7
Estudos Experimentais

A abordagem proposta neste trabalho emergiu principalmente de experimentos práticos realizados nas atividades de pesquisa deste curso de mestrado. No total cinco estudos experimentais foram realizados e estes são apresentados no decorrer deste capítulo. Os dois primeiros estudos envolvem padrões de projeto [26], o primeiro [27] [28] em implementações isoladas e o segundo no contexto de uma aplicação [8]. O terceiro estudo experimental é um sistema multi-agentes [30] para desenvolvimento e gerência de portais na internet. Os dois últimos estudos são sistemas de informação *web* para queixas em relação a serviços de saúde [44] e para acompanhamento de auto-estradas brasileiras [11], respectivamente. Todas as aplicações envolvem implementações orientadas a objetos (OO) e orientadas a aspectos (OA) feitas em Java e AspectJ. É importante observar que estes estudos tiveram um importante papel não só na avaliação da abordagem, mas também em seu amadurecimento.

7.1.
Sistemas Utilizados nos Estudos Experimentais

Os cinco estudos realizados no contexto desta dissertação foram selecionados por quatro razões principais. Em primeiro lugar, porque eles são bastante heterogêneos e envolvem aplicações de diferentes domínios. Além disso, os dois primeiros estudos são largamente baseados em padrões de projeto, e, portanto, fortes candidatos a atenderem requisitos de qualidade relevantes neste trabalho como reusabilidade. A terceira razão por termos selecionado tais estudos é que, a exceção do primeiro, eles refletem aplicações realistas feitas em Java e AspectJ. Finalmente, a quarta razão é que os estudos possuem variados graus de complexidade em relação à natureza de seus interesses.

Os estudos experimentais deste trabalho foram aplicados na avaliação dos três pilares básicos da abordagem proposta: método de avaliação (Capítulo 4), regras heurísticas (Capítulo 5) e ferramenta de medição e avaliação (Capítulo 6).

Entretanto, nem todos os estudos avaliam os mesmos elementos da abordagem. Por exemplo, no sistema Telestrada [11] é utilizada apenas uma versão preliminar da ferramenta. Por outro lado, três dos cinco estudos são utilizados para avaliar e evoluir os principais elementos de abordagem de forma integrada. A Tabela 10 ressalta quais elementos da abordagem recebem contribuições de quais estudos. Nas linhas desta tabela são listados os estudos e nas colunas os elementos da abordagem, i.e., método, regras e ferramenta. Cada célula marcada com "X" significa que no elemento daquela coluna é utilizado o estudo correspondente à linha.

Tabela 10 – Elementos da abordagem avaliados nos estudos experimentais

Estudos	**Elementos Principais da Abordagem**		
	Método	**Regras**	**Ferramenta**
Padrões GoF Individuais	X	X	
Composição de Padrões	X	X	X
Portalware	X	X	X
Health Watcher	X	X	X
Telestrada			X

7.1.1. Padrões de Projetos

O primeiro estudo experimental [27] [28] [29] realizado no contexto desta dissertação compara implementações OO e OA dos 23 padrões de projeto descritos pela *Gang-of-Four* (GoF) [26]. As implementações dos padrões avaliados foram desenvolvidas por Hannemann e Kiczales [36], e estes autores fazem uma comparação qualitativa dos padrões baseada em atributos da engenharia de software que não são bem conhecidos, tais como, "*componibilidade*" e "*conectabilidade*"[12]. Em nosso estudo, entretanto, são utilizados atributos rigorosos da engenharia de software no critério de avaliação, tais como acoplamento, coesão, tamanho e separação de interesses. Desta forma, este estudo experimental pode ser considerado complementar ao trabalho de Hannemann e Kiczales por utilizar critérios diferentes na comparação das soluções OO e OA.

[12] Tradução para os termos em inglês *composability* e *pluggability* usados no trabalho de Hannemann e Kiczales [36].

Com o objetivo de permitir que as distintas implementações de um mesmo padrão sejam comparáveis são feitas pequenas alterações no código original disponibilizado por Hannemann e Kiczales em [67]. Dois exemplos destas alterações são: (i) garantia de um mesmo estilo de programação e (ii) adição (ou remoção) de funcionalidades que ocorre em uma das implementações e não ocorre em outra. A decisão entre adicionar ou remover uma funcionalidade foi tomada dependendo de sua relevância para o contexto do padrão. Esta etapa de garantir implementações comparáveis é chamada de alinhamento do código. Após o alinhamento do código, são aplicadas as três primeiras fases do método de avaliação: medição, aplicação de regras e análise.

Na atividade de medição, o conjunto de métricas definido na Tabela 1 (Subseção 4.2.1) é utilizado sobre as implementações Java e AspectJ dos padrões GoF. Com o resultado das medições, as regras são então aplicadas de tal forma a gerarem alertas. Estes alertas advertem sobre a possibilidade do interesse de algum padrão ser transversal ao sistema. Desta forma, tais alertas servem de guia para a fase de análise, na qual os possíveis interesses transversais são verificados. Neste estudo, a ferramenta de suporte ao método não foi utilizada porque esta ainda se encontrava em fase inicial de desenvolvimento. Os resultados completos da avaliação feita neste estudo encontram-se disponíveis nas referências [27] [28] [29] e no Apêndice A são apresentados os dados detalhados de três padrões: *Observer*, *Factory Method* e *Builder*.

7.1.2. Middleware OpenOrb: Um Estudo de Composição de Padrões

Um dos principais problemas na implementação de múltiplos padrões de projetos em um sistema é que estes padrões não se limitam a afetar as classes base da aplicação. Por outro lado, eles também interagem entre si de forma tão heterogênea que torna difícil a separação dos elementos que implementam cada padrão. No estudo de caso anterior, o objetivo foi avaliar de maneira isolada as implementações Java e AspectJ dos padrões de projeto feitas por Hannemann e Kiczales [36]. Neste segundo estudo, entretanto, os 23 padrões são avaliados no contexto de uma aplicação realística e na presença de grandes interações entre eles. As interações entre os padrões é resultado principalmente da forma em que

estes se compõem, variando desde uma simples chamada de método até o compartilhamento de uma ou mais classes do sistema.

A aplicação avaliada neste estudo compreende um sistema de *middleware* [3] [9] em que duas diferentes versões foram utilizadas, a primeira OO implementada em Java e a segunda OA implementada em AspectJ. A Figura 31 mostra a composição de diversos padrões em um diagrama de classes parcial da versão Java deste sistema. Estes padrões se associam para atingir os requisitos cruciais definidos neste software, como componibilidade e adaptabilidade. Cada número apresentado na figura representa um padrão específico e estes números são associados aos métodos e atributos que implementam o padrão correspondente. Por exemplo, a implementação do padrão *Decorator* (representado pelo número 1) inclui o atributo `bind` e vários métodos como `makeRequest()`, `breakBind()`, `rebind()`, `checkPreMethods()` e `checkPosMethods()` da classe `DecoratorBind`.

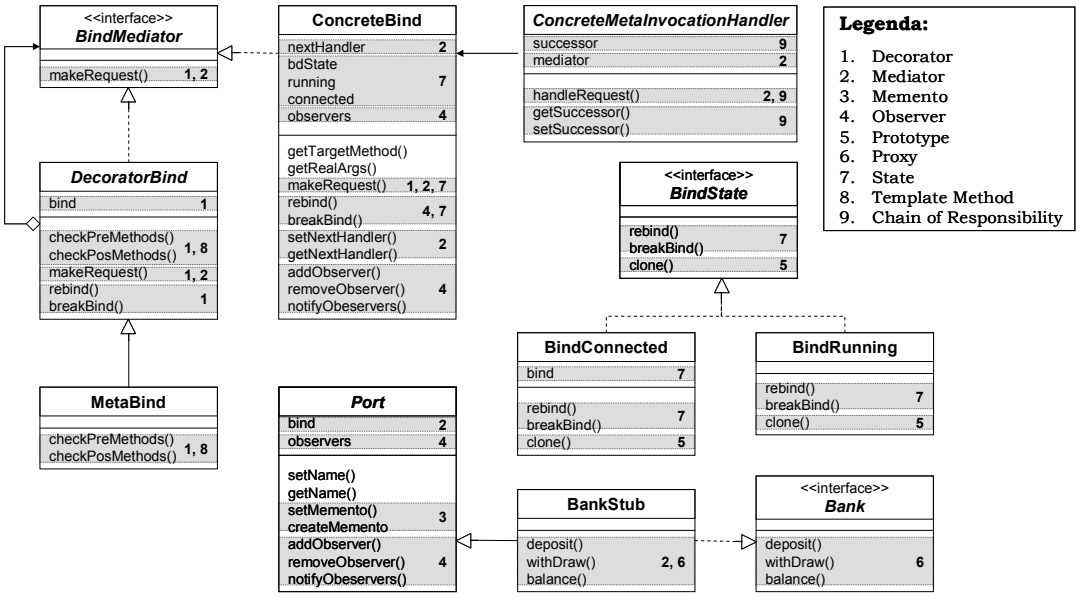

Figura 31 – Diagrama de classes OO parcial da middleware OpenOrb

Neste estudo experimental os padrões são avaliados em composições de dois a dois, ou seja, cada par de padrão que possui algum nível de interação é isolado do restante do sistema para que possa ser estudado. O processo de avaliação segue as atividades definidas no método apresentado no Capítulo 4 para um total de 62 composições. Desta forma, o primeiro passo é medir cada uma das composições utilizando as métricas descritas na Tabela 1 (Subseção 4.2.1). Em seguida, aplicar as regras heurísticas de separação de interesses, acoplamento e

coesão (Capítulo 5). Finalmente, a última atividade do processo de avaliação é avaliar os alertas gerados pela aplicação das regras. Note que, assim como no primeiro estudo, o objetivo é comparar as implementações e, portanto, não foram aplicadas refatorações nas versões do sistema.

A ferramenta apresentada no Capítulo 6 foi utilizada para automatizar as atividades de medição e aplicação das regras neste estudo. É importante ressaltar que este segundo estudo experimental foi de fundamental importância para o amadurecimento da ferramenta, em especial, do módulo Analisador de Regras. O resultado detalhado da avaliação de ambas as versões deste sistema de *middleware* pode ser encontrado nas referências [8] [9]. O Apêndice B deste documento apresenta, entretanto, as informações obtidas para 4 pares de padrões: *Observer* com *Factory Method*, *Singleton* com *Façade*, *Proxy* com *Interpreter* e *Prototype* com *State*.

7.1.3. Portalware

Sistemas Multi-Agentes (SMA) envolvem vários interesses, como Autonomia, Adaptação e Colaboração, que não são bem tratados pelas abstrações dos paradigmas tradicionais de desenvolvimento (por exemplo, OO). O quão reutilizável e manutenível é um SMA depende largamente de quão bem separados estão os interesses dos agentes nas fases de projeto e implementação [30] [31]. Assim, em nosso terceiro estudo experimental é utilizado um SMA para desenvolvimento e gerência de portais na internet, chamado Portalware [30]. Este sistema foi desenvolvido no Laboratório de Engenharia de Software (LES) desta universidade e implementado em duas versões: Java e AspectJ.

O objetivo da utilização do Portalware neste estudo é avaliar como as técnicas de desenvolvimento OO e OA podem ser usadas para separar os interesses de agentes. Para isso, foram comparadas as duas implementações desta aplicação (Java e AspectJ) que possuem exatamente as mesmas funcionalidades. A Figura 32 ilustra o projeto parcial da primeira versão do sistema feito em Java. A versão AspectJ foi obtida a partir de refatorações da implementação original e é parcialmente apresentada no diagrama de classes da Figura 33. Como nos dois estudos de caso anteriores, as três primeiras atividades do método de avaliação foram utilizadas: medição, aplicação de regras heurísticas e análise dos resultados.

A ferramenta de medição e avaliação também foi utilizada neste estudo, suportando as duas primeiras atividades citadas. Os dados de medição e aplicação das regras para os interesses Adaptação, Colaboração e Autonomia são apresentados no Apêndice C.

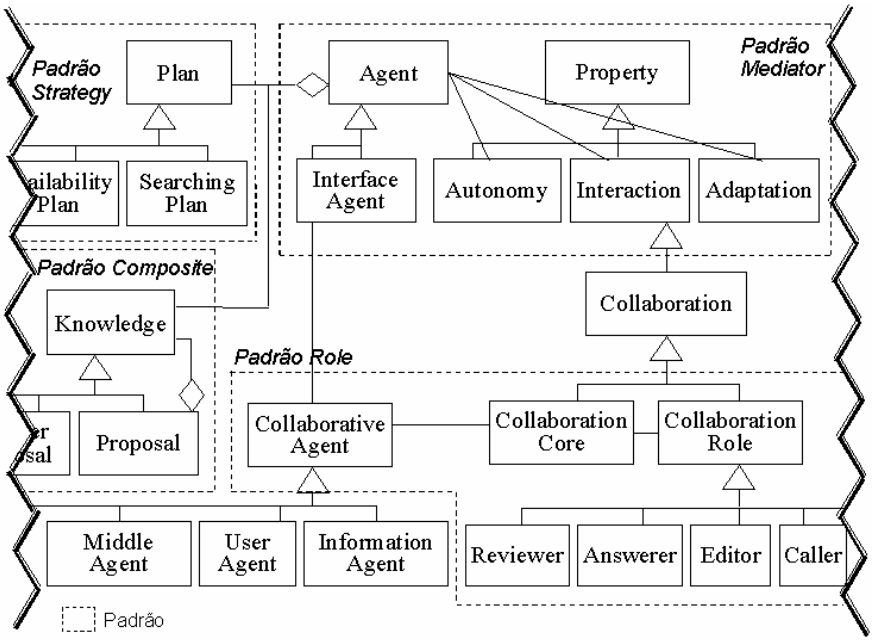

Figura 32 – Diagrama de classes OO parcial do Portalware

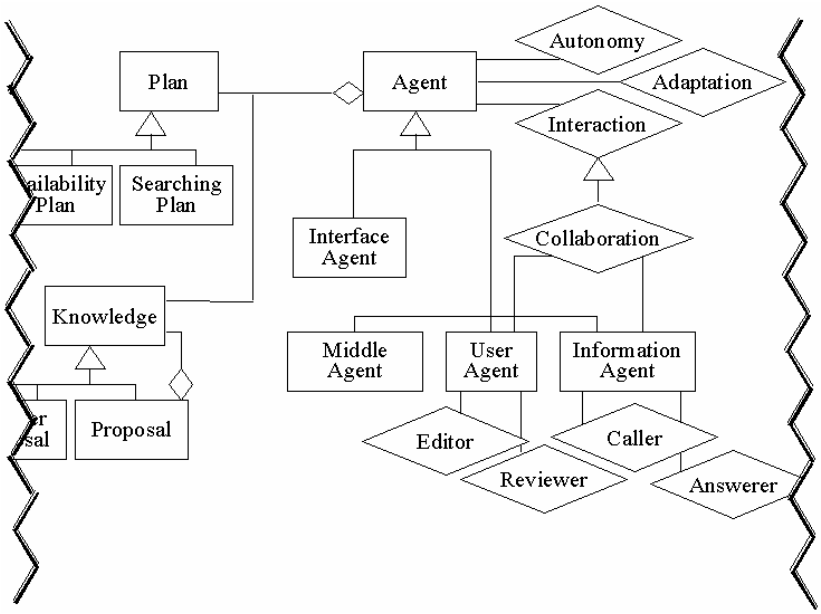

Figura 33 – Diagrama de classes OA parcial do Portalware

7.1.4. Health Watcher

O quarto estudo experimental deste trabalho consiste em comparar as implementações OO e OA de um sistema de informação baseado em *web*. A principal funcionalidade deste sistema, chamado Health Watcher [44], é o registro de queixas para melhoria da qualidade de serviços providos por instituições de saúde. O estudo envolve a avaliação de dois interesses transversais bem conhecidos na literatura – Concorrência e Distribuição – e seus efeitos nos componentes do sistema. A escolha do Health Watcher como estudo de caso se deve por três razões principais. Em primeiro lugar, este sistema pertence a um grupo externo ao nosso ambiente de pesquisa e tanto a versão OO quanto a versão OA foram desenvolvidas por pesquisadores da Universidade Federal de Pernambuco. A segunda razão é que avaliações anteriores, tanto qualitativas [7] quanto baseadas exclusivamente em métricas [44], já haviam sido conduzidas. Estas avaliações anteriores permitem comparação de resultados com a nossa abordagem baseada em regras heurísticas. O terceiro motivo pela escolha deste sistema é que sua implementação envolve várias tecnologias Java comumente adotadas pela indústria, como Invocação Remota de Método (RMI), *Servlets* e Conexão com Banco de Dados (JDBC).

As versões OO e OA do sistema de informação Health Watcher são implementadas nas linguagens Java e AspectJ, respectivamente. Na implementação Java, o padrão arquitetural "Em Camadas" [7] é usado para estruturar os componentes em quatro níveis principais: interface com o usuário, distribuição, negócios e dados. A Figura 34 apresenta o diagrama de classes parcial desta implementação e destaca as quatro camadas do padrão. A implementação AspectJ tem como propósito separar os interesses transversais relativos à Distribuição, Persistência e Concorrência. Esta versão ainda é estruturada seguindo o padrão "Em Camadas", entretanto, o interesse Distribuição não é mais implementado como uma camada e sim como um aspecto do sistema. O resultado da aplicação das métricas e regras heurísticas para os interesses Concorrência e Distribuição deste estudo é apresentado no Apêndice D.

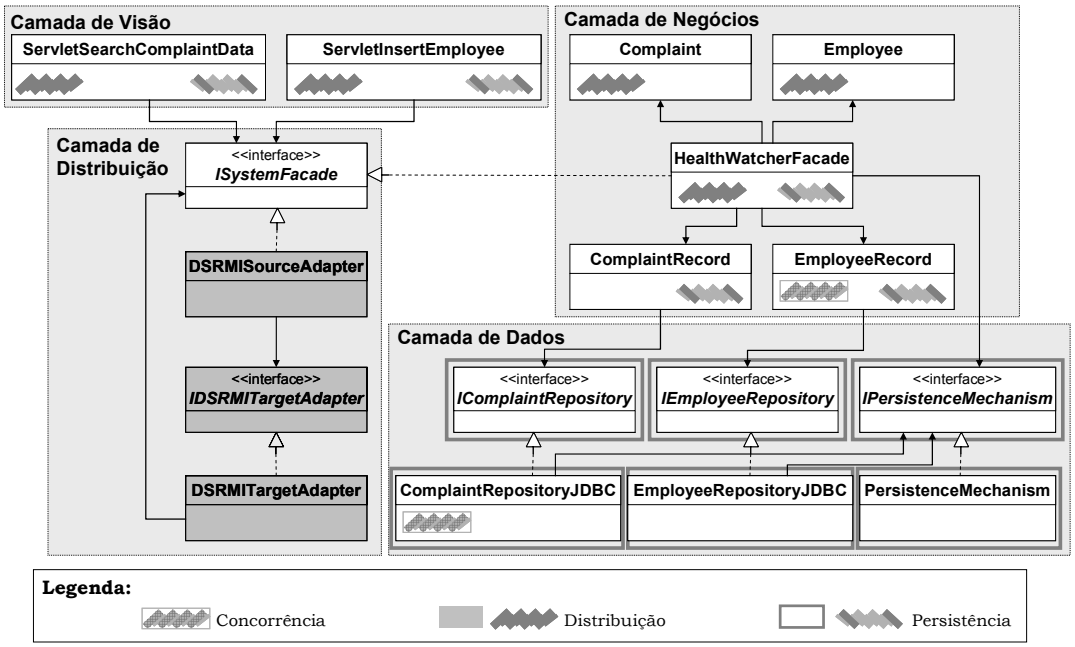

Figura 34 – Diagrama de classes OO parcial do Health Watcher

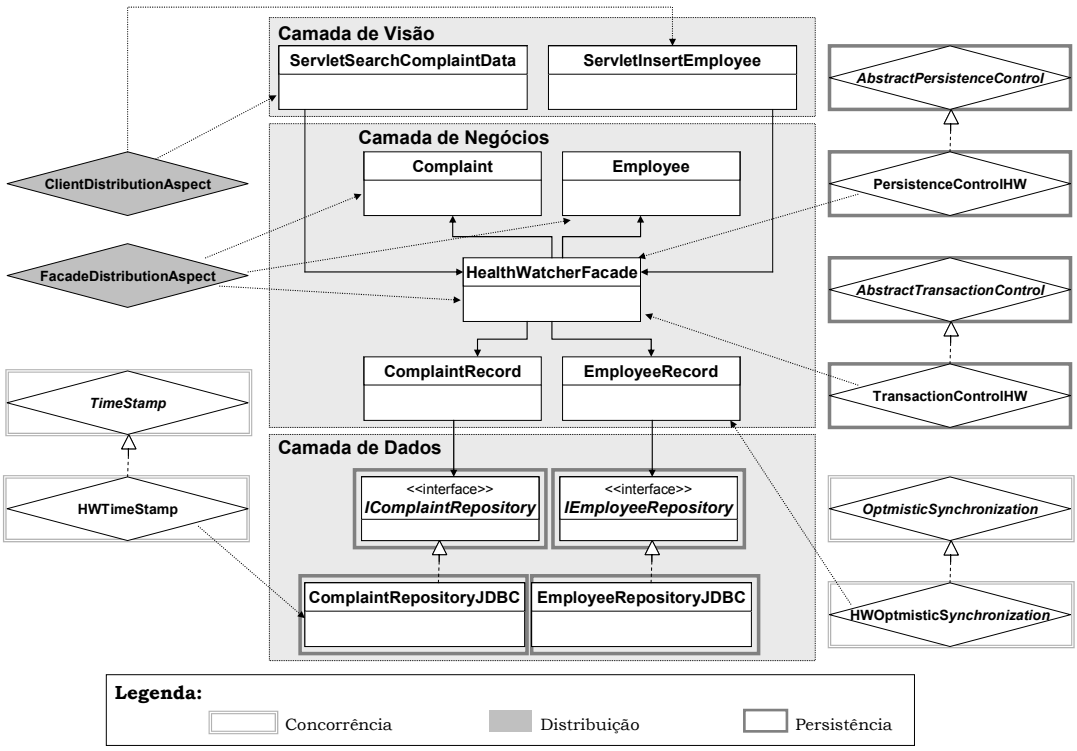

Figura 35 – Diagrama de classes OA parcial do Health Watcher

7.1.5. Telestrada

Em nosso último estudo experimental é utilizado o Telestrada [10] [11], um sistema de informação sobre auto-estradas brasileiras. Este sistema envolve cinco subsistemas: Central de Banco de Dados, Sistema de Informação Geográfica, Centro de Operação de Chamadas (*call-center*), Sistema de Operações em Estradas e Sistema de Gerência de Reclamações. O objetivo deste estudo é verificar se a separação do interesse de tratamento de exceções foi bem sucedida na implementação OA do sistema Telestrada. Para isso são comparadas as implementações antes (OO) e após (OA) este interesse ter sido separado em aspecto. O módulo de medição da ferramenta apresentada no Capítulo 6 foi utilizado e a avaliação foi baseada exclusivamente em métricas. Portanto, o método de avaliação e as regras heurísticas não foram utilizados, o que oferece oportunidade para um estudo futuro mais aprofundado. O resultado da avaliação baseada em métricas é reportado na referência [10].

7.2. Contribuições dos Estudos

Os cinco estudos de caso realizados neste trabalho contribuíram de forma unificada para avaliação e amadurecimento da abordagem. Pelas características individuais de cada estudo, eles puderam exercitar pontos diferentes da abordagem e contribuir de forma mais completa. Por exemplo, a organização do método em etapas emergiu destes estudos. Na verdade, o método de avaliação proposto neste trabalho reflete as atividades feitas por pesquisadores em estudos quantitativos. Desta forma, a grande contribuição do método é organizar e sistematizar as atividades de avaliação desempenhadas em outros estudos semelhantes. Em contra partida, os cinco estudos deste trabalho foram importantes para que pudéssemos perceber as prioridades na avaliação de software OA. Ou seja, percebemos as vantagens da avaliação orientada a interesse, uma vez que este atributo afeta a qualidade de outros atributos do software.

Em relação ao conjunto de regras heurísticas, os cinco estudos de caso realizados neste trabalho permitiram efetuar uma avaliação empírica das mesmas. Os estudos contribuíram para inferir novas regras e melhorar àquelas existentes.

Desta forma, o conjunto de regras foi constantemente reformulado a cada novo estudo de tal forma a se tornar mais eficaz na identificação de problemas. As regras que não foram eficazes nos estudos passaram por aperfeiçoamentos ou foram descartadas. Assim, o conjunto final de regras apresentado no Capítulo 5 tem se mostrado como um importante instrumento para identificar problemas não triviais em medições. A Tabela 11 destaca os principais problemas identificados pela utilização das regras heurísticas nos sistemas avaliados. Estes problemas não são facilmente detectáveis em uma avaliação baseada exclusivamente em métricas, como discutido na Seção 5.1.

Tabela 11 – Problemas identificados pelas regras heurísticas nos sistemas

Estudos de Caso		Interesses Avaliados	Problemas Identificados					
			A	B	C	D	E	F
1°	Padrão Builder	Papel Director					X	
	Padrão Factory Method	Papel Creator		X			X	
	Padrão Observer	Papel Observer						X
		Papel Subject						
2°	Composição Factory Method com Observer	Padrão Factory Method	X			X	X	
		Padrão Observer						
	Composição Façade com Singleton	Padrão Façade			X			
		Padrão Singleton						
	Compoição Prototype com State	Padrão Prototype						
		Padrão State					X	X
	Composição Interpreter com Proxy	Padrão Interpreter			X			
		Padrão Proxy						
3°	Health Watcher	Concorrência		X		X		
		Distribuição						
		Persistência						
4°	Portalware	Adaptação						
		Autonomia						
		Colaboração		X		X	X	X

A Tabela 11 apresenta os quatro estudos nos quais as regras heurísticas foram utilizadas e seis problemas associados à interpretação equivocada dos resultados de medições. Para o primeiro estudo destacamos três padrões nesta tabela e os papéis destes padrões avaliados. Para o segundo estudo são mostradas quatro composições de padrões e seus respectivos interesses (padrões) avaliados. As seis últimas colunas da Tabela 11 mostram os problemas identificados pelas regras em nossos estudos: (a) alerta falso por interesse espalhado e entrelaçado; (b) alerta falso por alto acoplamento e/ou baixa coesão; (c) resultados não mostram um problema existente; (d) resultados não indicam onde está o problema; (e) conflitos entre valores medidos; e (f) métricas não relacionam os problemas existentes. Estes problemas são apresentados e exemplificados na Seção 5.1.

Na Tabela 11, cada "X" significa que o conjunto de regras apontou um problema que não é facilmente identificado por avaliação exclusivamente baseada em medição. Note que a célula vazia nem sempre significa que o sistema não possui problemas de interesse transversal, em certos casos a ausência do "X" significa que o problema existente é facilmente identificado a partir dos números. Por exemplo, na avaliação heurística do papel *Subject* do padrão *Observer* [26] este interesse é classificado como "Secundário" (Apêndice A) e ele é realmente um interesse transversal [28] [36]. Entretanto, a Tabela 11 não apresenta nenhum "X" na linha relativa a este interesse porque a avaliação direta sobre os resultados de medição (sem regras) é suficiente para identificar este problema.

O quinto estudo de caso, Telestrada, foi utilizado para avaliar a ferramenta de medição e avaliação. Como neste estudo foi utilizada uma versão inicial da ferramenta, ele foi de fundamental importância para a identificação de *bugs*. Além disso, o Telestrada foi desenvolvido por pesquisadores externos ao nosso grupo de pesquisa o que é importante para garantir a robustez da ferramenta. A ferramenta AJATO também foi utilizada nos outros estudos de caso, exceto no primeiro, permitindo seu amadurecimento durante este trabalho. Atualmente, esta ferramenta vem sendo distribuída e utilizada por pesquisadores em diversas instituições como, por exemplo, Universidade de Lancaster (UK), Universidade Estadual de Campinas (UNICAMP), Universidade Federal do Rio Grande do Norte (UFRN), Universidade de Ciências Aplicadas de Kiel (Alemanha) e Universidade de Toronto (Canadá).

7.3. Restrições dos Estudos

Algumas métricas utilizadas neste estudo (LOC e LCOO, por exemplo) têm sido criticadas na literatura [37]. Na verdade, resultados obtidos por métricas de tamanho podem ser difíceis de interpretar, pois altos valores podem significar tanto replicação de código como melhoria de modularidade. Além disso, a métrica de coesão LCOO é criticada por não apresentar uma boa fundamentação teórica ou validação empírica. Apesar das limitações destas métricas, o maior problema com elas ocorre quando as métricas são utilizadas de forma isolada, o que não é o caso deste trabalho. Em nossos estudos, as métricas de acoplamento, coesão,

tamanho e separação de interesses são utilizadas de forma complementar e têm se apresentado como fontes confiáveis de informação sobre o sistema.

Os cinco estudos de caso explorados neste capítulo são dependentes de características das linguagens de programação utilizadas: Java e AspectJ. Entretanto, a decisão de utilizar estes sistemas foi tomada porque eles possuem boas implementações OO e OA. Além disso, as linguagens Java e AspectJ são as mais difundidas destes paradigmas e possuem ampla utilização tanto na comunidade acadêmica quanto na indústria. Um outro ponto em favor das linguagens utilizadas é que elas implementam os principais conceitos definidos para os paradigmas OO e OA.