Roteiro de Métricas de Software do SISP

Versão **2.2**

(Não Formatado)

Secretaria de Tecnologia da Informação
Ministério do Planejamento, Desenvolvimento e Gestão

SISP
Secretaria de Tecnologia da Informação
Ministério do Planejamento
BRASIL

Roteiro de Métricas de Software do SISP
Versão 2.2 (Não Formatado)

Presidente da República

Michel Miguel Elias Temer Lulia

Ministro do Ministério do Planejamento, Desenvolvimento e Gestão

Dyogo Henrique de Oliveira

Secretário de Logística e Tecnologia da Informação

Marcelo Daniel Pagotti

Diretora do Departamento de Governança e Sistemas de Informação

Ana Carolina Romao Degaspari

Coordenador-Geral de Sistemas de Informações

Orlando Batista da Silva Neto

Grupo de Métricas de Software do SISP

Aline Gonçalves dos Santos (STI/MP)

Felipe Corradi Carminati (STI/MP)

Maurício de Alves Lacerda (STI/MP)

Equipe de Elaboração da Versão 2.2

Aline Gonçalves dos Santos (STI/MP)

Felipe Corradi Carminati (STI/MP)

Maurício de Alves Lacerda (STI/MP)

Orlando Batista da Silva Neto (STI/MP)

MINISTÉRIO DO PLANEJAMENTO, DESENVOLVIMENTO E GESTÃO
Secretaria de Logística e Tecnologia da Informação

Roteiro de Métricas de Software do SISP
Versão 2.2 (Não Formatado)

Brasília

2016

Normalização Bibliográfica: CODIN/CGPLA/DIPLA

B823r

Brasil. Ministério do Planejamento, Desenvolvimento e Gestão. Secretaria de
Tecnologia da Informação.
Roteiro de Métricas de Software do SISP: versão 2.2 / Ministério do Planejamento, Desenvolvimento e Gestão. Secretaria de Logística e Tecnologia da Informação. – Brasília : MP, 2016.
83 p.: il.

1. Tecnologia da Informação. 2. Roteiro de Métrica de Software.
3. Ponto de Função. 4. Sistema de Administração dos Recursos de Tecnologia da Informação – SISP. I. Título.

CDU 004.4

Sumário

Índice de Figuras

Índice de Tabelas

1. Introdução

Diversas instituições públicas e privadas têm utilizado a métrica Ponto de Função (PF) nas estimativas e dimensionamento de tamanho funcional de projetos de software devido aos diversos benefícios de utilização desta métrica, destacando-se: regras de contagem objetivas, independência da solução tecnológica utilizada e facilidade de estimativa nas fases iniciais do ciclo de vida do software. É importante ressaltar que a Instrução Normativa SLTI/MP N° 4, de 11 de setembro de 2014, recomenda o uso de métricas em contratos de projetos de software, restringindo o uso da métrica de esforço homem-hora. Além disso, a Portaria SLTI/MP nº 31, de 29 novembro de 2010, recomenda o uso da métrica Ponto de Função para os órgãos integrantes do SISP, bem como a adoção do Roteiro de Métricas de Software do SISP na contratação de serviços de desenvolvimento e manutenção de soluções de software. O Tribunal de Contas da União (TCU) tem publicado vários acórdãos que recomendam a utilização da métrica Ponto de Função Não Ajustado em contratos de prestação de serviços de desenvolvimento e manutenção de sistemas, entre os quais podem ser citados:

- Acórdão nº **1.782/2007**: recomenda o uso da métrica Ponto de Função como forma de pagamento dos serviços contratados de desenvolvimento e manutenção de sistemas, ao invés de se realizar a conversão dos pontos de função em horas, baseado na produtividade média da tecnologia empregada.
- Acórdão nº **1.910/2007**: em atenção ao princípio da eficiência, faz duas recomendações: adotar a técnica de medição por ponto de função sem ajustes pelas características da aplicação (pontos de função não ajustados) e diferenciar, na fórmula de cálculo, os custos dos pontos de função para desenvolver novas funcionalidades, daqueles relativos a supressões ou alterações de funcionalidades existentes.
- Acórdãos nºs **1.125/2009** e **1.274/2010**: determinam não vincular a métrica de tamanho funcional (Ponto de Função) com a de esforço (homem-hora).
- Acórdãos nºs **2.348/2009** e **1.647/2010**: reforçam a determinação de não usar qualquer tipo de fator de ajuste na medição por pontos de função na contratação de serviços de desenvolvimento de software, para impossibilitar alterações na remuneração da funcionalidade medida, por se basear em interpretação subjetiva dos níveis das características gerais de sistemas, em desacordo com o previsto no art. 54, § 1º, da Lei nº 8.666/93 e art. 2º, XXIV, da IN SLTI nº 04/2014. Além disso, o acórdão 1.647/2010 determina que não se use exclusivamente o Manual de Práticas de Contagem (CPM) do IFPUG nas contratações de serviços de desenvolvimento, e que sejam adicionadas cláusulas complementares que elucidem pontos não abordados pelo CPM; e recomenda a diferenciação, em sua fórmula de cálculo, dos custos de pontos de função para o desenvolvimento completo de uma funcionalidade (todas as fases do ciclo de desenvolvimento) daqueles necessários à execução de apenas uma fase do ciclo.

O Manual de Práticas de Contagem de Pontos de Função (CPM 4.3) [IFPUG, 2010b], publicado pelo *International Function Point Users Group* (IFPUG), define as regras de contagem de pontos de função. É importante ressaltar que a métrica Ponto de Função foi concebida como uma medida de tamanho funcional para projetos de desenvolvimento e de melhoria (manutenção evolutiva) de software. No entanto, os projetos de software não estão limitados a projetos de desenvolvimento e de melhoria. Desta forma, torna-se essencial a definição de métricas para dimensionar o tamanho de outros tipos de projetos de manutenção, os quais são itens não mensuráveis pelo CPM.

Além disso, a contagem de pontos de função é baseada no projeto lógico da aplicação (*logical design*). Nas fases iniciais do ciclo de vida do software, o insumo para a definição das estimativas do projeto é um documento inicial de requisitos, por exemplo: documento de visão ou algum outro tipo de especificação elaborada pelo analista de negócios. Assim, torna-se importante o estabelecimento de métodos para estimar o tamanho dos projetos de software nas fases iniciais do ciclo de vida. Outro ponto a ser destacado é a importância da definição de métodos para geração de estimativas de prazo e custo dos projetos de software a partir do tamanho funcional estimado do projeto.

É importante frisar também que o CPM é um documento que se destina a mensurar o tamanho funcional de projetos de software, não tendo por objetivo principal suportar contratos de prestação de serviços de desenvolvimento e manutenção de sistemas. Assim, torna-se necessário criar roteiros complementares, contemplando questões não abordadas pelo manual do IFPUG, mas vivenciadas pelos órgãos e entidades do SISP.

O restante deste documento encontra-se organizado da seguinte forma: o capítulo 2 descreve os objetivos e as referências consultadas para a elaboração deste documento; o capítulo 3 apresenta algumas definições básicas para a contagem de pontos de função; o capítulo 4 define métricas baseadas em Ponto de Função para dimensionar projetos de desenvolvimento e vários tipos de projetos de manutenção de software; o capítulo 5 estabelece diretrizes para contagem de múltiplas mídias; o capitulo 6 define um processo de estimativas e recomendações para o gerenciamento de projetos contratados com base em métricas; o capítulo 7 traz uma proposta para conciliar o uso da métrica Ponto de Função em contratações de desenvolvimento de software usando métodos ágeis com o objetivo de minimizar os riscos para o tratamento de mudanças em funcionalidades; o capítulo 8 apresenta algumas atividades que não devem ser consideradas nas contagens de pontos de função; o capítulo 9 apresenta o processo de revisão deste guia de contagem; finalmente, o capítulo 10 conclui o documento, apresentando sugestões para trabalhos futuros.

2. Objetivo

Este documento tem como objetivo principal apresentar um roteiro de métricas, com base nas regras de contagem de pontos de função do Manual de Práticas de Contagem (CPM 4.3), para vários tipos de projetos de desenvolvimento e de manutenção de sistemas, promovendo o uso de métricas objetivas nos contratos de prestação de serviços desses projetos. Além da contagem de pontos de função, este roteiro apresenta um processo de estimativas com base na métrica Ponto de Função, visando apoiar as organizações nas estimativas de tamanho, custo, prazo e esforço de seus projetos desenvolvidos internamente ou contratados.

A versão 2.2 apresenta uma pequena variação com relação à versão anterior 2.1 especificamente para adequação às recomendação publicadas no Relatório por Área de Gestão nº 5 da CGU que tratou da contratação de serviços de desenvolvimento e manutenção de sistemas mensurados em Pontos de Função:

- Criação da seção 5.2 com orientações sobre a mensuração de log, trilha de auditoria e histórico
- Criação da seção 5.3 para reforçar a necessidade de identificação correta de um processo elementar.
- Ajuste e inclusão de itens no capítulo de Atividades Sem Contagem de Pontos por Função (capítulo 8).

A alteração do modelo de contratação de software, decorrente da implantação de um processo de medição de software mais objetivo, requer uma mudança cultural, devido à mudança do paradigma homem-hora para a nova forma de contratação com base na métrica Ponto de Função. Este roteiro tem como propósito apoiar os órgãos e entidades do SISP nessa mudança cultural. É recomendável a leitura do Anexo IV, pois apresenta vários tópicos importantes a serem observados pelos órgãos contratantes na utilização do modelo de contratação de software usando a métrica PF.

Duas premissas foram consideradas na elaboração deste roteiro:

• **Simplicidade:** Este roteiro deve ser simples para incentivar os órgãos e entidades do SISP a utilizar a métrica Ponto de Função como medida padrão no estabelecimento de contratos de prestação de serviços de desenvolvimento e manutenção de sistemas.

• **Consistência:** Este roteiro deve definir critérios objetivos, visando prover a consistência no uso de métricas em contratos de prestação de serviços de desenvolvimento e manutenção de sistemas. Deste modo, dois profissionais ao aplicarem o roteiro no dimensionamento de um projeto de software devem obter o mesmo resultado.

3. Contagem de Pontos de Função

A métrica PF mede o **tamanho funcional** de um projeto de software, observando as funcionalidades implementadas, considerando a visão do usuário. O tamanho funcional é definido como “tamanho do software derivado pela quantificação dos requisitos funcionais do usuário” [Dekkers, 2003]. A métrica PF é independente da metodologia e tecnologia utilizadas. A Análise de Pontos de Função (APF) é um método padrão para a medição de projetos de desenvolvimento e de manutenção de sistemas, visando estabelecer uma medida de tamanho do software em pontos de função, com base na quantificação das funcionalidades solicitadas e entregues, sob o ponto de vista do

usuário. Assim, a APF tem como objetivo medir o que o software faz, por meio de uma avaliação padronizada dos requisitos de negócio do sistema.

O Manual de Práticas de Contagem (CPM) [IFPUG, 2010b] apresenta as regras de contagem de pontos de função de projetos de desenvolvimento, projetos de melhoria e aplicações implantadas. A Figura 1 ilustra o procedimento de contagem de pontos de função, descrito nas seções seguintes.

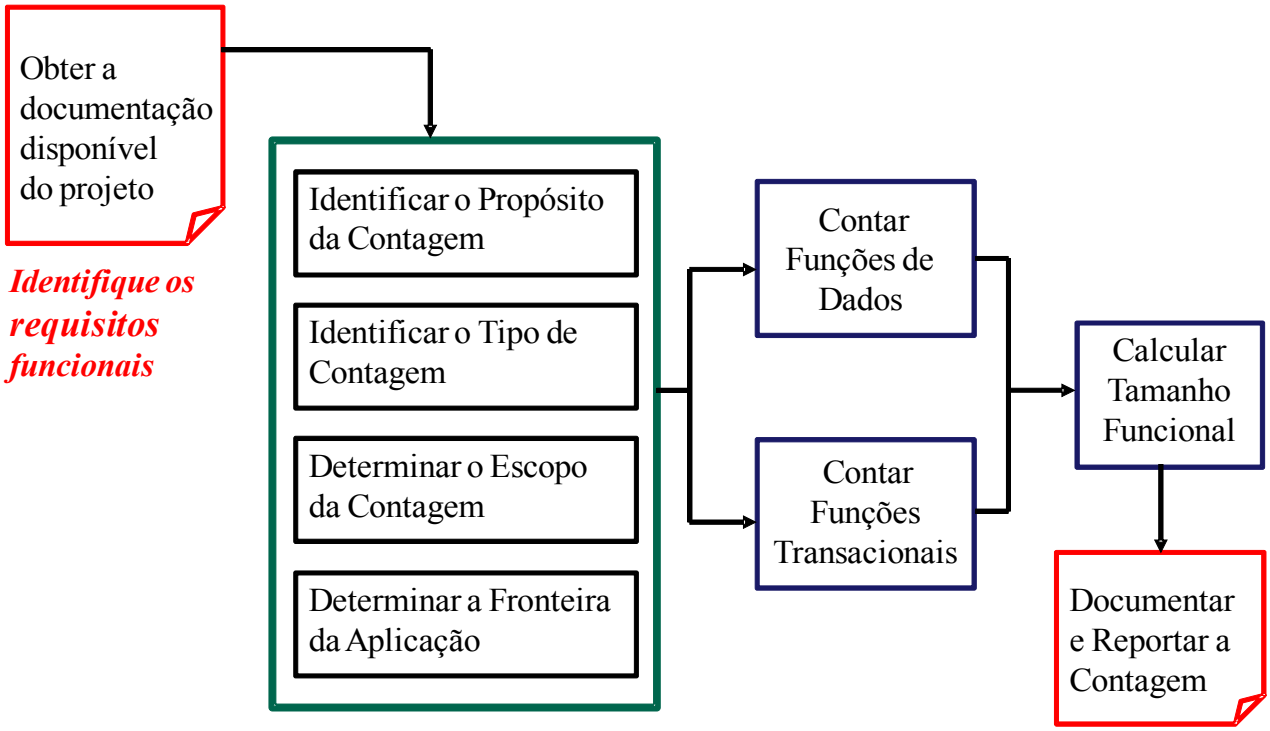

Figura 1: Procedimento de Contagem de Pontos de Função

3.1 Determinar Propósito, Tipo e Escopo da Contagem e Fronteira da Aplicação

A contagem de pontos de função se inicia com a análise da documentação disponível do projeto em questão, visando a identificação dos requisitos funcionais. O próximo passo é o estabelecimento do propósito da contagem, o qual fornece uma resposta para uma questão de negócio a ser resolvida, por exemplo: necessidade de dimensionar um projeto de um novo sistema para auxiliar o processo de contratação do mesmo. Com base no propósito da contagem são definidos o escopo da contagem e o tipo de contagem. O escopo da contagem identifica quais funcionalidades serão incluídas na contagem de pontos de função, e o tipo de contagem identifica se o projeto é de desenvolvimento, de melhoria ou aplicação instalada. A **fronteira da aplicação**, que é a interface conceitual que indica o limite lógico entre o sistema sendo medido e os usuários (também entre outras aplicações), deve ser definida com base na visão do usuário, desconsiderando questões de implementação. Deve-se ressaltar que toda contagem de pontos de função é realizada dentro de uma fronteira estabelecida.

O estabelecimento da fronteira da aplicação pode ser subjetivo, por exemplo, em uma aplicação com vários módulos, a fronteira pode ser estabelecida para cada módulo ou subsistema ou, ainda, pode-se considerar toda a aplicação, dependendo da visão do usuário. De fato, a definição da fronteira depende de processos de negócios, além disso, o posicionamento da fronteira influencia fortemente a contagem de pontos de função. Desta forma, devido a essa subjetividade, em editais para contratação de projetos de

manutenção é fortemente recomendado a definição das fronteiras de todas as aplicações a serem contratadas. Os roteiros de contagem dos órgãos e entidades também devem definir as fronteiras das aplicações implantadas em um anexo, e este deve ser atualizado sempre que for implantada uma nova aplicação. Mais informações sobre esse assunto podem ser encontradas no Anexo IV.

3.2 Identificar Funções de Dados e Funções Transacionais

Uma vez estabelecida a fronteira da contagem, o próximo passo é o mapeamento dos requisitos de dados e de funções transacionais para os tipos funcionais da APF, a saber:

• **Arquivo Lógico Interno (ALI):** é um grupo de dados, logicamente relacionados, reconhecido pelo usuário, mantido por meio de um processo elementar da aplicação que está sendo contada.

• **Arquivo de Interface Externa (AIE):** é um grupo de dados, logicamente relacionados, reconhecido pelo usuário, mantido por meio de um processo elementar de uma outra aplicação e referenciado pela aplicação que está sendo contada. O AIE é obrigatoriamente um ALI de outra aplicação.

• **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.

Após a identificação dos tipos funcionais para cada requisito funcional definido no documento de requisitos do sistema, deve-se avaliar a complexidade (Baixa, Média, Alta) e a contribuição funcional do mesmo para a contagem de pontos de função, observando as regras de contagem de pontos de função descritas no CPM. A identificação e a avaliação das complexidades dos tipos funcionais não podem ser realizadas de maneira subjetiva. A contagem de pontos de função deve seguir rigorosamente as regras de contagem do CPM e as definições complementares do roteiro de métricas do órgão, e deve ser realizada por profissionais capacitados do órgão.

A Tabela 1 apresenta a contribuição dos tipos funcionais na contagem de pontos de função.

	Complexidade		
Tipo Funcional	**Baixa**	**Média**	**Alta**
Arquivo Lógico Interno (ALI)	7 PF	10 PF	15 PF
Arquivo de Interface Externa (AIE)	5 PF	7 PF	10 PF

Entrada Externa (EE)	3 PF	4 PF	6 PF
Saída Externa (SE)	4 PF	5 PF	7 PF
Consulta Externa (CE)	3 PF	4 PF	6 PF

Tabela 1: Contribuição Funcional dos Tipos Funcionais (Fonte: CPM 4.3)

3.3 Calcular Tamanho Funcional

O Manual de Práticas de Contagem do IFPUG define dois tipos de projetos de software, a saber:

• **Projeto de Desenvolvimento:** projeto para desenvolver e entregar a primeira versão de uma aplicação de software. Seu tamanho funcional é a medida das funcionalidades entregues ao usuário no final do projeto. Também considera-se as funcionalidades de conversão de dados, caso seja requisitado no projeto a migração ou carga inicial de dados para a nova aplicação.

• **Projeto de Melhoria**: projeto de manutenção evolutiva ou melhoria funcional. Seu tamanho funcional é a medida das funcionalidades incluídas, alteradas e excluídas ao final do projeto. Também considera-se as funcionalidades de conversão de dados, caso seja requisitado a migração ou carga inicial de dados no projeto de melhoria.

Seguem abaixo as definições dos termos técnicos da Análise de Pontos de Função utilizados nas fórmulas de dimensionamento de projetos de software propostas neste roteiro:

• **PF_INCLUÍDO:** pontos de função associados às novas funcionalidades que farão parte da aplicação após um projeto de desenvolvimento ou de manutenção.

• **PF_ALTERADO:** pontos de função associados às funcionalidades existentes na aplicação que serão alteradas no projeto de manutenção.

• **PF_EXCLUÍDO:** pontos de função associados às funcionalidades existentes na aplicação que serão excluídas no projeto de manutenção.

• **PF_CONVERSÃO:** pontos de função associados às funcionalidades de conversão de dados dos projetos de desenvolvimento ou de manutenção. Exemplos de funções de conversão incluem: migração ou carga inicial de dados para popular as novas tabelas criadas (Entradas Externas) e relatórios associados à migração de dados, caso requisitado pelo usuário (Saídas Externas ou Consultas Externas). Observe que os dados carregados em um processo de migração não devem ser contados como Arquivos de Interface Externa.

Este roteiro recomenda a supressão do PF_CONVERSÃO das fórmulas de contagem de pontos de função de projetos de desenvolvimento e de melhoria nos casos específicos onde for caracterizado um esforço relativamente maior dessa atividade. Por exemplo, os projetos que envolvem a migração de dados de banco de dados hierárquico para banco de dados relacional e o tratamento de funções complexas de migração de dados. Nesses casos, recomenda-se tratá-los como projetos separados de migração de dados, descritos na seção 4.3.

3.4 Requisitos Não Funcionais

A métrica Ponto de Função é uma métrica de tamanho funcional, ou seja, dimensiona projetos de software com base nos requisitos funcionais da aplicação, não

contemplando diretamente os requisitos não funcionais do projeto.

Nesse sentido, em contratos de software baseados na métrica Ponto de Função é fundamental definir claramente no edital os requisitos não funcionais do projeto a serem atendidos pela empresa contratada. Os requisitos não funcionais impactam no esforço e, consequentemente, no custo do projeto.

Os requisitos não funcionais estão associados aos aspectos qualitativos de um software, considerando aspectos relacionados ao uso do software. Seguem abaixo alguns tipos de requisitos não funcionais, com exemplos, que podem ser mencionados nos editais:

- Usabilidade: a solução deve atender aos requisitos dos Padrões Web em Governo Eletrônico (e-PWG) – Cartilha de Usabilidade; a aplicação deve ter help on-line de sistema, tela e campo (sensível a contexto); a aplicação deve ser disponibilizada nos idiomas Português, Espanhol e Inglês.

- Técnicos: a aplicação deve funcionar adequadamente nos navegadores: Internet Explorer 7.0 ou superior e Mozilla Firefox 3.0 ou superior; a solução deve ser desenvolvida em linguagem Java com banco de dados PostgreSQL; para o desenvolvimento da solução, deve ser utilizado preferencialmente um dos seguintes frameworks Java: Demoiselle, Jaguar e MDArt; a solução deve atender aos requisitos do e-PWG; deve utilizar as ferramentas AWSTATS e Google Analytics para gerar estatísticas de acesso.

- Segurança: a aplicação deve realizar controle de segurança dos dados de acordo com politica de backup definida em conformidade com a norma ISO/IEC 27002.

- Acessibilidade: a solução deve ser aderente ao Modelo de Acessibilidade de Governo Eletrônico (e-MAG).

- Performance: o tempo de resposta da aplicação não deve exceder 10 segundos; a solução deve suportar até 1.000 acessos simultâneos.

- Interoperabilidade: a solução deve ser aderente aos Padrões de Interoperabilidade de Governo Eletrônico (e-PING).

4. Cálculo de Pontos de Função para o SISP

Este capítulo tem como propósito descrever os diversos tipos de projetos de software e definir métricas para seu dimensionamento baseadas nas regras de contagem de pontos de função do CPM.

Quanto à documentação de pequenos projetos de manutenção (menores que 100 PF), deve-se registrar a solicitação e documentar os requisitos do projeto de manutenção e da aplicação impactada pela demanda, de forma detalhada, visando apoiar a contagem de pontos de função da demanda. É importante também documentar as estimativas e a contagem de pontos de função. O Anexo II e Anexo III apresentam, respectivamente, um modelo de documento de requisitos e um modelo de documento de contagem de pontos de função para projetos de manutenção de pequeno porte (menores que 100 PF).

Cabe ressaltar que, em alguns casos, o órgão contratante pode não ter necessidade de contratar todas as fases do ciclo de vida do software. Dessa forma, a contratada será remunerada pela contagem de pontos de função considerando apenas os percentuais das fases contratadas, conforme os níveis percentuais sugeridos na Tabela 7 ou na metodologia do órgão (ver subseção 6.1.2.1). Exemplo: para um novo projeto de desenvolvimento de um sistema de treinamentos, que não exista a intenção de contratar as fases de requisitos e de testes, a contratada será remunerada pela contagem

de pontos de função desconsiderando os percentuais dessas fases.

Além disso, recomenda-se que as contagens de manutenção a partir do Roteiro de Métricas de Software do SISP sejam reportadas conforme determinado pelo CPM, ou seja, **S FP (IFPUG–IS–c)**, indicando que o resultado da contagem de pontos de função não mantém conformidade plena com o CPM e o padrão internacional de contagem de PF (ISO/IEC 20926:200x) e sim mantém conformidade com uma customização, neste caso, o Roteiro de Métricas de Software do SISP.

Assim: **S FP (IFPUG–IS–c)**

Onde:

S é o resultado da contagem de pontos de função;

FP (*Function Point*) é a unidade de tamanho do método FSM (F*unctional Size Measurement*) do IFPUG;

IS (*International Standard*) é o padrão internacional (ISO/IEC 20926:200x);

c representa um ou mais caracteres indicando que o resultado não mantém conformidade plena com o padrão internacional.

Exemplo: 250 PF***** (IFPUG–ISO/IEC 20926:200x–sisp)

**** FP na versão em português***

4.1 Projeto de Desenvolvimento

É o projeto para desenvolver e entregar a primeira versão de uma aplicação de software. Seu tamanho funcional é a medida das funcionalidades entregues ao usuário no final do projeto. Também considera-se as funcionalidades de conversão de dados. Segue a fórmula de cálculo utilizada no dimensionamento de projetos de desenvolvimento de software:

PF_DESENVOLVIMENTO = PF_INCLUIDO + PF_CONVERSÃO

Este roteiro recomenda a supressão do PF_CONVERSÃO das fórmulas de contagem de pontos de função de projetos de desenvolvimento quando for caracterizado um esforço relativamente maior dessa atividade, conforme descrito na seção 3.3.

4.2 Projeto de Melhoria

O Projeto de Melhoria (*enhancement*), também denominado de projeto de melhoria funcional ou manutenção evolutiva, está associado às mudanças em requisitos funcionais da aplicação, ou seja, à inclusão de novas funcionalidades, alteração ou exclusão de funcionalidades em aplicações implantadas.

Segundo o padrão IEEE Std 1219 [IEEE, 1998], esta manutenção seria um tipo de manutenção adaptativa, definida como: modificação de um produto de software existente para mantê-lo funcionando adequadamente em um ambiente que sofre mudanças. O projeto de melhoria é considerado um tipo de projeto de manutenção adaptativa com mudanças em requisitos funcionais da aplicação, ou seja, com funcionalidades incluídas, alteradas ou excluídas na aplicação, segundo o CPM 4.3.

Este roteiro separa o projeto de melhoria (quando as mudanças são associadas aos requisitos funcionais) do projeto de manutenção adaptativa (quando as mudanças

estão associadas aos requisitos não funcionais da aplicação). Um projeto de melhoria consiste em demandas de criação de novas funcionalidades (grupos de dados ou processos elementares), demandas de exclusão de funcionalidades (grupos de dados ou processos elementares) e demandas de alteração de funcionalidades (grupos de dados ou processos elementares) em aplicações implantadas em produção.

Segue a fórmula de cálculo utilizada no dimensionamento de projetos de melhoria de software:

PF_MELHORIA = PF_INCLUIDO + (FI x PF_ALTERADO) + (0,30 x PF_EXCLUIDO) + PF_CONVERSÃO

FI (Fator de Impacto) pode variar de 50% a 90% conforme condições abaixo:

A • **FI = 50%** para **funcionalidade de sistema desenvolvida ou mantida** por meio de um projeto de melhoria pela empresa contratada.

A2 • **FI = 75%** para **funcionalidade de sistema não desenvolvida ou mantida** por meio de um projeto de melhoria pela empresa contratada e **sem necessidade de redocumentação da funcionalidade.**

A3 • **FI = 90%** para **funcionalidade de sistema não desenvolvida ou mantida** por meio de um projeto de melhoria pela empresa contratada e **com necessidade de redocumentação da funcionalidade.** FI = 90% representa a adição de 15% como fator de redocumentação ao Fator de Impacto anterior (75%). Nesse caso, **a contratada deve redocumentar a funcionalidade mantida, gerando a documentação completa da mesma, aderente ao processo de software da contratante.** Se houver uma nova demanda de projeto de melhoria na funcionalidade em questão, será considerado que a contratada desenvolveu a funcionalidade. Observe que o percentual de 90% apenas será considerado na primeira demanda de projeto de melhoria em cada funcionalidade.

Este roteiro propõe um fator de redocumentação menor para projetos de manutenção (melhoria, corretiva e adaptativa) do que o fator proposto em projetos específicos de redocumentação (seção 4.12 deste roteiro). Isso porque, em projetos de manutenção de uma funcionalidade sem documentação, é necessário realizar o entendimento da funcionalidade para poder modificá-la e testá-la, ou seja, é necessário realizar a engenharia reversa da funcionalidade para executar os testes corretamente. Assim sendo, a redocumentação requisitada em projetos de melhoria requer um esforço menor do que em projetos de redocumentação, descritos na seção 4.12, onde é necessário remunerar todo o esforço de engenharia reversa e a atividade de documentação. Em projetos de manutenção, o fator de 15% está remunerando apenas a atividade de documentação.

Os percentuais de FI acima correspondem à contratação de todas as fases do processo de desenvolvimento de software. Caso alguma fase não seja contratada, deve-se aplicar ao FI um redutor que corresponde ao percentual da fase não contratada, conforme percentuais sugeridos na Tabela 7 ou na metodologia do órgão.

Os percentuais de multiplicação propostos são estimados, podendo ser reajustados conforme avaliação da base histórica dos serviços realizados no órgão ou entidade.

Este roteiro recomenda a supressão do PF_CONVERSÃO das fórmulas de contagem de pontos de função de projetos de melhoria quando for caracterizado um esforço relativamente maior dessa atividade, conforme descrito na seção 3.3.

Uma outra forma de dimensionar projetos de melhoria é usando a técnica EFP (*Enhancement Function Points*) publicada pela NESMA (*Netherlands Software Metrics Users Association*) no documento "*Function Point Analysis for Software Enhancement Guidelines*" [NESMA, 2009]. Essa técnica pode ser aplicada quando a contratante já possui uma base histórica de projetos concluídos com Contagem Detalhada de Pontos de Função e um processo de desenvolvimento implantado com documentação das aplicações a serem mantidas. O Anexo V apresenta um resumo da técnica EFP e a sua descrição completa pode ser obtida em [NESMA, 2009].

Seguem algumas considerações importantes para serem analisadas em projetos de melhoria.

Observação 1: Função Alterada

Uma função de dados (Arquivo Lógico Interno ou Arquivo de Interface Externa) é considerada alterada quando houver inclusão ou exclusão de Tipos de Dados (TD). De acordo com o glossário do CPM 4.3, um Tipo de Dados (DET – *Data Element Type*) é um atributo único, reconhecido pelo usuário e não repetido. Também é considerada alterada se algum tipo de dado sofrer mudança de tamanho (número de posições) ou tipo de campo (por exemplo: mudança de numérico ou alfanumérico), caso a mudança decorra de alteração de regra de negócio.

Uma função transacional (Entrada Externa, Consulta Externa e Saída Externa) é considerada alterada, quando a alteração contemplar:

- Mudança de tipos de dados;
- Mudança de arquivos referenciados;
- Mudança de lógica de processamento.

O CPM 4.3 define **lógica de processamento** como requisitos especificamente solicitados pelo usuário para completar um processo elementar. Esses requisitos devem incluir uma ou mais das seguintes ações:

- Validações são executadas;
- Fórmulas matemáticas e cálculos são executados;
- Valores equivalentes são convertidos;
- Dados são filtrados e selecionados através da utilização de critérios;
- Condições são analisadas para verificar quais são aplicáveis;
- Um ou mais ALIs são atualizados;
- Um ou mais ALIs ou AIEs são referenciados;
- Dados ou informações de controle são recuperados;
- Dados derivados são criados através da transformação de dados existentes, para criar dados adicionais;
- O comportamento do sistema é alterado;
- Preparar e apresentar informações para fora da fronteira;
- Receber dados ou informações de controle que entram pela fronteira da aplicação;
- Dados são reordenados.

Observação 2: Outros Tipos de Funções Alteradas

Este roteiro considera como função alterada qualquer mudança em funcionalidades da aplicação devido às mudanças de regras de negócio. Por exemplo, uma funcionalidade de cadastro envolvia a inclusão de um telefone do gerente. Devido a mudanças no processo de negócio, a funcionalidade deve sofrer uma manutenção para cadastrar dois telefones do gerente. Desta forma, o roteiro considera esta função como uma Entrada Externa alterada, PF_ALTERADO em um projeto de melhoria, mesmo que não existam mudanças de lógica de processamento, de tipos de dados ou de arquivos referenciados. Serão tratadas como manutenções adaptativas apenas as manutenções que implicarem exclusivamente em mudanças em requisitos não funcionais. Se uma mesma funcionalidade tiver mudanças em requisitos funcionais e não funcionais, esta deve ser contada apenas uma vez, como função alterada em um projeto de melhoria.

4.3 Projetos de Migração de Dados

Conforme mencionado na seção 3.3, este roteiro recomenda a supressão do PF_CONVERSÃO das fórmulas de contagem de pontos de função de projetos de desenvolvimento e de melhoria nos casos específicos onde for caracterizado um esforço relativamente maior dessa atividade, tais como, nos casos de migração de dados de banco de dados hierárquico para relacional, e no tratamento de funções complexas de migração de dados. Nesses casos, recomenda-se tratar esse serviço como projeto separado de migração de dados.

Os projetos de migração de dados devem ser contados como um novo projeto de desenvolvimento de um sistema, seguindo a fórmula abaixo:

PF_CONVERSÃO = PF_INCLUIDO

Um projeto de migração deve contemplar minimamente: os ALI mantidos pela migração, as Entradas Externas – considerando as cargas de dados nos ALI – e, caso seja solicitado pelo usuário, os relatórios gerenciais das cargas, que serão contados como Saídas Externas. Todas as contagens de PF devem ser realizadas com base nas funcionalidades requisitadas e recebidas pelo usuário.

4.4 Manutenção Corretiva

Mesmo com a execução de atividades de garantia da qualidade, pode-se identificar defeitos na aplicação entregue. A manutenção corretiva altera o software para correção de defeitos. Encontra-se nesta categoria, as demandas de correção de erros (*bugs*) em funcionalidades de sistemas em produção.

É importante destacar que as demandas de manutenção corretiva frequentemente precisam ser atendidas com urgência. Assim, o grau de criticidade do projeto poderá trazer impacto nas estimativas de custo e esforço. O padrão IEEE Std 1219 [IEEE,1998] define um tipo de manutenção corretiva, denominado de Manutenção Emergencial como “manutenção corretiva não programada executada para manter o sistema em estado operacional”.

Quando o sistema em produção tiver sido desenvolvido pela contratada, a manutenção corretiva será do tipo **Garantia** se estiver no período de cobertura e em conformidade com as demais condições de garantia previstas em contrato. Caso não exista cláusula contratual de garantia, deve ser considerada a garantia preconizada por lei (Código do Consumidor).

Quando o sistema estiver fora da garantia ou não tenha sido desenvolvido pela empresa contratada, deverá ser estimado e calculado o tamanho do projeto de manutenção corretiva. Nestes casos, a aferição do tamanho em pontos de função da funcionalidade ou das funcionalidades corrigidas deve considerar um **fator de impacto** (FI) sobre o PF_ALTERADO.

PF_CORRETIVA = FI x PF_ALTERADO

Fator de Impacto (FI):

- 50% quando estiver fora da garantia e a correção for feita pela mesma empresa que desenvolveu a funcionalidade.
- 75% quando estiver fora da garantia e a correção for feita por empresa diferente daquela que desenvolveu a funcionalidade.

As demandas de manutenção corretiva não contemplam atualização de documentação da funcionalidade corrigida, pois este roteiro considera que, normalmente, manutenção corretiva não se refere a erros de requisitos. Caso seja erro em requisitos, essa demanda deve ser tratada como projeto de melhoria (alteração de funcionalidade), descrito na seção 4.2. Porém, quando o erro for causado por documentação dúbia ou imprecisa (elaborada pela contratada) da funcionalidade corrigida, a manutenção corretiva poderá contemplar os ajustes na documentação, mesmo fora da garantia, mediante negociação entre as partes.

Caso seja demandada a redocumentação da funcionalidade corrigida, porque a documentação não existe ou está desatualizada, deve-se adicionar ao FI um fator de redocumentação de 15%, conforme descrito na seção 4.2.

Os percentuais de multiplicação são estimados, podendo ser reajustados conforme avaliação da base histórica dos serviços realizados no órgão ou entidade.

4.5 Mudança de Plataforma

São considerados nesta categoria, projetos que precisam ser migrados para outra plataforma. Por exemplo, um sistema legado em COBOL que necessita ser redesenvolvido em JAVA; o banco de dados de um sistema legado que precisa ser migrado para o DB2.

Recomenda-se enfaticamente a realização da análise de impacto das mudanças propostas, para efeito de determinação do percentual adequado para aplicação sobre o total de pontos de função das funcionalidades impactadas. Por exemplo, em uma análise de impacto pode ser identificado que não haverá mudanças no código-fonte ou em função transacional, sendo necessário apenas testar o sistema, então deve-se utilizar um percentual contemplando apenas a fase de testes. No caso do teste apontar a necessidade de atualizar alguma função transacional, não deve ser contado o esforço do teste, mas sim o esforço abordado nesta seção, conforme as fórmulas apresentadas nos tópicos seguintes.

As próximas subseções apresentam os tipos de projetos de mudança de plataforma. Os projetos de mudança de plataforma que se enquadram em mais de uma subseção, devem ser contados apenas uma vez, considerando o tipo de projeto com maior contagem de pontos de função. Os percentuais de multiplicação apresentados são estimados, podendo ser reajustados conforme avaliação da base histórica dos serviços realizados no órgão ou entidade.

4.5.1 Mudança de Plataforma - Linguagem de Programação

Nesta categoria encontram-se as demandas de redesenvolvimento de sistemas em outra linguagem de programação. Como os projetos legados, frequentemente, não possuem documentação, devem ser considerados como novos projetos de desenvolvimento. Assim, será utilizada a fórmula de projetos de desenvolvimento do CPM 4.3.

Observe que caso não exista mudança nas funções de dados, ou seja, o banco de dados da aplicação seja mantido, as funções de dados não devem ser contadas. No entanto, nesse caso, deve ser realizada a contagem das funções de dados a fim de compor a documentação da contagem final do projeto.

Outro ponto a ser observado são as fases contratadas. Caso o projeto já possua documentação de requisitos, a fase de requisitos não será contratada. Deve-se considerar apenas os percentuais das fases contratadas.

PF_REDESENVOLVIMENTO_LINGUAGEM = PF_INCLUÍDO + PF_CONVERSÃO

Este roteiro recomenda a supressão do PF_CONVERSÃO da fórmula de contagem de pontos de função de projetos de redesenvolvimento quando for caracterizado um esforço relativamente maior dessa atividade, conforme descrito na seção 3.3.

4.5.2 Mudança de Plataforma - Banco de Dados

Nesta categoria encontram-se as demandas de redesenvolvimento de sistemas para utilizar um outro sistema gerenciador de banco de dados.

Observe que caso não exista mudança nas funções de dados, ou seja, o banco de dados da aplicação seja mantido, então as funções de dados não devem ser contadas. No entanto, nesse caso, deve ser realizada a contagem das funções de dados a fim de compor a documentação da contagem final do projeto.

Em casos de mudança de banco hierárquico para relacional, em sistemas sem documentação, devido às mudanças envolvidas, deve-se considerar como um novo projeto de desenvolvimento, ou seja, as funções de dados e funções transacionais devem ser contadas. Assim, será utilizada a fórmula de projeto de desenvolvimento do CPM 4.3, conforme fórmula abaixo:

PF_REDESENVOLVIMENTO_BD_HIERÁRQUICO = PF_INCLUÍDO + PF_CONVERSÃO

Nos projetos de redesenvolvimento de banco de dados hierárquico para relacional, recomenda-se a supressão do PF_CONVERSÃO da fórmula acima, conforme descrito na seção 3.3.

Caso o projeto já possua documentação de requisitos, então a fase de requisitos não deve ser contratada. É importante destacar que isso se aplica a qualquer fase que não se deseja contratar. Deve-se considerar apenas os percentuais das fases contratadas.

Caso a demanda de redesenvolvimento seja de um sistema gerenciador de banco de dados relacional para outro relacional, deve ser utilizada a seguinte fórmula:

PF_REDESENVOLVIMENTO_BD_RELACIONAL = (PF_ALTERADO X 0,30) + PF_CONVERSÃO

O PF_ALTERADO deve considerar apenas as funcionalidades impactadas. As funcionalidades que possuem apenas demandas de testes, devem ser contadas usando o percentual da fase de testes (ver Tabela 7).

Nos projetos de redesenvolvimento de banco de dados relacional para outro relacional, recomenda-se tratar o PF_CONVERSÃO dentro do mesmo projeto.

Na mudança de banco relacional para relacional, geralmente a estrutura de dados não é alterada, desta forma não contamos as funções de dados.

4.6 Atualização de Versão

São consideradas nesta categoria, demandas para uma aplicação existente - ou parte de uma aplicação existente - executar em versões diferentes de *browsers* (ex: *Internet Explorer, Firefox*, *Chrome*, etc) ou de linguagens de programação (ex: versão mais atual do *JAVA*). Também são consideradas nesta categoria atualização de versão de banco de dados.

Nesta categoria foram observadas demandas de diferentes tipos de projetos, descritos nas próximas subseções. Os percentuais de multiplicação apresentados nessas subseções são estimados, podendo ser reajustados conforme avaliação da base histórica dos serviços realizados no órgão ou entidade.

Outro ponto a ser observado é a classificação, em alguns casos, dessas demandas como componente interno reusável (seção 4.15).

Recomenda-se enfaticamente a realização da análise de impacto das mudanças propostas para efeito de determinação do percentual adequado para aplicação sobre o total de pontos de função das funcionalidades impactadas. Por exemplo, em uma análise de impacto, pode ser identificado que não haverá mudanças no código-fonte ou em função transacional, sendo necessário somente testar o sistema, então deve-se utilizar um percentual contemplando apenas a fase de testes. No caso do teste apontar a necessidade de atualizar alguma função transacional, não deve ser contado o esforço do teste, mas sim o esforço abordado nesta seção, conforme as fórmulas apresentadas nas subseções seguintes.

4.6.1 Atualização de Versão – Linguagem de Programação

Nesta categoria encontram-se as demandas de atualização de versão de linguagem de programação de sistemas. As funções de dados não devem ser contadas. Estas demandas devem ser dimensionadas de acordo com a fórmula abaixo.

PF_ATUALIZAÇÃO_VERSÃO_LINGUAGEM = PF_ALTERADO x 0,30

O PF_ALTERADO deve considerar apenas as funcionalidades impactadas. As funcionalidades que possuem apenas demandas de testes, devem ser contadas usando o percentual da fase de testes (ver Tabela 7).

Cabe ressaltar que o redutor depende da linguagem da programação utilizada, considerando o grau de complexidade de implementação da mudança de versão no sistema em questão. Desta forma, recomenda-se fortemente a análise do percentual redutor da fórmula de contagem pelo órgão.

4.6.2 Atualização de Versão – Browser

Nesta categoria encontram-se as demandas de atualização de aplicações Web para executar em novas versões de um mesmo *browser* e para suportar a execução em mais de um *browser*. É importante destacar que este tipo de procedimento usualmente é realizado quando é necessário resolver algum problema de incompatibilidade. As funções de dados não devem ser contadas. Estas demandas devem ser dimensionadas de acordo com a fórmula abaixo.

PF_ATUALIZAÇÃO_VERSÃO_BROWSER = PF_ALTERADO x 0,30

O PF_ALTERADO deve considerar apenas as funcionalidades impactadas. As funcionalidades que possuem apenas demandas de testes, devem ser contadas usando o percentual da fase de testes (ver Tabela 7).

Essas atualizações podem implicar em manutenções em componentes específicos da plataforma utilizada. Nesse caso, a demanda deve ser contada como componente interno reusável, descrita na seção 4.15 deste roteiro.

Recomenda-se enfaticamente a realização da análise de impacto das mudanças propostas para efeito de determinação do percentual adequado. Por exemplo, para sistemas que já atendem ao padrão W3C (*World Wide Web Consortium*) o esforço é menor, podendo usar, neste caso, um percentual diferente do citado acima. É importante ressaltar que os sistemas Web devem seguir o padrão W3C, como recomendado na e-Ping. Caso seja necessário fazer a adequação do sistema para atendimento ao padrão W3C, pode-se usar a fórmula acima.

4.6.3 Atualização de Versão – Banco de Dados

Nesta categoria encontram-se as demandas de atualização de versão do sistema gerenciador de banco de dados. As funções de dados não devem ser contadas. Estas demandas devem ser dimensionadas de acordo com a fórmula abaixo.

PF_ ATUALIZAÇÃO_VERSÃO_BD = PF_ALTERADO x 0,30

O PF_ALTERADO deve considerar apenas as funcionalidades impactadas. As funcionalidades que possuem apenas demandas de testes, devem ser contadas usando o percentual da fase de testes (ver Tabela 7).

4.7 Manutenção em *Interface*

A manutenção em *interface*, denominada na literatura de manutenção cosmética, é associada às demandas de alterações de *interface*, por exemplo: fonte de letra, cores de telas, logotipos, mudança de botões na tela, mudança de posição de campos ou texto na tela. Também se enquadram nessa categoria as seguintes manutenções:

- Mudanças de texto em mensagens de erro, validação, aviso, alerta, confirmação de cadastro ou conclusão de processamento;
- Mudança em texto estático de e-mail enviado para o usuário em uma funcionalidade de cadastro. A demanda deve ser contada como manutenção em interface na funcionalidade de cadastro;
- Alteração de título de um relatório;
- Alteração de *labels* de uma tela de consulta.

Nestes casos, a aferição do tamanho em pontos de função das funções transacionais impactadas será realizada com a aplicação de um fator de redução de modo a considerar 20% da contagem de uma função transacional de mais baixa complexidade (3 PF), ou seja 0,6 PF, independentemente da complexidade da funcionalidade alterada. Neste tipo de manutenção não são contadas funções de dados.

PF_INTERFACE = 0,6 PF x QUANTIDADE DE FUNÇÕES TRANSACIONAIS IMPACTADAS

Está contemplada a atualização da documentação das funcionalidades da aplicação impactadas pela manutenção nas demandas desta categoria. Assim, a documentação (documento de requisitos, documento de interface, protótipo, entre outros) das funcionalidades alteradas deve ser atualizada. Caso não exista documentação para as funcionalidades alteradas, não será contemplada a redocumentação das funcionalidades da aplicação impactadas pela manutenção nas demandas desta categoria.

Observação 1 – Help: As demandas de projetos de desenvolvimento de sistemas ou de manutenção de funcionalidades contemplam o desenvolvimento ou atualização do help da funcionalidade em questão, sendo tratada como uma atividade de documentação no processo de software. No caso de demandas específicas de desenvolvimento ou atualização de help estático de funcionalidades, estas podem ser enquadradas nesta seção e poderá ser usado um valor de multiplicação inferior a 0,6 PF conforme análise de impacto das mudanças propostas. Em caso de requisitos de usuário para o desenvolvimento de funcionalidades de manutenção de help, deve-se contar a função de dados de help e as funcionalidades de manutenção de help (por exemplo: incluir help de tela, consultar help de campo) de acordo com o CPM 4.3.

O percentual de multiplicação é estimado, podendo ser reajustado conforme avaliação da base histórica dos serviços realizados no órgão ou entidade.

4.8 Adaptação em Funcionalidades sem Alteração de Requisitos Funcionais

São consideradas nesta categoria as demandas de manutenção adaptativa associadas a solicitações que envolvem aspectos não funcionais, sem alteração em requisitos funcionais. Seguem alguns exemplos:

– Aumentar a quantidade de linhas por página em um relatório;

– Colocar paginação em um relatório;

– Limitar a quantidade de linhas por página em uma consulta existente;

– Permitir exclusões múltiplas em uma funcionalidade que antes só possibilitava a exclusão de um item;

– Adaptação de uma funcionalidade para possibilitar a chamada por um *webservice* ou para outro tipo de integração com outros sistemas;

– Replicação de funcionalidade: chamar uma consulta existente em outra tela da aplicação;

– Alteração na aplicação para adaptação às alterações realizadas na interface com rotinas de integração com outros softwares, por exemplo, alteração em sub-rotinas chamadas por este software;

– Modificar o servidor a ser acessado em uma funcionalidade de *download* de arquivo;

– Adequar mensagem do sistema que em algumas telas apresenta “Usuário Não está Habilitado a ver esta Página”, para que passe a enviar uma mensagem mais adequada ao fato do usuário não possuir mais uma sessão ativa e ainda estar navegando no sistema. A demanda deve ser contada como manutenção adaptativa considerando as funcionalidades impactadas. Observe que trata-se de mudança em validação com regra de negócio não funcional.

Nestes casos, a aferição do tamanho em pontos de função da funcionalidade ou das funcionalidades que sofreram impacto deve considerar um fator de impacto (FI) sobre o PF_ALTERADO, seguindo os conceitos do CPM 4.3, apresentados na seção 4.2.

PF_ADAPTATIVA = FI x PF_ALTERADO

FI (Fator de Impacto) pode variar conforme condições abaixo:

- **FI = 50%** para **funcionalidade de sistema desenvolvida ou mantida** por meio de um projeto de melhoria pela empresa contratada.
- **FI = 75%** para **funcionalidade de sistema não desenvolvida ou mantida** por meio de um projeto de melhoria pela empresa contratada.

Deve-se destacar que além da adequação das funcionalidades em questão, a documentação do projeto de manutenção adaptativa deve ser realizada. Além disso, caso exista a documentação das funcionalidades impactadas, estas deverão ser atualizadas, caso contrário, se for demandada a redocumentação dessas funcionalidades, deve-se adicionar ao FI um fator de redocumentação de 15%, conforme descrito na seção 4.2.

Os percentuais de multiplicação são estimados, podendo ser reajustados conforme avaliação da base histórica dos serviços realizados no órgão ou entidade.

4.9 Apuração Especial

São funcionalidades executadas apenas uma vez para: corrigir problemas de dados incorretos na base de dados das aplicações ou atualizar dados em bases de dados de aplicações, detalhados na subseção 4.9.1; gerar um relatório específico ou arquivo para o usuário por meio de recuperação de informações nas bases da aplicação, detalhados na subseção 4.9.2. A subseção 4.9.3 considera os casos de reexecução de uma apuração especial.

Caso a apuração seja de correção de dados devido a erros de funcionalidades de aplicações desenvolvidas pela contratada, observar as cláusulas contratuais com relação a garantias e prazos de correção.

Recomenda-se fortemente ao órgão contratante sempre solicitar formalmente para a empresa contratada o armazenamento do *script* para permitir posterior reexecução.

Cabe ressaltar que o órgão deve avaliar a complexidade das demandas típicas de apuração especial, podendo utilizar um percentual redutor nas fórmulas descritas nas subseções seguintes. Por exemplo, o redutor percentual pode ser aplicado em função da complexidade das demandas, documentação demandada e/ou do processo de desenvolvimento utilizado.

4.9.1 Apuração Especial – Base de Dados

Este tipo de apuração especial é um projeto que inclui a geração de procedimentos para atualização da base de dados. Deve-se destacar que estas funções são executadas apenas uma vez, não fazendo parte da aplicação, visando a correção de dados incorretos na base de dados da aplicação ou atualização em função de modificação da estrutura de dados, por exemplo inclusão de valor “sim” ou “não” no campo “indicador de matriz” referente ao CNPJ. Normalmente, nesse tipo de atualização são afetados múltiplos registros. Nestes casos, considera-se a contagem de pontos de função das funcionalidades desenvolvidas. Geralmente, estas funcionalidades são classificadas como Entradas Externas. Nesse caso, como artefato de homologação da demanda, deve ser gerado um relatório para validação do usuário.

É importante ressaltar que as funções de dados associadas aos dados atualizados **não** devem ser contadas, considerando que não há mudanças nas estruturas dos Arquivos Lógicos Internos.

Foram identificados três tipos de Apuração Especial - Base de Dados, cujas fórmulas de cálculo são apresentadas a seguir:

a) Atualização de Dados sem Consulta Prévia

PF_APURAÇÃO_BD = PF_INCLUÍDO

b) Consulta Prévia sem Atualização

Em alguns casos de Apuração Especial – Base de Dados, o usuário solicita uma consulta prévia das informações. Deve-se ressaltar que essa consulta deve ser realizada antes da construção da funcionalidade, não se trata de homologação. A consulta prévia não é definida pela empresa contratada, obrigatoriamente essa deve ser solicitada pelo órgão contratante para a avaliação da viabilidade de implementar a Apuração Especial - Base de Dados. De fato, é uma prática interessante para evitar informações errôneas na base de produção dos sistemas. Esta consulta prévia, classificada como Consulta Externa ou Saída Externa deve ser dimensionada considerando-se o tamanho da funcionalidade em questão, conforme a fórmula abaixo:

PF _CONSULTA_PRÉVIA = PF_INCLUÍDO

c) Atualização de Dados com Consulta Prévia

Caso a Apuração Especial - Base de Dados seja solicitada após uma demanda de consulta prévia, deve-se aplicar um fator de 60% na fórmula de contagem da Apuração Especial - Base de Dados, seguindo a fórmula abaixo.

PF_APURAÇÃO_BD_PÓS_CONSULTA_PRÉVIA = PF_INCLUÍDO x 0,60

4.9.2 Apuração Especial – Geração de Relatórios

Este tipo de apuração especial é um projeto que inclui a geração de relatórios em uma ou mais mídias para o usuário. Em alguns casos, são solicitadas extrações de dados e envio dos dados para outros sistemas. Caso, neste envio de dados, sejam requisitadas atualizações no sistema de origem, então essas funções transacionais são Saídas Externas, devido à atualização do Arquivo Lógico Interno.

Deve-se destacar que essas funções são executadas apenas uma vez, não fazendo parte da aplicação. Nestes casos, considera-se contagem de pontos de função das funcionalidades desenvolvidas. Frequentemente, estas funcionalidades são classificadas como Saídas Externas. Também podem ser classificadas como Consultas Externas, caso não possuam cálculos ou criação de dados derivados.

É importante ressaltar que as funções de dados associadas aos dados atualizados **não** devem ser contadas, considerando que não há mudanças nas estruturas dos Arquivos Lógicos.

PF_APURAÇÃO_RELATÓRIOS = PF_INCLUÍDO

4.9.3 Apuração Especial – Reexecução

Em alguns casos, a empresa contratante pode ter interesse em executar uma apuração especial mais de uma vez. Nestes casos, ela deve solicitar formalmente à contratada o armazenamento do *script* executado. Desta forma, se for solicitada a reexecução de uma apuração especial, esta deve ser dimensionada com a aplicação de um fator redutor de 10% na contagem de pontos de função da apuração especial em questão, da seguinte maneira:

PF_REEXECUÇÃO_APURAÇÃO = PF_NÃO_AJUSTADO x 0,10

O percentual de multiplicação proposto no item acima é estimado, podendo ser reajustado conforme avaliação da base histórica dos serviços realizados no órgão ou entidade.

4.10 Atualização de Dados

Em alguns casos, as demandas de correção de problemas em base de dados estão associadas a atualizações manuais (de forma interativa), diretamente no banco de dados em um único registro, e que não envolvem cálculos ou procedimentos complexos. São exemplos desse tipo de demanda, a atualização do valor de um campo de uma tabela cadastrado erroneamente ou a exclusão de um registro de uma tabela.

Nestes casos, a aferição do tamanho em Pontos de Função deve considerar 10% do PF de uma Entrada Externa e os Tipos de Dados da Entrada Externa são todos os TD considerados na funcionalidade – campos atualizados e campos utilizados para a seleção do registro.

PF_ATUALIZAÇÃO_BD = PF_INCLUÍDO x 0,10

Deve-se ressaltar que neste tipo de demanda não há gestão de configuração (armazenamento de *script*, versionamento, etc) das atualizações. Caso a contratante identifique a necessidade de realização de gestão de configuração das atualizações no banco de dados, então a demanda será classificada como Apuração Especial - Base de Dados (subseção 4.9.1).

O percentual de multiplicação proposto acima é estimado, podendo ser reajustado conforme avaliação da base histórica dos serviços realizados no órgão ou entidade.

4.11 Desenvolvimento, Manutenção e Publicação de Páginas Estáticas de *Intranet, Internet* ou Portal

Nesta seção são tratados desenvolvimentos e manutenções específicas em páginas estáticas de portais, *intranets* ou w*ebsites*. As demandas desta seção abrangem a publicação de páginas *Web* com conteúdo estático. Por exemplo: criação de página HTML, atualização de menu estático, atualização de texto ou *banner* estáticos em páginas HTML existentes.

Caso o desenvolvimento de páginas estáticas esteja contido em um projeto de desenvolvimento, então elas serão contabilizadas no projeto de desenvolvimento e não devem ser mensuradas em separado. Ou seja, esta seção 4.11 se aplica quando ocorrer a demanda exclusivamente para o desenvolvimento ou manutenção de páginas estáticas.

Estas demandas são consideradas como desenvolvimento de consultas. Nestes casos, considera-se 20% dos pontos de função das consultas desenvolvidas. Cada página é contada como uma consulta. As consultas são consideradas consultas externas simples (3 PF). Ou seja, 0,6 PF por cada página desenvolvida ou mantida, de acordo com a fórmula abaixo:

PF_PUBLICAÇÃO = 0,6 PF x Quantidade de Páginas Alteradas ou Incluídas

As demandas de criação de logomarcas ou identidade visual, além de outras demandas de criação de arte, associadas à área de Comunicação Social, não são enquadradas nessa categoria. Tais demandas não se referem a contratos de prestação de serviços de desenvolvimento e manutenção de sistemas, portanto não são consideradas neste roteiro.

É recomendada a construção de portais com ferramentas que apoiem a construção de conteúdo pelo usuário, os chamados Gerenciadores de Conteúdo, de modo a minimizar as demandas de criação de páginas estáticas.

O percentual de multiplicação proposto acima é estimado, podendo ser reajustado conforme avaliação da base histórica dos serviços realizados no órgão ou entidade.

4.12 Manutenção de Documentação de Sistemas Legados

Nesta seção são tratadas demandas de documentação ou atualização de documentação de sistemas legados. Observe que o desenvolvedor deve realizar uma engenharia reversa da aplicação para gerar a documentação. Para este tipo de projeto foi definido o fator de impacto de 25% dos pontos de função da aplicação em questão, considerando a fase de requisitos e a geração de artefatos associados a requisitos, conforme a fórmula abaixo.

PF_DOCUMENTAÇÃO = PF_NÃO_AJUSTADO x 0,25

Caso a demanda seja a geração de artefatos de documentação de outras fases do processo de desenvolvimento, deve-se considerar um outro fator de impacto, considerando as fases do ciclo de vida e os demais artefatos a serem gerados. As premissas utilizadas devem ser definidas nas cláusulas contratuais e documentadas no documento de estimativas do projeto.

O percentual de multiplicação proposto acima é estimado, podendo ser reajustado conforme avaliação da base histórica dos serviços realizados no órgão ou entidade.

4.13 Verificação de Erros

As verificações de erro ou análise e solução de problemas são as demandas referentes a todo comportamento anormal ou indevido apontado pelo cliente nos sistemas aplicativos. Neste caso, a equipe de desenvolvimento da contratada se mobilizará para encontrar as causas do problema ocorrido. Se for constatado algum erro de sistema, a demanda será atendida como manutenção corretiva (seção 4.4).

Entretanto, uma vez não constatado o problema apontado pelo cliente ou o mesmo for decorrente de regras de negócio implementadas ou utilização incorreta das funcionalidades, será realizada a aferição do tamanho em pontos de função das funcionalidades verificadas que o cliente reportou erro. Caso não exista documentação de testes disponível dessas funcionalidades verificadas, será considerado 20% do tamanho funcional dessas funcionalidades com solicitação de análise pelo órgão contratante, segundo a fórmula abaixo:

PF_VERIFICAÇÃO = PF_Funcionalidade_Reportada_Com_Erro x 0,20

Caso exista documentação de testes das funcionalidades verificadas, então será considerado 15% (mesmo percentual da fase de Testes, conforme Tabela 7) do tamanho funcional das funcionalidades analisadas, segundo a fórmula abaixo:

PF_VERIFICAÇÃO = PF_Funcionalidade_Reportada_Com_Erro x 0,15

É importante ressaltar que a demanda de verificação de erros deve ser associada a uma funcionalidade específica. Os casos de sistema fora do ar por conta de problemas de rede ou banco de dados devem ser tratados como serviços de suporte e não serviços de desenvolvimento e manutenção de sistemas. Esses serviços de suporte não fazem parte do escopo desse roteiro de métricas, não se aplicando verificação de erros nestes casos.

O percentual de multiplicação proposto acima é estimado, podendo ser reajustado conforme avaliação da base histórica dos serviços realizados no órgão ou entidade.

4.14 Pontos de Função de Teste

Muitas vezes, em projetos de manutenção, o conjunto de funções transacionais a serem testadas é maior do que a quantidade de funções a serem implementadas, isto é, além das funcionalidades que são afetadas diretamente pelo projeto de manutenção, outras precisam ser testadas [NESMA, 2009]. O tamanho das funções a serem apenas testadas deve ser aferido em Pontos de Função de Teste (PFT). Não considerar as funcionalidades incluídas, alteradas ou excluídas do projeto de manutenção na contagem de Pontos de Função de Teste.

A contagem de PFT será o somatório dos tamanhos em pontos de função das funções transacionais envolvidas no teste:

PFT = Somatório dos Tamanhos das Funções Transacionais Testadas

A conversão do PFT em ponto de função deve ser feita de acordo com a fórmula abaixo:

PF_TESTES = PFT x 0,15

É importante ressaltar que no caso de uma função ser testada várias vezes, com cenários diferentes, a função só pode ser contada uma vez. Outra observação é que as funções testadas, consideradas no PFT, devem ser documentadas pela contratada considerando-se a documentação de testes definida no processo de desenvolvimento da contratante. Observe que estas funções farão parte do escopo do projeto de manutenção.

4.15 Componente Interno Reusável

Em alguns casos são demandadas manutenções em componentes, que implementam regras de negócio, específicos de uma aplicação e estes são reusados por várias funcionalidades da aplicação. Por exemplo, uma mudança em uma rotina de validação de um CPF usada em várias funcionalidades de cadastro. Se considerarmos o método de contagem de projetos de melhoria do CPM, seriam contadas todas as funcionalidades impactadas por essa mudança.

No entanto, este roteiro propõe que o componente, o qual deverá ser testado, seja considerado como um processo elementar independente e sua alteração seja contada aplicando-se um fator de impacto (FI) sobre o PF_ALTERADO, seguindo os conceitos do CPM 4.3, apresentados na seção 4.2 - Projeto de Melhoria. Além disso, as funcionalidades da aplicação que necessitem de teste devem ser requisitadas pela contratante e dimensionadas por meio da métrica Pontos de Função de Teste proposta na seção 4.14.

PF_COMPONENTE = FI x PF_ALTERADO

Exemplo de manutenção de componentes:

• Mudança em tópico de um menu de um sistema em PHP que aparece em todas as telas da aplicação. A contagem pode ser realizada considerando o componente “Apresentar Menu”.

• Além disso, existem casos onde são realizadas manutenções de valores de elementos internos de configuração que afetam o comportamento ou a apresentação do sistema de forma geral, tais como páginas de estilos (arquivos CSS de sistemas Web), arquivos com mensagens de erro, arquivos de configuração de sistema e arquivos de internacionalização. Nestes casos, a aferição do tamanho em pontos de função será realizada com a aplicação de um fator de redução de modo a considerar 20% da contagem de uma função transacional de mais baixa complexidade (3 PF), ou seja 0,6 PF. Assim sendo, deve ser utilizada a seguinte fórmula de cálculo:

PF_COMPONENTE_ARQUIVO = 0,6 PF x QTD_ARQUIVOS_ALTERADOS

5. Orientações Complementares para Contagem

Este capítulo tem o propósito de apresentar diretrizes complementares ao Manual de Práticas de Contagem do IFPUG para contagens de pontos de função e reforçar pontos sensíveis nas contratações atuais que podem impactar significativamente o resultado de uma contagem em caso de falhas.

As diretrizes para contagens de pontos de função envolvendo Múltiplas Mídias têm como base o artigo “Considerations for Counting with Multiple Media” Release 1.1 publicado pelo IFPUG [IFPUG, 2010a].

5.1 Contagem de Pontos de Função com Múltiplas Mídias

A contagem de PF de funcionalidades entregues em mais de uma mídia, na aplicação das regras de contagem de pontos de função definidas no CPM, tem levado a duas abordagens alternativas, a saber: *single instance* e *multiple instance*.

É importante enfatizar que o IFPUG reconhece ambas abordagens, *single instance* e *multiple instance,* para a aplicação das regras definidas no CPM. A determinação da contagem de PF seguindo a abordagem *multiple instance* ou *single instance* depende da avaliação do Escritório de Métricas da instituição.

As estimativas e contagens de PF abordadas neste documento são baseadas em *multiple instance*, com exceção dos casos de consultas em .pdf, .doc, .xls e consultas idênticas em tela e papel, que serão consideradas uma única funcionalidade.

A seguir são descritos os termos comuns definidos pelo IFPUG [IFPUG, 2010a]:

- **Canal:** também refere-se a mídia. Múltiplos canais é sinônimo de múltiplas mídias.
- **Mídia:** descreve a maneira como os dados ou informações se movimentam para dentro e para fora de uma fronteira de aplicação, por exemplo, apresentação de dados em tela, impressora, arquivo, voz. Este termo é utilizado para incluir, dentre outros, diferentes plataformas técnicas e formatos de arquivos como diferentes mídias.
- **Múltiplas Mídias**: quando a mesma funcionalidade é entregue em mais de uma mídia. Frequentemente, apenas uma mídia é requisitada para um usuário específico em um determinado momento, por exemplo consulta de extrato bancário via *Internet* como oposto a consulta de extrato bancário via terminal do banco.
- **Multi-Mídia:** quando mais de uma mídia é necessária para entregar a funcionalidade , por exemplo, uma nova notícia publicada na *Internet* que é apresentada em vídeo e texto. Observe que a notícia completa só é apresentada para o usuário se ele ler o texto e assistir o vídeo.
- **Abordagem *Single Instance***: esta abordagem não reconhece que a mídia utilizada na entrega da função transacional é uma característica de diferenciação na identificação da unicidade da função transacional. Se duas funções entregam a mesma funcionalidade usando mídias diferentes, elas são consideradas a mesma funcionalidade em uma contagem de pontos de função.
- **Abordagem *Multiple Instance***: esta abordagem especifica que o tamanho funcional é obtido no contexto do objetivo da contagem, permitindo uma função de

negócio ser reconhecida no contexto das mídias que são requisitadas para que a funcionalidade seja entregue. A abordagem *multiple instance* reconhece que a mídia para entrega constitui uma característica de diferenciação na identificação da unicidade da função transacional.

Os cenários descritos nas seções seguintes não representam uma lista completa de situações de múltiplas mídias. O entendimento dos exemplos a seguir facilitará o entendimento de outros cenários envolvendo múltiplas mídias.

Este roteiro deve ser atualizado considerando a publicação de novas diretrizes do IFPUG e novos cenários que emergirão nas contagens de PF de projetos dos órgãos e entidades do SISP.

5.1.1 Cenário 1: Mesmos dados apresentados em tela e impressos

Neste cenário, uma aplicação apresenta uma informação em uma consulta em tela. A mesma informação pode ser impressa, caso requisitado pelo usuário, na tela em questão.

Nesses casos, sugere-se a abordagem *single instance*, considerando que dados idênticos sendo apresentados em tela e em relatório impresso devem ser contados como uma única função. Caso as lógicas de processamento da consulta em tela e do relatório em papel sejam distintas, o processo elementar não é único e, portanto, a funcionalidade será contada duas vezes (*multiple instance*). Neste caso, duas funções são contadas: apresentação de dados em tela e apresentação de dados impressos.

5.1.2 Cenário 2: Mesmos dados de saída como dados em arquivo e relatório impresso

Uma aplicação grava dados em um arquivo de saída e imprime um relatório com informações idênticas às gravadas no arquivo.

Nesses casos, sugere-se a utilização da abordagem *single instance* considerando que os dados impressos e os dados apresentados no arquivo de saída sejam idênticos e que a ferramenta de desenvolvimento apoie a geração dessas múltiplas saídas. Assim, apenas uma funcionalidade será incluída na contagem de pontos de função. Caso as lógicas de processamento da geração do arquivo de saída e do relatório em papel sejam distintas, o processo elementar não é único e, portanto, a funcionalidade será contada duas vezes. Além disso, se a geração das múltiplas saídas não seguirem o padrão da ferramenta de desenvolvimento e tiverem que ser customizadas para o cliente, então será utilizada a abordagem *multiple instance*.

5.1.3 Cenário 3: Mesmos dados de entrada *batch* e *on-line*

Uma informação pode ser carregada na aplicação por meio de dois métodos: arquivo *batch* e entrada *on-line*. O processamento do arquivo *batch* executa validações durante o processamento, da mesma forma que o processamento da entrada *on-line* também executa validações das informações. Neste caso, sugere-se a utilização da abordagem *multiple instance,* que conta duas funcionalidades: a entrada de dados *batch* e a entrada de dados *on-line*. Geralmente, a lógica de processamento utilizada nas validações em modo *batch* é diferente da lógica de processamento das validações nas entradas de dados *on-line*.

5.1.4 Cenário 4: Múltiplos canais de entrega da mesma funcionalidade

Uma funcionalidade deve ser disponibilizada em múltiplos canais, por exemplo: consulta de dados em página Web e consulta de dados no telefone celular. Neste caso, sugere-se a abordagem *multiple instance,* que conta duas funcionalidades: consulta de dados na Web e consulta de dados via celular.

Considera-se que a funcionalidade é desenvolvida duas vezes, uma para cada canal de saída. Algumas vezes, são até projetos de desenvolvimento distintos, um projeto relativo ao sistema Web e outro para o sistema via celular. Lembrando que caso o projeto seja claro o suficiente para dizer que o desenvolvimento é o mesmo, poderá ser utilizada a abordagem *single instance.*

5.1.5 Cenário 5: Relatório em múltiplos formatos

Um relatório deve ser entregue em diferentes formatos, por exemplo: um arquivo *html* e um arquivo com valores separados por vírgula (.csv).

Nestes casos, conforme sugerido na abordagem *multiple instance*, considera-se a ferramenta utilizada na geração dos relatórios. Se a equipe de desenvolvimento precisar desenvolver o relatório nos dois formatos na ferramenta em questão, serão contadas duas funcionalidades. No entanto, se a ferramenta de desenvolvimento suportar um gerador de relatórios que o usuário visualize o relatório em tela e o gerador permita ao usuário imprimir o relatório, salvar em *html* ou salvar no formato de valores separados por vírgula, então se contará apenas uma vez, observando que a funcionalidade será da ferramenta e não da aplicação.

5.2 Log, Trilha de Auditoria e Histórico

O objetivo dessa sessão é descrever orientações sucintas a respeito de contagem de log, trilha de auditoria e histórico.

5.2.1 Log

Conceituamos o termo “Log” como o registro de procedimentos ou ações realizados pela aplicação, em determinado período de tempo, com o objetivo de apoiar a auditoria do ambiente tecnológico e a identificação das causas raízes de falhas em sistemas. Diante desse conceito, definimos que o Log não deve ser mensurado com Pontos de Função, já que ele não armazena informações negociais reconhecidas pelo usuário da aplicação.

5.2.2 Trilha de Auditoria

Conceituamos “Trilha de Auditoria” como a funcionalidade que tem o objetivo de armazenar informações referentes às ações realizadas pelos usuários da aplicação no passado, de modo que seja possível apurar quais foram as ações executadas quando da utilização do sistema.

Para isso, devem existir no mínimo as informações para identificar quem realizou a ação, quando e o que foi realizado, além de outras informações que o usuário da aplicação defina como necessárias.

A trilha de auditoria deve ser solicitada pelo usuário da aplicação e, para a contagem, deve existir funcionalidade de consulta a tais dados.

Caso a trilha de auditoria faça parte da política corporativa de segurança da informação adotada pelo contratante para todos os sistemas do órgão, ela deve ser considerada como um requisito não funcional e, portanto, não será mensurável em ponto de função.

Diante do exposto, a principal diferença entre o Log e a Trilha de Auditoria é:

- **Log**: apoia a coleta de informações no âmbito tecnológico, ou seja, em problemas decorrentes da arquitetura tecnológica que precisam ser investigados, por meio da análise do conjunto de procedimentos executadas pela aplicação, como exemplo a baixa performance no sistema, travamentos e outros comportamentos inesperados.
- **Trilha de Auditoria**: apoia a auditoria para os dados de negócio, armazenando informações das ações realizadas pelo usuário na aplicação.

5.2.2 Histórico

Conceituamos “Histórico” como um registro de estados com informações anteriores de um registro em determinado momento. O usuário poderá consultar a evolução dessas informações em uma linha do tempo e sua existência é justificada pelo negócio. Assim, para fazer parte do tamanho funcional, deve ser solicitado pelo gestor e deverá existir funcionalidade de consulta a tais dados.

A função de consulta aos dados de um histórico deverá ser contada de acordo com as regras de contagem das funções transacionais do CPM.

Não devem ser consideradas na contagem funções de transação separadas para incluir, alterar e excluir as informações históricas, pois o armazenamento dessas informações é parte integrante das mesmas funcionalidades que processam os dados de negócio. Apenas quando o histórico for mantido de forma independente do registro principal, por exemplo no caso do ALI principal ter sido excluído, o histórico se torna um ALI independente e não um registro lógico do ALI relacionado.

5.3 Identificação de Processo Elementar

Um Processo Elementar é a menor unidade de atividade que é significativa para o(s) usuário(s). O Processo Elementar deve ser auto contido e deixar o negócio da aplicação que está sendo contada em um estado consistente.

Um processo elementar com múltiplas formas de processamento lógico não deve ser dividido em múltiplos processos elementares. Se um processo elementar é subdividido inapropriadamente, o mesmo não reúne os critérios de um processo elementar.

Ressalta-se a importância do atendimento a todos os critérios listados no Manual de Práticas de Contagem do IFPUG e da observação dos seus exemplos para a correta identificação de um processo elementar.

6. Considerações Especiais para Planejamento e Acompanhamento de Projetos

Este capítulo tem como propósito apresentar diretrizes para o planejamento e acompanhamento de projetos com o auxílio da métrica Ponto de Função e de técnicas relacionadas. Com base nesta finalidade é descrito um processo de estimativas de projetos de software aderente à área de processo de Planejamento de Projeto do CMMI (*Capability Maturity Model Integration*). Nesse contexto, são apresentados: o método Contagem Estimativa de Pontos de Função (CEPF) para estimar o tamanho dos projetos de software em PF, o modelo simplificado de estimativas para estimar o esforço dos projetos em homem-hora (HH) e a fórmula de Capers Jones para estimar os prazos dos

projetos. Também são apresentadas recomendações para o gerenciamento de: mudanças de requisitos, projetos cancelados e progresso de projetos, e considerações sobre redução de cronograma e fator de criticidade de solicitação de serviços.

6.1 Diretrizes para Planejamento: Estimativas de Projetos de Software

Esta seção define métodos para estimativas de projetos de software. A Figura 2 ilustra um processo de estimativas de projetos de software, descrito nos parágrafos seguintes.

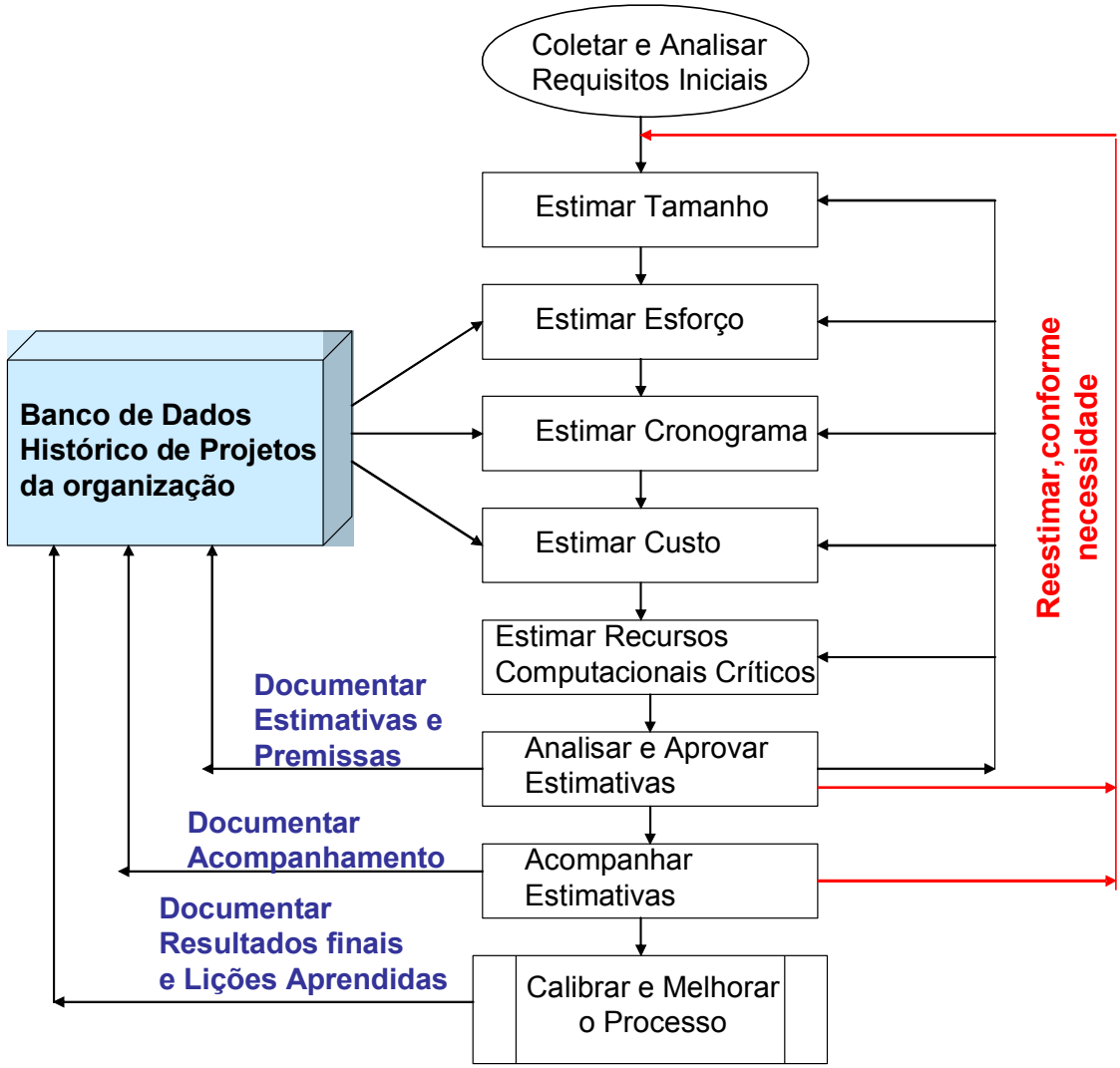

Figura 2: Processo de Estimativas de Projetos de Software [Hazan, 2008]

O principal insumo (artefato de entrada) para um processo de estimativas é o documento de requisitos. Como as estimativas devem ser realizadas no início do processo de desenvolvimento de software, então o artefato a ser utilizado é um documento inicial de requisitos, por exemplo, o documento de visão ou formalização simples de requisitos. O estimador deve analisar os requisitos para garantir a qualidade e então estimar o tamanho do projeto de software. O próximo passo é a derivação das estimativas de esforço, prazo (cronograma), custo (orçamento) com base na estimativa de tamanho e nos dados históricos de projetos concluídos da organização, assim como o

estabelecimento da estimativa de recursos computacionais críticos e dos recursos da equipe a ser alocada ao projeto. Neste ponto, as principais estimativas foram geradas e precisam ser documentadas. As premissas e suposições utilizadas na geração das estimativas, dentre as quais: complexidade do projeto, plataforma de desenvolvimento, tipo do projeto, percentual de evolução de requisitos, também devem ser documentadas [Hazan, 2008].

A realização das estimativas por um analista de métricas que não atue na equipe do projeto, constitui uma prática recomendada. O analista de métricas deve analisar também a consistência da documentação utilizada na estimativa. No decorrer do processo de desenvolvimento, as estimativas devem ser acompanhadas conforme o refinamento dos requisitos. O projeto deve ser reestimado após a fase de requisitos, quando for gerada a especificação de casos de uso, e sempre que ocorrerem mudanças significativas nos requisitos funcionais ou não funcionais. Quando o projeto é concluído, deve-se aferir e documentar o tamanho, prazo, custo, esforço e recursos realizados, assim como outros atributos relevantes do projeto, visando a coleta de dados para a melhoria do processo de estimativas. As lições aprendidas também devem ser documentadas [Hazan, 2008].

Portanto, para os contratos de projetos de software, baseados na métrica Ponto de Função, as estimativas devem ser realizadas em, no mínimo, três marcos do processo de desenvolvimento de software, a saber:

- **Estimativa inicial**: realizada após o fechamento do escopo do projeto. Geralmente é baseada em um documento inicial de requisitos como, por exemplo, o documento de visão. Constitui uma boa prática a previsão de evolução de requisitos, especialmente em projetos de desenvolvimento de médio ou grande porte. Nessa etapa é importante destacar os seguintes conceitos na área de estimativas:

  - Uma **Estimativa** é obtida por meio de uma atividade técnica, utilizando métodos de estimativas. Não deve sofrer interferências políticas;

  - A **Meta** é um desejo, em função de necessidades de negócio, estabelecida politicamente;

  - Um **Compromisso** é um acordo da gerência com as equipes técnicas para alcançar uma meta [Parthasarathy, 2007].

  Em um cenário ideal, os resultados da estimativa atendem às metas de negócio. Quando este cenário não é real, é fundamental a redução de escopo do projeto, de modo que a meta se adapte aos resultados da estimativa.

- **Contagem de Pontos de Função de Referência**: realizada após o aceite dos requisitos. Geralmente, leva em consideração a especificação dos casos de uso e regras de negócio da aplicação. Pode ser aplicada a contagem estimada ou a detalhada.

- **Contagem de Pontos de Função Final**: realizada após a homologação da aplicação. Esta contagem considera as funcionalidades efetivamente entregues para o usuário pela aplicação. Neste caso, deve ser aplicada a contagem detalhada.

Para fins de faturamento, realizado durante o desenvolvimento, deve-se considerar a Contagem de Referência e posteriormente realizar os ajustes no faturamento após a Contagem Final.

É importante ressaltar que as mudanças de requisitos também serão consideradas

no tamanho do projeto a ser faturado (ver subseção 6.2.1). Além disso, se estas mudanças forem significativas, maiores que a evolução de requisitos (*scope creep*) prevista na estimativa inicial, o prazo do projeto deve ser reestimado. Toda mudança de requisito deve passar por uma análise de impacto entre contratante e contratada.

As subseções seguintes apresentam os métodos de estimativas de tamanho, prazo, custo e esforço a serem utilizados nos projetos de software em contratos.

6.1.1 Contagem Estimativa de Pontos de Função (CEPF)

Antes de definir o método de estimativas – Contagem Estimativa de Pontos de Função (CEPF), é importante destacar que “estimar significa utilizar o mínimo de tempo e esforço para se obter um valor aproximado dos pontos de função do projeto de software investigado” [Meli, 1999]. Assim, é recomendável sempre fazer uma distinção entre os termos e conceitos: contagem de pontos de função e estimativa de pontos de função.

• **Contagem de Pontos de Função:** significa medir o tamanho do software por meio do uso das regras de contagem do IFPUG [IFPUG, 2010b];

• **Estimativa de Pontos de Função:** significa fornecer uma avaliação aproximada do tamanho de um software utilizando métodos diferentes da contagem de pontos de função do IFPUG.

O método CEPF visa aferir o tamanho em PF de maneira simplificada, com base no conhecimento dos requisitos iniciais do projeto [Hazan, 2005]. A CEPF foi definida com base nas diretrizes adotadas no método Contagem Estimada de Pontos de Função da NESMA [NESMA, 2005]. A diferença é que o método da NESMA não recomenda a análise das funções identificadas, considerando todas as funções de dados identificadas com complexidade Baixa e as funções transacionais com complexidade Média. A CEPF propõe a análise das funcionalidades identificadas, e caso não seja possível determinar a complexidade, então são adotadas as diretrizes do método Contagem Estimada da NESMA. A CEPF também apresenta algumas dicas para ajudar um estimador no mapeamento dos requisitos iniciais nos tipos funcionais da Análise de Pontos de Função. Segue a descrição da CEPF [Hazan, 2005].

Primeiramente, os requisitos funcionais iniciais documentados nas propostas comerciais, nos documentos de visão, formalização simples de requisitos ou em qualquer especificação inicial do sistema do usuário são mapeados nos tipos funcionais da Análise de Pontos de Função: Arquivo Lógico Interno (ALI), Arquivo de Interface Externa (AIE), Entrada Externa (EE), Consulta Externa (CE) e Saída Externa (SE) (Figura 3). Posteriormente, os pontos de função são associados a cada função identificada, baseando-se nas tabelas de complexidade e de contribuição funcional do CPM (Tabela 1).

O estimador deve realizar uma leitura do documento inicial de requisitos, buscando informações relevantes para a identificação de processos elementares. O processo elementar é definido como a menor unidade de atividade significativa para o usuário. O processo elementar deve ser completo em si mesmo, independente e deixar a aplicação em um estado consistente [IFPUG, 2010b]. Em outras palavras, os processos elementares são funções transacionais independentes, isto é, funções sequenciais pertencem a um mesmo processo elementar e funções independentes constituem processos elementares diferentes.

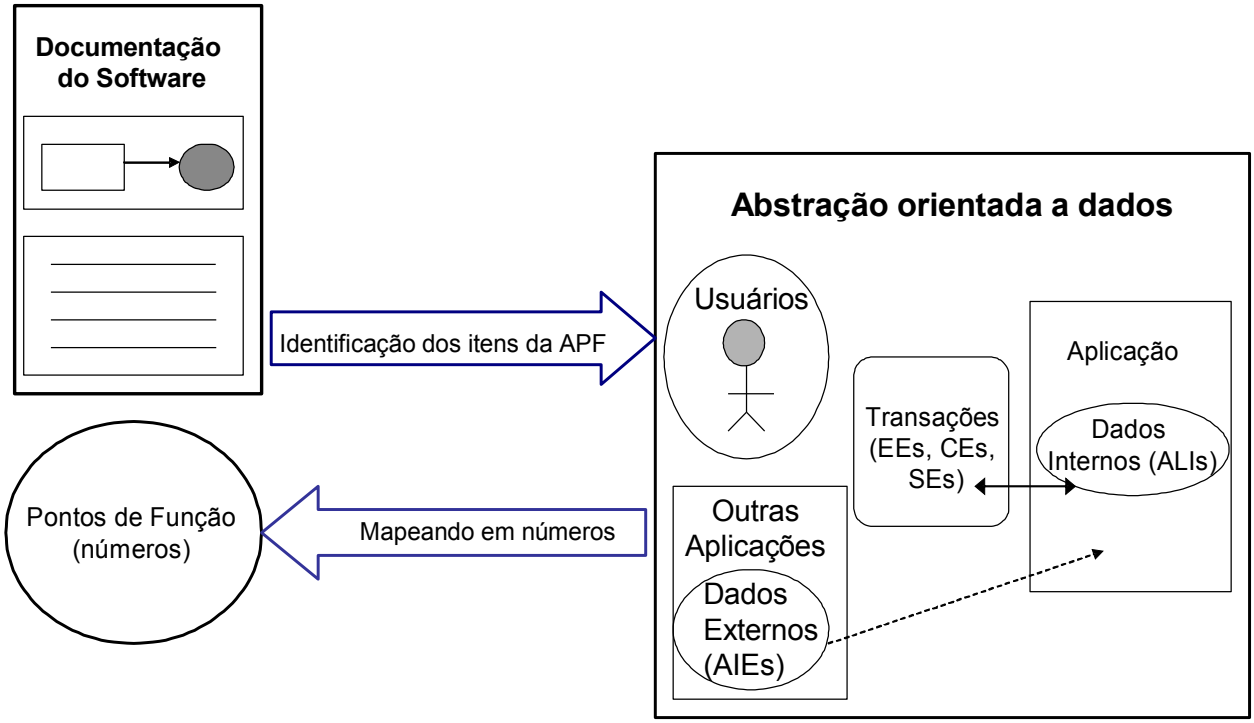

Figura 3: Modelo Lógico da Análise de Pontos de Função

Uma vez identificado o processo elementar, o estimador deve buscar o entendimento deste para classificá-lo em Entrada Externa, Consulta Externa ou Saída Externa. Adicionalmente, o estimador deve descobrir os dados associados ao processo elementar, visando a determinação da complexidade funcional da função identificada. Caso não seja possível a identificação da complexidade da funcionalidade em questão, recomenda-se a utilização da complexidade Média. Na análise do processo elementar também são identificados os grupos de dados lógicos da aplicação, que são classificados como Arquivos Lógicos Internos ou Arquivos de Interface Externa. Caso não seja possível a identificação da complexidade da função de dados em questão, recomenda-se a utilização da complexidade Baixa. É importante ressaltar que se o estimador identificar mais de um Registro Lógico no Arquivo Lógico Interno, recomenda-se utilizar a complexidade Média.

A seguir são apresentadas dicas para ajudar no mapeamento dos requisitos funcionais da aplicação nos tipos funcionais da APF. As necessidades e funcionalidades especificadas para o projeto, contidas no documento inicial de requisitos, devem ser enquadradas em uma das seguintes tabelas:

Tabela 2 - Contagem dos Arquivos Lógicos Internos (ALI): banco de dados lógico da aplicação (tabelas e arquivos mantidos pela aplicação).

Considerações: Identifique os grupos de dados lógicos de aplicação nos modelos de dados ou diagrama de classes ou a partir dos requisitos funcionais, descritos nos documentos de requisitos (documento de visão, relação de casos de uso, etc). Não considere arquivos físicos, arquivos de índices, arquivos de trabalho e tabelas de relacionamento sem atributos próprios (tabelas que existem para quebrar o relacionamento m x n e apenas transportam as chaves estrangeiras). As entidades fracas também não são consideradas um ALI. Se possível, tente descobrir os atributos lógicos, campos reconhecidos pelo usuário, e subgrupos de dados existentes para obter a complexidade funcional, segundo as regras de contagem do CPM. Caso não seja possível, a experiência tem mostrado que a maioria dos ALI dos sistemas são de complexidade **Baixa.**

N° ALI Baixa:	X 7 PF
N° ALI Média:	X 10 PF
N° ALI Alta:	X 15 PF
Total PF:	

Tabela 2: Identificação dos Arquivos Lógicos Internos da Aplicação

Tabela 3 - Contagem de Arquivos de Interface Externa (AIE): banco de dados de outras aplicações, **apenas referenciados** pela aplicação que está sendo estimada (tabelas e arquivos mantidos por outra aplicação).

Considerações: Identifique os grupos de dados lógicos de outras aplicações referenciados pela aplicação que está sendo estimada. Frequentemente, o referenciamento de dados ocorre para a validação de informações em cadastros ou consultas. Algumas vezes, relatórios ou consultas referenciam dados externos de outras aplicações, também considerados AIE. Não são considerados AIE arquivos físicos, arquivos de índice, arquivos de trabalho, tabelas de relacionamento sem atributos próprios e entidades fracas.

Geralmente, os AIE dos sistemas possuem a classificação de complexidade **Baixa,** porque são considerados para a determinação da complexidade funcional do AIE apenas os atributos referenciados pela aplicação que está sendo contada.

N° AIE Baixa:	X 5 PF
N° AIE Média:	X 7 PF
N° AIE Alta:	X 10 PF
Total PF:	

Tabela 3: Identificação dos Arquivos de Interface Externa da Aplicação

Tabela 4 - Contagem de Entradas Externas (EE): funcionalidades que mantêm os Arquivos Lógicos Internos (ALI) ou alteram o comportamento da aplicação.

Considerações: Identifique as funcionalidades de manutenção de dados. Conte separadamente a inclusão, alteração e exclusão de dados, isto é, cada função independente de inclusão, alteração ou exclusão deve ser contada separadamente. A aplicação possui funções de entrada de dados que alteram o comportamento dela, por exemplo: processamentos *batch* ou processamento de informações de controle? Caso positivo, estas funções também devem ser identificadas como Entradas Externas. Se você não possui conhecimento sobre o processo elementar (funcionalidade analisada), considere as Entradas Externas identificadas com complexidade **Média**.

N° EE Baixa:	X 3 PF
N° EE Média:	X 4 PF

N° EE Alta:	X 6 PF
Total PF:	

Tabela 4: Identificação das Entradas Externas da Aplicação

Tabela 5 - Contagem de Consultas Externas (CE): funcionalidades que apresentam informações para o usuário **sem** a utilização de cálculos ou algoritmos. São os processos elementares do tipo “lê - imprime”, “lê - apresenta dados”, incluindo consultas, relatórios, geração de arquivos pdf, xls, *downloads*, entre outros.

Considerações: Você está desenvolvendo uma função para apresentar informações para o usuário: uma consulta, relatório, *listbox*, *download*, geração de um arquivo, geração de arquivo pdf, xls? Esta função **não** possui cálculos ou algoritmos para derivação dos dados referenciados nem altera um Arquivo Lógico Interno e nem muda o comportamento do sistema? Caso positivo, estas funções devem ser identificadas como Consultas Externas. Se você não possui conhecimento sobre o processo elementar (funcionalidade analisada), considere as Consultas Externas com complexidade **Média**.

N° CE Baixa:	X 3 PF
N° CE Média:	X 4 PF
N° CE Alta:	X 6 PF
Total PF:	

Tabela 5: Identificação das Consultas Externas da Aplicação

Tabela 6 - Contagem de Saídas Externas (SE): funcionalidades que apresentam informações para o usuário **com** utilização de cálculos ou algoritmos para derivação de dados ou atualização de Arquivos Lógicos Internos ou mudança de comportamento da aplicação. São as consultas ou relatórios com totalização de dados, relatórios estatísticos, gráficos, geração de arquivos com atualização *log*, *downloads* com cálculo de percentual, entre outros.

Considerações: Você está desenvolvendo uma funcionalidade para apresentar informações para o usuário: uma consulta ou relatório com totalização de dados, etiquetas de código de barras, gráficos, relatórios estatísticos, *download* com percentual calculado, geração de arquivo com atualização de *log*? Caso positivo, estas funções devem ser identificadas como Saídas Externas. Observe que esta função *deve* ter cálculos ou algoritmos para processar os dados referenciados nos arquivos lógicos ou atualizar campos (normalmente indicadores) nos arquivos ou mudar o comportamento da aplicação. Se você não possui conhecimento sobre o processo elementar (funcionalidade analisada), considere as Saídas Externas com complexidade **Média.**

N° SE Baixa:	X 4 PF
N° SE Média:	X 5 PF
N° SE Alta:	X 7 PF
Total PF:	

Tabela 6: Identificação das Saídas Externas da Aplicação

A estimativa de tamanho do projeto em PF deve ser gerada com a totalização dos PF obtidos nas **Tabelas 2**, **3**, **4**, **5 e 6.**

A fórmula de contagem ou de estimativa de pontos de função para projetos de desenvolvimento é a seguinte:

PF_DESENVOLVIMENTO = PF_INCLUIDO + PF_CONVERSÃO

Este roteiro recomenda a supressão do PF_CONVERSÃO das fórmulas de contagem de pontos de função de projetos de desenvolvimento, conforme descrito na seção 3.3.

6.1.2 Estimativa de Esforço de Projetos de Software

Uma vez que o tamanho do projeto foi estimado em pontos de função, o próximo passo é estimar o esforço de desenvolvimento do projeto, **bem como sua distribuição pelas fases do ciclo de vida do desenvolvimento do software**. A Engenharia de Software possui vários modelos para estimar esforço de projetos de software, baseados em pontos de função, sendo o Modelo Simplificado de Estimativas [Vazquez, 2012] e o Modelo COCOMO II [Boehm, 2009] os mais utilizados. Neste roteiro é adotado o Modelo Simplificado de Estimativas.

O Modelo Simplificado de Estimativas consiste em obter um índice de produtividade em horas/PF para o projeto específico em questão, e então multiplicar o tamanho em PF do projeto pelo índice de produtividade, conforme a fórmula [Vazquez, 2012]:

Esforço (horas) = Tamanho (PF) x Índice de Produtividade (HH/PF)

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.

6.1.2.1 Distribuição de Esforço por Fase do Projeto

O próximo passo é a definição da distribuição de esforço pelas macroatividades (fases) do projeto, visando definir o valor agregado ao projeto após cada fase do ciclo de vida.

A Tabela 7 é uma sugestão de macroatividades e distribuição de esforço proposta neste roteiro. Ressaltamos que o órgão pode definir outras macroatividades e subdividi-las para melhor aderência à sua metodologia e aos marcos de entrega. Além disso, os percentuais de esforço sugeridos podem variar de acordo com o tipo de projeto e o processo de desenvolvimento utilizado no órgão. Nesses casos, as macroatividades e distribuição de esforço devem estar documentadas na metodologia do órgão (especificada contratualmente) ou formalizadas diretamente no contrato.

Macroatividades do Processo de Desenvolvimento de Software	Percentual de esforço (%)
Engenharia de Requisitos	25%
Design / Arquitetura	10%
Implementação	40%
Testes	15%
Homologação	5%
Implantação	5%

Tabela 7: Distribuição de Esforço por Macroatividades do Projeto

6.1.3 Estimativa de Prazo de Projetos de Software

As estimativas de prazo não são lineares com o tamanho do projeto. O melhor tempo de desenvolvimento (onde há uma melhor relação custo x benefício de alocação de recursos e menor prazo de desenvolvimento, dado o tamanho de um projeto específico), conforme fórmula descrita abaixo, é sugerido e utilizado nas estimativas de prazo deste roteiro.

Jones [Jones, 2007] propõe uma fórmula para o cálculo do melhor tempo de desenvolvimento, denominado Td e de Região Impossível (RI) de desenvolvimento (Figura 4). Na Região Impossível (RI), a adição de mais recursos ao projeto não implicará em redução no prazo. Note que a curva mostra que quanto menor o prazo almejado para a conclusão do projeto, maior será o esforço requerido e, consequentemente, maior o custo do projeto. O aumento do esforço para reduzir o prazo acontece através da realização de horas extras e da inclusão de pessoal adicional, gerando retrabalho. No entanto, a redução de prazo tem um limite, como demonstra a Região Impossível da Figura 4.

O método utilizado para estimar o prazo dos projetos (Td) é baseado na fórmula de Capers Jones [Jones, 2007]. Esta estima o prazo, baseando-se no tamanho do projeto em pontos de função, da seguinte maneira:

$$Td = V^{t}$$

Onde:

Td: prazo de desenvolvimento

V: tamanho do projeto em pontos de função

t: o expoente t é definido de acordo com a Tabela 8

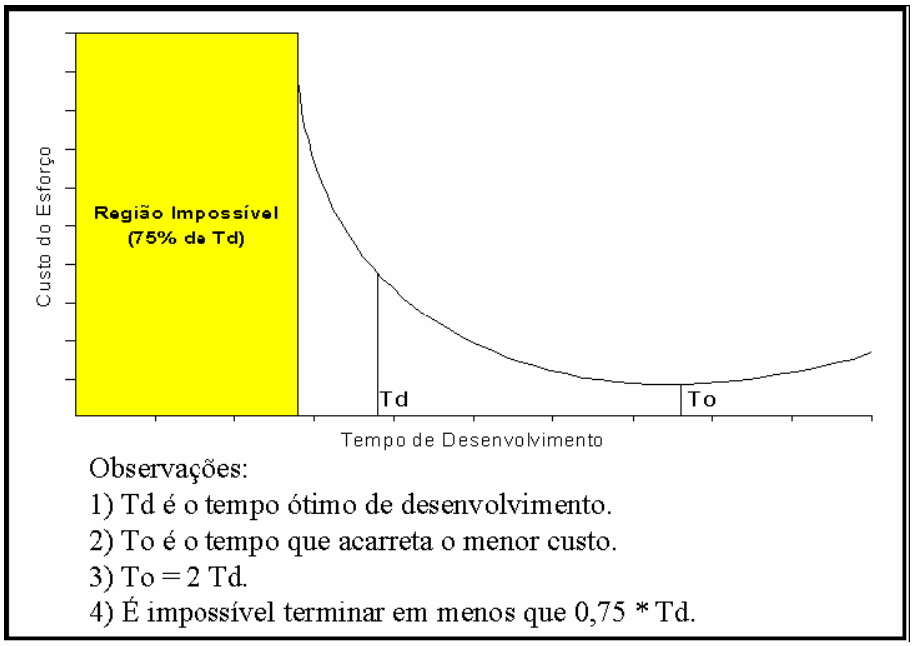

Figura 4: Relação entre a Estimativa de Prazo e de Esforço

Tipo de Sistema	Expoente t
Sistema Comum – Mainframe (desenvolvimento de sistema com alto grau de reuso ou manutenção evolutiva)	0,32 a 0,33
Sistema Comum – WEB ou Cliente Servidor	0,34 a 0,35
Sistema OO (se o projeto OO não for novidade para equipe, não tiver o desenvolvimento de componentes reusáveis, considerar sistema comum)	0,36
Sistema Cliente/Servidor (com alta complexidade arquitetural e integração com outros sistemas)	0,37
Sistemas Gerenciais complexos com muitas integrações, Datawarehousing, Geoprocessamento, Workflow	0,39
Software Básico, Frameworks, Sistemas Comerciais	0,40
Software Militar (ex: Defesa do Espaço Aéreo)	0,45

Tabela 8: Expoente t por Tipo de Projeto

É importante destacar que o método só deve ser aplicado para projetos com mais de 100 PF. Caso o órgão possua dados históricos de projetos, então este deve trabalhar com seus dados históricos e modelos de estimativas. Caso o projeto seja menor, o prazo deve ser obtido por meio da definição de prazo máximo por tamanho funcional com base em dados históricos do órgão, conforme a Tabela 9.

Tamanho do Projeto	Prazo máximo (em dias úteis)	
	Projetos Complexidade Baixa	Projetos Complexidade Média
Até 10 PF	9 dias	15 dias
De 11 PF a 20 PF	18 dias	30 dias
De 21 PF a 30 PF	27 dias	45 dias
De 31 PF a 40 PF	36 dias	60 dias
De 41PF a 50 PF	45 dias	75 dias
De 51 PF a 60 PF	54 dias	90 dias
De 61 PF a 70 PF	63 dias	105 dias
De 71 PF a 85 PF	70 dias	110 dias
De 86 PF a 99 PF	79 dias	110 dias

Tabela 9: Estimativa de Prazo de Projetos menores que 100 PF

Observação: Para os projetos de baixa complexidade foi considerada a produtividade de 7 hh/PF. Para projetos de média complexidade foi considerada a produtividade de 12 hh/PF, sendo o limite 110 dias úteis, equivalentes a 5 meses,que é o resultado da fórmula de Capers Jones para projetos de 100 PF - Td = $100^{0,35}$ = 5 meses. No caso de sistemas com complexidade alta, deve haver uma avaliação do órgão.

O prazo calculado considera todo o ciclo de vida do projeto, desde a fase de requisitos até a implantação. Assim, caso a estimativa tenha sido realizada ao final da fase de requisitos, descontar do prazo restante o tempo gasto com a fase de requisitos.

Caso seja necessário receber o projeto em um prazo menor que o calculado, recomenda-se propor um processo de desenvolvimento incremental, priorizando funcionalidades em cada iteração de acordo com a necessidade dele. Caso, ainda assim, a estimativa não atenda às necessidades do cliente, pode-se reduzir o Td em até 25%, observando-se a Região Impossível. No entanto, quanto mais perto da Região Impossível, o esforço e o custo do projeto aumentam de maneira exponencial. Assim, a redução de prazo de 10% implica no aumento de esforço de 20%; a redução de prazo de 20% implica no aumento de esforço de 50%; a redução de prazo de 25% implica em um aumento de esforço de 70%. Não é recomendada a redução de prazo em mais de 20%.

Os percentuais de aumento de esforço são estimados, podendo ser reajustados conforme avaliação da base histórica dos serviços realizados no órgão ou entidade.

Na seção seguinte é abordada a questão da distribuição de esforço e alocação de pessoas ao projeto.

6.1.4 Alocação de Equipe ao Projeto

Na alocação de equipe, deve-se considerar a estimativa de prazo e de esforço. Sugere-se utilizar a fórmula seguinte:

Equipe = Esforço (HH) / (21 x ProdDiária x Prazo)

Onde:

Prazo = Td em meses

ProdDiária = 6h/dia ou 7h/dia (recomenda-se considerar 6 horas/dia)

21 = dias úteis contidos em 1 mês

O tamanho da equipe é obtido em quantidade de recursos para o desenvolvimento do projeto e deve-se considerar percentuais de alocação. Por exemplo, suponha uma equipe de projeto de 2,2 recursos. Esta equipe pode conter 5 pessoas, sendo que 4 pessoas com 50% de alocação e um líder de projeto com 20% de alocação ao projeto.

6.1.5 Método para Estimativa de Custo

A estimativa de custo do projeto deve levar em consideração o custo de um ponto de função. Este custo deve abranger o custo da hora de todos os profissionais envolvidos no desenvolvimento da solução de software. O cálculo do custo do projeto (CP) será então da seguinte forma:

CP = QPF x CPF

Onde:

QPF = Tamanho do projeto em PF

CPF = Custo para implementar um ponto de função na plataforma em questão

6.1.6 Estimativa de Recursos Computacionais

A estimativa de recursos computacionais também deve ser considerada, pois constitui um componente importante para as estimativas de custo dos projetos. Um recurso computacional é um hardware que precisa ser adquirido ou que existe mas precisa ser configurado. Exemplos de recursos computacionais incluem, dentre outros: espaço em disco para o sistema entrar em produção, um servidor específico para teste ou homologação do sistema. Devem ser registradas as seguintes informações associadas aos recursos computacionais críticos:

• **Nome do Recurso Computacional:** [considere exclusivamente hardware: micro, periférico, expansão de memória, área em disco, banda de rede, etc]

• **Descrição:** [definição das características do recurso necessárias ao atendimento ao projeto]

• **Responsável pela Disponibilização:** [defina quem é o responsável pela disponibilização do recurso para o projeto]

• **Data Limite:** [informe a data limite para disponibilização do recurso]

• **Parâmetros:** [características do recurso: quantidade, perfil, configuração, etc]

• **Tipo do Recurso:** [D: recurso para ambiente de Desenvolvimento; P: recurso para ambiente de Produção; H: recurso para ambiente de Homologação]

• **Custo (Opcional):** [Custo do recurso computacional. Não considerar custos de processamento ou custos operacionais de produção]

Caso o projeto a ser desenvolvido não possua nenhum recurso computacional crítico, isto deve ser registrado no documento de estimativas.

6.2 Diretrizes para Acompanhamento de Projetos

Esta seção apresenta considerações especiais sobre o gerenciamento de mudança de requisitos, projetos cancelados, progresso de projetos, assim como o tratamento de redução de cronograma e fator criticidade.

6.2.1 Considerações sobre Mudança de Requisitos

Em projetos de desenvolvimento e de manutenção de software é bastante observada a mudança de requisitos anterior à implantação do projeto, conforme o usuário e o desenvolvedor adquirem mais conhecimento sobre as necessidades e funcionalidades de negócio [Sommerville, 2007]. O CPM denomina este fenômeno de *Scope Creep*.

Nas estimativas iniciais de tamanho de projetos de desenvolvimento, após a fase de especificação, considerando-se o documento de visão inicial do projeto, recomenda-se utilizar um percentual de 30% a 40% para evolução de requisitos. Este percentual é sugerido, ficando a critério da instituição estabelecê-lo contratualmente. Por exemplo, suponha que após a análise do documento de visão de um projeto, aplicando-se a CEPF, foi obtido o tamanho de 200 PF, então o tamanho estimado desse projeto é de 270 PF (200 + 35%), utilizando-se a premissa de evolução de requisitos em 35%. Esta premissa do projeto deve ser documentada. Nas estimativas, após a fase de requisitos, utilizando-se como insumo as especificações de casos de uso, deve-se considerar um percentual de 20% a 30% para evolução de requisitos.

Uma mudança de requisito anterior à implantação do projeto gera retrabalho para a equipe de desenvolvimento, aumentando assim o esforço e o custo do projeto. Neste roteiro, as demandas de mudança de requisitos serão dimensionadas como PF_RETRABALHO, contadas à parte do projeto de desenvolvimento ou de manutenção.

Cabe ressaltar que para evitar as solicitações de mudança de requisitos devido a falhas na execução da fase de engenharia de requisitos, é importante que seja dada atenção especial à atividade de validação e aceitação dos requisitos.

O método de contagem de mudança de requisitos descrito neste roteiro tem os seguintes pressupostos:

- As demandas de mudança de requisitos são contagens à parte da contagem do projeto de desenvolvimento ou de manutenção e devem considerar as funcionalidades **antes da mudança**;
- A quantidade de PF_RETRABALHO apurada leva em conta o esforço já realizado no processo de desenvolvimento da funcionalidade até o momento da solicitação de mudança de requisitos. Nos projetos onde exista o gerenciamento e o acompanhamento do seu progresso (subseção 6.2.3), **é preciso aplicar o percentual das atividades concluídas** das fases do processo de desenvolvimento até o momento da mudança de requisitos na fórmula do cálculo do PF_RETRABALHO. Nos projetos sem gerenciamento do seu progresso, **é**

preciso aplicar o percentual das fases concluídas na fórmula do cálculo do PF_RETRABALHO. A distribuição de esforço sugerida na Tabela 7 estabelece os percentuais por fase, de forma a permitir a contagem de mudança de requisito conforme o estágio do projeto. Ressaltamos que os percentuais de esforço sugeridos podem variar de acordo com o tipo de projeto e o processo de desenvolvimento utilizado no órgão. Essa distribuição de esforço deve ser definida no contrato de software.

- A contagem do projeto de desenvolvimento ou de manutenção deverá ser atualizada a cada demanda de mudança de requisitos, visando refletir as funcionalidades **após a mudança**.
- Para fins de planejamento ou de faturamento, a quantidade total de pontos de função será obtido da seguinte forma:

$$PF\_TOTAL = PF\_PROJETO + \sum PF\_RETRABALHO$$

Onde: PF_PROJETO é a última versão da contagem do escopo do projeto (PF_DESENVOLVIMENTO, PF_MELHORIA, PF_ADAPTATIVA, etc).

A contagem de PF_RETRABALHO leva em conta as seguintes características:

- Requisito original: é o requisito do projeto de desenvolvimento ou de manutenção original, que pode ser incluir, alterar ou excluir funcionalidades de um aplicativo.
- Tipo da mudança do requisito: é a natureza da mudança de requisitos no projeto em andamento, que pode ser acrescentar um requisito, alterar um requisito definido ou desistir de um requisito (retirar do escopo do projeto).

A Tabela 10 resume os percentuais que devem ser aplicados sobre as funções alteradas (considerando o tamanho **antes da mudança**) para obtenção de PF_RETRABALHO:

Fator			Requisito Original		
			Incluir Função	Alterar Função	Excluir Função
Tipo da Mudança de Requisito	Acréscimo		-	-	-
	Alteração	Alteração de Requisitos	50%	50%	-
		Alteração de Interface	0,6 PF	0,6 PF	-
	Desistência		130%	80%	30%

Tabela 10 – Percentuais definidos para a mudança de requisitos

Cabe ressaltar que a quantidade de PF_RETRABALHO obtida, para fins de planejamento, gestão e faturamento, usa na sua fórmula o percentual das fases ou atividades concluídas até o momento da solicitação da mudança de requisitos, conforme descrito acima.

Observações Importantes:

1. Recomenda-se que o registro das demandas de alteração de requisitos seja realizado em separado, sendo contado em uma planilha de PF_RETRABALHO à parte da contagem de PF do projeto. Apesar das medições em separado, elas ainda devem guardar vínculo com o projeto em andamento, fazendo parte da sua *baseline* de tamanho.
2. O cálculo do PF_RETRABALHO deve registrar o percentual das **fases concluídas** do processo de desenvolvimento até o momento da mudança de requisitos, para projetos que não tenham o gerenciamento do seu progresso, conforme descrito na subseção 6.2.3. Nos projetos onde exista o gerenciamento do seu progresso, o PF_RETRABALHO deve registrar o percentual das **atividades concluídas** das fases do processo de desenvolvimento, no momento da mudança de requisitos, usando os registros de acompanhamento do progresso do projeto ilustrados na subseção 6.2.3.

A seguir são descritos os tipos de mudança nos projetos.

1. **Acréscimo de funcionalidades ao escopo do projeto**

As mudanças que não tragam impacto aos requisitos originais do projeto, caracterizadas pelo **acréscimo de funcionalidades ao escopo do projeto** de desenvolvimento ou de manutenção, serão acrescentadas na contagem de PF do projeto e não geram contagem de PF_RETRABALHO, ou seja, representam um trabalho adicional e não retrabalho. Enquadram-se nesta situação a inclusão, a alteração ou a exclusão de funções que não constavam no escopo do projeto original.

2. **Alteração de função**

A contagem de PF_RETRABALHO referente à alteração deve considerar o percentual de 50% sobre o tamanho da função antes da alteração, independentemente do requisito original. Este item se refere somente à alteração de requisitos de funcionalidades que estavam sendo criadas ou alteradas no projeto original (**Caso 1**).

Em caso de mudanças em interface (cosméticas), conforme apresentado na seção 4.7, considerar o percentual de 20% da contagem de uma função transacional de mais baixa complexidade (3 PF), ou seja 0,6 PF, independentemente da complexidade da função antes da alteração (**Caso 2**).

Sobre a quantidade de PF_RETRABALHO obtida, para fins de gestão e faturamento, deverá ser aplicado o percentual das **fases concluídas** até o momento da solicitação de mudança de requisitos, para projetos que não tenham o gerenciamento do seu progresso, conforme descrito na subseção 6.2.3, e o percentual das **atividades concluídas**, para projetos que tenham o gerenciamento do seu progresso, conforme os registros de acompanhamento do progresso do projeto, ilustrados na subseção 6.2.3.

A contagem de PF do projeto deve ser atualizada para refletir o novo grau de complexidade da função após a mudança.

Exemplo:

Considerando-se que um projeto de melhoria tinha como escopo a alteração de uma EE (complexidade alta - 6 PF) e a criação de uma CE (complexidade baixa - 3 PF) e uma SE (complexidade baixa - 4 PF). Além disso, não é feito o gerenciamento do progresso desse projeto. A contagem de PF_MELHORIA é:

- *Inclusão de CE e SE: 3 PF + 4 PF = 7 PF*
- *Alteração de EE: 6 PF * 50% = 3 PF*
- *PF_MELHORIA $_{v1}$ = 10 PF*

Caso 1: Alteração de requisitos

No início da homologação foram solicitadas mudanças nos requisitos da EE e da CE, sendo que a complexidade da CE passou a ser média (4 PF) após a mudança. Nesta situação hipotética, a contagem de PF_RETRABALHO será a seguinte:

- *EE original: 6 PF*
- *CE original: 3 PF*
- *PF_RETRABALHO = (6 PF + 3 PF) x 50%[Nota 1] = 4,5 PF*
- *PF_RETRABALHO = 4,5 PF x 90%[Nota 2]*
- *PF_RETRABALHO = 4,05 PF*

Nota 1: 50% é o percentual a ser aplicado sobre o tamanho da função original antes da sua alteração, conforme apresentado na Tabela 10.

*Nota 2: No contexto do exemplo e usando a distribuição de esforço da Tabela 7, o projeto na fase de testes (a **última fase concluída** antes da fase de homologação) registra progresso de 90%. Assim, para fins de gestão e faturamento, o* valor *do PF_RETRABALHO seria o correspondente a: 4,5 PF * 90% = 4,05 PF "cheios".*

A contagem de PF_MELHORIA deverá ser atualizada para refletir o aumento da complexidade da CE alterada:

- *Inclusão de CE alterada e SE: 4 PF + 4 PF = 8 PF*
- *Alteração de EE alterada: 6 PF * 50% = 3 PF*
- *PF_MELHORIA $_{v2}$: 11 PF*

Caso 2: Alteração de interface

Durante a fase de implementação foi solicitada uma alteração na função SE, que é um relatório. A demanda é para alterar o tipo de fonte do título do relatório (alteração de interface - cosmética). A complexidade da função SE se mantém a mesma (complexidade baixa - 4 PF) após a mudança. Nesta situação hipotética, a contagem de PF_RETRABALHO será a seguinte:

- *SE original: 4 PF*
- *PF_RETRABALHO = 0,6 PF [Nota 3]*
- *PF_RETRABALHO = 0,6 PF x 35%[Nota 4]*
- *PF_RETRABALHO = 0,21 PF*

Nota 3: 0,6 PF corresponde a 20% de uma função de baixa complexidade (3PF), independente do tamanho da função original antes da sua alteração, conforme apresentado na Tabela 10.

*Nota 4: No contexto do exemplo e usando a distribuição de esforço da Tabela 7, o projeto na fase de Design/Arquitetura (a **última fase concluída** antes da fase de implementação, onde ocorreu a solicitação de mudança) registra progresso de 35%. Assim, para fins de gestão e faturamento, o valor do PF_RETRABALHO seria o correspondente a: 0,6 PF * 35% = 0,21 PF "cheios".*

Nesse caso de mudança de requisitos com alteração de interface (cosmética), a contagem de PF_MELHORIA do projeto original não sofre alteração, visto que a complexidade da função SE não é alterada.

3. Desistência de incluir, alterar ou excluir uma função

Em caso de desistência de incluir, alterar ou excluir uma função, deve-se verificar qual era o requisito original, pois o percentual a ser utilizado na contagem de PF_RETRABALHO varia para cada situação, conforme apresentado na Tabela 10. Além do trabalho de retirar o que foi requisitado (percentuais definidos na Tabela 10), deve-se considerar também em PF_RETRABALHO, o trabalho realizado (fases ou atividades concluídas do processo de desenvolvimento) até o momento da desistência desse requisito. Por fim, o requisito original deve ser removido do PF_MELHORIA. Enquadram-se nesta situação somente as desistências de incluir, de alterar ou de excluir funcionalidades que constavam no escopo do projeto.

Quando a mudança no projeto for deixar de incluir uma função, aplica-se o percentual de 130% ao tamanho da função original. Esse valor é resultado da soma do percentual de 100% da inclusão (escopo original) com os 30% correspondentes à exclusão dessa mesma função.

Quando a mudança no projeto for deixar de alterar uma função, aplica-se o percentual de 80% ao tamanho da função original. Esse valor é o resultado da soma do percentual de 50% da alteração (escopo original) com os 30% referentes à exclusão dessa mesma função.

Quando a mudança no projeto for deixar de excluir uma função, aplica-se apenas o percentual de 30% referente à exclusão da função original.

Em todos os casos, a contagem de PF_MELHORIA deve ser atualizada removendo-se as funções que não fazem mais parte do escopo do projeto.

Da mesma forma que no item 2 (**Alteração de função**), para fins de gestão e faturamento, sobre a quantidade de PF_RETRABALHO é aplicado o percentual das **fases concluídas** até o momento da solicitação de mudança de requisitos, para projetos que não tenham o gerenciamento do seu progresso, conforme descrito na subseção 6.2.3, e o percentual das **atividades concluídas**, para projetos que tenham o gerenciamento do seu progresso, conforme os registros de acompanhamento do progresso do projeto, ilustrados na subseção 6.2.3.

Exemplos:

Desistência de incluir função

Suponha que um projeto de melhoria para a criação do relatório XPTO, contado como uma SE de complexidade média com 5 PF, teve uma demanda de exclusão do projeto de melhoria durante a fase de implementação (ou seja, o relatório não será mais construído). Suponha, também, que não é feito o gerenciamento do progresso desse projeto. Desta forma a contagem de PF_RETRABALHO será a seguinte:

- *SE original: 5 PF*
- *PF_RETRABALHO = 5 PF x 130%[Nota 5] = 6,5 PF*
- *PF_RETRABALHO = 6,5 PF x 35%[Nota 6]*
- *PF_RETRABALHO = 2,275 PF*

Nota 5: 130% é o percentual a ser aplicado sobre o tamanho da função original antes da desistência da sua inclusão, conforme apresentado na Tabela 10.

*Nota 6: No contexto do exemplo e usando a distribuição de esforço da Tabela 7, o projeto na fase de Design/Arquitetura (a **última fase concluída** antes da fase de implementação, onde ocorreu a solicitação de mudança) registra progresso de 35%. Assim, para fins de gestão e faturamento, o valor do PF_RETRABALHO seria o correspondente a: 6,5 PF * 35% = 2,275 PF "cheios".*

A contagem de PF_MELHORIA do projeto deve ser atualizada para que o relatório XPTO deixe de constar na medição, conforme fórmula abaixo:

- *Inclusão de SE: 5 PF*
- *$PF\_MELHORIA_{v2} = PF\_MELHORIA_{v1}$ – (Inclusão de SE)*
- *$PF\_MELHORIA_{v2} = PF\_MELHORIA_{v1} – 5\ PF$*

Desistência de alterar função

Se, no exemplo anterior, o relatório XPTO estivesse sendo originalmente alterado (ao invés de incluído), a única diferença seria no percentual aplicado em PF_RETRABALHO:

- *SE original: 5 PF*
- *PF_RETRABALHO = 5 PF x 80% [Nota 7]= 4 PF*
- *PF_RETRABALHO = 4 PF x 35%[Nota 8]*
- *PF_RETRABALHO = 1,4 PF*

Nota 7: 80% é o percentual a ser aplicado sobre o tamanho da função original antes da desistência da sua alteração, conforme apresentado na Tabela 10.

*Nota 8: No contexto do exemplo e usando a distribuição de esforço da Tabela 7, o projeto na fase de Design/Arquitetura (a **última fase concluída** antes da fase de implementação, onde ocorreu a solicitação de mudança) registra progresso de 35%. Assim, para fins de gestão e faturamento, o valor do PF_RETRABALHO seria o correspondente a: 4 PF * 35% = 1,4 PF "cheios".*

A contagem de PF_MELHORIA do projeto deve ser atualizada para que o requisito original (alteração do relatório XPTO, contado como uma SE de complexidade média com 5 PF) deixe de constar na medição.

- *Alteração de SE: 5 PF * 50% = 2,5 PF*
- *PF_ MELHORIA $_{v2}$ = PF_ MELHORIA $_{v1}$ – (Alteração de SE)*
- *PF_ MELHORIA $_{v2}$ = PF_ MELHORIA $_{v1}$ – (2,5 PF)*

4. Desistência de alterar uma função seguida de exclusão da função

Quando a solicitação de mudança seja não só deixar de fazer o que estava no projeto original, mas também excluir a função da aplicação, deve-se considerar esses dois aspectos separadamente, como se fossem duas mudanças consecutivas:

A) Conta-se a desistência de alterar a função conforme descrito no item 3 (**Desistência de incluir, alterar ou excluir uma função**), apurando a quantidade de PF_RETRABALHO correspondente e a atualização do PF_MELHORIA;

B) Conta-se o acréscimo ao escopo do projeto (excluir a função da aplicação) conforme descrito no item 1 (Acréscimo ao escopo do projeto), atualizando-se PF_MELHORIA.

6.2.2 Considerações sobre Projetos Cancelados

Em alguns casos, devido a mudanças no ambiente da contratante, uma demanda ou parte de um projeto de desenvolvimento ou manutenção pode ser cancelado a critério da contratante. Nestes casos, o tamanho funcional das funcionalidades canceladas será aferido por meio da contagem de pontos de função das funcionalidades canceladas e um fator de impacto.

O fator de impacto é definido com base no percentual de esforço alocado à construção da funcionalidade em questão, observando a Tabela 7 de distribuição de esforço contida na subseção 6.1.2.1 ou alguma diretriz específica de distribuição de esforço do contrato em questão. O fator de impacto deve ser aplicado na contagem de pontos de função das funcionalidades em questão. É importante ressaltar que em um processo de desenvolvimento incremental uma funcionalidade pode, por exemplo, estar em fase de requisitos e de testes, porque o plano de testes é construído na fase de requisitos. O progresso das atividades executadas em cada funcionalidade do projeto deve ser obtido por meio do acompanhamento do plano do projeto descrito na subseção seguinte.

6.2.3 Gerenciamento de Progresso de Projetos

O acompanhamento do projeto deve identificar o progresso de cada requisito do projeto, ou seja, o percentual de conclusão de cada fase do processo de software para o requisito em questão.

A apuração do percentual concluído em cada fase deve ser definido em comum acordo entre o órgão contratante e a empresa contratada, de acordo com os artefatos entregues em cada fase. Os artefatos que estão na fábrica, mas não foram entregues, não devem ser considerados nessa apuração.

Segue um exemplo de acompanhamento do progresso do desenvolvimento de um Sistema de Gestão de Projetos, que mostra para cada um dos requisitos o percentual concluído de cada fase:

Requisito	Tamanho	Engenharia de Requisitos	Design, Arquitetura	Implementação	Testes	Homologação	Implantação
Caso de Uso 1 Atividade Incluir Ativ. Alterar Ativ Excluir Ativ Consultar Ativ	19 PF	50,00%	20%	0%	10%	0%	0%
Caso de Uso 2 - Relatório de Projetos	5 PF	100 %	100%	50%	20%	0%	0%
....							

Supondo a mudança de requisitos no Caso de Uso 2 do exemplo acima, para inclusão de uma nova informação a ser apresentada no Relatório, a contagem de PF do requisito original é a seguinte:

Caso de Uso 2 – Relatório de Projetos – 5 PF

Macroatividades	Esforço da Fase	Tamanho	Esforço realizado	Tamanho realizado
Engenharia de Requisitos	25%	1,25 PF	100%	1,25 PF
Design, Arquitetura	10%	0,5 PF	100%	0,5 PF
Implementação	40%	2 PF	50%	1 PF
Testes	15%	0,75 PF	20%	0,15 PF
Homologação	5%	0,25 PF	0%	0 PF
Implantação	5%	0,25 PF	0%	0 PF

Total: 2,9 PF

O tamanho realizado do requisito original é de 2,9 PF. Conforme descrito na subseção 6.2.1, para o cálculo do PF_RETRABALHO do requisito alterado será considerado o fator de impacto de 50% na contagem de PF. Portanto, a contagem do PF_RETRABALHO é 2,9 x 0,50 = 1,45 PF.

6.2.4 Considerações sobre Redução de Cronograma

Como apresentado anteriormente, as estimativas de prazo não são lineares com o tamanho do projeto. Jones [Jones, 2007] propõe uma fórmula, descrita na subseção 6.1.3, para o cálculo do melhor tempo de desenvolvimento (onde há uma melhor relação custo x benefício de alocação de recursos e menor prazo de desenvolvimento), dado o tamanho de um projeto específico.

Alguns projetos, devido à legislação e a outros fatores externos, já se iniciam com um prazo imposto. Se este prazo for igual ou superior ao prazo calculado pela fórmula de *Capers Jones* (expoente t) ou, em caso de projetos pequenos (menores que 100 PF), igual ou superior a um prazo calculado considerando o trabalho da equipe de 7 horas/dia

nos dias úteis (conforme sugerido na subseção 6.1.3), então o projeto é tratado como normal.

No entanto, se o projeto tiver um prazo imposto inferior ao prazo calculado, então pode-se considerar a seguinte proposta como uma sugestão de valores:

•Redução de prazo de 10%: aumento de esforço de 20% (projetos urgentes);

•Redução de prazo de 20%: aumento de esforço de 50% (projetos críticos);

•Redução de prazo de 25%: aumento de esforço de 70% (projetos de alta criticidade).

Os valores acima devem ser avaliados e definidos a critério do órgão, caso esse cenário possa ocorrer durante o contrato.

Deve-se ressaltar que não é possível uma redução de prazo maior que 25%, devido aos cálculos de Região Impossível e ainda, conforme nos aproximamos da Região Impossível, o esforço e o custo do projeto aumentam de maneira exponencial.

Como os riscos da redução de cronograma também são altos, não é recomendada a redução de cronograma. Deve-se tentar priorizar funcionalidades trabalhando com o processo incremental.

Caso o contrato seja baseado em preço por pontos de função, este aumento de esforço será refletido na contagem de PF. Assim, um aumento de esforço de 20% implica em aumento de 20% no custo de PF; aumento de esforço de 50% implica em aumento de 50% no custo de PF; e o aumento de esforço de 70% implica em aumento de 70% no custo de PF.

Não é recomendado o uso de redução de cronograma, pode-se utilizar processos incrementais de desenvolvimento e trabalhar com definição de prioridades. É importante ressaltar que estas questões devem ser definidas em cláusulas contratuais e devem ser consideradas no orçamento do contratante.

6.2.5 Fator de Criticidade de Solicitação de Serviço

Em função da criticidade e da necessidade de alocação de recursos extras para atendimento da demanda no prazo estipulado pelo cliente, sugere-se adotar um fator de criticidade de 1,35 (um vírgula trinta e cinco), que deverá ser multiplicado pelo tamanho funcional da demanda considerada crítica, de modo a remunerar adequadamente o aumento do esforço de atendimento. Este fator é considerado para demandas que devem ser atendidas em finais de semana, feriados e fora do horário comercial. Entende-se como horário comercial o horário de 08:00 às 18:00.

É importante ressaltar que estas questões devem ser definidas em cláusulas contratuais e devem ser consideradas no orçamento do contratante.

7. Contagem de Pontos de Função no Desenvolvimento de Software utilizando Métodos Ágeis

Este capítulo descreve orientações sobre o uso da métrica Ponto de Função na medição e remuneração de serviços de desenvolvimento de software com métodos ágeis, a fim de subsidiar as contratações desses serviços na Administração Pública Federal (APF).

Uma das principais dificuldades e desafios na adoção de métodos ágeis em contratação de desenvolvimento de software é definir um modelo de remuneração que seja equilibrado, remunerando de forma justa o esforço da contratada para atender o volume de refinamentos e mudanças em funcionalidades e, ao mesmo tempo, não onerando de forma excessiva a contratante (instituições públicas), ou seja, o valor pago deve corresponder aos serviços recebidos e o ciclo do processo ágil de desenvolvimento de software não deve influenciar negativamente o ciclo de faturamento do projeto. Devido às características inerentes ao processo ágil, entende-se que os refinamentos e as mudanças em funcionalidades são mais constantes e recorrentes nesse cenário de desenvolvimento de software, pois pressupõe-se um escopo mais aberto. Entretanto, o processo ágil de desenvolvimento de software em contratações não deve comprometer os princípios de economicidade e efetividade dos resultados previstos e entregues com a garantia da exequibilidade do projeto.

É importante observar que, conforme a Súmula TCU 269, a remuneração nas contratações de serviços de Tecnologia da Informação deve estar vinculada à entrega de resultados ou ao atendimento de níveis de serviço. Portanto, é relevante que o modelo de remuneração, níveis de serviço e critérios de qualidade para a aceitação dos resultados entregues ao final das iterações (*sprints*) do processo ágil estejam claramente definidos no instrumento convocatório e sejam observados e aferidos durante a gestão do contrato.

O propósito deste trabalho é possibilitar a adoção de um modelo de faturamento baseado na métrica Ponto de Função, capaz de medir a entrega de resultados do projeto, em contratações de desenvolvimento de software usando métodos ágeis. Os objetivos e premissas considerados neste trabalho foram:

- Buscar a simplicidade na medição do desenvolvimento de software com métodos ágeis para viabilizar o seu uso em contratações com responsabilidade e garantir o alcance dos benefícios do processo ágil ao negócio;
- Fortalecer a necessidade e importância de registro das mudanças em funcionalidades para promover a criação de uma base histórica de projetos com desenvolvimento ágil, além de permitir a rastreabilidade e o atendimento de auditorias nos projetos;
- Minimizar o ônus na gestão de projeto advindo da utilização do processo ágil em contratações de software, tanto para a contratante (gestor de TI) quanto para a contratada;
- Simplificar o ônus de gestão e controle de mudanças de forma a minimizar impactos sobre a agilidade do processo de desenvolvimento;
- Prever, medir e remunerar o esforço e o volume de mudanças em funcionalidades em um projeto com desenvolvimento ágil;
- Promover a oferta de preços exequíveis para a realização de contratos de prestação de serviços de desenvolvimento de software com métodos ágeis, a partir da publicidade de características do projeto e da equipe de desenvolvimento

previamente especificadas no edital de contratação;

- Incentivar o uso de desenvolvimento ágil no governo com o aprimoramento da maturidade e competência da equipe envolvida, buscando a eficiência do processo e dos resultados esperados.

Nesse sentido, são apresentadas recomendações de medição em Ponto de Função, que podem ser adotadas nas contratações de desenvolvimento de software com métodos ágeis, para o tratamento dos refinamentos e mudanças em funcionalidades durante o projeto. Essas recomendações foram definidas após o levantamento e avaliação da literatura e de guias de contagem de órgãos do governo que utilizam alguma abordagem de medição para o cenário de desenvolvimento de software com métodos ágeis [CAIXA, 2012], [Castro e Hernandes, 2014], [Horvath, 2012], [Keote, 2010], [NESMA, 2009] e [PROCERGS, 2013]. Realizou-se, ainda, um *benchmarking* em órgãos do governo que já implementam contratos de desenvolvimento de software com métodos ágeis e, por fim, foram realizadas reuniões técnicas com especialistas em desenvolvimento ágil e na métrica Ponto de Função para discutir e refinar a proposta apresentada neste documento.

7.1 Conceitos

No cenário de desenvolvimento de software com métodos ágeis e dentro do contexto deste roteiro, é importante alinhar os seguintes conceitos:

Release: É um ciclo que perpassa pelas fases do processo de desenvolvimento de software com o objetivo de entregar, ao final do ciclo, um **produto pronto** a ser colocado em produção para uso. A duração de cada *release* será definida pela contratante na fase de planejamento do projeto conforme seu *backlog* priorizado de forma a garantir uma entrega de valor antecipada aos usuários.

***Sprint*:** É uma unidade de período de tempo fixo (*time box*) dentro da *release*, com datas de início e fim pré-definidas, dentro da qual é executado um conjunto de atividades de desenvolvimento do projeto previamente estabelecidas, gerando ao final um incremento do produto aceito e potencialmente implantável.

Ciclo de Pagamento: período definido para fins de pagamento e apuração dos resultados entregues, podendo consistir de uma iteração (*sprint*), de um conjunto de iterações, ou de uma *release*. Considerando os critérios adotados para o projeto, como o tamanho da iteração (*sprint*), o tamanho da *release*, a produtividade da equipe do projeto e a expectativa de fluxo de caixa da contratada para manter seu equilíbrio econômico-financeiro no atendimento do contrato, deve-se realizar o faturamento por iteração (*sprint*), por grupo de iterações, ou por *release*, desde que sempre devidamente associado aos produtos entregues e aceitos de uma ordem de serviço.

Produto Pronto: Visando a remuneração da contratada a partir da medição de resultados gerados em um "ciclo de pagamento", entende-se que um produto está "pronto" se foi entregue e aceito. Cabe observar que o desenvolvimento de uma funcionalidade pode perpassar mais de uma *sprint* e conter várias histórias de usuários prontas e validadas em *sprints* diferentes. Ness caso, a funcionalidade só será considerada para fins de pagamento ao final do ciclo de pagamento em que estiver com todas as suas histórias componentes "prontas".

Refinamentos: são quaisquer mudanças ocorridas sobre uma função transacional ou de dados já previamente trabalhada(s) na *release* corrente (seja por meio de uma inclusão, alteração ou exclusão), provocadas pelo aprofundamento,

detalhamento e complementação de requisitos durante o processo de desenvolvimento.

7.2. Orientações

O desenvolvimento de software utilizando métodos ágeis deve respeitar uma abordagem específica que considere as características dos métodos ágeis, tanto no desenvolvimento quanto na gestão do projeto. Entretanto, essas características podem requerer adaptações para o contexto de contratações de software na APF, no sentido de atender o cumprimento da legislação e dos princípios de economicidade e eficiência. Nesse cenário, algumas considerações e sugestões são propostas para o desenvolvimento de software utilizando métodos ágeis na APF:

- Remuneração baseada nos resultados entregues e aceitos (Produto Pronto);
- Remuneração sempre atrelada a uma ordem de serviço;
- Promover o fluxo de demandas do projeto e o equilíbrio econômico-financeiro da contratada;
- Divisão do projeto de desenvolvimento ou manutenção em *releases;*
- Ciclo da *sprint* (iteração) de 2 até 4 semanas;
- Ciclo da *release* não deve ser igual ao ciclo da *sprint*, ou seja, *release* formada por apenas uma *sprint* não permite a adoção das orientações trazidas neste documento;
- Ciclo da *release* deve, sempre, promover o aumento do percentual de completude do sistema (entrega de valor agregado ao negócio);
- Para as funcionalidades que precisem de mais de uma *sprint* para serem desenvolvidas, recomenda-se que sejam contadas somente na *sprint* em que forem entregues e aceitas;
- Realizar a contagem estimativa do projeto, a fim de definir o tamanho estimado ao final de um “ciclo de pagamento” para efeito de planejamento do projeto e geração das ordens de serviço de desenvolvimento ou manutenção de software.

Para efeito de gestão das mudanças e geração de indicadores, recomenda-se que as mudanças em funcionalidades sejam registradas em planilha separada da contagem do projeto de desenvolvimento. Nessa planilha de mudanças devem ser registradas todas as funcionalidades incluídas, alteradas e excluídas, porém, para efeito de faturamento, sugere-se que as funcionalidades incluídas sejam remuneradas somente na contagem final do “ciclo de pagamento”, conforme modelo de remuneração adotado para o projeto, a fim de não existir duplicidade de remuneração.

Caso o “ciclo de pagamento” seja por *sprint*, sugere-se adotar, como critério de remuneração para a *sprint*, um valor percentual do tamanho total planejado da *release* ou uma medida que reflita o valor agregado dos produtos prontos da *sprint* dentro da *release*. Essa sugestão visa não onerar as partes envolvidas com a medição, controle e gestão de mudanças a cada *sprint*, principalmente, quando a duração dessas iterações são muito curtas. Caso a remuneração seja realizada por *sprint*, sugere-se reter um percentual do total da remuneração da iteração (*sprint*) para o final da *release*, principalmente, quando a conclusão da implantação do produto pronto da *sprint* ocorrer no final da *release*.

O item Diretrizes para Acompanhamento de Projetos deste Roteiro não deve ser

aplicado para projetos de desenvolvimento de software com métodos ágeis, em virtude da adoção das orientações contidas neste capítulo específico para projetos dessa natureza.

7.3 Tratamento de Mudanças em Funcionalidades no Processo Ágil

Esta seção apresenta orientações sobre o tratamento de mudanças em funcionalidades para contratos de desenvolvimento de software com métodos ágeis usando a métrica Ponto de Função.

As mudanças em funcionalidades podem ser decorrentes de mudanças no domínio do negócio - como alteração de escopo, de regras de negócio - ou mudanças legais/regulamentares ou, ainda, refinamentos de requisitos. Considerando os aspectos do desenvolvimento ágil, as mudanças em funcionalidades que ocorrerem após o término da release em que essas funcionalidades ficaram prontas, devem ser tratadas de acordo com o item Projeto de Melhoria deste Roteiro, uma vez que este guia considera que, no desenvolvimento de software com métodos ágeis, o ciclo de trabalho evolutivo em funcionalidades desenvolvidas em uma release encerra-se ao final da release. Assim, como é prática comum existirem mudanças em uma funcionalidade ainda durante a execução das sprints de uma release, este guia sugere que as mudanças em funcionalidades ocorridas dentro dessas características não sejam contadas e, consequentemente, não sejam remuneradas durante a release (ou seja, nos ciclos de pagamento do projeto), mas que já estejam absorvidas pela contratada como parte inerente do processo ágil de desenvolvimento adotado para o projeto.

Nesse sentido, é fundamental que o instrumento convocatório de licitação especifique o máximo de fatores, características e aspectos relevantes do projeto que podem influenciar no volume de mudanças em funcionalidades em um projeto de desenvolvimento com métodos ágeis para que as empresas candidatas ao certame avaliem adequadamente as possibilidades de atendimento do contrato, fornecendo profissionais qualificados e preço de ponto de função exequível para o contrato. Dessa forma, é importante destacar a necessidade do órgão contratante avaliar e controlar a sua gestão de riscos pela adoção de um contrato de desenvolvimento de software com métodos ágeis. O risco poderá se mostrar inversamente proporcional ao detalhamento dos fatores, características e aspectos do projeto expostos no edital de contratação que possam interferir no desenvolvimento e no sucesso do projeto.

7.3.1 Fatores que Influenciam o Número de Mudanças em Funcionalidades no Processo Ágil

Nesta subseção apresentamos alguns fatores que influenciam o número de mudanças em funcionalidades no projeto de desenvolvimento de software com métodos ágeis. Esses fatores podem ser levantados, por exemplo, de uma base histórica de projetos similares do órgão, experiências registradas de projetos com desenvolvimento ágil já realizados pelo órgão e entrevistas com prestadores de serviços de desenvolvimento de software com métodos ágeis.

Como foi dito na seção anterior, dentro de uma *release*, as mudanças em funcionalidades desenvolvidas previamente na mesma *release* não são contadas e remuneradas durante o projeto, pois são absorvidas pela contratada como parte do processo de desenvolvimento ágil. Entretanto, caso essas mudanças ocorram em *releases* diferentes, sugere-se remunerar conforme os itens de manutenção abordados neste Roteiro, tal como, a manutenção evolutiva aplicando-se o fator de impacto sobre o tamanho da funcionalidade impactada, conforme sugerido no item **Projeto de Melhoria** deste Roteiro.

Apesar de não serem contadas em ponto de função, essas mudanças em funcionalidades já desenvolvidas dentro da mesma *release* devem ser registradas e atendidas pelo contrato, mas sem remuneração adicional ao total de pontos de função da contagem detalhada final da *release,* pois se entende que são relativas à evolução de requisitos do processo de desenvolvimento adotado no projeto. Portanto, na contagem detalhada final da *release* não deve haver nem acréscimo de ponto de função nem de qualquer outra natureza.

Alguns fatores devem ser considerados e avaliados para a estimativa do volume de mudanças em funcionalidades em um projeto de desenvolvimento com métodos ágeis:

- maturidade dos requisitos do projeto;

- conhecimento do negócio pelo *product owner;*

- maturidade do processo ágil implantado no órgão (nível de aderência às práticas);

- disponibilidade e experiência com métodos ágeis da área de negócio (*product owner*);

- nível de experiência com métodos ágeis da equipe da contratante (principalmente do *product owner*);

- nível de experiência com métodos ágeis requerido para a equipe de desenvolvimento da contratada;

- tamanho da *sprint* e da *release;*

- volume de mudanças em funcionalidades de projetos similares já executados.

7.3.1.1 Exemplo de Aplicação da Proposta

Para exemplificar a aplicação dessa proposta de tratamento das mudanças em funcionalidades em um projeto de desenvolvimento com métodos ágeis, suponha o planejamento das iterações (*sprints*) de uma *release* (*Release N*) com quatro funcionalidades a serem desenvolvidas (incluídas) e duas funcionalidades prontas em *releases* anteriores para serem alteradas, conforme apresentado na Tabela 11. O tamanho estimado do *backlog* da *Release N* é de 21 PF. Considere que, ao final da *Release N*, as funcionalidades incluídas devem estar prontas para serem remuneradas para a contratada.

Release N (composta de 3 Sprints)	Processo Elementar (PE)	Categoria (Inc, Alt, Exc, Refin)	Tipo (ALI, AIE, EE, CE, SE)	Complex.	PF	Observação
Sprint 1	Incluir Aluno	Inc	EE	Baixa	3	
	Aluno	Inc	ALI	Baixa	7	
	Incluir Disciplina	Alt	EE	Baixa	1,5	Alteração caracterizada como ***Projeto de Melhoria*** (PE "Incluir Disciplina" desenvolvido e pronto na Release N-1). Aplicado o fator de impacto de 50% (3PF*0,5=1,5PF).
Sprint 2	Alterar Aluno	Inc	EE	Baixa	3	
	Alterar Disciplina	Alt	EE	Baixa	1,5	Alteração caracterizada como ***Projeto de Melhoria*** (PE desenvolvido e pronto na Release N-1). Aplicado o fator de impacto de 50% (3PF*0,5=1,5PF).
Sprint 3	Emitir Relatório de Alunos por Disciplina	Inc	SE	Média	5	
Total de PF da *release*					**21**	

Tabela 11 – Planejamento do *Backlog* das *Sprints* da *Release* N

A contagem detalhada de pontos de função da *Release N* está apresentada na Tabela 12. Observa-se que não existe medição para as mudanças em funcionalidades desenvolvidas na mesma *Release N*. Mas, o objetivo principal é mostrar a necessidade de registrar as funcionalidades incluídas, alteradas, excluídas e refinamentos (mudanças em funcionalidades desenvolvidas na mesma *release*) durante a *release*, independente do registro e da identificação da iteração (*sprint*) onde elas ocorreram. Nesse sentido, é facultativo o registro das contagens por *sprint* desde que a contagem da *release* registre as novas funcionalidades desenvolvidas, bem como, as mudanças em funcionalidades.

A Tabela 12 apresenta a contagem de pontos de função e o detalhamento dos serviços realizados na execução da *Release N* e suas *sprints*, incluindo os processos elementares planejados (inclusão e alteração em funcionalidades) e as mudanças em funcionalidades previamente desenvolvidas na mesma *release*, essas categorizadas como "**Refinamento**", que foram identificadas durante o desenvolvimento da *Release N*. Destaca-se, por exemplo, que o processo elementar "*Alterar Disciplina*" foi alterado na *sprint* 2, sendo essa mudança categorizada como "**Alteração**" do Projeto de Melhoria. Em seguida, na *sprint* 3 houve uma nova mudança no mesmo processo elementar "*Alterar Disciplina*", sendo essa mudança categorizada como "**Refinamento**" pois trata-se de uma mudança em funcionalidade já trabalhada na mesma *release*. Além disso, observe que houve a necessidade de alterar o processo elementar "*Emitir Relatório de Disciplinas*" identificada durante o desenvolvimento da funcionalidade planejada "*Emitir Relatório de Alunos por Disciplina*". Essa alteração foi classificada como "**Alteração**" do Projeto de Melhoria, pois a funcionalidade "*Emitir Relatório de Disciplinas*" já estava pronta (ou seja, foi desenvolvida em *release* anterior). Portanto, no desenvolvimento de projetos com métodos ágeis, uma mudança em funcionalidade poderá ser classificada em "**Refinamento**" ou "**Alteração**", a depender se a funcionalidade foi desenvolvida na mesma *release* ou não.

Na Tabela 12, as mudanças em funcionalidades do tipo "Refinamento" foram absorvidas pela contratada e, portanto, não houve remuneração adicional ao total de pontos de função da Release N. Assim sendo, conforme apresentado na Tabela 12, a contagem final da Release N foi de 22,5 PF, que representa o valor a ser remunerado à

contratada e corresponde às funcionalidades incluídas (18 PF) e alteradas (4,5 PF – alterações caracterizadas como Projeto de Melhoria) na Release N. Portanto, considerando o "ciclo de pagamento" adotado, a remuneração da contratada para a *release* corresponderá ao tamanho em pontos de função das funcionalidades incluídas acrescida do valor em pontos de função das mudanças (alterações e exclusões caracterizadas como Projeto de Melhoria) em funcionalidades ocorridas durante a mesma release, excluindo-se os refinamentos (mudanças em funcionalidades desenvolvidas na mesma release) que não são contados e remunerados durante o projeto.

Sugere-se que o registro das mudanças em funcionalidades seja feito em uma planilha de contagem separada aplicando-se as regras de medição sugeridas neste Roteiro. O campo "Categoria" nas Tabelas 11 e 12 mostra, além dos tipos **Inc** (inclusão), **Alt** (alteração) e **Exc** (exclusão) de funcionalidades, o tipo **Refin** (Refinamento) para representar as mudanças em funcionalidades desenvolvidas na mesma *release*.

Release N (composta de 3 Sprints)	Processo Elementar (PE)	Categoria (Inc, Alt, Exc, Refin)	Tipo (ALI, AIE, EE, CE, SE)	Complex.	PF	Observação
Sprint 1	Incluir Aluno	Inc	EE	Baixa	3	
	Aluno	Inc	ALI	Baixa	7	
	Incluir Disciplina	Alt	EE	Baixa	1,5	Alteração caracterizada como ***Projeto de Melhoria*** (PE "Incluir Disciplina" desenvolvido e pronto na Release N-1). Aplicado o fator de impacto de 50% (3PF*0,5=1,5PF).
Sprint 2	Alterar Aluno	Inc	EE	Baixa	3	
	Aluno	Refin	ALI	Baixa	-	Mudança caracterizada como ***refinamento*** de funcionalidade para atender o PE "Alterar Aluno". Sem custo PF.
	Alterar Disciplina	Alt	EE	Baixa	1,5	Alteração caracterizada como ***Projeto de Melhoria*** (PE desenvolvido e pronto na Release N-1). Aplicado o fator de impacto de 50% (3PF*0,5=1,5PF).
Sprint 3	Emitir Relatório de Alunos por Disciplina	Inc	SE	Média	5	
	Incluir Aluno	Refin	EE	Baixa	-	Mudança caracterizada como ***refinamento*** de funcionalidade para atender o PE "Emitir Relatório de Alunos por Disciplina". Sem custo PF.
	Incluir Disciplina	Refin	EE	Baixa	-	Mudança caracterizada como ***refinamento*** de funcionalidade para atender o PE "Emitir Relatório de Alunos por Disciplina". Sem custo PF.
	Alterar Disciplina	Refin	EE	Baixa	-	Mudança caracterizada como ***refinamento*** de funcionalidade para atender o PE "Alterar Disciplina". Sem custo PF.
	Emitir Relatório de Disciplinas	Alt	EE	Baixa	1,5	Alteração caracterizada como ***Projeto de Melhoria*** (PE desenvolvido e pronto na Release N-1). Aplicado o fator de impacto de 50% (3PF*0,5=1,5PF).
Total de PF da *release*					**22,5**	

Tabela 12 – Contagem Detalhada de Pontos de Função da *Release N*

A Tabela 13 apresenta a contagem de pontos de função da *Release N* para a *baseline* da aplicação. Observe que nessa contagem aparecem apenas as funcionalidades incluídas e não devem constar as funcionalidades alteradas, excluídas e refinamentos durante a *Release N*, a menos que tais mudanças tenham impactado na complexidade da função de dados ou transacional. O total de pontos de função da *Release N* para a *baseline* da aplicação é de 18 PF.

	Processo Elementar (PE)	**Tipo (ALI, AIE, EE, CE, SE)**	**Complex.**	**PF**	**Observação**
Contagem da *Release N*	Aluno	ALI	Baixa	7	Na contagem da *release* para a *baseline* da aplicação, não devem constar as funcionalidades alteradas, excluídas e refinamentos.
	Incluir Aluno	EE	Baixa	3	
	Alterar Aluno	EE	Baixa	3	
	Emitir Relatório de Alunos por Disciplina	SE	Média	5	
			Total de PFs da *Release*	**18**	

Tabela 13 – Contagem de PF da *Release N* para *Baseline* da Aplicação

8. Atividades Sem Contagem de Pontos de Função

Deve-se ressaltar que, no processo de desenvolvimento de um projeto de software, há atividades que devem ser consideradas como complementares ou pré-requisitos ao processo de desenvolvimento, de modo que os esforços e produtos entregues devem ser contratados e remunerados em itens distintos do desenvolvimento por não se tratarem de atividades de desenvolvimento do software ou inerentes ao processo desenvolvimento do software. São atividades categorizadas nessa condição:

- Definição de Processo de Desenvolvimento de Soluções: são as demandas para definição de Processos de Software, aderentes às melhores práticas do CMMI e à Instrução Normativa SLTI n° 4, de 12 de novembro de 2010 que devem estar definidos antes da contratação de serviços de desenvolvimento de software.
- Desenvolvimento de Cursos para EaD: são as demandas de elaboração de conteúdo e montagem de material para um curso na modalidade de Ensino a Distância (EaD). Se enquadram no mesmo caso dos Treinamentos citado anteriormente.
- Mapeamento de Processos de Negócio: são as demandas de elaboração de documentação contendo o mapeamento de processos de negócio de uma organização ou de parte de uma organização. Essa é uma atividade que deve ser realizada antes da abertura do projeto de desenvolvimento de software. É importante ressaltar que essa atividade é de responsabilidade dos analistas de negócio da empresa contratante, de acordo com a Instrução Normativa SLTI n° 4, de 12 de novembro de 2010. No entanto, por falta de pessoal, alguns órgãos e entidades têm contratado estas atividades, que antecedem a fase de requisitos – primeira fase do processo de software.
- Treinamentos em Tecnologia da Informação em geral: são as demandas de treinamentos em linguagens de programação, ferramentas de gestão, processos, modelos da qualidade, métricas, etc.

Outras atividades contidas em um processo de software devem ser gerenciadas dentro do projeto de desenvolvimento e são inerentes ao processo de desenvolvimento de software, não devendo ser mensuradas separadamente . São elas:

- Acompanhamento de Projetos: é a atividade que a contratada faz internamente de modo a se organizar e planejar o atendimento dos cronogramas e outras demandas recebidas da contratante, cuja natureza é intrínseca ao desenvolvimento e manutenção de sistemas. Ou seja, ao desenvolver e manter sistemas, a tarefa de acompanhar e gerir o projeto por parte da contratada figuram como seus deveres contratuais, não cabendo pagamento por atividades que dizem respeito à sua própria gestão interna;
- Correção de erros: erros e bugs que venham a se manifestar em ambiente de produção dentro do período de garantia contratado.
- Especificação de Requisitos: em metodologias ágeis, o levantamento de requisitos é inerente ao processo de desenvolvimento de software, não devendo ser mensurado e remunerado separadamente. Em outras metodologias, caso o órgão opte por realizar o levantamento de requisitos separadamente do processo de desenvolvimento de software, esse deve ser remunerado por horas de consultoria.

- Projeto e desenvolvimento de Banco de Dados: as atividades de banco de dados associadas ao projeto de desenvolvimento, modelagem dos bancos seguindo as políticas de dados da contratante, preparação de ambiente (testes, homologação, implantação), desempenhadas pela contratada já devem ser consideradas dentro do projeto de software, não cabendo cobrança adicional.
- Treinamento para Implantação: são demandas de treinamentos sobre utilização do sistema desenvolvido pela contratada a ser implantado, para os gestores de solução do cliente e usuários e devem ser tratadas no escopo da fase de transferência do conhecimento para a contratante.

Finalmente, tendo em vista que já foram identificados casos concretos do uso incorreto do Ponto de Função, cabe reforçar que atividades cuja a natureza difere totalmente do objeto contratado (serviços de desenvolvimento de software) não podem ser remuneradas por pontos por função, são exemplos:

- Deslocamentos e viagens de integrantes da contratada para prestação dos serviços em diferentes localidades;
- Suporte ao Usuário e à Rede no uso do software desenvolvido, principalmente quando englobando atividades como instalação de microcomputadores e demais periféricos.

9. Processo de Revisão do Roteiro de Contagem

9.1 Revisão para Correção de Inconsistências e Situações não Previstas

A revisão deste roteiro será feita sempre que se verificarem inconsistências entre uma definição do CPM e uma regra constante deste documento, e situações não previstas neste roteiro. Essas situações, sempre que necessário, serão documentadas, gerando novas versões deste roteiro.

9.2 Revisão para Adoção de Novas Versões do CPM

A adoção de nova versão do CPM como referência para este roteiro de contagem não será imediata à sua publicação. Nesse caso, deverá haver uma avaliação da nova versão para se decidir sobre a atualização deste documento. Em caso de utilização de roteiro de métricas em contratos de software, a atualização do roteiro deve ser negociada entre órgão contratante e a empresa contratada.

10. Conclusão

Este documento apresentou um roteiro para o dimensionamento de tamanho de vários tipos de projetos de software da contratante, visando a aderência desses tipos de projetos desenvolvidos na instituição às diretrizes da Instrução Normativa SLTI/MP N° 4, de 11 de setembro de 2014. A estimativa de tamanho utiliza a métrica Ponto de Função Não Ajustado como unidade de medida, conforme recomendado nos Acórdãos 1.910/2007, 2.348/2009 e 1.647/2010 do Tribunal de Contas da União (TCU) e na Portaria SLTI/MP Nº 31, de 29 novembro de 2010.

É importante ressaltar que o uso de métricas em contrato de software é uma boa prática, visando proporcionar uma gestão efetiva dos contratos com base em dados quantitativos e objetivos. A implantação desta modalidade de contrato implica na definição de processos de gestão de requisitos e de gestão de projetos baseados nas melhores práticas. Outro ponto a ser destacado é a implantação de um Escritório de Métricas com servidores capacitados para realizar contagens e estimativas em pontos de função. Estes servidores serão responsáveis pela revisão das contagens de pontos de função e estimativas realizadas pelo Escritório de Métricas da empresa contratada e pela manutenção do roteiro de métricas do órgão.

Como trabalho futuro, recomenda-se a revisão e atualização deste roteiro sempre que for verificada inconsistência entre alguma definição do IFPUG publicada em versões futuras do CPM ou em *White Paper*, ou quando for detectado um novo tipo de serviço associado ao desenvolvimento de software não previsto neste roteiro. Neste sentido, como trabalho futuro está programada a elaboração de um modelo de mensuração para serviços de desenvolvimento e manutenção referentes a projetos de DW, Geoprocessamento, *Workflow* e Portais utilizando Gerenciadores de Conteúdo, que representam cenários existentes em alguns órgãos do SISP.

10. Referências Bibliográficas

[Boehm, 2009] BOEHM, B.W. ***Software Cost Estimation With COCOMO II*.** Prentice Hall, New Jersey, 2009.

[CAIXA, 2012] CAIXA. **Guia de Orientação - Métricas**, versão 10, 2012.

[Castro e Hernandes, 2014] CASTRO, M. V. B. de; HERNANDES, C. A. M.. ***A Metric of Software Size as a Tool for IT Governance***. Procedings in: SBES, 2014.

[Dekkers, 2003] DEKKERS, C. “Measuring the logical or functional” Size of Software Projects and Software Application”. Spotlight Software, ISO Bulletin May 2003, pp10-13.

[Hazan, 2005] HAZAN C.; STAA, A.v. **Análise e Melhoria de um Processo de Estimativas de Tamanho de Projetos de Software.** Monografias em Ciências da Computação nº 04/05, Departamento de Informática PUC-Rio, ISSN 0103-9741, Fevereiro 2005.

[Hazan, 2008] HAZAN, C. ***Análise de Pontos de Função: Uma Aplicação nas Estimativas de Tamanho de Projetos de Software***. Engenharia de Software Magazine, Edição 2, Devmedia, pp.25-31.

[Horvath, 2012] HORVATH, D.. ***Function Point Analysis and Agile Methodology****. Q/P Management Group*, Inc. 2012.

[IEEE,1998] IEEE Computer Society. ***IEEE Standard for Software Maintenance***. IEEE Std 1219, 1998.

[IFPUG,2010a] IFPUG. ***Considerations for Counting with Multiple Media***. Release 1.1, April, 2010.

[IFPUG,2010b] IFPUG. ***Counting Practices Manual***. Version 4.3, January, 2010.

[Jones, 2007] JONES, C. ***Estimating Software Costs***. Second Edition, Mc Graw Hill, 2007.

[Keote, 2010] KEOTE, A. K.. ***Function Points and Agile – Hand in Hand***. Accenture, 2010.

[Meli, 1999] MELI, R.; SANTILLO, L. ***Function Point Estimation Methods: A***

Comparative Overview. Proceedings of FESMA 99, Amsterdam, Netherlands, October 1999, pp. 271-286.

[NESMA, 2005] NESMA. Neetherlands Software Metric Association. **The application of Function Point Analysis in the early phases of the application life cycle. A Practical Manual: Theory and case study**, 2005.

[NESMA, 2009] NESMA. ***Function Point Analysis for Software Enhancement Guidelines***. Version 2.2.1, 2009

[Parthasarathy,2007] PARTHASARATHY, M. A. ***Practical Software Estimation: function point methods for insourced and outsourced projects***. Addison Wesley, New York, 2007.

[PROCERGS, 2013] PROCERGS. **Guia de contagem da PROCERGS**. Versão 2.0 – Alterações referentes ao Edital de Fábrica de Software de Sistemas, Atualizado em 13/06/2013.

[Roetzheim, 2005] ROETZHEIM, W. **Estimating and Managing Project Scope for New Development**. CrossTalk, Vol. April, 2005.

[SERPRO, 2008] SERPRO. ***Métodos para Estimativa de Projetos de Software Baseado em Pontos de Função***. Relatório do Grupo de Trabalho para Definição da Utilização de Pontos de Função nos Serviços de Desenvolvimento e Manutenção de Sistemas. 2008.

[Sommerville, 2007] SOMMERVILLE, I. ***Software Engineering***. Pearson Education Limited, 8th Edition, 2007.

[Vazquez, 2012] VAZQUEZ, C. et al. ***Análise de Pontos de Função: Medição, Estimativas e Gerenciamento de Projetos de Software***. 12ª Edição, Editora Érica Ltda, São Paulo, 2012.

Anexo I – Portaria SLTI/MP Nº 31, de 29 novembro de 2010

Dispõe sobre recomendações técnicas para a utilização da métrica Análise de Ponto de Função no âmbito da Administração Pública Federal direta, autárquica e fundacional e dá outras providências.

A SECRETÁRIA DE LOGÍSTICA E TECNOLOGIA DA INFORMAÇÃO DO MINISTÉRIO DO PLANEJAMENTO, ORÇAMENTO E GESTÃO, no uso de suas atribuições que lhe conferem o Decreto nº 7.063, de 13 de janeiro de 2010, o Decreto nº 1.048, de 21 de janeiro de 1994,e o Decreto nº 1.094, de 23 de março de 1994, resolve:

Art. 1º A métrica de Pontos de Função foi concebida como uma medida de tamanho funcional para projetos de desenvolvimento e de melhoria (manutenção evolutiva) de software.

§ 1º A métrica Ponto de Função é definida pelo organismo International Function Point Users Group (IFPUG).

§ 2º O manual de práticas de contagem de Pontos de Função publicado pelo IFPUG define as regras básicas orientativas de contagem de Pontos de Função para projetos de desenvolvimento e melhoria de soluções de software.

§ 3º Por permitir a medição objetiva de serviços de desenvolvimento de soluções de software, sua utilização é uma boa prática na contratação de serviços e está aderente ao estabelecido na Instrução Normativa SLTI nº 4 de 12 de novembro de 2010.

Art. 2º O Roteiro de Métricas de Software do SISP é um documento técnico complementar que visa esclarecer questões técnicas, harmonizar entendimento e abordar assuntos relativos à contratação de soluções de software não contempladas pelo manual de contagem do IFPUG.

Parágrafo Único. Além dos projetos de desenvolvimento de novas soluções de software e de melhoria de software, também há necessidade de medir projetos de manutenção adaptativa de software. Assim, torna-se relevante a definição de procedimentos complementares de medição para dimensionar projetos de manutenção adaptativa de software cuja mensuração não são abordadas pelo manual de prática de contagem do IFPUG.

Art. 3º Recomenda-se que os órgãos integrantes do Sistema de Administração dos Recursos de Informação e Informática (SISP) adotem o roteiro de contagem nas suas contratações de serviços de desenvolvimento e manutenção de soluções de software.

Art. 4º Esta Portaria entra em vigor na data de sua assinatura.

MARIA DA GLÓRIA GUIMARÃES DOS SANTOS

Anexo II – Formalização Simples de Requisitos – Projetos de Manutenção Pequenos (< 100 PF)

Dados Gerais	
Número da OS	
Nome do Sistema Mantido	
Tecnologia Adotada	
Data do Início do Serviço	DD/MM/AAAA
Data do Término do Serviço	DD/MM/AAAA
Descrição da Solicitação	

Descrição do Serviço Executado

Requisito	**Detalhamento**
1.	1.1 1.2...
2.	2.1 2.2...

Identificação da Manutenção

	Tipo
	Melhoria
	Migração de Dados
	Corretiva
	Mudança de Plataforma - Linguagem de Programação
	Mudança de Plataforma - Banco de Dados
	Atualização de Versão – Linguagem de Programação
	Atualização de Versão – Browser
	Atualização de Versão – Banco de Dados
	Manutenção em Interface (Cosmética)
	Adaptação em Funcionalidades sem Alteração de Requisitos Funcionais
	Apuração Especial – Base de Dados
	Apuração Especial – Geração de Relatórios
	Apuração Especial – Reexecução
	Atualização de Dados
	Desenvolvimento, Manutenção e Publicação de Páginas Estáticas de *Intranet, Internet* ou Portal
	Manutenção de Documentação de Sistemas Legados
	Verificação de Erros
	Pontos de Função de Teste (Execução de Testes em funcionalidades não mantidas)
	Componente Interno Reusável

Foi demandada a redocumentação da funcionalidade mantida? ☐ **Sim** ☐ **Não**

Aplicar Fator Criticidade? ☐ **Sim** ☐ **Não**

Observações relevantes quanto ao tipo de manutenção:

Descrição dos Requisitos de Manutenção (para cada funcionalidade alterada, utilizar um quadro)

a) Tabelas Modificadas pela Manutenção

Nome da Tabela	
A Tabela é atualizada por alguma funcionalidade da aplicação: ☐ Sim ☐ Não	
A Tabela é atualizada por outra aplicação: ☐ Sim ☐ Não	
A Tabela foi: ☐ Incluída ☐ Alterada ☐ Excluída	
Total de Campos da Tabela após a Manutenção =	
Campos Incluídos/Alterados/Excluídos	

A funcionalidade será **apenas** testada?

☐ Sim ☐ Não

b) Entradas de Dados Afetadas pela Manutenção (telas ou arquivos de carga)

Nome da Entrada	
Total de Campos na Entrada =	
Nome das Tabelas Acessadas (Lidas e Gravadas) =	
Campos Incluídos/Alterados/Excluídos	

Houve mudança na regra de negócio (validações, lógica de processamento, regras de cálculo)?

☐ Sim ☐ Não

A funcionalidade será **apenas** testada?

☐ Sim ☐ Não

c) Consultas Afetadas pela Manutenção

Considere a tela de parâmetros e a de resultados da consulta como apenas uma única Consulta. Caso a consulta seja do tipo lista e consulta detalhes, considere como funções independentes e preencha quadros diferentes.

Nome da Consulta	
Total de Campos da Consulta	
Tabelas Acessadas	
Total de Campos Afetados =	
Total de Campos Calculados ou Totalizadores =	
Existe atualização de dados (log, indicador...) ☐ Sim ☐ Não	
Campos Incluídos/Alterados/Excluídos	

Houve mudança na regra de negócio (validações, lógica de processamento, regras de cálculo, campos de filtro)?

☐ Sim ☐ Não

A funcionalidade será **apenas** testada?

☐ Sim ☐ Não

d) Relatórios Afetados pela Manutenção

Considere a tela de parâmetros e a de resultados do relatório como apenas um único Relatório.

Nome do Relatório	
Total de Campos no Relatório	
Tabelas Acessadas	
Total de Campos Afetados =	
Total de Campos Calculados ou Totalizadores =	
Existe atualização de dados (log, indicador...) ☐ Sim ☐ Não	
Campos Incluídos/Alterados/Excluídos	

Houve mudança na regra de negócio (validações, lógica de processamento, regras de cálculo, campos de filtro)?

☐ Sim ☐ Não

A funcionalidade será **apenas** testada?

☐ Sim ☐ Não

Anexo III – Modelo de Documento de Contagem de Pontos de Função – Projetos de Manutenção Pequenos (< 100 PF)

Documento de Contagem de Pontos de Função de Projetos de Manutenção Pequenos

Cliente:

Histórico da Revisão

Data	Versão	Descrição	Autor	Aprovador

Nome Projeto:

Número da OS**:**

Responsável pela Contagem:

Descrição da Solicitação de Mudança:

Descrição da Atividade	Contagem PF	Tipo de Manutenção / Total PF

Observações Relevantes:

Conforme a tabela de atividades acima, o total de pontos de função realizados no Projeto _____ na OS _____ é de _____ PF.

Anexo IV - Como Evitar Armadilhas em Contratos de Desenvolvimento e Manutenção de Sistemas

Claudia Hazan

claudia.hazan@serpro.gov.br

Serviço Federal de Processamento de Dados (SERPRO)

Este anexo tem como propósito apresentar algumas dicas para as organizações contratantes evitarem armadilhas em contratos de desenvolvimento e manutenção de software baseados em preço fixo por pontos de função.

Obtenha um Documento de Requisitos de Qualidade

Conforme mencionado, a métrica PF mede a funcionalidade requisitada e recebida pelo usuário. O documento de requisitos constitui um acordo comum entre os órgãos contratantes e empresas contratadas. Assim, é fundamental a existência de um "Termo de Aceite" associado aos documentos de requisitos, assinado pelo gestor do sistema ou gestor do contrato do órgão contratante. Além disso, o contratante deve garantir a qualidade do documento de requisitos encaminhado para a contratada. Observe que se o contratante fornece um documento de requisitos com um requisito incompleto, a empresa contratada entregará o produto sem a funcionalidade esperada e a organização contratante terá que pagar por isso.

A Engenharia de Requisitos apresenta várias técnicas para suportar as atividades de verificação e validação de Documentos de Requisitos, no entanto, estas técnicas são muito custosas. Sugere-se a utilização do método Contagem Estimativa de Pontos de Função, além de apoiar nas estimativas do projeto, o método suporta a detecção de defeitos em documentos de requisitos pelo estimador, enquanto ele está estimando o projeto, sem custo ou esforço adicional, conforme demonstrado por Hazan [Hazan, 2005]. Considerando as revisões e auditorias em contagem de pontos de função dos projetos contratados, é importante que o documento de requisitos e o documento de contagem de PF ou estimativas estejam consistentes.

Estabeleça Regras para o Tratamento das Mudanças de Requisitos

A Engenharia de Requisitos e a indústria reconhecem que os requisitos não permanecem "congelados" até a conclusão do projeto de software. Os requisitos evoluem desde a sua concepção até mesmo após o sistema entrar em produção, devido a diversos fatores descritos por Kotonya [Kotonya, 1998]. Assim, é fundamental que o contrato de software estabeleça cláusulas para tratamento das mudanças de requisitos. É importante ressaltar que o gestor do contrato deve evitar encaminhar para a contratada os requisitos de negócio que estejam em fase de definição, senão poderão emergir muitas mudanças em requisitos elevando o custo do projeto em questão. Recomenda-se a implantação de processos de gerenciamento de projetos e gerenciamento de requisitos pelos contratantes, aderentes às melhores práticas de modelos da Qualidade de Software, visando uma gestão efetiva dos projetos contratados.

Estabeleça Cláusulas de Garantia da Qualidade

Conforme mencionado, a métrica PF considera a funcionalidade requisitada e recebida pelo usuário. Portanto, a remuneração da empresa contratada deve considerar as funcionalidades entregues, somente se estas não apresentarem defeitos. Contudo, o seguinte cenário pode ocorrer: a empresa contratada entrega as funcionalidades requisitadas com defeitos; o gestor do contrato reclama, a empresa contratada corrige os erros da funcionalidade em questão; a contratante recebe o sistema de volta com outros defeitos que surgiram com a correção do erro relatado. Esse tipo de problema é comum em fábricas de software com um processo de testes inexistente ou inadequado. Observe que essa situação pode gerar um grande atraso no recebimento do sistema, podendo gerar atritos entre a área de TI do órgão contratante e os gestores do sistema que estão aguardando a entrega do sistema funcionando. Assim, recomenda-se o estabelecimento de cláusulas contratuais para garantir a entrega de um projeto de desenvolvimento ou manutenção de sistemas com qualidade. Sugere-se incluir no contrato uma cláusula de multa associada à qualidade do produto entregue, considerando o indicador defeitos/PF. Por exemplo, pode-se estabelecer que não é aceitável a entrega de mais de 0,3 defeitos/PF. É importante definir no contrato os tipos de defeitos, a saber: *bugs*, defeitos na documentação, código fonte não estruturado, etc. Pode-se estabelecer também níveis de severidade de defeitos.

Estabeleça Cláusulas Contratuais de Prazo e Taxa de Entrega

Algumas organizações contratantes estabelecem cláusulas contratuais associadas à produtividade. Por exemplo, a empresa contratada deve ter uma produtividade de 15 HH/PF em JAVA. Em alguns casos, o órgão contratante pede para a contratada relatar a taxa de produtividade. Esta prática não é adequada. A produtividade é uma informação estratégica de uma empresa e ela não pode ser obrigada a divulgar estas informações. Além disso, deve-se ressaltar que em um contrato baseado em PF, o controle da produtividade da empresa contratada não faz sentido. De fato, os órgãos contratantes empregam esta prática para resolver o problema de demandas recebidas com atraso de cronograma. A solução é estabelecer no contrato o método de estimativa de prazo a ser utilizado. Recomenda-se que este método utilize o tamanho em PF estimado do sistema na derivação da estimativa de prazo. Além disso, deve-se incluir cláusulas de multa considerando o atraso da entrega do projeto. Para as organizações que não possuem um processo de estimativas definido, sugere-se a utilização da fórmula de Capers Jones descrita em [Jones, 2007]. É importante ressaltar que a fórmula é adequada apenas para projetos maiores que 100 PF. Em relação aos projetos pequenos, o contrato deve fixar prazos de acordo com o tamanho do projeto. Por exemplo, para projetos com até 5 PF o prazo de entrega é de 8 dias úteis.

Outro cenário a ser considerado é o seguinte: a empresa contratada ganha um pregão fornecendo um preço muito baixo por PF e ao ganhar o contrato ela busca forçar o aumento do preço do PF contratado, definindo regras próprias para a contagem de PF. Como os órgãos públicos estão se capacitando em contagem de pontos de função, o gestor do contrato não aceita a contagem de PF majorada. Então, a empresa contratada aloca apenas um recurso para atendimento daquele contrato, ressaltando que os demais recursos estão trabalhando em contratos mais lucrativos. E as demandas de manutenção críticas do contratante ficam pendentes no atendimento. Portanto, visando evitar este problema, é importante definir cláusulas contratuais estabelecendo uma taxa de entrega mínima de PF/mês, por exemplo, 200 PF/mês. Deve-se incluir uma cláusula de multa tratando essa questão. O estabelecimento de uma taxa de entrega mensal máxima e mínima também é importante para a empresa contratada dimensionar suas equipes para um melhor atendimento ao contrato.

Estabeleça o CPM como a Base para as Contagens de PF ao invés de Conversões

Alguns órgãos contratantes estabelecem seus contratos com base na métrica Ponto de Função, no entanto não possuem capacitação adequada em contagem de pontos de função. Em alguns casos, estes órgãos delegam a contagem para a empresa contratada, que estabelece roteiros de contagem com regras que podem majorar a contagem de PF. Algumas vezes, o dimensionamento do tamanho do projeto em PF é realizado por meio de conversões de horas alocadas em pontos de função. Assim, é estabelecido com a empresa contratada um índice de conversão, por exemplo, 8 horas de trabalho corresponde a 1 PF, e então o pagamento da empresa contratada é feito por meio das horas alocadas ao projeto em questão convertidas em PF. Observe que se o recurso de desenvolvimento está em uma empresa contratada externa ao órgão contratante, este não pode gerenciar a quantidade de horas alocada ao projeto. Se o analista da empresa contratada está realizando seu trabalho nas instalações do contratante, isto é um tipo de terceirização de mão de obra. E ainda, se a contratada alocar um recurso com baixa produtividade, o custo do projeto será muito alto. A prática de conversão de horas para PF é simples, no entanto é inadequada. Se o contrato é baseado em pontos de função, a empresa deve realizar as contagens seguindo as regras de contagem do manual CPM.

Deve-se ressaltar que uma contagem de PF errônea pode levar a conseqüências desastrosas, tais como: pagamento incorreto do projeto contratado por PF, geração de dados para indicadores de qualidade – defeitos/PF e produtividade – horas/PF incorretos, geração de estimativas incorretas. É fundamental que as organizações que desejam estabelecer contratos de prestação de serviços de desenvolvimento e manutenção de sistemas com base em pontos de função criem seu Escritório de Métricas com profissionais especialistas em contagem de pontos de função. É recomendado que estes profissionais possuam certificação CFPS (*Certified Function Point Specialist*) e possuam experiência prática em contagem de PF e métodos de estimativas de projetos de software. As empresas contratadas também devem ter seu escritório de métricas para revisar a contagem de PF do órgão contratante. Seguindo estas recomendações é possível evitar ou diminuir a incidência de erros de contagem como os relatados em Hazan [Hazan, 2008].

Estabelecer Regras para Dimensionar Projetos de Manutenção

Muitas organizações estabelecem em seus contratos de prestação de serviços de desenvolvimento e manutenção de sistemas a prestação de serviços de desenvolvimento e manutenção de sistemas com base na métrica Ponto de Função. No entanto, o Manual de Práticas de Contagem (CPM) trata apenas os projetos de desenvolvimento e de manutenção evolutiva. Assim, torna-se importante a definição de métricas para os demais projetos de manutenção. Pode-se contratar tais projetos em homem/hora, no entanto, conforme mencionado a IN 04 preconiza evitar a contratação de serviços de desenvolvimento e manutenção de sistemas por meio da métrica homem_hora. Assim, recomenda-se a utilização de métricas baseadas nas regras de contagem de pontos de função para dimensionar os projetos de manutenção não contemplados no manual CPM.

O primeiro passo é a identificação de todos os tipos de projetos de manutenção que podem ser contratados pela organização. Posteriormente, deve-se definir métricas para dimensionar tais projetos. O Roteiro de Métricas do SISP sugere fórmulas de cálculo.

Estabelecer detalhes do processo de prestação de serviços no Edital

É importante ressaltar que em um processo de contratação de serviços de desenvolvimento e manutenção de sistemas, além da estimativa de tamanho do projeto em pontos de função, outros aspectos devem ser considerados, a saber: definição do processo de desenvolvimento a ser seguido pela contratada com detalhamento dos artefatos a serem entregues, cronograma do projeto, definição dos acordos de nível de serviço, definição com clareza do objeto a ser contratado, considerando os requisitos funcionais e os requisitos não funcionais do projeto. Observe que os requisitos não funcionais (performance, segurança, padrão de interface, etc) não contam pontos de função, no entanto, impactam nas estimativas de esforço, custo e prazo do projeto.

MELHORES PRÁTICAS EM CONTAGEM DE PONTOS DE FUNÇÃO

Em contratos baseados em métricas é fundamental garantir a qualidade da documentação das contagens de pontos de função dos projetos. Seguem algumas melhores práticas a serem consideradas na documentação das contagens e estimativas de pontos de função:

- Documentação da Fronteira da Aplicação

Toda contagem ou estimativa de pontos de função é realizada em uma fronteira da aplicação. Como a definição de fronteira de aplicação é baseada em processos de negócio, esta pode ser subjetiva. Por exemplo, um sistema de capacitação de empregados faz parte ou não da fronteira de um sistema de recursos humanos? Em algumas organizações, um sistema de capacitação pode ser visto como parte do processo de negócio de gestão recursos humanos. Neste caso, trata-se de uma fronteira única de aplicação de recursos humanos, abrangendo o módulo de capacitação. Em outras organizações, a capacitação de empregados pode ser tratada como um processo de negócio independente. Neste caso, tem-se duas fronteiras: sistema de recursos humanos e sistema de capacitação. Desta forma, é fundamental estabelecer e documentar as fronteiras das aplicações. Recomenda-se que as fronteiras dos sistemas a serem mantidas estejam documentadas no edital de contratação, também pode ser documentada no roteiro de métricas do órgão. É importante definir também quais serão as fronteiras das novas aplicações a serem contratadas, que devem estar em conformidade com o Plano Diretor de TI do órgão contratante.

- Documentação dos Requisitos de Projetos de Manutenção

A grande maioria dos projetos de software das organizações é de manutenção de sistemas existentes. Em geral, é comum observarmos documentação de desenvolvimento de novos sistemas. No entanto, a documentação dos projetos de manutenção, muitas vezes, consiste na atualização da documentação da aplicação implantada. Cabe ressaltar que essa prática não é adequada para a contagem de pontos de função, porque na contagem de projetos de manutenção é necessária a identificação das funcionalidades impactadas (incluídas, alteradas ou excluídas). Por exemplo, o envio de uma tela como funcionalidade alterada em um projeto de manutenção, não permite a contagem de PF do projeto, porque esta tela pode trazer funcionalidades de inclusão, alteração, consulta, exclusão e combo boxes. Se a mudança for na lógica de validação de um campo, provavelmente, as funcionalidades impactas são apenas a inclusão e a alteração. Então, a consulta, a exclusão e as combo boxes não devem ser contadas. É fundamental a elaboração de um documento de requisitos específico, mesmo que simplificado, para o

projeto de manutenção. Este documento será a base para a contagem de pontos de função do projeto de manutenção. De fato, os documentos de requisitos da aplicação implantada também deverão ser atualizados.

- **Documentação da Contagem com Rastreabilidade para Requisitos**

Em contratos baseados em pontos de função, as contagens e estimativas de pontos de função devem ser auditáveis. Assim, além de um documento de requisitos com qualidade, é importante que a contagem de pontos de função seja rastreável para os requisitos utilizados como base para a contagem. Desta forma, recomenda-se que as planilhas de contagem possuam uma coluna denominada "requisito/observações/justificativa". Ao lado de cada tipo funcional identificado deve-se documentar qual o requisito de origem e, caso necessário, as observações e justificativas da contagem, por exemplo:

Identificação da Função	Tipo	....	requisito/observações/justificativa
Relatório de Treinamentos Ministrados por período	SE	...	Caso de Uso 5 – Foi contado como SE por conta da totalização dos cursos.
....	....	....	....

- **Documentação das Funcionalidades Identificadas**

Algumas padronizações na nomenclatura das funções identificadas podem ser adotadas para apoiar as revisões das contagens. Por exemplo: para as funções de dados, utilizar o nome das entidades do modelo de dados e no campo observação pode ser colocado o nome do arquivo físico implementado; para as funções transacionais, colocar o nome da funcionalidade considerando o documento de requisitos utilizado na contagem.

Além disso, também é recomendável a documentação dos Registros Lógicos e Arquivos Referenciados, a documentação pode ser registrada como comentário na coluna correspondente da planilha de contagem. A documentação dos Tipos de Dados pode ser muito trabalhosa, no entanto esta também influencia na complexidade do tipo funcional. Então cabe o bem senso, deve-se documentar os Tipos de Dados até a complexidade máxima do tipo funcional. Por exemplo, suponha que a funcionalidade - EE: Vender Produto possua 3 Arquivos Referenciados – Vendas, Produto e Vendedor. Esta EE com 5 Tipos de Dados já será contada como EE de complexidade alta. Então, basta documentar até 5 TD.

Cada órgão deve definir o modelo do seu documento de contagem de pontos de função, assim como as instruções para documentação das funcionalidades, visando tornar as contagens auditáveis.

REFERÊNCIAS BIBLIOGRÁFICAS

[Dekkers, 2003] DEKKERS, C. *Measuring the "logical" or "functional" Size of Software Projects and Software Application*. Spotlight Software, ISO Bulletin May 2003 pp10-13.

[Hazan, 2005] HAZAN, C.; BERRY, D.M.; LEITE, J.S.P. *É possível substituir processos de Engenharia de Requisitos por Contagem de Pontos de Função?* 8th International Workshop on Requirements Engineering (WER2005), Porto, Portugal, June 2005, pp. 197-208.

[Hazan, 2008] HAZAN, C. *How to Avoid Traps in Contracts for Software Factory Based on Function Point Metric*. 3rd International Software Measurement & Analysis Conference, Washington, United States, 2008.

[IFPUG, 2010] IFPUG. *Counting Practices Manual*. Version 4.3, January, 2010.

[Jones, 2007] JONES,C. *Estimating Software Costs – Bringing Realism to Estimating*. 2nd Edition, Mc Graw Hill, New York, 2007. New York.

[Kotonya, 1998] KOTONYA, G.; SOMMERVILLE, I. *Requirements Engineering: Processes and Techniques*. John Willey & Sons Ltd, 1998.

Anexo V – Resumo da Técnica EFP (*Enhancement Function Points*) publicada pela NESMA

Este anexo apresenta um resumo das fórmulas para o cálculo em projetos de melhoria, dos pontos de função incluídos, alterados e excluídos, segundo a técnica EFP estabelecida em [NESMA, 2009]:

PF_INCLUIDO = QPI;

PF_EXCLUIDO = QPE x 0.40;

PF_FD_ALTERADO = FD-QPA x FFD, sendo FFD entre 0,25 e 1,00, conforme tabela abaixo (para funções de dados);

PF_FT_ALTERADO = FT-QPA x FFT, sendo FFT entre 0,25 e 1,50, conforme tabela abaixo (para funções transacionais);

PF_ALTERADO = PF_FD_ALTERADO + PF_FT_ALTERADO.

Onde:

QPI = Quantidade de Pontos de Função incluídos;

QPE = Quantidade de Pontos de Função excluídos;

PF_FD_ALTERADO = Pontos de Função alterados para Funções de Dados;

PF_FT_ALTERADO = Pontos de Função alterados para Funções Transacionais;

FD-QPA = Quantidade de Pontos de Função alterados em funções de dados;

FT-QPA = Quantidade de Pontos de Função alterados em funções transacionais;

FFD = Fator de impacto de alterações em funções de dados;

FFT = Fator de impacto de alterações em funções transacionais.

O cálculo dos fatores de impacto são obtidos através dos percentuais de alterações conforme abaixo:

a) Funções de Dados:

Percentual de alterações em funções de dados: PFD = QTDA / QTDT x 100;

QTDA = Quantidade de Tipos de Dados Alterados;

QTDT = Quantidade de Tipos de Dados Totais na Função Original.

Com o valor obtido em PFD, procura-se na tabela qual o valor do fator de impacto:

Fator de Impacto em Funções de Dados Alteradas (FFD)				
PFD	Entre 0 e 33%	Entre 33% e 66%	Entre 66% e 100%	Maior que 100%
Fator de Impacto (FFD)	0,25	0,5	0,75	1

b) Funções Transacionais:

Percentual alterações em funções transacionais (PFT1) = QTDA / QTDT x 100;

Percentual alterações em funções transacionais (PFT2) = QFTRA / QFTRT x 100;

QTDA = Quantidade de Tipos de Dados Alterados;

QTDT = Quantidade de Tipos de Dados Totais na Função Original;

QARA = Quantidade de Arquivos Referenciados Alterados;

QART = Quantidade de Arquivos Referenciados Totais na Função Original.

Fator de Impacto em Funções Transacionais Alteradas (FFT)			
PFT2 / PFT1	Entre 0% e 66%	Entre 66% e 100%	Maior que 100%
Entre 0% e 33%	0,25	0,5	0,75
Entre 33% e 66%	0,5	0,75	1
Entre 66% e 100%	0,75	1	1,25
Maior que 100%	1	1,25	1,5

Caso FFT seja maior que 1, recomenda-se reconsiderar a melhoria.

Anexo VI – Notas Técnicas das Versões Anteriores deste Roteiro

Nota Técnica da Versão 1.0

A versão 1.0 do documento foi construída baseando-se no “Roteiro SERPRO de Métricas para Contratos de Software”, no Manual de Práticas de Contagem (CPM); nas publicações do *International Function Point Users Group* (IFPUG) e da *Netherlands Software Metrics Association* (NESMA); e na literatura de métricas e de engenharia de software. Também foram consultados outros roteiros de órgãos públicos que já utilizavam a métrica Ponto de Função em contratos de software.

A proposta inicial do roteiro foi submetida ao Grupo de Trabalho de Métricas do SISP, o qual contribuiu com sugestões de melhoria. Em seguida, as propostas foram analisadas em reunião com especialistas em métricas de órgãos e entidades do SISP para o fechamento e publicação da versão 1.0 do roteiro em novembro de 2010.

Equipe de Elaboração da Versão 1.0

Carlos Renato dos Santos Ramos – SLTI/MP

Claudia Hazan – SERPRO

Claudio Muniz Machado Cavalcanti – SLTI/MP

Emanuelle Monteiro Silva – SLTI/MP

Fátima Saldanha Cesarino – CEF

Gileno Dias dos Santos – SLTI/MP

Jose Romildo Araujo de Andrade – SLTI/MP

Lucinéia Turnes – SLTI/MP

Marcelo Paiva Fernandes – SLTI/MP

Maurício Koki Matsutani – DATAPREV

Patrícia Oliveira de Carvalho – SUSEP

Rafael Campos – SLTI/MP

Rafael Monteiro dos Santos Escolástico – MEC

Regiane Andrade Brito – BACEN

Rosa Maria da Costa Medeiros – INEP

Nota Técnica da Versão 2.0

A versão 2.0 do Roteiro de Métricas de Software do SISP foi elaborada com base em propostas de melhoria da sua versão anterior (versão 1.0), enviadas pelos órgãos do SISP, e em roteiros de métricas de órgãos que já utilizavam a métrica PF em contratos de desenvolvimento e manutenção de software, além de contar com sugestões obtidas do Grupo de Trabalho de Métricas do SISP e de discussões com um grupo de especialistas em métricas de órgãos e entidades do SISP.

A publicação da versão 2.0 do roteiro ocorreu em setembro de 2012.

Equipe de Elaboração da Versão 2.0

Carlos Alberto Pires de Castro – DATAPREV

Carlos Renato dos Santos Ramos – Câmara dos Deputados

Claudia Hazan – SERPRO

Edviges Mariza Campos de Magalhães – DATAPREV

Emanuelle Monteiro Silva – SLTI/MP

Gileno Dias dos Santos – SLTI/MP

Inerves José dos Santos Filho – STN

Lucinéia Turnes – SLTI/MP

Marcelo Paiva Fernandes – SLTI/MP

Marcelo Teixeira Duarte – DATAPREV

Marcus Vinícius Borela de Castro – TCU

Maurício Koki Matsutani – DATAPREV

Patrícia Oliveira de Carvalho – SUSEP

Rafael Monteiro dos Santos Escolástico – AGU

Regiane Andrade Brito – BACEN

Silvia Viviane de Souza Belarmino – IPHAN

Nota Técnica da Versão 2.1

A versão 2.1 apresenta uma pequena variação com relação à versão anterior 2.0:

- alteração no fator de impacto de 40% para 30% para as funcionalidades excluídas de um Projeto de Melhoria (seção 4.2).
- ajuste no tratamento e contagem em pontos de função para o item denominado Componente Interno Reusável, apresentado na seção 4.15 deste documento.
- correção dos percentuais da Tabela 10 referentes ao retrabalho decorrente de mudanças de requisitos durante o desenvolvimento ou manutenção do software. Consequentemente, ainda na subseção 6.2.1 – *Considerações sobre Mudanças de*

Requisitos foram atualizados os exemplos que mostram a aplicação desses percentuais da Tabela 10.

• criação de um novo capítulo (Capítulo 7) que aborda o tratamento das mudanças em funcionalidades em um projeto de desenvolvimento usando métodos ágeis, principalmente, para o cenário de contratações de software usando a métrica Ponto de Função.

Equipe de Elaboração da Versão 2.1

Cinthya Hiromi Seko de Oliveira – SERPRO
Claudia Hazan - SERPRO
Daniella Elizabeth Carneiro - SERPRO
George Atsushi Murakami - TCU
Igor de Mesquita Barbosa - CGU
Jose Romildo Araujo de Andrade – DTI/SE/MP
Gileno Dias dos Santos – DTI/SE/MP
Kátia Cristina Barbosa Loschi de Melo - SERPRO
Lucineia Turnes – SLTI/MP
Marcia Regina Guiotti Bomfim - MPT
Marcus Vinicius Borela de Castro - TCU
Tadeu Rocha - CGU
Valeria Maria Siqueira Bezerra – SLTI/MP