Faculdade de Engenharia da Universidade do Porto

Integração Aplicacional Sincronização de Dados de Referência no Sector de Retalho

Ricardo António Rocha Espírito Santo Veloso

Versão Final

Relatório de Projecto realizado no Âmbito do Mestrado Integrado em Engenharia Informática

Orientador: João Canas Ferreira (Professor Auxiliar)

Julho de 2008

© Ricardo Veloso, 2008

Integração Aplicacional – Sincronização de Dados de Referência no Sector de Retalho

Ricardo Veloso

Relatório de Projecto realizado no âmbito do Mestrado Integrado em Engenharia Informática

Aprovado em provas públicas pelo Júri:

Presidente: João Sousa Cardoso (Professor Auxiliar Convidado)

_____

Arguente: Isabel Ramos (Professora Auxiliar)
Vogal: João Paulo de Castro Canas Ferreira (Professor Auxiliar)

15 de Julho de 2008

Resumo

Este relatório descreve uma integração aplicacional do sector de ERP de retalho para grandes empresas, tendo como cenário o sistema de um dos primeiros 5 retalhistas mundiais que pretende integrar o ERP Oracle Retail.

Mais precisamente, foi analisado o campo de processos de Pricing promocionais e adicionalmente estudada a performance da solução global de integração presente no mesmo sistema.

Um estudo das propriedades divergentes de dois sistemas promocionais foi elaborado e concluído com a apresentação de métodos de conciliação e aproveitamento das capacidades de ambos, devidamente testadas e implementadas no retalhista em questão.

São analisados e testados os benefícios e limitações do mecanismo de multi-threading e as complexidades relacionais dos dados de transacção de retalho.

Abstract

This report describes an applicational integration effort in the context of ERPs for major corporations, using as setting one of the major 5 retailers in the world's information system, which is intended to integrate the Oracle Retail ERP.

Specifically, Promotional Pricing processes were analyzed, along with the performance of the system's global integration solution for OR, the ORIB.

A study of two different promotional systems was executed, ending with a presentation of synchronization methods that take advantage of both systems' capabilities, tested and implemented in the retailer.

Benefits and limitations of the multi-threading mechanisms available are also presented, along with the complex aspects of the relational data of a retailer's transactions.

Agradecimentos

Em primeiro lugar, gostaria de agradecer o apoio directo na tarefa de introdução à realidade de UK e da Índia proveniente de toda a equipa da Enabler.

Seguidamente gostaria de demonstrar o meu apreço pela orientação e apoio do Prof. João Canas Ferreira na tarefa de concretizar um Projecto de Mestrado num meio empresarial.

A minha presença foi também guiada pelo meu orientador do lado do proponente, Marco Aurélio Rocha, tutor natural.

Numa perspectiva tecnológica tanto como amigável, sem a ajuda do Mário Correia e do Hélder Ribeiro, não teria aprendido tanto, sobre tanto, em tão pouco tempo.

Por fim, a minha capacidade de prevalecer frente ao desafio de trabalhar tão intensamente não estaria lá sem a Sabina e a minha família, que, no tempo em que estive no estrangeiro, continuaram bem perto.

Ricardo Veloso

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Este relatório pretende documentar o Projecto de Mestrado Integrado de Engenharia Informática e Computação (MIEIC) desenvolvido pelo aluno Ricardo António Rocha Espírito Santo Veloso da Faculdade de Engenharia da Universidade do Porto (FEUP) no primeiro semestre do ano curricular de 2008/2009.

O orientador deste projecto é o Professor João Canas Ferreira, o proponente é a empresa da área de Information Technology (IT) Enabler Wipro e o responsável pelo acompanhamento no proponente é o Engenheiro Marco Aurélio Rocha.

1.1 Projecto

Este projecto insere-se na área de Information Technology (IT), mais especificamente de Enterprize Applications Integration (EAI) e insere-se num esforço continuado de melhorar as aplicações de ERP que lideram o mercado. Os desenvolvimentos e o trabalho nesta área podem levar a que, idealmente, não haja desperdício de recursos da parte das empresas para se adaptar aos processos do software que utilizam para efectuar os seus negócios, levando a um mercado mais competitivo, eficiente, com uma gestão inteligente e bem informada.

Foi pretendida a integração do sistema de Enterprize Resourse Planning (ERP) da Oracle para o sector de retalho, i.e. o Oracle Retail (OR), num sistema de informação de um grande retalhista mundial.

No OR, os processos relativos a promoções comerciais do cliente são geridos pela aplicação Oracle Retail Price Management (ORPM) e diferem dos processos da Legacy System (LS) operacional de gestão de loja com a qual o OR tem que integrar.

Dado que este LS é central e distribui a informação promocional por todos os outros LS, alterá-lo teria um custo excessivo, pelo que o ORPM foi adaptado para sincronização de processos.

Também a aplicação de integração do OR, o Oracle Retail Integration Bus (ORIB), teve que ser modificada para acomodar as mudanças na transmissão de informação para os sistemas externos ao OR.

Foi elaborada documentação de análise, de desenho, técnica e de testes unitários juntamente com o código-fonte das modificações necessárias às ferramentas da Oracle.

Foi então acompanhada uma implementação bem sucedida do projecto, inicialmente num ambiente de desenvolvimento e posteriormente no sistema de testes de integração do cliente.

Adicionalmente, foi necessário realizar testes de performance ao ORIB na arquitectura do sistema completo.

Estes foram feitos aos interfaces mais críticos do ORIB do sistema, responsáveis por importar dados de LS para o Oracle Retail Merchandising System (ORMS).

Nesta fase não foram pretendidas alterações arquitecturais ao ORIB, mas sim um Fine-Tuning de performance com vista a alcançar os volumes e tempos mínimos necessários.

Para tal, foi desenvolvida uma plataforma de testes capaz de recolher amostras de mensagens, analisar a sua distribuição pelas threads e por Business Object (BO) associado, de as alterar para evitar estados de erro na sua repetição, tal como de as publicar nos volumes esperados para o sistema.

Conseguiram-se atingir os alvos de performance necessários para a entrada do sistema em fase de produção graças à solução de multi-threading desenhada, juntamente com alguma configuração específica do ORIB.

Este projecto foi realizado pelo aluno e integrado no plano de trabalhos geral da Enabler junto do retalhista. A responsabilidade da integração da parte de modificação ao módulo de ORPM foi inteiramente deste, havendo um esforço de equipa conjunto com a autoridade de negócio e requisitos do sistema do cliente e com os elementos da Enabler responsáveis pelas alterações funcionais à aplicação de ORPM em si. A parte de testes de performance foi realizada juntamente com um outro elemento de integração da Enabler, o Mário Correia, que previamente desenhou as interfaces em teste e desenvolveu alguns elementos com os quais a

framework de testes foi elaborada. A sua orientação foi crucial para a realização dos testes, havendo no entanto a liberdade de desenvolvimento necessária para a análise, desenho e implementação do aluno.

1.2 Estrutura do Relatório

Este relatório começa por expor no segundo capítulo os conceitos que levam ao actual desenvolvimento de tecnologia no panorama dos ERPs para o retalho. Alguns conceitos base de retalho e algumas das tecnologias actuais utilizadas nos sistemas para retalho são aqui apresentados.

Foi optado que a descrição propriamente dita do projecto elaborado fosse guiada por exposições de problemas e de métodos para os resolver, com o objectivo sempre presente de manejar a integração de aplicações para auxiliar o desenvolvimento de uma solução geral de ERP customizada.

Numa primeira parte do terceiro capítulo que trata deste tópico é apresentado o contexto do cliente e do sistema específico onde este projecto foi aplicado. Seguidamente é feita a análise e posterior exposição do desenvolvimento em dois aspectos principais de integração, de adaptação de processos inter-aplicacionalmente e da performance das soluções de integração utilizadas.

Por fim, é feita uma conclusão e avaliação dos resultados obtidos.

Capítulo 2

Revisão Bibliográfica

2.1 Contexto

2.1.1 Introdução ao Retalho

Na área dos processos empresariais, um negócio de grandes volumes tende a ter uma grande complexidade e criticidade de processos e fluxos de informação que são únicos ao sector a que pertence e, em muitos casos, únicos em cada empresa.

A actividade de retalho é a maior área de negócio mundial totalizando 5.541.706M€ [EM06], e, nesta área em particular, a maioria destes fluxos e processos dividem-se pelas principais actividades específicas de negócio, diferindo assim das necessidades genéricas de uma empresa, como as finanças e os recursos humanos.

O negócio do retalho baseia-se fundamentalmente na gestão mercadológica, i.e. gestão dos items passíveis de serem vendidos e no seu percurso desde a origem, i.e., o fornecedor, até à sua subsequente venda e passagem a propriedade de terceiros no chamado Point of Sale (POS). A gestão de preços, chamada de Pricing, tal como todas as actividades promocionais é um caso específico da gestão mercadológica, fundamental para o negócio.

Tem de ser portanto garantida a manutenção de informação sobre todas as lojas e armazéns, ou entrepostos, e da mercadoria neles existente.

No caso particular do entreposto, a gestão eficiente do stock é crítica para um retalhista por nele assentar tipicamente toda a distribuição de items por um elevado número de lojas.

Por exemplo, os elaborados processos de selecção, ou Picking, das paletes de mercadoria que devem sair do armazém, podem levar a uma poupança significativa de recursos.

Outro conceito fundamental na actividade do retalhista é o transporte de mercadorias, ou o chamado Shipping. A gestão deste processo é crítica dado que pouca eficiência neste processo, ou a sua falha, pode levar a elevados prejuízos na empresa.

Outros conceitos como o pedido de compra ao fornecedor ou Purchase Order (PO), as devoluções de produtos, chamada de Return To Vendor (RTV), a recepção de recibos ou Receipts ou até mesmo a oportunidade para os funcionários encomendarem produtos, chamada de Open To Buy (OTB), governam a interacção do retalhista com os fornecedores.

Destes processos pode resultar uma grande capacidade de decisão sobre estratégias de uma empresa. No entanto, é preciso que haja uma extracção de conhecimento interessante a partir dos dados disponíveis. A este processo se chama Business Inteligence (BI).

No entanto, todos estes processos são relativamente gerais na actividade de um retalhista. Dada a grande competitividade da área de retalho, uma evolução em qualquer um deles que leve a uma vantagem numa actividade específica do negócio pode dar oportunidade para sucessos de outra forma inexploráveis e é pretendida por todos os grandes retalhistas.

2.1.2 IT no Retalho

A eficiente gestão de todas as actividades de negócio, tal como em qualquer outra área de negócio, dita o sucesso ou o insucesso de um retalhista.

Os grandes volumes de dados gerados pela constante transacção de items, que têm depois impacto directo nos relatórios financeiros da empresa, por exemplo, seriam impossíveis de gerir sem a ajuda de uma tecnologia que permitisse guardar eficientemente milhões de registos de compra e venda.

A capacidade de gerir preços de lojas e responder às necessidades de mudança quase em tempo real por toda a estrutura de lojas em simultâneo podem evitar a empresa de ter enormes prejuízos ou levá-la a obter magníficos ganhos. Neste caso específico são por isso procurados sistemas capazes de se manter operacionais a gerir estas tarefas mesmo em momentos de falha, conceito chamado de High Availability (HA), e com performance adequada.

Por outro lado, a criticidade de alguns dos fluxos de dados, como no caso dos pedidos de compras, requer que a informação seja transmitida

correctamente, i.e. nenhuma informação pode ser perdida no fluxo nem repetida sem intenção, necessidade abordada por mecanismos como o Two Phase Commit e persistência intermédia de dados.

No entanto, a maioria destas tecnologias foram criadas há muito tempo e para outros fins. O maior interesse de desenvolvimento de IT na área está, por consequência, na procura de criar aplicações mais adaptadas aos processos do negócio. Desenvolvimentos estes que resultam frequentemente num prazo de compensação pelos custos envolvidos, conhecido como Return Of Investiment (ROI), na ordem dos meses.

2.1.3 Engenharia do Software no Retalho

No que respeita a sistemas de informação empresariais, é preciso ter em conta que nenhum sistema consegue gerir por si só todos os processos da empresa, que está inclusive em constante mudança, pelo que se torna necessário facilitar a integração de várias aplicações informáticas com vista a satisfazer todas as necessidades de uma empresa, tal como substituir, quando necessário, qualquer uma destas aplicações.

Neste aspecto, torna-se relevante a capacidade de uma aplicação ser portável e ter uma arquitectura aberta ou facilmente adaptável e, consequentemente, portável.

Outra questão, que aborda necessidades de sistemas de retalho enunciadas na secção anterior, é a da prática de controlo de qualidade nos desenvolvimentos, através de testes unitários, de integração, de sistema e de performance. Sem estes, corre-se o risco de não completar com sucesso qualquer sistema, seja pela grande criticidade no tempo de ROI envolvido, seja pela desadequação da aplicação a exigências tão grandes dos sistemas ou até mesmo por falhas não antecipadas e não detectadas em fases intermédias do desenvolvimento.

2.1.4 ERPs Versus Aplicações Costumizadas

Uma abordagem à temática da sincronização entre as necessidades da empresa e as funcionalidades das aplicações é a de criar um sistema modular, com uma arquitectura possivelmente ortogonal em que os subsistemas são contidos e com interfaces bem definidos. Este sistema seria capaz de se adaptar às necessidades de qualquer empresa, como nos produtos de alguns vendedores de ERPs, como a Oracle e a SAP, usando módulos substituíveis ou complementares. Para além da óbvia vantagem do Out-sourcing de desenvolvimento das aplicações, este modelo de ERPs permite uma solução uniformizada que implemente boas práticas, o que

ajuda nas áreas presentes em todos os processos do negócio, como as finanças da empresa.

No entanto, apesar das vantagens, esta adaptação nunca será tão completa que deixe de ser necessário modificar os processos da empresa para a utilização da aplicação. Por outro lado, empresas que implementem estes produtos passam a executar o negócio de modo semelhante, sem oportunidade para vantagens competitivas provenientes da optimização dos seus sistemas de informação.

Outra abordagem completamente diferente é a de desenvolver um sistema customizado, ou seja, criar um sistema para satisfazer exclusivamente as necessidades da empresa. Esta solução é por vezes excessivamente custosa para as empresas, no entanto, caso esta limitação seja ultrapassada pelo impacto da personalização nos processos de negócio da empresa, pode ser uma opção vencedora, como no exemplo da Wal-Mart, que apenas comprou o módulo financeiro a um vendedor de ERPs [SAP07].

Entre estas duas abordagens encontra-se a customização de aplicações adquiridas comercialmente, ou Comercial Off-The-Shelf (COFT). Neste caso, empresas como a Enabler são contratadas dado o seu vasto conhecimento em ERP's específicos e a sua capacidade em conseguir modificá-los, oferecendo tipicamente serviços de consultoria, customização e suporte.

2.2 Referências Tecnológicas

Na área de retalho, as aplicações utilizadas hoje em dia são criadas já sobre uma pilha tecnológica impressionante, onde a portabilidade e integração são conceitos fundamentais para o desenvolvimento de novas ferramentas informáticas.

Seguem-se algumas referências a tecnologias que possibilitam a criação de aplicações de ERP aplicadas ao retalho, seguidas por uma introdução ao ERP de retalho.

2.2.1 J2EE e o Service Oriented Architecture

Em 1995, a Sun disponibilizou uma linguagem de programação orientada a objectos pretendida portável no respeitante aos sistemas onde as aplicações nela desenvolvidas correm, sob o guia orientador "Write Once Run Anywhere" (WORA) [JAVA98].

Actualmente, a linguagem de programação Java é amplamente utilizada nas mais diversas actividades de tecnologia da informação.

A arquitectura portável e constante desenvolvimento levaram a um gradual aumento de aplicação desta linguagem em servidores empresariais.

Com a crescente necessidade de capacidades da linguagem orientadas para o cenário empresarial, eventualmente a Sun disponibilizou uma framework especificamente orientada para os sistemas desenvolvidos pelas empresas, chamado Java 2 Enterprize Edition (J2EE). Os elementos presentes nesta edição são usados primariamente em sistemas prestadores de serviços, como servidores de aplicações web, de web services e de aplicações chamadas de Enterprise Jave Beans (EJB).

O modelo arquitectural orientado aos serviços destes sistemas é conhecido como Service Oriented Architecture (SOA). Nestes, as aplicações encontram-se num servidor especialmente desenhado para a função de as alojar, também elas prestadoras de serviços a aplicações cliente chamado de servidor ou contentor de aplicações.

As aplicações servidor são instaladas no contentor de aplicações através de mecanismos de importação, conhecidos como Deploy, usando um ficheiro onde todos os ficheiros necessários são contidos, como o Enterprize ARchive (EAR).

O servidor e o cliente, que é tipicamente uma aplicação noutra máquina, estão normalmente unidos por um middleware construído sobre o Java Remote Method Invocation (Java RMI), como no caso dos EJBs.

Frequentemente a aplicação cliente é também implementada como um servlet a correr num plug-in Java de um browser. Neste caso, a aplicação servidor precisa de manter uma porta e url disponíveis no servidor HTTP do servidor de aplicações para que haja um início da disponibilização dos serviços da aplicação.

2.2.2 Enterprise Java Beans

Os EJBs são aplicações que correm num contentor de aplicações J2EE e implementam serviços disponibilizados remotamente. A IBM e a Java introduziram estes elementos para facilitar a programação distribuída, construindo-os por cima dos protocolos de invocação remota definidos no RMI.

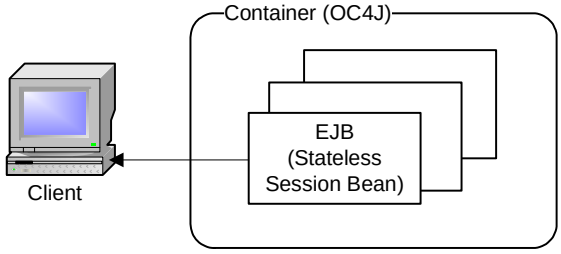


Figura 1.1: Modelo de Serviços de EJBs

Existem três tipos de EJBs disponíveis. Os Stateful Session Beans (SFSB) mantêm um estado entre chamadas do cliente, contrapostos aos Stateless Session Beans (SLSB), que implementam as lógicas de negócio que são corridas uma só vez ou que não têm dependências entre pedidos feitos.

Os Message-Driven Java Beans (MJB) foram desenvolvidos, por outro lado, para tratar dos pedidos assincronizadamente, utilizando a tecnologia de serviço de mensagens da plataforma Java, o Java Message Service (JMS). Estes distinguem-se também por não disponibilizarem métodos directamente ao cliente, mas apenas respondendo aquando a recepção de mensagens, uma característica que os torna apropriados para a construção de sistemas orientados a eventos, ou Event-Driven.

2.2.3 Java Message Service

Os Message Oriented Middlewares (MOM) foram introduzidos com intuito de flexibilizar a prestação de serviços usando a portabilidade e interoperabilidade de protocolos de mensagens. As comunicações tornam-se assíncronas ao se recorrer a um armazenamento intermédio para a eventualidade de um receptor das mensagens estar indisponível no momento do envio.

A especificação de MOM da Java, o Java Messsage Service, define, entre outros, interfaces para objectos produtores e consumidores de mensagens e para objectos onde se armazenam as mensagens. Estes últimos podem ser do tipo Queue ou do tipo Topic.

A Queue, como o nome indica, é uma fila de mensagens à espera de serem consumidos por ordem de chegada, ou seja, First-In, First-Out (FIFO), sendo retirados no momento em que tal sucede. Assim sendo, apenas um consumidor pode receber cada uma das mensagens.

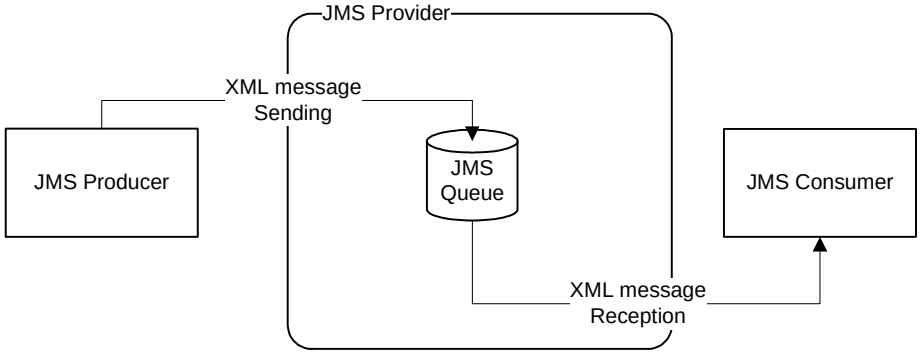


Figura 1.2: Modelo de Envio/Recepção

O Tópic está inserido no paradigma de publicação e subscrição de mensagens. Este permite várias recepções de uma mensagem, mais especificamente, uma por cada consumidor que tenha uma subscrição registada no tópico, chamado de subscritor. Esta subscrição pode ser feita persistentemente, i.e., até o subscritor se desregistar do tópico, as mensagens no tópico não são apagadas até ele as consumir, ou não persistentemente, caso em que o subscritor precisa de estar activo no momento da chegada da mensagem ao tópico, de modo a recebê-la.

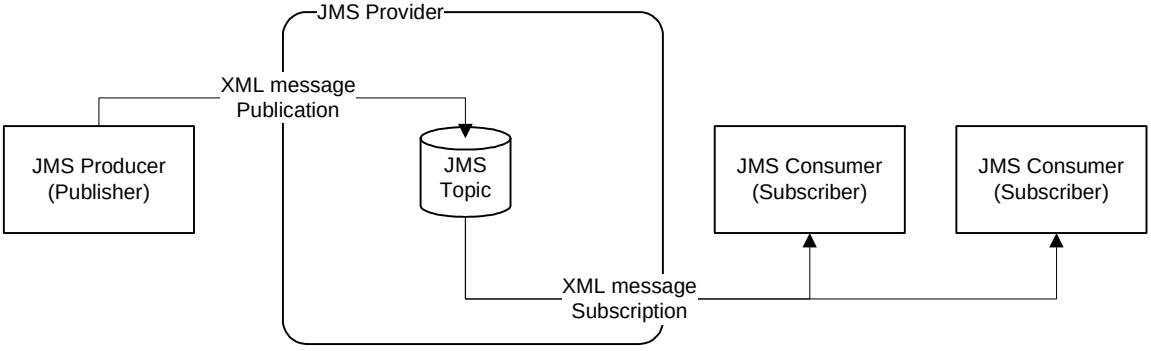


Figura 1.3: Modelo de Publisher/Subscriber

2.2.4 Bases de Dados

As bases de dados relacionais têm uma longa história na IT. No entanto, continuam a ser a base de uma aplicação empresarial. Vários vendedores de bases de dados mantêm desenvolvimento dos seus produtos. No entanto, A Oracle Corporation mantém a liderança [Gartner].

Hoje em dia, praticamente todas as aplicações seriam teoricamente possíveis de integrar com qualquer base de dados, precisando para isso apenas do adaptador certo, dado que as potencialidades das várias bases de dados são semelhante, tendo apenas processos diferentes de fazer o mesmo.

2.2.5 Frameworks de Persistência

De modo a facilitar o desenvolvimento de aplicações que recorressem à BD e dada a dificuldade de controlar os elementos da DB na maior parte das linguagens de programação, ferramentas de persistência são utilizadas, prestando serviços de acesso a estes elementos.

Uma das plataformas mais utilizadas hoje em dia é o Hibernate [HIB06], que utiliza mapeamentos no formato eXtended Markup Language XML entre os elementos da DB e objectos Java, e apresenta-os como interface de acesso à DB, podendo esta ser manipulada utilizando objectos Java.

Estas plataformas, para além do desenvolvimento rápido que proporcionam numa fase avançada do ciclo de desenvolvimento, podem também criar os mapeamentos da BD para Java automaticamente, acelerando o arranque do projecto.

2.2.6 Enterprise Resource Planing de Retalho

No mundo dos ERPs, há duas empresas que se têm destacado, a Oracle Corporation e a SAP AG que em 2004 tinham, respectivamente, 19% e 43% do mercado [GAR07].

No contexto das necessidades enunciadas anteriormente, a Oracle produz um software de ERP para retalho chamado Oracle Retail (OR, aqui analisado na versão 12.0) [OR06], que utiliza tecnologias abertas como J2EE e SOA, permitindo uma modificação facilitada do mesmo e uma consequente adequação aplicacional aos processos da empresa. Adicionalmente, o pacote disponibilizado pela Oracle é modular, permitindo a utilização de qualquer um dos módulos e a sua integração, através de alguns protocolos abertos como o avançado JMS e a simplificada transmissão de ficheiros por FTP, com aplicações externas.

A SAP veio também a introduzir na sua aplicação de ERP, nos anos que se seguiram à adopção destes elementos abertos de arquitectura por parte da Oracle, alguns elementos como o desenvolvimento em Java, e não só, sobre uma linguagem de proprietário [SAPJ07], como tinha vindo a fazer até então.

No entanto, na área do retalho, ainda não é certo qual o produto de ERP que liderará o mercado no futuro, tendo neste momento a SAP, actual líder mundial de venda de ERPs, uma fatia de mercado de ERPs de retalho ainda superior à Oracle depois da compra da Retek, empresa originalmente criadora do OR (na altura chamado de Retek), por parte desta.

2.3 Oracle Retail

A OR tem dois paradigmas principais sobre os quais constrói os seus módulos. Um é o J2EE, alojando EJBs no contentor de aplicações criado pela empresa, o Oracle Containers for Java (OC4J), e outro é de PL/SQL contido numa BD da Oracle, a Oracle Database 10g.

As aplicações J2EE comunicam com o servidor da Oracle utilizando um adaptador entre Java e DBs. Por outro lado, o interface gráfico deste corre na aplicação do cliente em Java também, utilizando as ferramentas da plataforma J2EE.

É o paradigma de aplicações J2EE que o OR utiliza para os seus principais mecanismos de integração internos ao OR e com aplicações externas, através do modelo de publish / subscribe de JMS Topics.

As aplicações PL/SQL utilizam a tecnologia de Oracle Forms para implementar os seus GUIs. Embora não seja tão padronizada na indústria, esta tecnologia está completamente integrada na DB da OR, e tem a característica de permitir um desenvolvimento facilitado.

2.3.1 Oracle Retail Integration Bus

Esta é uma das aplicações J2EE do OR, sendo o principal mecanismo de integração aplicacional desta. O modelo arquitectural baseia-se no paradigma de Publisher / Subscriber sobre a tecnologia de JMS Topics [ORIB06a].

O Oracle Retail Integration Bus (ORIB) garante a entrega de mensagens utilizando transacções que seguem o protocolo de two-fase commit de XA [XA] e a persistência intermédia em memória e ficheiros do Operative System (OS) do servidor de ORIB.

A utilização de JMS Topics e de Persistant Subscribers permite, por outro lado, a entrega de mensagens a todos os sistemas aos quais seja necessário fazer a transmissão.

O Publisher é, então, um EJB do tipo SLSB que insere uma mensagem num tópico, enquanto que um Subscriber é um EJB do tipo MDB, começando o processamento quando recebe uma mensagem no tópico ao qual está associado.

As mensagens são construídas em XML, com o elemento de base sendo o RibMessages, que contém pelo menos uma mensagem, sob o elemento RibMessage.

O ORIB abstrai cada publicador e subscritor em tópicos usando adaptadores chamados de eWays, onde estão contidos os processos de endereçamento, multi-threading e routing entre outros.

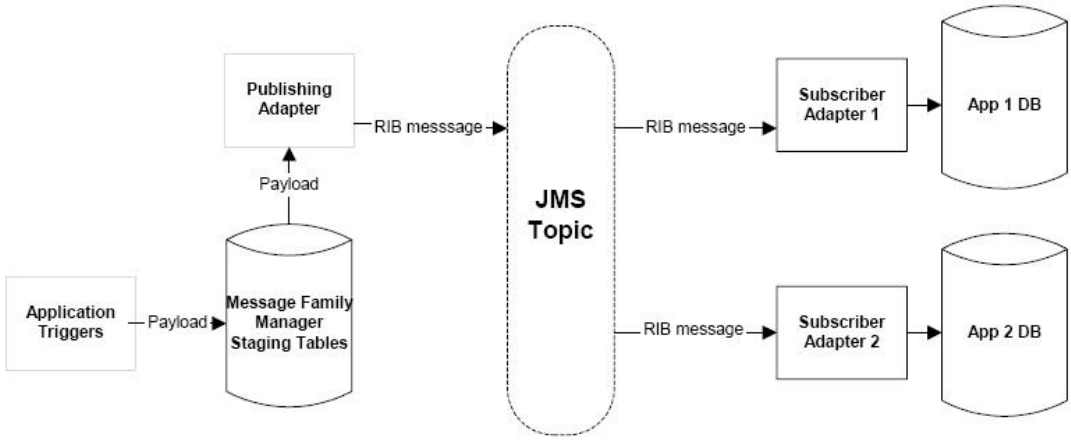


Figura 1.4: Paradigma de Publicação/Subscrição em OR

Há ainda dentro do ORIB um tipo de adaptadores especiais que subscrevem a mensagens e, sem saírem do contexto do ORIB, por sua vez publicam novas mensagens. O nome destes provém de Transformation Address Filter/Router (TAFR) e servem para executar operações de integração específicas de alguns fluxos de mensagens, podendo-se desta forma alterar o seu conteúdo, filtrar segundo algum parâmetro e direccioná-las para os diferentes destinos.

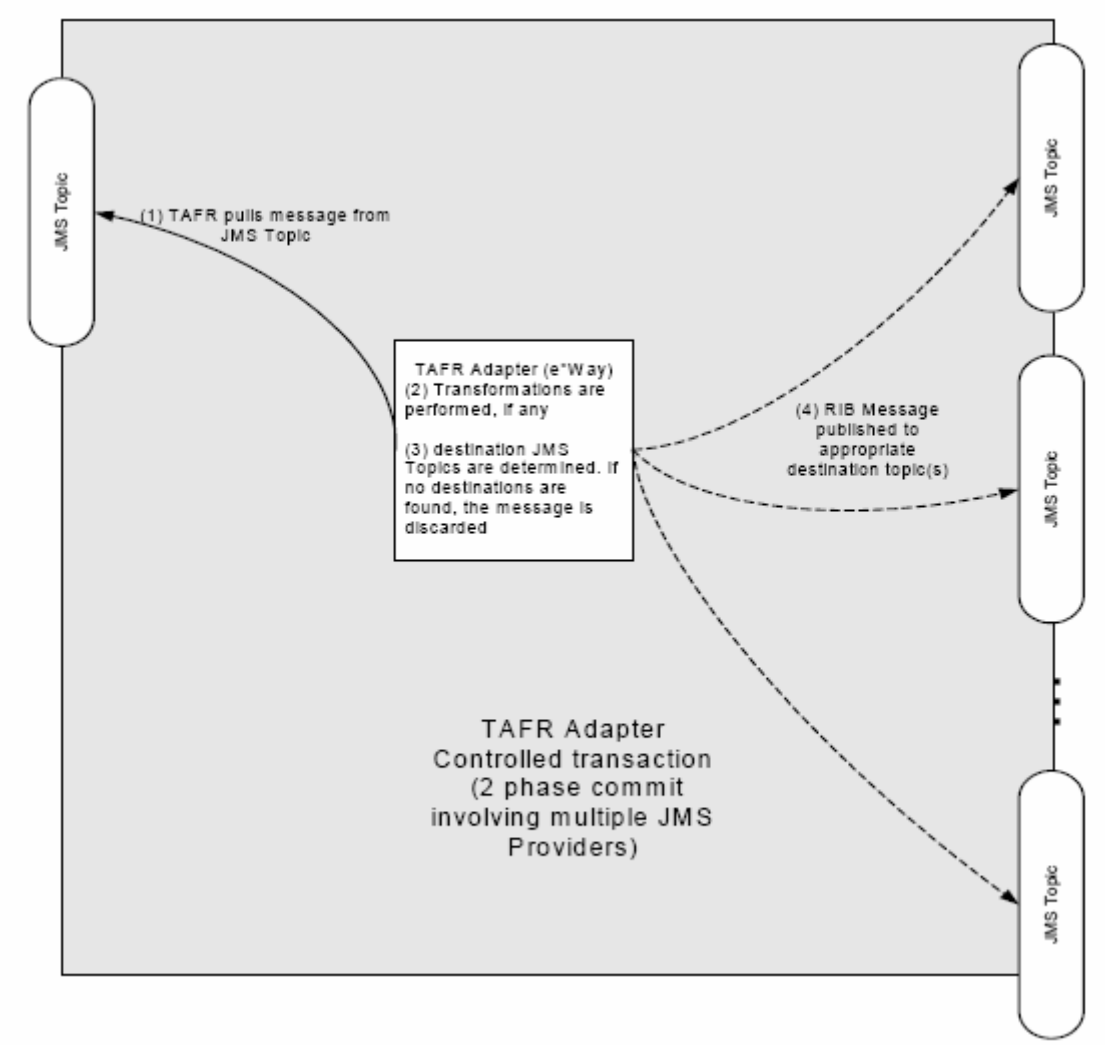


Figura 1.5: Processo de TAFRs

Adicionalmente, a implementação de ORIB é flexível nas localizações físicas dos adaptadores, permitindo que uma mensagem atravesse uma rede com recurso a uma bridge.

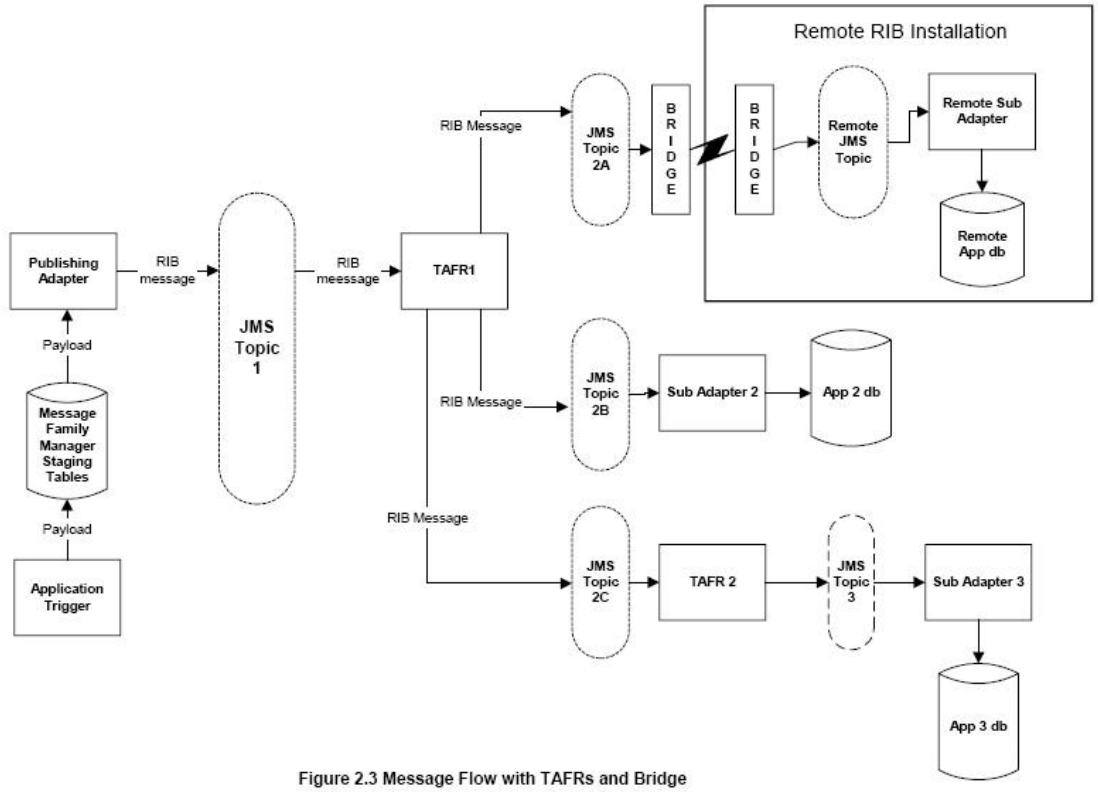


Figura 1.6: Flexibilidade de Integração

Os fluxos de informação transmitida pelo ORIB são criados sobre as mais variadas necessidades e processos de negócio do retalhista, sendo que um agrupamento de tipos de mensagens relacionados com a mesma actividade / processo é chamado de família de mensagens ou Message Family. Tipicamente, os tópicos JMS (e os publicadores e subscritores por consequência) são associados a uma message family. Adicionalmente, as variações nas mensagens dentro da mesma família de mensagens são as chamadas de tipos de mensagens, ou Message Types.

O conteúdo das mensagens é por sua vez chamado, no abstracto, de payload e contém a informação transaccional a transmitir. A payload tem que ser preenchida na aplicação publicadora e tem também que seguir as regras definidas por um documento XML na forma de um XML Schema Definition (XSD). Estes documentos, associados às famílias de mensagens, são utilizados para efeitos de parsing pelo ORIB, indicando qual a informação na mensagem e em que estrutura se encontra.

2.3.2 Integração de Módulos OR

Há diversos métodos de integração no OR, sendo que o ORIB é o principal destes.

Havendo duas tecnologias nas quais se implementam os módulos de OR, o ORIB tem dois interfaces principais.

Na integração com aplicações PL/SQL, o início do envio de mensagens começa na alteração de tabelas relevantes e accionamento de um trigger de base de dados associado à operação feita a estas e a uma família de mensagens. Após o devido processamento, há uma inserção numa tabela intermédia, chamada de Staging Table, de dados sobre a mensagem a enviar, e a responsabilidade da aplicação na publicação acaba, continuando no adaptador de ORIB.

A criação das payloads pode ser feita em dois momentos distintos. A primeira abordagem da Oracle foi a de criar as payloads directamente em XML no trigger. Os tipos de dados na DB resultantes eram os Character Large Object Bynaries (CLOBS).

Eventualmente, dado o relativamente longo tempo de parsing que estes objectos requeriam, a Oracle passou a utilizar uma tecnologia da sua base de dados que reunia características de programação orientada a objectos (OO) como hierarquia e polimorfismo e tipos de dados de DBs, chamada de Oracle Objects. Estes eram bastante mais rápidos de processar dado os seus elementos estarem em tipos de dados nativos da DB em vez de XML.

Mesmo depois da adesão aos Oracle Objects, alguns interfaces de ORIB mantiveram-se em CLOBs, pelo que na versão 12 de OR analisada, ambos os interfaces existem. Tanto num caso como noutro, a publicação e subscrição de mensagens é feita recorrendo a uma chamada, no caso da publicação, a uma função da base de dados chamada GETNXT e, no caso da subscrição, a uma função chamada CONSUME, por parte do adaptador para tópicos JMS da SeeBeyond, chamado de eWay.

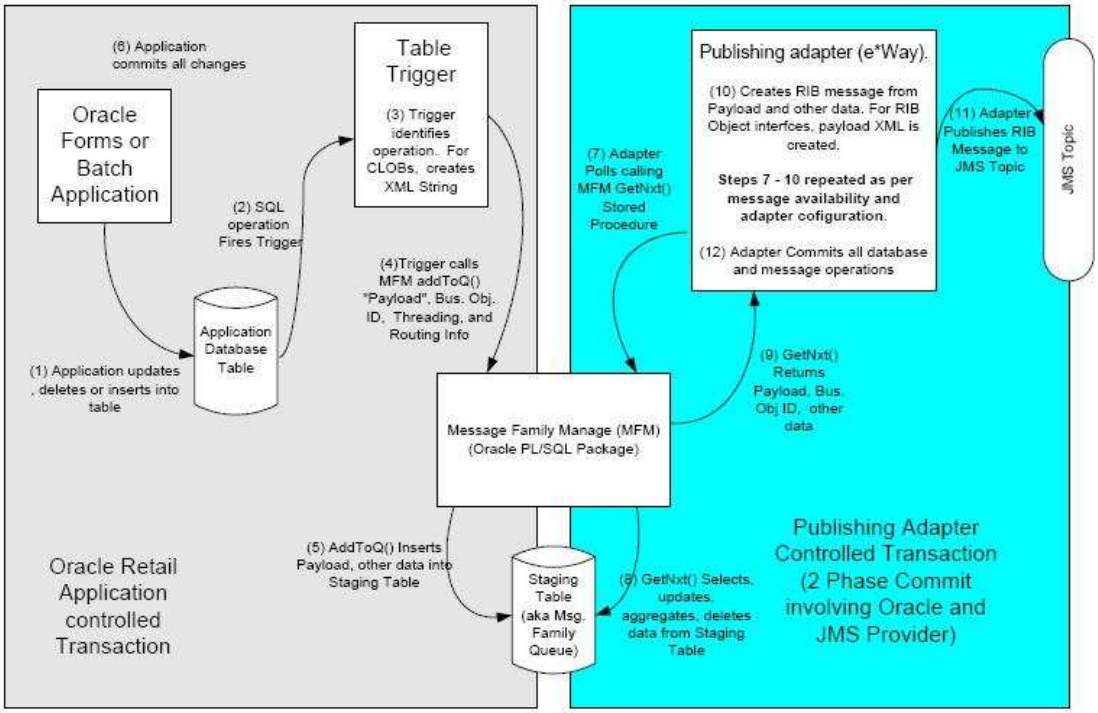


Figura 1.7: Mecanismo de Publicação PL/SQL

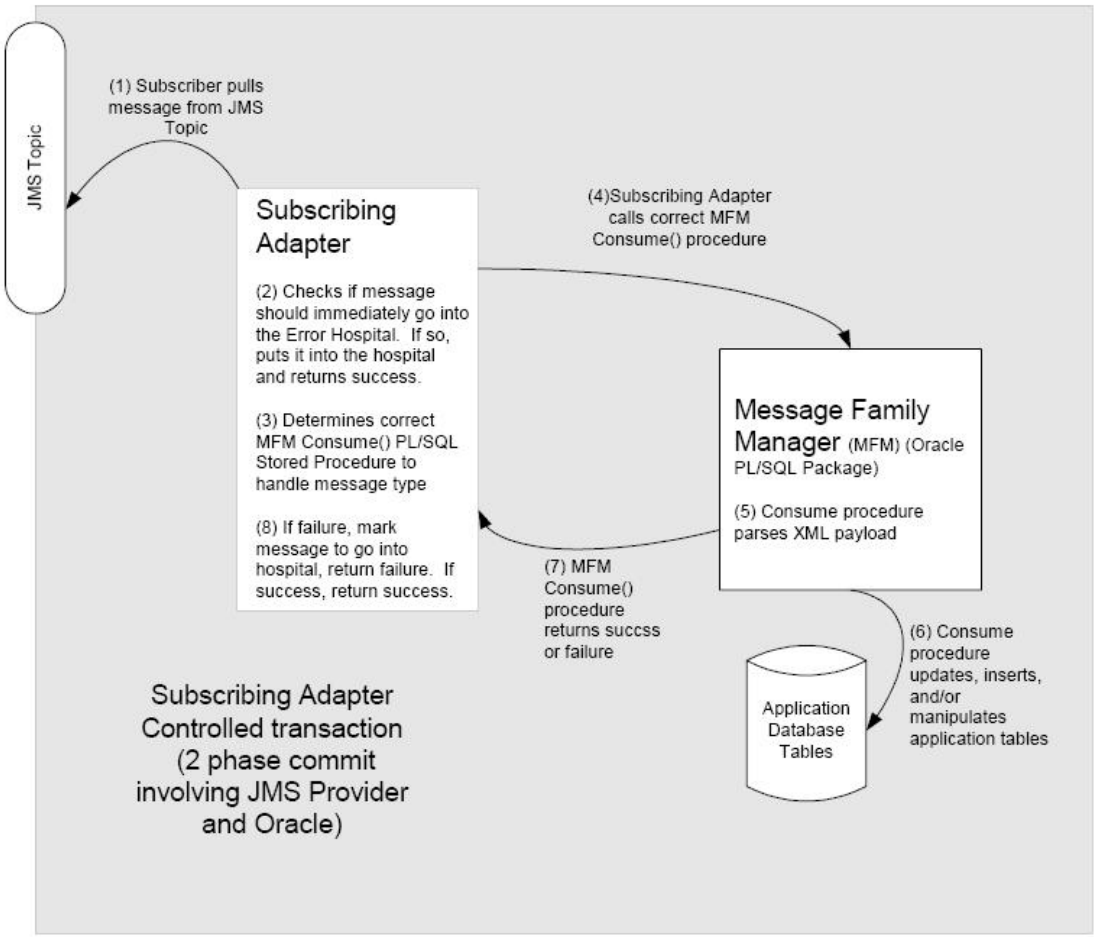


Figura 1.8: Mecanismo de Subscrição PL/SQL

Paralelamente, a integração com aplicações J2EE começa com a inserção de informação de payload da parte do EJB que executa as lógicas do negócio. Seguidamente, é chamada uma classe responsável por transformar as payloads em XML e injectar a mensagem resultante na JMS.

As classes em Java que representam as payloads, juntamente com os Oracle Objects criados para o mesmo fim, têm mapeamento directo com o XML das mensagens e com a definição das famílias e tipos de mensagem, os XSD's.

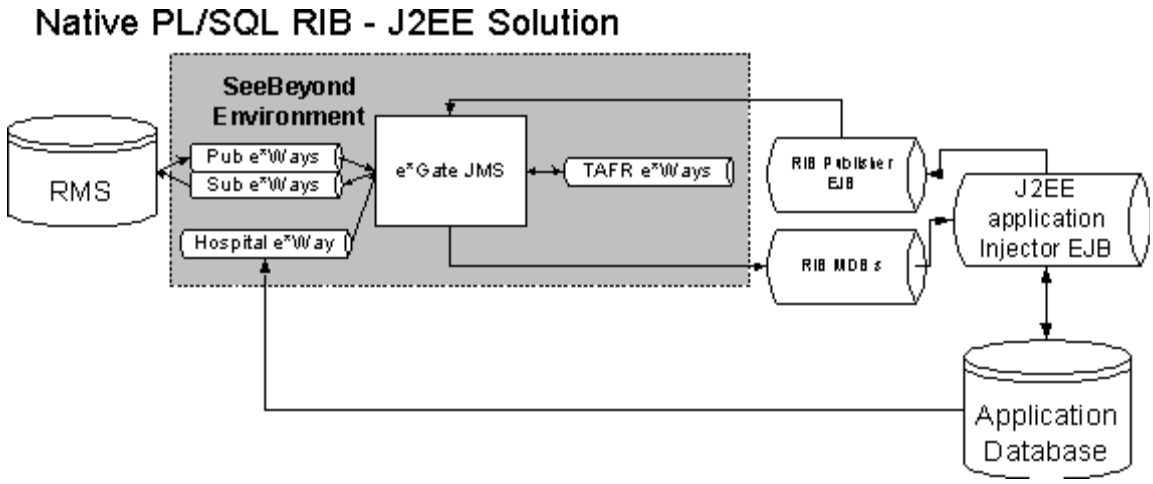


Figura 1.9: Mecanismo de Publicação/Subscrição J2EE

Adicionalmente, ainda há a possibilidade de integrar aplicações enviando ficheiros pelo File Transfer Protocol (FTP). Esta técnica é mais apropriada aos processos que dispensam a interacção do utilizador, chamado de processamento em lote, ou Batch.

Dentro do ORIB, há um mecanismo de integração também apropriado ao uso dentro do modelo de processamento Batch. Este é constituído por um eWay que faz a ponte entre ficheiros de batch e os JMS topics, sendo uma boa alternativa ao uso de FTP por ser caracterizado de todas as vantagens do modelo de publicação / subscrição.

O ORIB, para além de manter as mensagens persistidas para a eventualidade de, no momento de recepção, o subscritor estar offline, mantém também na integridade dos fluxos de dados através de um registo de mensagens em erro, para que estes sejam mais tarde corrigidos. As mensagens em erro vão parar a um mecanismo de repetição da tentativa de envio e de, no caso da falha persistir, alojamento das mensagens de erro chamado de ORIB Hospital.

O ORIB Hospital, no seu funcionamento normal, aloja também as mensagens seguintes dependentes de uma mensagem de erro, i.e., se uma entidade de negócio representada por uma mensagem está em erro, o ORIB assume que as mensagens seguintes referentes a essa entidade podem depender do processamento correcto da primeira mensagem. Este mecanismo evita, por exemplo, que no caso de uma mensagem de criação de uma promoção falhar a sua recepção, mensagens seguintes de modificação dessa mesma promoção sejam enviados para o hospital sem precisarem de tentativa de processamento, que pode levar a estados corruptos dos dados da DB.

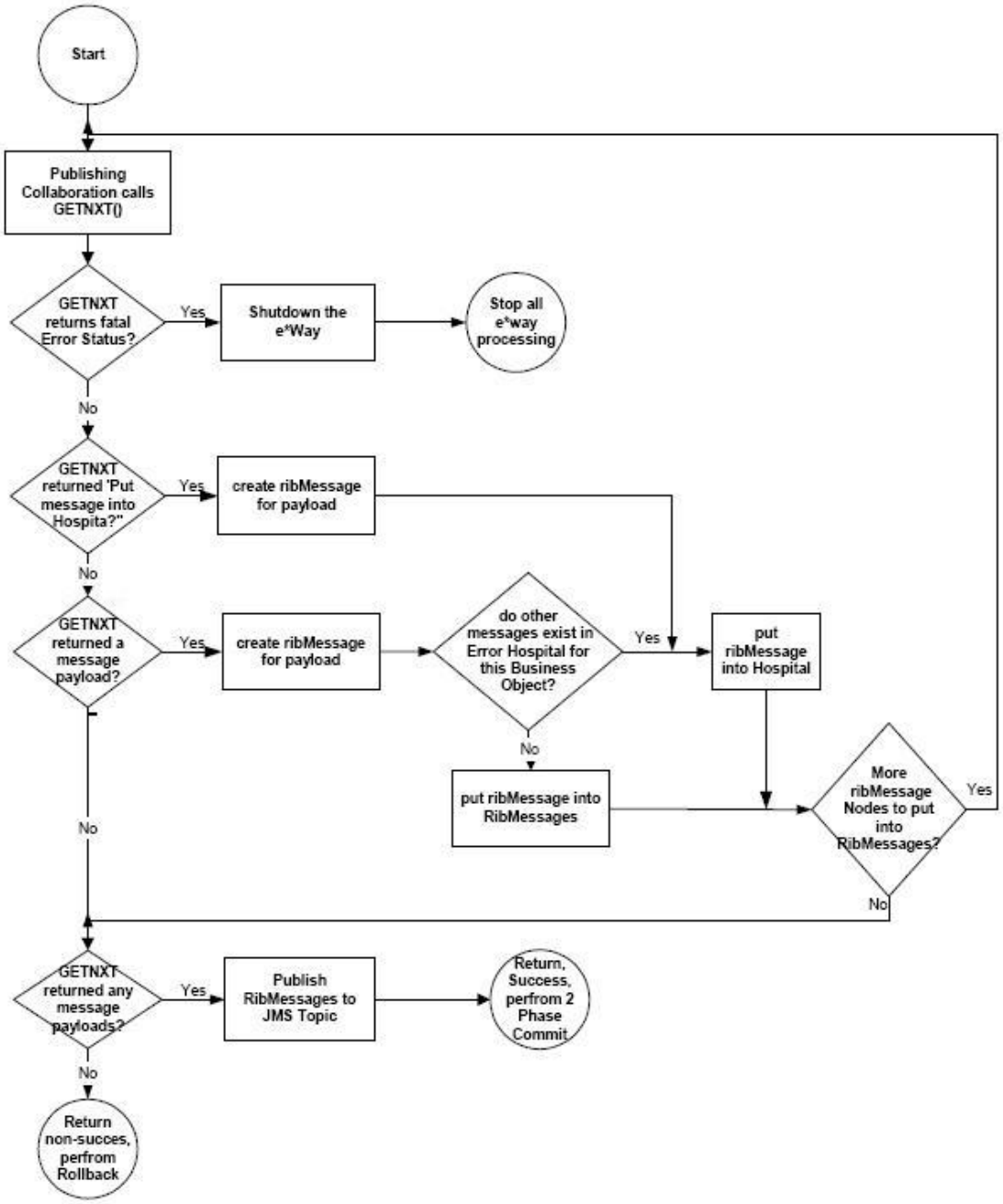


Figura 1.10: Publicação e ORIB Hospital

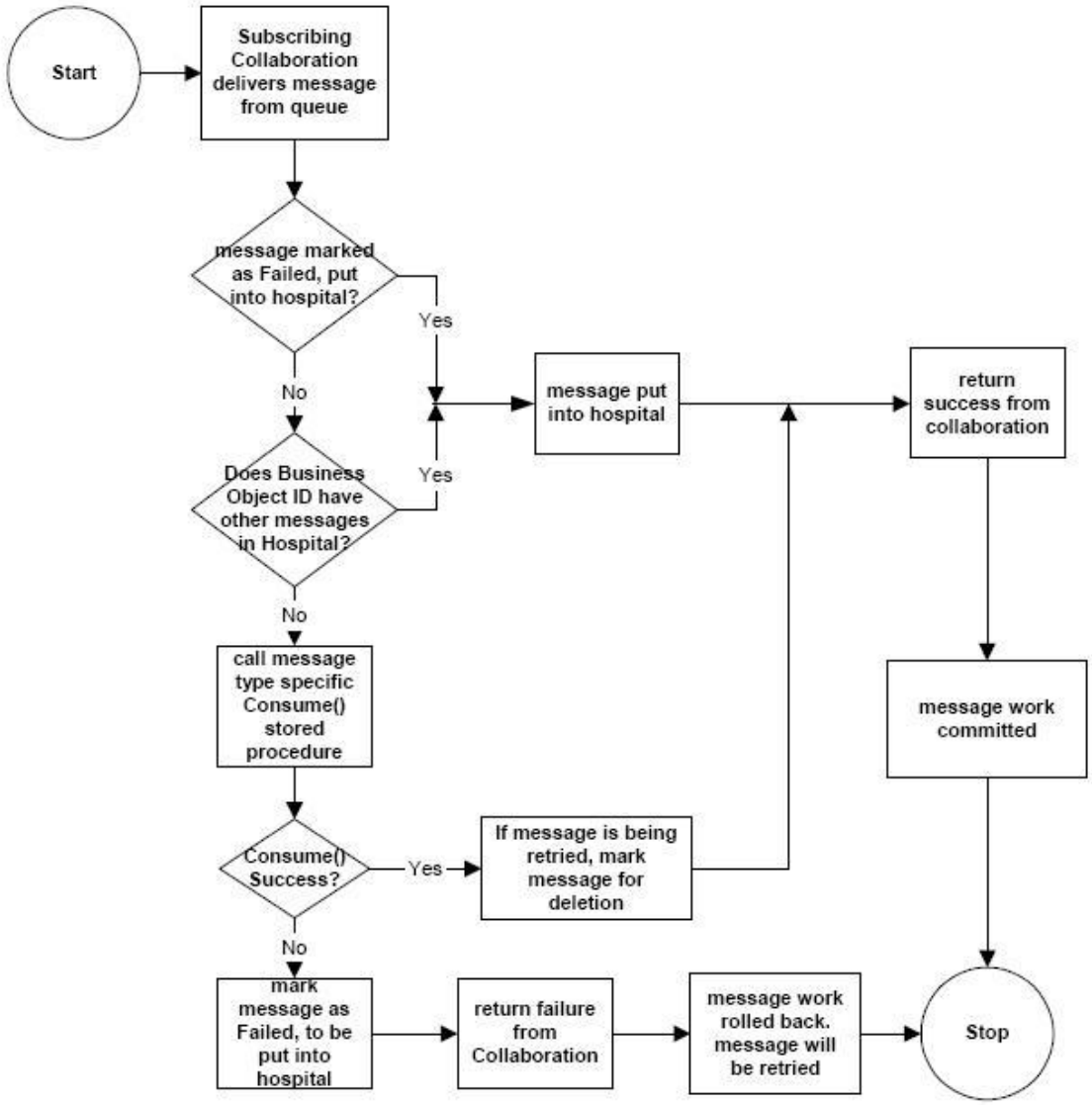


Figura 1.11: Subscrição e ORIB Hospital

2.3.3 Oracle Retail Price Manager

A manutenção de preços do sistema de um retalhista é feita no OR pelo Oracle Retail Price Manager (ORPM). Esta aplicação permite gerir os preços dos items para venda regular, os de liquidações de stock, tal como os diferentes tipos de promoções e as estratégias de preços do retalhista [ORPM06a].

O ORPM é implementado em J2EE e seguem um modelo de camadas, que auxilia a estruturação da aplicação e define uma metodologia de desenvolvimento onde as funcionalidades se encontram compartimentadas horizontalmente por níveis de abstracção e verticalmente por lógicas de negócio e casos de uso.

As camadas definidas por esta arquitectura são as de cliente, de serviços da aplicação, de núcleo do negócio ou Core, de persistência e de DB.

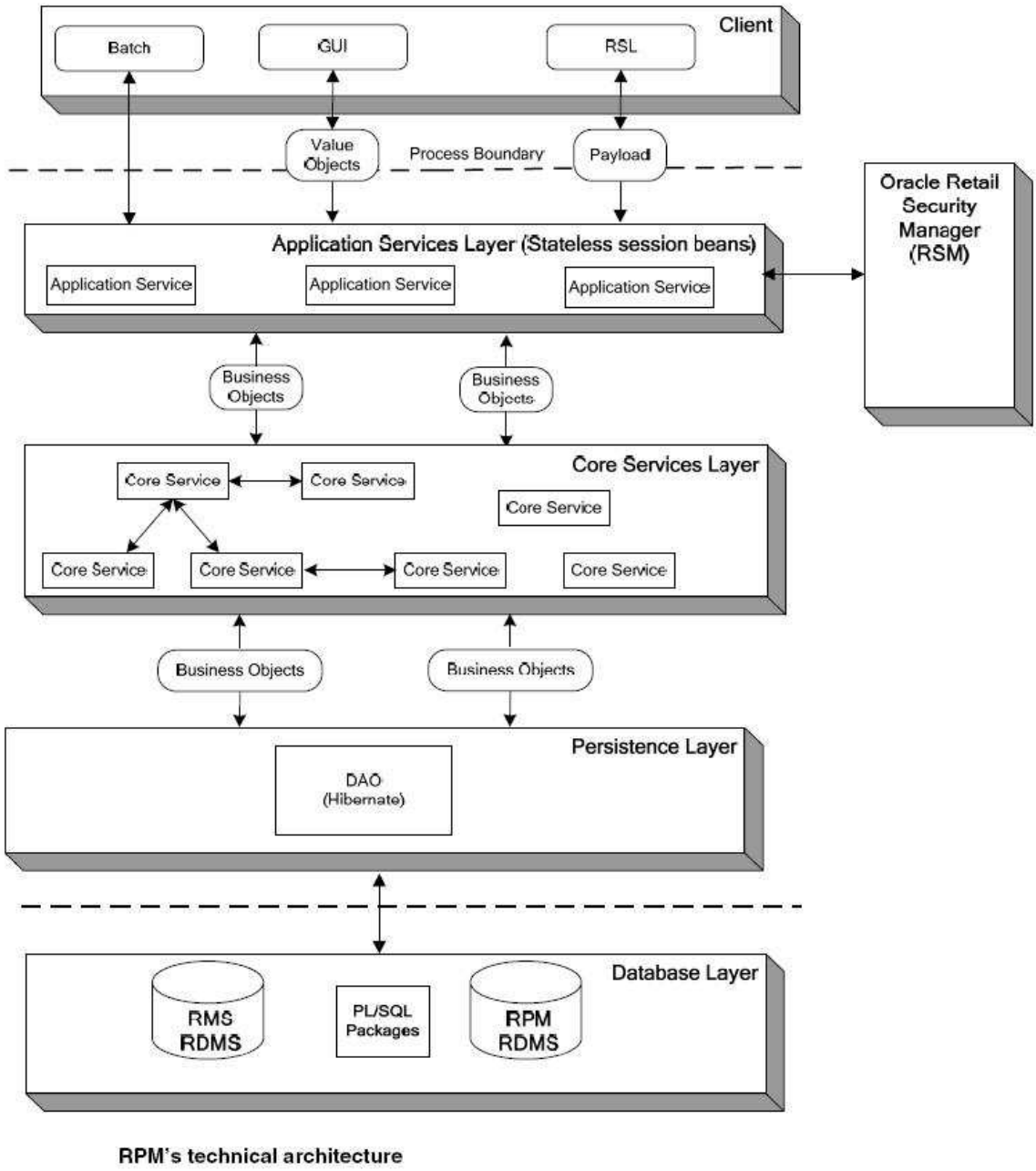


Figura 1.12: Arquitectura por Camadas de ORPM

A camada de cliente efectua pedidos à camada de serviços da aplicação e pode ser implementada sob a forma de uma aplicação gráfica Java criada sobre a plataforma gráfica SWING. A comunicação entre o cliente GUI e a camada de serviços da aplicação é feita com recurso a objectos-valor, ou Data-Objects, que apenas contém os valores necessários para o display e referência dos elementos presentes nas páginas da aplicação cliente e são de transmissão e processamento rápidos, auxiliando a boa performance da aplicação.

Outro método de aceder aos serviços da aplicação ainda dentro da camada de cliente é a de batch, onde uma classe java contida num EJB

RPM interage directamente com a camada respectiva por invocações a métodos Java.

Uma terceira implementação de cliente pode aceder aos serviços da aplicação através do protocolo Oracle Retail Service Layer (RSL), no qual, mediante o uso de classes Java que implementem os interfaces definidos pelo protocolo, outras aplicações OR utilizam alguns dos serviços do ORPM. A comunicação com a camada de serviços faz-se pelo envio de payloads de mensagens definidas nos protocolos de ORSL.

A camada de serviços de aplicação processa os pedidos do cliente e serve de interface com a camada de Core. Para efectuar um serviço pedido pelo cliente, esta pode chamar vários serviços da camada de Core ou até mesmo outros serviços da camada de serviços da aplicação.

A camada de serviços da aplicação comunica ainda com o módulo de segurança do OR, o Oracle Retail Security Manager (ORSM), para efectuar a validação das permissões do utilizador a cada uma das funcionalidades da aplicação, dado o seu nível de acesso.

A camada de Core efectua os processos de negócio do ORPM, independentemente dos processos do cliente da aplicação. Os processos são criados à volta de objectos de Negócio, ou Business Objects (BO). De modo a assegurar atomicidade, consistência, isolamento e durabilidade de alguns dos objectos da lógica do negócio, ou seja os atributos atomicity, consistency, isolation and durability (ACID), é, em alguns casos implementada, uma máquina de estados finita, ou Finit State Machine (FSM) para gerir o ciclo de vida dos objectos. Estes objectos podem estar completamente contidos nesta camada, mas são tipicamente recebidos de e enviados para a camada de persistência e servem também como objectos de comunicação entre a camada de Core com a camada de serviços da aplicação.

A camada de persistência, em ORPM, está implementada em Hibernate, que faz o mapeamento das tabelas da base de dados relacional para objectos java, sendo capaz de os persistir e de os preencher com a informação na base de dados.

A camada de base de dados está implementada na base de dados da Oracle, contendo as tabelas necessárias aos processos de negócio tal como as packages em PL/SQL necessárias, onde o processamento é bastante mais rápido do que em Java. A DB de ORPM está inserida na mesma DB do ORMS, descrita na secção que se segue.

2.3.4 Oracle Retail Merchandising System

O sistema de gestão de retalho central do OR, o Oracle Retail Merchandising System (ORMS), está envolvido em praticamente todos os

processos de negócio do OR, desde a gestão de pedidos aos fornecedores e acordos, ou Deals, sobre compras de items aos mesmos, à previsão de vendas, ou Forecasting e a reposição de stock, ou o Replenishment [ORMS06a].

O ORMS tem vindo a ser desenvolvido ao longo das várias versões de OR, e mantém-se inteiramente implementado na tecnologia estável da Oracle, o PL/SQL na base de dados 10g da Oracle juntamente com os Oracle Forms.

Muitos dos processos mais volumosos em termos de comunicação do OR estão presentes no ORMS dada a necessidade de controlo de escalonamento dos processos de retalho da empresa, como a recepção diária pontual de grandes volumes de Receipts, Shipments, Finantials, etc. [ORMS06b].

Capítulo 3

Integração Aplicacional em Retalho

3.1 Introdução

3.1.1 Cliente

O cliente da Enabler é um retalhista internacional dentro dos 5 maiores retalhistas do mundo, sendo que a média da venda destes 5 é de 87.255,76M€. Embora quase metade destas vendas sejam da responsabilidade do primeiro da tabela, o Wal-Mart, a média dos restantes quatro, de 54.160,73M€, continua a ser indicador de uma magnitude de negócio considerável [DEL08].

Disto resulta que este projecto se insere no desenvolvimento de um sistema encarregue por uma carga transaccional elevada, sendo responsável por gerir mais de 50% das instalações e processos de retalho da empresa, que se esperam triplicar no espaço de 3 anos.

O cliente está adicionalmente activo em outras áreas não relacionadas com o retalho, que estão fora do âmbito da Oracle Retail, dos projectos da Enabler e do presente projecto.

3.1.2 Sistema do Cliente

Após alguns anos de implementação de soluções de ERP "vanilla", i.e., sem customizações, ou com relativamente baixo nível de customização, o cliente da Enabler decidiu que a experiência adquirida nos vários anos de actividade lhe dava maior visão sobre as características necessárias num ERP para retalho do que as empresas que até então lhe desenhavam os sistemas.

Nessa nova perspectiva, o cliente comprou os direitos de utilização e costumização do OR e de outros sistemas empresariais e contratou a Enabler e outras empresas especializadas nos restantes sistemas para auxílio no desenho, desenvolvimento e implementação de um ERP que satisfizesse os seus requisitos de negócio específicos.

O sistema resultante é uma evolução do ERP anterior do cliente, mantendo alguns dos sub-sistemas deste e migrando toda a informação fundamental da empresa para o seu substituto, criado à volta do OR.

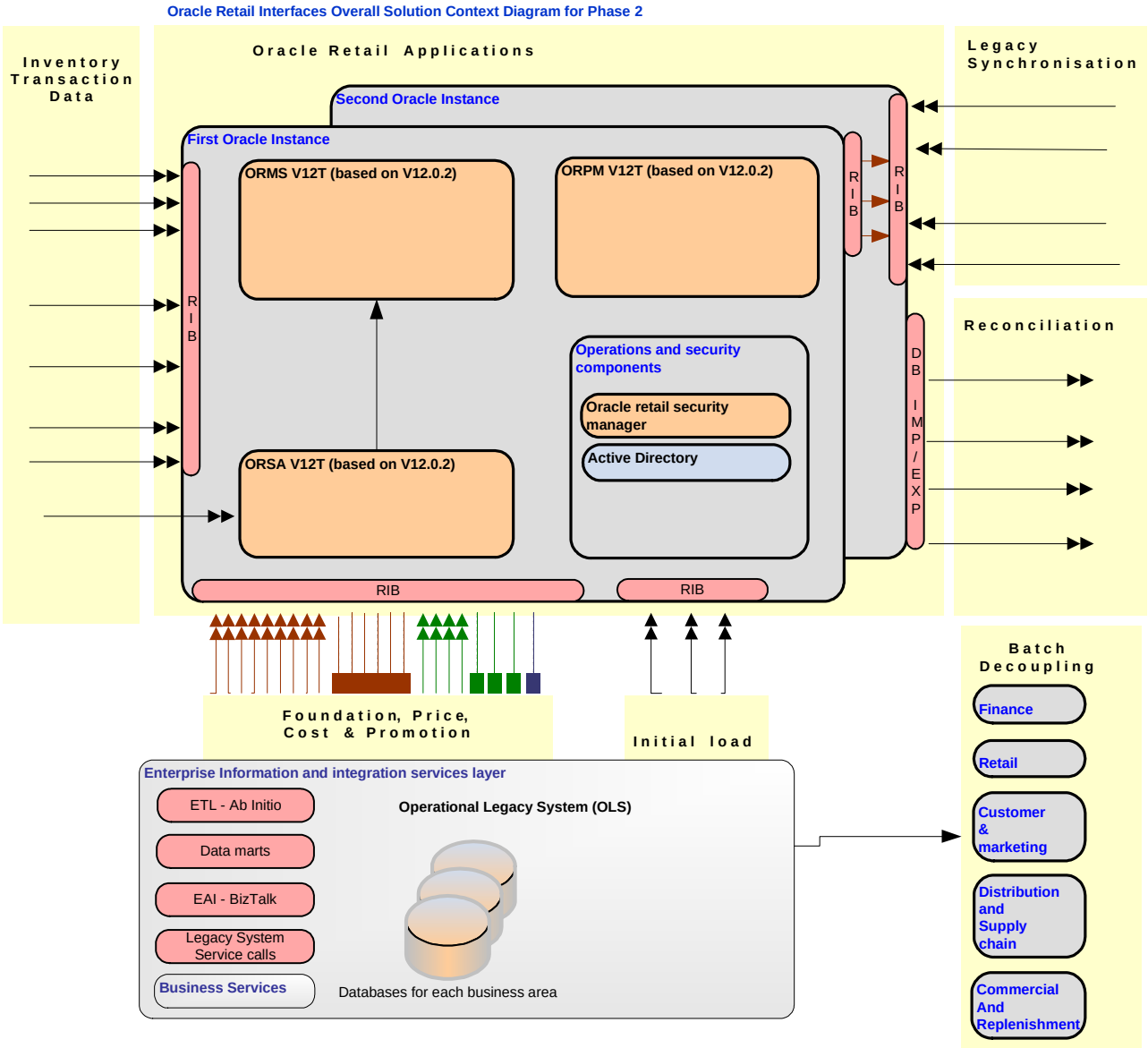


Figura 1.13: Diagrama Integracional de Referência da Solução do Cliente

3.2 Planeamento

O sistema geral do cliente e o projecto actual tiveram que seguir uma metodologia própria de desenvolvimento. No caso do projecto em

particular, o planeamento seguiu o modelo cascata, onde acabando uma actividade se seguiu para a seguinte, sem grande paralelismo de tarefas nem ciclos de desenvolvimento.

3.2.1 Fases de Desenvolvimento do Sistema do Cliente

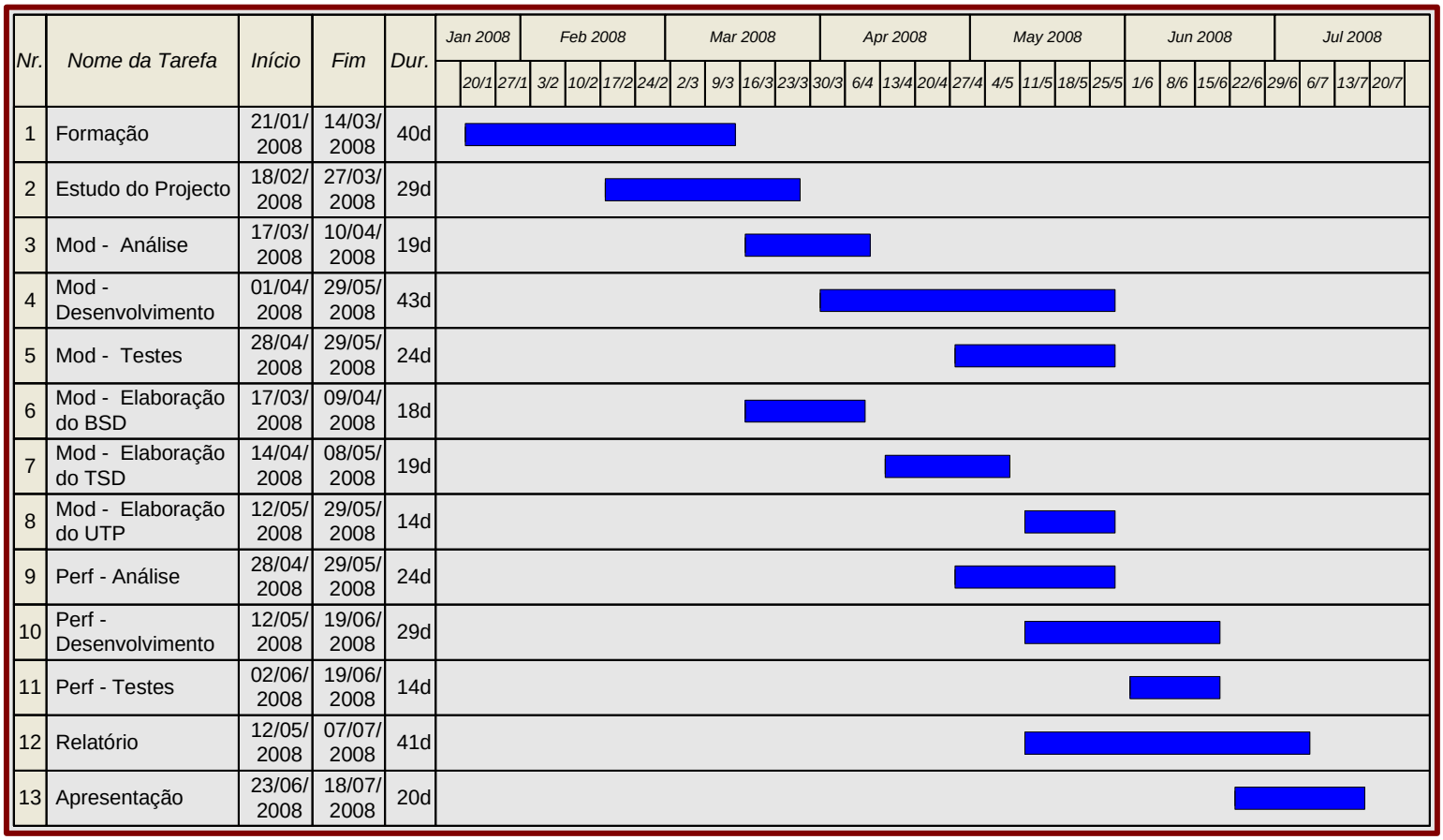
O desenvolvimento foi faseado, passando primeiro por uma versão intermédia implementada e funcional que mantinha alguns dos sistemas Legacy para além dos planeados para a versão final. Esta versão recebe então os dados de migração e continua a funcionar paralelamente ao sistema anterior. O objectivo é ter um meio de conferir se o âmbito do projecto deve mudar numa fase intermédia.

As migrações seriam estruturadas por uma imagem instantânea inicial do sistema, ou snap-shot, chamada de Cut-Over, introduzida para o novo sistema. Após esta introdução inicial, passam a ser migrados os dados de um dia inteiro de actividade em ficheiros individuais, um por cada dia de funcionamento desde o momento do Cut-Over até ao momento em que se procedeu à migração. A seguir, o sistema está pronto para receber dados paralelamente ao sistema a substituir. Espera-se nesta fase que os testes e ajustes sejam feitos para avaliar os desenvolvimentos adicionais não planeados.

Seguidamente, o projecto segue para alteração a este sistema intermédio para o novo sistema. Só então é que os sistemas Legacy desnecessários são todos excluídos e se procede ao teste final de sistema, havendo de novo migração de dados.

3.2.2 Planeamento do Projecto

O projecto teve originalmente um prazo máximo de 3 meses, tendo sido depois estendido em mais um mês para terminar algumas novas actividades de testes de performance ao sistema.



Nr.	Nome da Tarefa	Início	Fim	Dur.
1	Formação	21/01/2008	14/03/2008	40d
2	Estudo do Projecto	18/02/2008	27/03/2008	29d
3	Mod - Análise	17/03/2008	10/04/2008	19d
4	Mod - Desenvolvimento	01/04/2008	29/05/2008	43d
5	Mod - Testes	28/04/2008	29/05/2008	24d
6	Mod - Elaboração do BSD	17/03/2008	09/04/2008	18d
7	Mod - Elaboração do TSD	14/04/2008	08/05/2008	19d
8	Mod - Elaboração do UTP	12/05/2008	29/05/2008	14d
9	Perf - Análise	28/04/2008	29/05/2008	24d
10	Perf - Desenvolvimento	12/05/2008	19/06/2008	29d
11	Perf - Testes	02/06/2008	19/06/2008	14d
12	Relatório	12/05/2008	07/07/2008	41d
13	Apresentação	23/06/2008	18/07/2008	20d

Figura 1.14: Plano de Gantt

3.2.3 Metodologia

A metodologia de desenvolvimento adoptada pela Enabler é a de V, havendo uma produção de documentação gradualmente menos abstracta e cada vez mais técnica, desde a análise de requisitos, passando pelo desenho de sistema, pela a arquitectura de sistema e modular até chegar ao ponto de desenvolvimento de código. Após isto começa a fase de testes, que começam por ser sobre o produto técnico nos unit tests, passando por testes de integração, depois de sistema e finalmente de aceitação.

3.3 Modificação ao Módulo de ORPM e de ORIB

Na primeira parte do projecto descrito neste relatório adaptou-se o ORPM e o ORIB para integrar os processos promocionais do cliente.

Uma promoção no ORPM é uma alteração dos preços de certos items tipicamente numa data futura, sendo que este permite várias modalidades promocionais. A informação sobre as alterações a ser feitas aos preços é guardada em tabelas de promoções (anexo A) e de preços

futuros. A consistência dos dados de preço é mantida através de uma complexa verificação de conflitos entre promoções e através do ciclo de vida de cada promoção, tendo esta que passar por criação, aprovação e activação na loja. No dia anterior ao início de uma promoção, as alterações são passadas das tabelas de preços para as lojas.

3.3.1 Integração no Sistema Actual

O sistema promocional anterior do cliente, assente sobre o modelo de batch, possui algumas diferenças para o sistema Real-Time do ORPM, pelo que algumas adaptações tiveram que ser feitas.

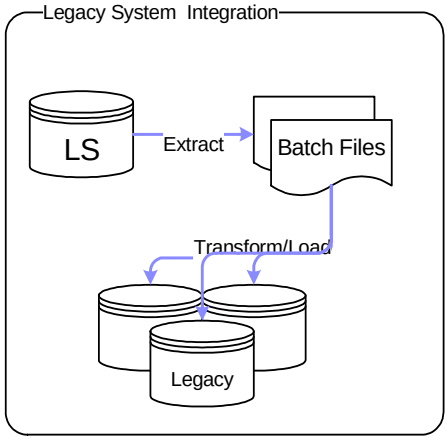


Figura 1.15: Integração no Sistema Anterior

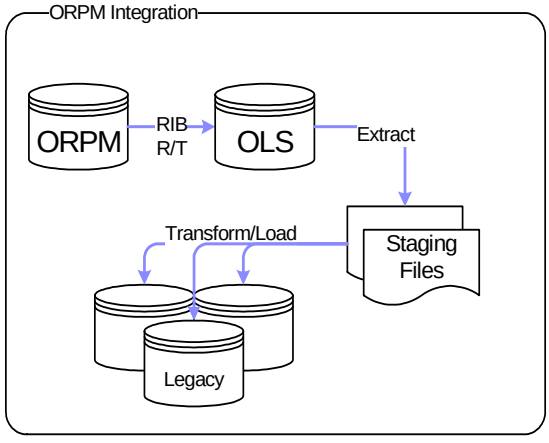
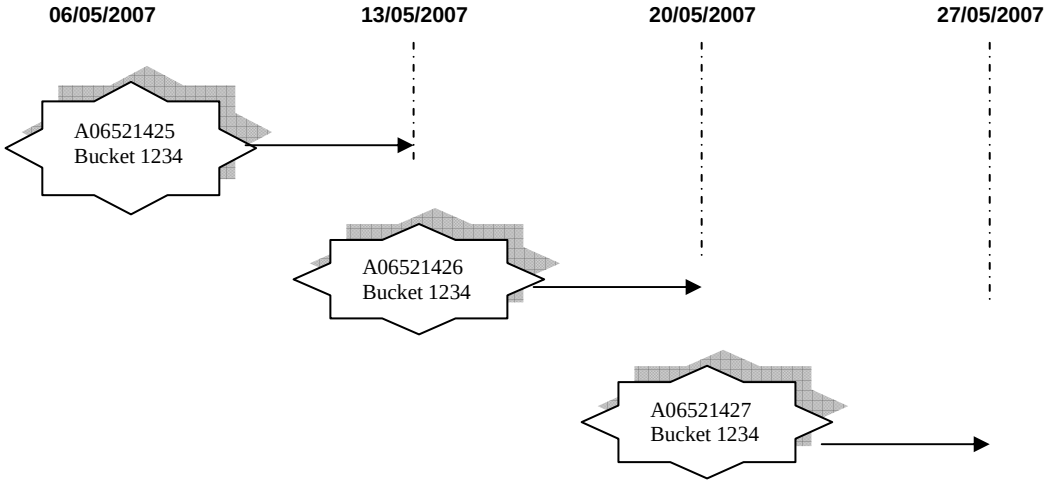


Figura 1.16: Integração no Novo Sistema

3.3.2 Alocação de Ids de Promoção

A alocação de ids de promoções no sistema anterior, mantida no sistema a desenvolver, é feita seguindo a técnica de atribuição de ids temporários, na qual um id é apenas válido durante um período determinado no tempo. Desta forma, só se consegue identificar uma

promoção tendo o id, neste contexto chamado de Bucket, e uma data na qual essa promoção está activa.



Promoção: A065214252 – 06/05/07 – 12/05/2007 Bucket 1234
Promoção: A065214253 – 13/05/07 – 19/05/2007 Bucket 1234
Promoção: A065214254 – 20/05/07 – 26/05/2007 Bucket 1234

Figura 1.17: Atribuição de Ids (Buckets)

3.3.3 Ciclo Promocional no Sistema Actual

As promoções no sistema actual seguem um ciclo de vida muito simples, sendo criadas uma vez, momento em que recebem um ID, podendo ser depois alteradas, nunca deixando de existir.

Os estados são de "Não Autorizada", "Autorizada", "Em Rectificação Mecânica", "Em Rectificação de Produto" e "Desactivada".

Os processos de integração são também muito simples, não se eliminando as promoções a partir do momento em que estas são aprovadas, podendo ser apenas desactivadas.

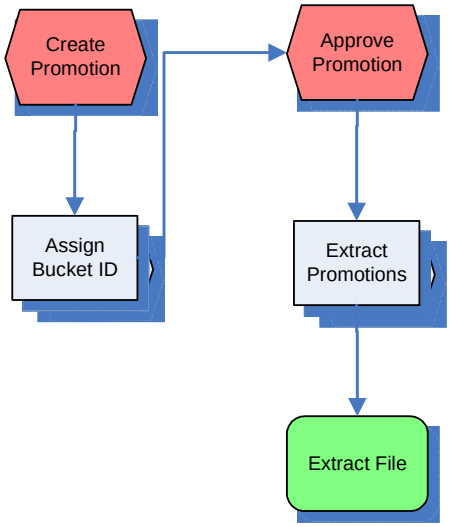


Figura 1.18: Ciclo de Vida Promocional Anterior

3.3.4 Ciclo Promocional no ORPM/ORIB

As promoções seguem um ciclo de vida especificado pelo novo sistema desenhado pelo cliente e a sua operação está distribuída por vários sistemas: legacy e novos.

Como estão implementadas, as promoções do ORPM seguem um ciclo diferente. Ambos os modelos de ciclos de vida de promoções, o do ORPM e o do sistema onde este será integrado, contêm estados de espera por aprovação, para controlar e autorizar a passagem das promoções a efectivas nas lojas.

No modelo do ORPM, as promoções começam por estar no estado de rascunho, ou de Worksheet, o único estado que permite alteração dos dados da promoção. Podem depois ser submetidas para aprovação ou aprovadas directamente dadas permissões suficientes do utilizador. Do estado de submetidas, podem ser aprovadas, remetidas para rascunho de novo ou rejeitadas. De rejeitadas podem passar de novo para submetidas, rascunho ou serem aprovadas. As promoções, uma vez aprovadas, podem passar ao estado de rascunho para rectificação de qualquer pormenor eventualmente esquecido.

Para apagar uma promoção é necessário passá-la primeiro para o estado de rascunho e só então apagá-la.

De aprovadas, na data de início das promoções, estas passam a activas nas lojas através de um processo batch. Deste estado passam para completas na data de fim, mais uma vez por batch, ou para canceladas, caso seja necessário terminar a promoção prematuramente.

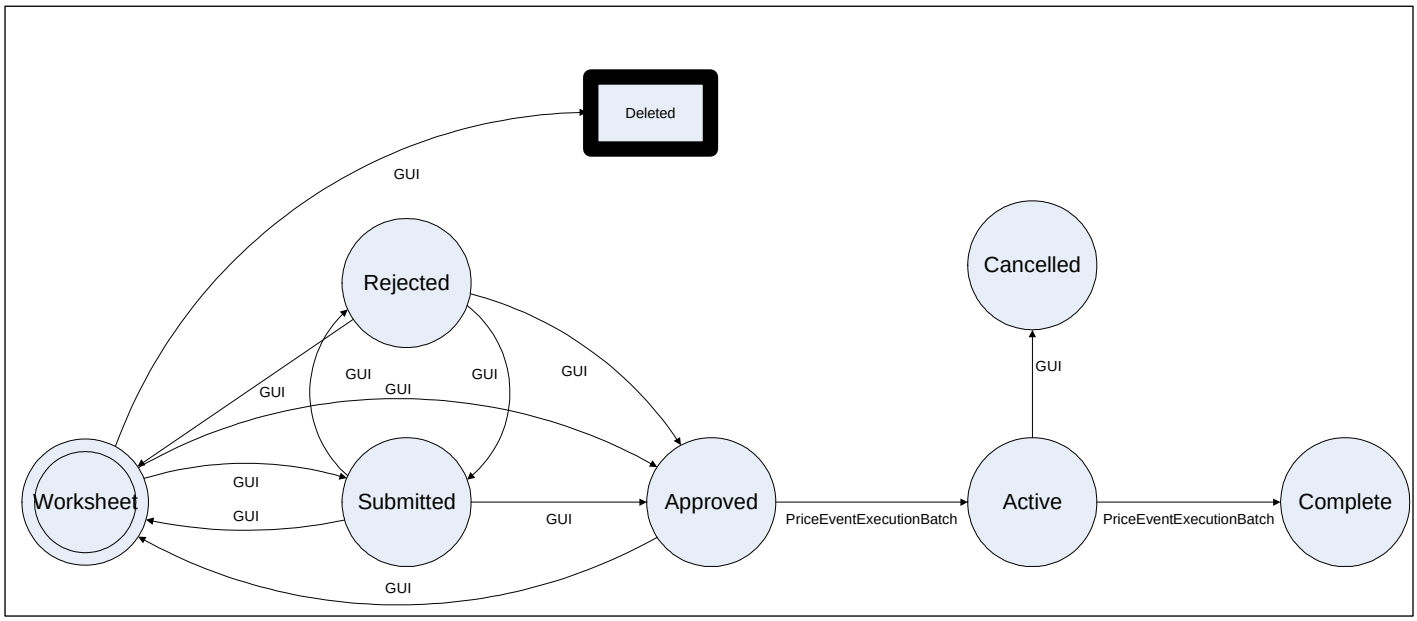


Figura 1.19: Ciclo de Vida Promocional de ORPM

No ORPM, o conceito de promoções está dividido em promoção, componente de promoção e detalhe de componente de promoção [ORPMDM]. A informação sobre o estado da promoção encontra-se ao nível dos detalhes da promoção, sendo este o elemento ao qual se pode mudar o estado.

Numa perspectiva de integração, os processos de negócio levam ao envio de mensagens pelo ORIB para informar os outros sistemas das mudanças de estado das promoções.

No momento em que as promoções passam para o estado de aprovadas, uma mensagem é enviada com uma criação de promoção do tipo PrmPrcChgCre. Quando, em algum momento, deixam o estado de aprovadas ou, caso já tenham passado a data de início, de activas, é enviada uma mensagem de delete to tipo PrmPrcChgDel.

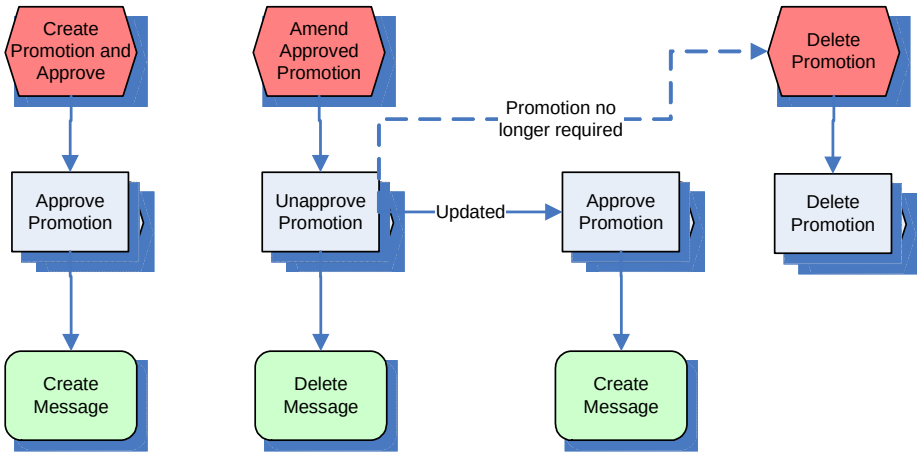


Figura 1.20: Novo Ciclo de Vida Promocional Previamente Desenhado

3.3.5 Análise de Integração

Os sistemas OR e os Legacy são conceptualmente incompatíveis no que diz respeito aos mecanismos de gestão de promoções.

Ao criar uma promoção, o ORPM envia uma mensagem de criação, e esta pode ser mapeada ao sistema Legacy. No entanto, quando é preciso fazer uma alteração da promoção, é enviada uma mensagem de eliminação de promoção seguida por uma outra de criação, a primeira em nada distinta da mensagem de criação original.

Os sistemas legacy não suportam este modelo de rectificações, precisando de distinguir o eliminar definitivo da mensagem de eliminação para rectificação.

Com o intuito de resolver a incompatibilidade, o ORPM passa a manter registo do estado da promoção do lado do OLS. Esta informação de estado é guardada numa nova propriedade do objecto de promoção do ORPM.

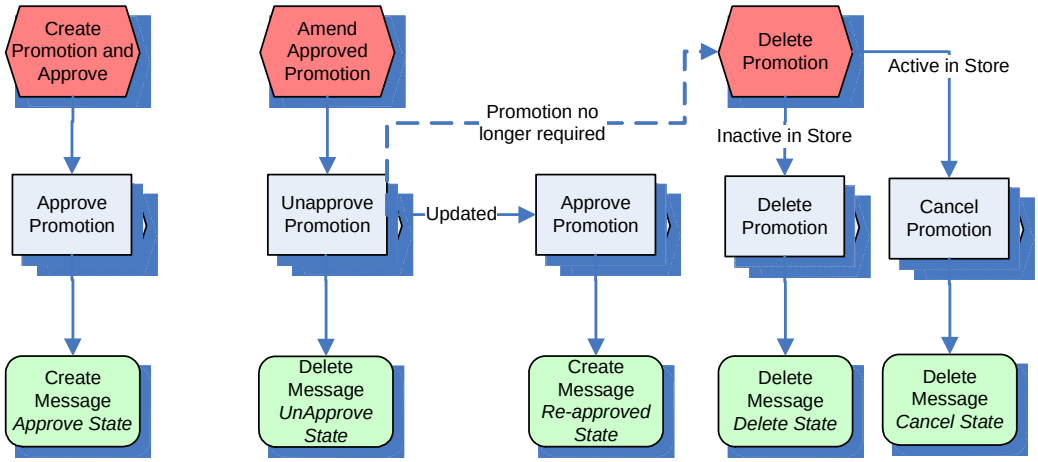


Figura 1.21: Novo Ciclo de Vida Promocional Actualizado

Os novos estados de promoção passam a ser de "Não inicializada", ou null, quando é criada, "Nova" quando é aprovada pela primeira vez, "Desaprovada" quando está em rectificação, "Reaprovada" quando é aprovada depois de rectificações e "Cancelada".

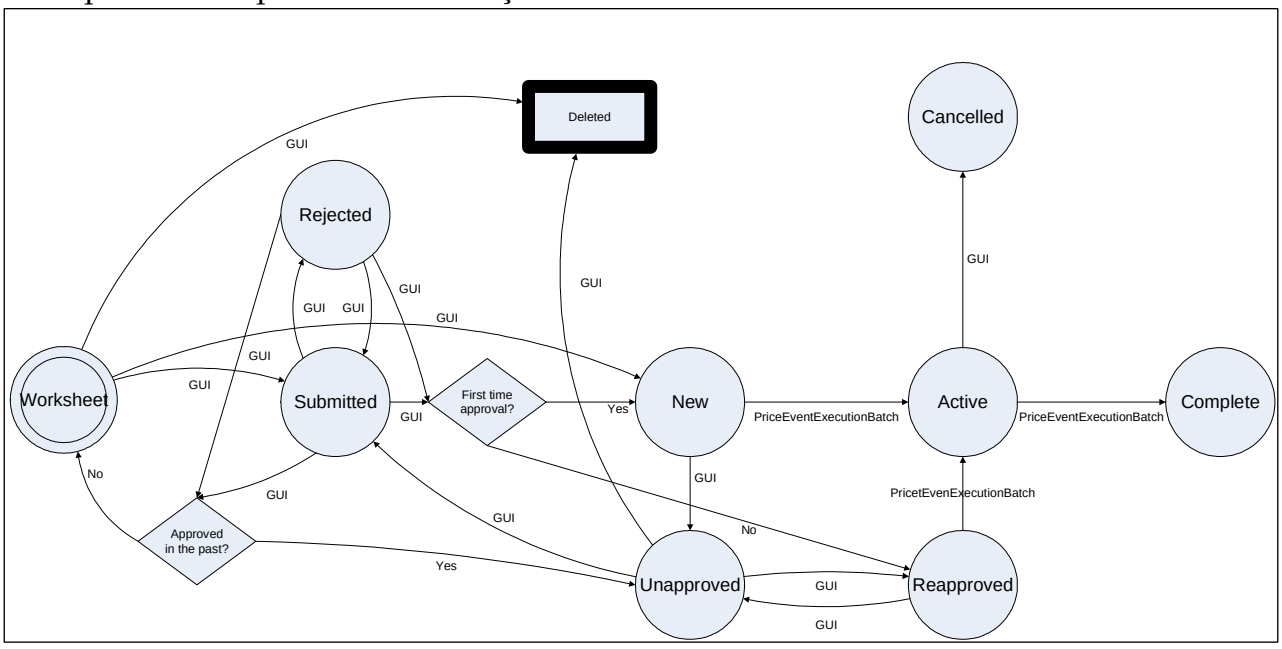


Figura 1.22: Novo Ciclo de Vida Promocional no ORPM

No respeitante ao novo modelo de integração, as mensagens terão que conter um campo que distingue as promoções novas das rectificadas, pelo que este novo campo de estado está presente nas novas especificações de mensagens.

Por outro lado, o ORPM não envia mensagens no momento de eliminação de promoções. O processo antigo não precisava deste envio porque partia do pressuposto de que o sistema operacional tinha já

recebido uma mensagem de eliminação na passagem para Worksheet e por isso não retinha nenhuma informação sobre a promoção.

Para o OLS apagar esta promoção, um novo envio de mensagens de eliminação teve que ser integrado na aplicação.

Assim sendo, no momento de eliminação final de promoções, uma mensagem de eliminação com o novo campo de estado a *deleted* deve ser enviada, distinta das mensagens de eliminação para rectificação, que contêm *unapproved* no novo campo.

Paralelamente a estes processos, as mensagens rejeitadas, após um período pré-estipulado, são apagadas do sistema por uma aplicação batch de purga de promoções. Estas eliminações, no caso de existência de informação no OLS sobre a promoção, i.e., no caso de ser uma promoção que já foi aprovada no passado, precisam de ser enviadas (comportamento até agora não implementado).

Tabela 1.1: Mudanças de Estados e Publicações Consequentes

Estado	OLS	Acção	Novo Estado	Novo OLS	Tipo de Public.	Entidade
Worksheet	Null	Approvar	Approved	New	PrmPrcChgCre	GUI
Worksheet	Unapproved	Approvar	Approved	Reapproved	PrmPrcChgCre	GUI
Worksheet	Null/Unapproved	Submeter	Submitted	< Sem Alter.>	<N/A>	GUI
Worksheet	Null	**Eliminar**	<Sem Alter.>	< Sem Alter.>	<N/A>	GUI
Worksheet	Unapproved	**Eliminar**	< Sem Alter.>	< Sem Alter.>	PrmPrcChgDel*	GUI
Submitted	Null	Approvar	Approved	New	PrmPrcChgCre	GUI
Submitted	Unapproved	Approvar	Approved	Reapproved	PrmPrcChgCre	GUI
Submitted	Null/Unapproved	Worksheet	Worksheet	< Sem Alter.>	<N/A>	GUI
Submitted	Null/Unapproved	Rejeitar	Rejected	<Sem Alter.>	<N/A>	BATCH
Rejected	Null	**Eliminar**	< Sem Alter.>	< Sem Alter.>	<N/A>	BATCH
Rejected	Unapproved	**Eliminar**	< Sem Alter.>	< Sem Alter.>	PrmPrcChgDel*	BATCH
Rejected	Null/Unapproved	Worksheet	Worksheet	< Sem Alter.>	<N/A>	GUI
Rejected	Null	Approvar	Approved	New	PrmPrcChgCre	GUI
Rejected	Unapproved	Approvar	Approved	Reapproved	PrmPrcChgCre	GUI
Approved	New/Reapproved	Worksheet	Worksheet	Unapproved	PrmPrcChgDel	GUI
Approved	New/Reapproved	StartDate	Active	< Sem Alter.>	<N/A>	BATCH
Active	New/Reapproved	EndDate	Complete	< Sem Alter.>	<N/A>	BATCH
Active	New/Reapproved	Cancelar	Cancelled	Cancelled	PrmPrcChgDel	GUI
Complete	New/Reapproved	**Eliminar**	< Sem Alter.>	< Sem Alter.>	<N/A>	BATCH
Cancelled	Cancelled	**Eliminar**	< Sem Alter.>	< Sem Alter.>	<N/A>	BATCH

Nota: * A Publicação deve enviar um "deleted" no novo campo de estado da mensagem, mesmo não representando este o novo estado da promoção.

O sistema desenhado para resolver a integração do ORPM com o OLS, no que respeita o ORIB, teve que conter os novos fluxos de mensagens, especificações das mesmas, tal como os mecanismos de publicação, subscrição e de criação de payloads alterados.

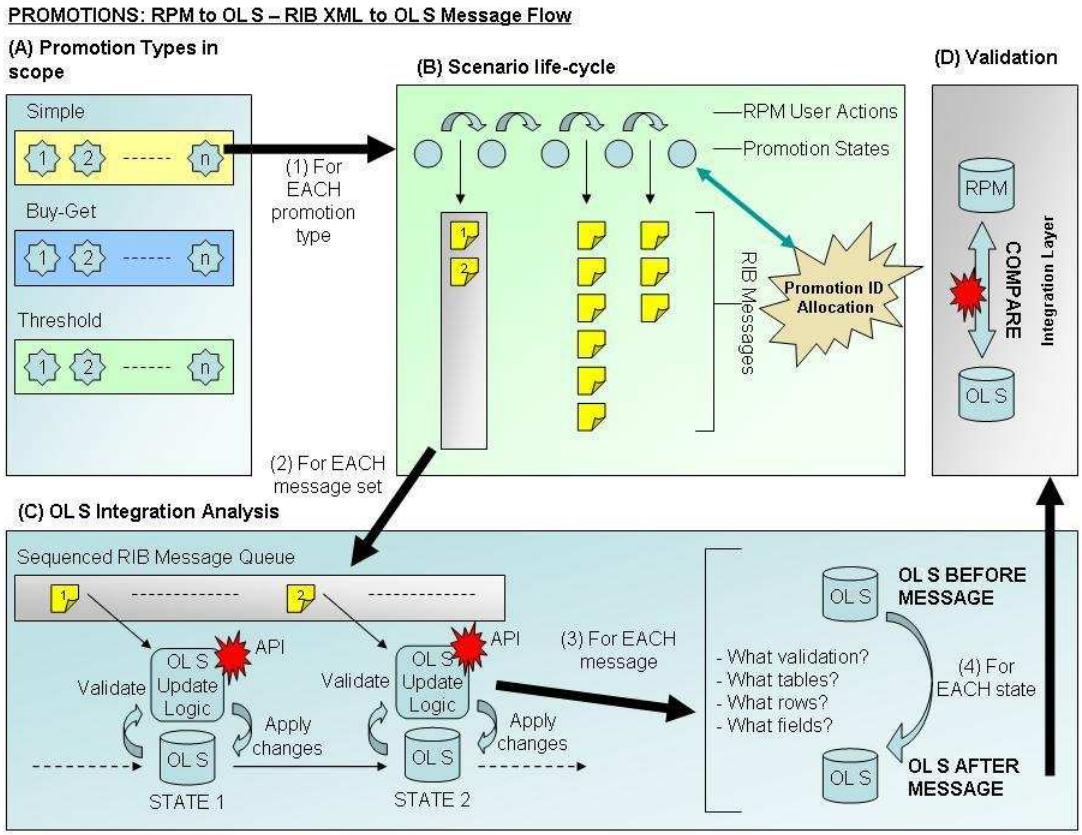


Figura 1.23: Fluxo de Informação Promocional

3.3.6 Implementação On-line

O fluxo normal de publicação de mensagens teve que ser alterado para que o sistema implementasse as novas especificações de negócio. As alterações necessárias para as novas publicações foram desde a camada de serviços de núcleo de negócio até ao adaptador da publicação no ORIB, passando pela camada de DB.

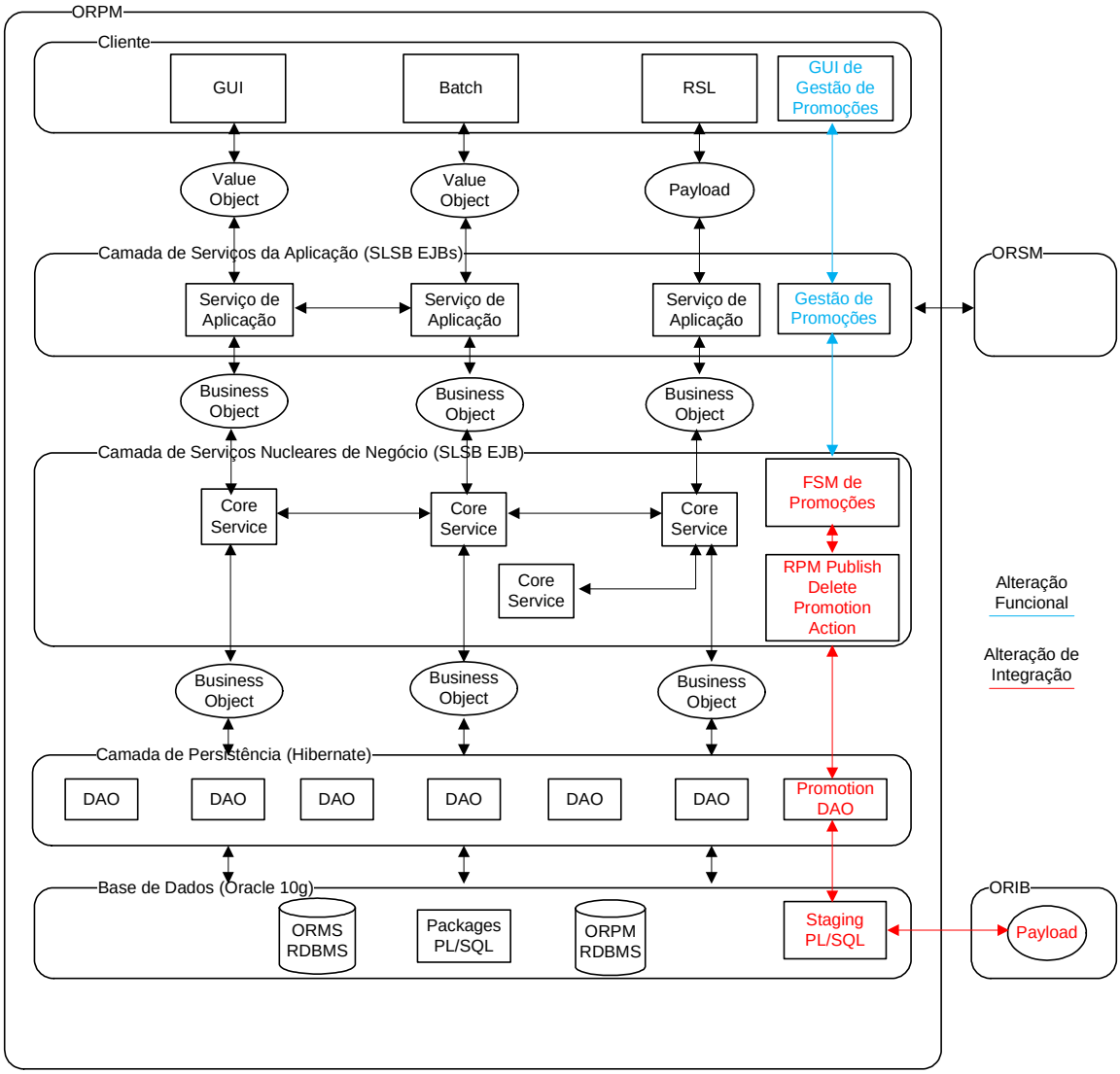


Figura 1.24: Alterações ao ORPM por Camadas

O ponto de entrada nos processos de publicação da mudança de estado promocional foi introduzido na máquina de estados finita das promoções. Esta encontra-se na camada da lógica de negócio e chama uma nova acção.

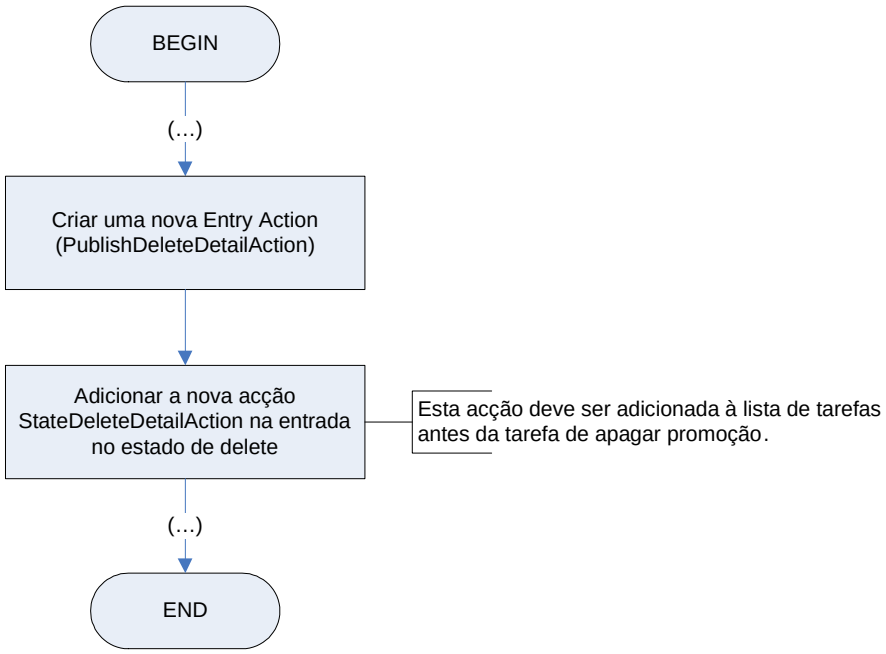


Figura 1.25: Ponto de Entrada para Modificação

Foi adicionada uma nova acção de publicação no momento de alteração de estado.

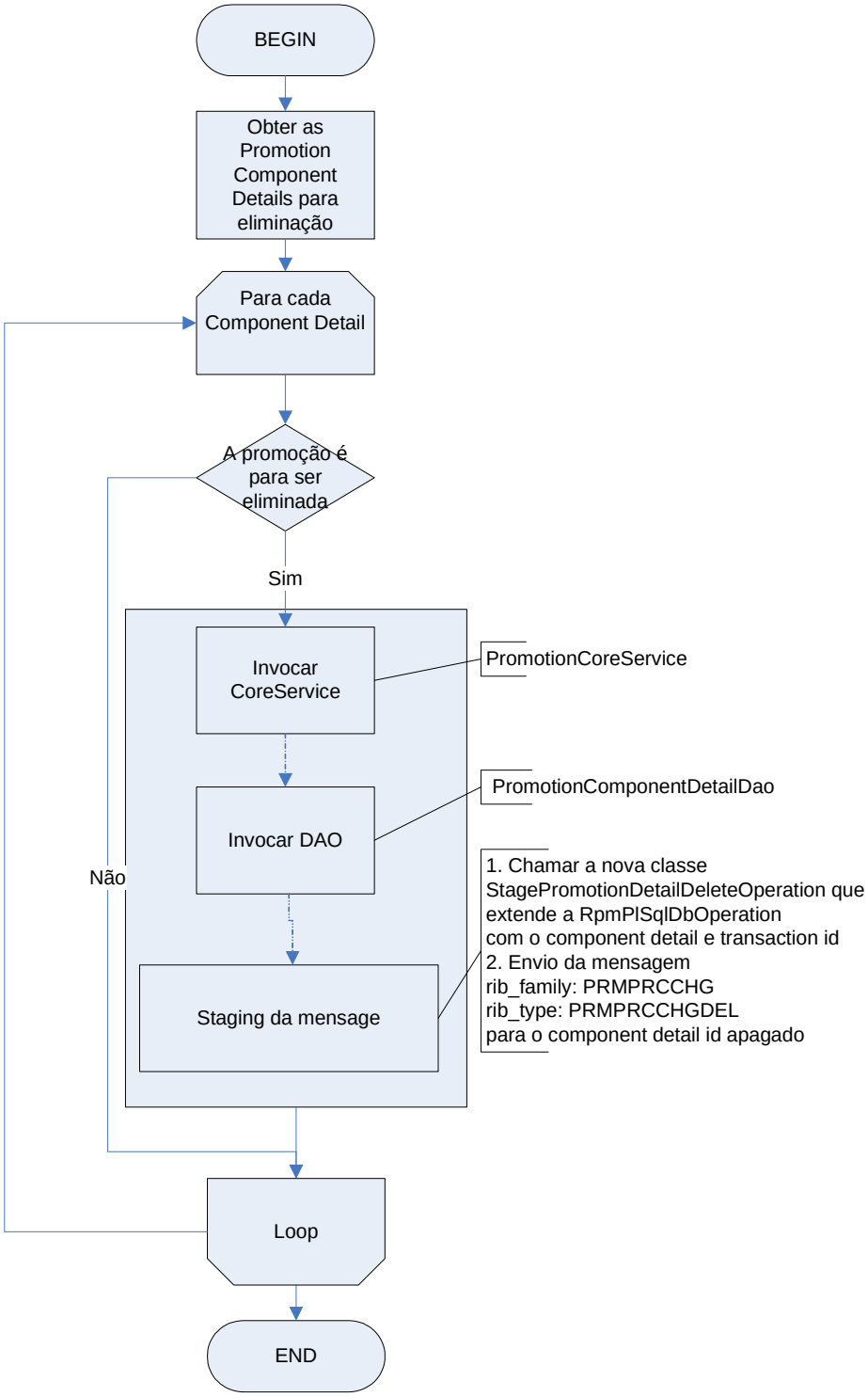


Figura 1.26: Nova Acção no FDS Promocional

A nova acção efectua um pedido à camada de persistência.

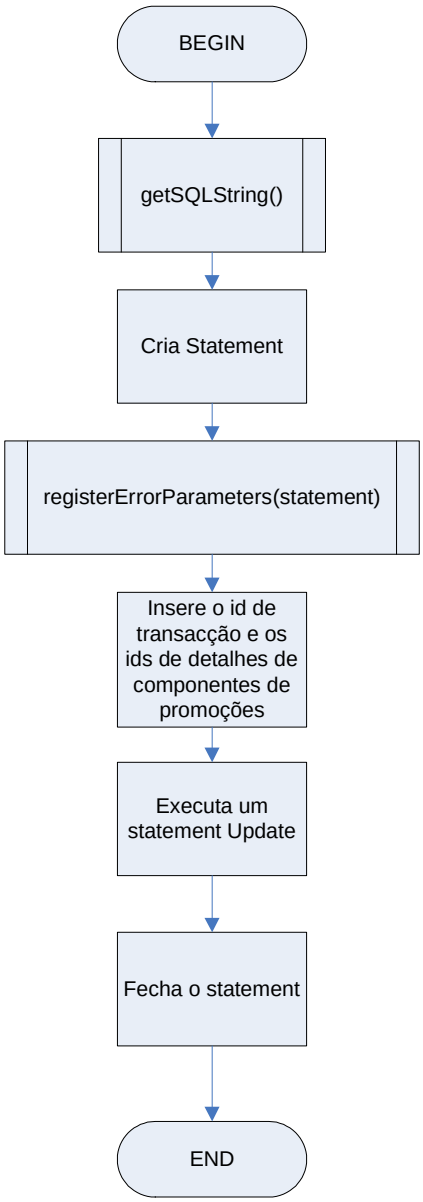


Figura 1.27: DAO da Camada de Persistência

Esta, por sua vez, chama uma package da DB responsável pela inserção da informação na tabela de staging da payload. Esta package tem que reunir toda a informação necessária para a criação da mensagem a partir das referências recebidas da camada de persistência, utilizando para isso um cursor.

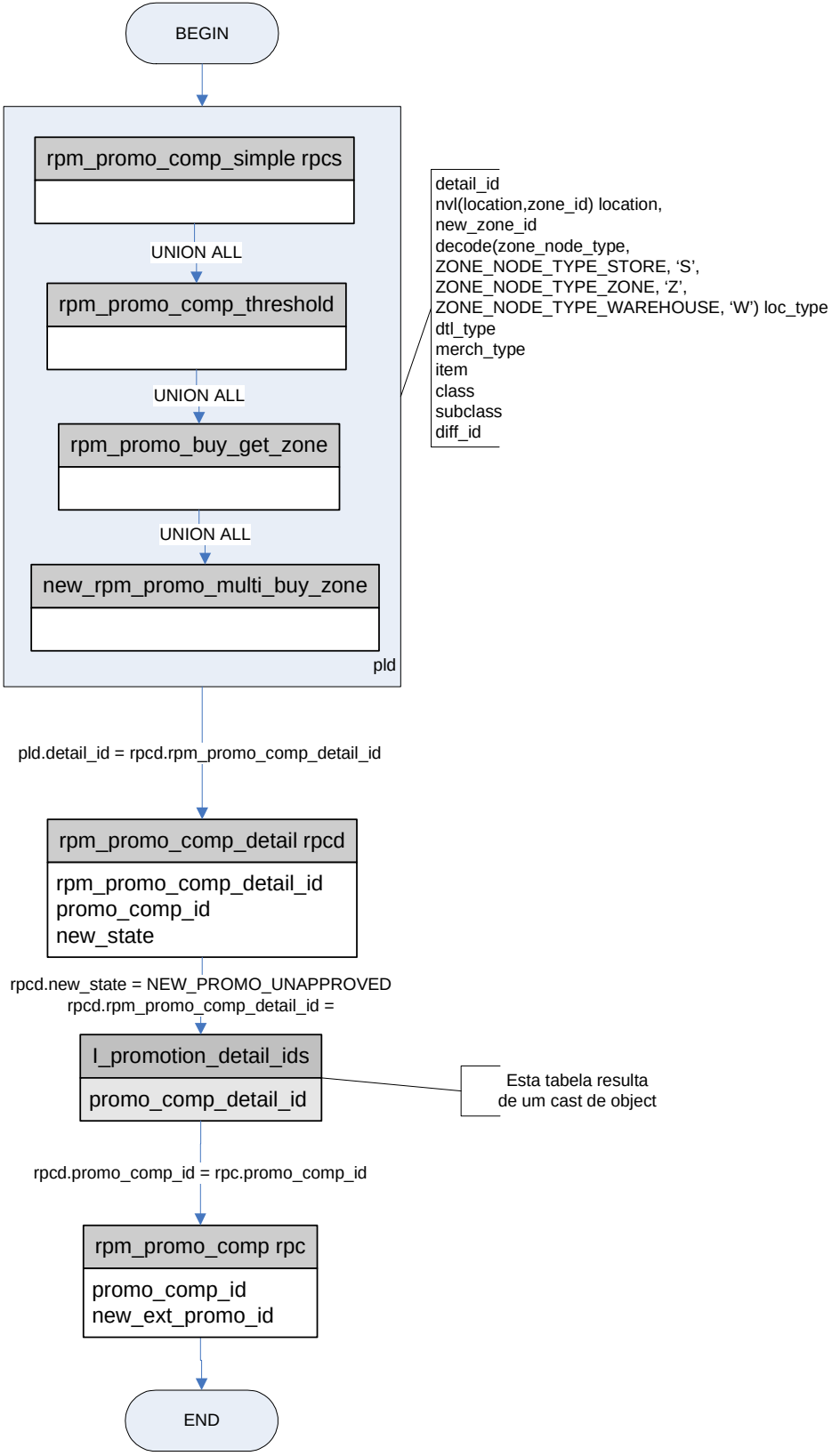


Figura 1.28: Cursor de Dados de Promoções

Depois de reunida a informação, a função trata de inserir a informação das promoções a eliminar na tabela de payload.

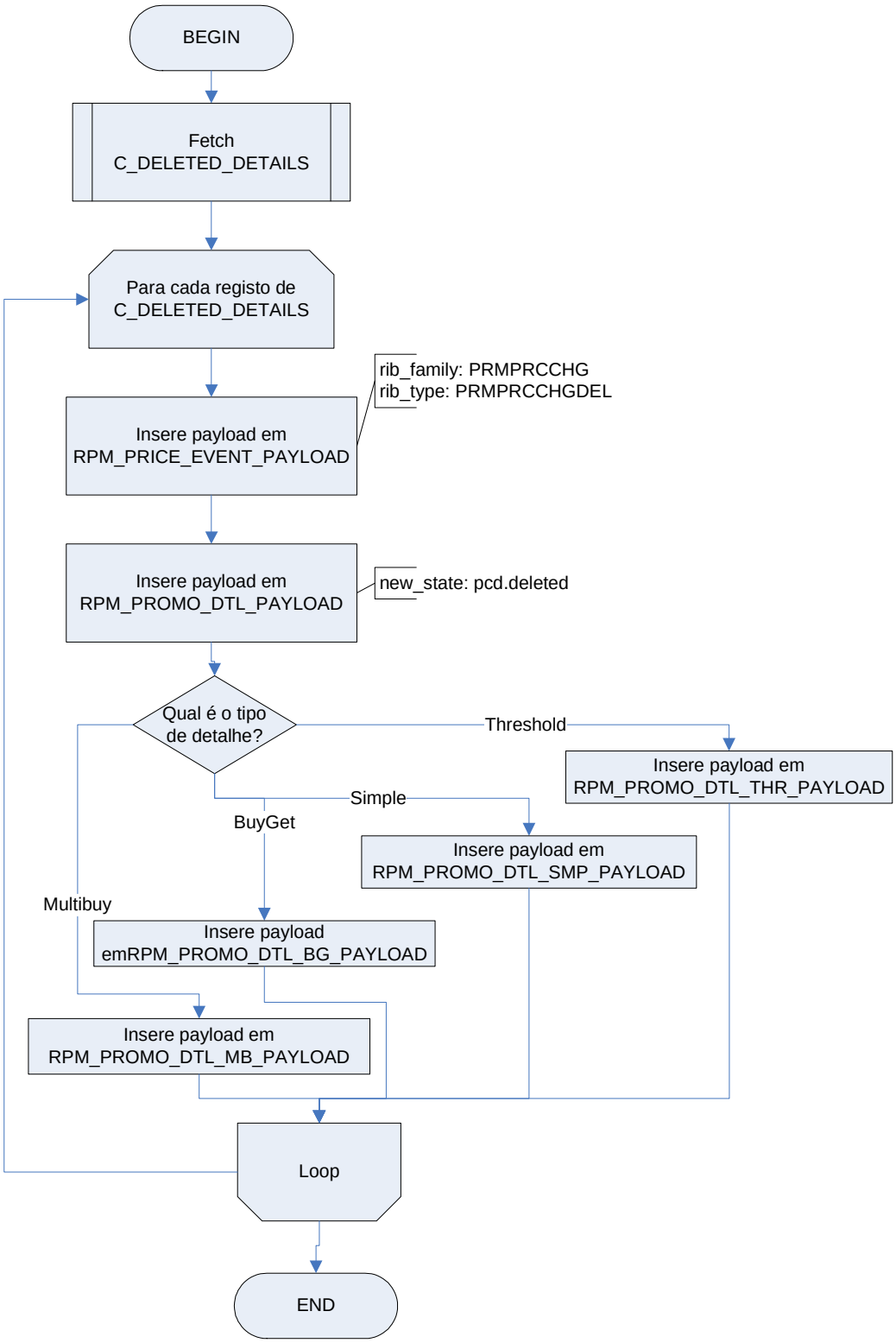


Figura 1.29: Inserção nas Tabelas de Staging

3.3.7 Implementação no Batch de Purga

No caso do batch de purga de promoções, a classe que implementa as eliminações teve que ser também alterada para invocar uma nova operação de persistência. No momento de chamada, a aplicação batch já

escolheu as promoções a ser purgadas, cujos ids se encontram numa tabela intermédia chamada de Global Temporary Table (GTT).

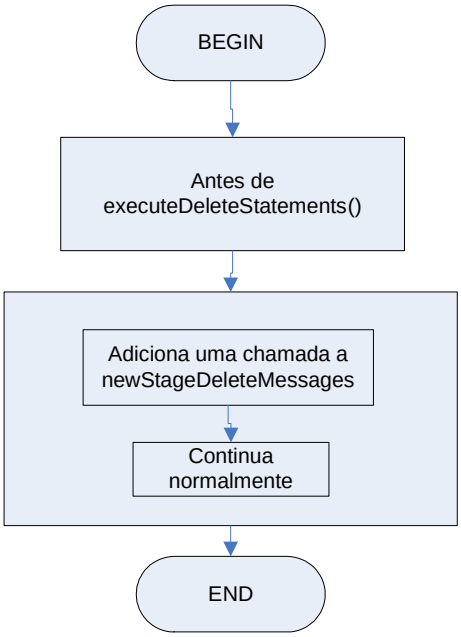


Figura 1.30: Ponto de Entrada da Modificação ao ORPM

A nova operação de persistência invoca uma chamada de PL/SQL e, caso não tenha havido nenhuma publicação, anula a transacção de ORIB.

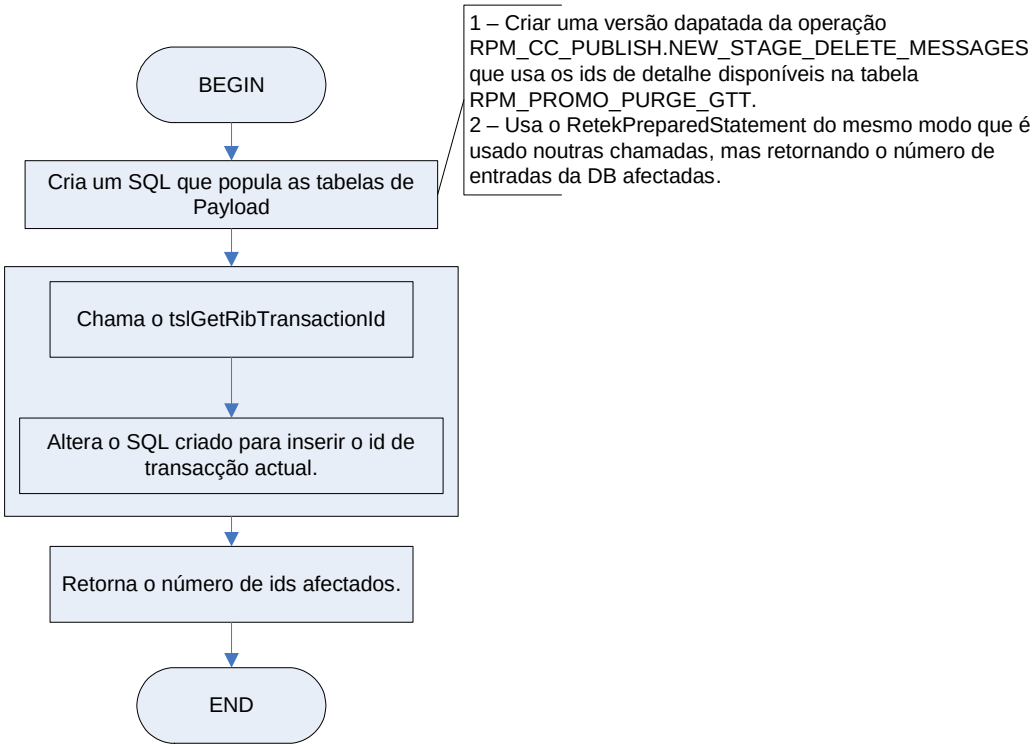


Figura 1.31: Operação de Persistência

O procedimento de verificação é seguido para não levar o sistema a uma operação ilegal. A inserção na tabela de payload precisa de um id de

transacção. A chamada ao RibTransactionManager que retorna esse id inicializa essa transacção, o que leva a uma tentativa de publicação na finalização do código do RibTransactionManager. Em alguns casos, a chamada PL/SQL não publica nenhuma eliminação, nomeadamente quando as promoções purgadas não têm o novo estado inicializado no OSL. Como o RibTransactionManager não está preparado para ser inicializado sem uma publicação no fim (não há validação da existência de uma payload ao executar uma chamada a next), uma excepção de Java é levantada.

Para resolver esta limitação arquitectural do ORIB, a transacção é cancelada e o RibTransactionManager é alterado para verificar se a transacção foi cancelada antes de chamar os métodos de publicação.

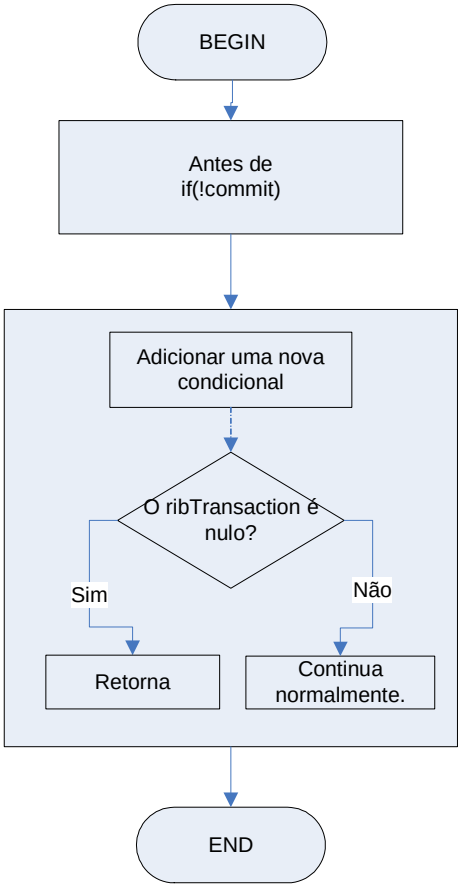


Figura 1.32: Alteração ao RibTransactionManager

A chamada de PL/SQL tem um código semelhante ao da package de publicação online, com a grande distinção de apenas publicar as mensagens no estado de rejeitadas e ir buscar os ids das promoções com eliminação elegível para publicação à GTT, previamente preenchida pelo código de purga. Tal como nos outros processos do batch de purga de promoções, a chamada envia os comandos PL/SQL para a base de dados para lá serem corridos, em vez de chamar uma função armazenada na DB.

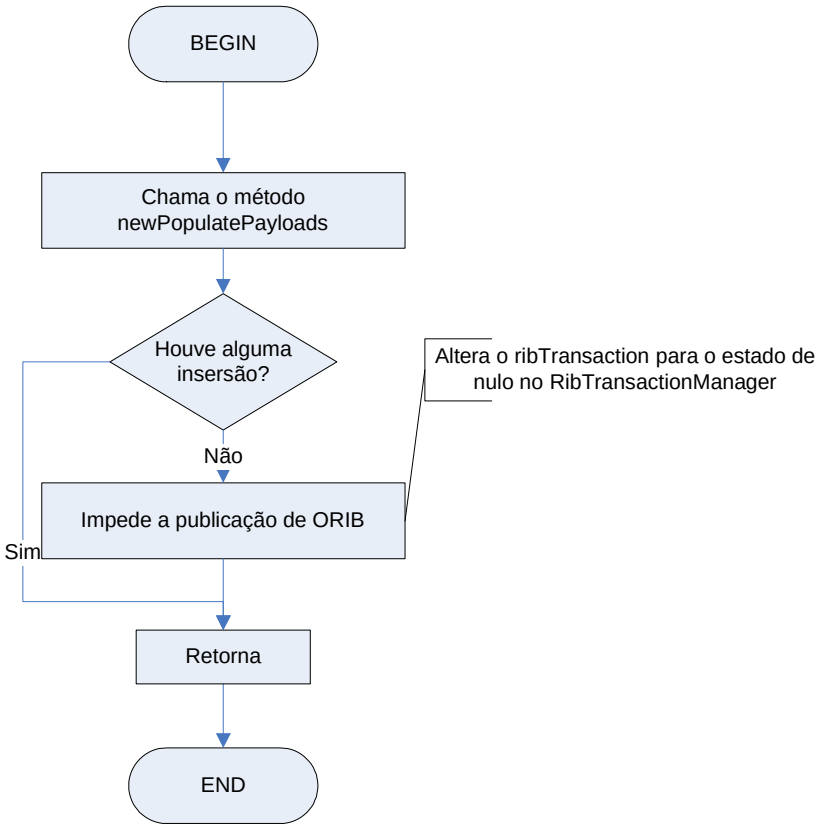


Figura 1.33: Invocação da Camada de DB

3.3.8 Implementação nos Interfaces ORIB

Para que a publicação se faça, é ainda necessário que os adaptadores de ORIB estejam preparados para introduzir o novo campo de estado nas mensagens enviadas. Tal é conseguido alterando a classe responsável por inserir os valores de payload em objectos que representam a mensagem.

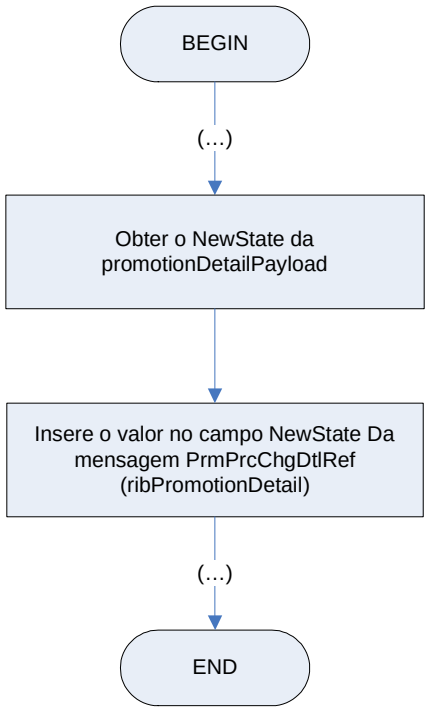


Figura 1.34: Integração da Informação na Classe Java de Payload

Seguidamente, o próprio objecto de payload precisa de ser criado. Este é um mapeamento directo do XML de especificação XSD, específico do tipo de mensagens a enviar. Bastou neste caso adicionar um campo do tipo String chamado new_state à especificação. Depois deste passo, os objectos de payload precisaram de ser refeitos para reflectir a alteração de especificação, senão o parsing interno das mensagens resultaria em erro.

A versão compilada destas classes deve ser depois adicionada aos jars de payloads, presentes na aplicação ORPM e ORIB.

3.3.9 Testes Unitários

Foram por fim efectuados testes unitários, quer nas instalações principais de desenvolvimento, quer no centro de desenvolvimento do cliente na Índia, onde foi feita uma formação sobre a posterior utilização dos conhecimentos adquiridos nesta parte do projecto

Tabela 1.2: Resumo dos Testes Unitários Efectuados

Program Unit	Delete Promotions	
Pre Conditions	**Test Description**	**Expected Behaviour**
1. Created/updated payloads and ORPM classes deployed. 2. Promotion Event created in ORPM.	Change a Promotion Component Component Detail state from Worksheet to Approved in ORPM. This should trigger the publication of a message to the	The PrmPrcChgCre Message in the etPrmPrcChgFromRPM topic should contain all the promotion event structure and the field *new_state* is presented and contains the Approved value.

	etPrmPrcChgFromRPM topic.	
1. Created/updated payloads and ORPM classes deployed; 2. Promotion Event created in ORPM.	Delete a Promotion Component Detail in ORPM. This should trigger the publication of a message to the etPrmPrcChgFromRPM topic.	The PrmPrcChgDel Message in the etPrmPrcChgFromRPM topic should contain all the promotion event structure of the deleted Promotion Event and the new field *new_state is presented and contains the Deleted value.*
1. Created/updated payloads and ORPM classes deployed; 2. Promotion Event created in ORPM.	Delete a Promotion Component Detail in ORPM. This should trigger the publication of a message to the etPrmPrcChgFromRPM topic.	The PrmPrcChgDel Message in the etPrmPrcChgFromRPM topic should contain all the promotion event structure of the deleted Promotion Event and the new field *new_state is presented and contains the Deleted value.*
1. Created/updated payloads and ORPM classes deployed. 2. Promotion Event created in ORPM.	Create a Promotion with the same start date as the vdate. Create a Component Detail for it and then approve it. It should become active. Cancel the detail. This should trigger the publication of a message to the etPrmPrcChgFromRPM topic.	The PrmPrcChgCre Message in the etPrmPrcChgFromRPM topic should contain all the promotion event structure and the field *new_state* is presented and contains the Cancelled value.
1. Promotion Purge Batch properly configured to run. 2. A Detail should exist in ORPM that presents the necessary conditions to be purged and is in the new_state of pcd.unapproved.	Delete a rejected Promotion Components Detail from the Promotion Purge Batch. This should trigger the publication of a message to the etPrmPrcChgFromRPM topic.	1. Verify that the promotion details were deleted in agreement with the system parameter that defines the number of months after which a completed promotion is purged. Remember that this parameter is also used to purge cancelled and rejected promotions. 2. The PrmPrcChdDel Message in the etPrmPrcChgFromRPM topic should contain all the promotion event structure of the deleted Promotion Event and the new field *new_state is presented and contains the Deleted value.*

.

3.4 Testes de Performance ao Módulo de ORIB

Após os testes de integração e de sistema, foram aplicados ao sistema do cliente testes de performance e o ORIB entrou no âmbito desta fase de testes.

3.4.1 Análise

Tentou-se que o ambiente de testes de performance fosse o mais parecido possível com o de produção, dado que o objectivo final dos testes é avaliar a capacidade do sistema atingir as performances alvo num ambiente de produção.

Embora idealmente indesejado, é aceitável que o ambiente de testes seja pior do que o sistema de produção, desde que se consiga chegar aos alvos de performance. Caso contrário, é necessário o aumento dos recursos do ambiente para verificar se o mesmo sucederia no ambiente de produção.

Tabela 1.3: Comparação Entre Ambientes de Produção e Performance

Servidor de Testes	**Agrupamento**	**# CPUs (Teste)**	**# CPUs (Produção)**	**Memória em Gb (Teste)**	**Memória em Gb (Produção)**
1	Database Tier RPM DB RMS Batch ORSA	19	32	76	330
2	RMS Application Tier RMS ORSA Configuration	1	4	8	48
3	RPM Application Tier RPM RSL? RSM? RIB4RPM	2	10	8	96
4	Integration Tier RIB JMS RIB4RMS	8	17	32	184

Nas fases anteriores de testes foram já identificados e eleitos para análise os interfaces mais críticos deste módulo. Compreensivamente, pertencem ao sistema central do OR, o ORMS, e são de recepção de dados ou interfaces In-Bound.

Para cada um dos interfaces a testar, há um determinado volume de mensagens que teria que ser processado num certo intervalo de tempo para atingir os requisitos mínimos de performance esperados do sistema.

Tabela 1.4: Tempos e Volumes Alvo Por Interface

Família Tópico	Designação do Interface	Número de registos Máximo na DB	Tempo para Processamento (em Horas)
asnout	idorms09	1000000	1
invadjust	idorms12	40000	2
invadjust	idorms03	300000	2
invadjust	idorms03+12	340000	2
xorder	idorms06	300000	2
xorder	idorms11	300000	2
xorder	idorms11+06	600000	2
receiving	idorms05	250000	2
receiving	idorms07	200000	2
receiving	idorms07+05	450000	2

No momento em que estes testes de performance foram efectuados, a migração de dados, por limitações de desenvolvimento de outras equipas do projecto, ainda estava a decorrer. Inicialmente, apenas dados do Cut-Over (ver 3.2.1) foram disponibilizados. Depois de analisados, estes dados foram considerados atípicos por terem uma natureza de migração, i.e., de passagem de dados consecutivos de uma DB para a outra, e uma distribuição de tempos de processamento não representativa dos dados que seriam corridos no sistema em produção.

Após o início do projecto foram disponibilizadas, para alguns dos interfaces, mensagens de processamento real, desta forma representando melhor as mensagens esperadas nos interfaces. No entanto, mesmo em alguns dos interfaces disponibilizados, a amostra de mensagens não era suficientemente grande para atingir os volumes mínimos de processamento.

Após análise da distribuição dos detalhes das mensagens, concluiu-se que estas têm tipicamente um tempo de processamento previsível e, em grande número, semelhante, podendo ser repetidas sem estragar a distribuição de tempos de processamento.

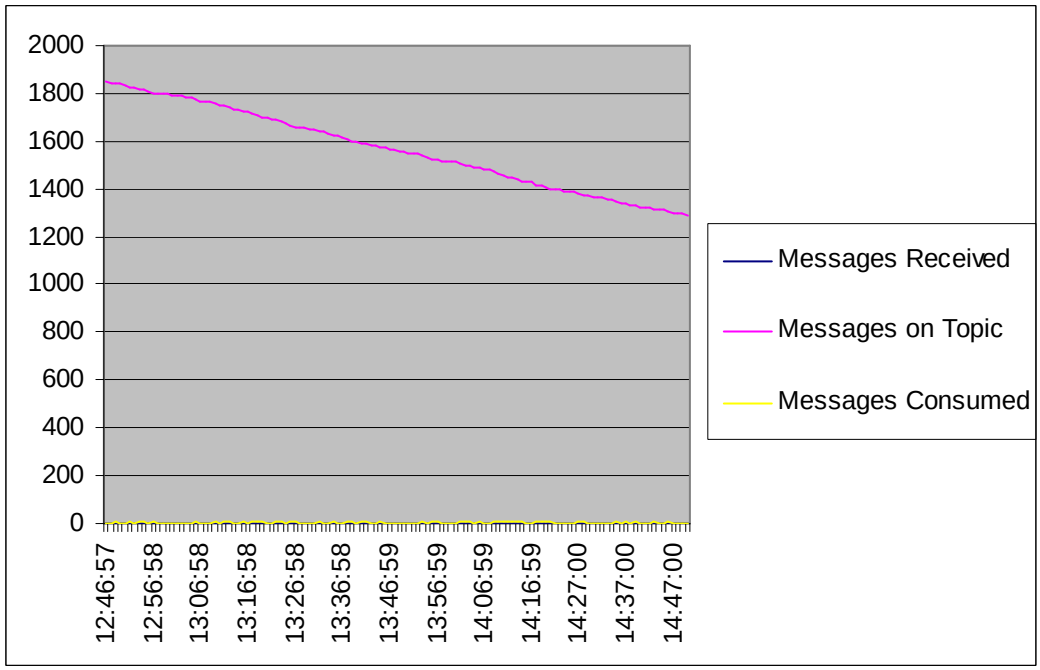


Figura 1.35: Tempos de Processamento de Mensagens

Porém, o ORIB garante a transmissão de cada mensagem uma vez e não mais, i.e., qualquer mensagem repetida irá parar ao hospital. No entanto, é possível alterar as mensagens de modo a tornar as mensagens operacionais (passando a ser consideradas como novas pelo ORMS) e representativos das mensagens originais, i.e., que levam às mesmas operações da aplicação subscritora, o ORMS.

Esta abordagem, no entanto, não é facilmente praticável nas interfaces de Order e de Receipt, em que a primeira contém dependências entre as mensagens e a segunda depende da primeira, contendo referências a Orders já existentes (uma mensagem de recibo de entrega não pode ser recebida antes de uma mensagem de criação de envio).

No primeiro caso avaliou-se a possibilidade de utilizar dados de Cut-Over para realizar os testes de performance. Dado que estes são fundamentalmente criações de registos na base de dados, operação conhecida por ser morosa, considerou-se que se estava a mudar para um cenário pior, assumindo assim que resultados aceitáveis com estas mensagens garantiriam uma performance ainda melhor com dados live.

No segundo caso, porém, decidiu-se esperar que a integração de dados de Order fosse feita para proceder aos testes, tendo sido deixadas as condições necessárias para proceder ao mesmo após o término do tempo do projecto.

Por fim, o facto dos testes de integração do sistema terem falhado na detecção de erros, algumas interfaces estavam a obter demasiados erros. Este facto, aliado a uma performance significativamente pior no

tratamento de mensagens em erro, leva à incapacidade do sistema de se comportar como esperado num ambiente de produção. Depois de algumas correcções, concluiu-se que não havia possibilidade de, dentro do tempo limite do projecto, eliminar todos os erros.

Porém, os erros encontrados indicavam que as mensagens que iriam parar ao hospital eram típicas e não inseriam dependências. Alguns dos erros eram apenas derivados da falta de configuração do sistema para reconhecer caracteres internacionalizados, tal como vários outros erros que, caso fossem processados correctamente, não influenciariam a performance do sistema. Posto isto, a sua remoção não alteraria os testes de performance significativamente.

Tabela 1.5: Influência de Erros nas Amostras

Família / Tópico	Designação do interface	Mínimo de mensagens	Máximo de mensagens	Tempo limite de processamento em Horas	Qt. de erros	Número mínimo de erros	Número máximo de erros	Tempo mínimo em erros: horas (25s/msg)	Tempo máximo em erros: horas (25s/msg)
asnout	idorms09	1000	3333	1	0.13	**125**	**416.625**	**0.69**	**2.31**
invadjust	idorms03+12	2900	5800	2	0.11	**319**	**638**	**1.77**	**3.54**
xorder	idorms11+06	10500	165000	2	0.40	**4200**	**66000**	**23.33**	**366.67**

Adicionalmente, é por vezes útil apagar, por diversos motivos, as mensagens presentes nos tópicos JMS, como por exemplo, para ter a oportunidade de adiar o processamento de mensagens não alteradas já presentes no tópico JMS, retirando-as deste.

Na eventualidade do sistema ser incapaz de atingir os tempos esperados para os testes e tendo em conta que nenhuma mudança arquitectural é permitida na fase de testes de performance, alguns ajustes ao ORIB podem ser realizados juntamente com activação dos mecanismos de processamento em paralelo, chamados de multi-threading, de ORIB.

Uma vez conhecidos os resultados de performance, uma decisão sobre o número de threads deverá ser aplicado a cada interface. Tenta-se manter estes valores no mínimo possível para atingir os tempos-alvo, dado que o multi-threading, embora acelere o processamento, obriga a uma utilização extra dos recursos, o que, numa aplicação excessiva desta técnica, piora as performances.

A utilização de multi-threads leva, no entanto, a um problema de recursos, dado que estes correm em transacções separadas e que muitas vezes reservam o acesso a campos da DB (locking), não permitindo que outras threads acedam a estes. O ORIB está preparado para esta eventualidade ao enviar a mensagem para o hospital e retentando-a, mas os locks resultantes são excessivos, pois muitas mensagens consecutivas acedem aos mesmos campos.

Uma vez que as mensagens do hospital pioram significativamente os tempos da aplicação, um mecanismo auxiliar tem que levar a que a mesma thread se encarregue de processar as mensagens que acedam ao mesmo campo, mantendo uma relativa distribuição de tarefas pelas threads existentes.

Para além disto, o ORIB tem que garantir que a ordem de entrega de mensagens relacionadas é sempre a mesma, pelo que nestes casos as mensagens referentes ao mesmo objecto de negócio são todas enviadas para o mesmo thread.

Por fim, dado que cada teste chega a demorar horas a terminar, a aplicação dos testes é uma tarefa em que um erro leva a desperdícios significativos de tempo e recursos. Para compensar este problema, um mecanismo de execução automática de testes pode ser desenvolvido de modo a deixar a correr testes fora das horas de actividade da equipa, durante a noite.

3.4.2 Desenvolvimento de uma Framework de Testes

Para proceder à implementação dos processos definidos na análise do problema, foram desenhadas aplicações capazes de satisfazer os requisitos da solução pretendida.

Para começar, foi necessário arranjar uma forma de controlar o fluxo de mensagens a sair dos tópicos JMS. Este fluxo poderia ser iniciado e interrompido ligando e desligando os adaptadores que consomem mensagens dos tópicos, os eWays.

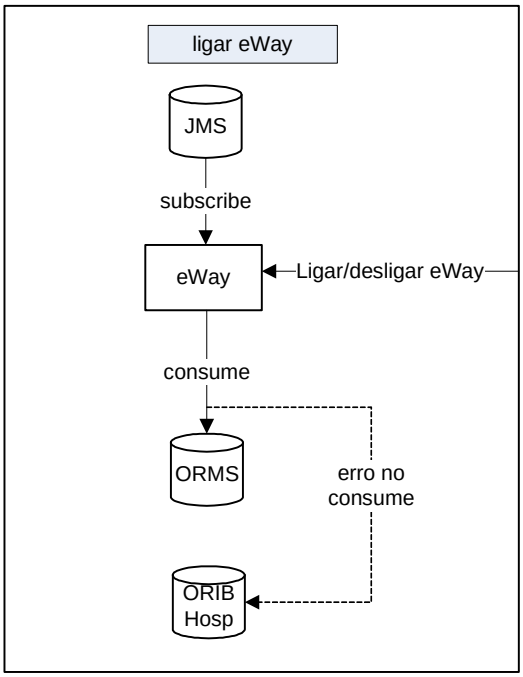


Figura 1.36: Ferramenta de Controlo do eWay

Para interceptar as mensagens dirigidas ao ORMS, pretende-se que uma aplicação subscreva ao tópico de JMS e grave as mensagens como ficheiros do sistema operativo, em formato xml, processo chamado de Dump. Este processo, sendo demorado, é paralelizado por pseudo-threads, sendo dividida por vários processos a responsabilidade de recolher as mensagens num intervalo das mensagens.

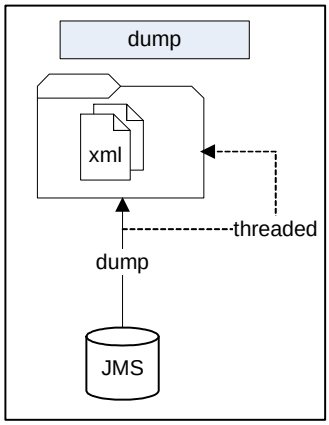


Figura 1.37: Ferramenta de Dump

Seguidamente, antes da republicação de cada mensagem, é necessário alterar os campos que colidiriam com ids de mensagens já presentes no ORMS, passando a mensagem a ser aceitável para este. Um identificador único é utilizado, constituído pela data do sistema e por um id do teste efectuado. Adicionalmente, o número da thread para que a mensagem será enviada é imposto, podendo ser resultado de um algoritmo de hashing de acordo com um campo da mensagem.

A republicação é feita criando um publicador de mensagens no tópico JMS e enviando as mensagens em ciclo até atingir os volumes de dados pretendidos.

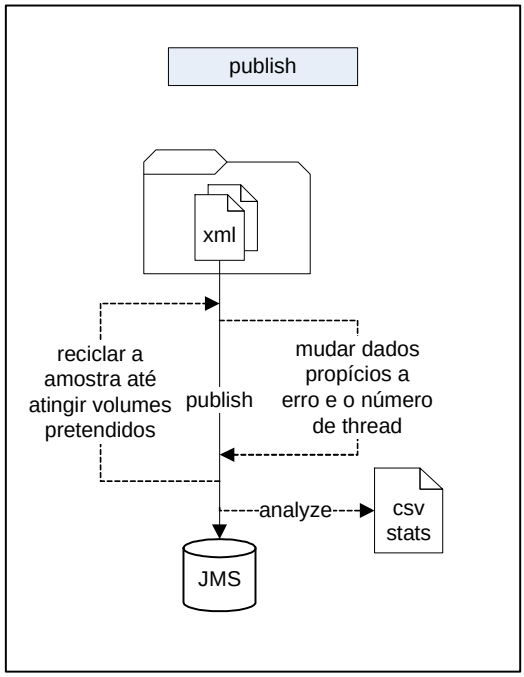


Figura 1.38: Ferramenta de Publicação

No entanto, com o intuito de decidir qual o campo a usar para o hash especificador de thread, por exemplo, ou para ter a noção do número de detalhes numa amostra, um mecanismo de análise das mensagens retiradas do tópico JMS, sem publicação, é também desejada.

Com vista a limpar as mensagens de erros, haverá a possibilidade de verificar na DB quais as mensagens em ficheiro, presentes numa pasta do sistema operativo, que após publicação foram ter ao hospital, fazer um backup das mesmas e eliminá-las da amostra.

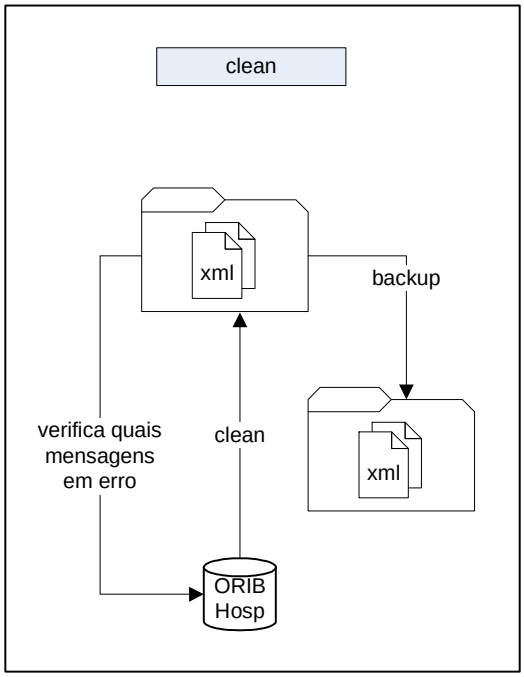


Figura 1.39: Ferramenta de Limpeza de Erros

É também permitida a eliminação de mensagens do tópico JMS, processo que, sendo demorado, é paralelizado do mesmo modo que o processo de Dump.

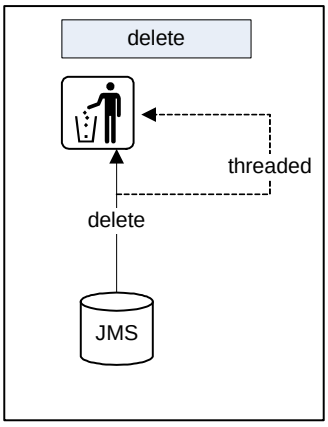


Figura 1.40: Ferramenta de Eliminação de Mensagens de Tópicos

A aplicação é também responsável por monitorizar os processos do ORIB e da integração das mensagens no ORMS, recolhendo tempos de bases de dados, de processamento interno aos eways e de hospital, guardando estes dados em ficheiros de estatísticas.

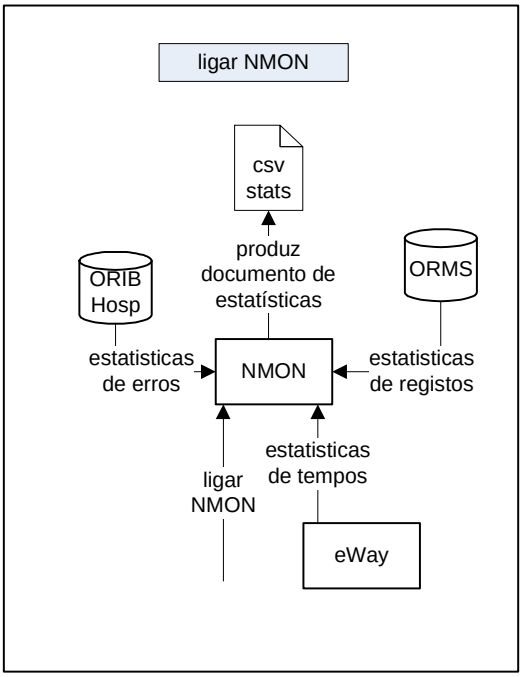


Figura 1.41: Processo de Logging

O eWay analisado deverá ser iniciado no momento em que começa o teste (para as mensagens no tópico JMS começarem a ser consumidas). Ao terminar, por outro lado deverá, para além de compilar as estatísticas, desligar o eway.

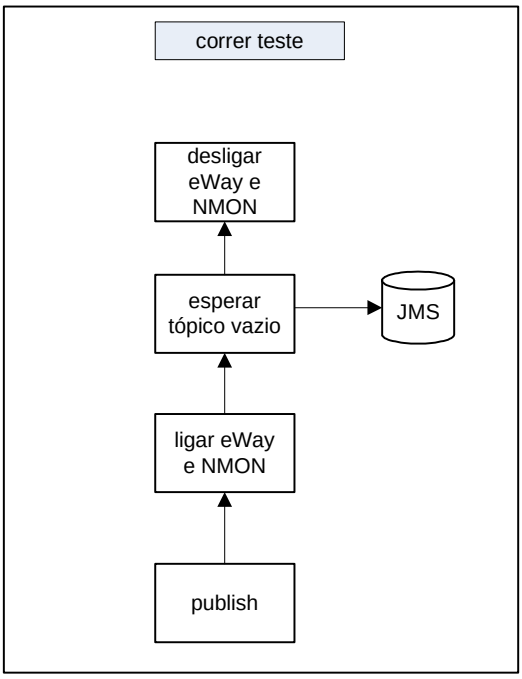


Figura 1.42: Sessão de Teste Típico

Por fim, para agendar testes fora das horas de actividade da equipa, uma aplicação de execução de testes automáticos foi desenhada, na qual podem ser executadas qualquer uma das tarefas anteriormente definidas,

em série ou em paralelo, sendo os tópicos JMS monitorizados, no caso da publicação, para levar a cabo a finalização dos testes.

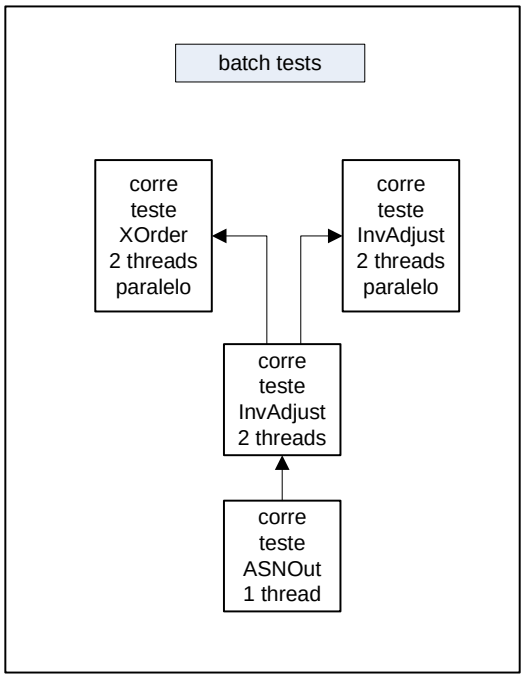


Figura 1.43: Exemplo de Utilização da Ferramenta de Testes Automáticos

3.4.3 Execução dos Testes

A execução dos testes revelou alguns pontos importantes de modelação de sistemas multi-thread de ORIB.

Os resultados, embora não cheguem ainda aos valores necessários de performance em todos os interfaces, demonstram progressos.

Em primeiro lugar, a utilização das ferramentas permitiu validar um algoritmo de hashing adequado para cada um dos interfaces.

Seguidamente, permitiu avaliar qual o nível de multi-threading necessário para conseguir chegar aos tempos alvo.

No entanto, no caso do interface ID-ORMS-09, parece ter-se chegado perto do limite de melhoramentos derivados do multi-threading. A passagem de 4 threads para 8 não levou a grande melhoria por parte deste interface, em comparação com a passagem de 1 para 4 e dos outros interfaces.

Tabela 1.6: Resultados de Multi-threading

	interfaces	# threads	detalhes/min	tempo de processamento	erros de lock
ASNOut	ID-ORMS-09	1	3144.75	05:18:12	0
	ID-ORMS-09	4	8196.52	02:02:05	0
	ID-ORMS-09	8	9902.61	01:41:03	0
InvAdjust	ID-ORMS-03	1	3997.36	01:15:03	0
	ID-ORMS-03	4	25003.42	00:12:00	0
	ID-ORMS-12	1	5026.87	00:08:00	0
	ID-ORMS-12	4	8043	00:05:00	0
	ID-ORMS-03&12	1	8292.19	00:41:02	0
	ID-ORMS-03&12	4	26173.54	00:13:00	0
Xorder	ID-ORMS-11	1	2999.08	00:50:02	0
	ID-ORMS-11	4	6493.34	00:23:01	35
	ID-ORMS-11	8	11453.92	00:13:00	60

Capítulo 4

Conclusão e Trabalho Futuro

A integração de um sistema promocional de um retalhista, dada a grande variedade de possibilidades de modelos de negócio que apresenta, passa pelo desafio de conseguir unir as vantagens dos processos de cada uma das aplicações que se tenta integrar, por vezes impedindo o aproveitamento das capacidades destes por inteiro.

No caso específico da integração do ORPM com os outros sistemas do retalhista, o projecto levou a uma completa sincronização dos processos de ambos, sendo que as dinâmicas e multi-facetadas modalidades de promoções do ORPM foram completamente portadas para os restantes sistemas do cliente, tendo por sua vez sido adaptados os modelos de identificação e de estados do ORPM para funcionar com a identificação por Buckets e ciclo de vida promocional do OLS.

Os testes de performance efectuados, por sua vez, serviram de proof-of-concept para o cliente na adaptação da tecnologia utilizada (ORIB) para as diferentes interfaces, exigências de performance e desafios no processamento eficiente, sem atrasos dos dados de negócio. O estudo de algoritmos de hash e de campos aos quais aplicar o hash servirão para, de futuro, se ultrapassar os problemas de locks e más distribuições. A plataforma de testes de performance tornará o desenvolvimento de qualquer teste futuro bastante mais rápido e os conhecimentos adquiridos sobre as limitações do multi-threading guiarão as considerações de escalabilidade no desenho de interfaces.

Por fim todo o conhecimento foi adequadamente transmitido para equipas do cliente distribuídos geograficamente entre a Europa e a Ásia, que continuarão, no caso da adaptação ao ORPM, a aplicar os conceitos de

sincronização do mesmo e, no caso dos testes de performance, irão aplicar a experiência dos poucos interfaces afinados ao resto do sistema.

Referências

[EM06] Report da Euromonitor International sobre retalho em 2006, disponível a partir de http://www.euromonitor.com/

[DEL08] Deloitte 2008 Global Powers of Retailing © 2008 Deloitte Development LLC.

[SAP07] Declaração pública da SAP AG disponível online em http://www.sap.com/about/press/press.epx?pressid=8440

[GAR06] Reports sobre retalho da Gartner em 2006 dísponíveis online em http://www.gartner.com/

[SAPJ07] Artigo sobre utilização de J2EE sobre o ERP da SAP AG disponível online em https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/c0491a51-0c6d-2a10-3dbc-9069ee2099b4

[JAVA98] Origens da linguagem de programação Java disponíveis online em http://java.sun.com/features/1998/05/birthday.html

[HIB06] Especificações da plataforma de persistência Hibernate disponíveis online em http://www.hibernate.org/5.html

[J2EE07] Especificação da plataforma J2EE disponível online em http://java.sun.com/javaee/reference/

[EJB04] Especificação dos EJB disponível online em http://java.sun.com/products/ejb/docs.html

[JMS02] Especificação da API JMS disponível online em http://java.sun.com/products/jms/docs.html

[OC4J07] Documentação do servidor de aplicações disponível online em http://www.oracle.com/technology/documentation/appserver.html

[DB10G07] Especificações da tecnologia de bases de dados relacionais da Oracle disponíveis online em http://www.oracle.com/technology/documentation/database.html

[OR06] Documentação da Oracle Retail disponibilizada online em http://www.oracle.com/technology/documentation/oracle_retail.html

[ORIB06a] Oracle Retail Integration Bus Technical Architecture Guide Release 12.0 Maio 2006 Copyright © 2006, Oracle

[XAJ01] Especificação da implementação de XA em Java disponível online em http://java.sun.com/j2ee/sdk_1.3/techdocs/api/javax/transaction/xa/XAResource.html

[ORIB06b] RIB Integration Guides © 2006 Oracle

[ORIB06c] Oracle Retail Integration Bus Operations Guide Release 12.0 Maio 2006 Copyright © 2006, Oracle

[ORPM06a] Oracle® Retail Price Management User Guide Release 12.0 Maio 2006 Copyright © 2006, Oracle

[ORPM06b] Oracle Retail Price Management Operations Guide Release 12.0 Maio 2006 Copyright © 2006, Oracle

[ORPM06c] Oracle Retail Price Management Data Model Release 12.0 Maio 2006 Copyright © 2006, Oracle

[ORMS06a] Oracle® Retail Merchandising System User Guide Release 12.0 Maio 2006

[ORMS06b] Oracle Retail Merchandising System Operations Guide - Volume 2 Message Publication and Subscription Designs Release 12.0 Maio 2006 Copyright © 2006, Oracle

[SB05a] SeeBeyond JMS Intelligent Queue User's Guide Release 5.0.5 for Schema Run-time Environment (SRE) © 2005 SeeBeyond Technology Corporation

[SB05b] Java Generic e*Way Extension Kit Release 5.0.5 for Schema Run-time Environment (SRE) © 2005 SeeBeyond Technology Corporation

[NMONa] Documentação da ferramenta de monitorização de sistemas UNIX e AIX NMON por Nigel Griffiths disponível online em http://www-941.haw.ibm.com/collaboration/wiki/display/Wikiptype/nmon

[NMONb] NMON_Analyser User Guide for V3.3 Stephen Atkins, MBCS CITP Consulting IT Specialist IBM Systems and Technology Group IBM UK Stephen Atkins/UK/IBM or steve_atkins@uk.ibm.com, disponível online em http://www-941.haw.ibm.com/collaboration/wiki/display/Wikiptype/nmonanalyser

[LOG4J] Manual de utilização da utilizade de logging LOG4J disponível online em http://logging.apache.org/log4j/1.2/manual.html

Anexo A

Modelo de Dados - Promoções ORPM

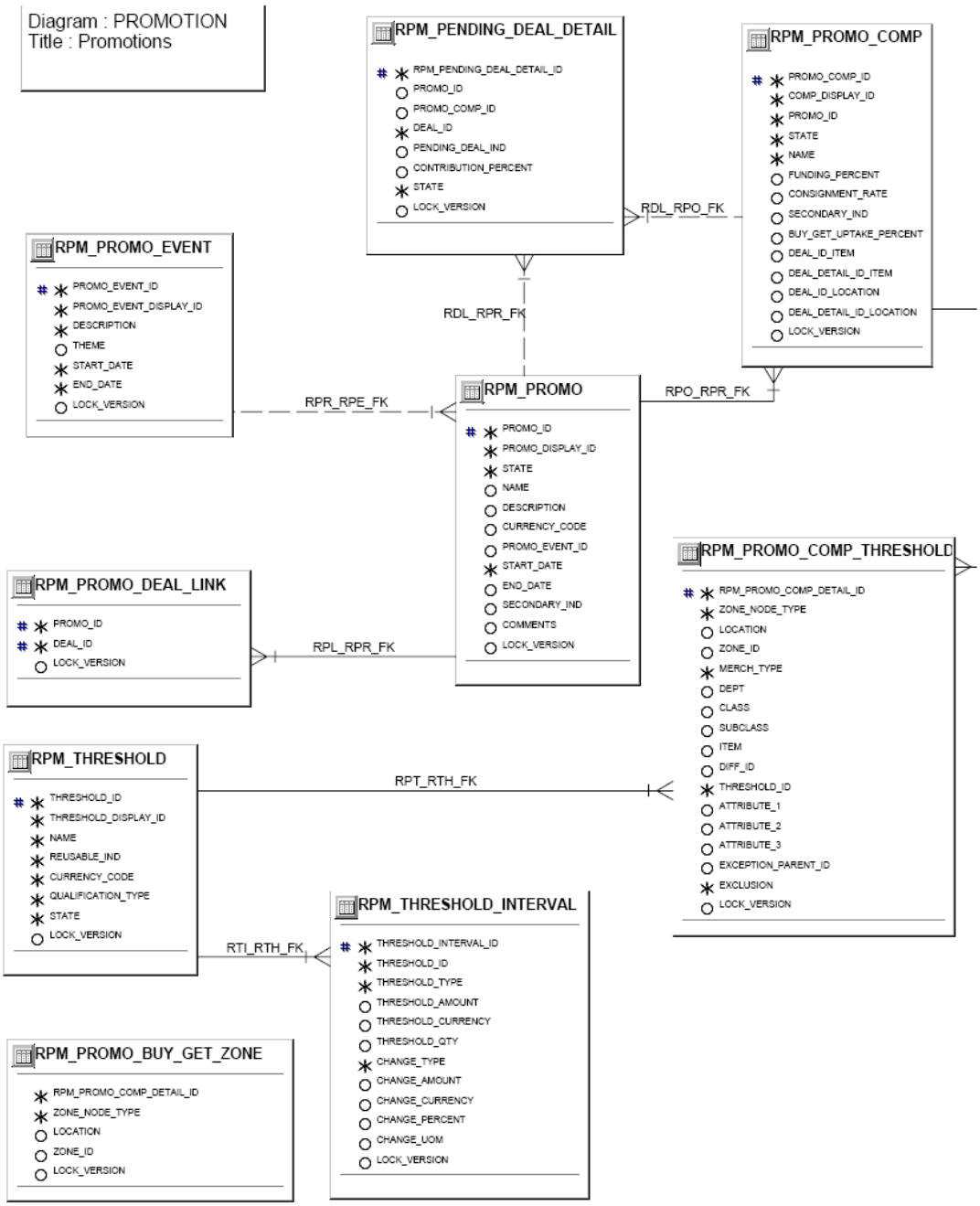


Figura 1.44: Modelo de Dados de Promoção ORPM

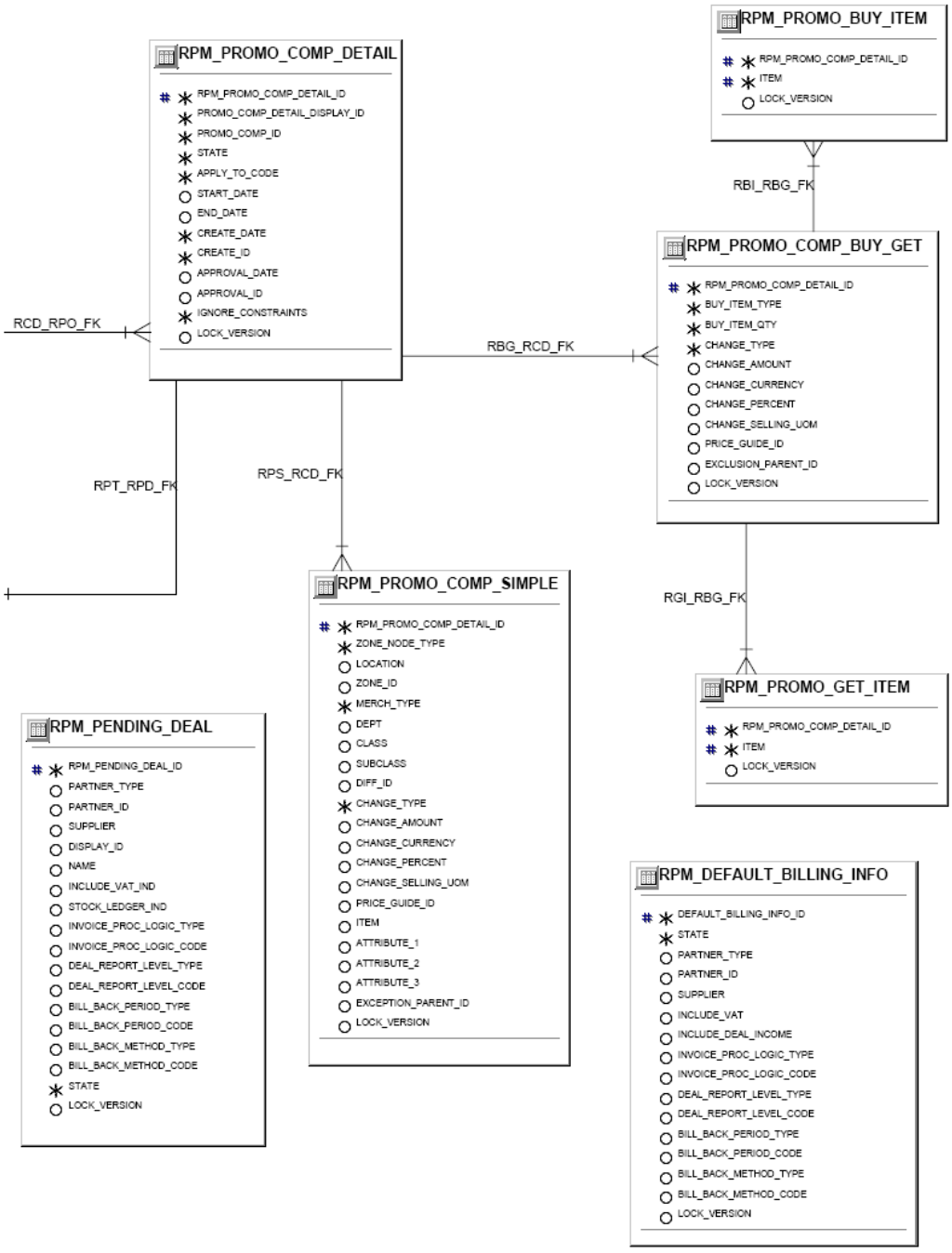


Figura 1.45: Modelo de Dados do ORPM (Cont.)