5
Conclusão

Neste capítulo serão citadas as contribuições deste trabalho. Em seguida, será mostrada uma análise comparativa entre os trabalhos relacionados, mostrando as vantagens e limitações de cada solução. Finalmente serão listadas possíveis continuações da arquitetura e implementação em trabalhos futuros.

5.1. Contribuições

Esta arquitetura se mostrou robusta e escalável, visto que os agentes podem estar distribuídos em diversos servidores, evitando que um único servidor fique onerado com a responsabilidade de replicar todas as bases de dados.

Além disso, a abordagem de divisão de sistemas complexos através de múltiplos agentes utilizando o ambiente *JEE* se mostra útil para diversos sistemas além de replicadores. Este trabalho pode ser a base para sistemas de coordenação de *workflow*, com a inclusão de agentes de *software* se relacionando diretamente com seres humanos.

Esta arquitetura também se mostrou possível para pequenos ambientes de replicação como, por exemplo, uma base homogênea na origem e no destino de um fluxo. Porém, o *overhead* introduzido para dar suporte a ambientes mais complexos abre espaço a soluções mais simples e porém menos escaláveis do que a proposta.

5.2. Trabalhos Relacionados

Para desenvolvimento deste trabalho, foram analisadas algumas ferramentas de replicação disponíveis no mercado. Neste capítulo, a ferramenta de replicação do *SQL Server* é analisada comparativamente com o RePAE. Em seguida, também são discutidas as ferramentas *Daffodil Replicator* e *Delta Replicatior*.

5.2.1. SQL Server

O serviço de replicação do *SQL Server* [SQL 2007] fornece vários tipos de replicação:

- **Snapshot**: copia um banco integralmente
- **Merge**: permite trabalho *offline*
- **Transactional**: replica as diferenças periodicamente

Contudo, este serviço é limitado a replicar objetos pré-existentes. Não é possível replicar *Data Definition Language* (DDL). Ou seja, se novas tabelas forem criadas pela aplicação, a replicação deve ser reconfigurada. Além disso, a possibilidade de realizar tratamentos especiais a regras de negócios é limitada. O SQL Server oferece poucos pontos de flexibilidade no caminho dos dados desde a origem até o destino.

A replicação *snapshot* é realizada neste trabalho pelo serviço copiador. Já a replicação Merge é equivalente ao serviço sincronizador.

5.2.2. Daffodil Replicator

Além dos serviços de Snapshot e *Merge*, o *Daffodil* [DAFFODIL, 2007] possui duas possibilidades para replicação transacional:

- **Push**: serviço onde a base mestre informa as alterações.
- **Pull**: serviço onde a base escrava solicita as alterações.

No Daffodil as alterações são persistidas em arquivos de log e transmitidas via tecnologia *RMI*. Já neste trabalho, as alterações são transportadas via tecnologia JMS cuja persistência varia de acordo com servidor JEE podendo ser arquivos binários, documentos xml, tabelas de banco de dados, etc.

O Daffodil já implementa um serviço de réplica bidirecional. Por outro lado, ainda não suporta a replicação de tabelas que possuem colunas auto-incrementadas nem suporta tipos de dados binários como chave primária.

5.2.3. Delta Replicator

O *Delta Replicator* [DELTA, 2007] guarda as alterações em fila de arquivos transmitidas via *sockets*. Sua arquitetura é baseada em um esquema produtor-

consumidor para cada nó de replicação tal qual ao par de agentes exportador e importador deste trabalho.

Contudo, o versionamento é realizado via *timestamps*, o que pode ser problemático em nós que diferem de fuso horário ou formatação de datas. Além disso, com *timestamps* não é possível garantir que a ordem da aplicação das transações no destino ocorrerá exatamente como na origem. Devido a estes problemas, este trabalho optou pelo versionamento numérico incremental usando apenas inteiros.

5.3. Trabalhos Futuros

As possibilidades para trabalhos futuros incluem:

- **Notificações**. Implementar notificadores de maneira que o administrador seja automaticamente solicitado a realizar ajustes manuais no ambiente de replicação.
- **Configuração via interface *web***. Possibilitar que o administrador faça ajustes de ambiente através de uma interface *web*, definindo alertas, parâmetros de ambiente e fluxos de replicação.
- **Replicação bi-direcional**. Possibilitar que as bases sejam tanto sincronizadas em tempo real com fluxo em ambas as direções.
- **Replicação Multi-SGBD**. Possibilitar o serviço replicar bases cujos sistemas de gerenciamento de banco de dados (SGBD) venham de diferentes fornecedores. Por exemplo, replicar uma base em *SQL Server* para uma base em *Oracle*. Atualmente, pode-se utilizar qualquer SGBD, mas o mesmo necessita estar na origem e no destino.
- **Ajustes automáticos**. Definição de novas estratégias de ajustes inteligentes de parâmetros para melhorar a desempenho automaticamente
- **Plugins**. Definir uma arquitetura plugável de forma a facilitar a inclusão de novos tipos de bases de dados, transformando a arquitetura definida e implementada num *framework*. Além disso, pode-se criar um conjunto de *plugins* capazes de satisfazer um grande conjunto de aplicações.

- **Mobilidade dinâmica**. Definição de um caminho virtual de modo que os agentes possam migrar entre ambientes em tempo de execução.
- **Filas paralelas em topologia fechada**. Permitir que um mesmo repositório possa ser atualizado em várias frentes vindas de origens distintas, de forma a minimizar o atraso total entre cópias primárias e secundárias.
- **Integração com sistemas P2P**. Permitir que o replicador interaja com arquiteturas P2P de maneira a intensificar a entrada de arquivos no ambiente de replicação. Por sua vez, os sistemas P2P poderiam se beneficiar da utilização do replicador como sendo um túnel de alta velocidade para pontos que sejam grandes fornecedores de dados.