

ProjectNCode

Copyright 2009

Ricardo JL Rufino

Índice

Programação dos Casos de Uso3
DCU002 - Opções de Geração4
DCU003 - Configurações5
DCL (Core) - Configuração6
DCL (Core) - Exception8
DCL (Core) - MetaData (Hierarquia)9
DCL (Core) - MetaDataFactory11
DCL (Core) - Template Engine12
DCL (Geral) - Analyzer13
DCL (Geral) - Builder16
DCL (Plugin) - FormMetaData (Rules)18
DCL (Plugin) - MetaData (Implementação)19
DCL (Plugin) - MetaData (Relacionamentos)21
DCL - Constants23
DCL - Java API24
DS - Obtendo um ProgramaticLanguage26
DSE001 - DCU002.1 Gerar Formulário27
DA - Logica dos Analyzers28
DA - Processo de Geração30

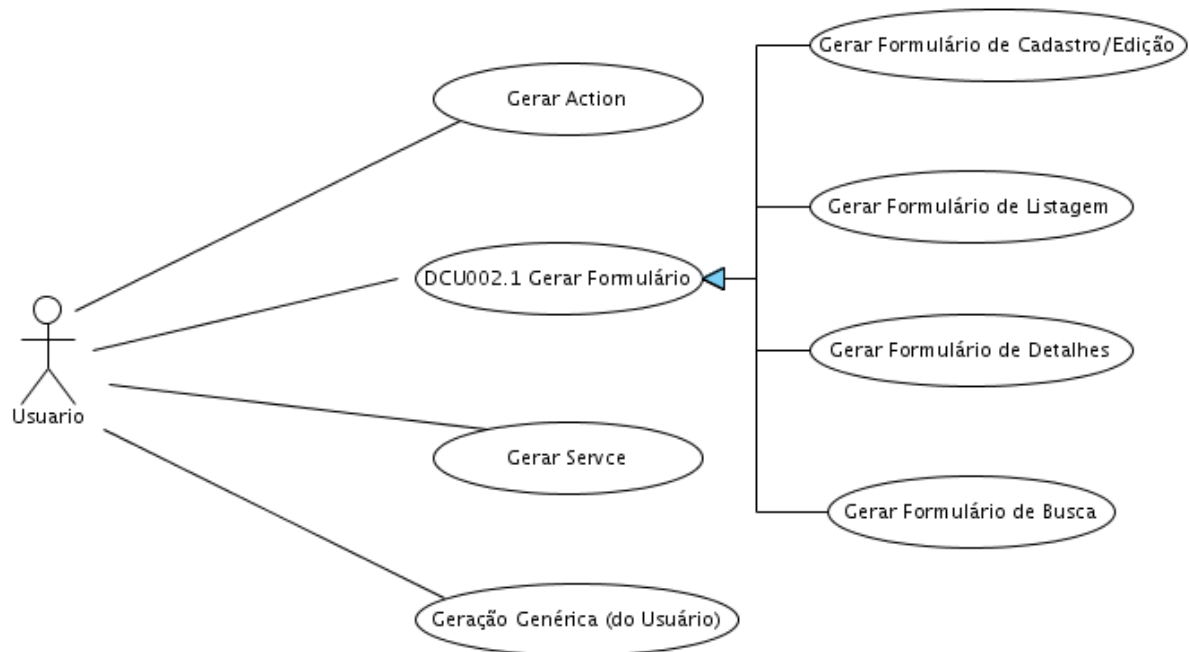
Tabela de Figuras

DCU002 - Opções de Geração	4
DCU003 - Configurações	5
DCL (Core) - Configuração	6
DCL (Core) - Exception	8
DCL (Core) - MetaData (Hierarquia)	9
DCL (Core) - MetaDataFactory	11
DCL (Core) - Template Engine	12
DCL (Geral) - Analyzer	13
DCL (Geral) - Builder	16
DCL (Plugin) - FormMetaData (Rules)	18
DCL (Plugin) - MetaData (Implementação)	19
DCL (Plugin) - MetaData (Relacionamentos)	21
DCL - Constants	23
DCL - Java API	24
DS - Obtendo um ProgramaticLanguage	26
DSE001 - DCU002.1 Gerar Formulário	27
DA - Logica dos Analizers	28
DA - Processo de Geração	30

Programação dos Casos de Uso

Prioridade	Nome do Caso de Uso	Justificativa
High	DCU003.1 - Propriedades do Projeto	Mostar o processo de Carregamento dos dados
High	DCU003.2 - Preferencias Globais	
Unspecified	DCU002.1 Gerar Formulário	
Unspecified	Gerar Action	
Unspecified	Gerar Formulário de Busca	
Unspecified	Gerar Formulário de Cadastro/Edição	
Unspecified	Gerar Formulário de Detalhes	
Unspecified	Gerar Formulário de Listagem	
Unspecified	Gerar Service	
Unspecified	Geração Genérica (do Usuário)	
Unspecified	Cancelar	
Unspecified	Editar Template	
Unspecified	Opções Gerais	
Unspecified	Salvar Configurações	
Unspecified	Salvar Template	
Unspecified	Templates	
Unspecified	Visualizar Template	

Diagrama de Caso de Uso

DCU002 - Opções de Geração**Sumário**










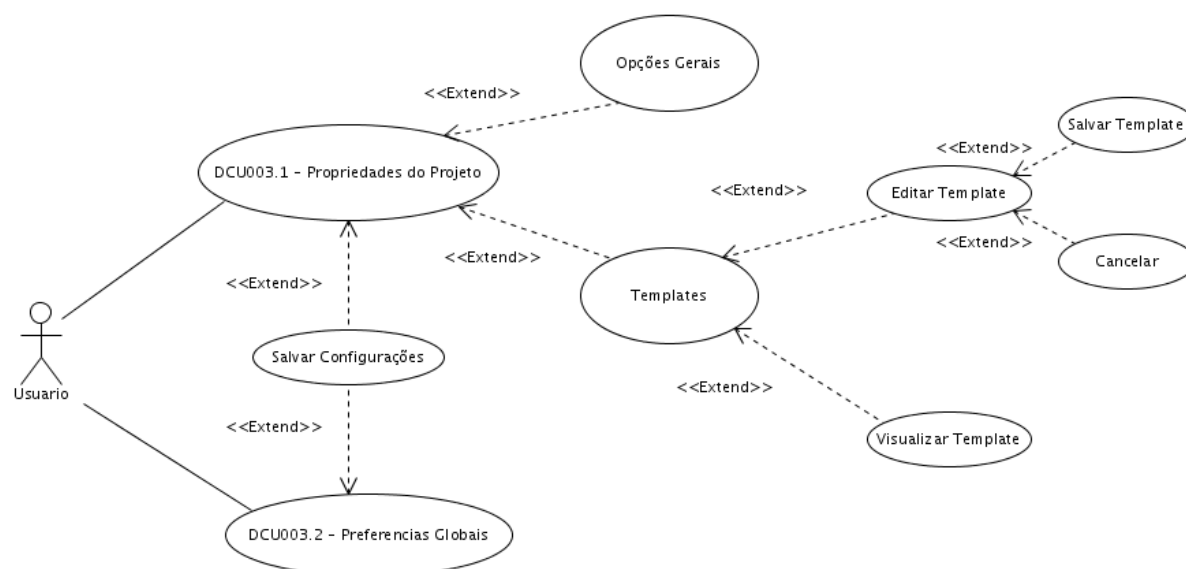
Nome	Documentação
 DCU002.1 Gerar Formulário	
 Gerar Action	
 Gerar Formulário de Busca	
 Gerar Formulário de Cadastro/Edição	
 Gerar Formulário de Detalhes	
 Gerar Formulário de Listagem	
 Gerar Service	
 Geração Genérica (do Usuário)	
 Usuario	

Diagrama de Caso de Uso

DCU003 - Configurações



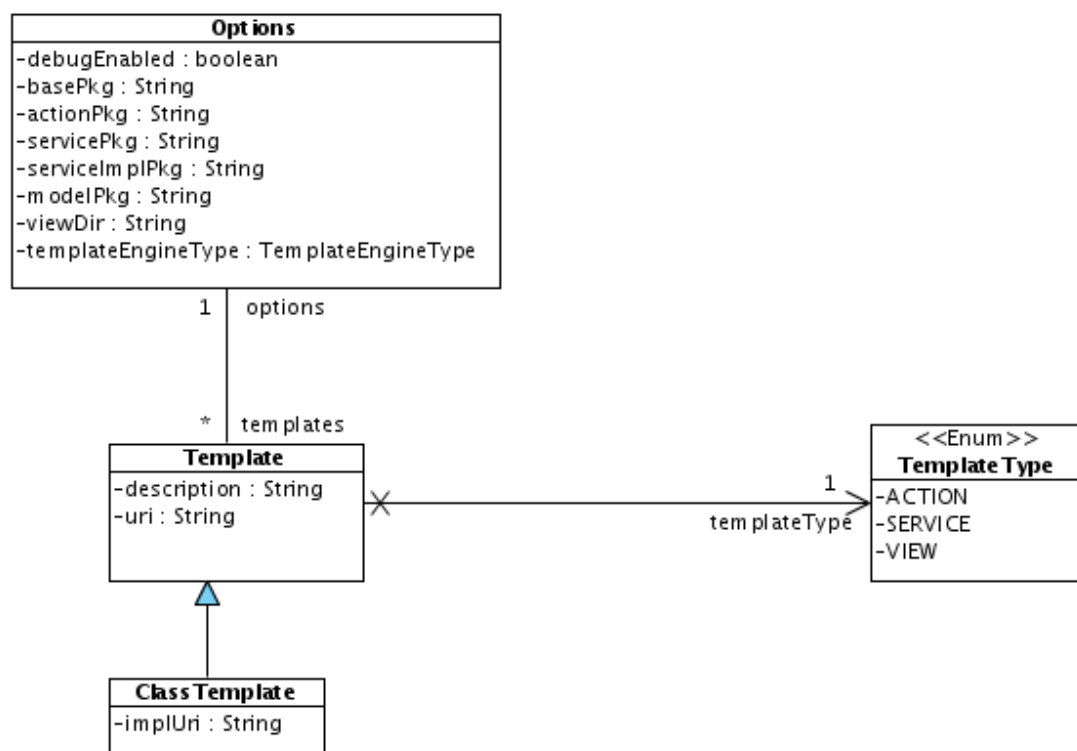
Sumário

Nome	Documentação
● Cancelar	
● DCU003.1 - Propriedades do Projeto	
● DCU003.2 - Preferencias Globais	
● Editar Template	
● Opções Gerais	
● Salvar Configurações	
● Salvar Template	
● Templates	
● Usuario	
● Visualizar Template	

Diagrama de Classe

DCL (Core) - Configuração

Armazena informações sobre as Preferencias de geração, quais os templates a serem usados, qual o nome dos pacotes que ele utiliza, etc...



Sumário






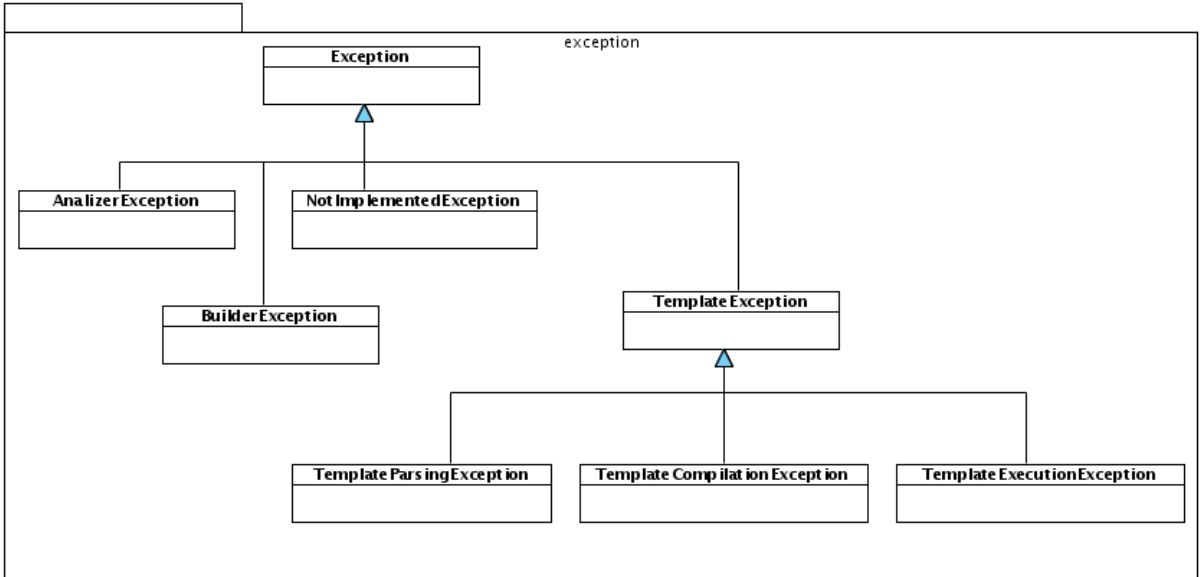
Nome	Documentação
	Armazena informações sobre as Preferencias de geração, quais os templates a serem usados, qual o nome dos pacotes que ele utiliza, etc...
 ClassTemplate	Template destinado a Classes Java, que podem ou não ter uma implementação @author Ricardo JL Rufino (ricardo.jl.rufino@gmail.com) @date 01:15:46 20/08/2009 @update
 Options	TODO: Documentar Classe @author Ricardo JL Rufino (ricardo.jl.rufino@gmail.com) @date 10:57:10 19/08/2009 @update
 Template	Classe de Modelo que Indentifica um Template @author Ricardo JL Rufino (ricardo.jl.rufino@gmail.com) @date 11:05:48 19/08/2009 @update
 TemplateType	Identifica o Tipo de Template @author Ricardo JL Rufino (ricardo.jl.rufino@gmail.com) @date 11:11:50 19/08/2009 @update

Diagrama de Classe

DCL (Core) - Exception



Sumário










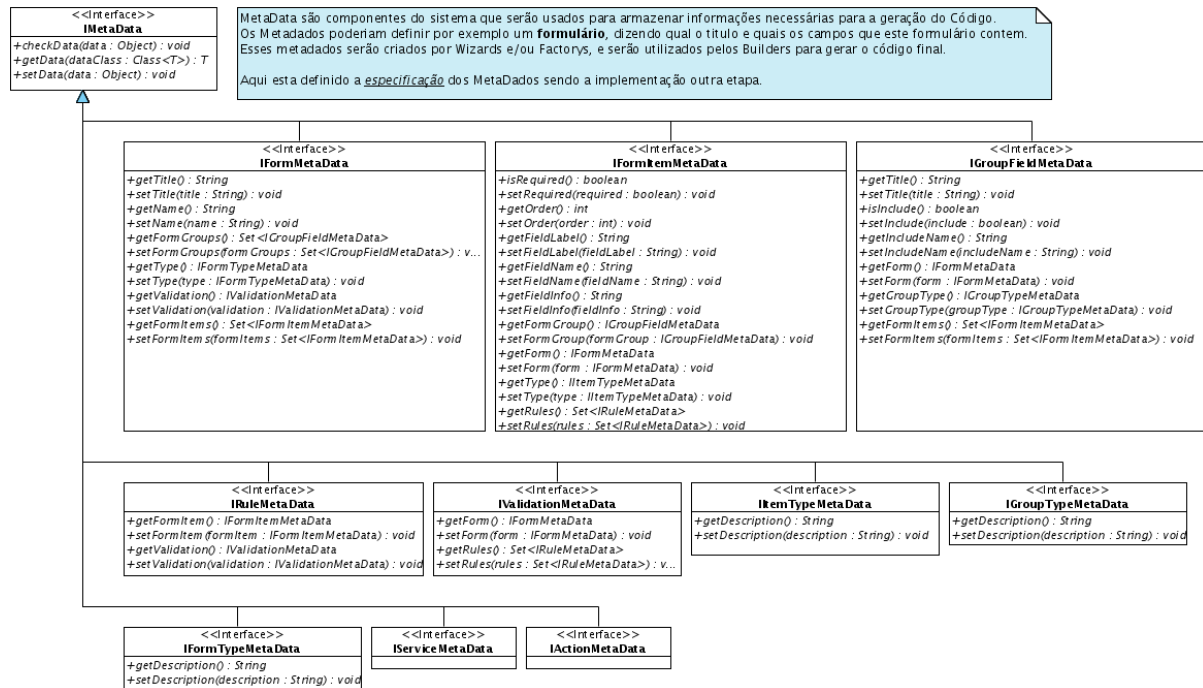
Nome	Documentação
 exception	
 AnalyzerException	
 BuilderException	
 Exception	
 NotImplementedException	
 TemplateCompilationException	
 TemplateException	
 TemplateExecutionException	
 TemplateParsingException	

Diagrama de Classe

DCL (Core) - Metadata (Hierarquia)



Sumário













Nome	Documentação
	<p>MetaData são componentes do sistema que serão usados para armazenar informações necessárias para a geração do Código. Os Metadados poderiam definir por exemplo um formulário, dizendo qual o título e quais os campos que este formulário contem. Esses metadados serão criados por Wizards e/ou Factorys, e serão utilizados pelos Builders para gerar o código final.</p> <p>Aqui esta definido a <u>especificação</u> dos MetaDados sendo a implementação outra etapa.</p>
 IActionMetaData	
 IFormItemMetaData	
 IFormMetaData	
 IFormTypeMetaData	
 IGroupFieldMetaData	
 IGroupTypeMetaData	
 IItemTypeMetaData	
 IMetaData	
 IRuleMetaData	
 IServiceMetaData	
 IValidationMetaData	

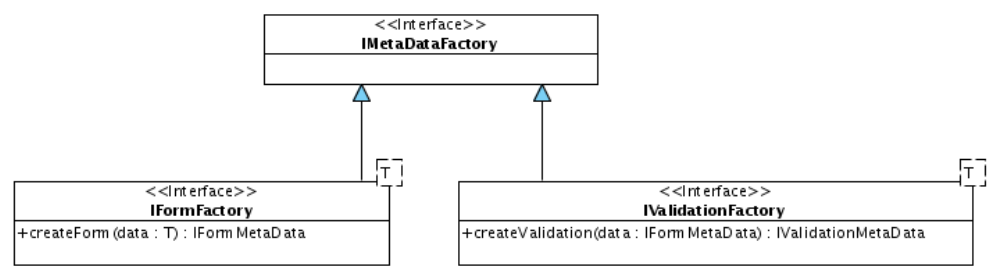
Diagrama de Classe

DCL (Core) - MetaDataFactory

As Factorys de MetaData serão componentes usados para criar metadados iniciais ou completos apartir de uma determinada fonte, que pode ser por exemplo um classe java.

Elas podem usar uma espécie de **inteligência artificial** no momento da criação desse modelo.

Por exemplo: Se você estiver gerando um Cadastro de Funcionário, e anteriormente já fez um cadastro de Clientes ele pode pegar grande parte da configuração desse formulário, seja do Banco de Dados ou Arquivos XML



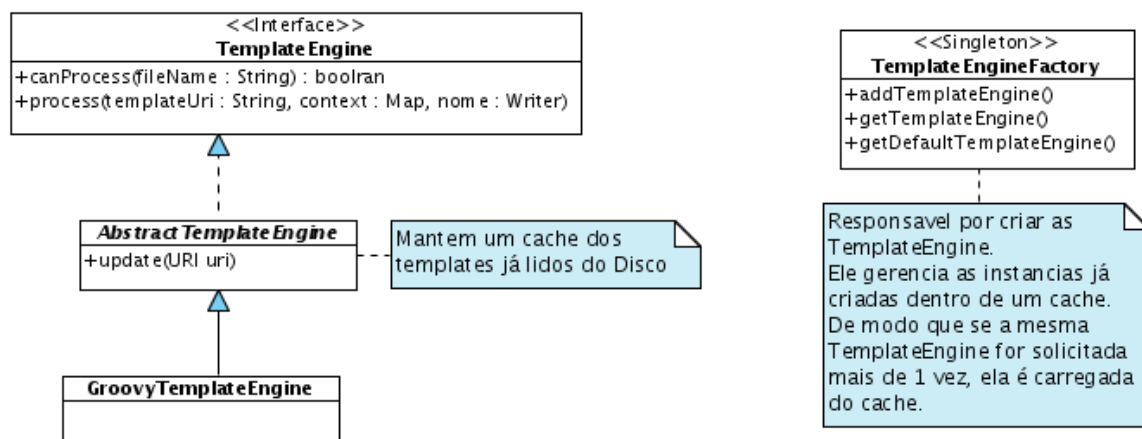
Sumário

Nome	Documentação
	<p>As Factorys de MetaData serão componentes usados para criar metadados iniciais ou completos apartir de uma determinada fonte, que pode ser por exemplo um classe java.</p> <p>Elas podem usar uma espécie de inteligência artificial no momento da criação desse modelo.</p> <p>Por exemplo: Se você estiver gerando um Cadastro de Funcionário, e anteriormente já fez um cadastro de Clientes ele pode pegar grande parte da configuração desse formulário, seja do Banco de Dados ou Arquivos XML</p>
IFormFactory	
IMetaDataFactory	
IValidationFactory	

Diagrama de Classe

DCL (Core) - Template Engine

Componentes da aplicação responsáveis por interpretar os Templates de Código.
Existe várias engines de template que podem ser usadas como: Velocity, JSP, FreeMaker, Groovy, etc...

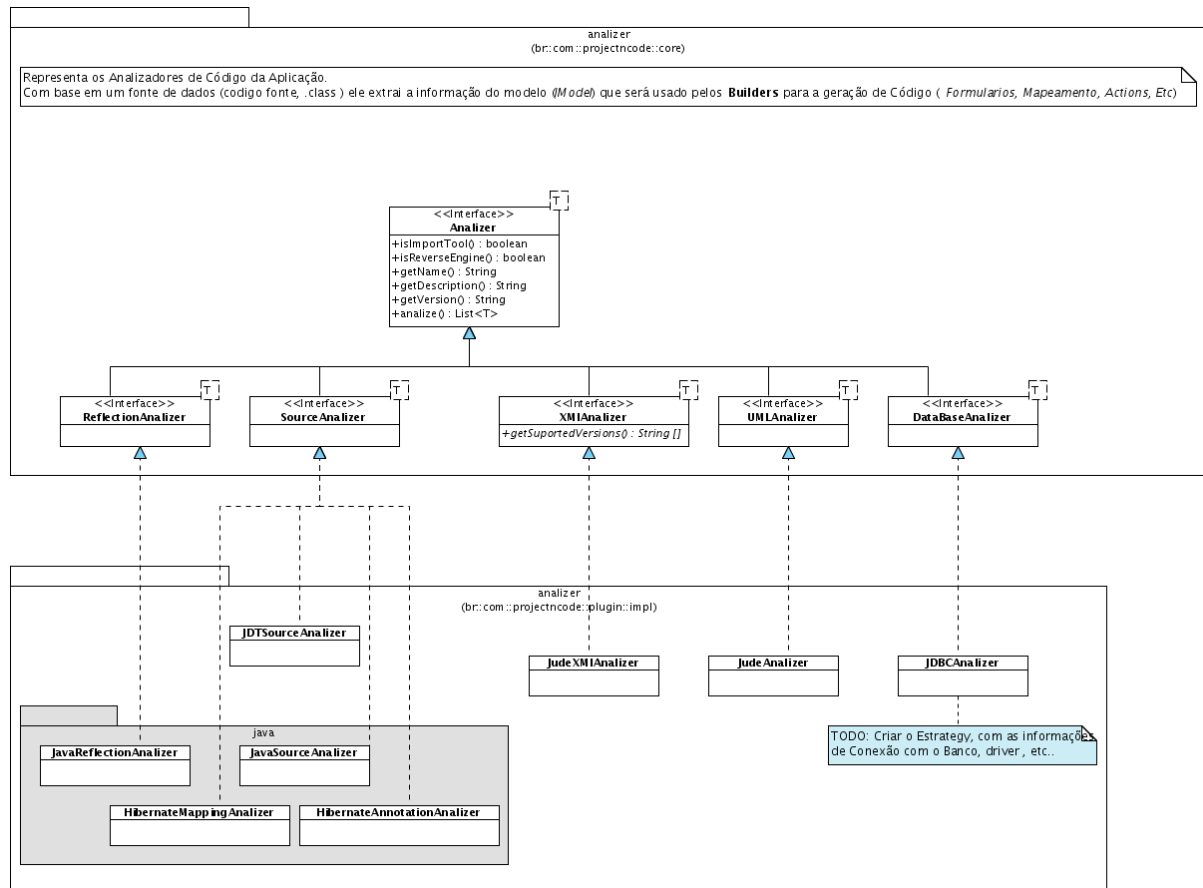


Sumário
















Nome	Documentação
	Responsavel por criar as TemplateEngine. Ele gerencia as instancias já criadas dentro de um cache. De modo que se a mesma TemplateEngine for solicitada mais de 1 vez, ela é carregada do cache.
	Mantem um cache dos templates já lidos do Disco
	Componentes da aplicação responsáveis por interpretar os Templates de Código. Existe várias engines de template que podem ser usadas como: Velocity, JSP, FreeMaker, Groovy, etc...
AbstractTemplateEngine	
GroovyTemplateEngine	
TemplateEngine	
TemplateEngineFactory	

Diagrama de Classe

DCL (Geral) - Analyzer



Sumário

Nome	Documentação
 analyzer	
	Representa os Analizadores de Código da Aplicação. Com base em um fonte de dados (codigo fonte, .class) ele extrai a informação do modelo (<i>IModel</i>) que será usado pelos Builders para a geração de Código (<i>Formularios, Mapeamento, Actions, Etc</i>)
 Analyzer	Representa a interface que define um Analizador de Dados, que pode ser considerado tambem como um Sistema de Importação ou Engenharia Reversa. TODO: Explicar como os analizers devem se registrar para serem visualizados pelos Windards, e explicar as possibilidades de desenvolvimento de Plugins para analizers
 DataBaseAnalyzer	
 ReflectionAnalyzer	Analizer que extrai informações de classes já compiladas (ou em tempo de execução), para a isso a linguagem deve prover mecanismos de Reflexão. Um Exemplo é a linguagem Java que já tem uma API para fazer a reflexão de classes.
 SourceAnalyzer	Analizer que extrai informações do Código Fonte da Linguagem. Algumas linguagens possuem APIs para isso, as mais comuns são as de gerar a documentação, um exemplo é o JavaDoc da SUN.
 UMLAnalyzer	
 XMIAAnalyzer	
 analyzer	
	TODO: Criar o Estrategy, com as informações de Conexão com o Banco, driver , etc..
 JDBCAnalyzer	
 JDTSrcAnalyzer	
 JudeAnalyzer	
 JudeXMIAAnalyzer	
 java	

ProjectNCode





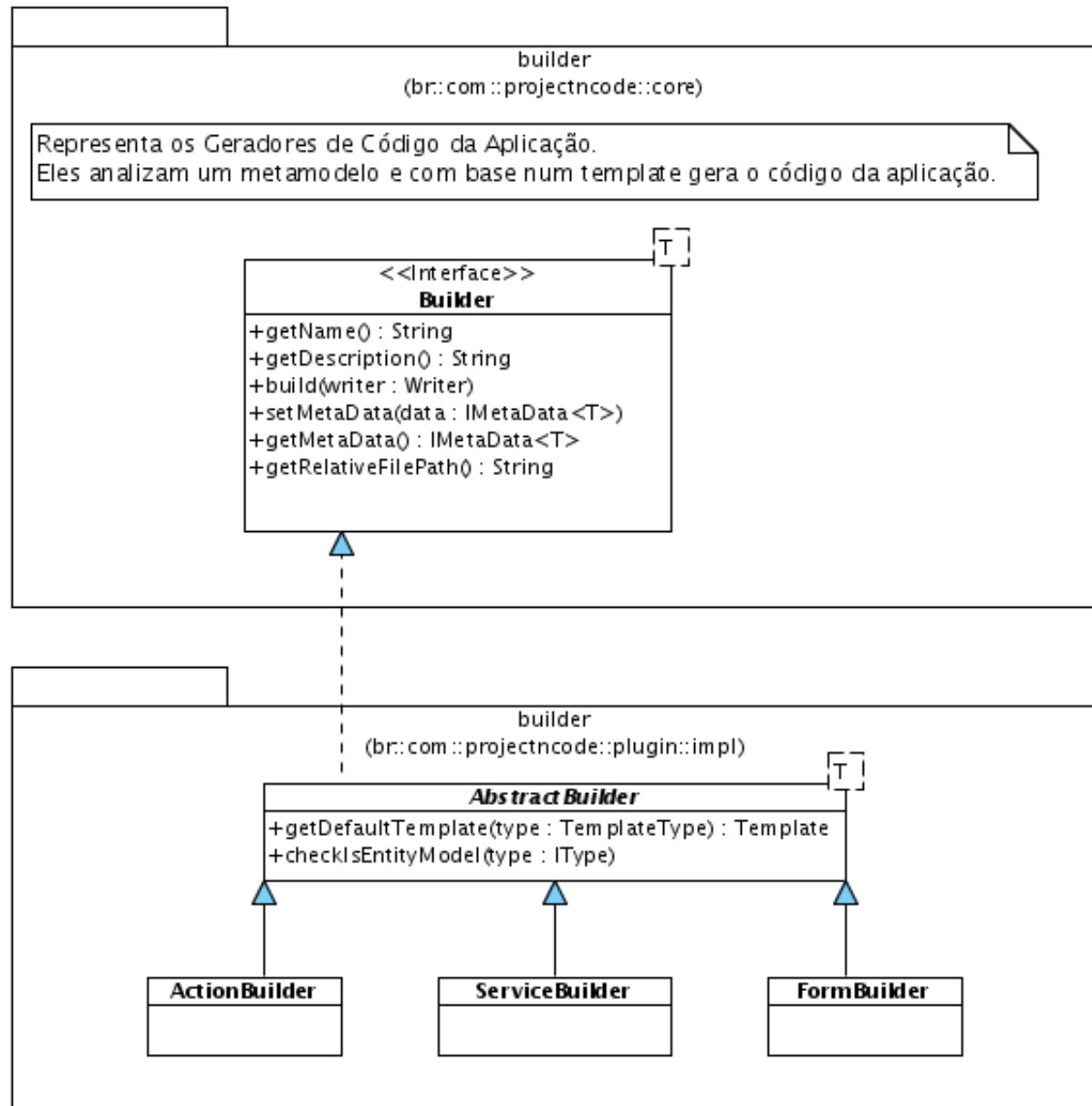
 HibernateAnnotationAnalyzer	
 HibernateMappingAnalyzer	
 JavaReflectionAnalyzer	
 JavaSourceAnalyzer	

Diagrama de Classe

DCL (Geral) - Builder



Sumário









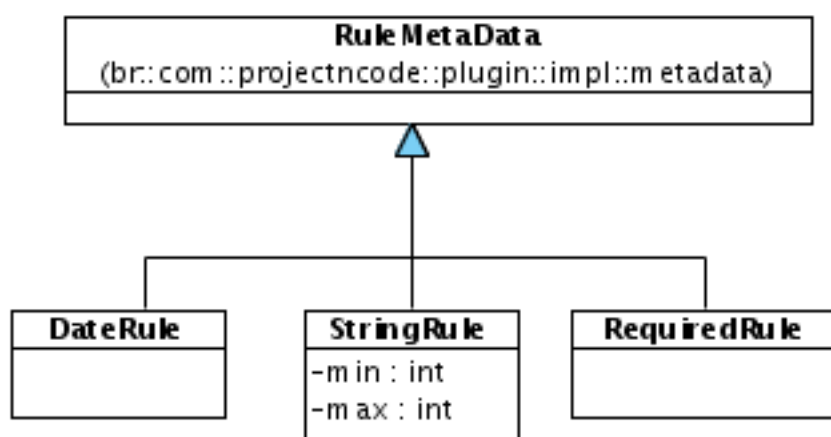
Nome	Documentação
 builder	
	Representa os Geradores de Código da Aplicação. Eles analisam um metamodelo e com base num template gera o código da aplicação.
 Builder	Interface que define os Geradores de Código
 builder	
 AbstractBuilder	
 ActionBuilder	
 FormBuilder	
 ServiceBuilder	

Diagrama de Classe

DCL (Plugin) - FormMetaData (Rules)

Aqui é a estrutura do sistema de Validação do Sistema, como isso varia muito de Framework para framework, isso será pensado posteriormente



Sumário






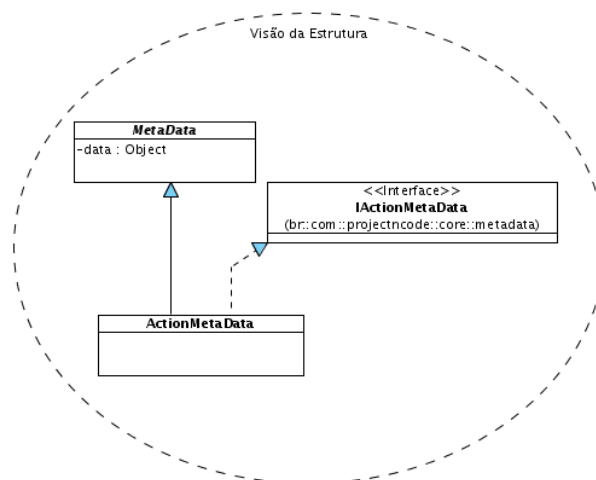
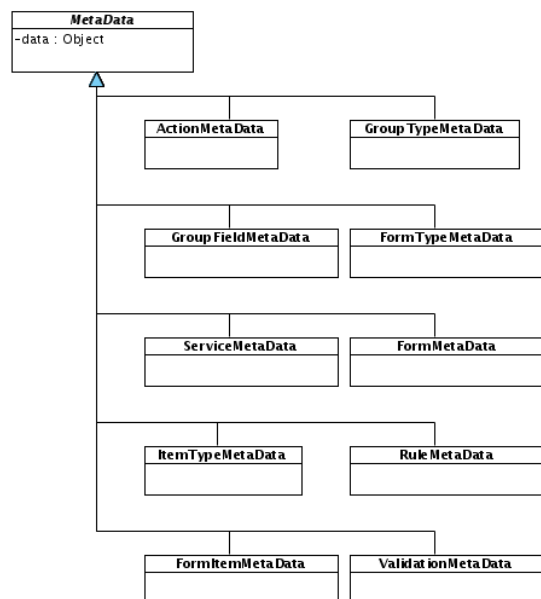
Nome	Documentação
	Aqui é a estrutura do sistema de Validação do Sistema, como isso varia muito de Framework para framework, isso será pensado posteriormente
 DateRule	
 RequiredRule	
 RuleMetaData	
 StringRule	

Diagrama de Classe

DCL (Plugin) - MetaData (Implementação)

Cada classe dessa implementação, *realiza* a interface correspondente no diagrama: DCL - MetaData (Hierarquia), seguindo o mesmo modelo exibido em: Visão da Estrutura



Sumário

















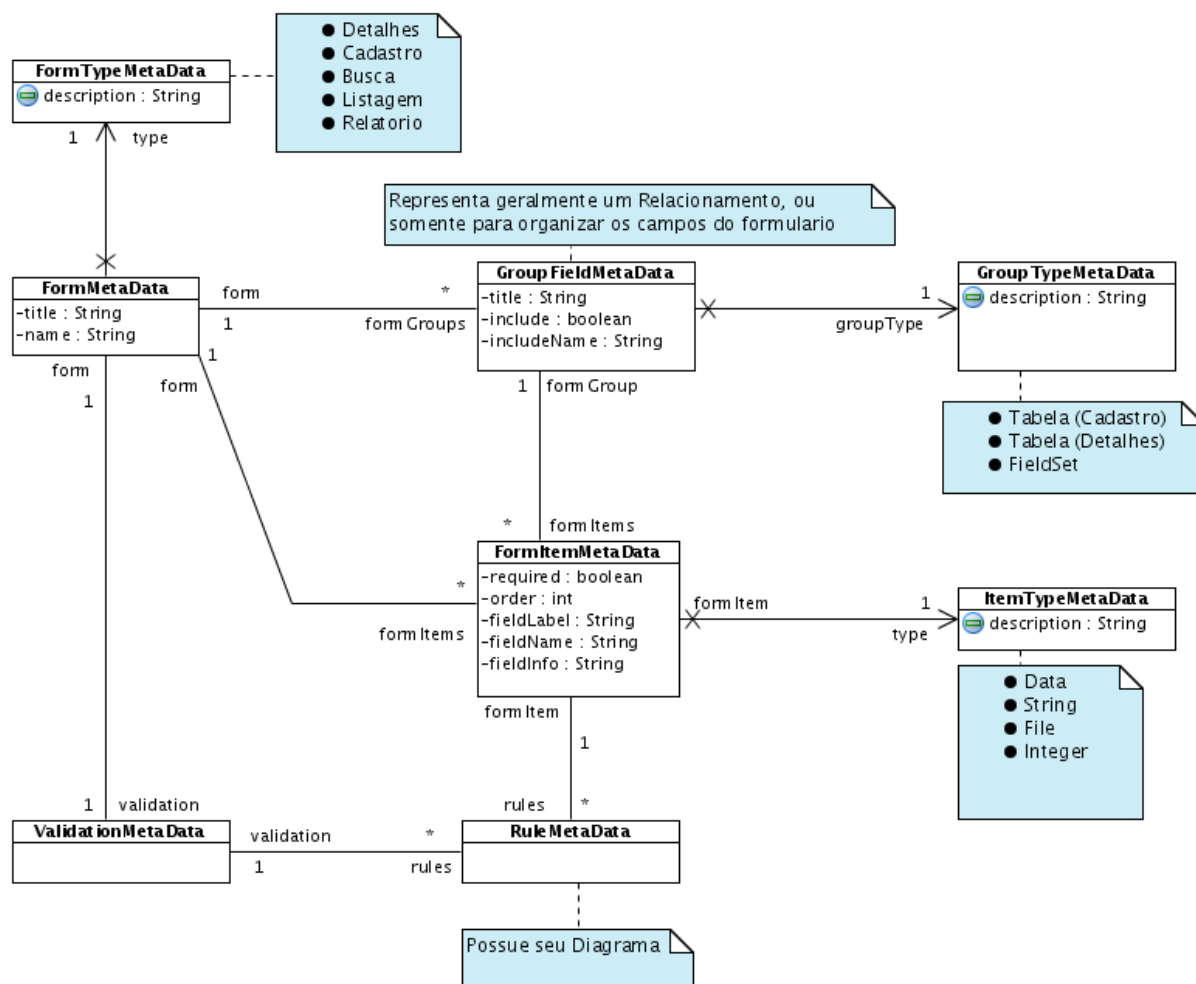
Nome	Documentação
	Cada classe dessa implementação, <u>realiza</u> a interface correspondente no diagrama: DCL - MetaData (Hierarquia) , seguindo o mesmo modelo exibido em: Visão da Estrutura
 ActionMetaData	
 FormItemMetaData	
 FormMetaData	
 FormTypeMetaData	
 GroupFieldMetaData	
 GroupTypeMetaData	
 ItemTypeMetaData	
 MetaData	
 RuleMetaData	
 ServiceMetaData	
 ValidationMetaData	
 Visão da Estrutura	
 ActionMetaData	
 IActionMetaData	
 MetaData	

Diagrama de Classe

DCL (Plugin) - MetaData (Relacionamentos)



Sumário














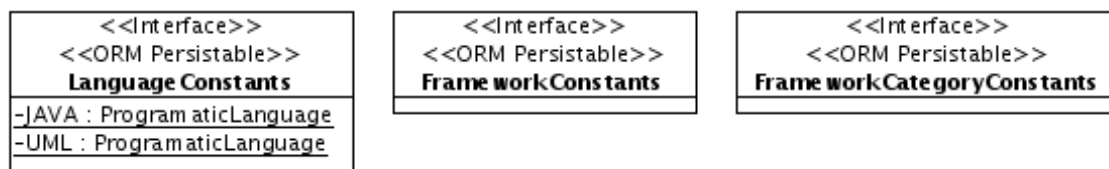
Nome	Documentação
	Representa geralmente um Relacionamento, ou somente para organizar os campos do formulario
	<ul style="list-style-type: none">• Tabela (Cadastro)• Tabela (Detalhes)• FieldSet
	<ul style="list-style-type: none">• Data• String• File• Integer
	<ul style="list-style-type: none">• Detalhes• Cadastro• Busca• Listagem• Relatorio
	Possue seu Diagrama
 FormItemMetaData	
 FormMetaData	
 FormTypeMetaData	
 GroupFieldMetaData	
 GroupTypeMetaData	
 ItemTypeMetaData	
 RuleMetaData	
 ValidationMetaData	

Diagrama de Classe

DCL - Constants



TODO: Deve ter um stratic, adicionando as linguagens no Pool, colocar os ID dos objetos tambem

Sumário





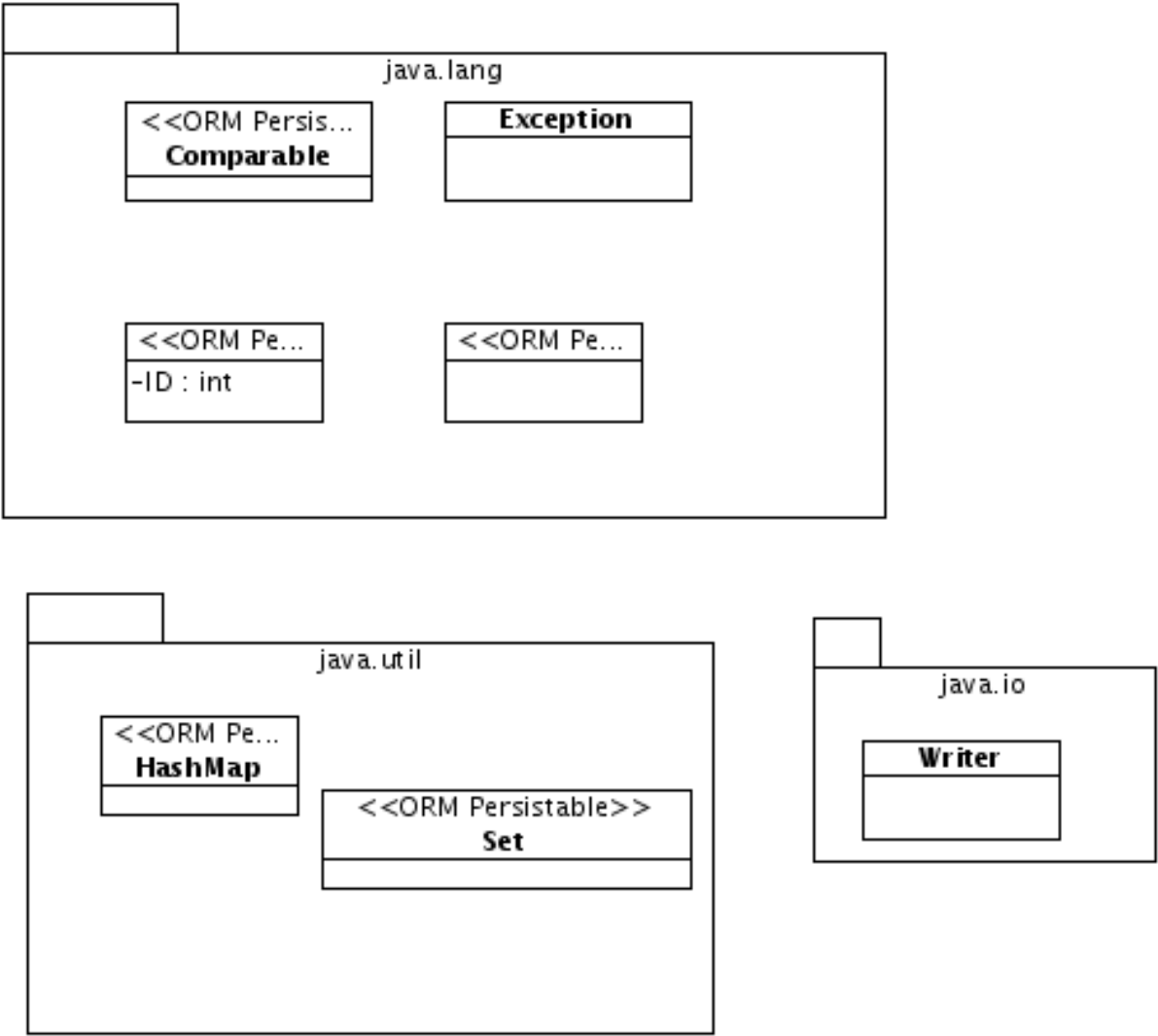
Nome	Documentação
	TODO: Deve ter um stratic, adicionando as linguagens no Pool, colocar os ID dos objetos tambem
 FrameworkCategoryConstants	
 FrameworkConstants	
 LanguageConstants	

Diagrama de Classe

DCL - Java API



Sumário











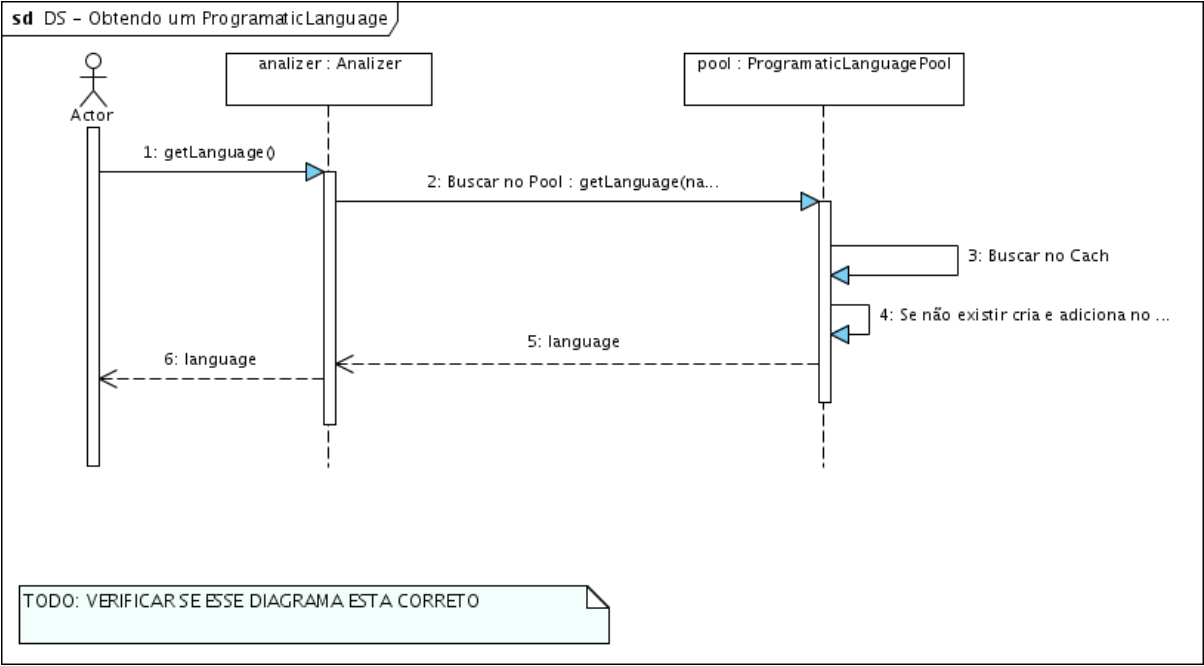
Nome	Documentação
 java.io	
 Writer	
 java.lang	
 Class	
 Comparable	
 Exception	
 Package	
 java.util	
 HashMap	
 Set	

Diagrama de Sequencia

DS - Obtendo um ProgramaticLanguage

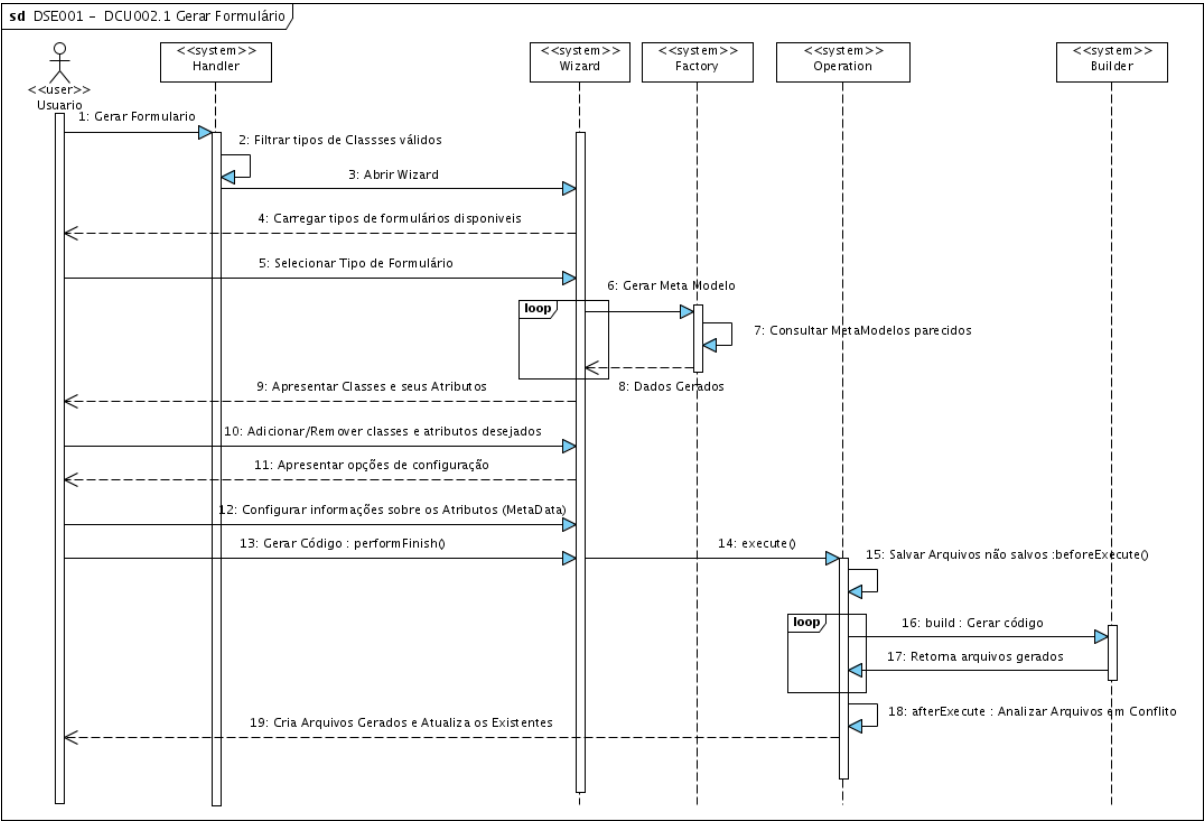


Sumário

Nome	Documentação
	TODO: VERIFICAR SE ESSE DIAGRAMA ESTA CORRETO
Actor	
analizer : Analyzer	
pool	

Diagrama de Sequencia

DSE001 - DCU002.1 Gerar Formulário

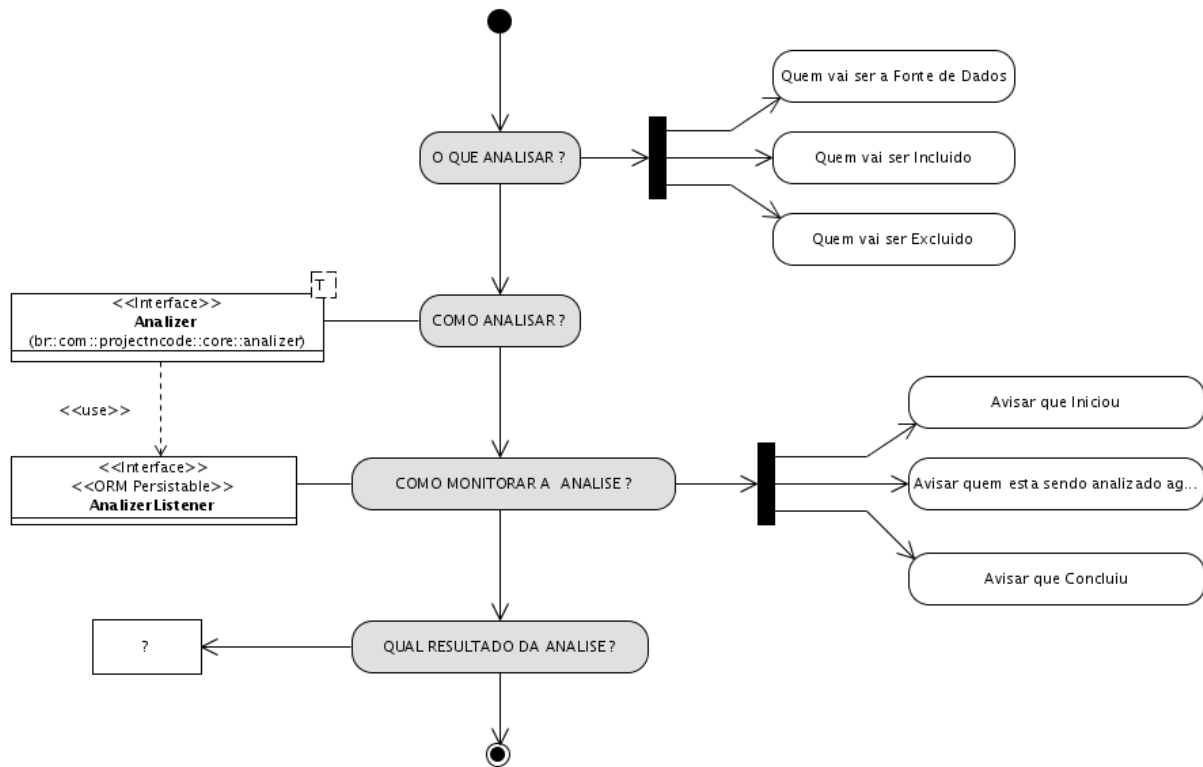


Sumário

Nome	Documentação
Builder	
CombinedFragment	Gera o Código para cada classes selecionada
CombinedFragment2	
Factory	
Handler	
InteractionOperand	
InteractionOperand	
Operation	
Usuario	
Wizard	

Diagrama de Atividades

DA - Logica dos Analizers



Sumário


















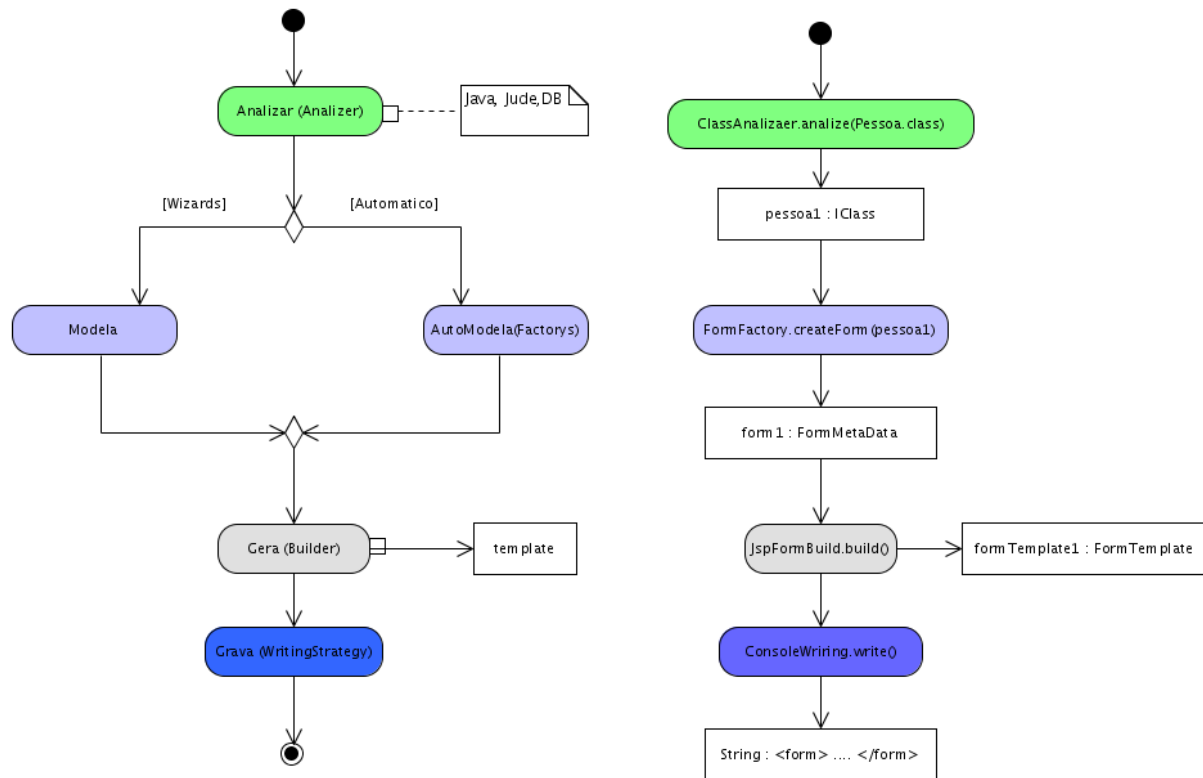




















Nome	Documentação
 ?	
 ActivityFinalNode2	
 Analizer	<p>Representa a interface que define um Analizador de Dados, que pode ser considerado tambem como um Sistema de Importação ou Engenharia Reversa.</p> <p>TODO: Explicar como os analizers devem se registrar para serem visualizados pelos Windards, e explicar as possibilidades de desenvolvimento de Plugins para analizers</p>
 AnalizerListener	
 Avisar que Concluiu	
 Avisar que Iniciou	
 Avisar quem esta sendo analisado agora	
 COMO ANALISAR ?	
 COMO MONITORAR A ANALISE ?	
 ForkNode	
 ForkNode2	
 InitialNode4	
 O QUE ANALISAR ?	
 QUAL RESULTADO DA ANALISE ?	
 Quem vai ser Excluido	
 Quem vai ser Incluído	
 Quem vai ser a Fonte de Dados	

Diagrama de Atividades

DA - Processo de Geração



Sumário

Nome	Documentação
	Java, Jude,DB
 ActivityFinalNode	
 Analizar (Analyzer)	
 AutoModela(Factorys)	
 ClassAnalizaer.analyze(Pessoa.class)	
 ConsoleWriring.write()	
 DecisionNode	
 DecisionNode2	
 FormFactory.createForm(pessoa1)	
 Gera (Builder)	
 Grava (WritingStrategy)	
 InitialNode2	
 InitialNode3	
 JspFormBuild.build()	
 Modela	
 String : <form> </form>	
 form1 : FormMetaData	
 formTemplate1 : FormTemplate	
 pessoa1 : IClass	
 template	