

# Apêndice A

## Especificações DCL 2.0

Para avaliar a linguagem proposta neste trabalho, o Capítulo 5 apresenta um experimento, onde um sistema real teve sua arquitetura especificada na linguagem DCL 2.0. Baseado no conceito de camadas, o sistema é composto de seis camadas arquiteturais, onde cada uma delas tem sua responsabilidade definida de acordo com as convenções arquiteturais. Neste capítulo são apresentadas em detalhes as especificações DCL 2.0 para cada uma das seis camadas.

### A.1 Camada Comum

Como o próprio nome diz, a camada comum tem como responsabilidade conter todos os artefatos que serão de uso geral da aplicação. Por exemplo, classes utilitárias e constantes. Por esta razão, o nível de acoplamento dessa camada deve ser baixo, pois qualquer aumento no acoplamento é propagado para todas as demais camadas, uma vez que todas outras camadas são dependentes dessa. Pode-se observar na especificação que são poucos os pacotes declarados nessa camada.

```
1  architecture comum{
2    ecossistema{matching: "xx.yyy.zzzzzzz.{?}";
3      sistema{matching: "{?}";
4        comuns{matching: "comuns";
5          constantes{
6            matching: "constantes";
7            Constante{
8              matching: "Constantes{?}";
9              restrictions{
10               can declare only platform.java.lang;
11             }
12           }
13         }
14       }
15     }
16   }
```

```

13     }
14     utils{matching: "utils";
15     Util{matching: "{?}Util";
16     restrictions{
17         can declare only platform.java.lang,
18         platform.java.util,
19         platform.java.text,
20         platform.wwwwwww.comuns.utils;
21     }
22     }
23 }
24 }
25 }
26 }
27 ignore "xx", "yyy", "zzzzzz", "java", ".classpath",
28 ".jazzignore", "main", "test", "classes", "resources",
29 ".project", ".settings*", "bin*", "target*", "bin",
30 "pom.xml", "src";
31 }

```

## A.2 Camada Domínio

Os componentes dessa camada são utilizados para transporte de dados, persistência e enumerações. Os componentes VOs são utilizados para garantir a rastreabilidade entre código e os conceitos essenciais do sistema, uma vez que eles representam os principais conceitos do domínio.

```

1  architecture dominio{
2      ecosistema{matching: "xx.yyy.zzzzzzz.{?}";
3      sistema{matching: "{?}";
4      entidades{matching: "entidades";
5      modulo{matching: "{?}";
6      VO{matching: "{?}VO";
7      restrictions{
8          must implement platform.java.io.Serializable;
9          must extend BaseVO message "";
10     }
11     }
12     AudVO{matching: "{?}AudVO"; description: "";
13     restrictions{
14         must extend
15         platform.wwwwwww.comuns.auditoria.ProAuditoriaVO;
16     }
17     }
18     restrictions{
19         requires VO,AudVO;

```

```

20         }
21     }
22     BaseVO{matching: "{?}BaseVO"; description: "";
23         restrictions{
24             must extend platform.wwwwwww.entidades.ProVO;
25         }
26     }
27     restrictions{
28         can declare only enums .**, entidades .**,
29         comum.ecosistema.sistema.comuns .**,
30         platform.wwwwwww.comuns .**,
31         platform.wwwwwww.entidades .**,
32         platform.org.hibernate.annotation ,
33         platform.org.hibernate.envers ,
34         platform.org.hibernate.envers.RevisionType ,
35         platform.java.lang ,
36         platform.java.util , platform.java.text ,
37         platform.javax.persistence , platform.ssc-interface .**;
38     }
39 }
40 enums{matching: "enums";
41 modulo{matching: "{?}";
42     Enum{matching: "{?}Enum";}
43     restrictions{
44         requires entidades.modulo.VO;
45     }
46 }
47 }
48 restrictions{
49     requires entidades.BaseVO;
50 }
51 }
52 }
53 ignore "xx", "yyy", "zzzzzzz", "java", ".classpath",
54 ".jazzignore", "main", "test",
55 "classes", "resources*", ".project", ".settings*",
56 "bin*", "target*", "bin", "pom.xml", "src";
57 }

```

## A.3 Camada Interface de Negócio

A camada de interface de negócio tem a função de criar um contrato entre as camadas *Web*, *mobile* e legado de um lado, e camada de negócio de outro. A camada é subdividida em três módulos: interfaces, exceções e comuns. Esta também é uma camada bastante restrita, suas dependências são mínimas, limitando-se a algumas anotações da plataforma JEE.

```

1  architecture negocio-interface{
2      ecosistema{matching: "xx.yyy.zzzzzzz.{?}";
3      sistema{matching: "{?}";
4      interfaces{matching: "interfaces";
5      modulo{
6          matching: "{?}";
7          IFacade{matching: "I{?}Facade";
8              restrictions{
9                  requires dominio.ecosistema.sistema.entidades.modulo.VO;
10             }}
11         IFacadeWS{
12             matching: "I{?}FacadeWS";
13             restrictions{
14                 requires dominio.ecosistema.sistema.entidades.modulo.VO;
15             }}
16         restrictions{
17             requires dominio.ecosistema.sistema.entidades.modulo.VO;
18         }}
19         IBaseFacade{
20             matching: "I{?}BaseFacade";
21             restrictions{
22                 requires dominio.ecosistema.sistema.entidades.BaseVO;
23             }}
24         IBaseFacadeWS{matching: "I{?}BaseFacadeWS";
25             restrictions{
26                 requires dominio.ecosistema.sistema.entidades.BaseVO;
27             }}}
28     comuns{matching: "comuns";
29     modulo{}}
30     excecoes{
31         matching: "excecoes";
32         modulo-excecoes{matching: "{?}";
33         Excecao{matching: "{?}Exception";
34         }
35         restrictions{
36             requires dominio.ecosistema.sistema.entidades.modulo.VO;
37         }}
38     BaseExcecao{matching: "{?}BaseException";
39     restrictions{
40         requires dominio.ecosistema.sistema.entidades.BaseVO;
41     }}}
42     restrictions{
43         can declare only platform.org.hibernate.annotation ,
44         platform.java.lang , platform.java.util , platform.java.text ,
45         platform.javax.persistence , platform.javax.ejb.Remote ,
46         platform.javax.ejb , platform.javax.jws , platform.javax.ws ,
47         platform.javax.xml.bind.annotation ,
48         platform.wwwwwww.entidades .**,
49         platform.wwwwwww.comuns.** , dominio.ecosistema.sistema.entidades .**,
50         platform.ssc-interface .**;}}}
51     ignore "java", ".classpath", ".jazzignore", "main", "test", "classes", "
52         resources", ".project", ".settings*", "bin*", "target*", "bin", "pom.xml", "
53         src";
54 }

```

## A.4 Camada Negócio

A camada de negócio inclui componentes responsáveis por implementar padrões de projeto. A sua responsabilidade principal é executar regras de negócios específicas via padrão de projeto *Especificação* [Evans, 2004]. Outro padrão de projeto contido nesta camada é o padrão conhecido como *SessionFacade*, muito comum em arquiteturas JEE . A camada de negócio tem acesso às camadas infra-estrutura e domínio e deve implementar *interfaces* da camada *negocio-interface*.

```

1  architecture negocio{
2    ecosistema{
3      matching: "xx.yyy.zzzzzzz.{?}";
4      sistema{
5        matching: "{?}";
6        negocio{
7          matching: "negocio";
8          modulo{
9            matching: "{?}";
10           IFacadeImpl{
11             matching: "I{?}FacadeImp";
12             restrictions{
13               requires dominio.ecosistema.sistema.entidades.modulo.VO;
14               can declare only platform.wwwwwww,
15               platform.org.hibernate.annotation,
16               platform.java.lang,
17               platform.java.util,
18               platform.java.text,
19               platform.javax.persistence,
20               dominio.ecosistema.sistema.entidades,
21               platform.ssc-interface;
22             }
23           }
24           Especification{
25             matching: "{?}RN";
26             restrictions{
27               requires dominio.ecosistema.sistema.entidades.modulo.VO;
28               can declare only Especification,
29               BaseEspecificacion,
30               dominio.ecosistema.sistema.entidades **,
31               dominio.ecosistema.sistema.enums **,
32               comum.ecosistema.sistema.comuns **,
33               negocio-interface.ecosistema.sistema.excecoes **,
34               infraestrutura.ecosistema.sistema **,
35               platform.ssc-interface.enumerations **,
36               platform.ssc-interface.interfaces **,
37               platform.ssc-interface.interfaces.dto **,
38               platform.wwwwwww.negocio **,
39               platform.wwwwwww.comum **,
40               platform.wwwwwww.entidades **,
41               platform.wwwwwww.comuns **,
42               platform.wwwwwww.persistencia **,

```

```

43         platform.wwwwwww.criptografia.utils ,
44         platform.wwwwwww.json ,
45         platform.org.hibernate.annotation ,
46         platform.org.slf4j ,
47         platform.org.apache.commons ,
48         platform.org.apache.commons.lang ,
49         platform.org.apache.commons.collections ,
50         platform.org.apache.commons.beanutils ,
51         platform.org.joda.time.format ,
52         platform.org.joda.time ,
53         platform.java.io ,
54         platform.java.io.Serializable ,
55         platform.java.lang ,
56         platform.java.util ,
57         platform.java.text ,
58         platform.javax.persistence ,
59         platform.javax.jws ,
60         platform.javax.xml.bind.annotation ;
61     }
62 }
63 EspecificationHelper{
64     matching: "{?}HelperRN";
65     restrictions{
66         requires dominio.ecosistema.sistema.entidades.modulo.VO;
67         can declare only platform.org.hibernate.annotation ,
68         platform.java.lang platform.java.util ,
69         platform.java.text ,
70         platform.javax.persistence ,
71         platform.javax.jws ,
72         dominio.ecosistema.sistema.entidades ,
73         platform.javax.xml.bind.annotation ,
74         platform.ssc-interface ;
75     }
76 }
77 }
78 BaseEspecification{
79     matching: "{?}BaseRN";
80 }
81 IBaseFacadeImpl{
82     matching: "{?}BaseFacadeImp";
83     restrictions{
84         requires dominio.ecosistema.sistema.entidades.BaseVO;
85         can declare only negocio .**,
86         dominio.ecosistema.sistema.entidades .**,
87         negocio-interface.ecosistema .**,
88         platform.ssc-interface .**,
89         platform.wwwwwww .**,
90         platform.org.hibernate.annotation ,
91         platform.java.lang platform.java.util ,
92         platform.java.text ,
93         platform.javax.persistence ,
94         platform.javax.ejb ,
95         platform.org.slf4j ,
96         platform.org.apache.commons.lang ,

```

```

97         platform.ssc-interface;
98     }
99 }
100 }
101 rest{
102     matching: "rest";
103     modulo{
104         matching: "{?}";
105     }
106     RestActivator{
107         matching: "{?}RestActivator";
108     }
109     RestFacadeImp{
110         matching: "{?}RestFacadeImp";
111     }
112 }
113 soap{
114     matching: "soap";
115     modulo{
116         matching: "{?}";
117         WSFacadeImp{
118             matching: "{?}WSFacadeImp";
119         }
120     }
121     BaseWSFacadeImp{
122         matching: "{?}BaseWSFacadeImp";
123     }
124 }
125 schedule{
126     matching: "schedule";
127     modulo{
128         matching: "{?}";
129         Schedule{
130             matching: "{?}Schedule";
131             restrictions{
132                 requires dominio.ecosistema.sistema.entidades.modulo.VO;
133             }
134         }
135         restrictions{
136             requires dominio.ecosistema.sistema.entidades.modulo.VO;
137         }
138     }
139     BaseSchedule{
140         matching: "{?}BaseSchedule";
141     }
142 }
143 }
144 restrictions{
145     cannot declare platform.java.lang.reflect;
146 }
147 }
148 ignore "xx", "yyy", "zzzzzz", "java", ".classpath", ".jazzignore",
149 "bin*", "target*", "bin", "pom.xml", "src";
150 }

```

## A.5 Camada Infraestrutura

A camada de infraestrutura é responsável por fornecer vários serviços de persistência e acesso a dados por meio de componentes que implementam o padrão *DAO*. Por esta razão, esta camada é acoplada a *frameworks* específicos para lidar com persistência, tais como, *Hibernate* e *JPA*. O padrão de projeto *DTO* também está presente nesta camada e sua responsabilidade é prover objetos de transporte de dados dentro da camada de infraestrutura.

```

1  architecture infraestrutura{
2    ecosistema{
3      matching: "xx.yyy.zzzzzzz.{?}";
4      sistema{
5        matching: "{?}";
6        persistencia{
7          matching: "persistencia";
8          modulo{
9            matching: "{?}";
10           DAO{
11             matching: "{?}DAO";
12             restrictions{
13               must extend BaseDAO;
14               requires dominio.ecosistema.sistema.entidades.modulo.VO;
15             }
16           }
17           AudDAO{
18             matching: "{?}AudDAO";
19             restrictions{
20               requires dominio.ecosistema.sistema.entidades.modulo.AudVO;
21             }
22           }
23           DAOHelper{
24             matching: "{?}DAOHelper";
25             restrictions{
26               requires DAO;
27             }
28           }
29         }
30         BaseDAO{
31           matching: "{?}BaseDAO";
32           restrictions{
33             must extend platform.wwwwwww.persistencia.ProBaseDAO;
34             requires dominio.ecosistema.sistema.entidades.BaseVO;
35           }
36         }
37       }
38     entidades{
39       matching: "entidades";
40       modulo{
41         matching: "{?}";
42         DTO{

```



```

43         matching: "{?}DTO";
44     }
45     restrictions{
46         requires dominio.ecosistema.sistema.entidades.modulo.VO;
47     }
48 }
49 }
50 restrictions{
51     can declare only entidades .**,
52     persistencia .**,
53     dominio.ecosistema .**,
54     comum.ecosistema.sistema .**,
55     negocio-interface.ecosistema.sistema .**,
56     platform.ssc-interface .**,
57     platform.wwwwwww .**,
58     platform.org.hibernate .**,
59     platform.org.springframework.ldap .**,
60     platform.org.springframework.context .**,
61     platform.org.springframework.context.support .**,
62     platform.org.apache.commons.lang ,
63     platform.org.apache.commons.collections ,
64     platform.org.slf4j ,
65     platform.java.io ,
66     platform.java.io.Serializable ,
67     platform.java.util ,
68     platform.java.lang ,
69     platform.javax.persistence ,
70     platform.javax.mail ,
71     platform.javax.mail.internet ,
72     platform.javax.naming ,
73     platform.javax.naming.directory ;
74 }
75 }
76 }
77 ignore "xx", "yyy", "zzzzzz", "java", ".classpath", ".jazzignore", "main";
78 }

```

## A.6 Camada Web

A camada web possui a responsabilidade de responder a eventos acionados pelo usuário. Basicamente, camada funciona da seguinte forma. (i) a camada fornece a interface de usuário no formato *HTML*; (ii) recebe as informação que vêm do usuário por meio do componente *Page*; (iii) transforma essas informações em objetos de domínio (*VOs*) utilizando o componente *Binder*; (iv) envia a informação para as camadas inferiores; (v) e exibe a resposta ao usuário por meio do componente *MessageHelper*. Esse conjunto de operações e a colaboração entre objetos é orquestrado pelo componente *Controller*.

```

1  architecture web{
2    ecosistema{
3      matching: "xx.yyy.zzzzzzz.{?}";
4      sistema{
5        matching: "{?}";
6        listener{
7          matching: "listener";
8          AdminBaseListener{
9            matching: "{?}AdminBaseListener";
10         }
11       }
12     controle{
13       matching: "controle";
14       modulo{
15         matching: "{?}";
16         Controller{
17           matching: "{?}Ctr";
18           restrictions{
19             requires dominio.ecosistema.sistema.entidades.modulo.VO;
20           }
21         }
22       }
23       BaseController{
24         matching: "{?}BaseCtr";
25         restrictions{
26           requires dominio.ecosistema.sistema.entidades.BaseVO;
27         }
28       }
29       WindowIntroducao{
30         matching: "{?}WindowIntroducao";
31       }
32       WindowIndex{
33         matching: "{?}WindowIndex";
34       }
35       restrictions{
36         can declare only platform.org.hibernate.annotation ,
37         platform.java.lang ,
38         platform.java.util ,
39         platform.java.text ,
40         platform.javax.persistence ,
41         platform.javax.naming ,
42         platform.javax.servlet .**, //Rever
43         platform.org.zkoss.zk .**,
44         platform.org.zkoss.zul .**,
45         platform.org.zkoss.image .**,
46         platform.org.zkoss.zkplus.databind.BindingListModelSet ,
47         platform.org.apache ,
48         platform.org.joda ,
49         platform.org.hibernate.envers.RevisionType ,
50         platform.org.slf4j ,
51         platform.ssc-interface .**,
52         platform.wwww.wwww.entidades .**,
53         platform.wwww.wwww.criptografia.utils .**,
54         platform.wwww.wwww.json .**,

```

```

55         platform.wwwwwww.controle .**,
56         platform.wwwwwww.controle.componente .**,
57         platform.wwwwwww.comuns .**,
58         platform.wwwwwww.comuns.utils .**,
59         platform.wwwwwww.visao.visao-heper .**,
60         negocio-interface.ecosistema.sistema.interfaces .**,
61         dominio.ecosistema.sistema.enums .**,
62         dominio.ecosistema.sistema.entidades .**,
63         comum.ecosistema.sistema.comuns .**;
```

```

64     }
65 }
66 helper{
67     matching: "helper";
68     modulo{
69         matching: "{?}";
70         HelperView{
71             matching: "{?}HelperView";
72             restrictions{
73                 requires controle.modulo.Controller;
74             }
75         }
76         MessageHelper{
77             matching: "{?}MessageHelper";
78             restrictions{
79                 requires controle.modulo.Controller;
80             }
81         }
82         BinderHelper{
83             matching: "{?}BinderHelper";
84             restrictions{
85                 requires controle.modulo.Controller;
86             }
87         }
88     }
89     BaseHelperView{
90         matching: "{?}BaseHelperView";
91         restrictions{
92             requires controle.BaseController;
93         }
94     }
95 }
96 visao{
97     matching: "visao";
98     modulo{
99         matching: "{?}";
100         Page{
101             matching: "{?}{extension=zul}";
102             restrictions{
103                 requires controle.modulo.Controller;
104             }
105         }
106     }
107 }
108 js{
```

```

109     matching: "js";
110     modulo{
111         matching: "{?}";
112         Script{
113             matching: "{?}{extension=js}";
114         }
115     }
116 }
117 css{
118     matching: "css";
119     modulo{
120         matching: "{?}";
121         Style{
122             matching: "{?}{extension=css}";
123         }
124     }
125 } test-funcionais{
126     matching: "test";
127     modulo{
128         matching: "{?}";
129         FunctionalTest{
130             matching: "{?}FunctionalTest";
131             restrictions{
132                 requires controle.modulo.Controller;
133             }
134         }
135     }
136     BaseFunctionalTest{
137         matching: "{?}BaseFunctionalTest";
138         restrictions{
139             requires controle.BaseController;
140         }
141     }
142     restrictions{
143         can declare only funtional-test.wwwwwww.test ,
144         funtional-test.com.thoughtworks .**,
145         funtional-test.org.openqa.selenium .**,
146         platform.java.lang;
147     }
148 }
149 }
150 }
151 ignore "xx", "yyy", "zzzzzz", "java", ".classpath",
152 ".jazzignore", "main", "classes", "resources*",
153 ".project", "webapp*", "versao.txt",
154 "mensagem.properties", ".settings*",
155 "bin*", "target*", "bin", "pom.xml", "src",
156 "imagens*", "WEB-INF*", "META-INF";
157 }

```

## A.7 Plataforma de Reúso

Como relatado no Capítulo 3, DCL 2.0 permite especificar tanto componentes internos como componentes externos ao sistema. Uma das principais vantagens dessa funcionalidade é permitir o reúso entre especificações DCL 2.0. No experimento, uma especificação foi criada para conter componentes relacionados a *frameworks*, *APIs*, utilitários e plataforma *Java*. Após a especificação dessa camada, a equipe de arquitetos percebeu a necessidade de dividir a especificação em mais partes devido a quantidade de componentes envolvidos. No entanto, essa atividade foi planejada para outro projeto.

```
1  architecture platform{
2    ssc-interface{
3      matching: "xx.yyy.zzzzzzz.ssc.*";
4    interfaces{
5      matching: "interfaces.*";
6    dto{
7      matching: "dto.*";
8    }
9    base{
10     matching: "base.*";
11   }
12 }
13 enumerations{
14   matching: "enumerations.*";
15 }
16 comuns{
17   matching: "comuns.*";
18   constantes{
19     matching: "constantes.*";
20   }
21 }
22 }
23 org{
24   matching: "org.*";
25   fest{
26     matching: "fest.*";
27     assertions{
28       matching: "assertions.*";
29     }
30   }
31   zkoss{
32     matching: "zkoss.*";
33     zk{
34       matching: "zk.*";
35       ui{
36         matching: "ui.*";
37         util{
38           matching: "util.*";
39         }

```

```
40     event{
41         matching: "event.*";
42     }
43 }
44 }
45 zul{
46     matching: "zul.*";
47     ext{
48         matching: "ext.*";
49     }
50 }
51 zkplus{
52     matching: "zkplus.*";
53     databind{
54         matching: "databind.*";
55         BindingListModelSet{
56             matching: "BindingListModelSet";
57         }
58     }
59 }
60 image{
61     matching: "image.*";
62 }
63 util{
64     matching: "util.*";
65     media{
66         matching: "media.*";
67     }
68 }
69 }
70 joda{
71     matching: "joda.*";
72     time{
73         matching: "time.*";
74         format{
75             matching: "format.*";
76         }
77     }
78 }
79 slf4j{
80     matching: "slf4j.*";
81 }
82 springframework{
83     matching: "springframework.*";
84     context{
85         matching: "context.*"; //org.springframework.context.support
86         support{
87             matching: "support.*";
88         }
89     }
90     ldap{
91         matching: "ldap.*";
92         filter{
93             matching: "filter.*";
94         }
95     }
96 }
```

```
94     core{
95         matching: "core.*";
96         support{
97             matching: "support.*";
98         }
99     }
100     odm{
101         matching: "odm.*";
102         core{
103             matching: "core.*";
104         }
105         annotations{
106             matching: "annotations.*";
107         }
108     }
109 }
110 }
111 apache{
112     matching: "apache.*";
113     commons{
114         matching: "commons.*";
115         collections{
116             matching: "collections.*";
117         }
118         lang{
119             matching: "lang.*";
120         }
121         beanutils{
122             matching: "beanutils.*";
123         }
124     }
125 }
126 hibernate{
127     matching: "org.hibernate.*";
128     transform{
129         matching: "transform.*";
130     }
131     criterion{
132         matching: "criterion.*";
133     }
134     annotation{
135         matching: "annotations.*";
136     }
137     core{
138         matching: "org.hibernate.*";
139     }
140     envers{
141         matching: "envers.*";
142         query{
143             matching: "query.*";
144             property{
145                 matching: "property.*";
146             }
147             criteria{
```

```

148         matching: "criteria.*";
149     }
150 }
151 RevisionType{
152     matching: "RevisionType";
153 }
154 entities{
155     matching: "entities.*";
156     mapper{
157         matching: "mapper.*";
158         relation{
159             matching: "relation.*";
160             query{
161                 matching: "query.*";
162             }
163         }
164     }
165 }
166 }
167 entities{
168     matching: "entities.*";
169     mapper{
170         matching: "mapper.*";
171         relation{
172             matching: "relation.*";
173             query{
174                 matching: "query.*";
175             }
176         }
177     }
178 }
179 }
180 jboss{
181     matching: "jboss.*";
182     security{
183         matching: "security.*";
184         annotation{
185             matching: "annotation.*";
186         }
187     }
188     ws{
189         matching: "ws.*";
190         api{
191             matching: "api.*";
192             annotation{
193                 matching: "annotation.*";
194             }
195         }
196     }
197 }
198 junit{
199     matching: "junit.*";
200     runner{
201         matching: "runner.*";

```



```
202     }
203   }
204 }
205 javax{
206   matching: "javax.*";
207   annotation{
208     matching: "annotation.*";
209     security{
210       matching: "security.*";
211     }
212   }
213   ejb{
214     matching: "ejb.*";
215     Remote{
216       matching: "Remote";
217     }
218   }
219   servlet{
220     matching: "servlet.*";
221     http{
222       matching: "http.*";
223     }
224     annotation{
225       matching: "annotation.*";
226     }
227   }
228   naming{
229     matching: "naming.*";
230     directory{
231       matching: "directory.*";
232     }
233   }
234   persistence{
235     matching: "persistence.*";
236   }
237   jws{
238     matching: "jws.*";
239   }
240   ws{
241     matching: "ws.*";
242     rest{
243       matching: "rs.*";
244       core{
245         matching: "core.*";
246       }
247     }
248   }
249   xml{
250     matching: "xml.*";
251     bind{
252       matching: "bind.*";
253       annotation{
254         matching: "annotation.*";
255       }
256     }
257   }
258 }
```

```
256     }
257     ws{
258         matching: "ws.*";
259         handler{
260             matching: "handler.*";
261         }
262     }
263 }
264 imageio{
265     matching: "imageio.*";
266     stream{
267         matching: "stream.*";
268     }
269 }
270 mail{
271     matching: "mail.*";
272     internet{
273         matching: "internet.*";
274     }
275 }
276 accessibility{
277     matching: "accessibility.*";
278 }
279 }
280 java{
281     matching: "java.*";
282     awt{
283         matching: "awt.*";
284         image{
285             matching: "image.*";
286         }
287     }
288     io{
289         matching: "io.*";
290         Serializable{
291             matching: "Serializable";
292         }
293     }
294     lang{
295         matching: "lang.*";
296         Cloneable{
297             matching: "Cloneable";
298         }
299         reflect{
300             matching: "reflect.*";
301         }
302     }
303     sql{
304         matching: "sql.*";
305     }
306     util{
307         matching: "util.*";
308         concurrent{
309             matching: "concurrent.*";
```

```
310     }
311   }
312   text{
313     matching: "text.*";
314   }
315   net{
316     matching: "net.*";
317   }
318   security{
319     matching: "security.*";
320   }
321   math{
322     matching: "math.*";
323   }
324   applet{
325     matching: "applet.*";
326   }
327 }
328 wwwwww{
329   matching: "br.yyy.wwwwww.*";
330   hibernate{
331     matching: "hibernate.*";
332   }
333   entidades{
334     matching: "entidades.*";
335     ProVO{
336       matching: "ProVO";
337     }
338     ProBaseVO{
339       matching: "ProBaseVO";
340     }
341   }
342   comum{
343     matching: "comum.*";
344   }
345   comuns{
346     matching: "comuns.*";
347     utils{
348       matching: "utils.*";
349       imagem{
350         matching: "imagem.*";
351       }
352     }
353     constantes{
354       matching: "constantes.*";
355     }
356     auditoria{
357       matching: "auditoria.*";
358       ProAuditoriaVO{
359         matching: "ProAuditoriaVO";
360       }
361     }
362     anotacoes{
363       matching: "anotacoes.*";
```

```
364     enumeracao{
365         matching: "enumeracao.*";
366     }
367 }
368 exception{
369     matching: "exception.*";
370 }
371 excecoes{
372     matching: "excecoes.*";
373 }
374 validacoes{
375     matching: "validacoes.*";
376 }
377 IProFacade{
378     matching: "IProFacade";
379 }
380 }
381 visao{
382     matching: "visao.*";
383     visao-heper{
384         matching: "helper.*";
385     }
386 }
387 controle{
388     matching: "controle.*";
389     ProCtr{
390         matching: "ProCtr";
391     }
392     componente{
393         matching: "componente.*";
394     }
395 }
396 negocio{
397     matching: "negocio.*";
398     ProBaseRN{
399         matching: "ProBaseRN";
400     }
401 }
402 persistencia{
403     matching: "persistencia.*";
404     ProBaseDAO{
405         matching: "ProBaseDAO";
406     }
407 }
408 json{
409     matching: "json.*";
410 }
411 criptografia{
412     matching: "criptografia.*";
413     utils{
414         matching: "utils.*";
415     }
416 }
417 test{
```

```
418     matching: "test.*";
419   }
420 }
421 netscape{
422     matching: "netscape.*";
423     javascript{
424         matching: "javascript.*";
425     }
426 }
427 }
```

## A.8 Plataforma de Reúso - Testes de Integração

Nesta seção, apresenta-se a especificação para os componentes externos correspondentes a *frameworks* e APIs relacionados a testes de integração. O testes de integração são utilizados para testar a interface de negócio do sistema. De maneira geral, essa camada é responsável por verificar se todas as regras de negócio estão funcionando adequadamente.

```
1  architecture integration-test {
2    org{
3      matching: "org.*";
4      jboss{
5        matching: "jboss.*";
6        arquillian{
7          matching: "arquillian.*";
8          container{
9            matching: "container.*";
10           test{
11             matching: "test.*";
12             api{
13               matching: "api.*";
14             }
15           }
16         }
17         junit{
18           matching: "junit.*";
19         }
20       }
21       shrinkwrap{
22         matching: "shrinkwrap.*";
23         api{
24           matching: "api.*";
25         }
26         spec{
27           matching: "spec.*";
28         }
29       }
30     }
31 }
```

```
29     }
30   }
31 }
32 }
```

## A.9 Plataforma de Reúso - Testes Funcionais

Nesta seção, apresenta-se a especificação para os componentes externos correspondentes a *frameworks* e APIs relacionados a testes funcionais. Os testes funcionais são utilizados para simular a ação do usuário ao utilizar o sistema. De maneira geral, essa camada é responsável por verificar se todos os comandos de interface de usuário estão funcionando adequadamente.

```
1  architecture functional-test{
2    org{
3      matching: "org.*";
4      openqa{
5        matching: "openqa.*";
6        selenium{
7          matching: "selenium.*";
8          firefox{
9            matching: "firefox.*";
10         }
11       }
12     }
13   }
14   com{
15     matching: "com.*";
16     thoughtworks{
17       matching: "thoughtworks.*";
18       selenium{
19         matching: "selenium.*";
20         webdriven{
21           matching: "webdriven.*";
22         }
23       }
24     }
25   }
26 }
```

# Apêndice B

## Gramática DCL 2.0

Este apêndice apresenta a gramática completa da linguagem DCL 2.0 na notação BNF (Backus-Nahur Form) [Sudkamp, 2005]. Na versão BNF utilizada, símbolos terminais são grafados em negrito e entre aspas. Símbolos não-terminais iniciam-se com maiúsculas. Além disso, (A)\* indica zero ou mais repetições de A e (A)? indica que A é opcional.

```
1  DCLModel: architecture ID_DCL "{ "  
2    (AbstractComponent)*  
3    (((ignore STRING) (",")?)* (";"))?  
4  "}" ;  
5  
6  ID_DCL: ('a'..'z' | 'A'..'Z' | '_' | '.' )  
7    ('a'..'z' | 'A'..'Z' | '_' | '-' | '.' | '0'..'9')* ;  
8  
9  AbstractComponent: MetaModule ;  
10  
11 MetaModule: ID_DCL "{ "  
12   (matching STRING ";") ?  
13   (description STRING ";") ?  
14   (AbstractComponent)*  
15   (restrictions "{ " (Restriction)* "}") ?  
16 "}" ;  
17  
18 Restriction: restriction (GroupClause)? (PermissionClause)?  
19             RelationType (GroupClause)?  
20             ((ComponentsBinRestrictionDeclaration)(",")?)*  
21             (message STRING)? ";";  
22  
23 AbstractNameConvention: STRING ;  
24  
25 QualifiedName: ID_DCL ('.' ID_DCL)* ;  
26  
27 ComponentsBinRestrictionDeclaration: ([AbstractComponent | QualifiedName])  
28                                     (Wildcard) ? ;
```

```
29 GroupClause: only | only this ;
30
31 PermissionClause: must | can | cannot ;
32
33 RelationType: access | declare | handle |
34 extend | implement | create | throw | use annotation | depend | requires ;
35
36 WildCard: " .* " | " .* * " ;
```