LUIS ALBERTO TCHAKERIAN

ANÁLISE COMPARATIVA DOS MÉTODOS COSMIC-FFP E APF DE MENSURAÇÃO FUNCIONAL DE SOFTWARE

Monografia apresentada à Escola
Politécnica da Universidade de
São Paulo para conclusão do curso de MBA
em Engenharia de Software

São Paulo
2004

LUIS ALBERTO TCHAKERIAN

ANÁLISE COMPARATIVA DOS MÉTODOS COSMIC-FFP E APF DE MENSURAÇÃO FUNCIONAL DE SOFTWARE

Monografia apresentada à Escola
Politécnica da Universidade de
São Paulo para conclusão do curso de MBA
em Engenharia de Software

Área de Concentração:
Engenharia de Software

Orientador:
Prof. Sérgio Martins Fernandes, Mestre

São Paulo
2004

FICHA CATALOGRÁFICA

RESUMO

O presente trabalho realiza uma análise da comparação dos métodos de mensuração funcional de software Análise de Ponto de Função e COSMIC-FFP.

Primeiramente é realizada uma descrição de cada método, abrangendo seus conceitos e processos, para em seguida confrontá-los.

É efetuada então a análise dos resultados deste confronto, e assim produzida uma conclusão sobre a aplicabilidade destes métodos de mensuração funcional de software.

ABSTRACT

This report presents an analysis of the comparison of Functional Size Measurement methods Function Point Analysis and COSMIC-FFP.

First each method is described, enclosing its concepts and processes, for after that compare them.

After this, is performed an analysis of the results of this comparison, and thus produced a conclusion on the applicability of these methods of functional size measurement.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

Tabela 3.8 – Componentes Funcionais Básicos do COSMIC-FFP e APF

Tabela 3.9 – Contribuição dos Componentes Funcionais Básicos por complexidade do COSMIC-FFP e APF

LISTA DE ABREVIATURAS E SIGLAS

APF	Análise de Ponto de Função
COSMIC	Consórcio Internacional Comum de Mensuração de Software
IFPUG	Grupo Internacional de Usuários de Ponto de Função
UKSMA	Associação de Métricas do Reino Unido
FFP	Pontos de Função Completos
ISO	Organização Internacional de Padrões
RFU	Requisito Funcional de Usuário
E/S	Entrada e Saída
UTFC	Unidade de Tamanho Funcional do COSMIC
ALI	Arquivos Lógicos Internos
AIE	Arquivos de Interface Externos
TED	Tipos de Elementos de Dados
TER	Tipo de Elemento de Registro
TAR	Tipo de Arquivo Referenciado
EE	Entrada Externa
SE	Saída Externa
CE	Consulta Externa
VFA	Valor do Fator de Ajuste
CGS	Característica Geral de Sistema
CFB	Componente Funcional Básico
COCOMO	COnstructive COst MOdel

1 INTRODUÇÃO

Analogia com a Construção Civil

Imaginemos que uma pessoa quer construir uma casa. Essa pessoa já tem uma idéia de como deve ser a casa, três quartos, dois banheiros, uma sala de estar, uma sala de jantar, garagem para dois carros e área de serviço. Esta pessoa contrata um arquiteto que vai ajudá-la a organizar e melhor descrever essa idéia de casa, resultando uma planta, já com a definição das metragens e disposição dos cômodos.

Com base nesta planta definida, um engenheiro civil pode estimar o prazo e custo desta obra com margem de erro pequena utilizando tabelas pré-definidas amplamente difundidas no mercado.

Com projetos de software tem-se uma situação semelhante, o usuário tem uma idéia do que ele quer, um profissional organiza e descreve melhor estes requisitos, mas o processo de estimar custo e prazo ainda não é trivial como na construção civil.

Importância da Mensuração

Os projetos de desenvolvimento e de manutenção de software representam aproximadamente 1% da economia do mundo nos dias de hoje. Se estimativas, realizadas pelo Standish Group (Standish Group 1999), mostram que apenas 16% dos projetos de software são concluídos no devido tempo e custo, o impacto dos 84% de projetos de software restantes é profundo.

Até agora nenhum dos métodos de mensuração de software é amplamente aceito para mensurar o resultado da indústria de software. Nós podemos mensurar o custo e quantas pessoas fazem parte desta indústria, mas não conseguimos mensurar o quanto é produzido. É como se a indústria automobilística conseguisse mensurar o

gasto de material e esforço, mas não conseguisse contar o número de carros produzidos.

Existe uma relação direta entre o sucesso de um projeto e a qualidade do gerenciamento deste. As estimativas de custo e esforço são um importante aspecto da gerência de projetos de desenvolvimento de software. A maioria dos métodos de estimativa de esforço requerem uma estimativa do tamanho do software. Além disso, o tamanho do software pode ser um indicador da viabilidade do projeto.

Existem várias meios de se mensurar um software, sendo que os mais populares são a mensuração do tamanho físico do software, que é basicamente uma contagem de linhas de código, e a mensuração do tamanho funcional do software, que se baseia na funcionalidade provida ao usuário.

Métodos de contagem de linhas de código, como COCOMO, foram amplamente utilizados na indústria de software para estimar produtividade e qualidade. Entretanto vários usuários e pesquisadores destes métodos contestam que estes métodos falham em várias situações. Por exemplo, linguagens de alto nível acabam contendo menos linhas de código do que linguagens de baixo nível para o mesmo programa e, este tipo de contagem acaba não privilegiando código otimizado e bem estruturado. Outra desvantagem destes métodos é que no começo do projeto o número de linhas de código é desconhecido então não é possível que estes métodos sirvam como uma estimativa de tamanho.

Hoje em dia são mais difundidos os métodos de mensuração do tamanho funcional de software, como Análise de Ponto de Função (APF), MarkII, Pontos de Função 3-D, e o recente COSMIC-FFP.

Dentre estes o método APF é um dos mais conhecidos e utilizados métodos para mensuração de tamanho funcional de software hoje em dia. O recente método COSMIC-FFP tras novos conceitos de mensuração, e é até classificado por alguns autores como a segunda geração de mensuração funcional de software. Este método

se propõe a mensurar qualquer tipo de software, seja ele de negócio ou não. Por esses motivos selecionamos estes dois métodos para neste texto realizarmos uma análise comparativa destes dois, de um lado o mais utilizado e de outro o que promete revolucionar este segmento.

Histórico

O método de Ponto de Função foi desenvolvido por Allan Albrecht nos anos 70 enquanto trabalhava para a IBM, com o intuito de mensurar a quantidade de software produzido. Sua intenção era de medir o tamanho de funcionalidades do ponto de vista do usuário, independente da implementação. Foi apresentada ao publico pela primeira vez em 1979 uma como uma métrica de tamanho funcional de software.

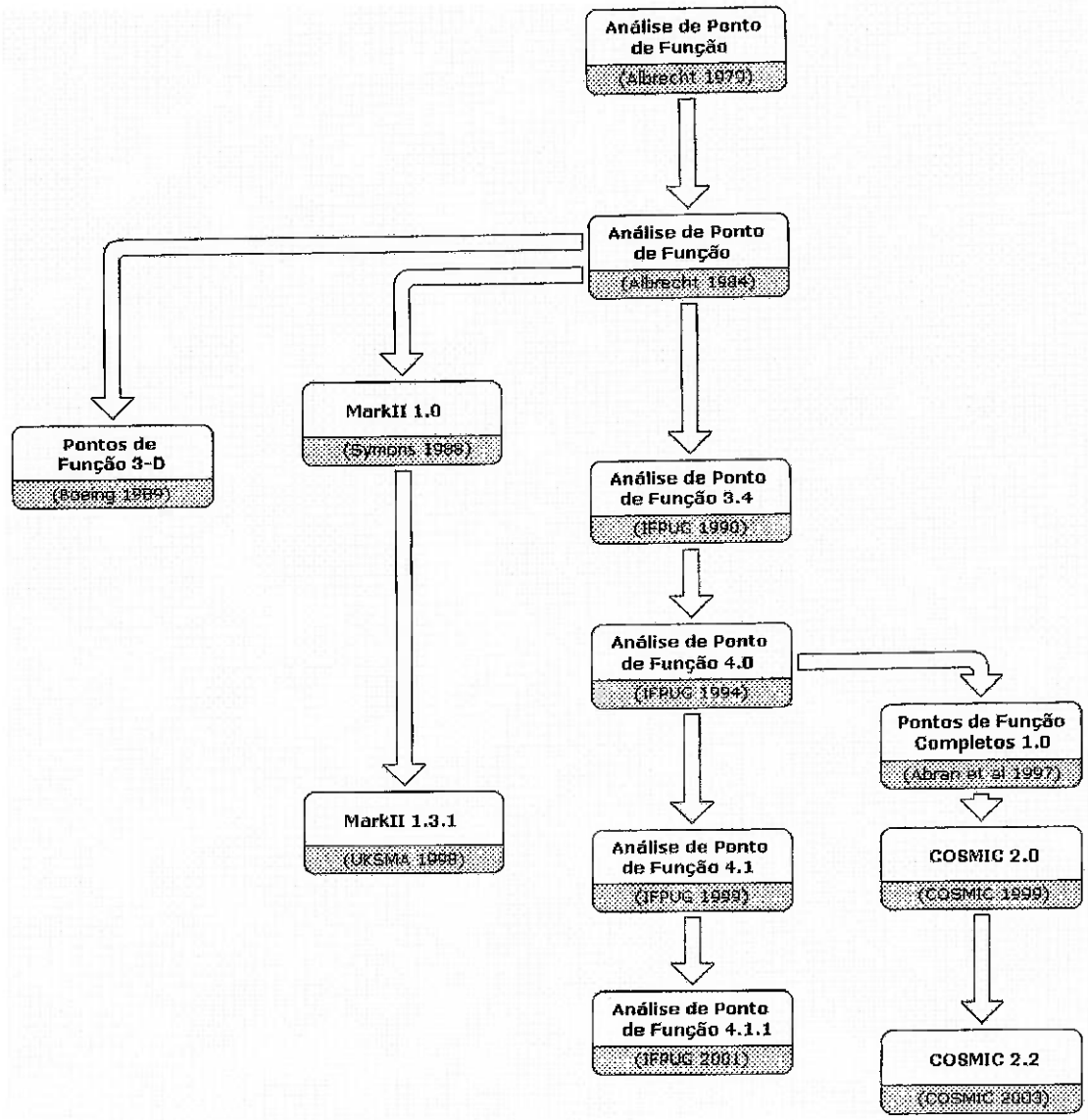

Figura 1.1 - Histórico dos Métodos de Mensuração.

Em 1986 foi formado o Grupo Internacional de Usuários de Ponto de Função (IFPUG) para manter o desenvolvimento do método APF de Allan Albrecht, e desde então são publicadas periodicamente novas versões de manuais de práticas de contagem. O IFPUG vem ao longo do tempo melhorando e também alterando as regras do método original de Albrecht.

Ao longo do tempo foram surgindo várias extensões e variações deste método para atender suas falhas.

Charles Symons publicou em (Symons, 1988) várias criticas ao método APF o que o levou a criação de um método derivado do APF intitulado Análise de Ponto de Função Mark II. Este método, comumente conhecido como MarkII, hoje é mantido pela Associação de Métricas do Reino Unido (UKSMA) e foi baseado na versão de 1984 da APF.

Também Scott Whitmire da Boeing, tendo como base a versão de 1984 do método APF, desenvolveu o método Pontos de Função 3-D entre 1989 e 1992, com foco na mensuração de software cientifico, de engenharia e tempo-real. Este método ainda é utilizado na Boeing, mas é pouco utilizado fora desta.

O método APF é comumente aplicado ao fim da fase de especificação de requisitos. Este método foi desenvolvido e otimizado para software de negócios e é hoje amplamente utilizado em qualquer software de gerenciamento de informação. Entretanto o método APF não atingiu a mesma aceitação do mercado para software de tempo real, pois, segundo Alain Abran, neste tipo de software a armazenagem de dados é geralmente simples e a manipulação destes dados é complexa. Sendo a armazenagem de dados um dos fatores que contribuem para o tamanho do software neste método a mensuração deste tipo de software acaba sendo subestimada.

Para suprir esta lacuna do método APF, é proposto em 1997 por Alain Abran uma extensão deste método, chamada de Pontos de Função Completos (FFP), para a mensuração do software que não se encaixava no modelo de software de negócio. Depois de alguns anos de desenvolvimento e testes de campo chegou-se a conclusão que este método também servia para mensurar não apenas o software que não se encaixasse no modelo de software de negócios, mas também para aquele que se encaixasse nesse modelo. Então em 1999 é formado o Consórcio Internacional Comum de Mensuração de Software (COSMIC) e publicada a versão 2.0 do Manual de Práticas de Contagem já como um método completo e independente para a mensuração de tamanho funcional de software, que daí em diante é chamado de método COSMIC-FFP.

2 MÉTODOS DE MENSURAÇÃO DE SOFTWARE

Este capítulo descreve uma visão geral dos métodos de mensuração de software: COSMIC-FFP e Análise de Ponto de Função (APF).

2.1 COSMIC-FFP

Introdução

Em 1997 Alain Abran e outros do Laboratório de Pesquisas de Engenharia de Software da Universidade de Québec publicaram a primeira versão do Manual de Contagem de FFP (Alain Abran et al. 1997) como uma extensão do método Análise de Ponto de Função da IFPUG para mensurar software de tempo real.

A segunda versão deste Manual foi publicada já pelo COSMIC em 1999 (COSMIC, 1999), que desde esta data concentra o desenvolvimento deste método.

Este método foi aceito em 2000 pelo ISO como um novo item de trabalho para padronização e em dezembro de 2002 foi publicada a norma ISO/IEC 19761 "Software Engineering – COSMIC-FFP – A functional size measurement method".

Ciclo de Vida

O processo de mensuração do método COSMIC-FFP consiste da aplicação de modelos, regras e procedimentos em uma especificação funcional de requisitos de um software, que terá como resultado um valor que representará o tamanho funcional deste. O processo é dividido em duas fases principais: a primeira consiste em mapear o software a ser mensurado para o modelo de software genérico do COSMIC-FFP; e a segunda fase, em mensurar certos aspectos deste modelo.

A entrada necessária para que a mensuração seja realizada é a especificação funcional de requisitos no modelo de software genérico do COSMIC-FFP. Esta é o resultado da fase de mapeamento, onde a especificação funcional de requisitos será obtida dos artefatos de software existentes, sejam estes expressos como uma especificação de requisitos, casos de uso, ou qualquer outra forma. Esta

transformação tem como principal finalidade excluir aspectos técnicos e de qualidade implícitos nos requisitos.

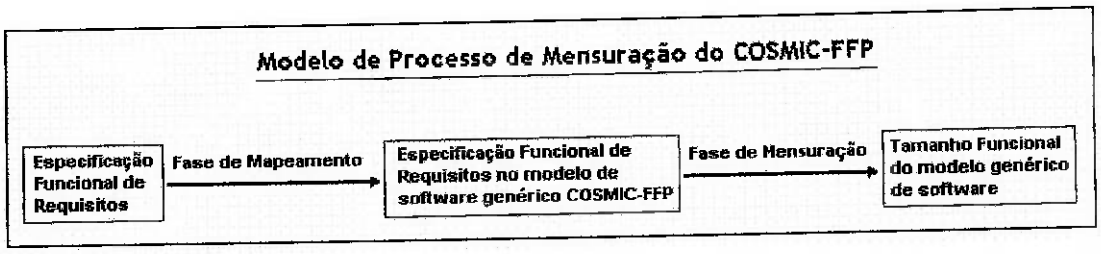

Figura 2.1.1 – Modelo de Processo de Mensuração do COSMIC-FFP

Descreveremos abaixo a fase de Mapeamento e em seguida a fase de Mensuração deste método. Cada fase descrita será dividida, com finalidade de melhorar a didática, em conceitos e processo.

Fase de Mapeamento

Conceitos

A seguir descreveremos conceitos fundamentais do COSMIC-FFP. Esses conceitos são essenciais para a compreensão deste.

Camada

Segundo COSMIC (2003), camada é o resultado do particionamento funcional do software, sendo que todos processos funcionais incluídos nesta executam no mesmo nível de abstração.

Limite

Limite é definido como a interface entre o software estudado e seus usuários.
O software a ser mensurado pode ser facilmente separado de outro software com o qual este troca dados por um Limite. Existe um Limite implícito entre cada par de software identificados que trocam dados.

O Limite torna clara a distinção entre os itens pertencentes ao software, a parte de dentro, e os pertencentes ao software do ambiente, a parte de fora. Por convenção todos os usuários ficam do lado de fora do Limite do software.

Usuários do Software

É possível identificar um ou mais usuários utilizando as funcionalidades providas por um pedaço de software em uma camada. Usuários podem ser seres humanos, outros dispositivos ou mesmo outro software. Conseqüentemente, pedaços de software vizinhos de camadas são considerados como usuários quando interagem com o software a ser mensurado.

Requisito Funcional de Usuário (RFU)

As partes dos requisitos de software que descrevem a natureza das funções a serem desempenhadas são designadas como os Requisitos Funcionais de Usuário (RFUs). São incluídas apenas as funcionalidades que são relevantes para a mensuração funcional do software. Os requisitos que descrevem como serão implementadas as funcionalidades, como requisitos técnicos e qualitativos, devem ser excluídos.

Abstração

Segundo (ISO/IEC 10746-2), é o processo de suprimir detalhes irrelevantes para obter um modelo simplificado do processo ou seu resultado.

Ponto de Vista

Segundo (ISO/IEC 10746-2) ponto de vista é uma forma de abstração que visa focar certos aspectos de um sistema, utilizando um conjunto de conceitos arquiteturais e regras estruturação.

Ponto de Vista do Desenvolvedor

É o ponto de vista que revela toda a funcionalidade de cada parte do software que deve ser desenvolvido para atender as particularidades de um RFU especifico. Neste ponto de vista, o Usuário é um ser humano.

Ponto de Vista do Usuário

É o ponto de vista que revela apenas as funcionalidades do software, excluindo toda a funcionalidade de qualquer outro software que faz a interação entre o Usuário e o software que esta sendo mensurado.

Requisito Técnico

Segundo Ambler (2003) este tipo de requisito é um aspecto técnico que o sistema deve atender. Enquanto o requisito funcional diz o que fazer, o requisito técnico diz como fazer. São exemplos de requisitos técnicos: estipular tempos mínimos de retorno de consulta, definir linguagens de programação e qual banco de dados deve ser utilizado.

Modelo de Contexto de Software

É um modelo utilizado para auxiliar na identificação do que faz parte ou não do software a ser mensurado.

Processo

A fase de mapeamento tem como entrada uma descrição dos Requisitos Funcionais do Usuário de um software, e consiste na aplicação de certas regras e procedimentos definidos para que se obtenha um modelo especifico de software, o modelo genérico de software do COSMIC-FFP.

Para se obter a entrada para a fase de Mapeamento deve-se levar em conta que o aspecto de software mais relevante para o método COSMIC-FFP é a funcionalidade, ou seja, o processamento de informação que o software executará para o usuário. As funcionalidades do software devem estar descritas nos Requisitos Funcionais dos Usuários (RFU). Este deve descrever somente cada funcionalidade do software, e não incluir qualquer requisito técnico ou de qualidade. Na prática, às vezes os RFUs existem na forma de um documento especifico, como uma especificação de requisitos, mas freqüentemente têm de ser obtidos a partir de outros artefatos de software. O fato de que os RFUs podem ser obtidos de artefatos de software que estão disponíveis antes do software ser desenvolvido nos leva à conclusão de que se pode mensurar o tamanho funcional de um software antes deste ser implementado.

Há também o caso de que um software pode ser desenvolvido com poucos ou nenhum artefato de software especificado, não havendo maneira de descrever os RFUs, como, por exemplo, em um sistema legado. Nesses casos há a possibilidade de se descrever os RFUs a partir da derivação de artefatos de software contidos no sistema, como o código do software, o manual do software e o banco de dados..

O esforço para extrair os RFUs dos diferentes tipos de artefatos de software e transformá-los na forma de modelo genérico de software da COSMIC-FFP deverá variar, mas o conteúdo dos RFUs devem conter apenas a definição de funcionalidade do software para o usuário.

Obtida então a entrada deve-se identificar em qual Modelo de Contexto de Software o software a ser mensurado se encaixa.

Modelo de Contexto de Software

Um fator importante para a definição deste modelo é a definição do que é considerado parte do software e o que é parte do sistema operacional, ou seja, software do ambiente.

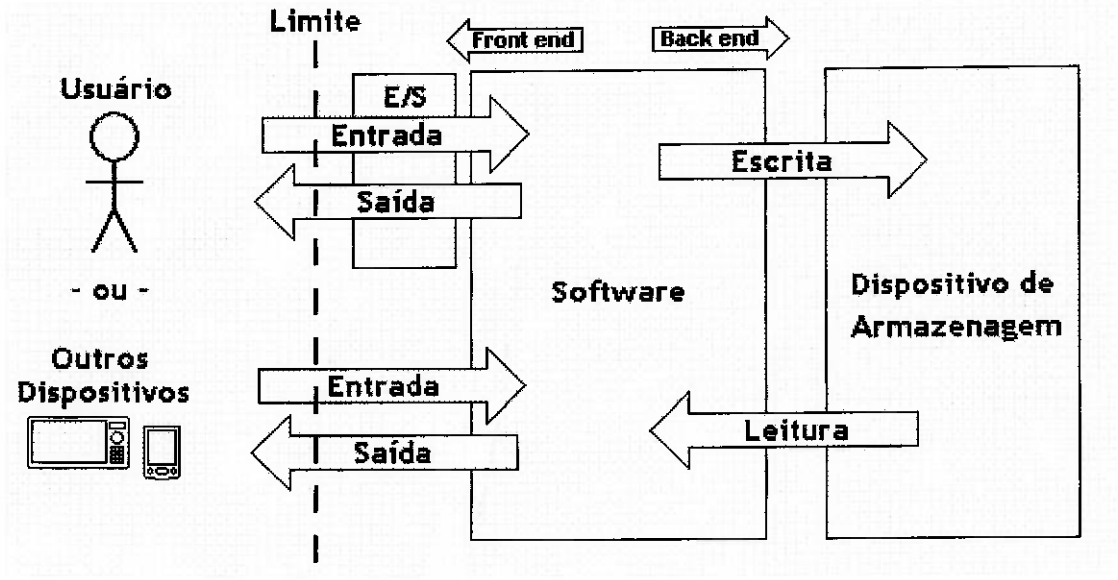

Figura 2.1.2 – Modelo Genérico de Fluxo de Dados

O Software é cercado pelo Hardware, gerando duas fronteiras, o front end e o back end. O front end é a fronteira com o usuário, por meio de dispositivos de entrada, como mouse, teclado, ou outro dispositivo que faz interface com este software. O Back end, é a fronteira com dispositivos de armazenagem, como discos rígidos e memórias voláteis.

O Fluxo de dados é definido entre quatro tipos quanto ao movimento:

- no front end, que faz a troca de informação entre o software e o usuário (ou outro dispositivo):
    1. Entrada
    2. Saída
- e no back end, que faz a troca de informações do software com os dispositivos de armazenagem:
    3. Leitura
    4. Escrita.

Dependendo do tipo do software, diferentes tipos de interação são assumidas. Por exemplo, se é um software de negócio, geralmente quem interagirá com este serão

um ou mais usuários e serão desprezadas a entrada e saída do hardware (E/S). Já se for um software de tempo real provavelmente será outro dispositivo que interagirá com este.

Limite é definido como a interface entre o software estudado e seus usuários.
Como o COSMIC-FFP é um método de mensuração de tamanho funcional de software, sua mensuração é baseada nos RFUs identificados. Assim que identificados os RFUs são divididos em requisitos de software e de hardware. Como a intenção é mensurar o software, apenas os requisitos pertencentes a este serão levados em conta.

Software, pela perspectiva da mensuração, pode ser dividido em dois casos principais: típico software de negócio e software multi-componente. Abaixo iremos analisar estes dois casos.

Típico Software de Negócio

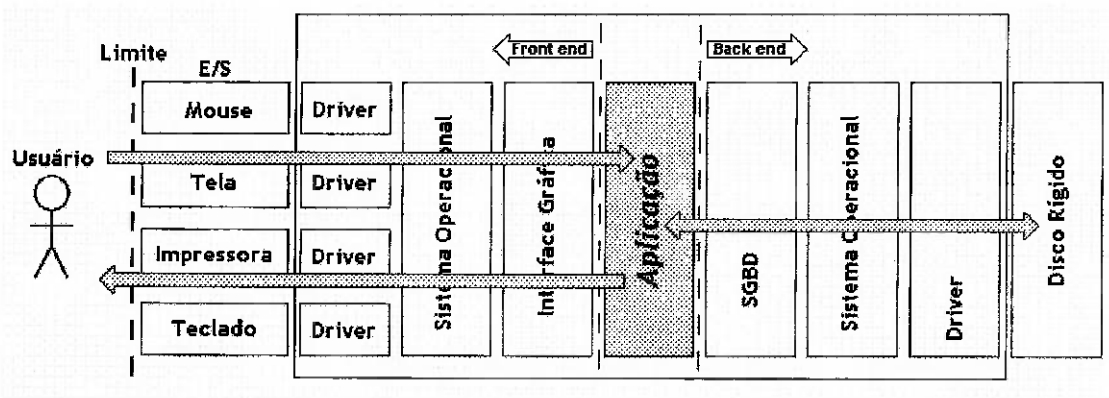

Figura 2.1.3.1 – Típico Software de Negócio

Neste caso os RFUs estão principalmente residentes no nível da aplicação e os outros software no ambiente são utilizados sem nenhuma modificação.

Desde que todos os RFUs estejam alocados na mesma parte do software e que esta seja facilmente distinguida do resto do software do ambiente, é simplesmente o caso

de identificar e mensurar a funcionalidade. No exemplo apresentado acima e para o propósito da mensuração funcional, uma abstração pode ser feita excluindo o hardware de E/S e todo o software do ambiente, levando em conta apenas o software do aplicativo. O Limite então fica definido como a interface entre a Aplicação e seus Usuários.

Software Multi-Componente

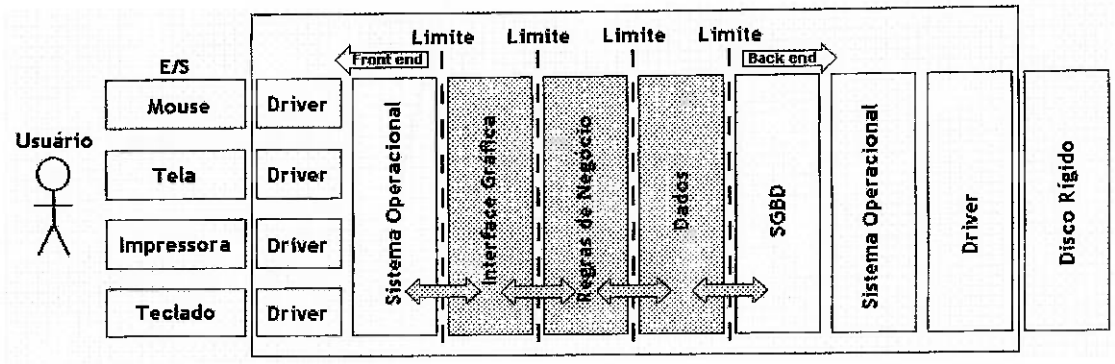

Figura 2.1.3.2 – Software Multi-Componente

Neste caso, suponhamos que os RFUs do software tenham sido divididos pelo desenvolvedor em três componentes:

- Interface Gráfica
- Regras de Negócio
- Dados

Todos os componentes estão no mesmo nível de abstração e trocam dados conforme um mesmo padrão, portanto estão na mesma camada. O desenvolvedor quer mensurar os componentes separadamente, pois serão desenvolvidos com tecnologias diferentes. Os RFUs da aplicação são refinados para representar os RFUs de cada componente.

Sobre as interações dos componentes podemos definir que cada um dos três componentes são usuários de seus componentes vizinhos e vice-versa. O componente de Interface Gráfica do software tem agora o sistema operacional em outra camada, sendo esse ultimo, usuário deste componente. Da mesma forma ficam relacionados o componente de Dados e o SGBD (Software Gerenciador de Banco de Dados), atuando este ultimo como usuário do componente de Dados.

Existem Limites, no conceito de Limite do COSMIC-FFP, entre os componentes e entre esses componentes e outras camadas. O método COSMIC-FFP pode mensurar o tamanho funcional do software sob o ponto de vista do desenvolvedor. Realizando a mensuração deste ponto de vista não há a necessidade de se mensurar a interação com o Usuário humano. Na pratica, os RFUs não são necessariamente alocados na mesma parte do software, por exemplo, podem estar distribuídos em vários componentes. Nos casos em que a arquitetura do sistema esteja disponível desde o começo, como no exemplo demonstrado acima, a alocação dos RFUs em especificas partes desta arquitetura ocasionará impacto no tamanho funcional de cada parte.

Para aqueles interessados apenas no tamanho funcional global dos requisitos, não é necessário considerar como esses requisitos são alocados, como no primeiro caso descrito. Já para aqueles que precisam mensurar o tamanho funcional de cada parte do software, esta maneira de identificação de limites será necessária, como no caso descrito acima, notando-se que os tamanhos obtidos nos dois casos serão diferentes.

O COSMIC-FFP provê uma maneira para ajudar na identificação dos RFUs de diferentes níveis de abstração, o conceito de camadas e componentes de software.

Modelo Genérico de Software

O Modelo Genérico de Software do COSMIC-FFP leva em conta estes princípios:

Principio 1

O software a ser mapeado e mensurado é alimentado por uma entrada e produz uma saída ou resultado útil para os usuários.

Principio 2

O software a ser mapeado e mensurado manipula porções de informação chamados de grupo de dados, que são compostos de atributos de dados.

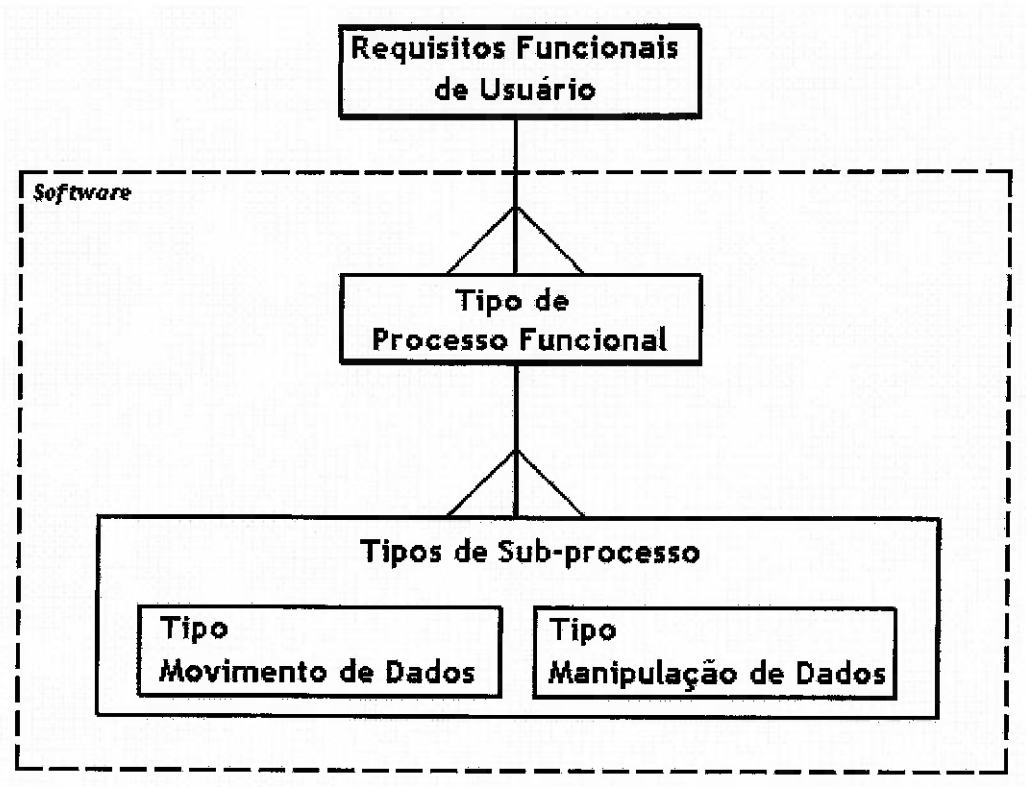

Figura 2.1.4 – Modelo de Software Genérico

Os RFUs podem ser decompostos em um conjunto de processos funcionais. Cada um desses processos funcionais é um único conjunto de sub-processos realizando um movimento de dados ou uma manipulação de dados.

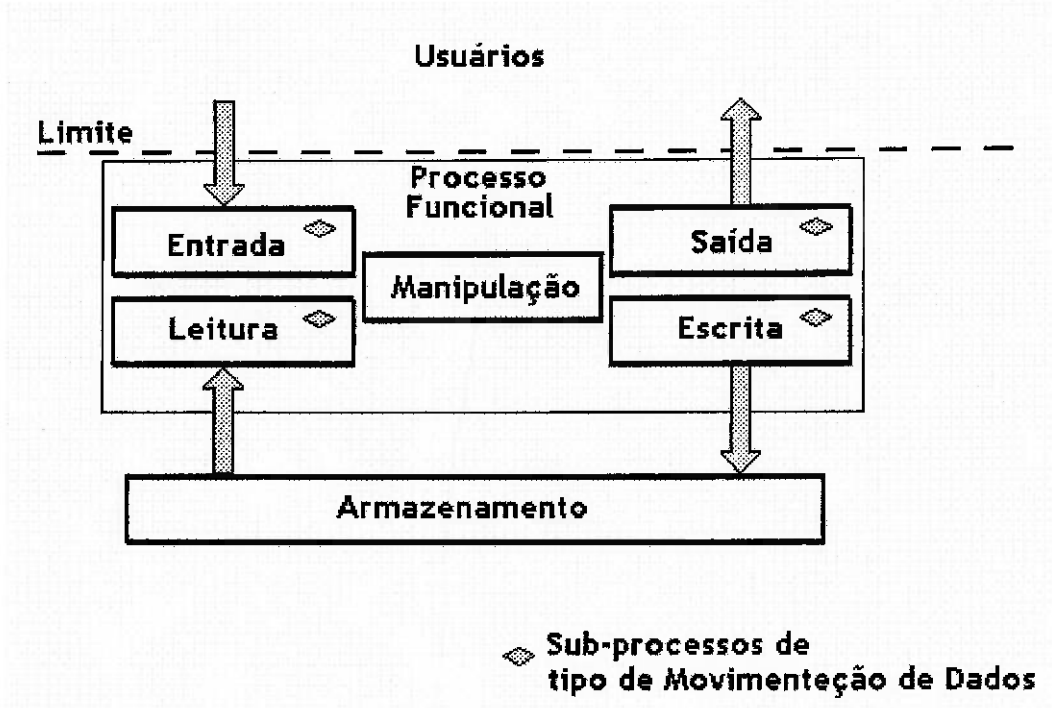

Figura 2.1.5 – Tipos de Sub-Processos

O modelo genérico de software do COSMIC-FFP define quatro tipos distintos de movimentação de dados: Entrada, Saída, Leitura e Escrita. Todo sub-processo de movimentação de dados move dados de apenas um grupo de dados. Entradas movem dados dos usuários pelo Limite até o processo funcional. Analogamente, Saídas movem dados de dentro do processo funcional para o usuário através do Limite. Leituras movem dados do dispositivo de armazenagem para um processo funcional e a Escrita move dados do processo funcional para o dispositivo de armazenagem. Estes movimentos de dados estão representados na figura logo acima.

Por utilizar os conceitos, definições e estrutura do COSMIC-FFP, os RFUs extraídos dos artefatos de uma parte de software acabam mapeados para o modelo genérico de software. Neste modelo estarão presentes todos os elementos necessários para mensurar o tamanho funcional, sendo que a informação não relevante para este fim, como requisitos técnicos e de qualidade, é descartada.

As regras e procedimentos da fase de mensuração são então aplicadas a este modelo para produzir um valor que represente a quantidade funcional daquele software.

Fase de Mensuração

Conceitos

A seguir as características principais das regras e procedimentos utilizados na fase de mensuração:

Função de Mensuração

Pela função de mensuração, a cada movimento de dados é atribuída uma quantidade de acordo com seu tipo.

Unidade de Mensuração

A unidade padrão de mensuração é o UTFC (Unidade de Tamanho Funcional do COSMIC), definido como equivalente a um movimento de dados, por convenção.

Sub-Unidade de mensuração

Quando é necessária uma maior precisão na mensuração, tem-se a opção de utilizar uma sub-unidade de mensuração, análoga ao sistema métrico, por exemplo, onde podemos dividir um metro quando há a necessidade. Seguindo esta linha de raciocínio podemos utilizar um simples atributo de dados como sub-unidade. Ensaios de mensuração executados em pequenos pedaços de software, não apresentaram grande diferença entre o número de atributos de dados de cada tipo de movimentos de dados. Sendo assim e também por facilitar o processo, tomaremos 1 UTFC como equivalente a uma movimentação de dados. A adoção desta medida pode fazer com que ocorram diferenças na mensuração quando o número de atributos de dados variar muito de um software para outro, tornando incompatível a comparação de um UTFC

de um software mensurado com esta sub-unidade opcional com o UTFC de um software mensurado sem esta opção. .

Adição

O tamanho funcional de um software é definido como a soma dos resultados da função de mensuração aplicada a cada um de seus movimentos de dados. Estendendo este conceito, o tamanho funcional de qualquer software em qualquer camada é a soma do tamanho funcional de suas partes. O tamanho funcional de qualquer pedido de mudança é definido como a soma de todo tamanho funcional adicionado, alterado e excluído inclusive.

Processo

Esta fase tem como entrada o modelo genérico de software, obtido na fase de mapeamento, e aplicando um conjunto de regras e procedimentos, produz um valor que é diretamente proporcional ao tamanho funcional do modelo utilizado na entrada. Esta definição segue o seguinte principio:

- O tamanho funcional de um software é diretamente proporcional ao número de seus movimentos de dados.

Por convenção, este valor numérico que representa a quantidade é estendido para representar o tamanho funcional do software.

Levando em conta as características acima descritas e utilizando 1 UTFC como unidade de mensuração, conclui-se que um software tem no mínimo tamanho funcional de 2 UTFCs, de acordo com o Principio Geral 1, que diz que um processo funcional tem no mínimo uma entrada e uma saída ou ainda uma leitura e uma escrita. Novamente tendo como base estas características, concluímos que não há limite máximo do tamanho funcional de software para um dado processo funcional.

O método COSMIC-FFP não pretende mensurar todos os aspectos de tamanho de um software, por exemplo, como já vimos anteriormente, ele despreza a quantidade de atributos de um movimento de dados. Também, a manipulação de dados não é levada em conta, partindo do principio que o software que o COSMIC-FFP propõe-se a mensurar não apresenta alta complexidade na manipulação de dados. Não obstante, a mensuração feita pelo COSMIC-FFP tem sido considerada uma boa aproximação para a finalidade e domínio propostos pelo método.

Parâmetros como a complexidade, podem ser considerados como um fator de tamanho, mas isso ainda hoje é um fator de grande pesquisa e discussão.

2.2 Análise de Ponto de Função

Introdução

O método de Análise de Ponto de Função (APF) foi desenvolvido por Allan Albrecht, enquanto trabalhava para a IBM, com o intuito de mensurar o software produzido. Sua intenção era de medir o tamanho de funcionalidades do ponto de vista do usuário, independente da implementação.

Apresentado ao publico pela primeira vez em 1979 como uma métrica de tamanho funcional de software, logo surgiram várias extensões e variações da proposta inicial, como MarkII.

Em 1986 foi formado o Grupo Internacional de Usuários de Ponto de Função (IFPUG) para manter o desenvolvimento do APF, e desde então são publicadas novas versões periodicamente, atualmente na versão 4.1.

Análise de Ponto de Função é um método padronizado para mensurar o desenvolvimento de software do ponto de vista do usuário.

A APF mensura o software quantificando a funcionalidade que o software provê ao usuário. Com isso o objetivo da APF é mensurar a funcionalidade que o usuário pede e a que recebe e mensurar o desenvolvimento e manutenção do software independentemente da tecnologia utilizada na implementação.

Além dos objetivos acima descritos a APF também se propõe a ser simples o suficiente para minimizar o esforço da mensuração e ser uma medida universal, entre projetos e entre empresas.

Tipos de Contagem de PF

O primeiro passo da contagem de ponto de função é identificar qual o tipo de contagem que será efetuado. Existem três tipos de contagem de ponto de função: projeto de desenvolvimento, projeto de melhoria e aplicação, que são descritos a seguir:

Projeto de Desenvolvimento

A contagem de ponto de função de um projeto de desenvolvimento de software mensura as funcionalidades entregues ao usuário na sua primeira versão, quando este é entregue.

Projeto de Melhoria

A contagem de ponto de função de um projeto de melhoria mensura as modificações sofridas pela aplicação, como adição, alteração e exclusão inclusive. Quando estas modificações são instaladas, a quantidade de pontos de função da aplicação também deve ser corrigida para refletir as modificações sofridas.

Aplicação

A contagem de ponto de função de uma aplicação é associada com uma aplicação já instalada. Este tipo de contagem também é conhecido como contagem de linha-base ou instalada. Esta contagem realiza a mensuração das atuais funcionalidades providas ao usuário pela aplicação. Esta contagem é iniciada quando a contagem de projeto de desenvolvimento é concluída, e é alterada quando uma contagem de projeto de melhoria altera as funcionalidades da aplicação.

Ciclo de Vida

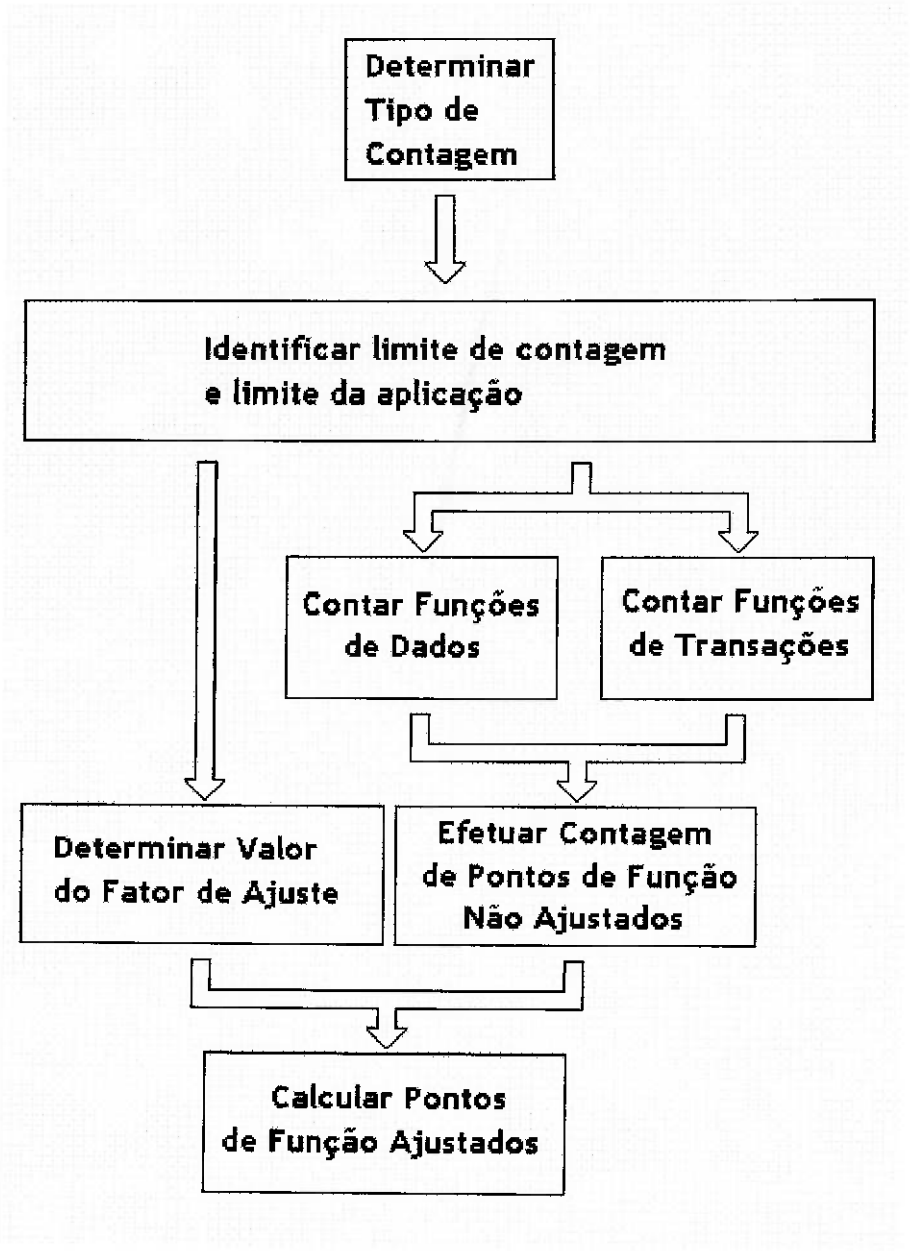

Figura 2.2.1 – Ciclo de Vida do método APF

Escopo de contagem e Limite de Aplicação

Conceitos

Propósito da Contagem

O propósito de uma contagem de ponto de função é prover uma resposta para um problema de negócio.

O propósito determina o tipo e o escopo da contagem para prover esta resposta e também influencia o posicionamento do limite entre o software a ser contado e o software a sua volta.

Escopo de Contagem

O escopo de contagem define as funcionalidades que serão incluídas numa contagem de ponto de função específica.

O escopo do conjunto de software a ser mensurado e é determinado pelo propósito da contagem. É função do escopo identificar quais funcionalidades serão incluídas na contagem como também prover as respostas relevantes para o propósito da contagem. O escopo pode englobar mais de uma aplicação.

No caso de uma contagem de melhoria o escopo inclui todas as funções que serão incluídas, alteradas e excluídas. O limite da aplicação que sofreu a alteração continua o mesmo. A funcionalidade da aplicação reflete o impacto das funcionalidades que foram adicionadas, alteradas e excluídas.

Em uma contagem de projeto de desenvolvimento o escopo inclui todas as funcionalidades desenvolvidas ou customizadas pelas atividades do projeto.

O Escopo de uma contagem de aplicação pode conter, dependendo do propósito, apenas as funcionalidades que serão utilizadas pelo usuário ou todas as funcionalidades entregues. Qualquer que seja o conjunto de funcionalidades escolhido o limite é o mesmo e independente do escopo.

Limite de Aplicação

O limite de aplicação define a borda entre o software a ser mensurado e o usuário. Este define o que é externo à aplicação. Limite é uma interface conceitual entre o interno, a aplicação, e o externo, o usuário. Comportando-se como uma membrana, pelo limite passam dados processados por transações. Também é característica do limite cercar os dados mantidos pela aplicação e ajudar na identificação de dados referenciados, mas não mantidos pela aplicação. O limite é independente da visão externa de negócio do usuário, ou seja, independente de aspectos de implementação ou técnicos.

Processo

A posição do limite da aplicação é importante, pois influencia a contagem de ponto de função. O limite da aplicação ajuda na identificação de dados que entram na aplicação e que serão incluídos no escopo da contagem.

Regras de Limite

- O limite é definido segundo a visão do usuário. O foco é no que o usuário pode entender e definir.
- O limite entre determinadas aplicações é baseado nas funcionalidades separadas que podem ser vistas pelo usuário e não por fatos técnicos.
- O limite inicial já estabelecido para a aplicação sendo modificada não é influenciado pelo escopo de contagem.

Procedimentos de Escopo de contagem e Limite de Aplicação

Quando se aplica uma contagem de ponto de função, as seguintes características da contagem devem ser documentadas:

- O propósito da contagem
- O escopo da contagem

- O limite da aplicação
- Qualquer outro relativo com estes descritos acima.

Contagem de Funções de Dados

Conceitos

Funções de Armazenamento de Dados

Funções de dados representam a funcionalidade provida ao usuário para atender os requisitos internos e externos de dados. Funções de dados são divididos entre dois tipos, arquivos lógicos internos (ALI) e arquivos de interface externos (AIE).

O termo arquivo nesta definição não significa arquivo no seu sentido físico, como estamos habituados. Nesta definição arquivo refere-se a um grupo de dados logicamente relacionado e não a implementação física desses grupos de dados.

Arquivos Lógicos Internos

Um Arquivo Lógico Interno (ALI) é um grupo de dados relacionados logicamente ou dados de controle mantidos dentro do limite da aplicação, que podem ser identificados pelo usuário. A principal finalidade de um ALI é armazenar informação utilizada por um ou mais processos da aplicação que esta sendo contada.

Arquivos de Interface Externos

Um Arquivo de Interface Externo (AIE) é um grupo de dados relacionados logicamente ou dados de controle que podem ser identificados pelo usuário, mas mantidos fora do limite da aplicação contada, isto é, que é mantida dentro do limite de outra aplicação. A principal finalidade de um AIE, como de um ALI, é armazenar informação utilizada por um ou mais processos dentro do limite da aplicação que esta

sendo contada, assim o AIE contado por esta aplicação deve ser um AIE de outra aplicação.

A diferença entre um ALI e um AIE está em quem o mantém, no caso do ALI é a aplicação que esta sendo contada e, no caso do ALI, é uma outra aplicação.

Tipos de Elementos de Dados

Um Tipo de Elemento de Dados (TED) é um campo que é unicamente reconhecido pelo usuário único e não repetido.

Tipo de Elemento de Registro

Um Tipo de Elemento de Registro (TER) é um sub-grupo de elementos de dados identificáveis pelo usuário e contidos em um ALI ou AIE.

Os sub-grupos de TERs estão divididos entre Opcionais e Mandatários.

Um sub-grupo Opcional é aquele sub-grupo em que o usuário tem a opção de criar ou não este sub-grupo de dados durante a adição ou criação desses dados por um processo.

Já em um sub-grupo Mandatário o usuário deve utilizar pelo menos uma vez este sub-grupo.

Processo

Regras de contagem de ALI e AIE

Existem dois tipos de regras de contagem:
- Regras de Identificação
- Regras de Complexidade e Contribuição

Regras de Identificação

Para se identificar um ALI, deve-se localizar grupos de dados que satisfazem a definição de ALI.

Estas regras de contagem devem ser válidas para que a informação seja contada como ALI:

- O grupo de dados ou informação de controle é lógica e pode ser identificada pelo usuário.
- O grupo de dados deve ser mantido por um processo dentro dos limites de aplicação que esta sendo contada.

Para se identificar um AIE, deve-se localizar grupos de dados que satisfazem a sua definição.

Estas regras de contagem devem ser válidas para que a informação seja contada como um AIE:

- O grupo de dados ou informação de controle é lógica e pode ser identificada pelo usuário.
- O grupo de dados é referenciado pela aplicação que esta sendo contada.
- O grupo de dados não é mantido pela aplicação que esta sendo contada.
- O grupo de dados é mantido por um ALI de outra aplicação.

Regras de Complexidade e Contribuição

O número de ALIs e AIEs e sua complexidade funcional determina a contribuição das funções de dados para a contagem de ponto de função não-ajustada.

Esta mensuração é feita atribuindo a cada ALI e AIE identificados uma complexidade funcional baseada no seu número de tipos de elementos de dados (TED) e tipos de elementos de registro (TER).

As regras para a contagem de TEDs são:

- Contar um TED para cada campo que seja unicamente reconhecido pelo usuário, e não repetido durante a armazenagem ou recuperação deste por um ALI ou AIE durante a execução de um dado processo.
- Quando duas aplicações mantêm ou referenciam o mesmo ALI ou AIE, mas cada uma mantém ou referencia TEDs separados, conta-se apenas os TEDs utilizados por cada aplicação como tamanho deste ALI ou AIE.
- Contar um TED para cada fragmento de dados requeridos por um usuário para estabelecer uma relação com outro ALI ou AIE.

Uma destas regras são aplicadas na contagem de TERs:
- Ou contar um TER para cada sub-grupo de um ALI ou AIE.
- Ou caso não exista nenhum sub-grupo, contar o ALI ou AIE como um TER.

Procedimentos de contagem de ALIs e AIEs

Passos para se identificar ALIs e AIEs:

- Identificar os ALIs da aplicação, seguindo as regras anteriormente descritas para este tipo de identificação.
- Identificar os AIEs da aplicação, seguindo as regras anteriormente descritas para este tipo de identificação.
- Calcular a Complexidade e a Contribuição de cada ALI e AIE, seguindo os procedimentos definidos a seguir.

Procedimentos de Complexidade e Contribuição

Abaixo descrevemos os passos para calcular a complexidade e contribuição de ALIs e AIEs em pontos de função não-ajustados.

Passo 1 – Utilize as regras de contagem de complexidade e contribuição descritas anteriormente para identificar e contar os TEDs e TERs.

Passo 2 – Calcule a complexidade funcional utilizando a seguinte tabela:

	1 a 19 TEDs	**20 a 50 TEDs**	**51 ou mais TEDs**
1 RET	Baixo	Baixo	Médio
2 a 5 RETs	Baixo	Médio	Alto
6 ou mais RETs	Médio	Alto	Alto

Tabela 2.2.1 – Complexidade de ALIs e AIEs

Passo 3 – Traduza a complexidade dos ALIs ou AIEs para pontos de função não ajustados utilizando as seguintes tabelas:

Complexidade Funcional	**Pontos de Função Não Ajustados**
Baixo	7
Médio	10
Alto	15

Tabela 2.2.2 – Contribuição de ALI

Complexidade Funcional	**Pontos de Função Não Ajustados**
Baixo	5
Médio	7
Alto	10

Tabela 2.2.3 – Contribuição de AIE

Passo 4 – Calcule as contribuições totais dos ALIs e AIEs da aplicação, somando os pontos de função não ajustados de cada um.

Contagem de Funções de Transação

Conceitos

Função de Transação

Funções transacionais representam a funcionalidade provida ao usuário pelo processamento de dados realizado por uma aplicação.

Estas podem ser divididas em Entrada Externa (EE), Saída Externa (SE) e Consulta Externa (CE).

Entrada Externa

Uma Entrada Externa (EE) é um processo elementar que processa dados ou informações de controle que vêm de fora do limite da aplicação. A principal finalidade de uma EE é manter um ou mais ALIs e/ou influenciar o comportamento da aplicação.

Saída Externa

Uma Saída Externa (SE) é um processo elementar que envia dados ou informações de controle para além do limite da aplicação. A principal finalidade de ema SE é apresentar informação ao usuário proveniente de dados ou informações de controle, que devem ser, mesmo que parcialmente, processadas. O processamento tem de conter no mínimo uma fórmula matemática ou cálculo, ou mesmo uma derivação de algum dado. Uma SE pode manter um ou vários ALIs e/ou também alterar o comportamento da aplicação.

Consulta Externa

Uma Consulta Externa (CE) é um processo elementar que envia dados ou informações de controle para além do limite da aplicação. A principal finalidade de uma CE é apresentar informação para um usuário proveniente de um ALI ou AIE. O processamento, ao contrário da SE, não deve conter nenhuma fórmula matemática ou cálculo, ou mesmo alguma derivação de algum dado. Também nenhum ALI é mantido durante o processamento, nem o comportamento da aplicação é alterada.

Tipo de Arquivo Referenciado

Um Tipo de Arquivo Referenciado (TAF) é um ALI lido ou mantido por uma função de transação ou um AIE lido por uma função de transação.

Processo

Regras de identificação de Processo Elementar

Para identificar processos elementares, procure por atividades de usuário que ocorrem na aplicação.
Os processos elementares devem seguir a estas regras:

- O processo é a menor parte de atividade que pode ser entendida pelo usuário.
- O processo executa e deixa a aplicação em um estado consistente.

Regras de Contagem de Funções Transacionais

Para classificar os processos elementares, é necessário determinar qual a principal finalidade deste e assim aplicar as regras deste tipo para identificar especificamente qual o tipo especifico de função de transação.

Entrada Externa

Para cada processo elementar em que a principal finalidade é manter um ALI ou alterar o comportamento da aplicação, devem ser aplicadas as regras elencadas abaixo para determinar se este pode ser classificado como uma EE. O processo deve atender a todas as regras para ser contado como uma EE única.

- O dado ou informação de controle é recebido de fora do limite da aplicação.
- Pelo menos um ALI deve ser mantido se a informação que entrou não for uma informação de controle que alterará o comportamento da aplicação.
- Uma das condições apresentadas a seguir deve ser verdadeira:
    - O processamento realizado é único dentre os processamentos realizados por outras EEs.
    - O conjunto de elementos de dados identificado é diferente de outros conjuntos de dados de outras EEs.
    - Os ALIs ou AIEs referenciados por esta EE são diferentes dos referenciados por outras EEs da aplicação.

Saída Externa

Para cada processo elementar em que a principal finalidade é apresentar informação ao usuário, devem ser aplicadas às regras elencadas abaixo para determinar se este é uma SE. O processo deve atender a todas as regras para ser contado como uma SE única.

- A função envia dados ou informações de controle para além do limite da aplicação.
- Uma das condições apresentadas a seguir deve ser verdadeira:
    - O processamento realizado é único dentre os processamentos realizados por outras SE.
    - O conjunto de elementos de dados identificado é diferente de outros conjuntos de dados de outras SE.
    - Os ALIs ou AIEs referenciados por esta SE são diferentes dos referenciados por outras SE da aplicação.
- Uma das condições apresentadas a seguir deve ser verdadeira:

- O processamento deste processo deve conter no mínimo uma fórmula matemática ou cálculo.

- O processamento deste processo deve criar dados derivados.

- O processamento deste processo deve manter no mínimo um ALI.

- O processamento deste processo deve alterar o comportamento da aplicação.

Consulta Externa

Para cada processo elementar em que a principal finalidade é apresentar informação ao usuário, devem ser aplicadas as regras elencadas abaixo para determinar se este é uma CE. O processo deve atender a todas as regras para ser contado como uma CE única.

- A função envia dados ou informações de controle para além do limite da aplicação.
- Uma das condições apresentadas a seguir deve ser verdadeira:

  - O processamento realizado é único dentre os processamentos realizados por outras CE.

  - O conjunto de elementos de dados identificado é diferente de outros conjuntos de dados de outras CE.

  - Os ALIs ou AIEs referenciados por esta CE são diferentes dos referenciados por outras CE da aplicação.

- O processamento deste processo deve recuperar dados ou informações de controle de um ALI ou AIE.
- O processamento deste processo não contém fórmula matemática ou cálculo.
- O processamento deste processo não cria dados derivados.
- O processamento deste processo não mantém um ALI.
- O processamento deste processo não altera o comportamento da aplicação.

Regras e Definições de Complexidade e Contribuição

O número de EEs, SEs e CEs e suas complexidades funcionais determinam a contribuição das funções de transação para a contagem de pontos de função não

ajustados. A cada EE, SE e CE é atribuído um valor de complexidade funcional baseado no número de tipos de arquivo referenciados e tipos de elementos de dados.

Regras de Complexidade e Contribuição de Entrada Externa

As regras de contagem de Tipo de Arquivo Referenciado para uma EE:
- Contar um TAR para cada ALI mantido.
- Contar um TAR para cada ALI ou AIE lido durante o processamento de uma EE.
- Contar apenas um TAR para cada ALI que é mantido e lido.

As regras de contagem de Tipo de Elemento de Dados para uma EE:
- Contar um TED para cada campo identificado de usuário e não repetido, que entra ou saí do limite da aplicação e é requerido para completar a EE.
- Não são contados campos que são recuperados ou derivados pelo sistema e armazenados em um ALI durante a execução do processo e não são enviados além do limite da aplicação.
- Contar um TED pela capacidade de enviar uma resposta de sistema além do limite da aplicação para sinalizar a ocorrência de um erro, confirmar processamento finalizado ou verificar se o processo deve continuar.
- Contar um TED pela capacidade de iniciar uma ação mesmo que haja vários métodos de executar o mesmo processo.

Regras de Complexidade e Contribuição de Saída Externa

As regras de contagem de Tipo de Arquivo Referenciado para uma SE:
- Contar um TAR para cada ALI ou AIE lido durante o processamento do processo elementar contado.
- Contar um TAR para cada ALI mantido durante o processamento do processo elementar contado.
- Contar apenas um TAR para cada ALI que é mantido e lido durante o processo contado.

As regras de contagem de Tipo de Elemento de Dados para uma SE:
- Contar um TED para cada campo identificado de usuário e não repetido, que entra, cruzando o limite da aplicação, e é necessário para especificar quando, porque e/ou como os dados serão recuperados ou gerados pelo processo.
- Contar um TED para cada campo identificado de usuário e não repetido, que saí do limite da aplicação.
- Se um TED entra e também sai dos limites da aplicação, ele é contado apenas uma vez para este processo.
- Contar um TED pela capacidade de enviar uma resposta de sistema além do limite da aplicação para sinalizar a ocorrência de um erro, confirmar processamento finalizado ou verificar que se o processo deve continuar.
- Contar um TED pela capacidade de iniciar uma ação mesmo que haja vários métodos de executar o mesmo processo.
- Não contar campos que são recuperados ou derivados pelo sistema e armazenados em um ALI durante o processo, se esses campos não cruzam o limite da aplicação.
- Não conte literais como TEDs.
- Não contar variáveis de número de páginas ou outras geradas pelo sistema operacional

Regras de Complexidade e Contribuição de Consulta Externa

As regras de contagem de Tipo de Arquivo Referenciado para uma CE:
- Contar um TAR para cada ALI ou AIE lido durante o processamento do processo elementar contado.

As regras de contagem de Tipo de Elemento de Dados para uma CE:
- Contar um TED para cada campo identificado de usuário e não repetido, que entra, cruzando o limite da aplicação, e é necessário para especificar quando, porque e/ou como os dados serão recuperados ou gerados pelo processo.
- Contar um TED para cada campo identificado de usuário e não repetido, que saí do limite da aplicação.

- Se um TED entra e também sai dos limites da aplicação, ele é contado apenas uma vez para este processo.
- Contar um TED pela capacidade de enviar uma resposta de sistema além do limite da aplicação para sinalizar a ocorrência de um erro, confirmar processamento finalizado ou verificar que se o processo deve continuar.
- Contar um TED pela capacidade de iniciar uma ação mesmo que haja vários métodos de executar o mesmo processo.
- Não contar campos que são recuperados ou derivados pelo sistema e armazenados em um ALI durante o processo, se esses campos não cruzam o limite da aplicação.
- Não conte literais como TEDs.
- Não contar variáveis de número de páginas ou outras geradas pelo sistema operacional

Procedimento para contagem de Transações Funcionais

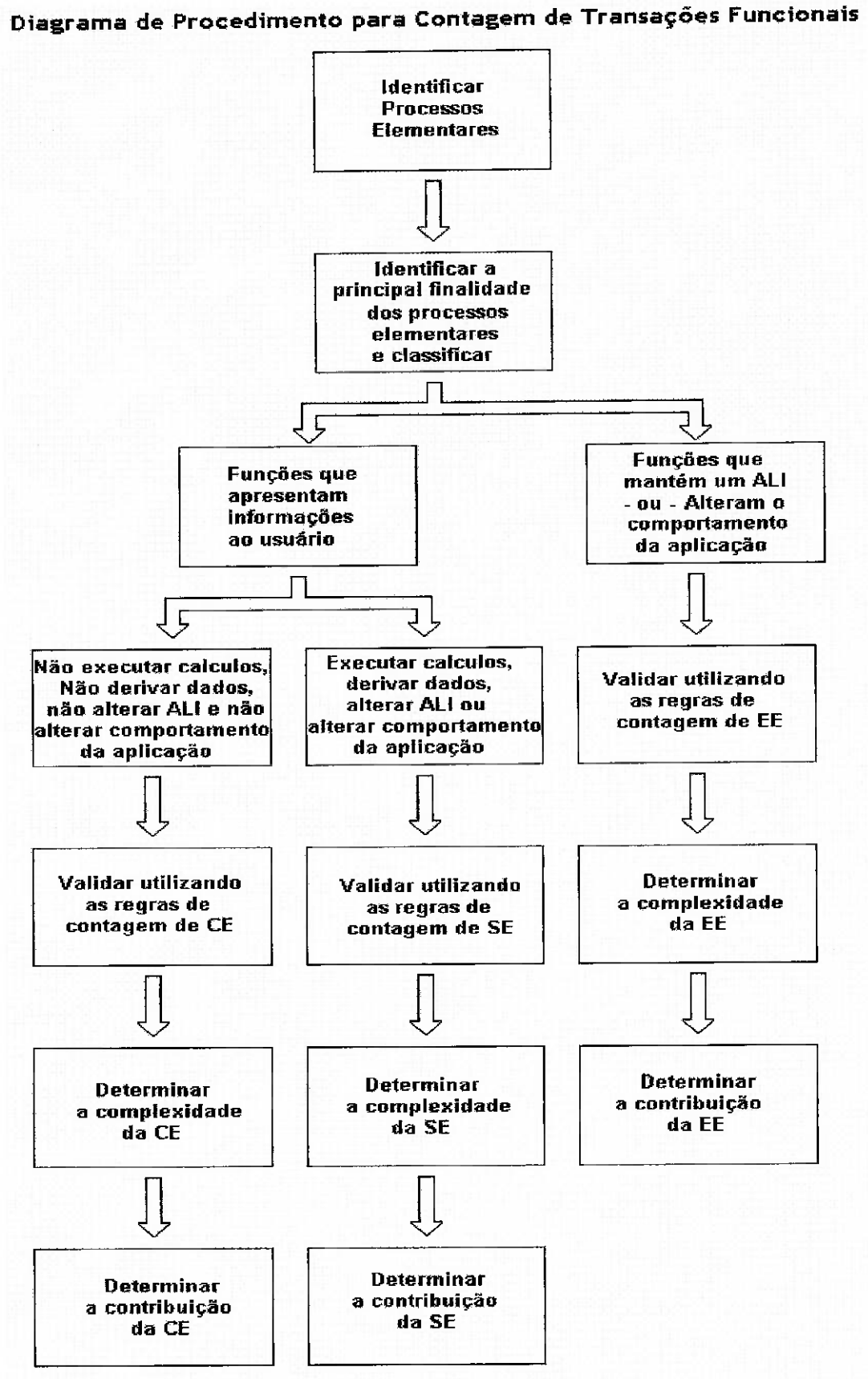

Figura 2.2.2 – Diagrama de Procedimento de Contagem de Transações Funcionais

Procedimentos de Complexidade

Para calcular a complexidade funcional de um EE, utiliza-se a seguinte tabela:

	1 a 4 TEDs	5 a 15 TEDs	16 ou mais TEDs
0 a 1 TAR	Baixo	Baixo	Médio
2 TARs	Baixo	Médio	Alto
3 ou mais TARs	Médio	Alto	Alto

Tabela 2.2.4 – Complexidade de EE

Para calcular a complexidade funcional de um SE, utiliza-se a seguinte tabela:

	1 a 5 TEDs	6 a 19 TEDs	20 ou mais TEDs
0 a 1 TAR	Baixo	Baixo	Médio
2 a 3 TARs	Baixo	Médio	Alto
4 ou mais TARs	Médio	Alto	Alto

Tabela 2.2.5 – Complexidade de SE

Para calcular a complexidade funcional de um CE, utiliza-se a seguinte tabela:

	1 a 5 TEDs	6 a 19 TEDs	20 ou mais TEDs
0 a 1 TAR	Baixo	Baixo	Médio
2 a 3 TARs	Baixo	Médio	Alto
4 ou mais TARs	Médio	Alto	Alto

Tabela 2.2.6 – Complexidade de CE

Procedimentos de Contribuição

Para calcular a contribuição funcional de um EE, utiliza-se a seguinte tabela:

Complexidade Funcional	**Pontos de Função Não Ajustados**
Baixo	3
Médio	4
Alto	6

Tabela 2.2.6 – Contribuição de EE

Para calcular a contribuição funcional de um SE, utiliza-se a seguinte tabela:

Complexidade Funcional	**Pontos de Função Não Ajustados**
Baixo	4
Médio	5
Alto	7

Tabela 2.2.7 – Contribuição de SE

Para calcular a contribuição funcional de um CE, utiliza-se a seguinte tabela:

Complexidade Funcional	**Pontos de Função Não Ajustados**
Baixo	3
Médio	4
Alto	6

Tabela 2.2.8 – Contribuição de CE

Valor do Fator de Ajuste

O Valor do Fator de Ajuste (VFA) é baseado em 14 Características Gerais de Sistemas (CGSs) que influenciam na funcionalidade geral da aplicação contada.

Cada característica tem descrições que ajudam a identificar o nível de influência desta na aplicação. O nível de influencia de cada característica vai de 0, sem influência, a 5, de forte influência.

Estas 14 CGSs são a base para o calculo do VFA. Quando aplicado, o VFA ajusta em +/- 35% os pontos de função não ajustados, dando origem aos pontos de função ajustados.

As 14 CGSs são:

1 – Comunicação de Dados

2 – Processamento de Dados Distribuídos

3 - Performance

4 – Configurações Utilizadas Frequentemente

5 – Taxa de Transações

6 – Entrada de Dados On-Line

7 – Eficiência do Usuário Final

8 – AtualizaçãoOn-Line

9 – Complexidade de Processamento

10 - Reusabilidade

11 – Facilidade de Instalação

12 – Facilidade de Operação

13 – Vários Sites

14 – Facilidade de Mudança

Graus de Influência

Baseado nos requisitos do usuário, cada CGS deve ser mensurado quanto a sua influência. O resultado deve ser um número de 0 a 5, sem influência e alta influência respectivamente.

0 – Sem Influência

1 – Influência incidental

2 – Influência moderada

3 – Influência média

4 – Influência significante

5 – Influência forte

Cálculo de Ponto de Função Ajustado

Cálculo para Projeto de Desenvolvimento

O cálculo para ponto de função de projeto de desenvolvimento consiste de três componentes:

- Funcionalidades de aplicação inclusas nos requisitos de usuário para o projeto
- Funcionalidades de conversão inclusas nos requisitos de usuário para o projeto
- Valor do Fator de Ajuste da aplicação

Funcionalidade de Aplicação

Funcionalidades de Aplicação são funções utilizadas pelo usuário depois da instalação do software, que satisfazem as necessidades do negócio do usuário.

Funcionalidades de Conversão

Funcionalidades de Conversão são funções providas apenas durante a instalação para migrar dados e/ou prover outros requisitos de migração especificados pelo usuário.

Valor de Fator de Ajuste de Aplicação

O Valor do Fator de Ajuste é determinado pela aplicação das 14 CGSs para determinar a complexidade funcional da aplicação.

Fórmula de Ponto de Função

A seguinte fórmula é utilizada para o calculo de ponto de função de um projeto de desenvolvimento:

PFD = (PFNA + PFC) * VFA

Onde:
PFD é o número de pontos de função ajustados do projeto de desenvolvimento.

PFNA é o número de pontos de função não ajustados das funcionalidades disponíveis após a instalação.
PFC é o número de pontos de função não ajustados das funcionalidades da conversão.
VFA é o valor do fator de ajuste da aplicação

Cálculo para Projeto de Melhoria

O cálculo para ponto de função de projeto de melhoria consiste de três componentes:
- Funcionalidades de aplicação inclusas nos requisitos de usuário para o projeto
- Funcionalidades de conversão inclusas nos requisitos de usuário para o projeto
- Valor do Fator de Ajuste da aplicação

Funcionalidade de Aplicação

Funcionalidades de Aplicação consiste de:
- Pontos de função identificados nas melhorias implementadas
- Pontos de função contados em funcionalidades alteradas durante o projeto de melhoria.
- Pontos de função contados em funcionalidades excluídas durante o projeto de melhoria.

Funcionalidades de Conversão

Funcionalidades de Conversão são pontos de função resultantes de funcionalidades de requisitos de migração especificados pelo usuário.

Valor de Fator de Ajuste de Aplicação

Os dois fatores de ajuste são:
- Valor de fator de ajuste da aplicação antes do projeto de melhoria começar.

- Valor de fator de ajuste da aplicação depois do projeto de melhoria completado.

Fórmula de Ponto de Função

A seguinte fórmula é utilizada para o calculo de ponto de função de um projeto de desenvolvimento:

PFM = [(ADIC + ALTER + PFC) * VFAA] + (EXCL* VFAD)

Onde:
PFM é o número de pontos de função ajustados do projeto de melhoria.
ADIC é o número de pontos de função não ajustados das funcionalidades que serão adicionadas pelo projeto de melhoria
ALTER é o número de pontos de função não ajustados das funcionalidades que serão alteradas pelo projeto de melhoria. É o tamanho funcional depois das modificações realizadas.
EXCL é o número de pontos de função não ajustados das funcionalidades que serão excluídas pelo projeto de melhoria
PFC é o número de pontos de função não ajustados das funcionalidades da conversão.
VFAA é o valor do fator de ajuste da aplicação antes do projeto de melhoria estar completo.
VFAD é o valor do fator de ajuste da aplicação depois do projeto de melhoria estar completo.

Cálculo para Aplicação

Existem duas variantes para o cálculo:
- Uma fórmula para calcular o número de pontos de função inicial de uma aplicação.
- Outra fórmula para atualizar o número de pontos de função de uma aplicação após a execução de um projeto de melhoria.

Fórmula para Estabelecer a Contagem Inicial

Inicialmente o usuário esta apenas recebendo novas funcionalidades, então a fórmula fica assim:

PFI = ADIC * VFA

Onde:
PFI é o número de pontos de função ajustados do projeto de melhoria.
ADIC é o número de pontos de função não ajustados das funcionalidades que serão adicionadas pelo projeto de melhoria
VFA é o valor do fator de ajuste da aplicação.

Fórmula para Atualizar PF de Aplicação

Quando um projeto de melhoria é instalado, os pontos de função da aplicação devem ser atualizados para refletirem as alterações sofridas. A funcionalidade da aplicação pode ser afetada das seguintes maneiras:
- Adicionando novas funcionalidades causando um aumento do tamanho funcional da aplicação.
- Alterando funcionalidades já existentes, causando um aumento, decremento ou até não alterando o tamanho funcional da aplicação.
- Excluindo funcionalidades, causando um decremento do tamanho funcional da aplicação.
- Alterando o fator de ajuste, causando um aumento, decremento ou até não alterando o tamanho funcional da aplicação.

A fórmula utilizada é descrita a seguir:

PFA = [(PFNNA + ADIC + ALTERA) - (ALTERB + EXCL)] * VFAA

Onde:

PFA é o número de pontos de função ajustados da aplicação atualizados

PFNAA é o número de pontos de função não ajustados contados antes do projeto de melhoria começar.

ADIC é o número de pontos de função não ajustados das funcionalidades que serão adicionadas pelo projeto de melhoria

ALTERA é o número de pontos de função não ajustados das funcionalidades que serão alteradas pelo projeto de melhoria, contados antes do projeto de melhoria começar.

ALTERD é o número de pontos de função não ajustados das funcionalidades que serão alteradas pelo projeto de melhoria, contados depois do projeto de melhoria começar.

EXCL é o número de pontos de função não ajustados das funcionalidades que serão excluídas pelo projeto de melhoria

VFAA é o valor do fator de ajuste da aplicação antes do projeto de melhoria estar completo.

3 COMPARAÇÃO ENTRE O MÉTODO COSMIC-FFP E O MÉTODO ANÁLISE DE PONTO DE FUNÇÃO

Definição de um Modelo Comum

Um método de Mensuração de Tamanho Funcional consiste na aplicação de um conjunto de regras e procedimentos em um dado software. O resultado da aplicação destas regras e procedimentos é um número que representa o tamanho funcional deste software. As regras e procedimentos do método COSMIC-FFP tanto como as do método APF, estão contidos em documentos intitulados "Manual de Práticas de Contagem" de cada método respectivamente.

Segundo Rob Kusters (Kusters 1999) podemos observar que estes dois métodos:
- São independentes de decisões de implementação.
- Consistem na aplicação de regras e procedimentos sobre alguns artefatos do software a ser mensurado.

Sendo assim, Rob Kusters em (Kusters 1999) propõe um Modelo Comum para auxiliar a comparação deste dois métodos.

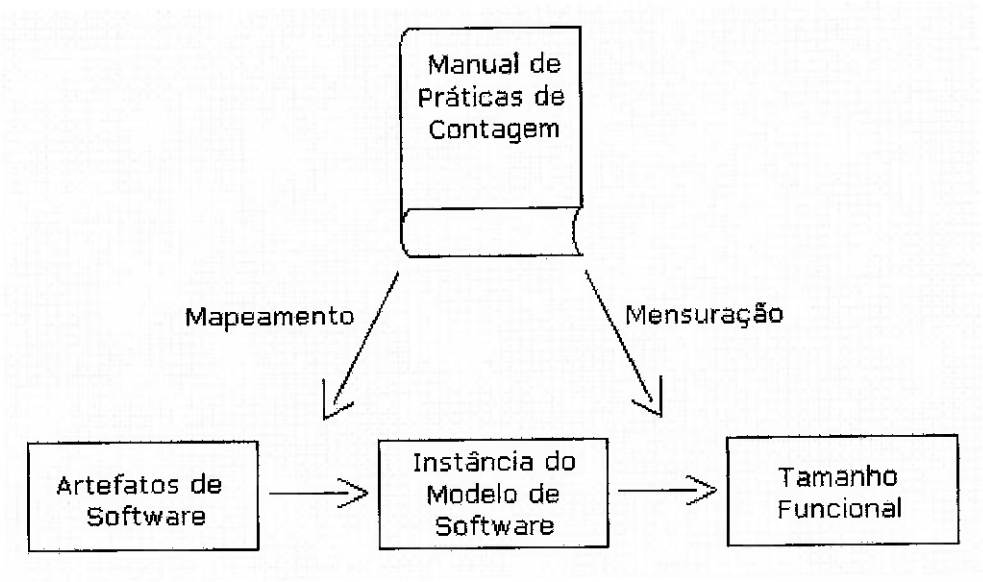

Figura 3.1 – Modelo Comum

Este Modelo Comum proposto mostra que antes de se aplicar as regras e procedimentos, o software a ser mensurado deve ser mapeado para um modelo que mantenha os conceitos e definições requeridas para uma mensuração funcional. Embora este Modelo Comum de Software não esteja explicitamente definido no Manual de Práticas de Contagem do método APF, este modelo, mesmo que implícito, existe neste método . Portanto podemos dividir o processo do método APF também, assim como o método COSMIC-FFP, em duas etapas, uma de mapeamento que consiste no mapeamento dos artefatos do software a ser mensurado para um Modelo Comum de Software e uma outra etapa, a de mensuração, que consiste em mensurar características específicas deste modelo.

A Etapa de Mapeamento tem como entrada os artefatos do software a ser mensurado e produz como resultado uma instância deste Modelo Comum de Software . Essa instância deste Modelo Comum de Software é definida pelos conceitos e definições de cada método. Em seguida a Etapa de Mensuração toma como entrada essa instância deste Modelo Comum de Software e produz como resultado um número que representa o tamanho funcional deste modelo. Este número é obtido aplicando-se um conjunto de regras e procedimentos, específicos de cada método, nesta instância de Modelo Comum de Software. Por convenção, este número também representará o tamanho funcional do software em que este foi baseado.

Comparação das Etapas de Mapeamento

A primeira etapa que compararemos é a Etapa de Mapeamento. Esta etapa tem como entrada os artefatos do software a ser mensurado e produz como resultado uma instância deste Modelo Comum de Software. Para a comparação destes Modelos Comuns de software gerados pelos métodos COSMIC-FFP e APF, precisamos antes extrair de cada um os conceitos que os caracterizam. O Modelo Comum de Software do método COSMIC-FFP possui seis conceitos enquanto que o método da APF possui quatro conceitos.

Conceito	**COSMIC-FFP**	**APF**
Limite	Limite	Limite
Usuário	Usuário	Usuário
Camada	Camada	
Objetos de Dados	Grupo de Dados	Arquivos Lógicos
Objetos de Processos	Processos Funcionais	Processos Elementares
Objeto de Sub-Processo	Sub-Processo	

Tabela 3.1 – Conceitos do COSMIC-FFP e APF

A seguir apresentaremos uma análise comparativa dos conceitos dos Modelos Comuns de Software de cada método.

Limite

Conceito	Definição	
	COSMIC-FFP	APF
Limite	Limite – "Limite é definido como a interface entre o software estudado e seus	Limite – "O limite de aplicação define a borda entre o software

	usuários" (COSMIC 2003)	a ser mensurado e o usuário" (IFPUG 1999)

Tabela 3.2 – Conceitos de Limite do COSMIC-FFP e APF

De acordo com os respectivos Manuais de Práticas de Contagem, o conceito de Limite dos dois métodos é o mesmo, ou seja, é a fronteira entre o software a ser mensurado e seus usuários.

Usuário

Conceito	Definição	
	COSMIC-FFP	APF
Usuário	Usuário – "Usuários podem ser seres humanos, outros dispositivos ou mesmo outro software".(COSMIC 2003)	Usuário – "é o ser humano que utiliza a aplicação".(IFPUG 1999)

Tabela 3.3 – Conceitos de Usuário do COSMIC-FFP e APF

A definição de usuário do método COSMIC-FFP inclui e estende a definição do método APF. Usuário pelo método COSMIC-FFP além de englobar os seres humanos que interagem com o software também inclui dispositivos e ou outro software.

Camada

Conceito	Definição	
	COSMIC-FFP	APF
Camada	Camada – "é o resultado do	

	particionamento funcional do software, sendo que todos processos funcionais incluídos nesta executam no mesmo nível de abstração". (COSMIC 2003)	

Tabela 3.4 – Conceitos de Camada do COSMIC-FFP e APF

O Conceito de camada somente é definido explicitamente no método COSMIC-FFP, onde este permite que o software a ser mensurado possa ser separado em partes com funcionalidades no mesmo nível de abstração.

Objeto de Dados

Conceito	Definição	
	COSMIC-FFP	APF
Objeto de Dados	Grupo de Dados – "é um distinto, não vazio, não ordenado e não redundante conjunto de atributos de dados, onde cada atributo descreve um aspecto deste".(COSMIC 2003) Tipo de Movimento de Dados – "é um Componente Funcional Básico* (CFB) que move um ou mais atributos de dados pertencentes a um único grupo de dados" (COSMIC	Funções de Dados – "representam a funcionalidade provida ao usuário para atender os requisitos internos e externos de dados" (IFPUG 1999). Existem dois tipos de Funções de Dados: arquivos lógicos internos (ALI) e arquivos de interface externos (AIE).

	2003). Existem quatro tipos de movimento de dados: entrada, saída, leitura e escrita.	

Tabela 3.5 – Conceitos de Objetos de Dados do COSMIC-FFP e APF

A definição de objetos de dados do método APF esta relacionada com o que o usuário pode identificar. A definição de objetos de dados do método COSMIC-FFP também os relaciona com a visão do usuário, mas por sua definição de usuário ser mais abrangente a sua definição de objetos de dados também é mais ampla. Nos dois métodos os objetos de dados são identificados de uma perspectiva lógica e não incluem questões de implementação, isto é, excluem questões de qualidade e técnicas.

Objeto de Processo

Conceito	Definição	
	COSMIC-FFP	APF
Objeto de Processo	Processo Funcional – "é um componente elementar de um conjunto de Requisitos Funcionais de Usuário compreendendo um único conjunto de movimento de dados coesamente e independentemente executável". (COSMIC 2003)	Processo Elementar – "é a menor unidade de atividade que é relevante ao usuário. (...) deve ser auto-contido e deixar a aplicação num estado consistente". (IFPUG 1999)

Tabela 3.6 – Conceitos de Objetos de Processo do COSMIC-FFP e APF

As definições dos objetos de processos dos dois métodos são baseadas implicitamente no tamanho. Na definição do método APF utiliza-se o adjetivo menor e no método COSMIC-FFP o adjetivo elementar para definir o objeto de processo. No Manual de Pratica de Contagem da APF embora não existem regras para definir o tamanho do processo, existem algumas dicas para essa finalidade. As definições dos dois métodos fazem referência ao fato de que seus objetos de processos são independentes, na definição do método APF usa-se o termo auto-contido. O método APF ainda explicitamente define que seu objeto de processo deve deixar a aplicação em um estado consistente.

Objeto de SubProcesso

Conceito	Definição	
	COSMIC-FFP	APF
Objeto de Sub-Processo	Sub-Processo – "é um Movimento de Dados que ocorre durante a execução de um Processo Funcional. Existem quatro tipos de Sub-Processos, cada um equivale a um dos Tipos de Movimento de Dados". (COSMIC 2003)	

Tabela 3.7 – Conceitos de Objetos de SubProcesso do COSMIC-FFP e APF

O conceito de sub-processo não é definido no método APF. No método COSMIC-FFP sub-processo representa cada Movimento de Dados do Processo Funcional. Cada sub-processo equivale a um Componente Funcional Básico, que é definido pela ISO (ISO/IEC 14143-1:1998) como a unidade elementar dos Requisitos Funcionais do Usuário (RFU) definido por Método de Mensuração Funcional de Software para fins de mensuração.

Comparação das Etapas de Mensuração

Para comparar as etapas de mensuração dos métodos COSMIC-FFP e APF, vamos primeiramente agrupar os passos de cada método em subprocessos conforme suas afinidades.

Para definir estes subprocessos levaremos em conta o elemento principal de um método de mensuração de tamanho funcional de software que é o Componente Funcional Básico (CFB), padronizado pelo ISO (ISO/IEC 14143-1:1998).

Os subprocessos propostos para dividir o processo de mensuração são:

1. Identificação dos Componentes Funcionais Básicos
2. Definição da Contribuição dos Componentes Funcionais Básicos
3. Cálculo do Tamanho Funcional

Identificação dos Componentes Funcionais Básicos

Este subprocesso do processo de mensuração compreenderá a identificação dos CFBs existentes no Modelo Comum proposto resultante da etapa de mapeamento anterior.

Estão sendo englobados neste subprocesso o passo de identificação dos tipos de Movimento de Dados do método COSMIC-FFP e os passos de identificação de Funções de Dados e Transações do método APF.

Tipos de Componentes Funcionais Básicos		
	COSMIC-FFP	APF
Repositório de Dados	-	Arquivo Lógico Interno (ALI)

		Arquivo de Interface Externo (AIE)
Transação	Entrada (E)	Entrada Externa (EE)
	Saída (S)	Saída Externa (SE)
	Leitura (L)	Consulta Externa (CE)
	Escrita (W)	

Tabela 3.8 – Componentes Funcionais Básicos do COSMIC-FFP e APF

O método COSMIC-FFP possui quatro tipos de CFBs distintos, são eles:

1. Entrada (E)
2. Saída (S)
3. Leitura (L)
4. Escrita (W)

Para a mensuração deste método são relevantes as transações de dados - explicitamente chamados de Movimentos de Dados em seu Manual de Práticas de Contagem - dos processos funcionais, ou seja, o tamanho funcional é proporcional à quantidade de movimento de dados.

Este método não leva em conta repositórios de dados, sejam eles internos ou externos.

O método APF possui cinco tipos de CFBs distintos, são eles:

1. Arquivo Lógico Interno (ALI)
2. Arquivo de Interface Externo (AIE)
3. Entrada Externa (EE)
4. Saída Externa (SE)
5. Consulta Externa (CE)

Já o método APF além de levar em conta transações de dados, que, conforme seu Manual de Práticas de Contagem, são chamados de Funções de Dados, também leva em conta os repositórios de dados do software.

Definição da Contribuição dos Componentes Funcionais Básicos

Este subprocesso consistirá na definição da contribuição de cada CFB existentes no Modelo Comum.

Neste subprocesso estão compreendidos o passo de aplicação de função de mensuração do método COSMIC-FFP e os passos de determinação de complexidade e contribuição de Funções de Dados e Transações do método APF.

Contribuição de Componentes Funcionais Básicos por Complexidade					
	CFB	Depende (Quantidade)	Contribuição/Complexidade		
			Baixa	Média	Alta
COSMIC-FFP	Entrada (E)		1		
	Saída (S)		1		
	Leitura (L)		1		
	Escrita (W)		1		
APF	Arquivo Lógico Interno (ALI)	TEDs e RETs	7	10	15
	Arquivo de Interface Externo (AIE)	TEDs e RETs	5	7	10
	Entrada Externa (EE)	TEDs e TARs	3	4	6
	Saída Externa (SE)	TEDs e TARs	4	5	7
	Consulta Externa (CE)	TEDs e TARs	3	4	6

Tabela 3.9 – Contribuição dos Componentes Funcionais Básicos por complexidade do COSMIC-FFP e APF

A contribuição de cada CFB do método COSMIC-FFP independe do tamanho deste CFB, e equivale a 1 UTFC.

Já no método APF, para cada CFB é definida a complexidade entre baixa, média e alta levando em conta os TEDs e se o CFB for de Função de Dados os RETs ou se o

CFB for de Função de Transação os TARs. Definida a complexidade, é atribuído a cada CFB uma quantidade de pontos segundo seu tipo e complexidade.

Cálculo do Tamanho Funcional

Este subprocesso compreenderá o processo de calcular o número que representará o tamanho funcional do software que esta sendo mensurado.

Neste subprocesso estão compreendidos o passo de aplicação de função de agregação do método COSMIC-FFP e os passos de determinação do Valor de Ajuste e de cálculo do total de pontos de função Ajustado do método APF.

Para se obter o número que representará o tamanho funcional do software mensurado, no método COSMIC-FFP, basta aplicar a função de agregação deste método, que nada mais significa do que somar a contribuição individual de cada CFB.

No método APF a soma da contribuição dos CFBs de Funções de Dados e de Transação estão implicitamente atribuídas aos seus respectivos passos de determinação de contribuição e complexidade.

Ao contrário do método COSMIC-FFP, onde o resultado já foi obtido, no método APF ainda é necessário um ajuste no número de pontos de função até agora obtidos. Estes pontos de função, chamados de Pontos de Função Não Ajustados, sofrerão uma variação de até 35%, para mais ou menos, dependendo do Valor de Ajuste calculado. Tendo-se o Valor de Ajuste e utilizando a equação equivalente ao tipo de contagem, obtém-se então o número de Pontos de Função Ajustados, que é o número que representa o tamanho funcional do software mensurado.

4 CONCLUSÃO

Neste capítulo apontaremos as semelhanças e diferenças entre os métodos APF e COSMIC-FFP, e também suas vantagens e desvantagens. Cada item deste será tratado em um tópico especifico.

Semelhanças

Ambos os métodos APF e COSMIC-FFP utilizam conceitos semelhantes de unidade funcional, isto é, o objeto de processo citado anteriormente, para a mensuração, chamado de processo elementar no método APF e processo funcional pelo método COSMIC-FFP.

Outro fator coincidente entre os dois métodos é o fato de que a movimentação de dados, tanto para fora como para dentro do processo, contribui para o tamanho funcional do software e também, nos dois métodos, o acesso a repositório de dados contribui para o tamanho funcional do software.

Algoritmos, transformações de dados e cálculos não contribuem para o tamanho funcional em nenhum dos dois métodos.

Diferenças

Segundo Koni Houston (Houston, Koni 2003) uma das principais diferenças entre o método APF e COSMIC-FFP está no que contribui para o tamanho funcional de cada um. O método APF leva em conta tanto a movimentação de dados para dentro ou fora dos processos, como também os dados que são persistidos pelo sistema, ou seja, o repositório de dados. No método COSMIC-FFP apenas a movimentação de dados para dentro ou para fora do processo contribui para seu tamanho funcional.

O método APF tem dois Componentes Funcionais Básicos (CFBs) de dados, Arquivo Lógico Interno e Arquivo de Interface Externo, e três CFBs de processo, Entrada Externa, Saída Externa e Consulta Externa. Já o método COSMIC-FFP possui CFBs de processo apenas, que estão no nível de sub-processo, são quatro, Entrada, Saída, Leitura e Escrita.

Uma unidade funcional, pode no método APF ter um tamanho mínimo de 3 pontos de função e máximo 7 pontos de função. No método COSMIC-FFP uma unidade funcional pode ter um tamanho mínimo 2 UTFCs e o tamanho máximo é infinito.

Para o método APF cada atributo de dados que cruza o limite do software que esta sendo mensurado, contribui para o tamanho funcional deste. No método COSMIC-FFP são os grupos desses atributos que vão contribuir para o tamanho funcional.

O acesso aos dados persistidos pelo sistema também contribui diferentemente para o tamanho funcional em cada método. No APF tanto a leitura e/ou escrita de um mesmo TAR é contada uma vez apenas. Já no COSMIC-FFP se houver uma leitura e uma escrita, os dois movimentos de dados são contados.

Vantagens do Método COSMIC-FFP

Segundo Alain Abran em (Abran, Alain 2001) uma das principais vantagens do método COSMIC-FFP é que este pode mensurar software de tempo-real além de software de gerenciamento de informação.

A utilização das mesmas regras de contagem para todos os processos, que é basicamente identificar os CFBs de sub-processo do processo e então contá-los, facilita o processo de mensuração do software.

Outra vantagem deste método, é que este define explicitamente regras para mensurar software de várias camadas.

O Manual de Práticas de Contagem e Estudos de Caso estão disponíveis gratuitamente.

Desvantagens do Método COSMIC-FFP

No método COSMIC-FFP a identificação de CFBs é realizada em um nível abaixo do de processo, o que torna necessário uma especificação mais bem detalhada.

O conceito de camadas ainda não está bem definido, causando contradição na identificação destas por diferentes pessoas.

Existem ainda poucos exemplos e dicas para contagem em diferentes ambientes. Também ainda não existe certificação, e literatura, treinamento e ferramentas ainda são poucos.

A resposta da indústria de software sobre este método ainda é pequena, por exemplo, o ISBGS (Grupo Internacional de Padrões de Testes de Software) conta com apenas 30 históricos de projetos de software que utilizaram o método COSMIC-FFP, contra mais de 1500 que utilizaram o método APF.

Vantagens do Método APF

Uma das vantagens deste método é que para determinar a complexidade de um CFB não é necessário a contagem exata do número de TEDs, TERs ou TARs, pois essa complexidade é dada pelo intervalo da contagem desses. Por exemplo, se para uma Entrada Externa já foram contados mais de dois TARs e mais de quatro TEDs já pode-se classificá-la como de alta complexidade.

Além disso, pode-se efetuar uma contagem rápida, embora que esta se torne imprecisa, contando-se apenas o número de ocorrências de CFBs e atribuindo a todos estes a mesma complexidade.

Por ser um método que já esta a muito tempo no mercado, existem maneiras de relacionar o ponto de função com esforço e custo.

Ao contrario do método COSMIC-FFP, para o método APF existe um amplo acervo de exemplos e dicas para contagem em diferentes ambientes.

Também existe uma certificação que é provida pela IFPUG, e existe bastante literatura, treinamentos e ferramentas tanto homologadas pela IFPUG como de terceiros.

Desvantagens do Método APF

Segundo Alain Abran em (Abran, Alain 2001), um dos problemas deste método é que freqüentemente os CFBs são incorretamente mensurados.

A principal causa da ocorrência deste erro de mensuração é que em alguns casos o processo de identificação da principal finalidade do CFB torna-se confuso, gerando identificações errôneas do tipo de CFB.

Outra causa da mensuração incorreta é quando o CFB faz parte de um sistema de multicamadas, já que este método não define explicitamente o conceito de camadas.

Conclusão Final

A idéia de mensurar o tamanho de uma especificação de requisitos de software independentemente da tecnologia utilizada para desenvolvê-lo, proposta por Albrecht nos anos 70 continua um desafio até hoje.

A engenharia de software mudou muito nas ultimas duas décadas, foram desenvolvidas novas metodologias, notações, conceitos, isso sem falar nas tecnologias.

Inevitavelmente os métodos de mensuração de tamanho funcional de software também têm de se adaptar a esse novo ambiente, já que o ambiente para que estes foram desenvolvidos, uso de mainframe, notações não orientadas a objeto, embora que ainda utilizados, não são mais o padrão de mercado. É isso o que método COSMIC-FFP vem a oferecer, a compatibilidade com essas novas metodologias, notações e tecnologias, o que o método APF em muitos casos não atende.