6
Conclusão

A manutenção de um sistema legado pode ser facilitada quando se aplica algumas práticas de engenharia reversa. Uma delas é a recuperação do modelo de sua estrutura. Visando este nicho, este trabalho produziu como contribuição uma metaferramenta capaz de exibir diagramas comportamentais ou qualquer outro que venha a ser configurado nos seus arquivos de metadados. Para que isso se tornasse uma realidade, era preciso um *framework* gráfico também orientado a metadados. Devido às dificuldades de encontrar um artefato que pudesse ser usado com sucesso, foi desenvolvido um produto interno para este objetivo.

A ferramenta foi aplicada com êxito em um estudo experimental que solucionou um problema encontrado por um projeto real, sem fins acadêmicos. Para atendê-lo, este trabalho teve não apenas que desenvolver a ferramenta citada, mas padrões para o reconhecimento de *procedures*, tabelas e seus relacionamentos no código fonte, já que o domínio da aplicação era a linguagem *PL/SQL*. No final de todo o trabalho, a ferramenta final customizada e aplicada na avaliação experimental foi aprovada com louvor, e trouxe como contribuição não só a recuperação do modelo da estrutura de um sistema legado, mas também a capacidade de auxiliar o desenvolvedor na tarefa de análise de impacto de uma mudança.

Este trabalho é proveitoso para as pessoas que possuem algum tipo de sistema legado e que encontram dificuldades em achar ferramentas para a prática da engenharia reversa, para aqueles que precisam de um *framework* de geração de diagramas orientado a metadados, ou para os que simplesmente querem olhar o estudo experimental e aprender com a experiência adquirida.

6.1.
Trabalhos Futuros

No capítulo 4, foi apresentado o *framework* gráfico desenvolvido para atender a ferramenta *Dependency Viewer* (Capítulo 3). Para sua utilização, é necessário configurar metadados que guardam todas as regras comportamentais

dos objetos e de como eles interagem entre si. Essas informações seriam suficientes para a geração de um editor de diagramas. Ele iria de encontro com a finalidade de prover uma *API* gráfica completa para projetos na área de engenharia reversa. Nem sempre o diagrama gerado é o final, apesar de estar correto. É comum haver alterações para, por exemplo, torná-lo mais legível. Além disso, é preciso disponibilizar funções que permitam ao usuário criar anotações e comentários no diagrama gerado. É um trabalho futuro o desenvolvimento deste editor.

É possível, similar ao editor, desenvolver e aprimorar o *framework* para que, no final, seja viável a geração de um esqueleto de código útil. Para isso, haveria de ter evoluções nos meta-dados existentes, principalmente no que diz respeito à sua ligação com o modelo de dados, pois hoje um objeto gráfico se relaciona com um *Object* Java. Esta implementação atendeu as expectativas até o momento, porém ela deve evoluir para virar uma composição de objetos, uma vez que podem possuir uma variedade de atributos não conhecidos a priori, além de poder conter instâncias em uma variedade de diagramas.

Com relação à ferramenta *Dependency Viewer*, foi possível atingir o objetivo proposto para a avaliação experimental. Porém, é necessário aplicá-la em outros casos. A ferramenta é customizada e populada através de meta-dados, mas é preciso por a prova esta customização com mais exemplos e avaliações experimentais.

A avaliação experimental relatada no capítulo 5 não foi apenas um caso acadêmico. As *procedures* homônimas, hoje, são tratadas pela quantidade de seus parâmetros. Porém, existem casos de *procedures* com o mesmo nome e o mesmo número de atributos de entrada, que apesar de serem diferentes, são tratadas como iguais pelo sistema. É muito pequena a quantidade de vezes que isso ocorre, sendo na prática insignificante. Apesar disso, se fosse aplicado em outros sistemas escritos em *PL/SQL*, poderia fazer diferença no resultado apresentado. Um trabalho futuro é tratar desses casos da forma correta, pelo tipo de parâmetro, e não pela quantidade.

Similar ao problema acima, alguns padrões apresentados na avaliação experimental foram desenvolvidos em cima de regras e estilo próprio do projeto específico, e não da linguagem PL/SQL. É desejável o desenvolvimento de outros padrões que possam ser aplicados de forma genérica.