2
Conceitos Básicos

Este capítulo aborda conceitualmente os tópicos mais relevantes do trabalho, a saber: rastreabilidade de requisitos, procedures Oracle, o conceito de processo e de definição de estratégias.

2.1.
Rastreabilidade de Requisitos

O termo “rastreabilidade”, é formado pelo adjetivo “rastreável” mais o sufixo “-dade”. O adjetivo de dois gêneros “rastreável” significa “o que se pode rastrear” Sendo assim, rastreabilidade segundo [13] é a qualidade do que é rastreável, ou seja, do que se pode rastrear ou “seguir o rastro de”, “seguir a pista de”. O termo “rastro” é o mesmo que “rasto” e significa sinal, pista, vestígios, indícios.

A conceituação do termo “rastreabilidade de requisitos”, considerando uma pesquisa sobre o estado da arte, foi amplamente estudada por Raja e Kamran [8]. De acordo com Raja e Kamran, podemos afirmar que Gotel & Finkelstein [17], Hamilton & Beeby [18] e o padrão ANSI/IEEE 830-1998 fornecem definições abrangentes do estado da arte da rastreabilidade de requisitos e que a definição de Gotel & Finkelstein [17] é a mais amplamente utilizada na literatura. Assim, de acordo com o ponto de vista de Gotel & Finkelstein, a rastreabilidade de requisitos é a “capacidade para descrever e acompanhar a vida dos requisitos, tanto no sentido para frente (e.g., requisitos para código) quanto para trás (e.g., código voltando para requisitos), ao longo do ciclo de desenvolvimento, implementação e refinamento”.

Segundo Aurum e Wohlin [19], a rastreabilidade pode ser dividida em rastreabilidade horizontal e rastreabilidade vertical, que significa se os objetos de informação relacionados pertencem ao mesmo tipo ou não. A rastreabilidade horizontal lida com versões ou variantes do mesmo tipo de informação, por exemplo, entre requisitos ou entre os componentes do sistema. A rastreabilidade vertical está preocupada com informações de rastreio entre as fases anteriores e posteriores do processo de desenvolvimento, entre objetos de informação de diferentes tipos. Um exemplo seria relacionar um requisito ao projeto feito com

base no requisito, e um seguida relacioná-los a um componente do sistema que implementa o requisito

De acordo com Gotel e Finkelstein [17] existem também dois aspectos relacionados à rastreabilidade de requisitos: rastreabilidade pré-ER (pré-especificação de requisitos) e rastreabilidade pós-ER (pós-especificação de requisitos). A rastreabilidade pré-ER está preocupada com os aspectos da vida dos requisitos até o ponto onde eles não são incluídos na especificação de requisitos [17] [19] [20]. Na rastreabilidade pré-ER os requisitos estão relacionados com a sua origem e a outros requisitos. A origem dos requisitos inclui, por exemplo, as partes interessadas (Stackholders), regras de negócios ou documentos anteriores, como mostrado na Figura 2. Como é possível observar, o requisito R2.1 está relacionado com os requisitos R1 e R2. As relações entre os requisitos R1 e R2, são exemplos de rastreabilidade horizontal.

Já a rastreabilidade pós-ER está preocupada com os aspectos da vida do requisito a partir do ponto em que ele está incluído na especificação de requisitos e daí para frente [19]. Ela pretende garantir que todos os requisitos sejam atendidos pelo projeto e pela implementação do sistema. A rastreabilidade pós-ER está relacionada com casos de testes que garantem que os componentes satisfazem esses requisitos. Além desses dois aspectos da rastreabilidade de requisitos, a rastreabilidade também é classificada nos quatro tipos a seguir [52].

1. Rastreabilidade para trás a partir dos requisitos (“Backward-from traceability”): Liga os requisitos às suas fontes que estão em outros documentos ou pessoas.
2. Rastreabilidade para frente a partir dos requisitos (“Forward-from traceability”): Liga os requisitos para a concepção e implementação de componentes.
3. Rastreabilidade para trás até os requisitos (“Backward-to traceability”): Liga o projeto e os componentes de implementação de volta aos requisitos.
4. Rastreabilidade para frente até os requisitos (“Forward-to traceability”): Liga outros documentos com os requisitos relevantes. Esses outros documentos podem ser, por exemplo, manuais de operação, descrevendo a funcionalidade do sistema.

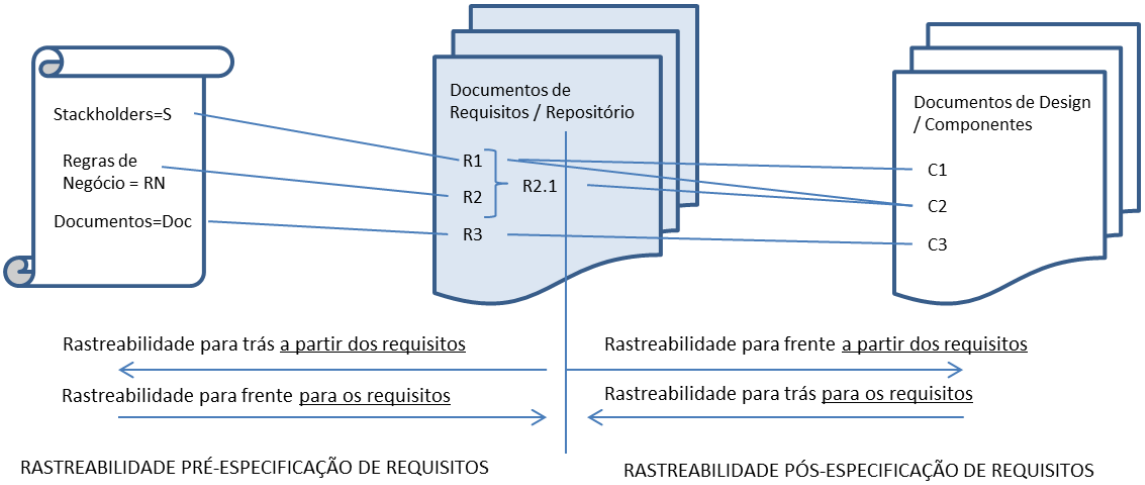

Figura 2 – Interdependência entre os requisitos e outros artefatos

Existe um consenso atualmente de que há uma interdependência entre os requisitos entre si e outros artefatos [19]. Na Figura 2 os requisitos R1, R2 e R2.1 são interdependentes. Desta forma, segundo Raja e Karam [8] regras bem definidas de rastreabilidade contribuem para rastrear o relacionamento entre os requisitos e outros artefatos.

A Figura 3 apresenta outra forma de visualização dos conceitos de rastreabilidade pré-ER e pós-ER [21].

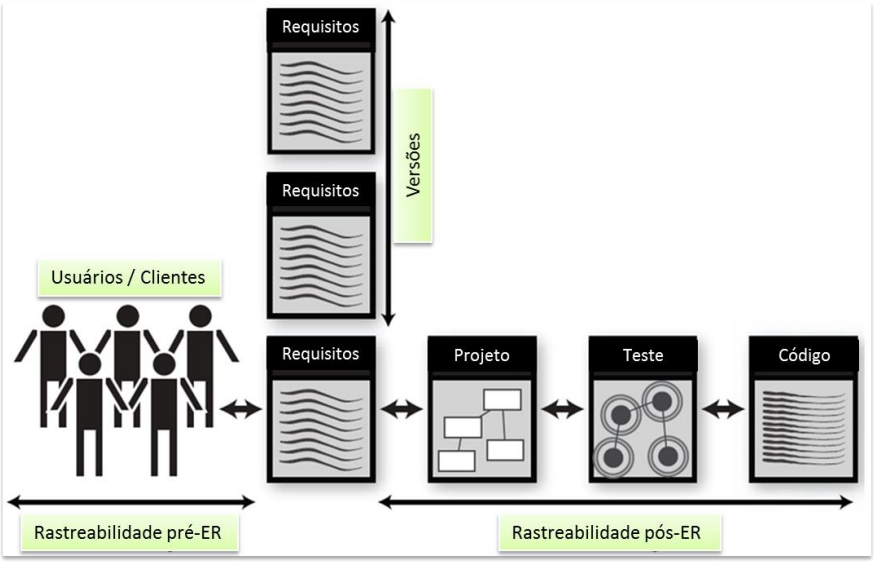

Figura 3 – Tipos de rastreabilidade (Adaptada de [21])

Raja e Kamran [8] também concluíram em suas pesquisas, que nem todos os autores, que de uma forma ou de outra procuram definir a rastreabilidade de

requisitos, abordam os dois aspectos citados por Gotel e Finkelstein [17] em seus textos. É possível tipificar a abordagem utilizada para definir a rastreabilidade de requisitos por cada fonte de pesquisa de duas formas: com cobertura completa e com cobertura incompleta [8]. A cobertura completa é utilizada quando os dois aspectos são abordados, ou seja, tanto a rastreabilidade pré-ER quanto a rastreabilidade pós-ER. A cobertura incompleta é utilizada quando somente um dos aspectos é abordado. Os artigos contendo as definições de rastreabilidade de requisitos utilizados na pesquisa de Raja e Kamran são apresentados na Tabela 1 [8].

Tabela 1 – Rastreabilidade de requisitos: definições e referências (Adaptada de [8])

Referências	Definição	Escopo da Definição	Cobertura
[1], [32] até [47]	Gotel & Finkelstein	Rastreabilidade Pré-ER e Rastreabilidade Pós-ER	Completa
[48], [49]	Ramesh & Jarke	Rastreabilidade Pós-ER	Parcial
[43], [47], [48], [50]	ANSI / IEEE std. 830-1998	Rastreabilidade Pré-ER e Rastreabilidade Pós-ER	Completa
[51]	DOD	Rastreabilidade Pré-ER e Rastreabilidade Pós-ER	Completa
[1]	Hamiltion & Beeby	Rastreabilidade Pré-ER e Rastreabilidade Pós-ER	Completa
[1]	Edward & Howell	Rastreabilidade Pós-ER	Parcial
[48]	Spanoudakis	Rastreabilidade Pós-ER	Parcial
[48]	Murray	Rastreabilidade Pós-ER	Parcial
[48]	Ramesh	Rastreabilidade Pós-ER	Parcial

Um trabalho também relevante que procurou abranger a conceituação de rastreabilidade de requisitos foi realizado por Sayão e Leite [4].

Vale destacar, que desde 2003 um prêmio tem sido concedido anualmente na IEEE Conferência Internacional de Engenharia de Requisitos para o artigo mais influente apresentado na conferência nos últimos 10 anos (onde são apresentados geralmente trabalhos importantes relacionados com o assunto rastreabilidade de requisitos). Em 2013, a conferência comemorou 21 anos, e os participantes aproveitaram este fato como uma oportunidade para refletir sobre os trabalhos mais influentes até hoje [5]. Duas sessões da conferência de 2013 destacaram os últimos trabalhos vencedores do prêmio, e forneceram aos autores a oportunidade de descrever a trajetória de seus trabalhos ao longo dos

dez anos (que os fizeram receber o prêmio), e de discutir seu impacto desde então. Em relação à rastreabilidade de requisitos, oito artigos foram apresentados: 3 (três) relacionados com a prática da rastreabilidade [11] [22] [23], 3 (três) relacionados com a rastreabilidade automática [24] [25] [26], 1(um) abordando uma compilação de todas as pesquisas sobre rastreabilidade nas conferências anteriores de engenharia de requisitos (RE's) [2], incluindo a conferência de 2013, e um último artigo sobre rastreabilidade que foi apresentado quando foram abordados os artigos mais influentes de todas as conferências já realizadas [17]. Podemos com isso enfatizar a importância da área de rastreabilidade de requisitos nas últimas duas décadas até os dias atuais.

2.2. Oracle PL/SQL e Procedures Oracle

A linguagem de consulta estruturada ou SQL (Structured Query Language) "é a linguagem de pesquisa declarativa padrão para banco de dados relacional (base de dados relacional). Muitas das características originais do SQL foram inspiradas na álgebra relacional" [53] [54].

O SQL foi desenvolvido originalmente no início dos anos 70 nos laboratórios da IBM em San Jose, dentro do projeto System R, que tinha por objetivo demonstrar a viabilidade da implementação do modelo relacional proposto por E. F. Codd. O nome original da linguagem era SEQUEL, acrônimo para "Structured English Query Language" (Linguagem de Consulta Estruturada, em Inglês), vindo daí o fato de, até hoje, a sigla, em inglês, ser comumente pronunciada "síquel" ao invés de "és-kiú-él", letra a letra. No entanto, em português, a pronúncia mais corrente é a letra a letra: "ésse-quê-éle" [53] [54].

Embora o SQL tenha sido originalmente criado pela IBM, rapidamente surgiram vários "dialetos" desenvolvidos por outros produtores. Essa expansão levou à necessidade de ser criado e adaptado um padrão para a linguagem. Esta tarefa foi realizada pela American National Standards Institute (ANSI) em 1986 e ISO em 1987 [53] [54].

O SQL foi revisto em 1992 e a esta versão foi dado o nome de SQL-92. Foi revisto novamente em 1999 e 2003 para se tornar SQL:1999 (SQL3) e SQL:2003, respectivamente. O SQL:1999 usa expressões regulares de emparelhamento, queries recursivas e gatilhos ("triggers"). O SQL:2003 introduz características relacionadas ao XML [53] [54].

O PL/SQL (acrônimo para a expressão inglesa “Procedural Language/Structured Query Language”) “é uma extensão da linguagem padrão SQL para o SGBD Oracle da Oracle Corporation. É uma linguagem procedural da Oracle que estende a linguagem SQL” [55] [56]. O Oracle é uma tecnologia de banco de dados relacional.

É a linguagem básica para criar programas complexos e poderosos, não só no banco de dados, mas também em diversas ferramentas Oracle [55] [56].

Em 1991 o PL/SQL 1.0 foi lançado com o Oracle versão 6.0. Ele era muito limitado nas suas capacidades. Já a versão 2.0 era uma atualização maior, que suportava “stored packages”, procedures, funções, tabelas PL/SQL, registros definidos pelo programador e “package extensions”. Esta versão foi lançada com o Oracle Versão 7.0 [55] [56].

O PL/SQL versão 2.1 foi liberado com a versão 7.1 do Oracle. Isto permitiu o uso de “stored functions” dentro de instruções SQL e a criação de SQL dinâmico pelo uso do pacote DBMS_SQL. Foi também possível executar instruções de Linguagens de Definição de Dados de programas PL/SQL [55] [56] [57].

A versão 2.2 PL/SQL foi lançada com a versão 7.2 do Oracle. Ele implementava uma proteção do código para programas PL/SQL e também o agendamento de trabalhos do banco de dados com o pacote DBMS_JOB [55] [56] [57].

A versão 2.3 do PL/SQL foi lançada com a versão 7.3 do Oracle. Esta versão aumentou as capacidades das tabelas PL/SQL e adicionou funcionalidades de E/S de arquivos [55] [56] [57].

A versão 2.4 do PL/SQL foi liberada com a versão 8.0 do Oracle. Esta versão suporta os melhoramentos do Oracle 8.0, incluindo Large Objects, projeto orientado a objetos, tabelas aninhadas e Oracle advanced queuing [55] [56] [57].

A partir da versão 8.0 do Oracle, as versões do PL/SQL passaram a ser sincronizadas com a versão do banco de dados [55] [56] [57].

O Oracle e o PL/SQL podem ser categorizados nos seguintes tópicos [56]:

Tabela 2 **–** Categorias do Oracle PL/SQL

Funções Oracle/PLSQL	
“Functions – Alphabetical”	Funções Oracle / PLSQL listados em ordem alfabética
“Functions – Category”	Funções Oracle / PLSQL listados por categoria

Chaves Oracle, Restrições e Índices

“Primary Keys”	Criar, alterar, excluir, habilitar e desabilitar as chaves primárias.
“Foreign Keys”	Criar, excluir, habilitar e desabilitar as chaves estrangeiras.
“Unique Constraints”	Criar, alterar, excluir, habilitar e desabilitar as restrições exclusivas.
“Check Constraints”	Criar, alterar, gota, habilitar e desabilitar restrições de verificação.
“Indexes”	Criar, renomear e eliminar índices.

Privilégios Oracle, sinônimos, papéis e senhas	
“Grant/Revoke Privileges”	Conceder ou revogar privilégios
“Synonyms (create, drop)”	Criar, substituir e soltar sinônimos.
“Roles”	Conjunto de privilégios
“Change Password”	Alterar a senha de um usuário

Administração do Banco de Dados Oracle	
“Tablespaces”	Espaço alocado onde objetos de esquema são criados
“Users”	Contas de banco de dados
“Schemas”	Grupo de objetos lógicos com esquemas

Programação Oracle	
“Functions”	Criar e excluir funções
“Procedures”	Criar e eliminar procedimentos
“Triggers”	Criar, soltar, ativar e desativar as triggers.
“Cursors”	Criar, soltar, ativar e desativar os cursores.
“Literals (Constants)”	Texto, inteiros e números literais.
“Declaring Variables”	Declarar variáveis e constantes
“Exception Handling”	Manipular exceção no código
“Loops and Conditional Statements”	Para “Loop”, “While”, “IF-THEN-ELSE” e outros
“Sequences (Auto number)”	Criar e excluir “sequences”
“Transactions”	“Commits” e “rollbacks”
“Comments within SQL”	Como criar comentários dentro de sua instrução SQL

Mensagens de erro da Oracle	
“Oracle Error Messages”	Explicações de mensagens de erro do Oracle e resoluções

Tabelas do Sistema Oracle	
“Oracle System Tables”	Tabelas e descrições do sistema da Oracle

Tipos de Consulta Oracle	
“SELECT Statement”	Recuperar registros de uma tabela
“INSERT Statement”	Inserir registros em uma tabela
“UPDATE Statement”	Atualizar registros em uma tabela
“DELETE Statement”	Excluir registros de uma tabela
“UNION Operator”	Combinar dois conjuntos de resultados (remove duplicatas)
“UNION ALL Operator”	Combinar dois conjuntos de resultados (inclui duplicatas)

“INTERSECT Operator”	Intersecção de dois conjuntos de resultados
“MINUS Operator”	Conjunto de resultados de um, menos o conjunto de resultados do outro.
“Subqueries”	Uma consulta em uma consulta

Oracle JOINS (Junções Oracle)	
“JOIN TABLES”	Junção Interior e Exterior (“Inner and Outer joins”)

Oracle Aliases	
“ALIASES”	Crie um nome temporário para uma coluna ou tabela

Oracle Clauses	
“DISTINCT Clause”	Recuperar registros exclusivos
“WHERE Clause”	Pesquisar resultados
“ORDER BY Clause”	Ordenar resultados da consulta
“GROUP BY Clause”	Grupo por uma ou mais colunas
“HAVING Clause”	Restringir os grupos de linhas retornadas

Funções do Oracle SQL	
“COUNT Function”	Retorna o número de linhas em uma consulta
“SUM Function”	Retorna a soma de uma expressão
“MIN Function”	Retorne o min de uma expressão
“MAX Function”	Retorne o máximo de uma expressão

Condições do Oracle	
“AND Condition”	Duas ou mais condições a serem cumpridas
“OR Condition”	Qualquer uma das condições é atendida
“AND OR”	Combina as condições “AND” e “OR”
“LIKE Condition”	Usa correspondência simples de padrão (curingas – “wildcards”) de uma cláusula “WHERE”
“REGEXP_LIKE Condition”	Usa expressão regular correspondente em uma cláusula “WHERE”
“IN Condition”	Alternativa para múltiplas condições “OR”
“NOT Condition”	Negar uma condição
“IS NULL Condition”	Teste para um valor “NULL”
“IS NOT NULL Condition”	Teste para um valor não nulo
“BETWEEN Condition”	Recuperar dentro de um intervalo (inclusive)
“EXISTS Condition”	Condição é satisfeita se a subconsulta retornar pelo menos uma linha

Tabelas Oracle e Visões	
“CREATE TABLE”	Criar uma tabela
“CREATE TABLE AS”	Criar uma tabela de outra definição de tabela e dados
“ALTER TABLE”	Adicionar, alterar ou excluir colunas em uma tabela; renomear uma tabela.
“DROP TABLE”	Excluir uma tabela
“GLOBAL TEMP Tables”	Tabelas que são distintas dentro da sessão SQL
“LOCAL TEMP	Tabelas que são distintas dentro dos módulos e de

Tables”	programas embutidos SQL
“VIEW”	Tabelas virtuais (pontos de vista de outras tabelas)

Tipos de dados da Oracle	
“Data Types”	Tipos de dados em Oracle/PLSQL

Perguntas e Respostas Oracle	
“Question & Answer”	Várias perguntas e respostas

Neste trabalho, é enfatizada a rastreabilidade pós-especificação de requisitos, mais especificamente em relação a requisitos de projetos codificados em PL/SQL. Em relação ao experimento prático considerado nos próximos capítulos, é realizada a instanciação do macroprocesso no ambiente Oracle, restrita ao código armazenado em “procedures” e “functions”. Desta forma, a conceituação mais detalhada da programação Oracle será restrita, às procedures e functions Oracle.

No Oracle PL/SQL, uma procedure (ou procedimento, conforme indicado na Tabela 2, no item “Programação Oracle”) é considerada um subprograma PL/SQL que pode, opcionalmente, aceitar parâmetros e pode ou não retornar um valor. Sua principal função é a de incorporar um processo da lógica de negócios e executar a manipulação de dados com a ajuda dos dados fornecidos. Ela pode retornar valores para o ambiente de chamada somente através de parâmetros “OUT”. Uma procedure pode conter uma ou mais declarações de retorno (“RETURN”), que, logicamente, devolve o controle para o ambiente de chamada, ignorando assim a execução posterior do corpo do procedimento.

O usuário deve ter o privilégio do tipo “CREATE [ANY] PROCEDURE” para criar uma “procedure”, também conhecida como “storage procedure” (procedimento armazenado). “Procedures” promovem a restauração da lógica, a sua reutilização e facilidade de manutenção.

A seguir é apresentada a sintaxe para criação de uma “procedure” [58]:

```
CREATE [OR REPLACE] PROCEDURE [NAME]
[PARAMETERS]
[IS | AS]
BEGIN
END;
```

Figura 4 – Exemplo de sintaxe para criação de “procedure”

Na sintaxe apresentada acima, o termo “REPLACE” é utilizado para preservar os privilégios associados à “procedure”.

Exemplo de uso de “procedure”:

Para exemplificar o uso de uma “procedure”, a “procedure” P_TEST_PROC abaixo (Figura 5), calcula a soma de dois números e retorna o resultado para o “host” ou para o ambiente de chamada [58].

```
CREATE OR REPLACE PROCEDURE P_TEST_PROC
(P_ARG1 NUMBER, P_ARG2 NUMBER, P_RESULT OUT
NUMBER)
IS
BEGIN
  P_RESULT := P_ARG1 + P_ARG2;
END;
```

Figura 5 – Exemplo de utilização de “procedure”

Uma “function” (ou função) no Oracle PL/SQL, é um subprograma PL/SQL que possui um nome. A “function” sempre retorna um único valor sobre a sua chamada. Ela funciona de forma semelhante aos procedimentos armazenados (“storage procedures”), com pequenas diferenças sintáticas e de objetivos. A finalidade principal da “function” é fazer um cálculo com base em um determinado conjunto de condições lógicas. Ela deve retornar um valor único para o host ou para o ambiente de chamada. Da mesma forma que as “procedures”, as “functions” contém um cabeçalho e seções executáveis, com uma cláusula lógica “RETURN” [59].

Exemplo de sintaxe:

```
CREATE [OR REPLACE] FUNCTION [NAME] [PARAMETER
LIST] RETURN [DATA TYPE]
[IS | AS]
...
BEGIN
        STATEMENT 1;
        STATEMENT 2;
...
RETURN [VALUE]
        STATEMENT N;
END;
```

Figura 6 – Exemplo de sintaxe para criação de “functions”

As “functions” podem ser armazenadas na base de dados em si, e também são conhecidas como “storages functions” (funções armazenadas). Elas permitem a reutilização da lógica, a flexibilidade de código e modularidade. Ao contrário das “procedures”, as “functions” também podem ser usadas em instruções SQL, se obedecem a regras específicas para evitar mudanças na base de dados [59].

```
CREATE OR REPLACE FUNCTION F_AVG_SALARY
RETURN NUMBER IS
L_AVG NUMBER;
BEGIN
SELECT        AVG(SALARY)
        INTO L_MAX
        FROM EMP;
RETURN L_AVG;
END;
```

Figura 7 – Exemplo de utilização de “functions”

Outra estrutura importante na codificação de “procedures” e “functions” são os “packages”. No Oracle PL/SQL, um “package” (pacote) é um grupo de construções programáticas combinadas ("empacotada") em uma única unidade. Os elementos de PL/SQL que podem ser incluídos em um pacote são “cursors” (cursores), “variables” (variáveis), “exceptions” (exceções), “PL/SQL records” (registros PL/SQL) e “collection types” (tipos de coleção), “procedures”, e “functions”.

Um “package” existe em duas partes: especificação e corpo.

A especificação do pacote contém as construções públicas e atua como a interface para o pacote. Note-se que ele contém apenas protótipos sem qualquer código lógico.

O corpo do pacote contém a definição dos construtos prototipados na especificação. Também pode conter as unidades privadas de programas ou localmente definidas, que podem ser utilizadas dentro do âmbito de apenas o corpo de “package”.

“Packages” lidam com encapsulamento, ocultamento de dados, sobrecarga de subprograma e modularização. Em termos de desempenho, os “packages” potencialmente têm efeitos tanto positivos como negativos. Na primeira chamada ao “package”, o “package” inteiro é carregado na área de Sistema Global (SGA),

o que pode degradar o desempenho geral. Depois que o “package” é carregado, no entanto, se qualquer subprograma no “package” precisa ser invocado, ele não requer acesso ao disco. Subprogramas empacotados também interrompem as dependências em cascata, o que evita a compilações desnecessárias [60].

```
CREATE [OR REPLACE] PACKAGE [NAME] IS
[PRAGMA]
[PUBLIC CONSTRUCTS]
[SUBPROGRAM1 PROTOTYPE]
[SUBPROGRAM2 PROTOTYPE]
...
[SUBPROGRAM N PROTOTYPE]
END;

CREATE [OR REPLACE] PACKAGE BODY [NAME] IS
[LOCAL CONSTRUCTS]
[CONSTRUCT1 DEFINITION]
[CONSTRUCT2 DEFINITION]
...
[CONSTRUCT N DEFINITION]
[BEGIN...END]
END;
```

Figura 8 – Exemplo de sintaxe para manipulação de “packages”

```
CREATE OR REPLACE PACKAGE PKG_UTIL IS

PROCEDURE P_ENAME(P_VAR VARCHAR2);

END;

/

“Package created”.

CREATE OR REPLACE PACKAGE BODY PKG_UTIL IS

PROCEDURE P_ENAME(P_VAR VARCHAR2) IS

BEGIN
```

```
DBMS_OUTPUT.PUT_LINE(P_VAR);

END;
END PKG_UTIL;

/

“Package body created”.
```

Figura 9 – Exemplo de utilização de “packages”

2.3.
Processo e Estratégia

O termo “estratégia” (do grego "στρατηγία" - stratēgia, "arte de líder da tropa”) [61] “é um planejamento de alto nível para atingir um ou mais objetivos em condições de incerteza” [61] (tradução nossa).

> Uma estratégia é importante, pois os recursos disponíveis para atingir esses objetivos são geralmente limitados.
>
> Segundo Henry Mintzberg da Universidade Pública McGill de Montreal, uma estratégia é como "um padrão em um fluxo de decisões" para contrastar com uma visão de estratégia como planejamento [62]. O escritor inglês Max McKeown argumenta que "estratégia é sobre a construção do futuro" e é a tentativa humana para chegar a "fins desejáveis com os meios disponíveis"[63]. Dr. Vladimir Kvint define estratégia como "um sistema de busca, formulação e desenvolvimento de uma doutrina que vai garantir o sucesso em longo prazo se seguida fielmente" [64] (tradução nossa).

Processo (do latim procedere) é um verbo que indica a ação de avançar, ir para frente (pro+cedere) e é um conjunto sequencial e particular de ações com objetivo comum. Pode ter os mais variados propósitos: criar, inventar, projetar, transformar, produzir, controlar, manter e usar produtos ou sistemas [65].

Na engenharia de software, a meta é criar um produto ou aprimorar um já existente; na engenharia de processos, a meta é desenvolver ou aprimorar um modelo de processo; corresponde a um caso de uso de negócios na engenharia de negócios [66].

No próximo capítulo é descrito o processo de gestão de estratégia de rastreabilidade.