



Um guia de rápido aprendizado para a  
**Feature-Driven Development**

**mar.2007**

**aXmagno**

Alexandre Magno Figueiredo  
axmagno@gmail.com

# O que é FDD?

**Feature-Driven Development (FDD)** é uma metodologia ágil para o processo de engenharia de software, que foi elaborada com foco na entrega frequente de “software funcionando” para os clientes e na utilização de boas práticas durante o ciclo de seu desenvolvimento.

Uma característica marcante da FDD é o fato dela favorecer fortemente o envolvimento de clientes (interno ou externo) ao processo de planejamento e desenvolvimento do software.

**Feature-Driven Development (FDD)** é um processo de desenvolvimento de software iterativo e incremental.

Diferentemente de outras metodologias, a FDD não é extremamente focada na programação ou no modelo, mas sim utiliza o bom senso para abstrair o melhor dos dois mundos.

# O que não é FDD?

A FDD não é uma metodologia descrita em uma coleção com 30(trinta) volumes de livros. Portanto, ela **não é uma “bíblia”** a ser seguida por sua equipe de desenvolvimento.

A FDD **não é uma metodologia de gerenciamento de projetos** de software. Apesar de, em suas práticas, existirem atividades relacionadas a esse fim, a FDD tem como principal foco cobrir o processo da engenharia de software, e não do gerenciamento.

A FDD **não é uma bala de prata**, portanto, ela **não resolverá todos os problemas do mundo, ou mesmo os da sua empresa**. No entanto, a FDD o auxiliará a tornar o processo de engenharia de software da sua empresa mais eficiente, e a criar em sua equipe uma cultura voltada à rápida entrega de software para o cliente.

# Por que devo usar FDD?

Porque em projetos de software **precisamos mais do que apenas código escrito** e funcionando.

Porque a FDD oferece **planejamento e modelo na medida certa!** Sem exageros, mas também sem ausência.

Porque os processos da FDD **favorecem a aproximação de especialistas de negócio, gerentes e desenvolvedores.**

Porque FDD é ágil, e nos permite realizar **entregas frequentes** aos nossos clientes.

Porque decompondo o produto em funcionalidades estou mais próximo de estar sempre **entregando algo de valor para o meu cliente.**

Porque utilizando a prática de proprietários de classes, garanto a responsabilidade, especialidade e familiaridade dos desenvolvedores com determinado pedaço de código. Com isso, tenho sempre **manutenção rápida e de qualidade** em qualquer parte do meu sistema.

Porque os mecanismos de visualização de progresso da FDD, tais como o Parking Lot, nos permitem ter, a qualquer momento, uma **visualização exata de onde estamos.**

Porque as práticas da FDD contribuem para um grande **envolvimento do cliente no projeto**, fazendo com que, rapidamente, este passe a chamar o “seu” projeto de “NOSSO PROJETO”.

Porque a inspeção de código e de design, que é uma das práticas da FDD, nos **garante qualidade** no produto final.

**Porque FDD funciona!**

# Quem é quem?



Olá! Eu sou a **Gerente do Projeto**, e como tal sou responsável por todos os assuntos administrativos do projeto, o que inclui o gerenciamento de recursos, orçamento, equipamentos e outros. Minha principal meta é fornecer subsídios para que nenhum fator externo atrapalhe a produtividade da equipe do projeto.

Como **Especialista do negócio** uso meu conhecimento no negócio para apresentar à equipe do projeto as necessidades para que o software possua valor real para nós. Estou sempre disponível para fornecer aos desenvolvedores maior detalhamento sobre determinada funcionalidade. Sou um membro fixo da equipe e sempre estou fornecendo *feedback* das entregas para todos os envolvidos.



Por possuir bastante experiência técnica em modelagem orientada a objetos, e habilidade para atuar como facilitador na absorção das regras de negócio, fui escolhido para ser **Arquiteto** deste projeto. Serei, portanto, responsável pela última palavra em toda arquitetura do sistema.



Como **Gerente de Desenvolvimento**, sou responsável por liderar o dia-a-dia do desenvolvimento do produto. Por ser o responsável por resolver qualquer conflito técnico que exista entre programadores-chefes, preciso possuir boa experiência no desenvolvimento de software e nas tecnologias que estarão sendo utilizadas no projeto.

Sou um dos **Programadores-chefes** da nossa equipe e sou responsável por liderar pequenos grupos de desenvolvedores durante a construção das funcionalidades do produto. Também atuo como desenvolvedor e, normalmente, é atribuída a mim a propriedade das classes mais complexas do sistema. Possuo, ainda, papel fundamental nas fases de absorção do conhecimento de negócio e no planejamento das funcionalidades.



Eu sou **programadora (class-owner)** e sempre estou compondo pequenas equipes de funcionalidades nas quais programo, diagramo, testo e documento as funcionalidades a mim atribuídas pelo Programador-chefe da equipe.

**A FDD** fornece, ainda, para aqueles projetos maiores e/ou mais complexos, **papéis auxiliares**, tais como: Gerente de Release, **Testadores**, Escritores técnicos, **Guru da linguagem**, Administrador de Sistema, Implantadores e outros.

# O que é feature?

Features(funcionalidades) são expressões granulares que representem algum valor para o cliente.

Funcionalidades são nomeadas através do uso do template

**<ação><resultado><objeto>**

Alguns exemplos de funcionalidades:

Calcular o desconto de uma venda

Listar os clientes ativos de uma empresa

Destacar os clientes devedores de uma empresa

Imprimir a nota fiscal de uma venda

Validar a senha de um usuário

Enviar e-mail com os resultados mensais de uma empresa

Uma funcionalidade deve ter uma granularidade que não ultrapasse o tamanho de sua iteração. Por exemplo, se foi definido que o projeto terá iterações de 2(duas) semanas, você não deve possuir funcionalidades que ultrapassem esse tamanho(tempo). Quando isso acontecer, você deve procurar decompô-la em mais features.

É um conjunto de funcionalidades que você entregará para seu cliente ao final de uma iteração(ou release), portanto, não economize tempo, atenção e criatividade no processo de definição das mesmas.

# O ciclo de vida da FDD

O ciclo de vida da FDD é composto de 05(cinco) práticas. São elas:

## 1. Desenvolver um modelo abrangente

Este processo abrange todo o projeto, o que significa que ele **será executado uma única vez no projeto**.

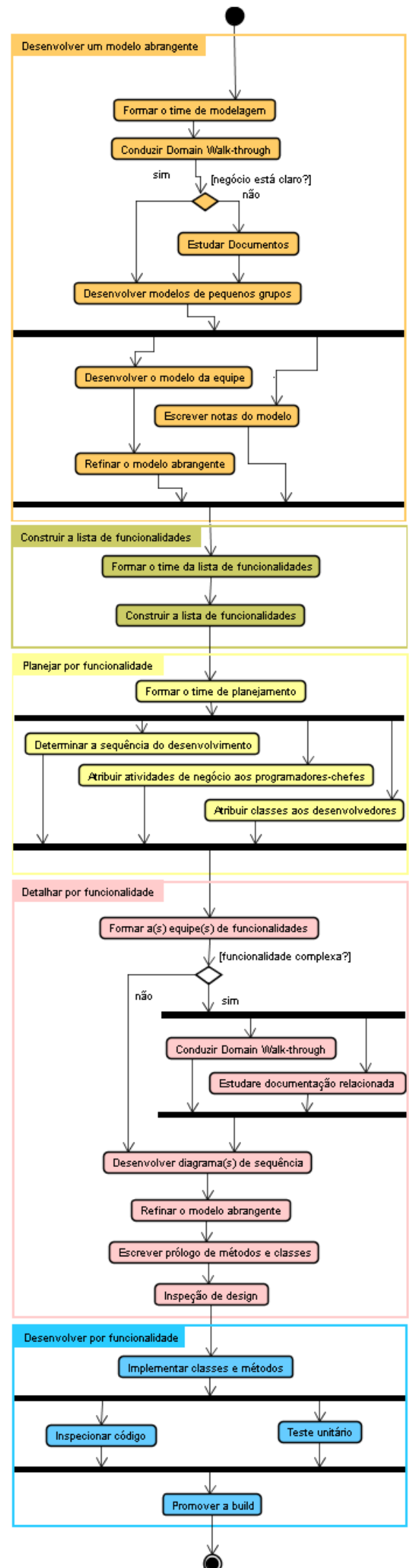
- Formar o time de modelagem: este time é normalmente composto por especialistas de negócio e programadores, sendo facilitados por um arquiteto com experiência em modelagem.
- Conduzir o Domain Walkthrough: os especialistas de negócio apresentarão ao restante da equipe uma visão do produto. Após isso, realizarão apresentações focadas em pequenas partes do negócio.
- Estudar documentação: Dependendo da complexidade da área de negócio apresentada, a equipe pode solicitar um intervalo para estudar a documentação fornecida pelo especialista de negócio.
- Desenvolver modelos de pequenos grupos: após cada apresentação(ou estudo), a equipe é dividida em pequenos grupos, que elaborarão uma proposta de modelo(sem detalhamento) para aquela parte específica do negócio que foi apresentada.
- Desenvolver o modelo da equipe: as propostas são apresentadas e uma delas, ou uma combinação delas, é escolhida por consenso para ser o modelo para aquela parte do negócio apresentada.
- Refinar o modelo abrangente: o modelo escolhido é incluso no modelo abrangente do produto. Este modelo abrangente é o resultado da junção de todos os modelos escolhidos para cada parte do negócio apresentada.
- Escrever notas: notas e observações são incluídas no modelo abrangente.

As atividades vão se repetindo até que tenhamos um modelo abrangente que cubra todas as partes de negócio previstas para o produto(ou release).

## 2. Construir a lista de funcionalidades

Este processo abrange todo o projeto, o que significa que ele **será executado uma única vez no projeto**.

- Formar o time da lista de funcionalidades: normalmente, este time é composto unicamente pelos programadores-chefes que participaram do processo anterior.
- Construir a lista de funcionalidades: as partes do produto, que foram identificadas e modeladas no processo anterior, são aqui identificadas como **áreas de negócio**. Dentro de cada área de



negócio, o time deve conseguir identificar as **atividades de negócio** daquela área específica e, dentro destas atividades, as **funcionalidades** que a compõem.

### 3. Planejar por funcionalidades

Este processo abrange todo o projeto, o que significa que ele **será executado uma única vez no projeto**.

- Formar o time de planejamento: normalmente este time é composto pelo gerente de projeto, gerente de desenvolvimento e programadores-chefes.
- Determinar a sequência do desenvolvimento: o time determina a sequência do desenvolvimento baseando-se nas dependências entre elas, na carga de trabalho da equipe de desenvolvimento e também na complexidade das funcionalidades a serem implementadas.
- Atribuir atividades de negócio aos programadores-chefes: cada programador-chefe fica responsável por um conjunto de atividades de negócio. Ele será o programador-chefe de todas as funcionalidades que compõem suas atividades.
- Atribuir classes aos desenvolvedores: cada classe passará a ter um “dono”. Este “dono”, que é um programador, será o responsável por qualquer manutenção necessária naquela classe. As classes são distribuídas pelo time levando em consideração a experiência, carga e sequência de trabalho de cada desenvolvedor.

### 4. Detalhar por funcionalidade

Este processo será executado **uma vez para cada funcionalidade**.

- Formar a(s) equipe(s) de funcionalidades: sabendo quais as classes que serão envolvidas no desenvolvimento de determinada funcionalidade, o programador-chefe convoca os desenvolvedores responsáveis por cada classe envolvida para fazer parte da equipe.
- Conduzir Domain Walkthrough: dependendo do nível de complexidade da funcionalidade e de quão clara ela está para a equipe, o programador-chefe pode convocar um especialista de negócio para promover uma apresentação detalhada daquela funcionalidade para a equipe.
- Estudar documentação relacionada: ainda dependendo do nível de entendimento do time, pode ser reservado um período para ser estudada documentação de negócio e anotações relacionadas àquela funcionalidade.
- Desenvolver diagrama(s) de sequência: o time desenvolve o(s) diagrama(s) de sequência relacionado(s) àquela funcionalidade.
- Refinar o modelo abrangente: já com um maior entendimento do negócio, o time se sente seguro em refinar o modelo abrangente, incluindo métodos e atributos nas classes envolvidas no desenvolvimento da funcionalidade.
- Escrever prólogo de métodos e classes: Com as informações geradas pelo(s) diagrama(s) de sequência, cada programador é responsável por criar os prólogos de suas classes. Isto inclui cabeçalhos de métodos com tipagem de parâmetros, atributos e outros. Vale lembrar que **apenas os prólogos** são criados aqui, nada de implementação deve ser realizado.
- Inspeção de design: o programador-chefe da funcionalidade deve convidar algum outro membro do time do projeto para avaliar o que foi feito em sua classe durante este processo.

### 5. Desenvolver por funcionalidade

Este processo será executado **uma vez para cada funcionalidade**.

- Implementar classes e métodos: cada desenvolvedor implementa suas classes e métodos de acordo com a visão abrangente e detalhamento realizados nos processos anteriores.
- Inspecionar código: cada desenvolvedor deve convidar algum outro membro do time (da funcionalidade ou do projeto) para avaliar o que foi feito em sua classe durante este processo.
- Teste unitário: cada desenvolvedor é responsável por executar os testes de unidade nos métodos de suas classes para garantir o alcance das necessidades do negócio.
- Promover a build: estando a classe inspecionada e testada, ela então pode ser promovida a build.



## A empresa, o projeto.










# Um projeto



A Equipe de Tecnologia da UNITY EVENTOS precisa iniciar um projeto para o desenvolvimento de um sistema que gerencie o processo de realização e inscrição em eventos realizados pela empresa.

A equipe optou por utilizar a FDD como metodologia, juntamente com o ferramental de desenvolvimento já utilizado pela empresa: Delphi, Together e SQL Server.

## O time.

<div>unitYeventos</div> <div></div> <div>Mônica Silva GERENTE DE PROJETO</div>	<div>unitYeventos</div> <div></div> <div>João Marcos ESPECIALISTA DE NEGÓCIO</div>	<div>unitYeventos</div> <div></div> <div>Thomáz Bin ARQUITETO</div>
<div>unitYeventos</div> <div></div> <div>Paulo Martins GERENTE DE DESENV. / PROG. CHEFE</div>	<div>unitYeventos</div> <div></div> <div>Carlos Júnior PROG. CHEFE</div>	<div>unitYeventos</div> <div></div> <div>Paula Pimenta PROGRAMADORA</div>
<div>unitYeventos</div> <div></div> <div>Raimunda Lins ESPECIALISTA DE NEGÓCIO</div>	<div>unitYeventos</div> <div></div> <div>José Pintado PROGRAMADOR</div>	<div>unitYeventos</div> <div></div> <div>Thiago Pires PROGRAMADOR</div>

# Processo 1: Desenvolver um modelo abrangente

- **Formar o time de modelagem**



**Mônica Silva**  
GERENTE DE PROJETO

Como nossa equipe é pequena, acredito que possa ser interessante que nosso time de modelagem seja composto por todos os membros do projeto.

- **Conduzir o Domain Walkthrough**



**Raimunda Lins**  
ESPECIALISTA DE  
NEGÓCIO

O processo de **organização de um evento** sobre o qual explanarei aqui é basicamente composto das seguintes etapas: primeiramente definimos o tema e período do evento, após isso realizamos uma tomada de preço em centro de convenções e hotéis que possuam auditório com a capacidade necessária para o evento. A lista de auditórios candidatos é realizada a partir de contatos anteriores e lista de páginas amarelas...

<< continua explicação >>

- **Estudar documentação**



**Thomáz Bin**  
ARQUITETO

Pessoal, como todos decidimos que a equipe necessita de um maior aprofundamento no assunto explanado pela Raimunda, realizaremos uma pausa de 45 minutos para estudarmos a documentação por ela entregue. Vocês podem realizar este estudo aqui mesmo ou em suas mesas, como preferirem. Portanto, às 15:45 retornaremos para iniciarmos o desenvolvimento do modelo.

- **Desenvolver modelos de pequenos grupos**



Agora, com um conhecimento mais abrangente sobre a organização de um evento, vamos nos dividir em grupos para a elaboração de propostas para um modelo do que foi exposto, seguindo o uso da UML em cores. Para isto é importante que cada equipe possua programador(es) e especialistas de negócio.

### **Equipe A**



**Paulo Martins**  
GERENTE DE DESENV.  
/ PROG. CHEFE



**João Marcos**  
ESPECIALISTA DE  
NEGÓCIO



**Paula Pimenta**  
PROGRAMADORA



**Thiago Pires**  
PROGRAMADOR

### **Equipe B**



**Carlos Júnior**  
PROG. CHEFE



**Raimunda Lins**  
ESPECIALISTA DE  
NEGÓCIO

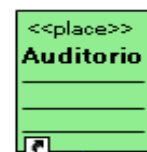


**José Pintado**  
PROGRAMADOR



**Paulo Martins**  
GERENTE DE DESENV.  
/ PROG. CHEFE

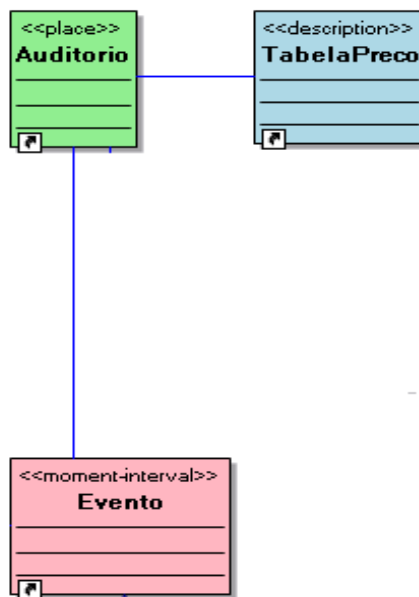
Esta é a nossa proposta de modelo para a área de organização de eventos...  
<<detalhamento >>





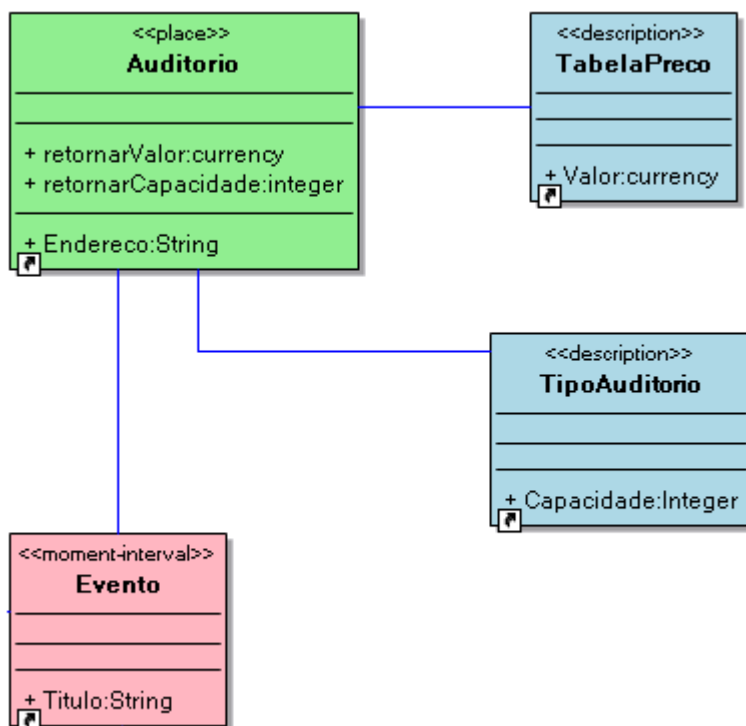
**José Pintado**  
PROGRAMADOR

Vejamos agora o nosso modelo...<<detalhamento >>



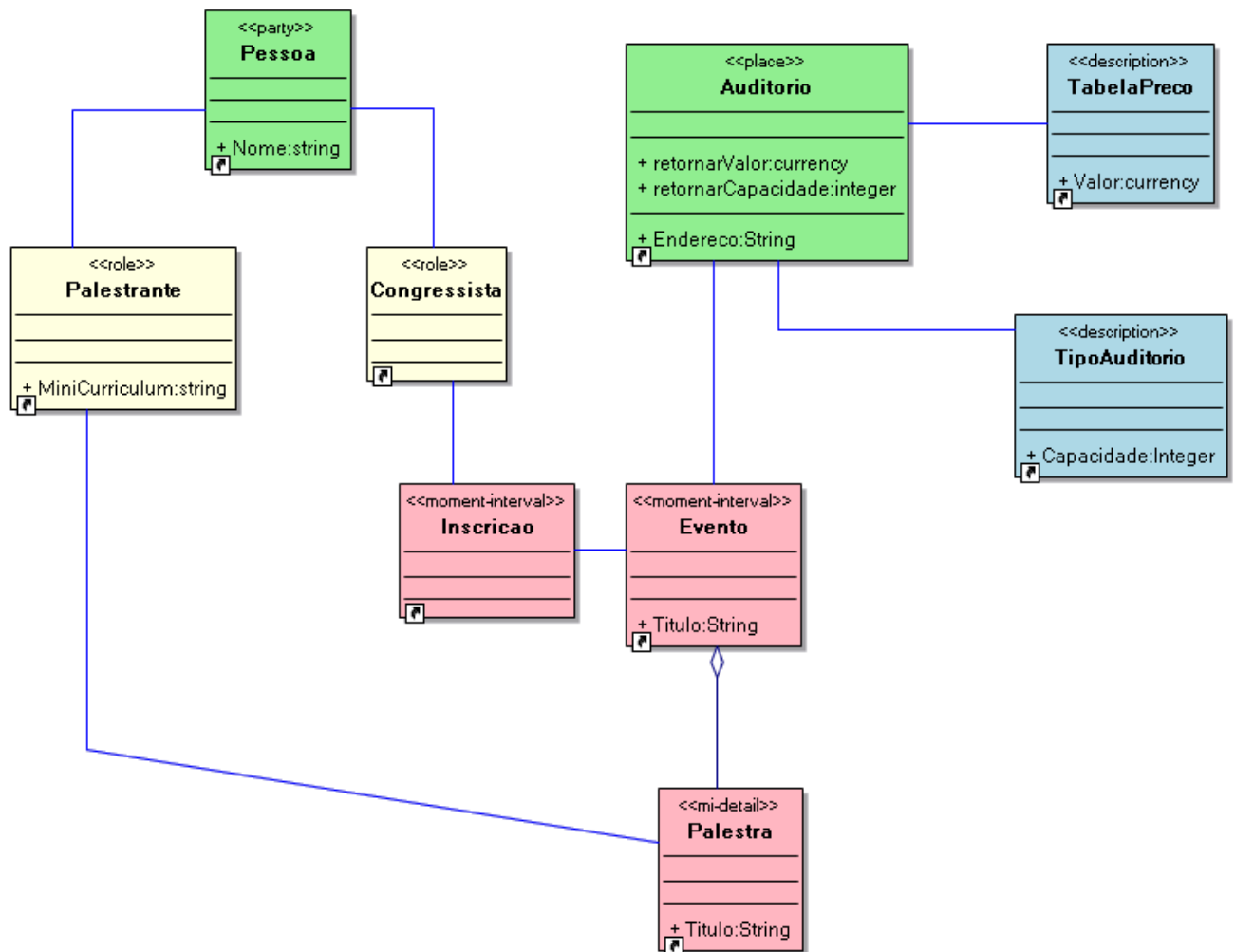
**Thomáz Bin**  
ARQUITETO

Bom pessoal, olhando o que cada um dos modelos tinha de bom, e com algumas sugestões minhas, temos aqui o nosso modelo para o processo de organização de eventos da Unity.



- **Refinar o modelo abrangente**

Após a explanação de outras áreas de negócio, tais como inscrição de congressistas no evento e recrutamento de palestrantes, o **modelo abrangente final** foi o seguinte:



## Processo 2: Construir a lista de funcionalidades

- **Formar o time da lista de funcionalidades**



**Mônica Silva**  
GERENTE DE PROJETO



**Paulo Martins**  
GERENTE DE DESENV.  
/ PROG. CHEFE

O time da Lista de Funcionalidades será composto pelos dois Programadores-Chefes presentes no primeiro processo.

- **Construir a lista de funcionalidades**



**Carlos Júnior**  
PROG. CHEFE

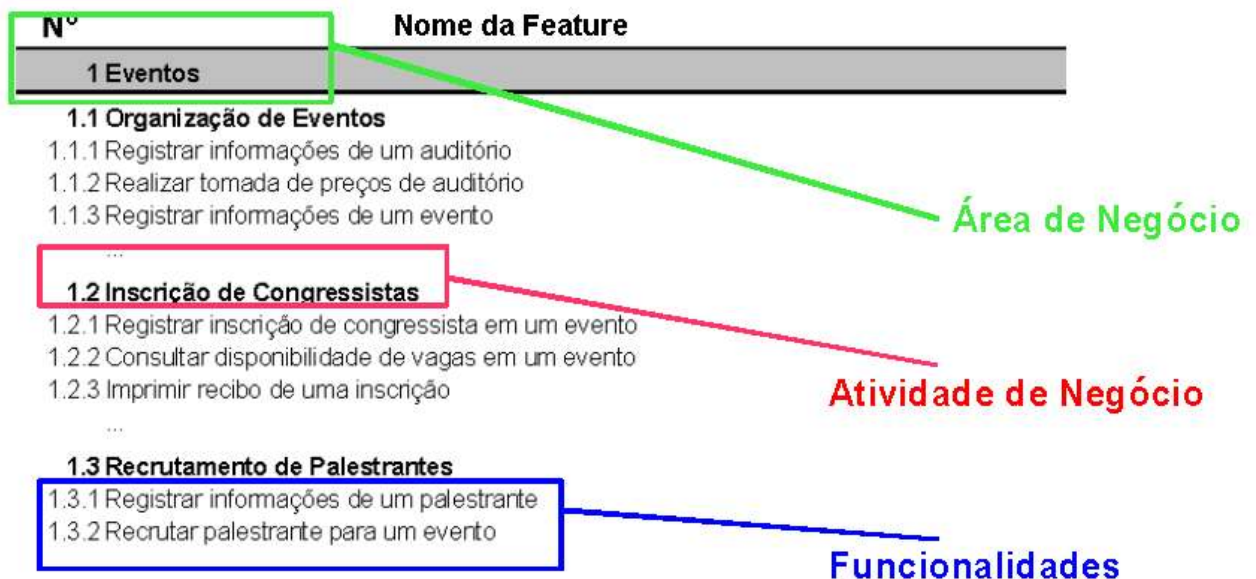
Paulo, nessa fase precisaremos identificar o conjunto de funcionalidades usando o conhecimento adquirido na Fase 1. Precisaremos identificar as **áreas de Negócio** apresentadas. Depois estas áreas serão decompostas em áreas de **atividades de negócio**, que, por sua vez, serão decompostas em **funcionalidade**. Estas funcionalidades são granulares e criadas a partir do *template*:

<ação> <resultado> <objeto>



**Paulo Martins**  
GERENTE DE DESENV.  
/ PROG. CHEFE

## UNITY



## Processo 3: Planejar por funcionalidades

- **Formar o time de planejamento**



**Paulo Martins**  
GERENTE DE DESENV.  
/ PROG. CHEFE

O time de Planejamento será formado pelo gerente de desenvolvimento, pelos programadores-chefes e pelo cliente (interno ou externo).

## • Determinar a sequência do desenvolvimento

Esta sequência é baseada em:

- Dependência entre as funcionalidades em termos de classes envolvidas;
- Distribuição de carga de trabalho entre os proprietários das classes;
- Complexidade das funcionalidades a serem implementadas;
- Adiantamento das atividades de negócio de alto risco ou complexidades;
- Prioridade do cliente;



**Raimunda Lins**  
ESPECIALISTA DE  
NEGÓCIO

Eu preciso que a parte de organização do evento esteja disponível o quanto antes, sendo que a montagem da grade de palestras pode ficar mais para frente. Outra atividade que também é de extrema importância é a de inscrição dos congressistas, pois assim que a organização do evento tiver sido finalizada, iniciaremos as inscrições.



**Carlos Júnior**  
PROG. CHEFE

Ok! Baseando-se então nesta sua priorização, analisaremos a dependência das classes, carga de trabalho e complexidade, para definirmos a sequência de desenvolvimento.

### 1 Eventos

#### 1.1 Organização de Eventos

- 1.1.1 Registrar informações de um auditório
- 1.1.2 Realizar tomada de preços de auditório para um evento
- 1.1.3 Registrar informações de um evento

...

#### 1.2 Inscrição de Congressistas

- 1.2.1 Registrar inscrição de congressista em um evento
- 1.2.2 Consultar disponibilidade de vagas em um evento
- 1.2.3 Imprimir recibo de uma inscrição

...

#### 1.3 Recrutamento de Palestrantes

- 1.3.1 Registrar informações de um palestrante
- 1.3.2 Recrutar palestrante para um evento



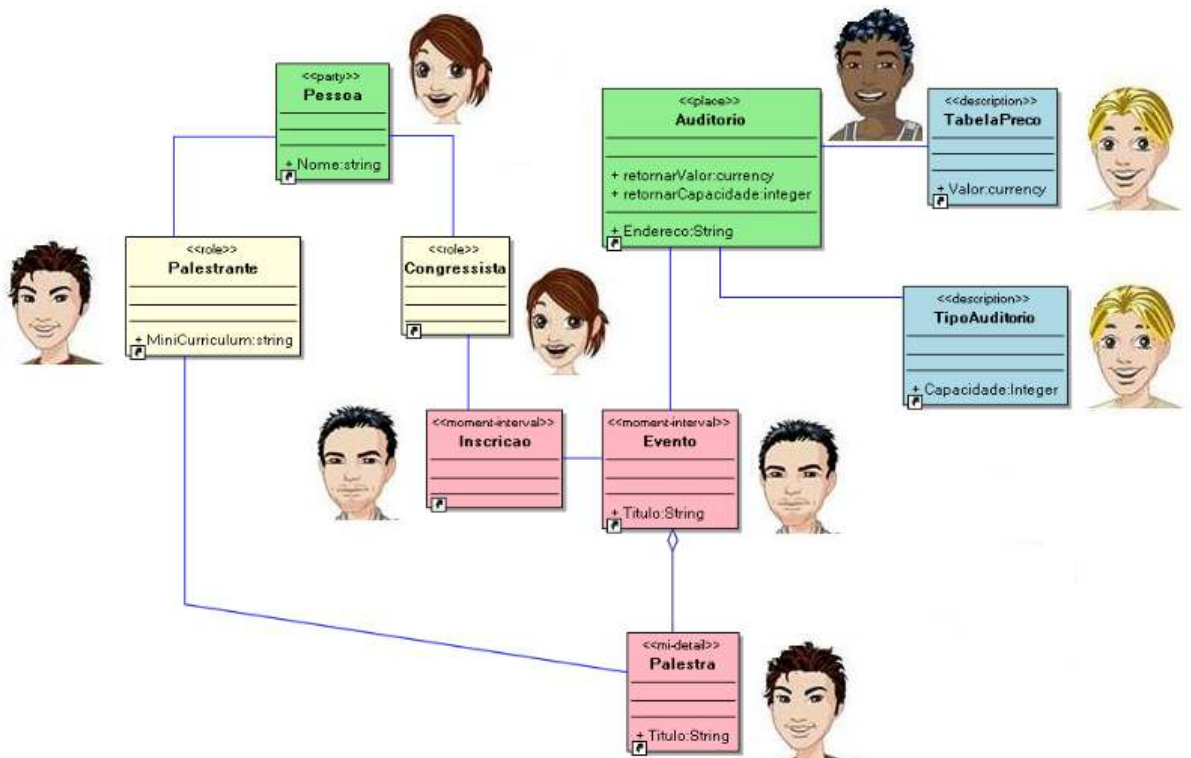
- Atribuir atividades de negócio aos programadores-chefes

Nº	Nome da Feature	Prog. Chefe	Planej.
1 Eventos			
1.1 Organização de Eventos			
1.1.1	Registrar informações de um auditório	ALL	20/Out
1.1.2	Realizar tomada de preços de auditório para um evento	ALL	20/Out
1.1.3	Registrar informações de um evento	ALL	03/Nov
...			
1.2 Inscrição de Congressistas			
1.2.1	Registrar inscrição de congressista em um evento	JMA	17/Nov
1.2.2	Consultar disponibilidade de vagas em um evento	JMA	03/Nov
1.2.3	Imprimir recibo de uma inscrição	JMA	17/Nov
...			
1.3 Recrutamento de Palestrantes			
1.3.1	Registrar informações de um palestrante	JMA	20/Out
1.3.2	Recrutar palestrante para um evento	JMA	17/Nov

Atribuição de Atividade de Negócios aos Programadores-Chefe

Planejamento / Estimativas

- Atribuir classes aos programadores





## Processo 4: Detalhar por funcionalidade

- Formar o time da funcionalidade



O time da funcionalidade **Registra informações de um auditório** será composto por:

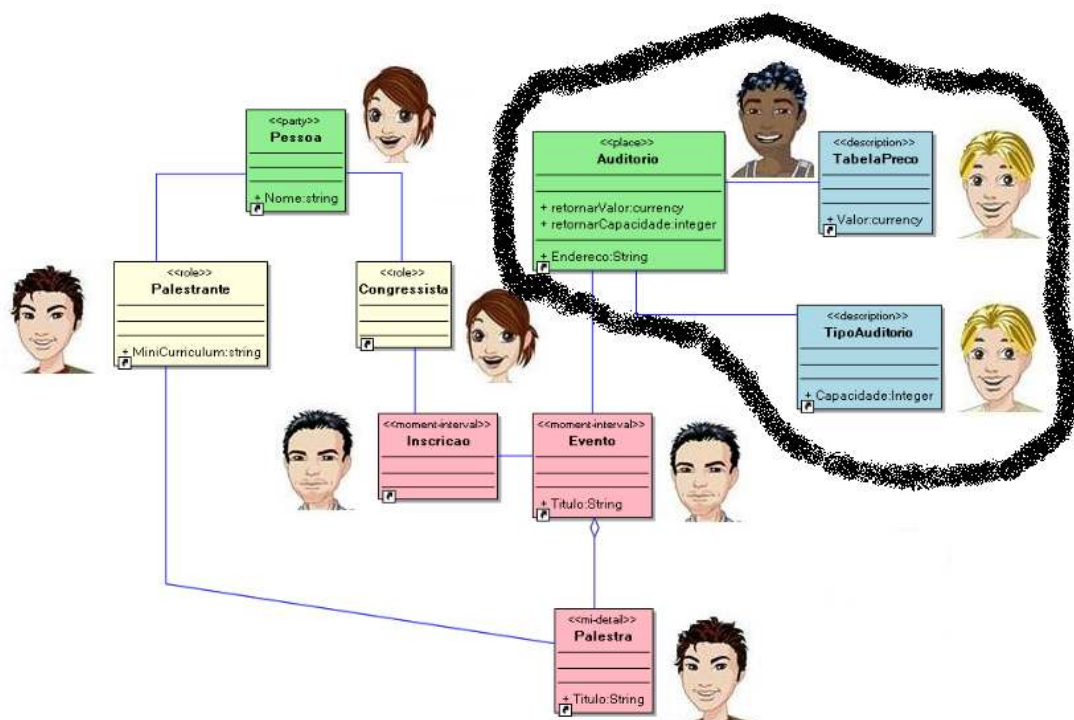


**Thiago Pires**  
PROGRAMADOR



**José Pintado**  
PROGRAMADOR

Como Carlos chegou a esses nomes para o time desta funcionalidade? Simples! Verificando as classes que seriam envolvidas na funcionalidade e verificando seus respectivos proprietários.



- Conduzir Domain Walkthrough

O Especialista do Negócio apresenta ao time da funcionalidade, uma visão mais detalhada da área de domínio a ser projetada.



**Raimunda Lins**  
ESPECIALISTA DE  
NEGÓCIO

<< detalhes sobre  
**registrar informações  
de um auditório** >>



**Thiago Pires**  
PROGRAMADOR

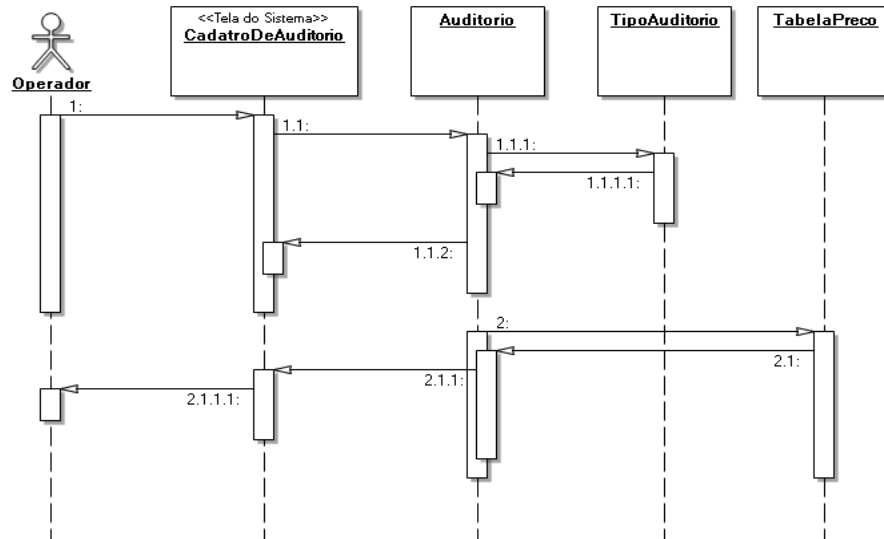


**José Pintado**  
PROGRAMADOR

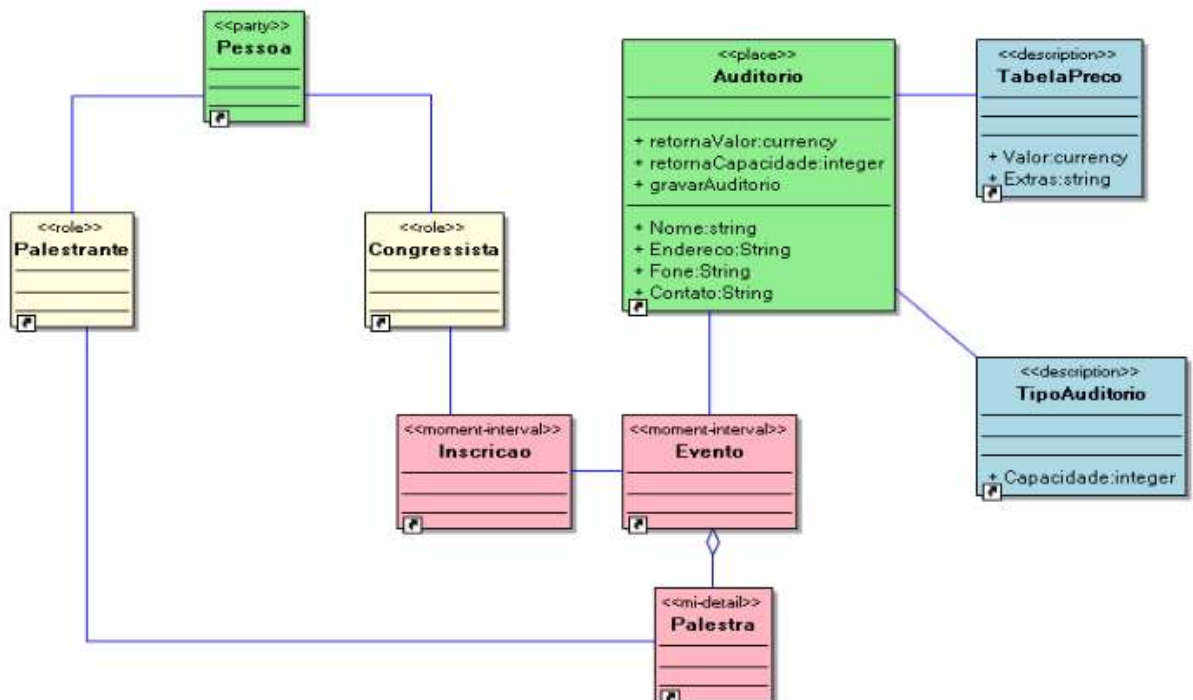
- **Estudar documentação relacionada**

A equipe da funcionalidade estuda o(s) documento(s) de referência. Desenhos de telas, memorandos, especificações de interface, e qualquer outra documentação considerada importante para um bom planejamento da funcionalidade.

- **Desenvolver diagrama(s) de sequência(s)**



- **Refinar o modelo abrangente**



- Escrever prólogo de métodos e classes

```

Welcome Page  untPrincipal  untAuditorios
+
+  unit untAuditorios;
+
+  interface
+
+  uses untTipoAuditorio, untTabelaPreco;
+
+  type
+
+  Auditorio = class
10  private
+    fNome : string;
+    fEndereco : string;
+    fFone : string;
+    fContato : string;
+
+  public
+    property Nome : string read fNome write fNome;
+    property Endereco : String read fEndereco write fEndereco;
+    property Fone : String read fFone write fFone;
+    property Contato : String read fContato write fContato;
20  function retornaValor: currency;
+    function retornaCapacidade: integer;
+    procedure gravarAuditorio;
+  end;
+
+  implementation
+
+  procedure Auditorio.gravarAuditorio;
+  begin
30  end;
+
+  function Auditorio.retornaCapacidade: integer;
+  begin

```

- Inspeção de design



**Carlos Júnior**  
PROG. CHEFE



Paula, você poderia por favor  
realizar a inspeção de design  
da feature **Registrar  
Informações de um  
Auditório?**



**Paula Pimenta**  
PROGRAMADORA

## Processo 5: Desenvolver por funcionalidade

O time da funcionalidade é mantido para agora, após detalhar e planejar o seu desenvolvimento, partir para a sua codificação.

- Implementar classes e métodos

Os proprietários de classes implementam os itens necessários para satisfazer os requisitos de suas classes para esta funcionalidade.



**Thiago Pires**  
PROGRAMADOR

```

Welcome Page  untPrincipal
290  FrmPrincipal.ColunaSelecionada := '';
+    FrmPrincipal.CntrCbXMetadado.PopupMenu := nil;
+  end;
+
+  procedure TfrmPrincipal.FormCreate(Sender: TObject);
+  var
+    x : Integer;
+  begin
+    vEdited := false;
+    PgCtrl1Metadados.ActivePage := TbShtLogo;
+    NomeMetadado := '';
300  Administrador := false;
+    ColunaSelecionada := '';
+
+    if not Conexao.LerArquivoIniConexao then
+      ParamForm := 'A' ( Parâmetro A = Abrir form )
+    else
+      ParamForm := 'F';
+  end;
+
+  procedure TfrmPrincipal.FormKeyPress(Sender: TObject; var Ke:
310  begin
+    vEnter := (Key = #13);
+  end;
+
+  procedure TfrmPrincipal.FormShow(Sender: TObject);
+  var
+    x : Integer;
+  begin
+    FillChar(ParamStr, SizeOf(ParamStr), 0);
+    try
320  frmLogin := TfrmLogin.Create(Application);
+    frmLogin.ShowModal;
+    finally
+      FreeAndNil(frmLogin);

```

- **Inspecionar código**

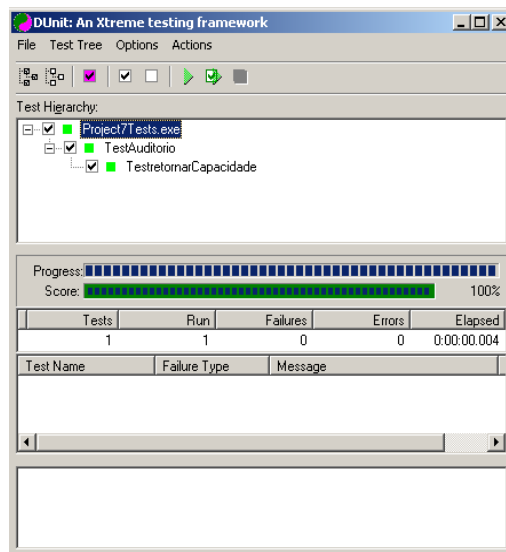
Seguindo o mesmo mecanismo utilizado na inspeção de design, o programador-chefe solicita que um outro programador, ou mesmo um outro programador-chefe, inspecione todo o código implementado para esta funcionalidade.

- **Teste de unidade**

Utilizando recurso da própria ferramenta de desenvolvimento adotada, o time realiza teste de unidade em todos os métodos implementados para esta funcionalidade. A política de teste de unidade, ou seja, por quem será realizado e em que exato momento, deve ser definida pelo programador-chefe da funcionalidade.



**Carlos Júnior**  
PROG. CHEFE



- **Promover a build**



**Carlos Júnior**  
PROG. CHEFE

Este é o momento de integrar todas as classes e métodos desenvolvidos para esta funcionalidade.



**José Pintado**  
PROGRAMADOR

**DONE!**



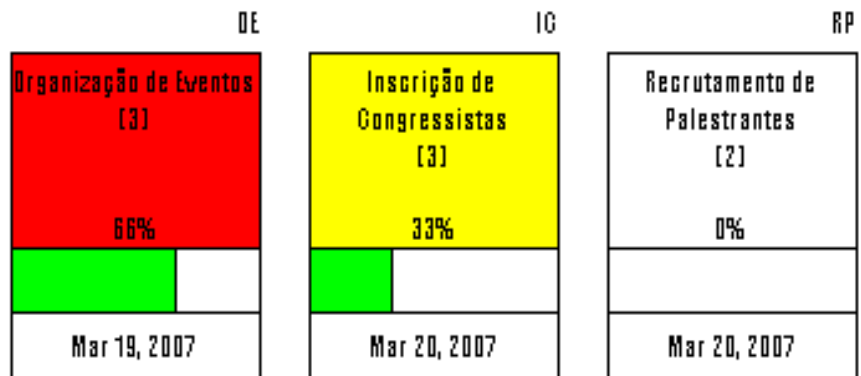
**Thiago Pires**  
PROGRAMADOR

# Visibilidade gerencial



**Mônica Silva**  
GERENTE DE PROJETO

Durante todos os releases e iterações do projeto, eu, como gerente de projeto, tenho que possuir mecanismos para visualizar o andamento das atividades, ou melhor, das funcionalidades. Consigo isto através do Parking Lot, um dos principais gráficos gerenciais da FDD.



## Seqüência do projeto

Após a promoção a *build* de uma funcionalidade(ou pacote de funcionalidades), os processos 4 e 5 vão sendo repetidos para cada nova funcionalidade(ou pacote) prevista para a iteração corrente. Ao final de cada iteração, a lista de funcionalidades(*backlog*) deve ser revisada, quando então poderão ser adicionadas ou removidas novas funcionalidades na lista.

# Dúvidas freqüentes

## **Existem ferramentas disponíveis para FDD?**

Sim, existem várias ferramentas específicas para o uso dos processos da FDD, dentre elas podemos citar: FDDTracker([www.fddtracker.com](http://www.fddtracker.com)) e FDDTools([fddtools.sourceforge.net](http://fddtools.sourceforge.net)) como as principais, sendo que a primeira é ideal para quem precisa de uma ferramenta mais abrangente, e a segunda para quem quer apenas controlar as funcionalidades do projeto e visualizá-las através do Parking Lot. Ferramentas de grandes fornecedores, como o Caliber da Borland e o RequisitePro da IBM/Rational, também suportam as funcionalidades da FDD, mas sem seus recursos específicos.

Vale ainda frisar que, com simples planilhas eletrônicas ou mesmo *post-its*, você consegue sem muitas complicações montar um ambiente para completo acompanhamento de projetos FDD.

## **Posso utilizar FDD em conjunto com outra metodologia?**

O uso da FDD em conjunto com alguma metodologia de gerenciamento de projeto é muito comum. Como exemplo podemos citar a “dobrinha” Scrum/FDD que, sendo bem planejada, pode ser de grande valia para seus projetos de software.

## **Na FDD, como é tratado o processo de levantamento de requisitos?**

R: A FDD pressupõe que, ao entrar na fase de desenvolvimento do modelo abrangente, você já tem em mãos os requisitos do seu projeto, sejam eles use-cases, storys ou qualquer outra coisa. Portanto, a FDD considera que o processo de levantamento de requisitos deva ser executado antes de você entrar nas práticas de engenharia, ou seja, antes de entrar na FDD. No post “FDD e requisitos” ([www.axmagno.com](http://www.axmagno.com)) comento mais sobre este assunto.

## **Posso utilizar FDD em projetos relacionais, ou seja, sem nenhum uso de orientação a objetos?**

Apesar da documentação da FDD ser sempre “recheada” das palavras classes, objetos e diagramas, nada nos impede de utilizá-la em projetos relacionais/procedurais. Uma vez que você considere o modelo abrangente da Fase 1 como um MER ao invés de um diagrama de classes, todas as práticas seguintes serão aplicadas a ele. David Anderson fala sobre um projeto relacional onde aplicou FDD no post “FDD in non-OO projects” em [www.agilemanagement.net](http://www.agilemanagement.net). Em breve devo estar escrevendo algo sobre a minha experiência neste tipo de projeto aqui mesmo em [www.axmagno.com](http://www.axmagno.com).

## **Como está sendo o uso da FDD pelo mundo?**

A Trail Ridge Consulting realizou uma pesquisa sobre o uso de práticas ágeis em projetos de software pelo mundo. Nela podemos perceber que a FDD, juntamente com a Extreme Programming são as duas metodologias ágeis para a engenharia de software de maior destaque. A Heptagon disponibilizou em seu site uma versão em português desta pesquisa em seu site [www.heptagon.com.br](http://www.heptagon.com.br), vale a pena dar uma conferida nos resultados.

## **Como faço estimativas em FDD?**

A FDD não define uma prática “oficial” para estimativas. No livro Agile

Management for Software Engineer, David Anderson sugere uma tabela de medidas não-lineares, baseadas em um certo grau de dificuldade para serem utilizadas em projetos FDD que façam uso dos arquétipos de Peter Coad, mas é apenas uma sugestão.

Com o uso do Scrum em conjunto com a FDD, a prática do Planning Poker para definir o tamanho das funcionalidades pode ser uma boa opção.

### **Ao utilizar FDD sou obrigado a utilizar UML em cores?**

Não. O modelo abrangente da FDD pode ser tranquilamente desenvolvido fazendo o uso da UML padrão, ou – como já citei anteriormente – do modelo entidade/relacionamento. No entanto, não podemos deixar de frisar que os arquétipos da UML em cores funcionam muito bem para projetos FDD e, na minha opinião, sempre que possível devem ser utilizados.

### **Onde consigo mais informações sobre FDD?**

aXmagno: <http://amagno.blogspot.com>

Heptagon: [www.heptagon.com.br](http://www.heptagon.com.br)

Nebulon – Jeff De Luca: [www.nebulon.com](http://www.nebulon.com)

FDD Oficial Site: [www.featuredrivendevelopment.com](http://www.featuredrivendevelopment.com)

David Anderson: [www.agilemanagement.net](http://www.agilemanagement.net)

Stephen Palmer: [www.step-10.com](http://www.step-10.com)

Grupo de Usuários da FDD: <http://br.groups.yahoo.com/group/gufdd>

## **Sobre o autor**



**Alexandre Magno Figueiredo** vive em São Paulo -SP, onde trabalha como consultor em liderança e gerenciamento de projetos de software através do uso de metodologias e processos ágeis, principalmente FDD e Scrum. Atua na área de software há mais de 15 anos, já tendo participado de projetos de variadas dimensões de *lead time*, escopo e investimento. É Certified ScrumMaster Practitioner, possuindo ainda certificações dos fornecedores IBM e Borland, e dos grupos OMG e PMI. Pode ser encontrado em [axmagno@gmail.com](mailto:axmagno@gmail.com), no Palestra Itália assistindo aos jogos do Palmeiras ou em diversos outros lugares passeando com sua família.