

# Resumo APF

---

## 1. Por que medir software?

Por que nos damos ao trabalho de medir projetos de software? Qual é o ganho nisso, e que respostas eu obtenho com métricas? Métricas ajudam a responder perguntas cruciais no desenvolvimento e gerenciamento de projetos de software.

**Para estimar o esforço necessário no atendimento de novos requisitos.** Métricas permitem estimar o tamanho do software em qualquer fase do ciclo de vida. Assim é possível saber qual é o real esforço por trás do atendimento a certos requisitos.

**Para acompanhar o progresso do meu projeto?** Métricas permitem controle e correção de problemas. Através delas é possível determinar se o projeto está atrasado, ou se estourou o orçamento e, ainda, quanto de trabalho ainda resta a ser realizado.

**Para tomar decisões.** Devo contratar mais recursos, diminuir o escopo, ou aumentar a produtividade? Métricas permitem justificar e defender decisões tomadas. Através delas é possível realizar análises de riscos e determinar se alguns marcos do projeto foram alcançados.

**Para análise de *make or buy*.** Devo “implementar do zero” ou comprar pronto? Devo terceirizar? Métricas ajudam na análise de “*make or buy*”. A empresa pode usar o seu histórico de desempenho e compará-lo aos níveis oferecidos pelo mercado para tomar esta decisão.

**Para subsidiar contratos.** Como devo contratar um fornecedor? Vou pagar por hora? Contratos de homem-hora remuneram somente a passagem do tempo. Métricas como Pontos de Função medem o produto e remuneram o resultado (funcionalidade) entregue.

## 2. O que medir?

Você deve medir aquilo que é relevante para análise. Pode ser (dentre outras coisas):

- Recursos e Custos (pessoas, ferramentas, orçamento, contabilização, etc.);
- Qualidade (tempo médio entre falhas, incidentes, problemas, etc.);
- Cronograma (atrasos, marcos, entregas, etc.);
- Progresso (entregáveis, funcionalidades prontas, documentos validados, etc.);
- Tamanho (PF).

Vamos nos concentrar no Tamanho do Software, que é a medida relacionada a Pontos de Função.

## Qual medida melhor representa o tamanho?

À primeira vista, a unidade mais intuitiva seria o número de linhas de código. Teoricamente, dois programas de mesma linguagem de programação poderiam ser comparados e o que tivesse mais linhas de código seria “maior”.

Porém, isto é uma falácia. A contagem de linhas de código envolve uma série de riscos que inviabilizam o seu uso como medida:

- **Não há padronização na contagem** (Conto comentários? E linhas em branco? E diretivas ao compilador? Devo contar cada comando como uma linha?);
- **LOC não significa nada para clientes e usuários.** Não tem foco no negócio, na funcionalidade. 1 milhão de linhas de código é muito ou pouco?
- **Não há como estimar confiavelmente a quantidade de LOC nas fases iniciais do ciclo de vida.**
- **Depender de linguagem de programação é inaceitável.** Isso gera problemas quando um sistema é implementado em várias linguagens, ou quando essas próprias linguagens evoluem (JSE 1.4, 5.0, 6.0, etc...).

Percebe-se, então, que é inviável utilizar linhas de código como medida.

O método adotado como padrão mundial para medir o tamanho funcional de um software é a Análise de Pontos de Função, sendo eles (os Pontos de Função) a medida adotada. APF é uma técnica de medição das funcionalidades fornecidas do ponto de vista do usuário e tem por objetivo tornar a contagem independente de tecnologia utilizada. APF busca medir o que o software faz, e não como ele foi construído.

Apesar disso, é importante perceber que APF não mede diretamente esforço, produtividade ou custo. É exclusivamente uma medida de tamanho funcional do software. Esse tamanho funcional, junto com outras variáveis e o histórico da sua organização é que pode ser usado para derivar outras informações.

## 3. Histórico da APF

Na década de 70, Allan Albrecht foi encarregado de medir a produtividade de vários projetos de software desenvolvidos na IBM, mas estes projetos tinham sido programados em linguagens distintas, inviabilizando a utilização da medida de linhas de código. Isso motivou a busca por uma medida que fosse independente da linguagem de programação utilizada.

Após a proposta inicial na IBM, o método se popularizou e deu início ao IFPUG. O IFPUG (*International Function Point Users Group*) é uma entidade sem fins lucrativos, composta por pessoas e empresas de diversos países, cuja finalidade é promover um melhor gerenciamento de processos de desenvolvimento e manutenção de software com o uso de pontos de função e outras técnicas de medição. É quem mantém o CPM (*Counting Practices Manual*), onde o método de contagem está descrito em detalhes.

Os principais objetivos do IFPUG Counting Practices Manual, Release 4.3, são

- Manter conformidade com a norma ISO/IEC 14143-1:2007 Information technology – Software measurement – Functional size measurement – Definition of concepts
- Prover uma descrição clara e detalhada da contagem de pontos de função
- Garantir que as contagens sejam consistentes com as práticas de contagem dos afiliados do IFPUG
- Fornecer um guia para permitir a contagem de pontos de função a partir dos entregáveis das metodologias e técnicas mais conhecidas
- Prover um entendimento comum para permitir que os fornecedores de ferramentas forneçam suporte automatizado para a contagem de pontos de função

Os grandes releases do CPM foram os seguintes:

- Após as definições iniciais na IBM, o crescimento do uso de pontos de função trouxe um grande aumento no número de aplicações medidas. Esta ampliação da aplicação desafiou a descrição original da medida e tornou necessária a criação de um guia para interpretação das regras originais em novos ambientes. Isto foi refletido na Release 2.0 (abril de 1988) do International Function Point Users Group (IFPUG) Function Point Counting Practices Manual.
- A Release 3.0 (abril de 1990) do IFPUG Function Point Counting Practices Manual foi o principal marco na evolução da medição do tamanho funcional. Pela primeira vez, o Comitê de Práticas de Contagem do IFPUG trabalhou para transformar o que era uma coleção de muitas interpretações das regras em um documento verdadeiramente coerente, representando uma visão consensual das regras de contagem de ponto de função. Neste sentido, este foi o primeiro passo para o estabelecimento de padrões reais para a medição de pontos de função, os quais poderiam ser aplicados a diversas organizações.
- A Release 4.0 (janeiro de 1994) foi o marco seguinte na evolução da medição do tamanho funcional. Essa release abordou o uso de pontos de função nas fases iniciais do desenvolvimento dos projetos, para estimar o tamanho do projeto utilizando disciplinas de engenharia da informação. O número rapidamente crescente de aplicações com janelas de interface gráfica do usuário (GUI) exigiu que fosse incluída a contagem de GUI nessa release. Devido a cada vez mais contagens estarem acontecendo em uma vasta variedade de situações, a release colocou ênfase na interpretação e prática das regras de contagem. Exemplos foram incluídos ao longo da documentação e estudos de casos complementaram o material. Finalmente, a Release 4.0 continuou a esclarecer e aumentar a consistência da contagem de pontos de função.
- Release 4.3 (janeiro de 2010). As regras e o processo da análise de pontos de função (APF) do IFPUG são concisos e fáceis de usar. Para refletir isto e tornar o Manual de Práticas de Contagem (CPM) cada vez mais atraente como um manual de referência, o Comitê de Práticas de Contagem (CPC) decidiu reestruturar o CPM 4.3 de modo a compatibilizá-lo com os padrões de formatação da ISO. Além disso, a Release 4.3 contém pequenas modificações e fornece novos exemplos, bem como explicações e

interpretações melhoradas para as regras existentes, que irão aumentar ainda mais a consistência entre contadores.

- Versão atual: CPM 4.3.1.

*Observação: outro grande grupo de usuários de pontos de função é a NESMA (Netherlands Software Metrics Users Association), sediado na Holanda. Suas ações e objetivos são bem próximos aos do IFPUG, inclusive com forte colaboração entre ambas, mas existem algumas diferenças, principalmente na contagem de projetos de melhoria.*

Hoje já existe um padrão ISO para métodos de Medição Funcional, sendo a norma ISO 14143 a que estabelece conceitos padronizados a serem seguidos por qualquer método de medição. Esta garante que todos os métodos de medição de tamanho funcional sejam baseados em conceitos similares e possam ser testados para assegurar que eles se comportam de maneira similar e da forma esperada por um método de medição. O método do CPM está de acordo com esse padrão, e é descrito na norma ISO 20926 (CPM 4.3).

### **Observações sobre o método de Pontos por Caso de Uso.**

Até o início da década de 90, existia uma noção de que APF não era adequada para sistemas orientados a objetos, apesar disto ser uma falácia. Além disso, processos que utilizam UML e Casos de Uso para especificar sistemas se popularizaram rapidamente. Assim, foi proposto o método de Pontos de por Caso de Uso.

O método consiste basicamente de:

1. Contar atores e casos de uso, identificado suas complexidades;
2. Calcular os PCUs não ajustados;
3. Ajustar os PCUs de acordo com a “complexidade técnica” e a “complexidade ambiental”.

A princípio os métodos de APF e PCU são parecidos, mas na prática há diferenças significativas:

1. PCU só pode ser aplicado a projetos que utilizem a técnica de casos de uso (como você faz para medir aplicações legadas, sem casos de uso, por exemplo?);
2. Devido ao processo de medição do PCU ser baseado em casos de uso, o método não pode ser empregado antes de concluída a análise de requisitos do projeto. APF permite você estimar o tamanho do software mesmo antes da finalização desta etapa.
3. É difícil conseguir medidas padronizadas e consistentes para PCUs, pois não existe um padrão único de escrita para casos de uso.
4. A determinação da “complexidade técnica” e “complexidade ambiental” está sujeita a certo grau de subjetividade que dificulta a aplicação consistente do método. O fator de ajuste da APF também tem o mesmo problema, mas o IFPUG tem várias diretrizes para minimizar o risco. Mesmo assim, o uso do fator de ajuste na APF tende a ser eliminado pelo IFPUG em breve.
5. Não há um grupo coeso para contagem de PCUs como existe para APF.

É um erro achar que o PCU será um método melhor que a APF se o projeto for especificado via casos de uso. APF pode ser (e é) aplicada a casos de uso, sem dificuldade alguma. Em suma, não benefício adicional do PCU sobre a APF.

### 3. Análise de Pontos de Função

Análise de Pontos de Função é um método para a medição de tamanho funcional de software a partir da visão do usuário.

#### O que é medido?

A análise de pontos de função mede o software quantificando as tarefas e serviços (isto é, funcionalidade) que o software fornece ao usuário, primordialmente com base no projeto lógico. Os objetivos da análise de pontos de função são medir:

- A funcionalidade implementada no software, que o usuário solicita e recebe;
- A funcionalidade impactada pelo desenvolvimento, melhoria e manutenção de software, independentemente da tecnologia utilizada na implementação.

#### Para quê medir?

As organizações podem aplicar este Padrão Internacional para medir o tamanho de um produto de software a fim de:

- dar suporte à análise de qualidade e produtividade;
- estimar o custo e recursos requeridos para o desenvolvimento, melhoria e manutenção do software;
- fornecer um fator de normalização para a comparação de software;
- determinar o tamanho de um pacote de aplicação adquirido, por meio do dimensionamento funcional de todas as funções incluídas no mesmo;
- ajudar os usuários a determinar o benefício provido por um pacote de aplicação para a sua organização, por meio do dimensionamento funcional das funções que correspondam especificamente aos seus requisitos.

#### Quais são as vantagens?

O processo de análise de pontos de função é:

- Suficientemente simples para minimizar o custo adicional introduzido pelo processo de medição;
- Uma medida consistente entre diversos projetos e organizações.

### 3.1. Aplicabilidade

**APF pode ser aplicado a todos os domínios funcionais.** O IFPUG publica constantemente artigos fornecendo orientações para a utilização do método em ambientes e domínios diferentes.

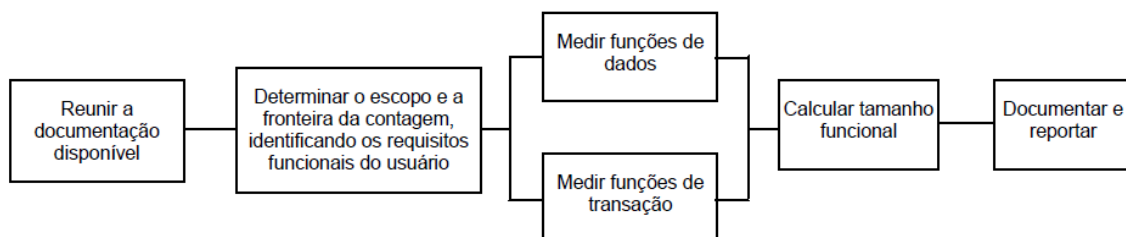
Os analistas de pontos de função do IFPUG identificaram diferentes taxas de entrega (horas para entregar um ponto de função) relacionadas à construção de aplicações em diferentes domínios funcionais, calibradas para diversos tamanhos de projeto e complexidade do software.

Existe uma organização, a ISBSG (*International Software Benchmarking Standards Group*), cuja missão é “manter um repositório público de métricas de projetos de software”. Neste repositório existem informações de mais de 6.000 projetos de software de dezenas de países, o que possibilita a análise de produtividade, boas práticas, custos, etc., tudo organizado por domínio e tecnologias utilizadas.

## 4. Processo de Medição

Para conduzir uma contagem de pontos de função devem ser executadas as seguintes atividades, a fim de identificar e classificar os componentes funcionais básicos (ALI, AIE, EE, SE, CE):

- a) Reunir a documentação disponível ;
- b) Determinar o escopo e a fronteira da contagem, identificando os Requisitos Funcionais do Usuário;
- c) Medir as funções de dados
- d) Medir as funções de transação
- e) Calcular o tamanho funcional
- f) Documentar a contagem de pontos de função
- g) Reportar o resultado da contagem de pontos de função



## 5.1 Reunir a documentação disponível.

A documentação de suporte a uma contagem de pontos de função deve descrever a funcionalidade entregue pelo software ou a funcionalidade impactada pelo projeto de software medido.

Deve ser obtida documentação suficiente para conduzir a contagem de pontos de função, ou acesso a especialistas no assunto capazes de fornecer informações adicionais para suprir quaisquer falhas na documentação.

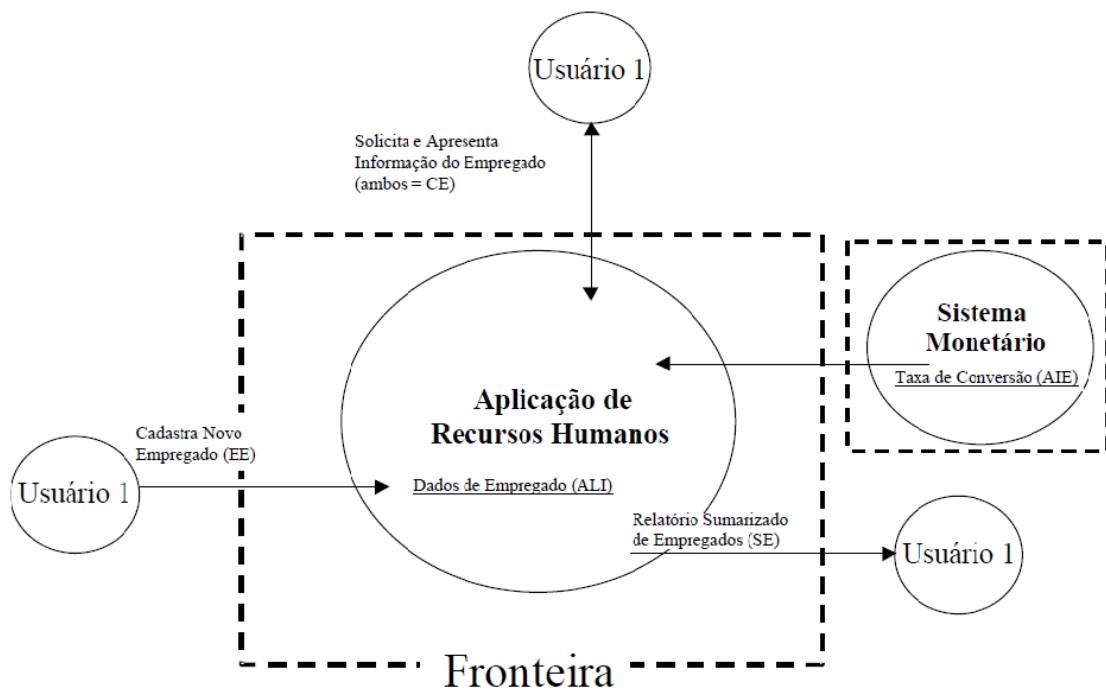
Geralmente, não há um único documento que resolva todas as necessidades de informação da medição; logo, o mais comum é que a medição seja feita com base na análise de mais de um tipo de documento.

Em geral, os itens a seguir são úteis quando se faz alguma medição de tamanho funcional:

- Documentos de requisitos
- Diagrama de entidades
- Modelos de objetos
- Modelos de dados
- Arquivo e esquemas banco de dados (com lógica, atributos necessários do usuário identificados)
- Acordos de Interface com descrições de entradas em lote/arquivos de transação e interfaces de/para outras aplicações
- Exemplos de relatórios, telas online e outras interfaces de usuário
- Demonstração da operação de aplicação
- Uma ou mais especialistas na aplicação (para a aplicação que está sendo medida)
- Um ou mais clientes/usuários de aplicação (passíveis de serem consultados durante o processo de medição)
- Guia de usuário, manual de treinamento e ajuda da aplicação
- Documentação do projeto do sistema
- Especificações funcionais

## 5.2 Determinar o escopo e a fronteira da contagem, identificando os Requisitos Funcionais do Usuário.

Exemplo de Aplicação (Recursos Humanos):



Para determinar o escopo e fronteira da contagem e identificar os Requisitos Funcionais do Usuário, devem ser executadas as seguintes atividades:

**a) Identificar o propósito da contagem**

*NOTA 1 Uma contagem de pontos de função é conduzida a fim de fornecer uma resposta para uma questão de negócio, sendo a questão de negócio que determina o propósito.*

*NOTA 2 O propósito da contagem determina o escopo da contagem.*

*EXEMPLO 1 O propósito da contagem poderia ser a determinação do tamanho de uma release de software específica.*

*EXEMPLO 2 O propósito da contagem poderia ser a determinação do tamanho de uma aplicação, como parte do esforço da organização para determinar o tamanho de seu portfólio de software.*

**b) Identificar o tipo de contagem, com base no propósito, como um dos seguintes:**

- 1) uma contagem de pontos de função de projeto de desenvolvimento;
- 2) uma contagem de pontos de função de aplicação;
- 3) uma contagem de pontos de função de um projeto de melhoria.

**Um projeto de desenvolvimento** é um projeto para desenvolver e fornecer a primeira versão de um software. O tamanho funcional do projeto de desenvolvimento é uma medida de funcionalidade oferecida aos usuários com a primeira instalação do software. De certa forma é uma contagem estimada, pois pode haver mudanças durante o desenvolvimento que modifiquem a contagem.



Um **projeto de melhoria** é um projeto para desenvolver e entregar manutenções no software. O tamanho funcional do projeto de melhoria é uma medida das funcionalidades adicionadas, alteradas e excluídas na conclusão de um projeto de melhoria. Segundo a ISO, as manutenções podem ser:

- Manutenção Adaptativa: acompanhar mudanças no ambiente, por exemplo, para acompanhar uma nova versão de sistema operacional;
- Manutenção Corretiva: correção de erros e problemas;
- Manutenção Perfectiva: melhorias em geral (documentação, desempenho, usabilidade, etc.).

Um tamanho funcional de uma **aplicação** é uma medida de funcionalidade que uma aplicação oferece ao usuário. Ela também é chamada de *baseline* (versão estabilizada) ou tamanho funcional instalado. Este tamanho fornece uma medida de funções atuais que o aplicativo fornece ao usuário. O número é inicializado quando o projeto de desenvolvimento da contagem de ponto de função é finalizado. É atualizado toda vez que um projeto de melhoria finalizado alterar funções da aplicação

**c) Determinar o escopo da contagem, com base no propósito e tipo de contagem.**

O escopo define quais funções serão incluídas na contagem, se ela abrangerá um ou mais sistemas ou apenas parte de um sistema. Por exemplo, você pode decidir se vai contar uma ou mais aplicações ou então apenas parte de uma aplicação. Além disso, pode abranger:

- Todas as funcionalidades disponíveis;
- Apenas as funcionalidades efetivamente utilizadas pelo usuário;
- Apenas algumas funcionalidades específicas (relatórios, transações cadastrais, etc.).

**d) Determinar a fronteira de cada aplicação contida no escopo da contagem com base na visão do usuário e não em considerações técnicas.**

A **fronteira** é uma interface conceitual entre o software sob estudo e seus usuários. A fronteira (também chamada de fronteira da aplicação):

- Define o que é externo à aplicação;
- Indica a fronteira entre o software que está sendo medido e o usuário;
- Atua como uma 'membrana' através da qual os dados processados pelas transações (EEs, SEs e CEs) passam para dentro e para fora da aplicação;
- Envolve os dados lógicos mantidos pela aplicação (ALIs);
- Auxilia na identificação dos dados lógicos referenciados mas não mantidos pela aplicação (AIEs);
- Depende da visão externa do negócio do usuário da aplicação. É independente de considerações de técnicas e/ou implementação.

Um **usuário** é qualquer pessoa ou coisa que se comunica ou interage com o software a qualquer momento. Podem ser hardware, outras aplicações, outros sistemas, etc. Se um

usuário fosse apenas uma Pessoa, não seria possível medir PFs para sistemas que não têm interface gráfica com o usuário.

A visão do usuário representa uma descrição formal das necessidades dos negócios do usuário, na linguagem do usuário. Os desenvolvedores traduzem a informação do usuário para informações em linguagem técnica a fim de prover uma solução.

A visão do usuário:

- É uma descrição das funções do negócio;
- Pode ser feito por declaração verbal pelo usuário através de seu ponto de vista;
- É aprovada pelo usuário;
- Pode ser usada para medir o tamanho funcional;
- Pode variar na forma física (ex., catálogo de transações, propostas, documento de requisitos, especificações externas, especificações detalhadas, manuais do usuário).

*EXEMPLO: Se o propósito for estimar o custo de uma melhoria nas aplicações de Recursos Humanos (RH) e Benefícios, então:*

*- o tipo de contagem é uma contagem de projeto de melhoria;*

*- o escopo deverá incluir as funções transacionais e de dados incluídas, alteradas ou excluídas referentes tanto à aplicação de RH quanto à de Benefícios, assim como quaisquer requisitos de conversão referentes a cada aplicação;*

*- a visão do usuário é que RH e Benefícios são áreas funcionais diferentes, por esse motivo são aplicações separadas;*

*- existe uma fronteira entre as aplicações de RH e de Benefícios, assim como entre cada aplicação e o usuário.*

A visão do usuário pode ser materializar através de alguns artefatos ou insumos do projeto, tais como (exemplos):

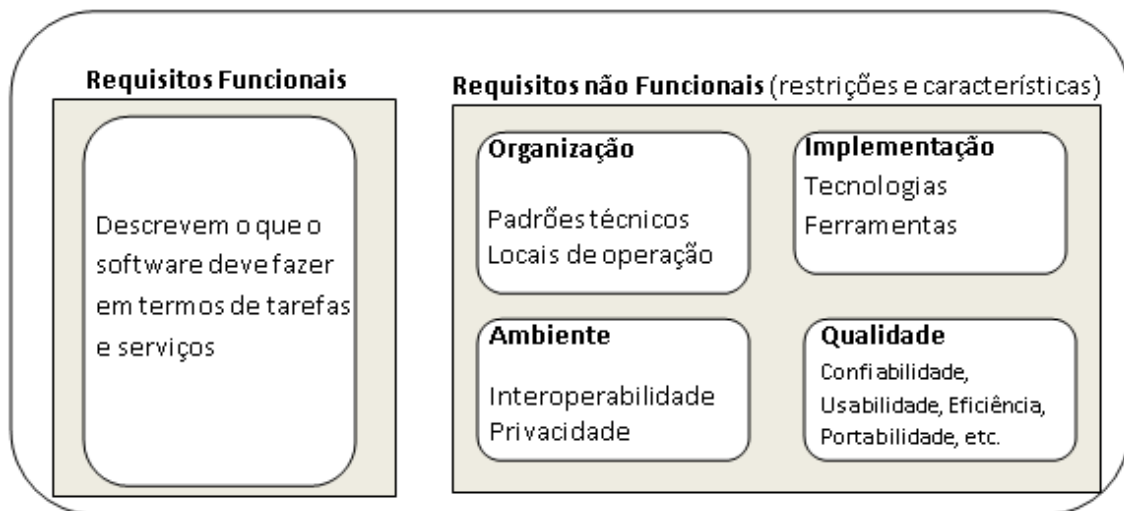
- História de Usuário;
- Proposta de Projeto;
- Especificação de necessidades de negócio;
- Especificação de casos de uso;
- Protótipos de interface, etc.

**e) Os requisitos do usuário podem incluir uma mistura de requisitos funcionais e não-funcionais; identificar quais requisitos são funcionais e excluir os não-funcionais.**

Requisitos Funcionais são aqueles que capturam o que o software deve fazer em termos de tarefas e serviços. <<Ver o curso de Requisitos para exemplos de Requisitos Funcionais>>.

Requisitos não funcionais são restrições ou qualidades e características do sistema.

Hierarquia de classificação de requisitos segundo a norma ISO/IEC 14143:



### 5.3 Medir as funções de dados

Uma função de dados representa a funcionalidade fornecida ao usuário para atender suas necessidades internas e externas de armazenamento de dados.

Toda a funcionalidade de dados dentro do escopo da contagem deve ser avaliada para identificar cada grupo lógico de dados.

Uma função de dados pode ser um arquivo lógico interno ou um arquivo de interface externa. O termo *arquivo* aqui não significa arquivo físico ou tabela. Nesse caso, arquivo se refere a um grupo de dados logicamente relacionados e não à implementação física destes grupos de dados. A chave é pensar em um modelo conceitual (entidades) e não no modelo lógico (tabelas).

*Exemplo: mesmo que uma Entidade (BD) esteja distribuída entre várias tabelas devido a técnicas de normalização, o Arquivo Lógico ainda é um só.*

Devem ser executadas as seguintes atividades para medir as funções de dados:

#### 1) Identificar funções de dados (ALI e AIE);

**Um arquivo lógico interno (ALI)** é um grupo de dados ou informações de controle, reconhecido pelo usuário e mantido dentro da fronteira da aplicação sendo medida. A principal intenção de um ALI é armazenar dados mantidos por um ou mais processos elementares da aplicação sendo medida.

O diagrama simplificado anterior apresenta um grupo de dados relacionado a empregado mantido dentro da Aplicação de Recursos Humanos como um exemplo de Arquivo Lógico Interno.

**Um arquivo de interface externa (AIE)** é um grupo de dados ou informações de controle, reconhecido pelo usuário, e que é apenas referenciado pela aplicação sendo medida, mas que são mantidos dentro da fronteira de outra aplicação. A principal intenção de um AIE é armazenar dados referenciados por um ou mais processos elementares da aplicação sendo medida. Isto significa que um AIE contado para uma aplicação deve ser *obrigatoriamente* um ALI em alguma outra aplicação.

O diagrama simplificado anterior mostra informação de taxa de conversão mantida pelo Sistema Monetário e que é referenciado pela Aplicação de Recursos Humanos como um exemplo de Arquivo de Interface Externa.

## **2) Contar DERs e RLRs para cada função de dados;**

**Dado Elementar Referenciado (tipo de dado elementar) DER:** atributo único, reconhecido pelo usuário e não repetido.

Por exemplo, a Aplicação A pode identificar e utilizar um endereço como: rua, cidade, estado e CEP. A Aplicação B pode ver o endereço como um bloco de dados sem considerar os componentes individuais. A Aplicação A contaria quatro DERs; a Aplicação B contaria um DER.

Por exemplo, a Aplicação A pode contar Data como três DERs, dia, mês e ano. Já a Aplicação B pode contar Data como um único DER, se os três campos sempre forem manipulados juntos.

Por exemplo, a Aplicação X mantém e/ou referencia um ALI que contém CPF, Nome, Rua, Caixa Postal, Cidade, Estado e CEP. A Aplicação Z mantém e/ou referencia Nome, Cidade e Estado. A Aplicação X contaria sete DERs; a Aplicação Z contaria três DERs.

Importante lembrar que não são contadas as várias ocorrências de um tipo de dado. Por exemplo, caso uma entidade tenha um conjunto de Telefones, mesmo que sejam vários, o conjunto é contado como um único tipo de dado.

**Registro Lógico Referenciado (tipo de registro elementar) RLR:** subgrupo de dados elementares referenciados, reconhecido pelo usuário, contido em uma função de dados.

Conte um RLR para cada função de dados (por padrão cada função de dado tem um subgrupo de DERs para ser contado como um RLR)

Por exemplo, em uma Aplicação de Recursos Humanos, são armazenados dados sobre os Funcionários da empresa. Ocorre que, cada Funcionário, pode ser Mensalista ou Diarista.

Se o analista fosse modelar isso em classes, provavelmente obteria três classes diferentes: a superclasse Funcionário e as subclasses Funcionário Mensalista e Funcionário Diarista, sendo as duas últimas subtipos da primeira. Nesse caso, deve-se contar apenas um ALI, no caso o Funcionário, que contém três RLRs: Funcionário (padrão), Funcionário Mensalista e Funcionário Diarista.

Podemos ainda mapear esses termos para a análise de pontos de função, conforme mostrado na seguinte matriz:

<b>Conceito da Modelagem de Dados</b>	<b>Termo da Modelagem de Dados</b>	<b>Termo de Base de Dados Relacional</b>	<b>Termo da APF</b>	<b>Conceito da APF</b>
Menor unidade de dado com nome que tem significado para o mundo real	Item de Dados	Atributo ou Coluna	Tipo de Dado Elementar (DER)	Um tipo de dado elementar (DER) é um campo único, não-repetido, reconhecido pelo usuário
Grupos de itens relacionados os quais são tratados como uma unidade	Registro	Linha ou Tupla	Tipo de Registro Elementar (RLR)	Um tipo de registro elementar (RLR) é um subgrupo de elementos de dados reconhecido pelo usuário e armazenado em um ALI ou AIE
Coleção de registros de um único tipo	Arquivo	Tabela	Arquivo Lógico (Arquivo Lógico Interno - ALI ou Arquivo de Interface Externa - AIE)	Arquivo refere-se a grupos de dados logicamente relacionados e não à implementação física desses grupos de dados

**3) Determinar a complexidade funcional de cada função de dados, de acordo com a seguinte tabela:**

### Complexidade das funções de dados

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

Analisando esta tabela, pode-se perceber que não é necessário contar EXATAMENTE o número de DERs e RLRs em uma aplicação. Na verdade, isto só é relevante quando a quantidade está nos limites das faixas dessa tabela de complexidade.

Além disso, uma eventual classificação incorreta da complexidade da função afeta de forma bem limitada o resultado final da medição. Mais importante é identificar corretamente a quantidade de funções (tanto de dados como de transações).

- 4) **Determinar o tamanho funcional de cada função de dados, de acordo com a seguinte tabela:**

### Tamanho das funções de dados

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

**Importante:** algumas entidades não são contadas como Arquivos Lógicos (Funções de Dados). Apesar de armazenarem dados, alguns arquivos são uma implementação de requisitos técnicos, e não devem influenciar o tamanho funcional da aplicação.

Exemplos:

1. **Dados de Código** – também chamados de Metadados, em geral são especificados pelo próprio desenvolvedor em resposta a requisitos técnicos.
  - a. Normalmente consistem do campo-chave e um ou dois outros campos.
  - b. Raramente mudam;
  - c. Podem ser dados estáticos, dados de substituição ou mesmo faixas de valores válidos;
  - d. Exemplo: Arquivos que armazenam a sigla e o nome das unidades da federação;
2. **Arquivos de Índice** – utilizados para melhorar o desempenho na recuperação de dados.
3. **Arquivos Temporários** – que, por exemplo, são utilizados em várias etapas de um processamento batch.
4. **Dados Distribuídos** – que são “espalhados” em vários arquivos físicos, porém representam uma única entidade lógica.
5. E outras (ver manual)...

## 5.4 Medir as funções de transação

Uma função de transação é um processo elementar que fornece funcionalidade ao usuário para processamento de dados.

Toda a funcionalidade de transação dentro do escopo da contagem deve ser avaliada, a fim de identificar cada processo elementar único

Uma função de transação pode ser uma entrada externa, saída externa ou consulta externa.

### 1) Identificar cada processo elementar requerido pelo usuário;

**Processo elementar:** menor unidade de atividade significativa para o usuário. Constitui uma transação completa, e é autocontida. Ao final de sua execução, deixa o negócio da aplicação em um estado consistente.

Por exemplo, os requisitos do usuário para adicionar um funcionário incluem informações de salário e dependentes. Um funcionário não terá sido criado se não forem incluídas todas as respectivas informações. Incluir separadamente apenas parte das informações deixará o negócio de incluir um funcionário em um estado inconsistente. Se forem incluídos tanto o salário do empregado quanto as informações do(s) dependente(s), a unidade de atividade será concluída e o negócio será deixado em um estado consistente.

*EXEMPLO 1 Um Requisito Funcional do Usuário pode estabelecer que deve ser fornecida uma função para Manter Informações de Empregado. Esse requisito é decomposto em unidades de trabalho menores tais como Incluir Empregado, Alterar Empregado e consultar Empregado.*

*EXEMPLO 2 Os requisitos individuais podem estabelecer a necessidade de incluir diversos tipos de informações de empregado (por exemplo, endereço, salário e informações de dependentes), mas a menor unidade de atividade significativa para o usuário é Incluir Empregado.*

**2) Classificar cada processo elementar como uma função de transação de Entrada Externa (EE), Saída Externa (SE) ou Consulta Externa (CE);**

**Uma entrada externa (EE)** é um processo elementar que processa dados ou informações de controle recebidas de fora da fronteira da aplicação. A intenção primária de uma EE é manter um ou mais ALLs e/ou alterar o comportamento do sistema.

O diagrama simplificado anterior mostra o processo de cadastrar um novo empregado na Aplicação de Recursos Humanos como um exemplo de uma Entrada Externa.

Um Processo Elementar deve ser classificado como Entrada Externa se o mesmo:

- 1) Incluir lógica de processamento para receber dados ou informações de controle que entrem pela fronteira da aplicação;
- 2) Tiver um das seguintes intenções primárias
  - i. Manter um ou mais ALLs, ou
  - ii. Alterar o comportamento da aplicação.

Em geral, o nome das transações de Entrada Externa possuem termos bem característicos, como incluir, alterar, excluir, editar, registrar, gravar, carregar, etc.

Exemplo de EE:

- Janela que permite adicionar, excluir ou alterar registros em um arquivo lógico.

**Uma saída externa (SE)** é um processo elementar que envia dados ou informações de controle para fora da fronteira da aplicação e inclui processamento adicional além daquele existente em uma consulta externa. A intenção primária de uma saída externa é apresentar dados ao usuário através de lógica de processamento que não seja apenas recuperação de dados ou informação de controle. A lógica de processamento deve contar ao menos uma fórmula matemática ou cálculo, e/ou criar dados, e/ou manter um ou mais ALLs, e/ou alterar o comportamento do sistema.

O diagrama simplificado anterior mostra o processo de gerar um relatório que sumariza todos os empregados da Aplicação de Recursos Humanos como um exemplo de Saída Externa.

Um Processo Elementar deve ser classificado como Saída Externa se o mesmo tiver a intenção primária de apresentar informações ao usuário e incluir pelo menos uma das seguintes formas de lógica de processamento:

- 1) Cálculos matemáticos são executados;
- 2) Um ou mais ALLs são atualizados;
- 3) Dados derivados são criados (exemplos: totalizações, percentagens, médias, etc.).



4) O comportamento da aplicação é alterado.

Exemplos de SE:

- Relatórios com totalização de dados;
- Relatórios que também atualizam arquivos;
- Consultas com cálculos ou apresentação de dados derivados;
- E outros...

**Uma consulta externa (CE)** é um processo elementar que envia dados ou informações de controle para fora da fronteira da aplicação. A intenção primária de uma consulta externa é apresentar dados ao usuário através de recuperação de dados ou informação de controle. A lógica de processamento não contém fórmula matemática, nem cálculo, nem cria dados derivados. Nenhum ALI é mantido durante o processamento, nem o comportamento do sistema é alterado.

O diagrama simplificado anterior mostra o processo de solicitar dados de empregado como um exemplo de Consulta Externa.

Um Processo Elementar deve ser classificado como Consulta Externa se o mesmo tiver a intenção primária de apresentar informações ao usuário e:

- 1) Referenciar uma função de dados para recuperar dados ou informações de controle;
- 2) Não satisfizer os critérios para ser classificado como uma SE.

Exemplos de CE:

- *Drop-Downs*, desde que recuperem dados de arquivos lógicos dinamicamente. *Drop-Downs* estáticos não são considerados.
- Menus gerados dinamicamente de acordo com alguma configuração da aplicação.

A Tabela a seguir apresenta um resumo do relacionamento entre a intenção primária e o tipo de função de transação.

#### **Relacionamento entre a intenção primária e o tipo de função de transação**

Função	Tipo de função de transação		
	EE	SE	CE
Alterar o comportamento da aplicação	IP	F	N/A
Manter um ou mais ALIs	IP	F	N/A
Apresentar informações ao usuário	F	IP	IP

legenda

IP            a intenção primária do tipo de função de transação  
F            uma função do tipo de função de transação que às vezes  
              está presente, mas que não é a intenção primária  
N/A          o tipo de função de transação não pode executar este  
              tipo de função

**3) Contar os ALRs - Arquivos Lógicos Referenciados e DERs - Dados Elementares Referenciados;**

Para cada função de transação, um **ALR** deve ser contado para cada função de dados (ALI ou AIE) única que for acessada (lida e/ou gravada) pela função de transação. Podem ser ALIs que são lidos/gravados pela aplicação ou AIEs que são lidos de outra aplicação.

Além disso, a fim de contar **DERs** para uma função de transação, as seguintes atividades devem ser executadas

- a) Revisar tudo o que acesse (entre e/ou saia) a fronteira,
- b) Contar um DER para cada atributo único, reconhecido pelo usuário e não repetido que acesse (entre e/ou saia) a fronteira durante o processamento da função de transação,

Exemplo: para atualizar um Cliente, o usuário deve fornecer o seu nome e CPF (dois DERs).

**4) Determinar a complexidade funcional de cada função de transação;**

A complexidade funcional de cada função de transação será determinada utilizando-se o número de ALRs e DERs, em conformidade com as tabelas a seguir:

**Tabela 6 — Complexidade funcional das EE**

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

**Tabela 7 — Complexidade funcional das SE e CE**

		DERs		
		1 – 5	6 – 19	> 19
ALRs	0 – 1	Baixa	Baixa	Média
	2 – 3	Baixa	Média	Alta
	> 3	Média	Alta	Alta
NOTA		Uma CE tem no mínimo 1 ALR.		

**5) Determinar o tamanho funcional de cada função de transação;**

O tamanho funcional de cada função de transação será determinado utilizando-se o tipo e a complexidade funcional, de acordo com a tabela a seguir:

## Tamanho das funções de transação

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

### 5.5 Calcular o tamanho funcional

A fórmula para calcular a contagem final de pontos de função depende do tipo de contagem.

O tamanho funcional de um **projeto de desenvolvimento** deverá ser calculado utilizando-se a Fórmula:

$$\text{DFP} = \text{ADD} + \text{CFP}^*$$

Onde,

- DFP é a contagem de pontos de função do projeto de desenvolvimento;
- ADD é o tamanho das funções a serem entregues ao usuário pelo projeto de desenvolvimento;
- CFP é o tamanho da funcionalidade de conversão.

\* São funções construídas e entregues pelo projeto (de desenvolvimento ou melhoria) para serem usadas no momento da instalação do projeto para converter dados ou fornecer outros requisitos de conversão especificados pelo usuário, como relatórios de verificação da conversão. A característica destas funções é que elas são descartadas após o seu uso, não fazendo parte da aplicação após sua instalação. Quando o sistema entra em operação, essas funções não são mais necessárias. Talvez um nome melhor para essas funções fosse "funções de transição".

Funções de conversão não são técnicas. São realizadas a partir do ponto de vista do usuário. Por exemplo, a migração de plataforma tecnológica não é contada como uma função de conversão.

*Exemplo: ao instalar uma nova aplicação de RH, o usuário precisa que dados sejam migrados de uma outra aplicação legada e populados na nova aplicação.*

O tamanho funcional de uma **aplicação**, medido após o projeto de desenvolvimento, ou a qualquer tempo no ciclo de vida da aplicação deverá ser calculado utilizando-se a fórmula a seguir:

$$\text{AFP} = \text{ADD}$$

Onde,

- AFP é a contagem de pontos de função da aplicação;
- ADD é o tamanho das funções a serem entregues ao usuário pelo projeto de desenvolvimento (excluído o tamanho de qualquer funcionalidade de conversão), ou a funcionalidade existente no momento da contagem da aplicação.

O tamanho funcional de um **projeto de melhoria** deverá ser calculado utilizando-se a fórmula a seguir:

$$\text{EFP} = \text{ADD} + \text{CHGA} + \text{CFP} + \text{DEL}$$

Onde,

- EFP é a contagem de pontos de função do projeto de melhoria;
- ADD é o tamanho das funções incluídas pelo projeto de melhoria;
- CHGA é o tamanho das funções alteradas pelo projeto de melhoria – conforme as mesmas estão / estarão após a implementação;
- CFP é o tamanho da funcionalidade de conversão;
- DEL é o tamanho das funções excluídas pelo projeto de melhoria.

**Observações sobre o Fator de Ajuste:** para adequar-se à norma ISO 14143 (medição do tamanho funcional), o IFPUG tornou o fator de ajuste opcional na aplicação da técnica de análise de pontos de função. Mesmo antes disso, vários usuários da APF já não utilizavam o fator de ajuste. O propósito do fator de ajuste é medir requisitos gerais da aplicação (não funcionais). Ele ajusta os pontos de função em + ou – 35% de acordo com a influência de 14 características gerais.

## 5.6 Documentar a contagem de pontos de função

Nem sempre o que interessa na contagem de pontos de função é apenas o número final que reflete o tamanho medido. Não basta entregar ao cliente o número final da medição (ou estimativa). É preciso que junto seja entregue também toda a memória de cálculo do processo de análise que resultou naquele número. Caso contrário, quem recebe a medição não tem como conferir se o resultado está correto ou não. Dessa forma, é muito comum que a medição seja entregue com a planilha de contagem final, permitindo que todo processo de análise seja verificado.

As vantagens disso são as seguintes:

- Agrega valor e confiabilidade à medição;
- Facilita um eventual processo de auditoria
- Minimiza os erros do analista responsável pela medição.

O nível de documentação deve estar alinhado ao propósito da contagem. Por exemplo, se o propósito for estimar uma ordem de grandeza de custo para o projeto, qual o sentido de detalhar ao máximo a medição? É importante lembrar que um dos objetivos fundamentais da medição é ser simples. Logo, o nível de documentação deve ser bem dosado para que equilibre o esforço a ser gasto na medição e a necessidade de informação que lhe agregue valor. Lembre-se que uma medição mais detalhada implica mais tempo e custo para esta atividade.

Assim, apesar do nível de documentação ser previamente acordado entre as partes interessadas na medição, no mínimo, a contagem de pontos de função deve ser documentada como segue:

- O propósito e o tipo da contagem;
- O escopo da contagem e a fronteira da aplicação;
- A data da contagem;
- Uma lista de todas as funções de dados e de transação, incluindo o respectivo tipo e complexidade, bem como o número de pontos de função atribuído a cada uma;
- O resultado da contagem;
- Quaisquer suposições feitas e questões resolvidas.

Além de outras informações opcionais...

## 5.7 Reportar o resultado da contagem de pontos de função

A prática de reportar consistentemente os resultados das contagens de pontos de função permitirá que os leitores identifiquem o padrão com o qual as mesmas mantêm conformidade.

Os resultados que mantenham conformidade com este Padrão Internacional deverão ser reportados como segue:

### **S FP (IFPUG–IS)**

Onde,

- S é o resultado da contagem de pontos de função;
- FP é a unidade de tamanho do método FSM do IFPUG;
- IS é este Padrão Internacional (ISO/IEC 20926:200x).

EXEMPLO 250 FP (IFPUG-ISO/IEC 20926:200x)

Os resultados que mantiverem conformidade com uma customização local deste Padrão Internacional deverão ser reportados como:

### **S FP (IFPUG–IS–c)**

Onde,

c representa um ou mais caracteres indicando que o resultado não mantém conformidade plena com este Padrão Internacional.

EXEMPLO 250 FP (IFPUG–ISO/IEC 20926:200x–a)

## 6. Fator de Ajuste

No cálculo de Pontos de Função brutos, ou não ajustados, NÃO são considerados aspectos técnicos ou baseados em requisitos não funcionais. Mesmo assim, é sabido que existem fatores não funcionais que afetam o “tamanho” da aplicação.

O Fator de Ajuste é uma tentativa de “compensar” ou “ajustar” alguns pontos de função na aplicação baseado em Características Gerais do Sistema que tentam refletir a complexidade do sistema. Estas características gerais afetam a aplicação como um todo e podem modificar a contagem de acordo com diversos “níveis de influência”.

É importante lembrar que o IFPUG não submeteu o Fator de Ajuste para fins de conformidade ao processo de medição previsto pela norma ISO 14143. Não obstante, ele ainda faz parte do Manual de Contagem como um apêndice e suas regras ainda são cobradas na prova de certificação. Mesmo assim, a tendência é a de extinção do fator de ajuste no futuro próximo (a ser explicado mais à frente).

As 14 características gerais do sistema são:

1. Comunicação de Dados
2. Processamento Distribuído
3. Performance
4. Configuração Intensamente Utilizada
5. Volume de Transações
6. Entrada de Dados On-Line
7. Eficiência do Usuário Final
8. Atualização On-Line
9. Processamento Complexo
10. Reusabilidade
11. Facilidade de Instalação
12. Facilidade de Operação
13. Múltiplos Locais
14. Facilidade de Mudança

Com base nos requisitos estabelecidos pelo usuário, cada característica geral do sistema (CGS) deve ser avaliada em termos de seus níveis de influência (NI) em uma escala de 0 a 5.

<b>Pontue como</b>	<b>Influência no Sistema</b>
0	Não presente ou sem influência
1	Influência Mínima
2	Influência Moderada
3	Influência Média
4	Influência Significativa
5	Forte influência

Para diminuir a subjetividade, cada uma das descrições das características gerais do sistema seguintes inclui diretrizes para a determinação do nível de influência.

Cada diretriz contém uma definição da CGS, regras para determinação do nível de influência e, em situações nas quais a regra requer esclarecimento adicional, são fornecidas dicas para ajudar a aplicar as regras consistentemente em todas as plataformas.

Não se pretende que as dicas cubram todas as situações. Ao invés disso, a intenção é que as mesmas forneçam orientação adicional para a determinação do nível de influência apropriado.

#### **Exemplo (CGS 06):**

A característica Entrada de Dados On-line descreve os níveis segundo os quais os dados são informados ou recuperados através das transações interativas. Interfaces on-line com o usuário para entrada de dados, funções de controle, relatórios e consultas são fornecidos pela aplicação.

0. Todas as transações são processadas de modo batch
1. 1% a 7% das transações são interativas
2. 8% a 15% das transações são interativas
3. 16% a 23% das transações são interativas
4. 24% a 30% das transações são interativas
5. Mais de 30% das transações são interativas

Neste caso, para a realidade das aplicações atuais, a grande maioria delas pontuará com 5. Percebe-se aqui uma grande defasagem nas diretrizes do IFPUG com a realidade atual.

Determinados os níveis de influência das 14 características gerais, o fator de ajuste é calculado com a seguinte fórmula:

$$\text{VAF} = (\text{TDI} \times 0,01) + 0,65$$

Onde TDI é o somatório dos níveis de influência das características gerais (pode chegar a 70). Sendo assim, o fator de ajuste pode resultar numa variação de mais ou menos 35% na contagem de pontos de função.

#### **Críticas ao Fator de Ajuste**

Mesmo antes do Fator de Ajuste se tornar opcional, poucos usuários utilizavam-no. Algumas razões levaram a isso

1. **A grande subjetividade na interpretação das CGS.** Por exemplo, a CGS 07 lida com “facilidade de uso do sistema por parte do usuário”. É difícil determinar isso objetivamente, mesmo com diretrizes.
2. **A constatação que algumas CGSs estão desatualizadas.** Como é o caso, claramente, da CGS 06.
3. **Apesar de serem subjetivas, podem afetar significativamente a contagem.** Uma pequena diferença de interpretação pode fazer com que sua contagem seja bastante modificada. Basta lembrar que cada mudança no nível de influência afeta o tamanho final em 1%.
4. **Não é apropriado que todas as CGSs possuam o mesmo peso para nível de influência, nem que o limite máximo de uma característica seja 5%.** Por exemplo, para determinados projetos, Reusabilidade pode ter um impacto muito maior do que 5%.
5. **Existem requisitos importantes que não são contemplados por nenhuma das 14 CGSs.** Por exemplo, você até encontra itens relacionados a Segurança da Informação na CGS 09, mas de maneira insuficiente, muito superficial.