas as coisas e aos meus familiares pelo apoio incondicional para que eu pudesse alcançar mais esta conquista.

Aos professores que compartilharam seu conhecimento, por todo o empenho e dedicação.

Aos colegas do Tribunal de Contas do Estado de Goiás pelas valiosas contribuições durante o curso.

A todos os demais que colaboraram de alguma forma para a elaboração deste trabalho.

“Uma ciência é tão desenvolvida quanto suas ferramentas de medição”. (Louis Pasteur)

RESUMO

Este trabalho analisa a utilização de métricas de engenharia de *software* por órgãos públicos para remunerar a terceirização de serviços de TI, referente ao problema de pagar por serviços não executados ao contar apenas as horas trabalhadas, sem medir os resultados efetivamente entregues. Além de apresentar os principais conceitos em torno de tais métricas, é explicado o método de análise de pontos de função para medir o tamanho de sistemas de informação. Mesmo controversa, a técnica é comumente adotada no setor público e foi utilizada neste trabalho como opção viável de métrica. Neste contexto, realizamos um estudo de caso no Tribunal de Contas do Estado de Goiás a fim de comprovar se a adoção de outras medidas baseadas em homem-hora gerou economia ou prejuízo na aplicação de recursos públicos.

Palavras-chave: Métricas de Engenharia de *Software*. Pontos de Função. Contratações de TI. Homem-Hora. Terceirização.

ABSTRACT

This paper analyzes the use of software engineering metrics by public agencies to remunerate the outsourcing of IT services, referring to the problem of paying for services not performed by counting only the hours worked, without measuring the results delivered. Are presented the main concepts around such metrics, also explained the method of function points analysis to measure the size of information systems. Even controversial, the technique is commonly adopted in the Brazilian public organizations, the method was suggested in this work as a viable metric option. In this context, we conducted a case study at the Court of Audit of the State of Goiás to verify whether the use of man hour metrics generated savings or losses in the application of public resources.

Keywords: Software Engineering Metrics. Function Points. Man hour. Software Development Outsourcing.

LISTA DE ABREVIATURAS

APF – Análise de Pontos de Função
ALI – Arquivo Lógico Interno
AIE – Arquivo de Interface Externa
ATRICON – Associação dos Membros dos Tribunais de Contas
BFPUG – *Brazilian Function Point Users Group* (Grupo Brasileiro de Usuários de Ponto de Função)
CPM – *Counting Practices Manual* (Manual de Práticas de Contagem)
CASE – *Computer-Aided Software Engineering* (Engenharia de *Software* Assistida por Computador)
CE – Consulta Externa
CF – Constituição Federal
CFB – Componente Funcional Básico
CGS – Características Gerais do Sistema
DER – Dados Elementares Referenciados
EE – Entrada Externa
GQM – G*oal-Question-Metric* (Objetivo-Questão-Métrica)
GNS – Gestão Nível de Serviço
IFPUG – *International Function Point Users Group* (Grupo Internacional de Usuários de Ponto de Função)
FP – *Function Point* (Ponto de Função)
FSM – *Functional Size Measurement* (Medição de Tamanho Funcional)
IEC – *International Electrotechnical Commission* (Comissão Eletrotécnica Internacional)
IEEE – Instituto de Engenheiros Eletricistas e Eletrônicos
ILB – Instituto Leopoldo de Bulhões
IN – Instrução Normativa
ISBG – *International Software Benchmarking Standards Group* (Grupo Internacional de Padrões de *Benchmarking* de *Software*)
ISO – *International Organization for Standardization* (Organização Internacional para Padronização)
MER – Modelo Entidade Relacionamento

MPOG – Ministério do Planejamento, Orçamento e Gestão

PF – Ponto de Função

RLR – Registros Lógicos Referenciados

SISP – Sistema de Administração dos Recursos de Tecnologia da Informação

SE – Saída Externa

SERPRO – Serviço Federal de Processamento de Dados

SEFTI – Secretaria de Fiscalização de Tecnologia da Informação

SLTI – Secretaria de Logística e Tecnologia da Informação

TCE – Tribunal de Contas do Estado

TCE-GO – Tribunal de Contas do Estado de Goiás

TCE-MT – Tribunal de Contas do Estado de Mato Grosso

TCE-PE – Tribunal de Contas do Estado de Pernambuco

TCE-SP – Tribunal de Contas do Estado de São Paulo

TCU – Tribunal de Contas da União

TI – Tecnologia da Informação

UML – *Unified Modeling Language* (Linguagem de Modelagem Unificada)

VAF – Valor do Fator de Ajuste

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

Desde o surgimento das primeiras técnicas de medição de sistemas, os gerentes de projeto e os engenheiros de *software* têm buscado encontrar uma métrica que seja de fácil compreensão e que reflita o real valor das aplicações. As métricas de *software* são utilizadas nos contratos públicos para remunerar os produtos entregues. Tanto estas métricas quanto as contratações têm evoluído nos últimos anos. Parte disso se deve às recomendações expedidas pelo TCU.

Neste contexto, o presente estudo aborda os conceitos fundamentais em torno das principais técnicas de medição utilizadas, destacando as dimensões comumente adotadas para medir o tamanho final de sistemas assim como as medidas relacionadas ao seu processo de desenvolvimento. A Análise de Pontos de Função é uma das técnicas apresentadas. Além disso, são descritos os problemas em torno de métricas baseadas em homem-hora.

Ainda hoje, muitas contratações realizadas pelo poder público são pagas apenas com base na quantidade de tempo e esforço gasto na execução dos serviços. Tal situação perdurou durante alguns anos em contratos de desenvolvimento de *software* mantidos pelo TCE-GO. Desta forma, surge a seguinte questão: nestes contratos, em que medida a utilização da métrica homem-hora trouxe prejuízos àquele órgão?

Este tipo de medida pode gerar prejuízos, pois permite que a falta de competência da equipe contratada tenha como consequência o pagamento por mais horas do que o necessário. Por sua vez, o ponto de função é rodeado de controvérsias, sendo visto como uma medida excessivamente burocrática e cujo processo de medição pode encarecer os custos, sendo assim, também capaz de gerar prejuízos. Assim sendo, a hipótese é que a forma como o TCE-GO encontrou para medir tais contratos pode ter gerado algum tipo de economia na aplicação dos recursos públicos, uma vez que a medição com base em homem-hora é de fácil gestão, permitindo mais agilidade e consequente economia.

O objetivo deste trabalho é demonstrar se o valor gasto pelo TCE-GO com o desenvolvimento de sistemas é compatível com os resultados efetivamente entregues pelas empresas contratadas. Neste contexto, procurou-se conhecer as principais técnicas existentes para medição do tamanho de *softwares*, quais fatores que despertam controvérsias e as recomendações do TCU sobre o assunto. Por fim,

busca-se propor uma métrica que permita avaliar e remunerar de forma justa e objetiva a entrega de produtos de *software*.

Os sistemas de informação estão presentes em quase todos os processos de trabalho do dia-a-dia, contudo, medir *software* ainda é um grande desafio para muitos entes públicos que terceirizam serviços de TI. Diante dos altos custos envolvidos neste tipo de contratação, é importante aprimorar os aspectos da medição. Isto significa gerir melhor os recursos financeiros. Neste sentido, os Tribunais de Contas surgem como instituições com a missão de resguardar o erário e devem não apenas orientar, mas também servir de modelo aos ordenadores de despesas que fiscalizam. Desta forma, os resultados desta análise poderão auxiliar os Tribunais de Contas e seus jurisdicionados a aperfeiçoar as contratações públicas de serviços de TI, trazendo benefícios à sociedade ao evitar o desperdício de recursos e outros prejuízos decorrentes da despesa pública antieconômica, o que se torna ainda mais relevante no atual tempo de austeridade.

A fundamentação teórica foi realizada com base em pesquisa de bibliografia relacionada à engenharia de *software*. Os conceitos de métricas descritos por Pressman e Sommerville serviram de guia para este estudo. Ademais, é apresentado o compêndio mantido pelo IFPUG para orientar a medição de tamanho funcional, bem como as principais decisões do TCU sobre contratações de TI.

Para o estudo de caso, foi-se a campo para reunir toda documentação e então medir o tamanho dos sistemas produzidos no TCE-GO. Em seguida, calcular qual seria o valor a ser pago utilizando a medida em pontos de função. A análise dos dados permitirá uma comparação das estimativas com o montante gasto pelo TCE-GO e avaliar se os valores foram compatíveis com os praticados por outros tribunais em contratos da mesma natureza.

O presente trabalho está dividido em cinco capítulos. Inicialmente, são apresentados os conceitos e definições em torno das métricas de *software*, abordando o que são processos de medição eficazes e os desafios envolvidos. O capítulo seguinte explica a análise de pontos de função, trazendo o passo-a-passo do processo de contagem. Os fatores que afetam a produtividade das equipes de desenvolvimento de *software* são descritos em seguida. Outro capítulo trata da utilização de métricas no setor público, entre elas as medidas baseadas na quantidade de homem-hora e as implicações do seu uso. O último capítulo descreve a metodologia e os resultados obtidos neste estudo de caso.

2 MÉTRICAS DE *SOFTWARE*

2.1 Conceitos e definições

No campo das ciências das exatas, as atividades de medição são corriqueiras. Aquilo que se pode medir é mensurável. Para determinar uma medida é necessário que de antemão haja uma base de referência, ou seja, um padrão de medição que possa ser usado para determinar o valor e o tamanho de certa grandeza. Há diversos padrões de referência que utilizamos no nosso dia-a-dia para medir pesos, alturas, comprimentos.

Contudo, esta não é a realidade da engenharia de *software*. Pressman (2011) aduz que mesmo se tratando de uma disciplina quantitativa, a engenharia de *software* não é fundamentada nas leis da física. O autor ressalta que diferentemente das matérias que lidam com medidas diretas, tais como tensão, massa, velocidade ou temperatura, o mundo do *software* trata de medidas indiretas, sendo assim, abertas a discussão. (PRESSMAN, 2011).

2.1.1 Medida, medição, métrica e indicadores

Sommerville (2005) define medição de *software* como a etapa do seu ciclo de vida que busca atribuir valor numérico para alguns dos atributos de um produto ou de um processo de *software*, de forma que a partir de padrões estabelecidos se possa tirar conclusões sobre a qualidade do *software* produzido ou dos procedimentos seguidos durante sua construção. O processo de medição é, portanto, o ato de atribuir determinada medida.

Neste mesmo contexto, Pressman (2011, p. 539) enfatiza que enquanto a medida é a “indicação quantitativa de extensão, quantidade, capacidade ou tamanho de algum atributo de um produto ou processo”, a métrica é por sua vez, a medida quantitativa do grau que um sistema ou processo possui de determinado atributo. Ao seu turno, os indicadores são construídos a partir da coleta de métricas, permitindo aos gerentes de projeto ou engenheiros de *software* determinar quais as melhorias necessárias. (PRESSMAN, 2011, p. 539).

Os indicadores e as métricas podem ser obtidos em diversas esferas que envolvem o desenvolvimento de um sistema. Utilizam-se métricas e medidas para estimar o tamanho que poderá ter um sistema ainda na fase de concepção do seu

projeto. Ao longo do processo de construção, podem-se também coletar dados para aferir quão aderente os procedimentos estão às boas práticas estabelecidas para este fim. As medições ocorrem em todas estas etapas, inclusive quando da sua entrega, onde é possível verificar se o que foi estimado equivale ao que foi produzido. Assim, de acordo com a etapa em que ocorre a medição, podem ser definidas métricas de produto ou métricas de processo.

2.1.2 Métricas de produto

As métricas de produto dizem respeito às características do sistema em si, que abrangem quesitos como o seu tamanho tanto no que se refere à quantidade de linhas de código ou caminhos de processamento quanto à quantidade e complexidade de suas funcionalidades. Sommerville (2005) distingue métricas de produtos em métricas dinâmicas (coletas que podem ser automatizadas com programas de computador) e métricas estáticas (coletas realizadas a partir de representações do sistema, tais como seu projeto ou sua documentação).

Algumas destas características guardam pouca relação com seus aspectos qualitativos. Um *software* com poucas linhas de código pode conter as mesmas funcionalidades e produzir os mesmos resultados que outro com milhares de linhas. Dessa forma, Sommerville (2005) defende que é preciso analisar a relação existente entre as métricas de produto do sistema com seus atributos de qualidade externos.

2.1.3 Métricas de processo e projeto

As métricas de processo e projeto possibilitam a compreensão do conjunto de atividades realizadas para a construção de um sistema. Pressman (2011, p. 583) define-as como “medidas quantitativas que permitem que você tenha o discernimento sobre a eficácia do processo de *software*”. O processo de *software* é o conjunto de atividades com o objetivo de desenvolver ou aprimorar um sistema de informação e/ou a sua documentação. Os processos, em geral, seguem um modelo específico. Os modelos de processo de *software*[1] são representações abstratas das

[1] Encontramos nos capítulos iniciais das obras de Pressman e Sommerville os principais modelos de processo de *software* e suas características.

atividades, papéis e artefatos que permeiam o desenvolvimento de um sistema de informação. Em geral, envolvem etapas de:

1) **Especificação/Definição de requisitos** – fase de levantamento do que o sistema deve fazer (funcionalidade) e quais as restrições;

2) **Desenvolvimento/Implementação** – consiste na produção do *software*, a escrita de códigos utilizando linguagens de programação;

3) **Homologação/Verificação** – etapa de validação do produto entregue quanto aos requisitos funcionais e não funcionais;

4) **Manutenção** – compreende as mudanças evolutivas, adaptativas ou corretivas realizadas após a entrega do *software*.

Ao comparar as médias obtidas a partir de medições realizadas em cada etapa executada de um projeto concluído, os resultados podem servir como parâmetro para aprimorar aspectos como qualidade e produtividade em novos projetos. Por meio da análise dos dados coletados é possível detectar desvios aos padrões estabelecidos para cada atividade, permitindo assim a melhoria contínua do processo de *software* utilizado por uma organização.

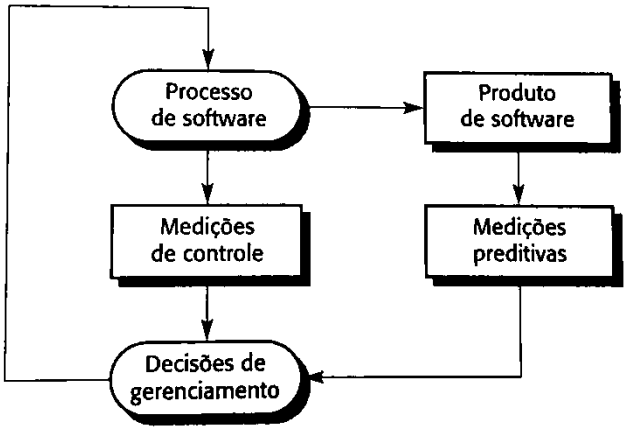
Neste sentido, as métricas de processo são utilizadas para fins estratégicos, já as métricas de projeto são consideradas métricas táticas, sendo utilizadas pelo gerente e pela equipe do projeto para estimar, monitorar e controlar o progresso das atividades. (PRESSMAN, 2011). Pressman (2011, p. 543) defende a utilização de métricas de produto para estimar o tamanho de projetos: “métricas de produto que proporcionem informações sobre a qualidade do modelo de análise são desejáveis”. O autor explica que é possível adaptar métricas que examinam o modelo de requisitos para prever o tamanho do *software* a ser construído, ainda que esta medida não seja sempre o indicador de complexidade mais apropriado. Uma das poucas métricas que são utilizadas dessa forma é o ponto de função, apresentada com detalhes no próximo capítulo.

2.1.4 Outras classificações para métricas e medidas

De acordo com Sommerville (2005), as medidas de *software* também podem ser classificadas em métricas de controle e métricas preditivas. Enquanto as medições que visam o controle se originam em métricas de processo e projeto, as métricas de produto são associadas a medições preditivas. Ambas irão influenciar o

processo de gerenciamento e tomada de decisão, conforme nos mostra a figura a seguir.

Figura 1 – Métricas preditivas e de controle



Fonte: Sommerville, 2005.

Enquanto algumas medidas de *software* podem ser obtidas de forma direta, outras só podem ser mensuradas de maneira indireta. Medidas diretas do processo de *software* podem incluir aspectos relacionados a custos e tempo gasto. Medidas diretas do produto dizem respeito à quantidade de linhas de código, tempo de execução, tamanho da memória. (PRESSMAN, 2011). Já as medidas indiretas se referem à qualidade e funcionalidade do *software*, envolvendo assim características de difícil aferição, tais como complexidade, eficiência, confiabilidade. (PRESSMAN, 2011).

Na engenharia de *software* é possível explorar uma infinidade de conceitos que possuem abordagem específica de métricas. Existem métricas orientadas a objetos, classes, componentes, operações, funções, tamanho. Há ainda métricas específicas para interfaces de usuário, manutenção, correção e testes. Esta grande quantidade de dimensões é um dos grandes problemas encontrados para que se possa medir *software* com precisão. É o que relata Fenton (1994), que é citado por Pressman (2011, p. 540): “O perigo de tentar encontrar medidas que caracterizem tantos atributos diferentes, é que inevitavelmente as medidas têm de satisfazer interesses em conflito. Isso é contrário à teoria representacional da medição”. É necessário então estabelecer métodos eficazes de medição.

2.2 Estabelecendo processos de medição eficazes

Assim como cada *software* possui complexidade própria, cada organização que produz sistemas também terá suas peculiaridades. Ainda assim é possível estabelecer processos de medição que permitirão a avaliação do que é produzido. Sommerville (2005, p. 470) descreve as etapas básicas de um processo de medição:

> 1. Escolha de medições a serem feitas
> Devem ser formuladas as questões às quais as medições se destinam a responder, e as medições exigidas para responder a essas questões devem ser definidas. As medições que não forem diretamente relevantes para essas questões não precisam ser coletadas. O paradigma de GQM (goal-question-metric — objetivo-questão-métrica), de Basili (Basili e Rombach, 1988), discutido no capítulo seguinte, é uma boa abordagem, que deve ser utilizada quando se decide que dados devem ser coletados.
> 2. Seleção de componentes a serem avaliados
> Pode não ser necessário ou desejável avaliar valores de métricas para todos os componentes em um sistema de *software*. Em alguns casos, uma seleção representativa de componentes pode ser escolhida para a medição. Em outros casos, podem ser avaliados os componentes que são particularmente importantes, como os componentes centrais, que estão em uso quase constante.
> 3. Medição de características dos componentes
> Os componentes selecionados são medidos e os valores de métricas são computados. Isso envolve processar a representação dos componentes (projeto, código etc.) utilizando uma ferramenta automatizada de coleta de dados. Isso pode ser especialmente escrito ou já pode estar incorporado nas ferramentas CASE, que são utilizadas em uma organização.
> 4. Identificação de medições anômalas
> Uma vez feitas as medições de componentes, elas devem ser comparadas entre si e com as medições precedentes, que tenham sido registradas em um banco de dados de medições. É preciso procurar valores altos ou baixos incomuns para cada métrica, uma vez que isso sugere que pode haver problemas com o componente que apresenta esses valores.
> 5. Análise de componentes anômalos
> Uma vez identificados os componentes com valores anómalos em métricas particulares, esses componentes devem ser examinados para decidir se os valores anômalos significam que a qualidade do componente está comprometida ou não. Um valor anômalo para complexidade (digamos) não significa necessariamente um componente de baixa qualidade. É possível que haja alguma outra razão para o valor alto, e pode não significar que haja problemas com a qualidade do componente. (SOMMERVILLE, 2005, p. 470).

Os estágios acima descritos guardam relação íntima com as cinco atividades descritas por Roche (1994) conforme citado por Pressman (2011, p. 540):

> - Formulação. A criação de medidas e métricas de software apropriadas para a representação do *software* considerado.
> - Coleção. O mecanismo usado para acumular dados necessários para criar as métricas formuladas.
> - Análise. A computação das métricas e a aplicação de ferramentas matemáticas.
> - Interpretação. A avaliação de métricas que resultam em informações sobre a qualidade da representação.
> - Feedback. Recomendações derivadas da interpretação de métricas de produto transmitidas para a equipe de *software*. (PRESSMAN, 2011, p. 540).

Para Roche (1994, apud PRESSMAN, 2011), tais atividades devem obedecer alguns princípios básicos tais como a coleta automática sempre que possível e o estabelecimento de relações válidas entre atributos internos e fatores de qualidade externos. Pressman (2011, p. 540) acrescenta que “as métricas de *software* só serão úteis se forem efetivamente caracterizadas e validadas de forma que demonstrem valer a pena”. Segundo Sommerville (2005), a relevância de métricas específicas é relativa e depende de cada projeto, das metas estabelecidas, das características presentes no processo de gerenciamento da qualidade e do tipo de *software*, de forma que haverá situações em que algumas métricas podem ser ou não apropriadas.

Para que se possa escolher o que medir, Basili e Rombach (1994) propõem o paradigma GQM (*Goal-Question-Metric* ou Meta/Questão/Métrica). Essa metodologia consiste em estabelecer metas fundamentadas em um levantamento de quais necessidades podem ser medidas, sejam elas relacionadas ao produto ou ao processo. Dessa forma é possível definir o que realmente deve ser medido e como utilizar as métricas escolhidas.

2.3 Os desafios de medir *software*

Guarizzo (2008) explica que a medição tem um papel relevante na engenharia de *software* e que isso independe do método adotado. Para a autora, sem métricas expressas em números, o que nos resta são apenas dados subjetivos. Nada obstante, ela observa que:

> Em geral os engenheiros de *software* adotam uma abordagem sistemática e organizada em seu trabalho, uma vez que essa é, com

> frequência, a maneira mais eficaz de produzir *software* de alta qualidade. No entanto a engenharia tem a ver, em grande parte, com a questão de selecionar o método mais apropriado para um conjunto de circunstâncias, e uma abordagem mais criativa e informal para o desenvolvimento pode ser eficaz em algumas circunstâncias. (GUARIZZO, 2008, p.3).

Estas abordagens criativas decorrem da necessidade de superar os obstáculos de medir sem um padrão de referência que não seja subjetivo. Neste sentido, os desafios se amplificam quando observamos que as metodologias de medição existentes têm seu foco voltado exclusivamente para medidas indiretas, tais como a qualidade.

Avaliar a qualidade é o principal argumento de Pressman (2011, p. 597) para defender a utilização de métricas de *software*: “Se você não medir, não haverá uma maneira real de determinar se está melhorando”. Somente avaliando medidas de produtividade e qualidade é que se pode estabelecer objetivos e controles adequados para gerir o desenvolvimento de *software*.

Pressman (2011) afirma que é um erro pensar que o *software* é algo incomensurável e que embora imperfeitas, as métricas existentes podem propiciar uma forma organizada de se avaliar a qualidade dos produtos gerados nesta disciplina. Sommerville (2005, p. 468) destaca que “o uso da medição e de métricas sistemáticas de *software* ainda é relativamente incomum. Existe uma relutância em introduzir a medição, porque os benefícios não são bem definidos”. Para ele, ainda não existem padrões para estas métricas e isto limita a coleta e a análise de dados que subsidiem o processo de contagem e medição. (SOMMERVILLE, 2005).

É fácil medir quando se tem uma base sólida e amplamente difundida. Com uma régua é possível detalhar pequenas medidas em milímetros. De igual forma, as longas distâncias são facilmente compreendidas, mesmo que correspondam a milhares de quilômetros. No entanto, mesmo depois de mais 40 anos de estudo e pesquisa ainda não foi estabelecida uma técnica que possibilite medir o *software* de maneira clara e simples de se computar. Apesar de existirem conexões intuitivas entre as características internas dos produtos de *software,* seus aspectos externos e de seu processo, houve poucos resultados científicos capazes de estabelecer relações mais específicas, não por falta de tentativa, mas pelo contrassenso de fazer este tipo de experimento. (FENTON, 1991 apud PRESSMAN, 2011).

Sommerville (2005, p. 471) lembra que “desde o início da década de 90, tem havido uma série estudos de métricas orientadas a objetos”. Para ele, estas métricas são menos desenvolvidas que as métricas orientadas a funções e por sua vez sua capacidade preditiva ainda é restrita. (SOMMERVILLE, 2005). Este mesmo autor relata que o problema com a coleta de dados quantitativos sobre os projetos de sistemas de informação está justamente em entender o que estes dados realmente significam. (SOMMERVILLE, 2005). Sommerville (2005) afirma que se medirmos a quantidade de modificações solicitadas para determinado sistema em certo período de tempo, podemos interpretar as resultados sobre diferentes perspectivas. Caso haja grande número de solicitações:

- O sistema foi construído em desacordo com as especificações idealizadas pelos seus requisitantes – necessidade de correções;
- Ou, o sistema foi construído de acordo com as necessidades, porém os requisitos é que se modificaram, sendo necessária a adaptação do sistema;
- Ou ainda, após receberem o sistema, os requisitantes perceberam que novas funcionalidades poderão ser incluídas a fim de atender novas necessidades que não foram incluídas inicialmente (evolução).

Caso haja poucas solicitações de modificação:

- O sistema foi construído perfeitamente de acordo com as necessidades elencadas, não havendo qualquer mudança nos requisitos ou correções necessárias;
- Ou, os requisitantes receberam o sistema, mas não utilizaram de fato a ferramenta, estando a sua validação prejudicada.
- Ou ainda, os usuários do *software* preferem conviver com as falhas existentes no sistema porque acreditam que nunca serão corrigidas.

Percebe-se assim, que há uma infinidade de interpretações possíveis para um único indicador. A incerteza torna-se um desafio e impõe que o processo de medição seja munido de critérios razoáveis e válidos, sob pena de se tornarem questionáveis ou imprecisos. Segundo Ejiogu (1991) e reproduzido por Pressman (2011, p. 542), para que uma métrica de *software* seja efetiva, ela deve satisfazer a alguns requisitos. Métricas devem ser:

- *Simples e computáveis.* Deverá ser relativamente fácil aprender a derivar a métrica, e sua computação não deve demandar esforço ou tempo fora do normal.
- *Empiricamente e intuitivamente persuasiva.* A métrica deverá satisfazer as ideias do engenheiro sobre o atributo do produto considerado (por exemplo, uma métrica que mede coesão de módulo deverá crescer em valor na medida em que aumenta o nível de coesão).
- *Consistente e objetiva.* A métrica deverá sempre produzir resultados que não sejam ambíguos. Um terceiro independente deverá ser capaz de derivar o mesmo valor da métrica usando as mesmas informações sobre o *software*.
- *Consistente no seu uso das unidades e dimensões.* A computação matemática da métrica deverá usar medidas que não resultem em combinações bizarras de unidades. Por exemplo, multiplicar número de pessoas nas equipes de projeto pelas variáveis da linguagem de programação no programa resulta em uma mistura duvidosa de unidades que não é claramente convincente.
- *Independente da linguagem de programação.* As métricas deverão ser baseadas no modelo de requisitos, modelo de projeto ou na própria estrutura do programa. Elas não deverão ser dependentes dos caprichos da sintaxe ou semânticas das linguagens de programação.
- *Um mecanismo efetivo para feedback de alta qualidade.* A métrica deverá fornecer informações que podem levar a um produto final de melhor qualidade. (EJIOGU, 1991 apud PRESSMAN, 2011, p. 542).

Entretanto, o que se observa é que muitas métricas não possuem todos estes atributos. As medidas diretas são objetivas, simples e facilmente computáveis, mas podem variar de acordo com a linguagem. As métricas indiretas exigem contagens complexas e o resultado pode variar ainda que sejam utilizados os mesmos parâmetros e informações sobre o *software*, como é o caso do ponto de função.[2]

Guarizzo (2008, p. 14) destaca outras barreiras que muitas organizações enfrentam quando o assunto é a utilização de métricas de *software*:

- Falta de comprometimento da alta gerência
- Medir custa caro
- Os maiores benefícios vêm a longo prazo
- Má utilização das métricas
- Grande mudança cultural necessária
- Dificuldade de estabelecer medições apropriadas e úteis
- Interpretações de dados realizadas de forma incorreta
- Obter o comprometimento de todos os envolvidos e impactados
- Estabelecer um programa de medições é fácil, o difícil é manter (GUARIZZO, 2008, p. 14).

[2] Para mais detalhes sobre Ponto de Função veja o próximo capítulo deste trabalho.

3 ANÁLISE DE PONTOS DE FUNÇÃO

3.1 Medidas relacionadas à função

É possível medir um sistema por ângulos diferentes. Conforme demonstrado no capítulo anterior, as métricas de *software* podem ocupar-se de diversas caraterísticas distintas, entre elas o tamanho funcional. Isto significa medir a complexidade do sistema com base na contagem de suas funcionalidades. Estas métricas são conhecidas como medidas orientadas à função.

Sommerville (2005) explica o que as medidas relacionadas a funções podem ser mais eficazes do que as métricas relacionadas a outros parâmetros, tais como a quantidade de linhas de código ou número de instruções. Isto porque, quando avaliamos a produtividade, o tipo de linguagem utilizado para construir o *software* pode alterar significativamente o resultado da avaliação, ou seja, seu tamanho.

Ainda nos primórdios da era da informação, as instruções dos programas de computador eram registradas em cartões perfurados, evoluindo rapidamente para linguagens cada vez mais sofisticadas. Se antes cada linha de código ou cartão significava uma instrução ao programa, hoje um pequeno trecho pode conter instruções complexas ou apenas simples comentários para auxiliar o desenvolvedor a situar-se. Dessa forma, não se pode medir com eficácia o tamanho de um sistema ou a produtividade para construí-lo, se observarmos apenas este tipo de parâmetro quantitativo.

O tamanho funcional, tratado pelas métricas orientadas à função permite medir um *software* de acordo com os seus requisitos. Dekkers (1998) define o tamanho funcional da seguinte forma:

> Tamanho funcional é uma medida de tamanho de *software*, baseada em uma avaliação padronizada dos requisitos lógicos dos usuários. Na indústria há atualmente várias maneiras para se medir tamanho funcional, a mais antiga das quais são os pontos de função. Semelhante aos metros quadrados de uma casa, pontos de função são independentes dos métodos físicos, ferramentas ou linguagem de desenvolvimento utilizados para construir o *software*.

Os requisitos referem-se às características de um sistema. Tais requisitos podem ser classificados como funcionais ou não funcionais. Os requisitos não funcionais dizem respeito a restrições, já os requisitos funcionais tratam de recursos, funcionalidades e serviços que um sistema deve oferecer ou como deve reagir

diante de determinadas situações. (SOMMERVILLE, 2005).

Em geral, são os usuários que apresentam os requisitos funcionais ao descrever quais são as necessidades que o sistema irá suprir. Assim, o tamanho funcional é medido a partir da visão do usuário. Este novo paradigma de medição de *software* a partir desta visão, ganhou força no final da década de 70, especialmente com o surgimento da técnica denominada Análise de Pontos de Função (APF). Campos (2011, p. 30) enfatiza que a métrica de pontos de função “está diretamente relacionada à visão de negócio do usuário e dependente dela”.

3.1 Breve histórico

Os conceitos sobre Pontos de Função foram inicialmente apresentados por Allan Albrecht na conferência SHARE/GUIDE/IBM Application Development Symposium em 1979 (DEKKERS, 1998). Pouco tempo depois, estes conceitos foram refinados e apresentados em uma metodologia formal (ALBRECHT; GAFFNEY, 1983). Logo em seguida, formou-se uma comunidade de usuários para promover padronizações nas regras de contagem. Surgiu então o Grupo Internacional de Usuários de Pontos de Função International Function Point Users Group (IFPUG), entidade sem fins lucrativos que foi formalmente instituída em 1986. (DEKKERS, 1998).

O IFPUG possui voluntários espalhados em diversos países. Eles são responsáveis pelo estabelecimento e publicação de diversos documentos amplamente reconhecidos como referência no assunto, entre eles o Manual de Práticas de Contagem – Counting Practices Manual(CPM) atualmente na versão 4.3.1, o Guia Para Medição de *Software* (The IFPUG Guide to IT and Software Measurement) e diversos Estudos de Casos detalhados (IFPUG, 2010).

A Análise de Pontos de Função (APF) é uma das técnicas de medição de *software* mais consolidadas e evoluiu bastante desde que foi criada. Wolfart (2012, p. 27-28) relata:

> No começo da década de 90 diante do grande número de métodos de Medição Funcional de tamanho (*Functional Size Measurement* - FSM), surgidos a partir da Análise de Pontos de Função proposto por *Albrecht* e com o objetivo de acabar com a inconsistência existente entre esses métodos e criar um método mais rigoroso de medição funcional, grupos de trabalho de métricas de *software* da Austrália,

> Reino Unido, Holanda e Estados Unidos formaram um grupo de trabalho denominado WG12 (*Working Group* 12), subordinado ao SC7 – *Sub-Committee Seven* do JTC1 – *Joint Technical Committee* One estabelecido pela ISO – *International Organization for Standardization* em conjunto com o IEC – *International Engineering Consortium* (VAZQUEZ et al., 2010; ASMA, 1999).
> Como resultado dos trabalhos do WG12, foi estabelecido um conjunto de padrões internacional chamado de norma 14143 (ISO/IEC 14143). A norma ISO/IEC 14143 foi estabelecida para garantir que todos os métodos de medição funcional sejam baseados em conceitos semelhantes, e possam ser testados para assegurar o comportamento similar e da forma esperada pelo método, dependendo dos domínios funcionais a que se aplicam (VAZQUEZ et al., 2010).
> No fim de 2002 a técnica de Análise de Pontos de Função mantida pelo IFPUG, foi aprovado como um método de medição funcional dentro dos padrões exigidos pela norma ISO/IEC 14143 sob a denominação ISO/IEC 20926:2002. Porém, essa técnica é contemplada apenas até a determinação dos pontos de função não ajustados. As características gerais do manual de contagem do IFPUG, que são utilizadas para estabelecer o fator de ajuste, possuem requisitos tecnológicos e de qualidade o que tornam essa parte do processo excluída do padrão de medição funcional da ISO. (WOLFART, 2012, p. 27-28).

Jones (2012) destaca que a métrica de pontos de função tem demonstrado seu valor, sendo uma referência em diversos estudos, e ganhado cada vez mais adeptos, inclusive em contratos firmados pelos governos do Brasil e da Coréia do Sul. De igual forma, algumas variações do método do IFPUG têm sido bastante difundidas, entre elas as técnicas COSMIC Function Points, NESMA Function Points, FISMA Function Points, entre outros. (JONES, 2012).

3.2 Visão geral sobre pontos de função

O ponto de função é a unidade de medida da APF. Pressman (2011) afirma que a métrica de ponto de função pode ser utilizada tanto para estimar os custos quanto para prever a quantidade de componentes que o *software* terá. Cada ponto de função equivale a um conjunto de características associadas às funcionalidades do *software*. O total de pontos de função é obtido ao medir ou estimar cada funcionalidade em conjunto com suas respectivas entradas e saídas internas ou externas, interações, interfaces e arquivos utilizados pelo sistema. (SOMMERVILLE, 2005).

O método proposto pelo IFPUG (2010) estabelece que devem ser medidos e

quantificados apenas os requisitos funcionais do usuário. Cada funcionalidade específica da aplicação é avaliada sob a ótica do que é efetivamente entregue ao usuário, e não como foi entregue. O tamanho funcional é obtido por meioda medição das funções de dados e de transações e é apresentado pela quantidade de pontos de função encontrados ou estimados.

Sobre o total de pontos de função podem ser aplicados fatores de ajuste de acordo com a complexidade definida, de forma que o resultado é obtido a partir de uma relação empírica que pondera medidas diretas com avaliações qualitativas de complexidade. (PRESSMAN, 2011). Atualmente, o CPM prevê a aplicação opcional do fator de ajuste (VAF) considerando características gerais de sistema (CGSs), porém este fator não foi recepcionado pelo padrão de FSM estabelecido na norma internacional ISO/IEC 14143-1:2007. (IFPUG, 2010). O VAF pode aumentar ou diminuir o total de pontos de função em aproximadamente 35% de acordo com o nível de influência de cada característica. (CAMPOS, 2011).

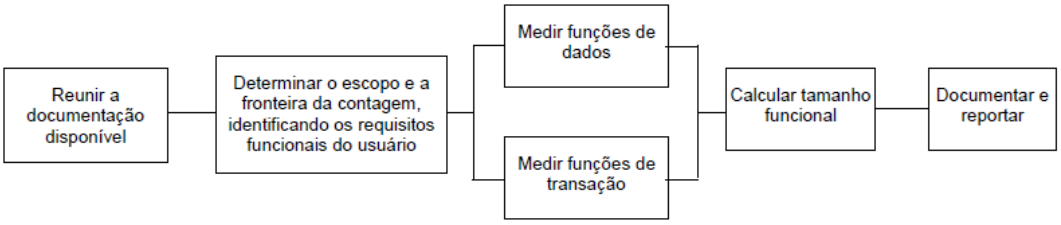
Wolfart (2012) relembra que a APF é atualmente um método bastante consolidado, sendo utilizado amplamente por profissionais e em empresas no Brasil. Além da possibilidade de utilizar pontos de função para conceber estimativas, o método é consolidado como uma unidade de medição corriqueira em contratos de desenvolvimento de *software* firmados pelo poder público.[3]

3.2 O processo de medição por meio da contagem de pontos de função

O processo de medição utilizando APF está descrito no Manual de Práticas de Contagem de Pontos de Função (CPM). (IFPUG, 2010). Trata-se de um documento que traz em detalhes como funcionam todos os procedimentos de contagem, inclusive com exemplos práticos. A contagem de pontos de função possui regras e conceitos que devem ser observados para que o resultado seja fidedigno. Apesar de simples, trata-se de um processo relativamente longo, onde termos e definições devem seguir os preceitos e a sequência estabelecida. (CAMPOS, 2011). A figura 2 demonstra as etapas que compõem o processo de contagem da APF.

[3] Este fator foi decisivo para que esta medida fosse adotada como parâmetro no estudo de caso apresentado neste trabalho, conforme abordado no capítulo de número 5.

Figura 2 – Visão geral do processo de contagem de pontos de função



Fonte: CPM (IFPUG, 2010).

Campos (2011, p. 38) enfatiza que “para conduzir uma contagem de pontos de função devem ser executadas as atividades de cada passo, conforme diagrama apresentado” buscando identificar os Componentes Funcionais Básicos (CFB). Determinar quais são estes componentes de acordo com a visão do usuário é crucial para que a contagem seja precisa. Os componentes funcionais básicos são unidades elementares do processo de medição e podem se referir a funções de transformação ou de armazenamento de dados. (CAMPOS, 2011).

3.2.1 Passo 1 – Reunir a documentação

O manual oficial (IFPUG, 2010, p. 9-10) estabelece a primeira etapa para iniciar o processo de medição em pontos de função:

> A documentação de suporte a uma contagem de pontos de função deve descrever a funcionalidade entregue pelo *software* ou a funcionalidade impactada pelo projeto de *software* medido.
> Deve ser obtida documentação suficiente para conduzir a contagem de pontos de função, ou acesso a especialistas no assunto capazes de fornecer informações adicionais para suprir quaisquer falhas na documentação. (IFPUG, 2010, p. 9-10).

Esta documentação refere-se a informações sobre requisitos, modelos de dados, diagramas e pode incluir ainda, descrições de procedimentos e funcionalidades, tais como relatórios e manuais, assim como outros artefatos relacionados ao *software*. (IFPUG, 2010). De forma que o primeiro passo é reunir e avaliar toda a documentação disponível. (CAMPOS, 2011).

3.2.2 Passo 2 – Delimitar o escopo, a fronteira da contagem e identificar os requisitos

O passo seguinte refere-se às atividades para delimitar o escopo e a fronteira da contagem, o que inclui identificar o propósito e o tipo de contagem. O propósito diz respeito à questão que a contagem irá responder. (IFPUG, 2010). Pode referir-se a uma estimativa que se busca antes de iniciar um projeto ou a aferição do tamanho de um sistema entregue.

De acordo com o CPM, as contagens de pontos de função podem ser classificadas em três tipos: de aplicação, de projeto de desenvolvimento, ou de projeto de melhoria. (IFPUG, 2010). Campos (2011) explica que a contagem de pontos de função de aplicação diz respeito à medida das funções de uma aplicação já instalada e entregue ao usuário. Já a contagem de pontos de função de projetos, medem as funcionalidades que serão entregues na conclusão de uma nova ferramenta ou de uma melhoria promovida em uma aplicação já existente. (CAMPOS, 2011).

Definidos o propósito e o tipo de contagem, é necessário então delimitar o escopo da contagem e a fronteira da aplicação. O Roteiro de Métricas do SISP (BRASIL, 2015) estabelece que o escopo é o que identifica quais funcionalidades serão incluídas na contagem, enquanto a fronteira da aplicação pode ser definida com uma delimitação conceitual da aplicação de acordo com a visão do usuário.

A fronteira da aplicação não depende da forma de implementação. As aplicações devem ser medidas de acordo com os processos de negócios, de forma que o limite lógico pode avançar a outras aplicações ou limitar-se a apenas a alguns módulos do sistema. Assim, considerando o escopo de contagem definido e a percepção de quem utiliza o sistema, haverá o posicionamento desta fronteira e isto pode afetar diretamente o total de pontos de função. (BRASIL, 2015).

O escopo relaciona-se ainda como os requisitos do usuário. Neste sentido, o CPM ressalta que a despeito da possibilidade de subsistir uma miscelânea de requisitos funcionais e não funcionais, é necessário “identificar quais requisitos são funcionais e excluir os não funcionais” do processo de medição. (IFPUG, 2010, p. 10). Desta maneira, não são levados em conta aspectos relevantes, tais como necessidade de acessibilidade ou desempenho, características que podem impactar no processo de desenvolvimento.

3.2.3 Passo 3 – Medir as funções

3.2.3.1 Medir funções de dados

Superados estes passos preliminares, inicia-se a contagem pelas funções de dados. Segundo o CPM, as funções de dados são aquelas cuja funcionalidade atende a requisitos funcionais de armazenamento ou de referência aos dados, denominadas respectivamente de ALI e AIE. (IFPUG, 2010). Wolfart (2012, p. 52-53) diferencia estas funções em face do seu local de armazenamento: "ALI é mantido dentro da fronteira da aplicação, enquanto que o AIE é apenas referenciado pela aplicação, mas mantido dentro da fronteira de outra aplicação", em seguida a autora destaca os tipos de funções de dados:

> - **Arquivo Lógico Interno (ALI):** É um grupo de dados ou informações de controle logicamente relacionados e identificados pelo usuário que são mantidos (adicionados, alterados ou excluídos) através de um ou mais processos elementares da aplicação sendo contada.
> - **Arquivo de Interface Externa (AIE):** É um grupo de dados ou informações de controle logicamente relacionados e identificados pelo usuário que são referenciados (lidos) por meio de um ou mais processos elementares dentro da fronteira da aplicação sendo contada. O AIE de uma aplicação é obrigatoriamente um ALI de outra aplicação. (WOLFART, 2012, p. 52-53).

O CPM descreve como classificar as funções de dados e seus passos preliminares. (IFPUG, 2010). Para definir uma função de dados como ALI ou AIE é necessário ter em mente que os modelos de dados a serem considerados para contagem dos pontos de função não se confundem com os diagramas de entidade-relacionamento utilizados na modelagem de bancos de dados.[4]

Cada função de dados é composta pelas informações que se relacionam conforme seu reconhecimento pelo usuário e não necessariamente conforme os relacionamentos que possuem as tabelas. Excluem-se todos os atributos não requeridos pelo usuário, tais como as denominadas chaves estrangeiras, conforme descrito no item 5.4.2 do CPM. (IFPUG, 2010).

O IFPUG (2010) prescreve ainda que, para cada função de dados é preciso

[4] Um estudo sobre o modelo entidade-relacionamento está disponível na obra de Elmasri e Navathe. ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistemas de Banco de Dados**. 6. ed. Pearson Education.

contar a quantidade de Dados Elementares Referenciados (DERs) e Registros Lógicos Referenciados (RLRs). Cada atributo único que pode ser devidamente identificado pelo usuário e que não se repete equivalerá a um DER. As informações podem ser classificadas como um único DER ou como vários DERs. Este agrupamento irá depender de como os processos elementares utilizam os atributos dentro da aplicação, como podemos observar no exemplo trazido pelo CPM:

> EXEMPLO Os atributos (primeiro nome, nome do meio, sobrenome) são agrupados e contados como
> — nome (primeiro nome, nome do meio, sobrenome) se esses atributos sempre forem utilizados juntos,
> — primeiros nomes (primeiro nome e inicial do meio) e sobrenome se, além do que está acima, o sobrenome for utilizado independentemente, ou
> — primeiro nome, inicial do meio e sobrenome, se os três puderem ser utilizados independentemente. (IFPUG, 2010, p. 12)

Por padrão, conta-se 01 RLR para cada função de dados. RLR é um subgrupo de dados reconhecido pelo usuário dentro de uma função de dados. Além disso, um RLR adicional é contado para cada subgrupo que possa ser identificado para manter relacionamentos entre duas ou mais funções. (IFPUG, 2010). Por exemplo, em uma mesma função de dados podem existir os dados do cliente e os dados de itens do pedido, resultando portanto, em pelo menos 02 RLRs. O CPM define uma relação entre o total de DERs e RLRs para estabelecer a complexidade das funções de dados, conforme o quadro 1. A quantidade de pontos de funções varia de acordo com complexidade da ALI ou AIE, como demonstrado no quadro 2.

Quadro 1 – Complexidade das funções de dados

		DERs		
		1 – 19	20 – 50	> 50
RLRs	1	Baixa	Baixa	Média
	2 – 5	Baixa	Média	Alta
	> 5	Média	Alta	Alta

Fonte: IFPUG (2010)

Quadro 2 – Quantidade de PF das funções de dados

		Tipo	
		ALI	AIE
Complexidade funcional	Baixa	7	5
	Média	10	7
	Alta	15	10

Fonte: IFPUG (2010).

3.2.3.2 Medir funções de transação

As funções de transação são aquelas que se referem a funcionalidades que realizam o processamento de informações, sejam elas associados ou não a uma função de dados. (CAMPOS, 2011). O IFPUG (2010) distingue as funções de transação em Entrada Externa (EE), Saída Externa (SE) ou Consulta Externa (CE). O Roteiro de Métricas do SISP conceitua cada uma dessas classificações da seguinte forma:

- **Entrada Externa (EE)**: é um processo elementar que processa dados ou informação de controle que entram pela fronteira da aplicação. Seu objetivo principal é manter um ou mais ALI ou alterar o comportamento do sistema.
- **Consulta Externa (CE)**: é um processo elementar que envia dados ou informação de controle para fora da fronteira da aplicação. Seu objetivo principal é apresentar informação para o usuário através da recuperação de dados ou informação de controle de ALI ou AIE.
- **Saída Externa (SE)**: é um processo elementar que envia dados ou informação de controle para fora da fronteira da aplicação. Seu objetivo principal é apresentar informação para um usuário ou outra aplicação através de um processamento lógico adicional à recuperação de dados ou informação de controle. O processamento lógico deve conter cálculo, ou criar dados derivados, ou manter ALI ou alterar o comportamento do sistema. (BRASIL, 2015, p.5).

Percebe-se que, para ser considerada uma função de transação, a funcionalidade deve constituir um processo elementar. Trata-se da menor unidade de atividade significativa para o usuário, devendo ser único no escopo da contagem. (CAMPOS, 2011). Para ser considerado um processo elementar, cada funcionalidade deve constituir uma transação completa, autossuficiente e consistente. (IFPUG, 2010).

Quadro 3 – Complexidade das funções de transação – Entradas Externas (EE)

		DERs		
		1 – 4	5 – 15	> 15
ALRs	0 – 1	Baixa	Baixa	Média
	2	Baixa	Média	Alta
	> 2	Média	Alta	Alta

Fonte: IFPUG (2010).

Quadro 4 – Complexidade das funções de transação - Saídas e Consultas (SE e CE)

		DERs		
		1 – 5	6 – 19	> 19
ALRs	0 – 1	Baixa	Baixa	Média
	2 – 3	Baixa	Média	Alta
	> 3	Média	Alta	Alta

Fonte: IFPUG (2010).

Cada função de transação pode referenciar uma ou mais arquivos lógicos, chamados de ALR. Um Arquivo Lógico Referenciado (ALR) “deve ser contado para cada função de dados única que for acessada (lida e/ou gravada) pela função de transação” (IFPUG, 2010, p. 20). Assim como nas funções de dados, os DERs também devem ser contados nas funções de transação, pois da mesma maneira, a complexidade funcional será determinada utilizando-se o número de ALRs e DERs, em conformidade com os quadros 3 e 4. A quantidade de pontos de função dependerá da complexidade de cada uma das funções. O quadro 5 demonstra os valores atribuídos às funções de transação.

Quadro 5 – Tamanho das funções de transação

		Tipo		
		EE	SE	CE
Complexidade Funcional	Baixa	3	4	3
	Média	4	5	4
	Alta	6	7	6

Fonte: IFPUG (2010).

3.2.4 Passo 4 – Calcular tamanho funcional

O tamanho funcional da aplicação corresponde à soma de pontos de todas as funções de dados e de transação identificadas. Podem ser incluídas nas contagens de projetos, as funções necessárias para a implantação do sistema, tais como a conversão de dados. Em projetos de melhoria, além de acrescer as funções de conversão e as funcionalidades adicionadas ao *software*, devem ser contabilizadas as funções removidas, o que pode aumentar ou diminuir o tamanho da aplicação.

3.2.5 Passo 5 – Documentar e reportar

As informações a serem reportadas podem incluir entre outros: a documentação de origem em que se baseou a contagem, os participantes com seus respectivos papéis e qualificações, o número de DERs, RLRs e ALRs de cada função, referências cruzadas e abstrações relacionadas. (IFPUG, 2010). O manual do IFPUG define o que deve ser documentado nas:

> — o propósito e o tipo da contagem;
> — o escopo da contagem e a fronteira da aplicação;
> — a data da contagem;
> — uma lista de todas as funções de dados e de transação, incluindo o respectivo tipo e complexidade, bem como o número de pontos de função atribuído a cada uma;
> — o resultado da contagem;
> — quaisquer suposições feitas e questões resolvidas. (IFPUG, 2010, p. 23).

Ao reportar o resultado é necessário acrescentar uma referência ao padrão seguido, logo após a quantidade de pontos de função que foi obtida. (CAMPOS, 2011). Por exemplo: 150 PF (IFPUG-ISO/IEC 20926:2010). O CPM diz ainda que caso haja customizações locais, devem ser acrescidos um ou mais caracteres indicando que o resultado não mantém conformidade plena com o padrão internacional. (IFPUG, 2010).

3.3 Desafios em torno da métrica de pontos de função

A análise de pontos de função vem ao encontro dos obstáculos impostos para medir *software*, uma vez quetraz consigo seus próprios desafios. Um destes

desafios é medir os projetos de manutenção. O SISP ressalta que métrica Ponto de Função foi concebida como uma medida para projetos de desenvolvimento ou de melhoria de *software*, contudo os projetos de *software* vão além, sendo necessário definir outras métricas para projetos de manutenção que contemplem itens não mensuráveis pelo CPM. (BRASIL, 2015).

Outro desafio é equacionar o preço do ponto de função, considerando as atividades acessórias em torno do desenvolvimento de *software*, ou ainda, a diferenciação entre a construção e a manutenção. Neste sentido, Vasquez, Simões e Albert (2004) sobre o preço praticado nos contratos:

> Ou ainda, o preço do ponto de função para a entrega apenas do *software* certamente é inferior ao preço do ponto de função onde, além do *software*, devem ser entregues vários documentos (subprodutos) como: modelo UML, manual de usuário, ajuda on-line e outros. É também bastante comum no mercado a diferenciação do preço do ponto de função de acordo com a plataforma tecnológica (mainframe, internet, cliente-servidor, etc).
> Deve-se destacar também que pode haver diferenciação de preço do ponto de função em projetos de melhoria para funcionalidades novas, para funcionalidades alteradas e para funcionalidades excluídas. (VAZQUEZ; SIMÕES; ALBERT, 2004, p. 6).

Apesar de avaliar a complexidade das funções, a medição padrão de pontos de função não considera os aspectos não funcionais. Desta forma, surge outro aspecto desafiador: tornar o método compreensível aos desenvolvedores, uma vez que a forma de implementação não é avaliada, mas tão somente aquilo que é significante para o usuário. (DEKKERS, 1998).

Embora o processo de medição utilizando pontos de função pareça relativamente simples, é cercado de nuances que podem assumir caráter subjetivo. Tal fator acaba permitindo que o mesmo *software*, quando avaliado por pessoas diferentes, resulte em duas medidas. Para Sommerville (2005, p. 440) tais variações são consequência do fato que “diferentes pessoas têm diferentes noções de complexidade”.

Paro (2005) cita alguns dos principais pontos negativos em a relação à utilização da métrica PF, entre elas: a necessidade de experiência para que a contagem seja mais precisa; a impossibilidade de automatização das contagens; e a necessidade de acesso a informações do *software* em nível muito detalhado. Tais desvantagens podem tornar o processo de medição um entrave no ciclo de vida

software. É necessário possuir profissionais treinados e dedicados a esta função ou investir em contagens realizadas por empresas especializadas, o que pode tornar o desenvolvimento de sistemas ainda mais caro do que já é.

Ainda assim, Pressman (2011, p. 542) defende sua utilização: “A métrica FP proporciona informações úteis e valor distinto, mesmo que não satisfaça a um atributo perfeitamente”. A favor de seu uso prevalecem as vantagens já destacadas: possibilidade de medição preditiva ou pós-desenvolvimento independente da linguagem de programação; evolução histórica suportada por uma ampla comunidade de usuários e instituições de renome internacional, além do amplo acervo de informações disponível.

4 MÉTRICAS DE PRODUTIVIDADE

4.1 Custos e produtividade no desenvolvimento de *software*

Antes da popularização de métricas como a APF, por um longo tempo muitas organizações, em especial aquelas que terceirizam a produção de *software*, remuneraram este tipo de contrato apenas com base na métrica homem-hora. Os custos para desenvolvimento de um *software* são altos e utilizar apenas este tipo de medida pode acarretar prejuízos. Isto porque, muitas equipes podem acabar ociosas e ainda assim continuar gerando receita para as contratadas. Uma maneira mais justa seria pagar apenas pelo que foi efetivamente entregue. A saída para este problema seria a verificação da produtividade.

Em qualquer sistema de manufatura, a produtividade “pode ser medida pela contagem do número de unidades produzida, dividindo-se o resultado pelo número de pessoa-hora requerido para produzir estas unidades”. (SOMMERVILE, 2005, p. 438). No desenvolvimento de *software*, a produtividade pode ser medida a partir da associação de informações sobre o esforço e tempo gastos nas atividades. O esforço distribuído ao longo do tempo tem um custo que deve ser pago para quem produz o *software*.

Na maioria dos projetos de *software*, o custo preponderante refere-se à mão de obra empregada. Outros custos como transporte, infraestrutura e ferramentas de trabalho (por exemplo, licenciamento de programas utilizados no processo de construção de *softwares*) são ínfimos se comparados aos salários dos desenvolvedores, que somados aos encargos sociais e trabalhistas envolvidos,

tornam o custo total para uma organização que produz sistemas, o equivalente ao dobro do que é pago aos engenheiros de *software.* (SOMMERVILLE, 2005). Espera-se que sejam devidamente restituídos todos os valores envolvidos na produção do *software*, o que inclui ainda os lucros razoáveis devido à realização das atividades.

O custo total, os índices de produtividade e o tamanho funcional, são medidas que podem auxiliar a verificar se o valor pago é coerente com o produto final, tanto por meiode avaliações das aplicações após a sua entrega, quanto na elaboração de estimativas de custos e tempo ainda nas fases iniciais do projeto. Sommerville (2005) destaca que as medidas relacionadas à função do *software* são as mais eficazes para aferir a produtividade de uma equipe de desenvolvimento de *software*, pois permitem expressar a produtividade em termos de funcionalidade útil produzida em determinado tempo.

4.2 Produtividade com pontos de função

A produtividade não faz parte do escopo da contagem de pontos de função, contudo, é possível utilizar suas informações para fazer inferências a partir da aferição do tempo de trabalho e do tamanho do produto entregue. (CAMPOS, 2011). As medidas de tamanho em pontos de função são utilizadas como base para se obter outros elementos necessários para planejamento, tais como tempo e custo. Neste contexto, a produtividade equivale à quantidade de horas necessárias para entregar um ponto de função.

O capítulo anterior demonstrou que uma aplicação pode ser medida após a sua entrega ou ainda, antes de sua construção (estimativa). Um dos métodos mais utilizados para estimar com base na produtividade é chamado de Modelo Simplificado de Estimativas. (VAZQUEZ, 2012 apud BRASIL, 2015). O método utiliza como base o índice de produtividade, calculado pela razão entre o esforço e o total de pontos de função. O esforço para construir uma aplicação pode ser expresso na quantidade de horas que a equipe dedicou ao projeto. O total de pontos é dividido então pela quantidade de esforço. O resultado desta conta é o valor médio de produtividade (H/PF).

O capítulo brasileiro do IFPUG cita em um de seus artigos, algumas fontes de dados sobre produtividade. Ao tratar sobre a produtividade para linguagem JAVA, o

BFPUG [200-?] traz informações do *International Software Benchmarking Standards Group* (ISBSG) e compara com dados do David *Consulting Group (*DCG), uma das empresas de APF mais conceituada nos EUA, conforme reproduzido na Tabela 1. Há ainda outras fontes que citam dados de produtividade em pontos de função para cada tipo de linguagem de programação conforme apresentado na tabela 2 e no quadro 6, apresentados a seguir.

Tabela 1 – Produtividade na linguagem JAVA, segundo a plataforma

Plataforma	**Produtividade**
Client/Server	6,7 H/PF
Web	4,4 H/PF
e-Business Web	7,5 H/PF

Fonte: DCG/BFPUG, [200-?].

Tabela 2 – Produtividade em horas por PF das principais linguagens

Linguagem	**Produtividade**	**Variação**
ASP	6h/PF	-2h e +6h
.Net (C#)	8h/PF	-3h e +6h
COBOL	11,5h/PF	-5,5h e +12,75h
Delphi	7,5h/PF	-1,5h e +2,5h
Java	10h/PF	-3h e +4,5h
Lotus Notes	4h/PF	-0,5h e +3h
Natural	9h/PF	-3h e +5h
PHP	5h/PF	-1h e +7h
SQL	6h/PF	-1,5h e +3h
VBA	8h/PF	-2,5h e +2h
Visual Basic	8h/PF	-2h e +3h

Fonte: AVILA, 2012.

Quadro 6 – Produtividade em Pontos de Função/hora por linguagem

Ambiente/Linguagem	Produtividade (horas/PF)		
	Baixa	Média	Alta
Mainframe			
COBOL	26,4 h	17,6 h	13,2 h
NATURAL	13,2 h	8,8 h	6,6 h
Micro e Cliente/Servidor			
Visual Basic	8,8 h	6,8 h	5,7 h
Delphi	8,8 h	6,8 h	5,7 h
ORACLE	13,2 h	8,8 h	6,6 h
WEB/Documentos			
ASP	12h	10 h	8 h
Java	17 h	15 h	12 h
Lotus Notes	5,5	3,9	3,1

Fonte: Hazan e Staa (2004).

É possível verificar que estes dados demonstram a produtividade em três níveis: baixa, média, alta. Hazan e Staa (2004) defendem que o nível de produtividade deve ser definido baseando-se na complexidade da especificação, experiência da equipe de desenvolvimento e outros fatores que influenciam na produtividade, eles explicam estas variações da seguinte forma:

> Note que o nível de produtividade Baixa está associado, dentre outros fatores, à: aplicações complexas ou equipes inexperientes na plataforma de desenvolvimento. Já o nível de produtividade Alta está associado à aplicações simples, ou a equipes experientes na plataforma. O nível de produtividade da equipe influencia diretamente nas estimativas de prazo, a produtividade Alta implica em prazos de desenvolvimento menores. (HAZAN; STAA, 2004, p. 41).

O SERPRO, uma estatal brasileira que atua na produção de *software*, difunde informações sobre produtividade em H/PF. A versão 7.0 do documento está disponível na internet[5] e traz o índice de produtividade de acordo com cada linguagem e o mesmo tipo de variação. Os dados são apresentados na tabela 3.

Tabela 3 – Produtividade em HH/PF por linguagem

LINGUAGEM/PLATAFORMA	BAIXA	MÉDIA	ALTA
ACCESS*	10	8	6
ASP	16	11	6
ASPNET	11	8	5
ASSEMBLER*	18	12	8

[5] BRASIL. Disponível em: <http://www.pgfn.fazenda.gov.br/acesso-a-informacao/tecnologia-da-informacao/ Roteiro_Contagem_ PF_SERPRO_%207.pdf>. Acesso em: 21 mar. 2017.

BASIC*	18	12	8
C*	24	18	12
C#	17	12	7
C++	18	12	8
CKAN*	18	14	8
CLIPPER*	12	8	6
COBOL	14	10	6
COMPONENTE – CÓDIGO PROPRIETÁRIO	26	19	12
COMPONENTE – CÓDIGO ABERTO	26	19	12
CSP*	16	10	8
DARDO /NETUNO*	18	14	12
DATA DISCOVERY*	16	12	6
DELPHI	12	8	6
DOT NET (.NET)	14	12	10
DW MICROSTRATEGY, POWER CENTRE, OUTRAS (APENAS OLAP)	14	10	5
DW MICROSTRATEGY, POWER CENTRE, OUTRAS (EXTRAÇÃO E OLAP)	16	12	6
DW PENTAHO (APENAS OLAP)	16	10	6
DW PENTAHO (EXTRAÇÃO E OLAP)	18	12	8
EXCEL*	6	5	4
FORMS/REPORTS/ORACLE	16	12	6
HTML	10	8	4
JAVA	14	10	6
JAVA ANDROMDA	15	10	5
JAVA DEMOISELLE V 1.0	16	11	6
JAVA DEMOISELLE V 2.0	19	13	7
JAVA FLEX*	15	12	8
JAVA SCRIPT	16	12	8
JAVA WEB NÃO DISTRIBUÍDA	16	11	6
JCUPIM	41	28	15
JOOMLA*	18	14	8
LASER XEROX*	30	20	16
LIFERAY	16	12	8
LIGHTBASE*	18	12	6
LOTUS NOTES	8	6	4
LTD*	18	13	8
MIDDLEWARE*	26	19	12
MOBILE – ANDROID	18	14	12
MOBILE - ANDROID E IOS	18	14	12
MOBILE - HTML 5 E JQUERY MOBILE*	18	14	12
MOBILE – IOS	18	14	12
MOBILE – PHONEGAP	18	14	12
MOBILE – WINDOWS PHONE *	18	14	12
NATURAL (BATCH E ON-LINE)	12	10	8
ORACLE DESIGNER 2000*	12	6	4
PENTAHO (PROJETOS PENTAHO NÃO BI)	7	6	5
PHP	15	10	5
PL/SQL DE DEMAIS SGBDS	11	9	7
PROJETOS DE GEOPROCESSAMENTO	27	19	11
PROJETOS DE GEORREFERENCIAMENTO*	27	19	11
PROJETOS DE WORKFLOW	24	18	16

PYTHON	18	14	8
RUBY ON RAILS	18	14	8
UNIX SHELL SCRIPTS	18	14	8
VB-SCRIPT	16	14	8
VISUAL BASIC /CRYSTAL REPORTS	12	8	6
VISUAL C++*	16	14	7
VISUAL GEN*	10	8	6
VISUAL INTERDEV*	24	14	8
WEBSERVICE*	26	19	12
ZOPE PLONE	17	11	5

* Nas linguagens marcadas, a produtividade foi definida por analogia e/ou pesquisa de mercado, pois não havia dados suficientes de projetos concluídos no SERPRO para uma análise estatística.

Fonte: SERPRO (2015).

O índice de produtividade é útil, pois permite que a organização estime quanto tempo será gasto para entregar um *software* com base em seus requisitos funcionais e a média histórica de produtividade de projetos similares. Da mesma forma, é possível estimar o custo dos projetos, uma vez que a quantidade de esforço relaciona-se com o valor da mão de obra.

A hora de cada profissional tem seu custo somado aos outros custos diretos ou indiretos da produção do *software*. O custo total pode ser dividido pelo total de pontos de função, obtendo-se um valor médio para cada ponto. Este valor médio pode ser utilizado para calcular o custo de uma aplicação de acordo com seu tamanho funcional, seja com base no que foi entregue ou apenas nos requisitos definidos.

Ressalte-se que ao medir o tempo ou custo gasto para construir um aplicativo a partir da quantidade de pontos de função, a organização deve ter em mente que a métrica de produtividade pode variar de acordo com cada realidade. O Roteiro de Métricas do SISP enfatiza que:

> O índice de produtividade depende de diversos atributos dos projetos, dentre outros: plataforma tecnológica, complexidade do domínio, segurança, desempenho, usabilidade, tamanho do projeto, tipo de manutenção, desenvolvimento de componentes.
> Cada órgão ou entidade deverá possuir sua própria tabela de produtividade para cada linguagem, considerando-se sempre dados históricos dos projetos já realizados. (BRASIL, 2015, p.34)

Isto significa que não há indicadores de produtividade genéricos. A FATTO,[6] uma empresa especializada em consultorias e treinamentos para utilização da

[6] BRASIL. Disponível em: <http://www.fattocs.com/pt/faq-36.html>. Acesso em: 20 mar. 2017.

métrica de pontos de função, alerta:

> Há uma tentação muito grande entre os profissionais envolvidos em estimativas de usar indicadores ditos "de mercado" para suas estimativas baseadas em pontos de função. Certamente existem várias fontes onde é possível obter números relacionados à produtividade com pontos de função para diversos contextos tecnológicos. O ISBSG (International Software Benchmarking Standards Group) é uma delas. No entanto, usar essas fontes desconsiderando o contexto de como aqueles números foram obtidos e o contexto de sua própria organização é um erro grave. Estimativas obtidas dessa maneira dificilmente terão a confiabilidade necessária ou alguma utilidade prática.
>
> A melhor forma de obter indicadores de produtividade que realmente sejam úteis nas estimativas com pontos de função é apurar esse indicador através dos projetos desenvolvidos pela organização. (FATTO, [201-]).

Ao ser questionado sobre a produtividade em pontos de função para determinada linguagem, o BFPUG[7] afirma que tal índice é eficaz apenas para comparações, ou seja, para avaliar se a produtividade foi maior ou menor do que a de outros projetos realizados em ambientes similares. Neste respeito, as variações nos índices podem ser consequência de diversos fatores que podem afetar a produtividade. Alguns destes fatores são demonstrados a seguir.

4.3 Fatores que afetam a produtividade

Não existe valor médio para a produtividade que não tenha a interferência de diversos fatores. O tipo de linguagem, por exemplo, é uma variável que afeta diretamente os índices. Quanto maior a complexidade do sistema, menor será a produtividade. O suporte técnico, as ferramentas e a capacitação do programador podem afetar significativamente seu trabalho. Outro aspecto é que as medidas expressas em volume/tempo não levam em conta as características não funcionais. Maior quantidade de funções pode não significar melhor qualidade de produto. (SOMMERVILLE, 2005).

Pressman (2011) afirma que a quantidade de linhas de código e o total de pontos de função podem ser utilizados para derivar métricas de produtividade, contudo, é necessário considerar os fatores que podem influenciar a capacidade de produção, para que as comparações não sejam interpretadas incorretamente. Entre

[7] BRASIL. Disponível em: <http://www.bfpug.com.br/Produtividade_Java.htm>. Acesso em: 20 mar. 2017.

os vários fatores que podem afetar a produtividade e consequentemente, aumentar os custos da produção de *software*, Sommerville (2005) destaca alguns que frequentemente interferem nos resultados, conforme apresentado no quadro a seguir.

O autor destaca ainda, que a quantidade de pessoas trabalhando em uma equipe também interfere diretamente na produtividade:

> A relação entre o número de pessoas que trabalham em um projeto, o esforço total requerido e o tempo de desenvolvimento não é linear. À medida que o número de pessoas aumenta, mais esforço pode ser necessário; as pessoas podem gastar mais tempo se comunicando; mais tempo é exigido para definir as interfaces entre as partes do sistema. Duplicar o número de pessoas (por exemplo) não significa que a duração do projeto será reduzida pela metade. (SOMMERVILLE, 2005, p. 454).

Quadro 7 – Fatores que influenciam a produtividade

Fator	Descrição
Oportunidade de mercado	Uma organização de desenvolvimento pode cotar um preço baixo porque deseja iniciar em um novo segmento do mercado de software. Aceitando um lucro baixo em um projeto, é possível ter a oportunidade de obter melhores lucros, posteriormente. A experiência obtida pode permitir o desenvolvimento de novos projetos.
Incerteza da estimativa de custo	Se uma organização estiver insegura sobre sua estimativa de custos, ela pode aumentar seu preço bem acima de seu lucro normal, justificando alguma despesa eventual.
Condições contratuais	Um cliente pode querer permitir que o desenvolvedor mantenha a propriedade sobre o código-fonte e o reutilize em outros projetos. O preço cobrado pode então ser menor do que se o código-fonte fosse entregue ao cliente.
Volatilidade dos requisitos	Se existe a probabilidade de que os requisitos sejam modificados, uma organização pode baixar seu preço para ganhar um contrato. Depois que o contrato for efetivado, altos preços podem ser cobrados por mudanças nos requisitos.
Saúde financeira	Desenvolvedores em dificuldades financeiras podem reduzir seu preço para conseguir um contrato. É melhor obter um lucro menor do que o normal ou mesmo apenas equilibrar o lucro e a despesa do que perder o contrato.

Fonte: Sommerville (2005).

Ainda assim, o mais significativo fator que interfere na capacidade produtiva dos desenvolvedores é a habilidade individual de cada um. (SOMMERVILLE, 2005). O desenvolvimento de *software* tem como matéria-prima de produção básica, a intelectualidade dos programadores. Desta forma, quanto maior for a habilidade da equipe, maior será sua produtividade.

Entretanto, é importante ressaltar que a utilização de índices de desempenho para aferir a habilidade pode ser controversa. Sommerville (2005, p. 442) aponta que isto pode se tornar “um problema em particular, se os gerentes utilizam as medidas de produtividade para julgar as capacitações do pessoal”, visto que um programador tido como menos produtivo pode produzir códigos mais confiáveis e isto pode ser mais vantajoso, pois demandará um menor custo de manutenção posterior do *software*. Assim sendo, é importante utilizar as métricas de produtividade com atenção às peculiaridades de cada equipe ou projeto.

Pressman (2011, p.615) nota que “você pode estimar o esforço (por exemplo, pessoas-mês) necessário para executar cada atividade do processo de *software*, para cada função de *software*” e então utilizar os valores médios do preço de mão de obra. Estes valores variam de acordo com cada tipo de tarefa. Desenvolvedores mais experientes costumam ter melhor desempenho em atividades estruturais iniciais, estes profissionais são mais caros do que profissionais menos experientes envolvidos em atividades mais operacionais. (PRESSMAN, 2011).

Há diversos métodos específicos para estimar a quantidade de esforço necessário para construir um *software*.[8] É recomendável utilizar outras medidas de produto ou processo para demonstrar a concordância entre as métricas. Assim como as medidas de produtividade, as técnicas de estimativa também possuem cada uma seus pontos positivos e negativos. (SOMMERVILLE, 2005).

Ao analisar tais métodos, o que se pode concluir é que as métricas de tamanho funcional ou outras medidas de tamanho do *software* podem ser utilizadas tanto para estimar quanto para aferir a produtividade. No entanto, é preciso estabelecer uma referência de comparação com base em dados históricos, preferencialmente optando por informações de projetos similares. (PRESSMAN, 2011). É necessário levar em conta ainda “que há muitas diferenças importantes entre os projetos anteriores e futuros” e as experiências passadas podem não condizer com a nova realidade. (SOMMERVILLE, 2005, p. 444). Novas tecnologias surgem a cada ano e isto torna cada vez mais difícil o processo de comparação entre as medidas. Dessa maneira, a realidade de cada organização deve ser considerada como um ponto determinante para que as medidas sejam mais precisas.

[8] As obras de Pressman (2011) e Sommerville (2005) possuem capítulos específicos que apresentam algumas das principais técnicas para estimativa de *software* utilizando métricas de *software*.

5 MÉTRICAS DE *SOFTWARE* NO SETOR PÚBLICO

5.1 A produção de *software* por meio de terceirização

Pressman (2011) afirma que a terceirização da produção de *software* é uma alternativa que diminui os custos financeiros. Especialmente devido à possibilidade de reduzir as despesas com a manutenção de quadro próprio de pessoal. Visto que manter uma unidade de tecnologia da informação, exclusivamente para este fim, custa mais caro do que contratar serviços especializados de maneira pontual. Além disso, as organizações que terceirizam o desenvolvimento de sistemas podem manter seu foco de atuação, investimentos e pessoal mais dedicados às atividades que façam parte do seu negócio principal.

Ao contratante resta a responsabilidade de gerenciamento destes serviços. Isto inclui o planejamento prévio e a verificação do que foi entregue para pagamento. Para tanto se faz necessário o uso de métricas de *software*, permitindo maior controle sobre o que é produzido. No Brasil, a utilização de métricas de *software* foi impulsionada pela terceirização de serviços no setor público.

Na obra *Gestão de contratos de terceirização na Administração Pública: teoria e prática* (VIEIRA et al. 2015, p. 36), a terceirização é definida como "um modelo de gestão em que a Administração contrata os serviços de terceiros (particulares), não podendo ser confundida com contratação de mão de obra". Segundo Vieira et al. (2015, p. 36):

> A terceirização é realizada para o cumprimento das atividades consideradas acessórias, ou seja, não é atividade principal de quem está repassando. Pode envolver a produção de bens, não sendo só de serviços, muito embora seja mais usual.

Tal posicionamento guarda sintonia com o Decreto nº 2.271/1997. O referido dispositivo legal lista, no rol de serviços que podem ser terceirizados, os de informática. Desta forma, a construção de sistemas de informação pode ser objeto de execução indireta, conforme reproduzido a seguir:

> Art. 1º No âmbito da Administração Pública Federal direta, autárquica e fundacional poderão ser objeto de execução indireta as atividades materiais acessórias, instrumentais ou complementares aos assuntos que constituem área de competência legal do órgão ou entidade.
> §1º As atividades de conservação, limpeza, segurança, vigilância, transportes, **informática**, copeiragem, recepção, reprografia,

> telecomunicações e manutenção de prédios, equipamentos e instalações serão, de preferência, objeto de execução indireta. (BRASIL, 1997, grifo nosso).

Entretanto, a Constituição Federal de 1988, no parágrafo único do artigo 70, deixa clara a necessidade de prestar contas do uso do dinheiro público. (BRASIL, 1988). Portanto, os órgãos devem atentar para que as contratações de serviços desta natureza não se tornem despesas antieconômicas. Um dos aspectos que podem causar prejuízos ao erário é a incorreta aplicação de métricas, tais como a remuneração de contratos com base apenas na medição homem-hora.

5.2 O problema da métrica homem-hora

Cavalcanti (2015) define um modelo de contratação de TI costumeiramente adotado pela Administração Pública:

> O antigo modelo consistia, em síntese, na reunião de todos os serviços de informática do órgão em um único e grande contrato, adjudicado a uma única empresa, com pagamentos realizados exclusivamente por hora trabalhada.
> Ou seja, caso determinado órgão necessitasse de um conjunto de Soluções de TI, esse órgão então reunia todo esse conjunto em uma única licitação, para contratação, via de regra, de uma única empresa que seria remunerada em função das horas trabalhadas de seu pessoal técnico envolvido.
> Esse procedimento foi invariavelmente adotado até o final dos anos 2000. Obteve relativo sucesso durante muito tempo, digamos, especialmente nas décadas de 70 a 90, até que a Administração e os órgãos de controle identificassem uma série de graves problemas nos contratos resultantes desse modo de contratação. (CAVALCANTI, 2015, p. 26).

Este modelo, conhecido por sua métrica (Homem/Hora – HH), foi comumente adotado em diversas atividades terceirizadas pelo Poder Público. São também chamados de contratação *body shop* ou *time and material*. O modelo homem-hora é de fácil administração. Sua flexibilidade permite atender rapidamente às demandas, propiciando uma resposta efetiva às constantes mudanças nos requisitos de *software.* (SIMÕES, 2004).

Em sentido oposto, quando não há o devido acompanhamento da produtividade, torna-se evidente a possibilidade de subutilização dos recursos. Simões (2004, p.27) destaca alguns pontos negativos da medição homem-hora:

“competência não necessariamente disponível, remuneração não vinculada a resultados, falta de estímulo ao aumento de produtividade”. Todos os fatores que influenciam a produtividade podem tornar morosa a entrega de um *software*. Desta maneira, se a medida utilizada para remunerar a construção do sistema leva em conta apenas o tempo que foi gasto para produzi-lo, corre-se o risco de pagar por horas improdutivas.

Em 2001, o Tribunal de Contas da União já alertava para a ausência de controle efetivo sobre a prestação de serviços terceirizados. O alerta tinha como alvo principal os contratos, cuja execução era medida por horas trabalhadas. (VIEIRA et al, 2015). Em relação às unidades de informática, o TCU constatou no Acórdão nº 1.603/2008 que:

> 106. Além disso, um total de 74% dos pesquisados informaram que não executam a gestão de níveis de serviço dos serviços contratados, ou seja, mesmo quando o órgão/entidade é cliente e não fornecedor, não há preocupação com a avaliação e o controle dos resultados. Assim, como em última instância um serviço contratado pela área de TI visa atender à necessidade dos seus clientes, a ausência da gestão externa tem as mesmas conseqüências da sua ausência da gestão interna dos níveis de serviço. (BRASIL, TCU – Acórdão nº 1.603/2008, p. 25).

O TCU orienta que serviços de TI sejam remunerados com base em resultados e não somente pela disponibilidade de mão de obra. (BRASIL, 2015). A constatação da corte federal é de que a métrica homem-hora estimula os fornecedores a consumirem mais horas remuneradas de trabalho, do que o necessário para alcançar o resultado contratado. (BRASIL, 2003). Cavalcanti (2015) explica porque tal modelo é antieconômico:

> A *antieconomicidade* do procedimento é revelada na medida em que o pagamento se dava com base exclusivamente em horas trabalhadas, sem considerar o produto ou resultado, o que possibilita a ocorrência do chamado *paradoxo do lucro-incompetência*: quanto menor a qualificação dos profissionais alocados na prestação de serviço, maior o número de horas necessário para executá-lo, e, assim, maior a margem de lucro da empresa contratada e maior o custo e o valor pago pela Administração. Está-se aqui a falar de ineficiência e não de má-fé, ressalte-se.
> Tinha-se, nessa situação, a remuneração de todas as horas de disponibilidade dos empregados da empresa, ainda que não produtivas, de modo que poderia haver remuneração, muitas vezes, sem que houvesse a contraprestação em serviços efetivamente realizados (hipótese de contratação por posto de serviço).

> Nesse modelo, a empresa apresentava uma fatura ao final de cada mês contendo o número de pessoas que teriam prestado o serviço, com base no contrato, e quantas horas cada uma delas teria trabalhado. Com base nisso, a administração efetuava os pagamentos.
> Em nossa opinião, trata-se da pior forma de pagamento que a Administração Pública pode realizar, pois incentiva a ineficiência no contratado, uma vez que quanto maior o tempo gasto na execução das tarefas contratadas maior será a remuneração dele.
> Até mesmo o pagamento por postos de serviço, utilizado em algumas situações, poderia ser considerado melhor que o pagamento por hora trabalhada; todavia, também comporta um grau de ineficiência, já que quando se paga dessa forma, a Administração o faz pela disponibilidade do serviço e não pelo serviço em si mesmo. (CAVALCANTI, 2015, p. 30-31).

5.3 As orientações do TCU

Considerando que o “antigo modelo, se empregado às atuais contratações dos bens e serviços de Tecnologia da Informação, representa clara ofensa ao ordenamento jurídico”, o que se propôs foi uma nova forma de contratação baseada em resultados, permitindo a observância da legislação, bem como das orientações jurisprudenciais constantes nos acórdãos do TCU a respeito do tema. (CAVALCANTI, 2015, p. 53).

Ao longo dos últimos anos, o TCU proferiu diversas decisões que contribuíram significativamente para a mudança de paradigmas na contratação de serviços de TI de um modo geral. A Nota Técnica nº 6/2010 pode ser considerada um compêndio destas decisões e reúne diretrizes que devem ser observadas pelo gestor público. Tais diretrizes nortearam também a elaboração de instruções normativas por parte dos jurisdicionados daquela corte. Uma delas é a IN-04 (Instrução Normativa MPOG/SLTI nº 04/2014).

Esta norma disciplina os processos de contratação de Soluções de Tecnologia da Informação pelos órgãos integrantes do Sistema de Administração de Recursos de Tecnologia da Informação (SISP), no âmbito do Poder Executivo Federal. (CAVALCANTI, 2015). Alguns dos pontos presentes na IN-04 também são destacados na nota técnica publicada do TCU, entre eles os que dizem respeito à utilização de métricas quantitativas para remunerar a produção de *software*.

5.3.1 Obrigatoriedade de remunerar por resultados

O Decreto nº 2.271/1997 define no parágrafo § 1º do artigo 3º que “sempre que a prestação do serviço, objeto da contratação, puder ser avaliada por determinada unidade quantitativa de serviço prestado”, além da previsão desta unidade no edital e no contrato, é necessário utilizar tal medida como parâmetro de aferição dos resultados. (BRASIL, 1997).

Neste mesmo sentido, o TCU posicionou-se a favor do pagamento por resultados no Acórdão nº 2.471/2008, do Plenário daquela Corte. No item 9.1.4, o órgão de controle recomendou ao Ministério do Planejamento, que expedisse normativo para regulamentar a contratação de serviços de TI, estabelecendo preferencialmente a execução indireta com medição por resultados. Naquele mesmo ano, atendendo a esta recomendação, editou-se a primeira versão da IN-04. (BRASIL, 2015). A referida norma impôs a medição por resultados e vedou à contratação de serviços de TI por posto de trabalho, ressalvados os casos com justificativa devidamente fundamentada. (BRASIL, 2014). Resta evidente que a Administração Pública deve proceder à efetivação de pagamentos somente de forma vinculada à entrega de resultados, conforme enfatiza a Nota Técnica do TCU:

> Desse modo, não há outro caminho juridicamente possível a ser adotado pelos gestores públicos de TI que não seja o de efetuar suas contratações com pagamentos vinculados à apresentação de resultados verificáveis, mensurados, sempre que possível, por unidades quantitativas. (BRASIL, 2010, p. 4).

5.3.2 Mensurabilidade dos resultados

Outro aspecto que pode ser destacado nas recomendações do TCU é a mensurabilidade dos resultados. Cabe aos órgãos públicos estabelecer mecanismos de mensuração dos resultados efetivamente produzidos pela empresa contratada. No que diz respeito à produção de *software*, só é possível fazer tal medição se existirem métricas bem definidas para este fim.

O TCU sugere que os serviços de TI contratados pelo poder público devem “ser adequadamente mensurados, seja por unidades de medida de tamanho (por exemplo,pontos de função em desenvolvimento de *software*) ou por indicadores de

nível de serviço". (BRASIL, 2010, p. 4). Enquanto os níveis de serviço podem ser utilizados para aferir atividades, tais como suporte a banco de dados e rede de computadores, os pontos de função são listados pelo TCU entre os exemplos de métricas para a produção de *software*, conforme reproduzido no quadro 8.

Quadro 8 – Exemplos de indicadores de desempenho

Serviço	Indicadores	Métrica
1. Desenvolvimento de Software.	a) tamanho funcional; b) índice de atrasos na entrega dos artefatos; c) índice de não conformidade com os requisitos; d) índice de erros na operação do software.	a1) ponto de função; a2) ponto de caso de uso; b) n. de entregas atrasadas/n. total de encomendas; c) n. de não conformidades por requisito; d) n. de erros por unidade de tempo.
2. Suporte ao Banco de Dados.	a) tempo médio para reestabelecimento do banco de dados; b) tempo médio entre falhas; c) índice de disponibilidade no mês do Banco de Dados.	a) minutos, horas; b) horas, meses, anos; c) percentual do tempo máximo de disponibilidade previsto.
3. Link de dados.	a) qualidade do *link* comunicação de dados; b) velocidade garantida do *link*; c) tempo médio para reestabelecimento do *link*; d) tempo médio entre falhas; e) índice de disponibilidade no mês do *link*.	a) percentual de pacotes perdidos; b) percentual da velocidade nominal; c) minutos; d) minutos; e) percentual do tempo máximo de disponibilidade previsto.
4. Suporte técnico a usuários.	a) índice de chamados resolvidos em até sessenta minutos do seu recebimento; b) Índice de chamadas atendidas em até trinta segundos.	a) percentual dos chamados totais; b) percentual dos chamados totais.

Fonte: Nota Técnica nº 6/2010 – SEFTI/TCU (BRASIL, 2010).

O estabelecimento de métricas e indicadores permite verificar se o fornecedor está cumprindo níveis mínimos de serviço, inclusive no que diz respeito ao cumprimento de prazos. Para o TCU é imprescindível que tais controles sejam estabelecidos. Aquela corte de contas fundamenta estes entendimentos na interpretação da Lei de Licitações e Contratos. Neste sentido, a Lei nº 8.666/1993 diz o seguinte:

> Art. 6º Para os fins desta Lei, considera-se:
> [...]
> IX - Projeto Básico - conjunto de elementos necessários e suficientes, com nível de precisão adequado, para caracterizar a obra

> ou serviço, ou complexo de obras ou serviços objeto da licitação, elaborado com base nas indicações dos estudos técnicos preliminares, que assegurem a viabilidade técnica e o adequado tratamento do impacto ambiental do empreendimento, e que possibilite a avaliação do custo da obra e **a definição dos métodos e do prazo de execução**, devendo conter os seguintes elementos:
> [...]
> e) subsídios para montagem do plano de licitação e **gestão da obra, compreendendo a sua programação**, a estratégia de suprimentos, as normas de fiscalização e outros dados necessários em cada caso; (BRASIL, 1993, grifo nosso).

Além da observância dos prazos e das metas, os indicadores e as métricas servem para indicar se a quantidade de serviço estabelecida no contrato está sendo entregue pelo fornecedor. Desta forma, é importante que os indicadores sejam formulados em conjunto com a utilização de métricas de tamanho/quantidade, a fim de se avaliar a produtividade do fornecedor. (BRASIL, 2010).

Ainda segundo a Nota Técnica do TCU, a correta aplicação da norma implica que as contratações de serviços de TI devem ser precedidas do estabelecimento de requisitos mínimos a serem cumpridos pelo fornecedor. Não se trata de acordos bilaterais firmados entre o contratante e o fornecedor, mas de “metas de serviço” em um nível mínimo estabelecido ainda na fase interna do processo licitatório. Tais metas devem ser descritas com clareza e precisão nos editais. (BRASIL, 2010).

Desta forma, estabelecer mecanismos de controle, por meiode métricas e indicadores de desempenho é essencial para acompanhar se os serviços prestados estão de acordo com o que foi contratado e assim remunerar os resultados.

5.3.3 Observância aos princípios da eficiência e da eficácia

O princípio da eficiência está insculpido no artigo 37 da Constituição Federal de 1988. A carta magna ainda atribui a cada poder, a função de “comprovar a legalidade e avaliar os resultados, quanto à eficácia e eficiência, da gestão orçamentária, financeira e patrimonial”. (BRASIL, 1988, art. 74).

Tais princípios são considerados pelo TCU de maneira recorrente, notadamente ao tecer orientações a respeito das contratações de TI. A efetivação de pagamentos vinculada à entrega de resultados é mais do que uma boa prática a ser observada, mas torna-se obrigatória ao passo que esta forma de contratação se mostra mais eficiente que remunerar o fornecedor somente por sua disponibilidade.

(BRASIL, 2010).

Da mesma maneira, “contratações com nível mínimo de serviço possibilitam uma forma eficaz de se remunerar o fornecedor por resultados” e deste modoestão aderentes aos princípios constitucionais da eficiência e da eficácia, conforme demonstra a instrução da dada pelo TCU:

> 54. Nesse contexto, a GNS pode auxiliar a contratação a ser eficaz por meio da clara especificação dos processos a serem realizados e dos produtos a serem entregues pelo fornecedor, indicando as metas de serviço que deverão ser alcançadas, além de disponibilizar indicadores de desempenho e métricas capazes de avaliar se o serviço prestado está em conformidade com aquilo que foi especificado.
> 55. Dessa forma, a empresa prestadora do serviço terá melhores condições de entregar o objeto da contratação em conformidade com as especificações feitas pela organização contratante, que, por sua vez, terá à sua disposição instrumentos capazes de aferir a prestação do serviço, aumentando, portanto, a chance de se alcançar eficácia à contratação. (BRASIL, 2010, p. 10).

5.3.4 Uso de Pontos de Função para medir *software*

Por meio do Acórdão nº 2.362/2015, o Plenário do TCU apreciou um Relatório de Auditoria Operacional que buscava verificar a eficácia e a eficiência do modelo de contratação de desenvolvimento e manutenção de sistemas adotados na esfera federal. No escopo da fiscalização estavam os riscos envolvidos nas contratações e as métricas adotadas. O relatório/voto integrante do referido acórdão, esclarece que o uso de pontos de função não é obrigatório (item 4.1.3, parágrafos 80 a 90 do relatório), porém reforça a necessidade remunerar com base em resultados:

> 215. Em termos de métricas, restou demonstrado que os serviços, aparentemente, estão sendo pagos com base em resultados e a métrica mais utilizada é a Análise de Pontos de Função. Em que pese a majoritária utilização da referida métrica, foi relatado também caso de utilização de outra métrica. Ambas foram consideradas adequadas, e a equipe propôs revisão de nota técnica do TCU a fim de esclarecer que a jurisprudência do TCU acerca da contratação de desenvolvimento de *software* é no sentido de que a obrigatoriedade é para que se contrate com base em resultados ou níveis mimos de serviço, não sendo obrigatória uma métrica específica. (BRASIL, TCU-Acórdão 2.362/2015, p. 33).

Mesmo sem tornar a métrica obrigatória, o TCU destaca que qualificação mínima dos profissionais e o local de prestação dos serviços são características que

influenciam o preço praticado pelos fornecedores, assim como outros fatores que impactam o desenvolvimento de *software*, conforme reproduzido do Acórdão nº 161/2012–TCU–Plenário:

> 9. [...] O valor R$/PF irá variar de acordo com o trabalho exigido para a entrega das funcionalidades do *software*, de acordo com o padrão técnico e de qualidade solicitado, como também conforme a quantidade de entregáveis (artefatos, documentos, modelos, etc) exigidos pelo cliente. Tudo aquilo que afeta custo de forma significativa, mas que não tem relação direta com o tamanho medido pela APF acaba sendo computado no preço do ponto de função.
> 10. Em resumo, não existe um preço único para ponto de função. Deve-se avaliar o conjunto de atividades relativas à disponibilização das funcionalidades medidas em pontos de função, o modelo de contrato que ditará a remuneração de um ponto de função, e também os aspectos não funcionais que são desconsiderados na medição dos pontos de função para obter uma referência confiável. É provável que uma organização empreenda projetos de diferentes tipos e neste caso deve-se proceder a análise do R$/PF para cada categoria de projetos, pois dificilmente um preço único será representativo para projetos de tipos distintos. (BRASIL, TCU-Acórdão 161/2012, p. 3)

Pelo exposto, percebe-se que persiste o desafio de estabelecer uma métrica adequada para cada caso. Independentemente da utilização de pontos de função ou de outro tipo de medida, resta evidente que a correta utilização de métricas baseadas em resultados permite que sejam evitados prejuízos na aplicação dos recursos públicos.

6 ESTUDO DE CASO

6.1 Objetivos

O principal objetivo deste trabalho é conhecer a realidade da aplicação de métricas de *software* no âmbito do Tribunal de Contas do Estado de Goiás. Assim como outros órgãos da Administração Pública, por um longo tempo o TCE-GO remunerou os serviços de desenvolvimento de *software* apenas com base na entrega de horas trabalhadas.

Mesmo após o conhecimento das recomendações do TCU, a utilização de métricas de *software* para gerir contratos com base em resultados só tornou-se

realidade no TCE-GO recentemente.[9] Desta forma, buscou-se verificar neste estudo de caso se a remuneração homem-hora gerou algum tipo de prejuízo àquela corte.

Conforme apresentado nos capítulos anteriores, para obter esta informação é necessário verificar qual foi a produtividade alcançada pela contratada, bem como os valores pagos pelo que foi efetivamente entregue. Para tanto, é necessário aplicar os conceitos de engenharia de *software*. Neste sentido, buscou-se avaliar alguns dos produtos de *software* entregues ao TCE-GO, medindo o seu tamanho funcional por meioda técnica de pontos de função.

A métrica foi utilizada juntamente com os dados disponíveis sobre o esforço empreendido em horas e o custo direto pago pelo Tribunal, permitindo verificar o desempenho e o valor médio de cada PF. Tais informações foram comparadas ao que foi contratado por outros órgãos com realidade semelhante, a fim de detectar e elucidar discrepâncias ocasionalmente detectadas.

Com esta análise, objetiva-se ainda, verificar se outros aspectos decorrentes da medição poderiam ter tornado a contratação aparentemente menos econômica, tais como eventuais custos adicionais ou particularidades dos projetos do órgão. Por fim, busca-se verificar se existe uma métrica eficaz para remunerar de maneira justa e objetiva a produção de *software*.

6.2 Metodologia

6.2.1 Seleção da amostra

A primeira fase deste estudo de caso consistiu em delimitar quais sistemas de informação constituiriam o estudo de caso. Foram selecionados projetos desenvolvidos inteiramente por empresas terceirizadas contratadas pelo TCE-GO, no qual foi utilizada a métrica homem-hora para fins de contabilização e faturamento. Optou-se por avaliar projetos mais recentes e que foram suficientemente documentados.

Entre os mais de 40 sistemas especialistas que foram construídos ao longo dos últimos anos, estão desde simples aplicativos até ferramentas bastante complexas. A maior parte destes sistemas é construída sob uma mesma plataforma de linguagem e banco de dados. Desta maneir,a buscou-se apresentar situações

[9] O estabelecimento de níveis mínimos de serviço para contratação de serviços de TI aparece pela primeira vez no edital da Concorrência nº 001/2016, disponível no *site* TCE-GO (www.tce.go.gov.br).

típicas e aplicações que pudessem ser rapidamente medidas a fim de evitar subjetividades decorrentes da aplicação da técnica de PF. De igual forma, espera-se validar o quanto as variáveis poderiam oscilar, optando-se por avaliar casos atípicos também. Assim, foi selecionada uma amostra de 03 sistemas com complexidade variada e requisitos distintos, porém construídos sob plataforma semelhante com o propósito de estabelecer dados estatísticos confiáveis.

6.2.2 Coleta de dados

A segunda fase consistiu na coleta de dados. O TCE-GO mantém em ferramentas e repositórios para documentar cada fase do processo de desenvolvimento de *software*. Com acesso a estas informações buscou-se levantar todos os dados necessários relacionados aos sistemas selecionados na fase anterior, tais como Modelos de Entidade de Relacionamento, Casos de Uso, Relatórios de Atividades, Informações de Gerenciamento dos Projetos, Notas Ficais, Apontamentos, Ordens de Serviço e os próprios sistemas em operação, uma vez que estes foram medidos após sua conclusão.

Coletou-se ainda, informações sobre o valor médio do ponto de função, praticado por outros Tribunais de Contas, para que este pudesse servir como parâmetro para estimar o valor de cada sistema. Neste caso, o método empreendido foi a pesquisa em fontes oficiais, tais como contratos e editais nos portais de transparência destes órgãos.

6.2.3 Análise e interpretação

Reunidas todas as informações, partiu-se então à terceira fase: de seleção, análise e interpretação dos dados. Considerando a problematização-hipótese foram separados os dados úteis. Nesta fase, o objetivo foi avaliar a qualidade da amostra, sob a perspectiva da possibilidade de fazer generalizações a partir dos dados. Foi obtido o valor pago pelo TCE-GO para cada sistema selecionado em razão das horas gastas e faturadas. Em seguida, estes sistemas foram medidos utilizando a técnica de pontos de função.[10] Novamente, tiveram seu valor calculado com base no

[10] A contagem de pontos de função foi realizada utilizando o modelo aceito pela norma ISO/IEC, sem a aplicação de fatores de ajuste.

seu tamanho funcional, multiplicando-se o valor médio praticado por outros Tribunais de Contas.

A comparação dos valores finais permitiu aferir se o *software* pago somente em função das horas gastas foi ou não menor do que se fosse pago utilizando o valor médio do ponto de função. As análises quantitativas foram também complementadas com análises qualitativas, com o propósito de avaliar qual poderia ser o impacto da adoção de outras técnicas para mensurar estes serviços ou que outros fatores poderiam ter afetado os custos de cada ferramenta desenvolvida.

6.3 Dados

Foram identificados durante a pesquisa, outros tribunais que utilizam pontos de função para remunerar contratos de desenvolvimento de *software*. Entre eles estão: Tribunal de Contas do Estado do Pernambuco (TCE-PE), Tribunal de Contas do Estado do Mato Grosso (TCE-MT) e o Tribunal de Contas do Estado de São Paulo (TCE-SP). O valor contratado por estes tribunais foi utilizado para calcular o valor médio pago por cada ponto de função entregue, conforme demonstrado no quadro 9.

Quadro 9 – Valor do Ponto de Função contratado por Tribunais de Contas

TCE-PE	TCE-MT	TCE-SP	**Valor médio**
R$ 539,90	R$ 485,00	R$ 592,00	**R$ 538,97**

Fonte: Elaborado pelo autor[11]

Este parâmetro foi utilizado para estipular um valor provável para cada sistema escolhido entre os produzidos no TCE-GO. Foi realizada a contagem de pontos de função destes sistemas e em seguida multiplicou-se o tamanho funcional pelo preço médio pago por outros TCEs, conforme tabela 4. Também foram reunidas informações a respeito dos valores efetivamente pagos pelo TCE-GO às empresas contratadas. A tabela 4 demonstra também o tempo e valor gastos, bem como os índices de produtividade e desempenho obtidos.

[11] Com base em Pernambuco (2015), Mato Grosso (2013) e São Paulo (2015).

Tabela 4 – Tamanho funcional e valor pago por sistemas do TCE-GO

	ILB-CURSOS	DECISÕES	JURISPRUDÊNCIA
TAMANHO FUNCIONAL (PF)	45	56	57
VALOR PAGO	R$ 28.935,85	R$ 54.021,50	R$ 16.692,10
VALOR CALCULADO*	R$ 24.253,65	R$ 30.182,32	R$ 30.721,29
DIFERENÇA	19,31%	78,98%	- 45,67%
CUSTO POR PONTO DE FUNÇÃO	R$ 643,02	R$ 964,67	R$ 292,84
TEMPO GASTO (HORAS)	461,5	757,7	244,0
PRODUTIVIDADE (HH/PF)	10,3	13,5	4,3

Fonte: Elaborada pelo autor

6.3.1– SISTEMA 1: ILB-CURSOS

A primeira aplicação medida foi o “Levantamento de necessidades de capacitação” do projeto denominado “Pesquisas-ILB/Catálogo de cursos”. O sistema entregue ao TCE-GO possui 45 PF e custou ao todo R$ 28.935,85. Este valor inclui todas as etapas: elaboração do projeto, levantamento e análise de requisitos e fase de implementação. Esta ferramenta foi produzida após a adoção pelo TCE de uma nova identidade visual para os seus portais. Desta forma, foi incluído neste projeto o desenvolvimento de uma nova interface, que seria posteriormente adotada em todos os demais sistemas desenvolvidos.

6.3.2– SISTEMA 2: DECISÕES

A segunda ferramenta medida neste trabalho foi a “Consulta de Decisões do TCE-GO”. Apesar de possuir apenas 56 PF, o sistema custou um total de R$ 54.021,50. O Tribunal já possuía uma ferramenta para consulta de julgados denominada TCE-JURIS, contudo para atender aos requisitos propostos pela Associação dos Membros dos Tribunais de Contas (ATRICON), o TCE-GO resolveu desenvolver uma nova forma para disponibilizar suas decisões.

Segundo o TCE-GO, inicialmente foi disponibilizada uma Consulta de Decisões utilizando a mesma tecnologia já adotada no sistema anterior, que consistia em realizar a busca diretamente no banco de dados. Porém, nesta versão as pesquisas que traziam mais resultados demoravam muito tempo para serem processadas, dificultando a utilização do sistema.

O projeto já previa inicialmente a utilização de uma nova sistemática para substituir a busca tradicional. O objetivo era adotar os mesmos mecanismos

utilizados por grandes sistemas de pesquisa, fazendo a busca textual em uma base paralela, o que tornaria os resultados muito mais rápidos. Contudo, a equipe de desenvolvimento relatou que, por tratar-se de um recurso novo ao qual não estavam habituados a lidar, haveria então a necessidade de aprofundar os conhecimentos técnicos. Este processo de curva de aprendizagem afetou diretamente o tempo de desenvolvimento da ferramenta. Foram gastas muitas horas testando a ferramenta, uma vez que o requisito não funcional era permitir a maior velocidade possível nas buscas, conforme registrado nos relatórios apresentados.

Requisitos não funcionais, tais como o desempenho esperado de um sistema, não podem ser medidos em pontos de função. Entretanto, no caso do sistema analisado, foi necessário gastar mais tempo neste tipo de requisito, do que na construção de funcionalidades propriamente ditas. Desta forma, o custo final foi bem acima da média, se comparado ao valor esperado com base apenas no seu tamanho funcional.

6.3.3– SISTEMA 3: JURISPRUDÊNCIA

Por sua vez, a terceira aplicação escolhida para compor a amostra faz parte do sistema de Jurisprudência do TCE-GO. Este sistema ainda estava em fase de construção durante o processo de medição realizado, mas os dados aqui apresentados referem-se somente ao que já foi efetivamente finalizado e pago pelo Tribunal. A contagem de pontos de função das funcionalidades concluídas totalizou 57 PF, que custaram ao contratante a soma de R$ 16.692,10.

As funções entregues se referem apenas aos primeiros cadastros que compõem o sistema. Esta ferramenta foi construída sob a plataforma já consolidada de sistemas do TCE-GO. Isto significa que a maioria dos componentes utilizados no desenvolvimento já estava pronta ou possuía um modelo bem estabelecido. Sendo assim, sua construção foi bastante célere.

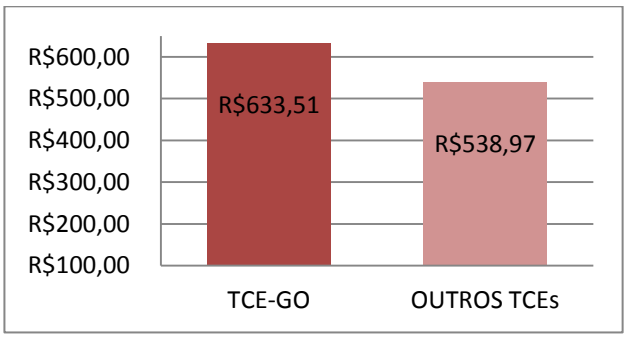
Diferente do outros dois sistemas da amostra, este aplicativo foi construído sob um contrato que não utilizava apenas a métrica homem-hora. Apesar de ainda não utilizar pontos de função, o novo modelo de contratação o TCE-GO estabeleceu níveis mínimos de serviço e entregas associadas a resultados, como recomenda o TCU. Ainda assim foi possível verificar que a quantidade de horas ainda é um dos parâmetros que compõe a métrica adotada pelo Tribunal.

Outro ponto que diferencia este dos demais sistemas avaliados, é que a fase de levantamento e análise de requisitos deste projeto foi iniciada com bastante antecedência por analistas do quadro próprio de servidores do TCE-GO, o que demandou menor esforço da equipe contratada para o início de suas atividades.

6.4 – Resultados

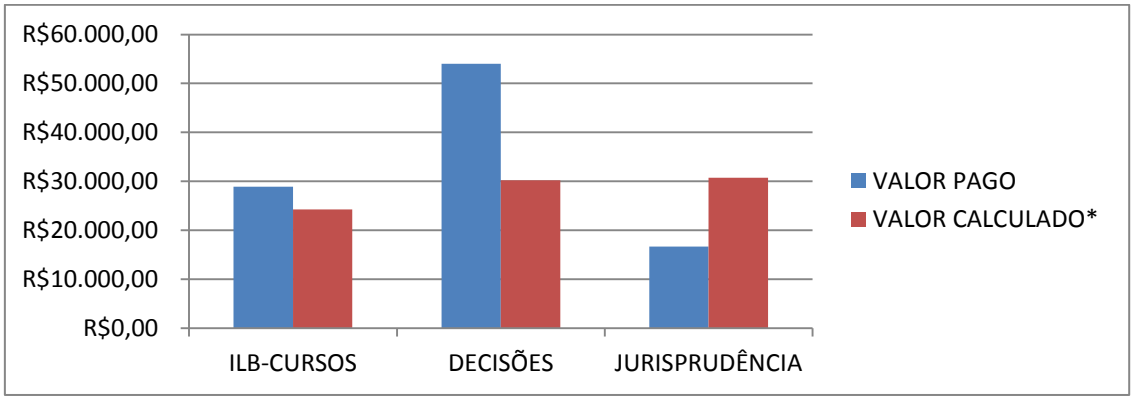
Após a análise dos dados, inferiu-se que o valor médio pago pelo TCE para cada ponto de função é 17,54% superior ao preço pago por outros tribunais de contas em contratos similares, conforme demonstrado no gráfico 1. Este resultado contraria a hipótese de que a utilização somente da métrica homem-hora permitiria o desenvolvimento de sistemas a custo menor.

Gráfico 1 – Comparativo do custo por ponto de função pago TCEs a partir de 2015



Fonte: Elaborado pelo autor

Gráfico 2 – Comparativo dos valores pagos pelo TCE-GO e valor calculado com base do tamanho funcional multiplicado pelo valor médio pago para cada PF por outros Tribunais



Fonte: Elaborado pelo autor

Observou-se ainda uma grande variação neste percentual, como pode ser verificado no gráfico 2. Enquanto o sistema que permite a consulta de Decisões apresentou quase 80% de sobrepreço, o sistema de Jurisprudência custou ao TCE-GO quase a metade do preço calculado utilizando o mesmo parâmetro. Neste sentido, é possível perceber que diversos fatores contribuíram diretamente para esta variação. Diversos dos aspectos que afetam a produtividade apresentados no capítulo 3, foram identificados nos projetos analisados, entre eles: a experiência e o conhecimento da equipe frente à adoção de novas tecnologias e o reaproveitamento de componentes.

Por outro lado, as limitações da métrica de pontos de função também refletiram nos resultados desta pesquisa. Ao medir somente os requisitos funcionais isto pode não refletir a quantidade de esforço necessária para construir um sistema. Ressalte-se que foi aplicada a técnica sem fator de ajuste, recurso que pode aumentar ou diminuir significativamente o total de pontos de função, conforme apresentado no capítulo 2.

Pelo exposto, verifica-se que o sobrepreço encontrado nos sistemas analisados é justificado por meiodas peculiaridades de cada projeto analisado. De igual forma, a estimativa baseada somente em pontos de função mostrou-se também menos econômica no caso do sistema Jurisprudência. Sendo assim, não é possível afirmar conclusivamente que a utilização da métrica homem-hora trouxe prejuízos diretos ao TCE-GO.

Ademais, o Tribunal não teve os gastos adicionais decorrentes de um processo de medição burocrático, mesmo assim exigiu e manteve um nível razoável de informação, o que permitiu conhecer exatamente em quais tarefas o tempo foi gasto. A despeito do sobrepreço apresentado, revelou-se que o TCE-GO exigiu e manteve um nível considerável de informações sobre a execução dos serviços. Tal fator permitiu conhecer exatamente em quais tarefas o tempo foi gasto, o que diminuiu a probabilidade de pagar por horas improdutivas. A corte ainda modificou recentemente a forma de contratação, estabelecendo níveis mínimos de serviço em conformidade com o modelo proposto pelo TCU.

CONCLUSÃO

A utilização de métricas de engenharia de *software* têm se ampliado ao longo dos últimos anos. Parte disso se deve às recomendações feitas pelo TCU. A corte federal buscou aprimorar a contratação de serviços de TI aduzindo que a utilização da métrica homem-hora traz consigo o problema de pagar por serviços não executados. Contar apenas as horas trabalhadas, sem medir os resultados efetivamente entregues, significa prejuízo e cabe a responsabilização do gestor que permite que esta situação se perpetue.

Para entender este quadro, o presente trabalho buscou apresentar os principais conceitos em torno das técnicas de medição do tamanho de *softwares* existentes. Foram descritos ainda os fatores que influenciam a produtividade das equipes e as recomendações do TCU sobre o assunto. Uma das alternativas demonstradas para medir com base em resultados é a utilização da análise de pontos de função. Foi dito que esta métrica tem sido amplamente adotada para medir o tamanho de sistemas de informação. Há quem ainda resista e afirme que a técnica é de difícil compreensão.

Ainda assim, os principais autores defendem que é válida sua utilização, desde que sejam ponderadas as particularidades de cada projeto. Restou bastante evidente que não existe uniformidade nos índices de produtividades e também não há que se falar em um preço universal para o ponto de função. Cada organização deve avaliar sua própria realidade, uma vez que o custo dos sistemas pode variar muito. Desta forma, as métricas precisam ser analisadas dentro do contexto na qual estão inseridas e sempre que for possível, devem ser validadas utilizando outras informações do projeto, tais como métricas de processo e produto, permitindo evoluir a qualidade dos produtos entregues.

A variação de custo do ponto de função ficou evidente neste estudo. A APF foi utilizada para medir uma amostra de sistemas desenvolvidos para o TCE-GO com o objetivo de demonstrar a compatibilidade dos valores gastos por aquele Tribunal com o desenvolvimento de sistemas. Apenas recentemente o TCE-GO passou a remunerar este tipo de contrato com base em resultados e níveis mínimos de serviço, conforme recomendado pelo TCU. E a dúvida era se os valores estavam de acordo com o preço praticado em outras instituições.

A aplicação prática da APF mostrou ser um método relativamente simples. De forma que o principal aspecto evidenciado neste estudo é que a medição utilizando APF é viável e pouco burocrática. Sendo assim, é possível inserir esta métrica para complementar qualquer processo de medição sem grandes impactos. A utilização desta técnica foi amparada nas recomendações expedidas pelo TCU em diversos dos seus acórdãos, conforme demonstrado.

A Corte de Contas da União é taxativa ao estabelecer que a medição deve ser baseada em resultados e não apenas da disponibilidade da mão de obra contratada. Entretanto, o TCE-GO por um longo tempo manteve formas de medição de produtos de *software*, baseadas em remuneração homem-hora. Esperava-se que esta métrica pudesse ter sido mais econômica devido à facilidade e agilidade nas medições. Contudo, havia o risco de que o Tribunal houvesse pagado pelo que não foi efetivamente entregue, admitindo-se também a possibilidade de prejuízos.

Os dados coletados, dados sobre os valores pagos por outros Tribunais de Contas para cada ponto de função, foram utilizados para comparar com valor pago pelo TCE. A ampla documentação mantida pelo TCE-GO permitiu medir o tamanho funcional de uma amostra de seus sistemas. Optou-se por uma amostra heterogênea, visando demonstrar situações típicas e atípicas para validar o quanto as variáveis poderiam oscilar. As análises quantitativas foram complementadas com avaliações qualitativas que envolveram pontos controversos sobre os pontos de função.

Os resultados finais demonstraram que a utilização de métricas baseadas somente em horas gastas não gerou custos menores como era esperado. Apesar de possuir indicadores de produtividade parecidos com os descritos neste trabalho, descobriu-se que o valor pago pelo TCE-GO foi quase 20% maior para cada ponto de função. A princípio poderiam ser apontados prejuízos nesta ordem, entretanto, o contexto de cada projeto e as especificidades da métrica de pontos de função evidenciados se mostraram compatíveis com os fatores que influenciam a produtividade.

REFERÊNCIAS

ALBRECHT, Allan J.; GAFFNEY, John E. **Software Function, Source Lines of Code, and Development Effort Prediction:** A Software. Science Validation. IEEE Transf. on Software Engineering. 1983. Disponível em: <http://ieeexplore.ieee.org/iel5/32/35937/01703110.pdf>. Acesso em: 1 dez. 2016.

AVILA, Claudio Henrique. **Produtividade das linguagens em pontos por função (APF)**. 24 de abril de 2012. Disponível em: <https://dicas.supera.com.br/dicas/dx/Produtividade_das_linguagens_em_pontos_por_função_(APF)>. Acesso em: 6 dez. 2016.

BASILI, Victor R.; CALDIERA, Gianluigi.; ROMBACH, H. Dieter. **Goal Question Metric Paradigm**. Encyclopedia of Software Engineering, John Wiley & Sons, 1994. p. 528-532. Disponível em: <https://www.cs.umd.edu/~basili/publications/technical/T89.pdf>. Acesso em: 25 nov. 2016.

BFPUG. Brazilian Function Point Users Group. **Qual a produtividade do JAVA?** Disponível em: <http://www.bfpug.com.br/Produtividade_Java.htm>. Acesso em: 6 dez. 2016.

BRASIL. Constituição (1988). **Constituição da República Federativa do Brasil**. Brasília, DF: Senado Federal, Centro Gráfico, 1988. Disponível em: <http://www.planalto.gov.br/ccivil_03/constituicao/constituicaocompilado.htm>. Acesso em: 11 dez. 2016.

BRASIL. Tribunal de Contas do Estado de Pernambuco. **Contrato nº 018/2015**. Referente à prestação de serviços presenciais e não presenciais de desenvolvimento, manutenção e testes de sistemas de informação em regime de fábrica de software, firmado com a empresa Ivia Serviços de Informática Ltda, conforme Processo Licitatório nº 84/2014 – Pregão Presencial nº 58/2014. Disponível em: <http://sistemas.tce.pe.gov.br/audinArquivos/licon/contrato/780/LICON_Contrato_780_2015_018_357848.pdf>. Acesso em: 2 jan. 2017.

BRASIL. Tribunal de Contas do Estado de Mato Grosso. **Contrato nº 51/2013**. Referente a serviços especializados em tecnologia da informação na área de desenvolvimento, manutenção e suporte de sistemas utilizando tecnologia Genexus, firmado com a empresa Ábaco Tecnologia de Informação Ltda, conforme procedimento de Pregão Presencial nº 25/2013 e 1º Termo Aditivo. Disponível em: <http://www.tce.mt.gov.br/arquivos/downloads/00054416/Contrato%2051-13%20Abaco.pdf>. Acesso em 2 jan. 2017.

BRASIL. Tribunal de Contas do Estado de São Paulo. **Contrato nº 063/2015**. Referente a prestação de serviços de desenvolvimento e manutenção de sistemas de informação nas plataformas Java, NET e Android em regime de Fábrica de Software, firmado com a empresa SINN Serviços E Soluções Em Tecnologia Ltda, conforme procedimento de Pregão Eletrônico nº 55/2015. Disponível em: <http://www4.tce.sp.gov.br/licitacao/2252902615>. Acesso em: 2 jan. 2017.

BRASIL. **Decreto nº 2.271, de 7 de julho de 1997**. Dispõe sobre a contratação de serviços pela Administração Pública Federal direta, autárquica e fundacional e dá outras providências. Disponível em: <http://www.planalto.gov.br/ccivil_03/decreto/d2271.htm>. Acesso em: 11 dez. 2016.

BRASIL. Tribunal de Contas do Estado de Goiás. **Edital da Concorrência nº 001/2016**. Disponível em: <http://www.tce.go.gov.br >. Acesso em: 20 mar. 2017.

BRASIL. Secretaria de Logística e Tecnologia da Informação (SLTI) do Ministério do Planejamento. **Instrução Normativa nº 4**. Dispõe sobre o processo de contratação de Soluções de Tecnologia da Informação pelos órgãos integrantes do Sistema de Administração dos Recursos de Tecnologia da Informação (SISP) do Poder Executivo Federal. Brasília, DF, 11 de setembro de 2014. Disponível em: <http://www.governoeletronico.gov.br/documentos-e-arquivos/1%20-%20IN%204%20%2011-9-14.pdf> . Acesso em: 11 dez. 2016.

BRASIL. **Lei nº 8.666, de 21 de junho de 1993**. Regulamenta o art. 37, inciso XXI, da Constituição Federal, institui normas para licitações e contratos da Administração Pública e dá outras providências. Disponível em: <http://www.planalto.gov.br/ccivil_03/leis/L8666cons.htm>. Acesso em: 11 dez. 2016.

BRASIL. Tribunal de Contas da União. **Nota Técnica nº 6/2010** – SEFTI/TCU – versão 1.3. Aplicabilidade da Gestão de Nível de Serviço como mecanismo de pagamento por resultados em contratações de serviços de TI pela Administração Pública Federal. Brasília, DF, 10 de novembro de 2015. Disponível em: <http://portal.tcu.gov.br/lumis/portal/file/fileDownload.jsp?fileId=8A8182A150D20B5E0150F69D58BB2C7B&inline=1>. Acesso em: 11 dez. 2016.

BRASIL. Tribunal de Contas da União. Processo de Contas. Auditoria de Natureza Operacional. Acórdão nº 2.362/2015 – Plenário. **Processo nº 002.116/2015-4**. Brasília, DF, 23 de setembro de 2015. Relator Ministro Augusto Nardes. Disponível em: <https://contas.tcu.gov.br/sagas/SvlVisualizarRelVotoAcRtf?codFiltro=SAGAS-SESSAO-ENCERRADA&seOcultaPagina=S&item0=537247>. Acesso em: 11 dez. 2016.

BRASIL. Tribunal de Contas da União. Processo de Contas. Relatório de Levantamento. Acórdão nº 1.603/2008 – Plenário. **Processo nº 008.380/2007-1**. Brasília, DF, 13 de agosto de 2008. Relator Ministro Guilherme Palmeira. Disponível em: <http://www.tcu.gov.br/Consultas/Juris/Docs/judoc/Acord/ 20080814/008-380-2007-1-GP.doc>. Acesso em: 9 dez. 2016.

BRASIL. Tribunal de Contas da União. Processo de Contas. Relatório de Auditoria. Acórdão nº 1.558/2003 – Plenário. **Processo nº 008.693/2003-3**. Brasília, DF, 15 de outubro de 2003. Relator Ministro Augusto Sherman. Disponível em: <http://www.tcu.gov.br/Consultas/Juris/Docs/judoc/Acord/20031022/TC%20008.693.doc>. Acesso em: 9 dez. 2016.

BRASIL. Tribunal de Contas da União. Processo de Contas. Relatório de Auditoria. Acórdão nº 2.471/2008 – Plenário. **Processo nº 019.230/2007-2**. Brasília, DF, 5 de novembro de 2008. Relator Ministro Benjamin Zymler. Disponível em: <http://www.tcu.gov.br/Consultas/Juris/Docs/judoc/Acord/20081110/019.230%202007-2-MIN-BZ.rtf>. Acesso em: 11 dez. 2016.

BRASIL. Ministério do Planejamento, Orçamento e Gestão. Secretaria de Logística e Tecnologia da Informação. **Roteiro de Métricas de Software do SISP**: versão 2.1. Secretaria de Logística e Tecnologia da Informação do MPOG. Brasília, 2015.

BRASIL. Serviço Federal de Processamento de Dados (SERPRO). **Roteiro SERPRO de Contagem de Pontos de Função e Estimativas**. Versão 7.0, junho de 2015. Disponível em: <http://www.pgfn.fazenda.gov.br/acesso-a-informacao/tecnologia-da-informacao/ Roteiro_Contagem_ PF_SERPRO_%207.pdf>. Acesso em: 6 dez. 2016.

BRASIL. Disponível em: <http://www.pgfn.fazenda.gov.br/acesso-a-informacao/tecnologia-da-informacao/ Roteiro_Contagem_ PF_SERPRO_%207.pdf>. Acesso em: 21 mar. 2017.

CAMPOS, Carlos Joaquim Esteves de. **APF Análise de Pontos de Função**. 1. ed. São Paulo: Nelpa, 2011.

CAVALCANTI, Augusto Sherman. **O novo modelo de contratação de soluções de TI pela Administração Pública**. 2. ed. Belo Horizonte: Fórum, 2015. p. 26.

DEKKERS, Carol A. Pontos de Função e Medidas. O que é um Ponto de Função?. **QAI Journal**. 1998. Disponível em: <http://www.bfpug.com.br/Artigos/DekkersPontosDeFuncaoEMedidas.Htm>. Acesso em: 30 nov. 2016.

EJIOGU, Lem O. **Software Engineering with Formal Metrics**. QED Technical Publishing Group, 1991. p. 37. apud PRESSMAN, Roger S. **Software Engineering with Formal Metrics**. QED Technical Publishing Group, 1991. p. 542.

ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistemas de Banco de Dados**. 6.ed. Pearson Education.

FATTO. **Qual indicador de produtividade (PF/homem-mês) ou taxa de entrega (horas/PF) deve ser usado para estimar o esforço de projetos de software?.** Disponível em: <http://www.fattocs.com/pt/faq-36.html>. Acesso em: 6 dez. 2016.

FENTON, Normam E. **Software measurement:** a necessary scientific basis. IEEE Trans. Software Engineering, v. SE-20, n. 3, p. 199-206, mar. 1994. apud PRESSMAN, Roger S. **Software measurement:** a necessary scientific basis. IEEE Trans. Software Engineering, v. SE-20, n. 3, p. 540, mar. 1994.

FENTON, Normam E. **Software metrics:** a rigorous approach. Chapman & Hall, 1991. apud PRESSMAN. Roger S. **Engenharia de Software:** uma abordagem profissional. 7. ed. Porto Alegre: Almedina, 2011. p. 540.

GUARIZZO, Karina. **Métricas de Software**. 2008. Monografia (Bacharelado em Ciência da Computação) – Curso de Ciência da Computação da Faculdade de Jaguariúna, Jaguariúna.

HAZAN, Claudia; STAA, Arndt von. Análise e Melhoria de um Processo de Estimativas de Tamanho de Projetos de Software. In: **VI SIMPROS – Simpósio Internacional de Melhoria de Processos de Software**. São Paulo, 26 novembro de 2004. Disponível em: <http://www.simpros.com.br/Apresentacoes_PDF/Artigos/Art_04_ Simpros2004.pdf>. Acesso em: 6 dez. 2016.

IFPUG. **Function Point Counting Practices Manual**. Release 4.3.1. IFPUG, 2010.

IFPUG. **Manual de Práticas de Contagem de Pontos de Função**. Versão 4.3.1. Tradução Brasileira do CPM. IFPUG, 2010.

JONES, Carper. **Proposed Suite of Thirteen Functional Metrics for Economic Analysis**. The IFPUG Guide to IT and Software Measurement. IFPUG. 2012. p. 20-25.

PARO, Caio Juliano. **Medidas de tamanho de desenvolvimento e de melhorias de software**. 2005. Disponível em: <http://www.bfpug.com.br/Artigos/Medidas%20de%20tamanho%20de%20 desenvolvimento%20e%20de%20melhorias%20de%20software.doc>. Acesso em: 4 dez. 2016.

PRESSMAN, Roger S. **Engenharia de Software:** uma abordagem profissional. 7. ed. Porto Alegre: AMGH, 2011.

ROCHE, Jhon M. **Software Metrics and Meansurement Principles**. ACM SIGSOFT Software Engineering Notes. Nova York, p. 76-85, jan. 1994.

SIMÕES, Guilherme. **Análise de Pontos de Função:** medição, estimativas e gerenciamento de projetos de software. Apresentado na SUCESU-ES em abril de 2004 Disponível em: <http://www.fattocs.com/files/pt/apresentacoes/apf-sucesues.pdf>. Acesso em: 9 dez. 2016.

SOMMERVILLE, Ian. **Engenharia de Software**. 6. ed. São Paulo: Pearson, 2005.

VAZQUEZ, Carlos Eduardo; SIMÕES, Guilherme Siqueira; ALBERT, Renato Machado. **Gestão de Contratos de Desenvolvimento de Software com a Análise de Pontos de Função**. Apresentado no Congresso Nacional da SUCESU-ES em 28 de abril de 2004. Disponível em: <http://fattocs.com/files/pt/artigos/sucesu2004.pdf>. Acesso em: 4 dez. 2016.

VIEIRA, Antonieta Pereira; VIEIRA, Henrique Pereira; FURTADO, Madeline Rocha; FURTADO, Monique Rafaella Rocha. **Gestão de contratos de terceirização na Administração Pública**: teoria e prática. 6. ed. Belo Horizonte: Fórum, 2015.

WOLFART, Daniele. **Estimativa de tamanho de software por meio da técnica de Análise de Pontos de Função**. 2012. Monografia (Especialização em Engenharia de Software). Universidade Tecnológica Federal do Paraná, Medianeira, 2012.