5
Conclusão

Esta tese mostrou, pela primeira vez, um novo modelo de bloqueio multigranular para o modelo de dados RDF. Foi apresentado como os problemas comuns de concorrência (*lost updates, dirty reads, non repeatable read, phantom reads*) podem ser evitados, utilizando o protocolo multigranular e os novos tipos de bloqueio propostos.

Além disso, foi feita uma avaliação de desempenho da proposta, por meio de várias simulações, em cenários distintos. Os resultados comprovaram a eficácia das ideias.

Para concluir esta tese, primeiro uma visão geral das contribuições dos capítulos anteriores será brevemente exibida. Em seguida, alguns trabalhos futuros serão enumerados.

5.1.
Contribuições

A principal contribuição deste trabalho é modelo de bloqueio multigranular, com seus grânulos e tipos de bloqueio, que permite transações bloquearem, de forma pessimista, itens de dados RDF, tirando proveito da estrutura de grafo do RDF.

Todo modelo foi implementado por meio de um *Lock Manager* completo e *thread-safe* na linguagem Ruby. O *Lock Manager* encontra-se totalmente pronto para produção, e, assim como o protocolo proposto, foi testado por meio de várias simulações realistas.

Para facilitar o uso do modelo de bloqueio, também foi criada uma pequena ontologia para ser utilizada juntamente com SPARQL CONSTRUCT, permitindo que a aplicação consiga determinar estritamente os itens de dado que, de fato, precisam ser bloqueados, evitando, portanto, bloquear partes do espaço de dados desnecessariamente.

Além disso, também foi construída e testada uma DSL interna em Ruby para especificação de *workflow* de transações Web. E juntamente com esta, acabou-se por definir um metamodelo e uma arquitetura para transações Web,

tendo como ponto central a noção de transação reificada em um objeto de primeira ordem. Reificação de transações facilita, sobremaneira, a criação de um código mais organizado, pois todas as informações (objetos) manipulados pela transação passam a ter um lugar pré-definido para residirem: a própria transação.

Uma outra contribuição muito importante foi a apresentação de um roteiro a ser seguido para se utilizar a *abordagem* otimista juntamente com o protocolo pessimista proposto. Em especial, foi apresentado como seria o equivalente dos grânulos e dos novos tipos de bloqueio, de remoção e inserção, numa visão otimista. Foi identificada também a oportunidade de se reusar a proposta *CBD - Concise Bounded Description* para se ter algo similar ao grânulo "*Resource*" no contexto otimista.

Ao longo deste trabalho foram publicados dois artigos. São eles:

- JACYNTHO, M. D. A.; Schwabe, D. *Models and Meta Models for Transactions in Web Applications*. 6th Model-Driven Web Engineering Workshop (MDWE 2010), 2010, Viena, Áustria. ICWE'10 Proceedings of the 10th international conference on Current trends in Web Engineering - LNCS, 2010. v. 6385. p. 37-48.
- JACYNTHO, M. D. A.; Schwabe, D. *Modelos e Meta-Modelos para Transações em Aplicações Web*. XVI Simpósio Brasileiro de Sistemas Multimídia e Web - WebMedia, 2010, Belo Horizonte - MG, Brasil. Anais do Simpósio Brasileiro de Sistemas Multimídia e Web. Porto Alegre : Sociedade Brasileira de Computação, 2010.

5.2. Trabalhos Futuros

Como primeiro trabalho futuro, podemos apontar a incorporação de um modelo de transações, utilizando a DSL interna para transações Web, ao ambiente de autoria Synth, criado em [Bomfim, 2011].

Uma vez incorporada a DSL ao Synth, poder-se-ia dar um passo adiante e criar templates de transações Web comumente utilizadas, com o *workflow* já pré-definido usando a DSL. Desta forma, o projetista apenas customizaria o template, onde fosse necessário, e se necessário.

Outro trabalho, ainda relacionado ao Synth, seria utilizar o *Lock Manager* no ambiente de execução do Synth. Vale lembrar que o Synth é um ambiente de autoria e execução de aplicações na Web semântica.

Uma vez incorporado ao Synth, seria interessante desenvolver vários protótipos usando o *Lock Manager* e medir os resultados.

Outro trabalho, complementar a este, é implementar um *UnitOfWork* RDF para garantir Atomicidade em nível de transações Web RDF. A ideia é simples: o UnitOfWork manteria o registro de todos os *statements* RDF a serem removidos e inseridos no banco de dados. Ao final da transação Web RDF, o UnitOfWork seria responsável por persistir, de uma só vez, todas as remoções e, em seguida, as inserções. Teria que se pensar em uma forma elegante das transações Web registrarem no UnitOfWork, se possível de forma transparente, sua modificações.

O *Lock Manager* poderia ser aprimorado de forma a decidir automaticamente qual o grânulo utilizar dependendo dos requisitos das transações e da carga atual do sistema. Seria o conceito de bloqueio dinâmico multigranular.

Uma questão que precisa ser melhor investigada é a integração do *Lock Manager* com raciocinadores e inferências. Por hora, assumimos a materialização das triplas inferidas. Neste caso, o raciocinador seria, periodicamente, disparado por uma transação que estabeleceria um bloqueio de escrita para inserção (*iW*) no grânulo "*Graph*". Esta transação teria como responsabilidade única o acionamento do racionador para inserir as novas triplas inferidas no banco de dados.

Outro trabalho de grande valia seria analisar a integração do modelo de bloqueio proposto com a Web de dados, em especial, a nuvem LOD.

Por fim, seria muito interessante, inserir o modelo de bloqueio dentro da arquitetura de um *Triple Store*, ou seja, implementar o *Lock Manager* como parte integrante de um banco de dados RDF.