Universidade Federal da Bahia
Universidade Estadual de Feira de Santana

DISSERTAÇÃO DE MESTRADO

Aplicando Síntese Temática em Engenharia de Software

Luciana Carla Lins Prates

Mestrado Multiinstitucional em Ciência da Computação – MMCC

Salvador - BA

2015

LUCIANA CARLA LINS PRATES

APLICANDO SÍNTESE TEMÁTICA EM ENGENHARIA DE SOFTWARE

Dissertação de mestrado apresentada ao Programa Multiinstitucional de Pós-Graduação em Ciência da Computação da Universidade Federal da Bahia, Universidade Salvador e Universidade Estadual de Feira de Santana, como requisito parcial à obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Manoel Gomes de Mendonça Neto

Coorientador: Prof. Dr. José Amâncio Macedo Santos

Salvador

2015

Prates, Luciana Carla Lins.

Aplicando Síntese Temática em Engenharia de Software / Luciana Carla Lins Prates. – Salvador, 2015.

205f. : il.

Orientador: Prof. Dr. Manoel Gomes de Mendonça Neto.
Coorientador: Prof. Dr. José Amâncio Macedo Santos.

Dissertação (mestrado) – Universidade Federal da Bahia, Instituto de Matemática, Doutorado Multiinstitucional em Ciência da Computação, 2015.

Referências bibliográficas.

1. Síntese Temática. 2. Engenharia de Software. 3. Code Smells.

I. Mendonça Neto, Manoel Gomes de. II. Universidade Federal da Bahia, Instituto de Matemática. III. Título.

CDU – 004.41

LUCIANA CARLA LINS PRATES

APLICANDO SÍNTESE TEMÁTICA EM ENGENHARIA DE SOFTWARE

Esta dissertação foi julgada adequada à obtenção do título de Mestre em Ciência da Computação e aprovada em sua forma final pelo Programa Multiinstitucional de Pós-Graduação em Ciência da Computação da UFBA-UNIFACS-UEFS.

Salvador, 18 de setembro de 2015.

_____

Prof. Manoel Gomes de Mendonça Neto (orientador), Ph.D.
Universidade Federal da Bahia – UFBA

_____

Prof. Dr. Cláudio Nogueira Sant"Anna.
Universidade Federal da Bahia – UFBA

_____

Prof. Dr. Rodrigo Oliveira Spínola
Universidade Salvador - UNIFACS

RESUMO

A revisão sistemática é um recurso importante para a engenharia de software baseada em evidências, que consiste em uma forma de síntese dos resultados de pesquisas primárias relacionados com um problema específico. O presente trabalho tem como objetivo apresentar um método refinado de síntese temática em Engenharia de Software com base na proposta de Cruzes e Dyba (2011a).

O método foi testado na análise qualitativa de um conjunto de estudos primários provenientes de uma revisão sistemática sobre o tema Code Smells. Os principais resultados são a apresentação de passos que compõem o método adotado, lições aprendidas, principais dificuldades durante o processo, bem como as descobertas relacionadas aos resultados obtidos para o tema analisado.

Palavras-Chave: Síntese Temática, Análise Qualitativa, Análise Secundária, Code Smells.

ABSTRACT

The systematic review is an important resource for evidence-based software engineering, which consists of synthesizing the results of primary studies related to a specific problem. The current work has the objective of refining a method of thematic synthesis in software engineering, based on the proposal of Cruzes & Dyba (2011a).

The method was tested on the qualitative analysis of a set of primary studies from a systematic review on *Code Smells*. The main results of the work are the refined steps of the method, lessons learned during its enactment, main difficulties during the process and the insights gained on the analyzed theme (code smells).

Keywords: Thematic synthesis, Qualitative Analysis, Secondary Analysis, Code Smells.

A Deus, minha família e orientadores pelo apoio, força, incentivo, companheirismo e amizade. Sem eles nada disso seria possível.

AGRADECIMENTOS

A Deus por toda força e persistência concedida ao longo deste tempo, por me amparar nos momentos difíceis, me mostrar o caminho certo e por suprir todas as minhas necessidades.

Agradeço à minha família, em especial à minha mãe Maria Lúcia Lins e Silva e meu filho Pedro José Galvão Nonato Alves Neto, símbolos de amor que me acompanham, e que sempre me conduziram a enfrentar com humildade e persistência as dificuldades da vida, encarando cada desafio com uma oportunidade de crescer e galgar novos patamares. Ao meu marido José Carlos da Silva Prates Júnior pelo apoio incondicional, amor, carinho, paciência e companheirismo, sendo o fôlego quando me faltava respiração para prosseguir.

Ao meu Mestre Profº Dr. Manoel Gomes de Mendonça Neto pelo crédito depositado em mim desde o ingresso no mestrado, pelo posicionamento sempre coerente e sensato que nos conduz à clareza de raciocínio, pelas críticas e sugestões que ajudaram a melhorar a qualidade desta dissertação, consolidando o exemplo de profissional que norteará para sempre os meus passos.

Ao meu coorientador Profº Dr. José Amâncio Macedo Santos pela paciência, disponibilidade, compartilhamento de conhecimento, confiança, onde as contribuições realizadas ao longo da realização de todo o trabalho aqui apresentado e da parceria firmada foram fundamentais para o desenvolvimento do estudo e concretização da realização da síntese.

A todos os meus colegas e amigos do mestrado, pela disponibilidade de parcerias em trabalhos, estudos solidários e, com certeza, pelo laço de amizade construído que será mantido em nossas vidas com muito carinho.

A todos os amigos pessoais que torceram por isso e continuam torcendo e que se fazem presentes na forma de apoio e incentivo.

Experiência não é o que acontece com um homem; é o que um homem faz com o que lhe acontece.

Aldous Huxley

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ATFD	*Acess to Foreign Data*
ES	Engenharia de Software
ESBE	*Evidence-Based Software Engineering*
ESE	Engenharia de Software Experimental
GQM	*Goal, Question, Metrics*
MS	Mapeamento Sistemático
MVC	*Model-view-controller*
OO	Orientado a Objetos
RS	Revisão Sistemática
SGBD	Sistema de Gerenciamento de Banco de Dados
SQL	*Structured Query Language*
TCC	*Tight Class Cohesion*
WMC	*Weighted Contagem Method*

Capítulo 1

Este capítulo apresenta a introdução do tema abordado nesta dissertação, aclarando a motivação para a realização do trabalho. Em seguida é apresentado o problema de pesquisa, os objetivos, a metodologia adotada e, por último, os resultados e contribuições desta tese de mestrado.

1 INTRODUÇÃO

Estudos experimentais são executados com a finalidade de descobrir algo desconhecido ou de testar hipóteses, permitindo a formulação de possíveis teorias científicas (Kitchenham e Dyba, 2004). Durante a realização de estudos experimentais, o investigador envolvido coleta os dados e realiza sua análise para determinar o que estes dados coletados significam.

Na área de saúde, o paradigma experimental, que originou a prática baseada em evidência, já está arraigado, apresentando-se como um fator crítico de sucesso para influenciar a investigação e prática na medicina clínica (Paterson, 2001). Por sua vez, as ideias e as práticas envolvidas em todo este processo têm sido adotadas e adaptadas por muitas outras disciplinas que dependem de dados empíricos para fins de construção de teorias e obtenção da compreensão das suas práticas. Dentre estas disciplinas destacamos a psiquiatria, enfermagem, política social, educação e a área de tecnologia da informação, mais especificamente a área de Engenharia de Software (Kitchenham e Dyba, 2004).

A experimentação em Engenharia de Software (ES) começou a ser enfatizada na década de 1980 (Yamashita e Counsell, 2013) e desde então vem despertando um crescente interesse. Pesquisadores estão cada vez mais focados em executar estudos experimentais para investigar teorias, métodos, técnicas e novas tecnologias. Para suportar esta evolução da área, tem havido uma crescente discussão sobre métodos para a realização destes experimentos (Nguyen-Duc *et al.*, 2015; Zazworka *et al.*, 2011).

Em 2004, Kitchenham *et al.* (2014) estabeleceu pela primeira vez um paralelo entre Medicina e ES, explicitando em seus estudos a importância da adoção do paradigma

baseado em evidências para ajudar a área de ES a caracterizar uma determinada tecnologia. Segundo os autores, a Engenharia de Software Baseada em Evidência (ESBE) (do inglês *Evidence-Based Software Engineering*) provê meios pelos quais melhores evidências provenientes da pesquisa são integradas com experiência prática e valores humanos no processo de tomada de decisão, considerando o desenvolvimento e a manutenção do software (Kitchenham *et al.,* 2014).

Segundo Biolchini e Travassos (2007), a experimentação em ES tem como objetivo caracterizar, avaliar, prever, controlar ou melhorar tanto os produtos, como também os métodos, recursos, modelos ou teorias. Para Basili *et al.* (1996), novos métodos, técnicas, linguagens e ferramentas para ES só deveriam ser apresentados para uso se passassem por um processo de experimentação antes que lhes garantisse maior segurança nos resultados.

Em vista disso, para atingir um nível adequado de evidência a respeito da caracterização de um determinado assunto relacionado à ES, a ESBE faz uso de dois tipos de estudos: estudos primários e secundários (Kitchenham e Charters, 2007). Por estudos primários, entende-se a condução de estudos que visem caracterizar uma determinada tecnologia em uso, dentro de um contexto específico, tendo como exemplos a etnografia, pesquisa-ação, *grounded theory*, experimentos controlados, pesquisa de opinião, estudos de caso e *surveys* (Nguyen-Duc *et al.*, 2015).

Estudos secundários correspondem a uma análise feita com o mesmo objetivo da análise primária, mas com técnica diferente ou com a finalidade de responder a novas perguntas, usando os estudos primários como fonte de pesquisa (Biolchini e Travassos, 2007). Ou seja, são estudos que visam identificar, avaliar e interpretar resultados relevantes de uma determinada questão de pesquisa, produzidos para estabelecer comparações, generalizações sistemáticas ou mapeamento de investigações individuais (Cruzes, 2007). Resultados obtidos através da análise de estudos primários correlatos atuam como fonte de informação a ser investigada por estudos secundários, podendo inclusive agregar resultados novos não percebidos nos estudos primários de forma independente, auxiliando na identificação de teorias e generalizações.

Alguns destes métodos passaram a ser aplicados pela ESBE nos últimos anos. O interesse por estudos secundários apresenta-se como uma demanda crescente na área de ES (Cruzes, 2007). Destacamos métodos como: revisão sistemática (Counsell, 1997), mapeamento sistemático (Arksey, O'Malley, 2005; Kitchenham e Charters, 2007). A recomendação de uso de revisões sistemáticas em ES foi uma iniciativa de Kitchenham *et al.*

(2014), pelo fato de ter constatado que as evidências obtidas em estudos da época possuíam as seguintes características: limitada, fragmentada, dispersa (Kitchenham *et al.,* 2014). Com base nestes resultados, Kitchenham *et al.* (2014) propuseram um método para a condução de revisões sistemáticas em ES.

Apesar de Kitchenham *et al.* (2014) concentrarem-se na discussão sobre revisões sistemáticas, existem mais mapeamentos do que revisões sistemáticas na área de ES (Cruzes e Dyba, 2011b; Da Silva, 2011; Kitchenham *et al.,* 2010). A realização de revisões sistemáticas (RS) esbarra na dificuldade em realizar a síntese, principalmente quando envolve análise qualitativa. Por análise qualitativa entende-se a abordagem em que os dados não se apresentam de forma mensurável, sendo importante um caráter mais exploratório das variáveis inerentes ao que se pretende estudar.

Para a realização de sínteses em RS, considerando a abordagem qualitativa, podemos citar os seguintes métodos: síntese temática (Cruzes e Dyba, 2011a), metassíntese (Sandelowski e Barroso, 2007), metaetnografia (Silva, 2013). Todos são importantes, mas alguns deles têm sido pouco adotados e requerem maior suporte teórico para ampliar seus níveis de utilização.

Segundo Thomas e Harden (2008), a análise temática tem sido identificada como um dos possíveis métodos que podem ser utilizados para a síntese de pesquisa, juntamente com metaetnografia e metassíntese (Sandelowski e Barroso, 2007). A percepção da ausência de clareza e precisão na descrição destas abordagens, ainda pouco aplicadas para a realização de sínteses, fez com que estes autores propusessem uma abordagem baseada na análise temática que ficou conhecida como "síntese temática". Assim, em 2008, surge esta abordagem utilizada para identificar, analisar e reportar padrões (temas) dentro de dados qualitativos extraídos de estudos primários. O método baseia-se em um conjunto de três passos sistemáticos.

Em 2011, Cruzes e Dyba refinaram a abordagem de Thomas e Harden, apresentando um conjunto de cinco passos para a realização de síntese temática. O método apresenta uma forma sistemática e transparente de realização de síntese temática para a área de ES.

Ambas as abordagens de síntese temática se preocupam com a identificação, análise e conclusões sobre temas recorrentes nos estudos primários. Para esta dissertação o foco é propor um método refinado de síntese temática, auxiliando a disseminação do uso da abordagem qualitativa em ES.

1.1 MOTIVAÇÃO

Alguns fatores podem ser apontados como causas prováveis da dificuldade de realização de sínteses: pouco conhecimento sobre os métodos qualitativos de análise, limitação quanto à disponibilidade do número ideal de pesquisadores necessários para sua execução (as verificações requerem pelo menos três pesquisadores envolvidos); carência de ferramental que suporte à realização de sínteses (seja em termos de ferramenta de software ou suporte teórico), dentre outras possibilidades. Os fatores citados acima são, por exemplo, o contexto do grupo Visualização de Software[1] (SoftVis) da Universidade Federal da Bahia (UFBA), formado por pesquisadores em diversas áreas relacionadas à ES.

Nos últimos 10 anos, os pesquisadores do SoftVis têm investido seus esforços em pesquisas relacionadas a diferentes áreas, como, por exemplo, visualização de software (Mendes, 2015; Novais, 2012), *code smells* (Santos, 2014; Santos e Mendonça, 2014; Santos e Mendonça, 2015; Santos, Mendonça, Silva, 2013), crenças em ES (Passos, 2014; Passos *et al.*, 2012), dívida técnica (Alves *et al.* 2014; Mendes, 2015; Soares, 2015). Estes trabalhos têm utilizado diferentes métodos empíricos em suas pesquisas, como mapeamentos sistemáticos (Alves *et al.*, 2015; Novais, 2013) e experimentos controlados (Novais, 2012). Apenas o trabalho proposto por Passos *et al.* (2012) envolve análise qualitativa, mas este apresenta características específicas, uma vez que a autora era pesquisadora e consultora das empresas usadas como objetos de estudo. Este, entretanto, não é o contexto mais comum do grupo. Acreditamos que o contexto do SoftVis é similar ao de outros grupos de pesquisa, especialmente no Brasil, em que alunos de doutorado e mestrado são os principais executores das pesquisas. Nestes casos, os pesquisadores, em geral, têm pouca experiência na fase inicial da execução de suas pesquisas, seja no conhecimento do tema ou do método de estudo adotado.

Recentemente, algumas questões de pesquisa para a área de *code smell* (Fowler, 1999) tem surgido como parte do trabalho de doutoramento de um membro do grupo. O conceito de *code smell* é bem aceito na área, mas alguns estudos têm apresentado descobertas divergentes sobre a utilidade prática do mesmo (Marinescu e Marinescu, 2011; Pope e Mays, 1995; Yamashita e Moonen, 2013b; Yamashita e Moonen, 2013c). O problema é que esses estudos, considerados isoladamente, não representam o potencial conhecimento sobre o

[1] SoftVis: http://softvis.dcc.ufba.br

assunto. A necessidade de compreensão geral sobre este assunto e o aumento da quantidade de estudos primários relacionados à área favorecem a aplicação de estudos secundários. Uma vez que os resultados (*findings*) são tipicamente apresentados de forma textual, uma abordagem para a síntese das conclusões da pesquisa qualitativa era necessária. A adoção de uma abordagem sistemática para análise qualitativa fornece o cunho científico esperado para a realização desta dissertação.

1.2 PROBLEMA DE PESQUISA

O problema geral desta dissertação é a dificuldade de encontrar suporte teórico para realização de síntese em análise qualitativa, considerando, principalmente, o contexto de grupos de pesquisa cuja execução ocorre por pesquisadores com pouca experiência. Esta falta de experiência, tipicamente percebida entre estudantes de mestrado e doutorado.

Mais especificamente, nos concentramos na demanda do grupo de pesquisa SoftVis, relacionada ao tema *code smell*. A área de *code smells* não dispõe de muitos estudos qualitativos, o que proporciona a oportunidade de explorar este lado e gerar resultados para os estudiosos e pesquisadores, baseando-se no caráter mais exploratório da pesquisa qualitativa.

A síntese temática, por permitir a aplicação da análise temática de uma forma mais explícita e por apresentar a vantagem de trabalhar com uma grande quantidade de resultados oriundos de diversos estudos primários, mostrou ser uma abordagem adequada aos objetivos do estudo.

Cruzes e Dyba (2011a) focam em ES, mas apresentam diretrizes e recomendações que norteiam o que deve ser feito. Para a realização prática, quando um grupo apresenta menos experiência em métodos qualitativos, as recomendações não se apresentam suficientes. Por exemplo, Cruzes e Dyba mencionam em determinado passo do método de síntese temática proposto que os segmentos importantes do texto devem ser rotulados e codificados, mas não explicitam como esta atividade deve ocorrer uma vez que não mencionam atividades relacionadas que deverão ser desempenhadas para que o objetivo seja atingido. Quando os pesquisadores envolvidos no estudo apresentam larga experiência, tanto nos temas, como nos métodos, estas diretrizes podem ser consideradas como uma abordagem formal, transparente, confiável e passível de auditoria. No contexto do grupo SoftVis que estamos considerando

neste estudo, este nível de experiência não é realidade. Em geral, os pesquisadores envolvidos possuem diferente formação no tema e na aplicação dos métodos experimentais, especialmente em se tratando de métodos qualitativos. Este fator, aliado às limitações percebidas no suporte teórico, tornam o processo de análise complexo e passível de erros.

1.3 OBJETIVOS

O objetivo geral deste trabalho é ampliar o suporte necessário para aplicação de síntese temática em ES, refinando o método proposto por Cruzes e Dyba (2011a) e permitindo aplicar a análise temática de uma forma mais explícita.

Como objetivos específicos, destacamos:

- Sumarizar o método de síntese temática proposto por Cruzes e Dyba para simplificar a sua compreensão.
- Refinar o método de síntese temática proposto por Cruzes e Dyba, oferecendo suporte na forma de recomendações a grupos onde a pesquisa é principalmente realizada por pesquisadores menos experientes (alunos de doutorado e mestrado).
- Aplicar o método refinado. A aplicação do método refinado ocorreu na realização de síntese temática para a área de code smells.

1.4 METODOLOGIA

A metodologia utilizada durante a pesquisa obedeceu à sequência lógica de macroatividades, conforme descrito na Figura 1.

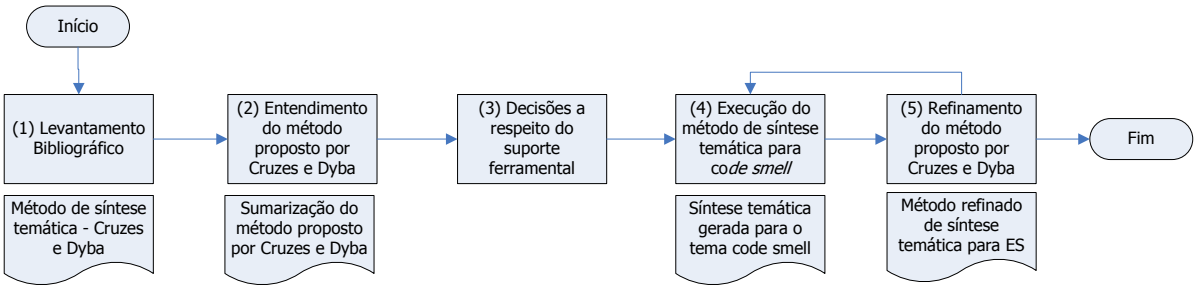


Figura 1 Metodologia Adotada

Na macroatividade 1, realizamos o levantamento bibliográfico. Durante este processo nos concentramos na consolidação de conhecimentos relacionados aos métodos de análise qualitativa e tema central da síntese (code smell). Identificamos durante esta macroatividade o método de síntese temática proposto por Cruzes e Dyba (2011a), já executado anteriormente e com foco na ES.

Na macroatividade 2, analisamos as diretrizes propostas e apresentamos todas as recomendações distribuídas no texto em formato de *checklist*. A intenção foi sistematizar e facilitar o nosso acesso às informações textuais. Além deste ganho, compreendemos que esta sumarização representa um suporte teórico mais simples e prático para aplicação do método de síntese temática.

Na macroatividade 3, realizamos as definições e criação de suporte ferramental. Foram definidas as ferramentas e instrumentos para apoiar a execução do trabalho.

- Para a filtragem: foi adotada a ferramenta Start (State of the Art through Systematic Review), na versão 4.0 (Fabbri, 2012; Zamboni *et al.*, 2010);
- Para a extração e codificação: houve uma tentativa de utilizar a ferramenta NVivo e a Atlas.TI, mas optamos por trabalhar de forma manual com um instrumento desenvolvido em Excel e denominado Planilha de Extração/Mapeamento. A escolha deu-se em grande parte pelo fato das ferramentas mencionadas serem proprietárias e pelo resultado obtido após uma avaliação da relação entre os benefícios obteníveis com as ferramentas e a curva de aprendizado.
- Para cruzamento dos dados: foi desenvolvida uma ferramenta em PHP (com uso do framework CakePHP) para sumarizar os dados coletados e codificados, permitindo que as análises fossem feitas com maior assertividade.

As macroatividades 4 e 5 foram executadas de forma iterativa uma vez que o processo de construção de um dependia/retroalimentava diretamente o outro. Consideramos

esta estratégia adequada dentro do contexto proposto para esta dissertação, pois possibilitou a construção gradual do conhecimento dos temas centrais da dissertação (método de síntese temática e *code smell*) por parte dos pesquisadores envolvidos, contribuindo para a aquisição de experiência durante a realização da pesquisa. Para a realização da síntese nesta dissertação, utilizamos tanto o referencial teórico da sumarização executada na macroatividade 2, quanto as descrições do método refinado, mencionado na macroatividade 5. Com a adoção de um ciclo de vida incremental e iterativo para execução do método, percebemos que houve uma construção do aprendizado entre os pesquisadores. Este aprendizado foi relacionado ao tema sobre o qual a síntese foi realizada e sobre o próprio método adotado.

Durante o processo de elaboração da síntese, muitas vezes não era possível registrar, de imediato, todas as recomendações percebidas pelo refinamento do método. Isso ocorreu em função da necessidade de produção dos resultados da síntese e da dependência de alguns resultados em etapas posteriores. Pesquisadores com diferentes perfis participaram das diferentes fases da aplicação do método. Destacamos duas figuras centrais para a realização do trabalho. A primeira é a da pesquisadora concentrada nos estudos relacionados à aplicação de métodos qualitativos de análise em ES (autora desta dissertação), com pouca experiência na área de *code smells.* A segunda é a do especialista no tema *code smell,* um pesquisador em fase final de doutoramento, mas com pouca experiência na aplicação de métodos qualitativos de análise.

No total, 5 pesquisadores estiveram envolvidos no estudo em momentos diferentes, assumindo diferentes papéis: professor orientador do grupo SoftVis; um aluno de doutorado, especialista no tema *code smell*; uma aluna de mestrado, autora da pesquisa; e mais dois estudantes de mestrado de áreas afins, que atuaram como voluntários na etapa de seleção de estudos primários. A participação dos estudantes e orientador deu-se da seguinte forma:

- Para a etapa de seleção dos estudos primários, os três estudantes de mestrado trabalharam junto com o especialista no tema.
- Para a fase de extração de dados para a síntese, o mesmo grupo de quatro pesquisadores (o especialista e os três alunos de mestrado) foram envolvidos.
- A fase de codificação, geração do mapa temático e da síntese temática foi realizada, principalmente, por dois pesquisadores: o especialista no tema e um aluno de mestrado, que é a autora desta dissertação, que também se

concentrou na elaboração das recomendações que refinam o método definido por Cruzes e Dyba, resultado central desta dissertação.

- Para todas as etapas, um pesquisador sênior, orientador do grupo SoftVis atuou como pesquisador independente, dirimindo dúvidas e realizando consenso, quando necessário.

Acreditamos que esta estrutura seja comum aos grupos de pesquisa, especialmente no caso do Brasil, onde as universidades são um dos principais espaços para realização de pesquisas científicas.

É importante ressaltar também que a realização da pesquisa ocorreu de forma iterativa e incremental, acompanhando o método de elaboração da síntese temática. Como o método de síntese temática proposto por Cruzes e Dyba (2011a) apresentava diretrizes sobre o que fazer, mas não trazia uma aplicação prática do como fazer, enfrentamos algumas dificuldades. Para conseguir chegar ao resultado da síntese temática para *code smells* foi necessário observar as práticas que podiam ser adotadas frente às especificidades do estudo (contexto em que foi realizado), na medida em que a síntese era executada. A cada ciclo que era rodado, retroalimentávamos o método com as necessidades que eram percebidas.

O registro do refinamento do método para demonstrar como os passos foram seguidos no contexto do nosso grupo de pesquisa não ocorreu completamente durante a execução do método. Este fato se deu, dentre outras coisas, pela necessidade de entendimento do que era proposto por Cruzes e Dyba, frente à necessidade de definições de como os passos seriam realizados na prática.

1.5 RESULTADOS E CONTRIBUIÇÕES DESTE TRABALHO

As contribuições desta dissertação podem ser sumarizadas conforme descrito a seguir:

- Sumarização do trabalho de Cruzes e Dyba (2011a), possibilitando que grupos de pesquisa em diferentes contextos possam valer-se das informações sistematizadas para facilitar o cumprimento das etapas.
- O refinamento do método de síntese temática, utilizando como base o método proposto pelos autores Cruzes e Dyba. Este refinamento apresenta-se útil em

um contexto comum de trabalho envolvendo grupos de pesquisa consonantes com os existentes em Universidades. Consideramos, para tanto, uma estrutura onde o professor pesquisador orienta estudantes de doutorado, mestrado e graduação na realização de pesquisas científicas. Durante a execução deste refinamento, algumas contribuições pontuais também ocorreram. Como exemplo, podemos citar o Checklist de Avaliação da Confiabilidade da Síntese proposto. A importância deste checklist está em aclarar as verificações que são importantes para a realização de uma síntese, oferecendo um subsídio menos subjetivo em comparação com diretrizes gerais.

- A síntese temática realizada para a área de *code smell*.

1.6 ORGANIZAÇÃO DA DISERTAÇÃO

Esta dissertação está dividida da seguinte forma: o Capítulo 2 apresenta o referencial teórico, contendo os principais conceitos trabalhados nesta dissertação.

No Capítulo 3 são apresentadas as diretrizes para realização de síntese temática em ES, seguindo o método proposto por Cruzes e Dyba (2011a).

No Capítulo 4 são apresentadas as considerações práticas para a realização do método de síntese temática, ou seja, um refinamento do método de síntese temática, embasado pela proposta dos autores Cruzes e Dyba (2011a).

No Capítulo 5 é apresentada a aplicação do método refinado para a área de *code smell*.

Finalmente, no Capítulo 6 é apresentada a conclusão, descrevendo os principais resultados obtidos, as lições aprendidas com a aplicação do método, as limitações e ideias para trabalhos futuros relacionados a esta dissertação.

No Apêndice A estão os resultados da síntese temática para *code smell*, envolvendo os principais temas identificados, principais resultados obtidos e a discussão sobre os resultados.

No Anexo 1 está o artigo que está sendo trabalhado, em inglês, para futura publicação. Este artigo contém, além da síntese temática, maiores detalhes sobre a RS realizada e os passos iniciais antes da realização da síntese propriamente dita.

Capítulo 2

Neste Capítulo será apresentada a revisão da literatura explorando temas relacionados à pesquisa qualitativa e alguns métodos que podem ser associados com a finalidade de auxiliar a realização de sínteses. Apesar da amplitude e complexidade do tema, uma abordagem voltada para a área de ES foi priorizada para o escopo desta dissertação.

2 REFERENCIAL TEÓRICO

Neste referencial teórico serão apresentados definições e conceitos relacionados à pesquisa qualitativa, análise secundária, síntese temática, abordagens e métodos adotados neste trabalho, focando sempre que possível no contexto da ES.

Apesar do foco principal deste trabalho ser o método de síntese temática e o mesmo mostrar-se útil para outros temas relacionados à área de ES, é importante considerar que a sua aplicação prática foi realizada para o tema *code smell*. Embora, o aprofundamento do debate sobre *code smell* esteja fora do escopo deste trabalho, é necessário explorar algumas definições e conceitos a fins de orientação.

2.1 PESQUISA QUALITATIVA

Sob o ponto de vista da abordagem do problema, a pesquisa pode ser classificada como qualitativa ou quantitativa, sendo estas interligadas e complementares.

A pesquisa quantitativa considera que tudo pode ser quantificável, ou seja, pode ser traduzido em números, opiniões e informações para classificá-las e melhor analisá-las. Esse tipo de abordagem além de requerer o uso de recursos e de técnicas estatísticas (percentagem, desvio-padrão, média, moda, mediana, coeficiente de correlação, análise de

regressão etc.), normalmente envolve a formulação de hipóteses e classificação da relação existente entre as variáveis (Cook e Reichardt, 1979).

O outro tipo de abordagem é conhecido como pesquisa qualitativa, foco desta dissertação. Este tipo de pesquisa considera que há uma relação dinâmica entre o mundo real e o sujeito, isto é, um vínculo indissociável entre o mundo objetivo e a subjetividade do sujeito que não pode ser facilmente traduzido em números. Considerada uma pesquisa descritiva, onde os pesquisadores tendem a analisar seus dados indutivamente, esta abordagem foca no processo e seu significado. A interpretação dos fenômenos e a atribuição de significados são fatores básicos no processo de pesquisa qualitativa e esta não requer o uso de métodos e técnicas estatísticas (Cook e Reichardt, 1979). O pesquisador é o instrumento-chave deste tipo de abordagem e os dados são coletados em seu ambiente natural.

A origem da pesquisa qualitativa vem da Antropologia e da Sociologia, justamente áreas que apresentam o propósito principal de realçar características e atributos da vida social e tem como principal objetivo abordar os valores, as crenças, os hábitos, as representações, as opiniões e atitudes. Desde o final do século XIX e início do século XX, esse termo começou a ser difundido em outras áreas para análise fenomenológica, como, por exemplo, nas áreas da Enfermagem, Psicologia, Educação, Administração de Empresas e ES (Cruzes e Dyba, 2011a).

Considerada por Barroso *et al.* (2003), como sendo a denominação para uma complexa e interconectada família de termos, conceitos e hipóteses, provenientes de várias disciplinas e áreas de conhecimento, esta pesquisa ficou genericamente conhecida como a atividade que coloca o observador no mundo, através de práticas interpretativas, tornando-o visível.

Por apresentar um caráter exploratório, descritivo e indutivo e para que seja possível esta iteração do observador com o mundo, podem ser comumente utilizadas algumas representações, incluindo a análise de estudos primários ou mesmo secundários, estudos de caso, entrevistas, discussão em grupos ou conversas, filmagens, registros de memórias pessoais, artigos, jornais, dentre outros. Segundo Dalfovo *et al.* (2008), existem ainda outras formas mais adequadas para a coleta e análise dos dados considerando este tipo de pesquisa, sendo algumas delas: análise documental (relatórios, cartas, impressos e diários, por exemplo), história de vida e observação participante.

O contato com o ambiente, objeto de estudo em questão, é mantido de forma direta pelo pesquisador, onde questões são analisadas imersas no ambiente em que elas se apresentam sem qualquer manipulação intencional do mesmo.

A pesquisa qualitativa busca o entendimento de um assunto especifico por meio de descrições, comparações e interpretações dos dados, ao contrário da pesquisa quantitativa que utiliza valores numéricos (Cook e Reichardt, 1979). Os dados coletados nessas pesquisas são descritivos, retratando o maior número possível de elementos existentes na realidade estudada, preocupando-se muito mais com o processo do que com o produto. Na análise dos dados coletados, não há preocupação em comprovar hipóteses previamente estabelecidas, porém estas não eliminam a existência de um quadro teórico que direcione a coleta, a análise e a interpretação dos dados.

Seaman (1999) considera a possibilidade de surgirem argumentos relacionados ao fato do comportamento humano ser um dos poucos fenômenos que apresenta complexidade que justifique aplicação de método qualitativo para analisá-lo. Isto leva a crer que qualquer outro fator pode ser adequadamente explicado através de estatísticas e outros métodos quantitativos. Porém, para Engenharia de Software, Seaman afirma que, devido à mistura dos aspectos técnicos e comportamentais humanos, a combinação dos métodos qualitativos e quantitativos apresenta-se como uma solução viável para melhorar o aproveitamento dos pontos fortes de ambos.

Anualmente, diversos trabalhos voltados à pesquisa qualitativa são publicados, sendo a maioria deles relacionados a novas ferramentas, métodos e técnicas para auxiliar na análise e interpretação dos dados. A interpretação dos dados apresenta-se, nesse contexto, como o cerne da pesquisa qualitativa, apresentando diferentes aspectos dependentes da abordagem.

Cassel e Symon (1994) destacam algumas características básicas da pesquisa qualitativa, tais quais: o fato de não apresentar foco na quantificação, apresentando-se centrada na interpretação que os participantes possuem quanto à situação investigada; o fato de enfatizar a subjetividade e a flexibilidade no processo de condução da pesquisa, orientando-se para o processo e não para o resultado; e o fato de preocupar-se com o contexto, reconhecendo que o pesquisador influencia a situação de pesquisa e é por ela também influenciado.

Existe uma diversidade de propósitos e técnicas dentre os trabalhos qualitativos das mais variadas áreas de atuação, mas Godoy (1995) enumerou um conjunto de

características essenciais para a identificação de uma pesquisa com esta abordagem, sendo elas:

(1) Ambiente natural como sendo a fonte direta dos dados e o pesquisador como sendo o instrumento fundamental;
(2) Caráter descritivo;
(3) O processo é o foco principal de abordagem e não o resultado ou o produto;
(4) A análise dos dados é realizada de forma intuitiva e indutivamente pelo pesquisador (enfoque indutivo);
(5) Não requer o uso de técnicas e abordagens estatísticas;
(6) Apresenta preocupação maior com a interpretação de fenômenos e a atribuição de resultados.

Segundo Campos (1984), existia uma dificuldade inerente à aplicação de pesquisas qualitativas e esta se aclarava na necessidade de delimitação de critérios e passos, ou seja, na ausência expressa de uma sistematização metodológica e procedimentos apropriados escritos na literatura que norteasse a aplicação da pesquisa. Alguns trabalhos mais recentes, em áreas distintas, vêm tentando resolver esta lacuna. Para a área de ES, foco desta dissertação, percebe-se que existem trabalhos que apresentam uma forma sistemática de aplicar alguns métodos qualitativos (Cruzes, 2007; Cruzes e Dyba, 2011a; Cruzes e Dyba, 2011b).

Cook e Reichardt (1979) propõem uma tipologia para a distinção entre métodos quantitativos e qualitativos, listando as principais características das duas abordagens, conforme Tabela 1.

Tabela 1 Tipologia para distinção entre os métodos quantitativos e qualitativos adaptado de (Cook e Reichardt, 1979)

Abordagem	**Descrição**
Quantitativa	- Analisam o comportamento humano, do ponto de vista do ator, utilizando a observação naturalista e não controlada; - São subjetivos e estão perto dos dados (perspectiva de dentro), orientados ao descobrimento; - São exploratórios, descritivos e indutivos; - São orientados ao processo e assumem uma realidade dinâmica; - São holísticos e não generalizáveis.
Qualitativa	- São orientados à busca da magnitude e das causas dos fenômenos sociais, sem interesse pela dimensão subjetiva e utilizam procedimentos controlados; - São objetivos e distantes dos dados (perspectiva externa), orientados à verificação, apresentando-se de forma hipotético-dedutiva; - Assumem uma realidade estática;

	- São orientados aos resultados, replicáveis e generalizáveis.

Segundo Pope *et al.* (2006), embora haja uma distinção quanto à forma e à ênfase, não seria correto considerar os métodos qualitativos e quantitativos como opostos, nem tão pouco como métodos excludentes. Pelo contrário, a associação dos métodos mencionados traz diferentes visões da realidade, podendo melhorar o entendimento do fenômeno que está sendo analisado. Seaman (1999) também afirma que qualquer problema de engenharia de software é investigado com maior profundidade quando usamos uma combinação de métodos qualitativos e quantitativos.

Desde 1979, Jick já se referia à combinação de métodos qualitativos e quantitativos citando o trabalho de Campbell e Fiske (1959), que desde esta época já faziam menção a utilização conjunta de ambos os métodos, com a denominação de "validação convergente" ou "multimétodo". Para Jick esta combinação foi denominada de "triangulação".

Para Morse (1991) o uso de métodos qualitativos e quantitativos ao mesmo tempo foi denominado de "triangulação simultânea", deixando claro que na fase de coleta dos dados a interação entre os dois métodos não é tão intensa, mas na fase de análise e resultados essa relação se fortalece e os métodos tornam-se complementares para que os objetivos da pesquisa sejam atingidos. Morse ainda sugere a "triangulação sequenciada", método no qual os resultados obtidos com a aplicação de um método são utilizados como base para o planejamento do outro, sendo que ambos se complementam.

Diversos benefícios podem ser percebidos com a aplicação conjunta dos métodos, visto que seus objetivos não se sobrepõem, mas se completam, enriquecendo a visão do pesquisador e tornando os resultados da pesquisa mais fortes e abrangentes. Duffy (1987) apresenta como sendo alguns destes benefícios mencionados:

(1) Possibilidade de congregar controle dos vieses (pelos métodos quantitativos) com compreensão da perspectiva dos agentes envolvidos no fenômeno (pelos métodos qualitativos);

(2) Possibilidade de congregar identificação de variáveis específicas (pelos métodos quantitativos) com uma visão global do fenômeno (pelos métodos qualitativos);

(3) Possibilidade de completar um conjunto de fatos e causas associados ao emprego de metodologia quantitativa com uma visão da natureza dinâmica da realidade;

(4) Possibilidade de enriquecer constatações obtidas sob condições controladas com dados obtidos dentro do contexto natural de sua ocorrência;
(5) Possibilidade de reafirmar validade e confiabilidade das descobertas pelo emprego de técnicas diferenciadas.

2.2 ANÁLISE SECUNDÁRIA EM ESE

Como nos diversos campos da ciência baseados em experimentação, a Engenharia de Software Experimental contém dois tipos de investigação: estudos primários e secundários (Yamashita e Moonen, 2013c).

Segundo Cruzes (2007), estudos primários são dirigidos para avaliar diretamente a hipótese formulada pelo pesquisador, testando-a sob condições estabelecidas no controle metodológico observacional ou experimental. Já estudos secundários são aqueles produzidos para estabelecer comparações, generalizações sistemáticas ou mapeamento das investigações individuais, selecionadas cientificamente de um conjunto de estudos primários (Cruzes, 2007).

Com o aumento da experimentação em ES, o interesse por estudos secundários nesta área também tem se mostrado maior (Creswell, 2007; Wohlin *et al.*, 2000). Esse contexto favorece o aparecimento de métodos para síntese de estudos experimentais em ES.

Cruzes (2007) sumariza alguns métodos de síntese que tem sido utilizados em ES (não todos necessariamente utilizados em estudos secundários). Estes métodos originam-se geralmente de outras áreas de conhecimento e são adaptados para o contexto da ES. Na Tabela 2 encontra-se a sumarização feita por Cruzes (2007) para a síntese de evidências qualitativas e quantitativas, acrescida de alguns métodos estudados ao longo da pesquisa realizada para esta dissertação.

Tabela 2 Resumo de métodos para síntese de evidências adaptado de (Cruzes, 2007)

Método	**Esboço da Abordagem**	**Problemas**	**Vantagens**
Sumário Narrativo	Descrição narrativa e ordenação de evidências primárias (talvez selecionadas) com comentário e interpretação.	Falta de transparência na seleção e também nos estágios posteriores do processo.	Procedimentos flexíveis; pode lidar com grandes bases e também com diversos tipos de evidência.

Grounded theory	O método comparativo constante identifica padrões e inter-relacionamentos em dados primários; a amostragem responde à análise; os princípios da amostragem teórica podem ser usados.	Falta de transparência.	Busca por explanações/ teorias generalizadas. Podem incluir tanto evidências qualitativas quanto quantitativas.
Codificação ou *Coding*	Método muito utilizado na etapa de análise e interpretação dos resultados, logo após a realização das coletas dos experimentos primários, através de técnica de codificação.	Falta de definição do grau de formalidade na estrutura dos dados de codificação.	Oferece um procedimento analítico, por meio do qual os dados qualitativos são divididos, conceitualizados e integrados para formar uma teoria.
Análise qualitativa e comparativa	Método de análise booleana das condições necessárias e suficientes para que os resultados particulares sejam observados, baseadas na presença/ausência das variáveis e dos resultados em cada estudo primário.	Focado na determinação da causalidade, aspectos não interpretativos dos dados qualitativos.	Sistemático transparente; pode incorporar evidências qualitativas e quantitativas.
Meta análise Bayesianas	Método embasado pela opinião quantificada sobre efeitos das variáveis dos estudos qualitativos formalmente combinados (com o paradigma de Bayes) com a evidência dos estudos quantitativos.	Conceitualmente simples, mas pode ser tecnicamente complexo para executar (e pode perder assim a transparência).	O impacto da opinião prévia dos analistas pode explicitamente ser explorado.
Metassíntese ou Metaestudo	Corresponde a uma síntese interpretativa de achados qualitativos, oriundos de estudos fenomenológicos, etnográficos, da teoria fundamentada nos dados e outros.	Não há consenso quanto à terminologia para descrever o processo.	Vão além da soma das partes, uma vez que fornecem uma nova interpretação dos resultados encontrados. Esta nova interpretação são inferências derivadas das análises dos estudos primários.
Meta-análise	Método de combinação estatística dos dados por meio dos estudos para gerar a estimativa sumária dos efeitos. O termo "efeito" refere-se a qualquer medida de associação entre a exposição e o resultado (por exemplo, relação das probabilidades). Uma meta análise é geralmente a etapa final em uma revisão sistemática.	Os dados extraídos de estudos primários devem satisfazer vários requisitos (sendo o mais importante ter um alto nível de homogeneidade entre os estudos).	Pode ser feito sem uma revisão sistemática. Menos consumidor de tempo do que a revisão sistemática.
Contagem de Votos	Método onde os resultados diferentes dos testes das hipóteses são categorizados como significativamente positivos, significativamente negativos, ou sem efeito significativo. Cada estudo conta um "voto" na sustentação dos relacionamentos acima e os números dos votos são contados. Se a razão entre votos positivos e negativos for maior que um limite predeterminado, um relacionamento para a	Não depende dos valores reais do tamanho do efeito e de medidas comparáveis. O método supõe que há um fenômeno comum subjacente, por exemplo, quando um único coeficiente de correlação é aplicado. Não considera a qualidade dos estudos.	É mais simples do que meta-análise e pode ser aplicado tanto para estudos qualitativos quanto quantitativos.

	variável específica é identificado.		
Síntese Temática	Abordagem que utiliza os princípios da análise temática frequentemente utilizada para identificação, análise e identificação de padrões (temas) dentro de dados observados em uma pesquisa qualitativa primária, tendo como saída um modelo de alto nível dos temas encontrados.	Nem sempre os estudos primários tratam das questões da revisão sistemática que está sendo feita. Desta forma, torna-se necessária a aplicação da inovação conceitual que seriam novas proposições geradas pelos revisores, à luz da síntese realizada com finalidade de atender a essas questões.	Por utilizar a técnica de codificação permite a tradução de conceitos entre os estudos, apresentando um processo claro e transparente, além de permitir uma visão gráfica do mapeamento dos temas identificados, facilitando a compreensão e a confiabilidade dos resultados obtidos.
SecESE	Abordagem definida para análise secundária em ESE, baseada na contagem de votos e formalização de atividades.	Apresenta necessidade de aprimoramento das técnicas aplicadas e de algumas das fases propostas, sendo necessária a realização de mais estudos sobre sua utilização.	Específica para ESE, permitindo formalização da análise pela execução de um processo repetível, podendo utilizar tanto dados qualitativos como quantitativos para a análise.
Etnografia	Abordagem focada na observação e investigação, permitindo o aumento da percepção em relação aos costumes e práticas adotadas por determinados grupos que estão sendo analisados.	Requer permissão do grupo que está sendo analisado para que a observação possa ser feita, bem como exige imparcialidade do pesquisador para não haver interferência nos resultados.	Coleta de dados in loco, permitindo ao pesquisador escolher a técnica de análise que irá utilizar.
Meta-etnografia	Abordagem utilizada para integração interpretativa de resultados qualitativos provenientes de uma RS, MS ou um conjunto de estudos etnográficos, permitindo que traduções interpretativas ampliadas de todos os estudos analisados em determinado domínio sejam criadas, de modo que seu resultado seja fiel à tradução interpretativa de cada estudo em particular que foi considerado na síntese.	Existem preocupações expressas em relação à perda de contexto explicativo, quando os resultados de vários estudos são combinados para análise e interpretação dos dados.	Pode ajudar a gerar teorias mais abrangentes e generalizáveis, proporcionando um nível mais elevado de análise, preservando as propriedades de interpretação dos dados primários.

Existem estudos realizados com base na combinação de alguns dos métodos para síntese de evidências, acreditando que os resultados podem agregar mais à área ao qual se referem. Um exemplo de abordagem definida para o contexto de análise secundária especificamente para a Engenharia de Software Experimental é a abordagem SecESE, proposta por Cruzes (2007). O foco desta abordagem é combinar o que há de melhor em dois dos métodos de síntese de evidências, sendo eles contagem de votos e revisões sistemáticas, evidenciando a repetibilidade do processo e formalização de atividades (como controle de qualidade dos artigos).

Segundo Cruzes (2007), uma das vantagens da SecESE é a formalização e garantia de qualidade da análise pela execução de um processo repetível, além do fato da abordagem analisar os resultados num nível de abstração que traz o benefício de inclusão, já que tanto resultados qualitativos como quantitativos podem ser utilizados para a composição da análise. Essa abordagem usa métodos explícitos e reproduzíveis que são executados sistematicamente (de acordo com um procedimento definido) e abertamente (assegurando que o procedimento da análise seja visível e reproduzível por outros pesquisadores).

A Figura 2 apresenta a abordagem SecESE, com suas quatro fases (planejamento da análise, avaliação de qualidade, extração de informações e análise dos resultados) e seus quatro respectivos resultados (detalhes da análise, avaliações de qualidade, resultados e contextos, conclusões).

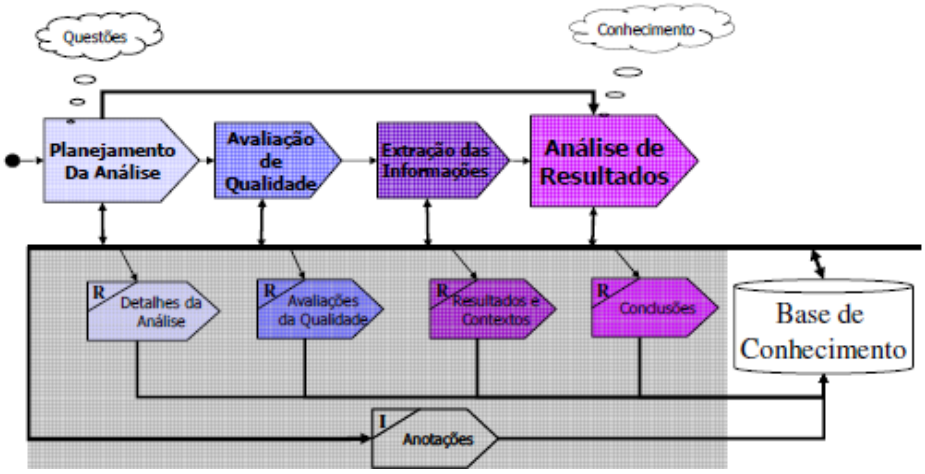


Figura 2 Abordagem SecESE para Análise de Resultados em ES extraída de (Cruzes, 2007)

As fases desta abordagem são:

- Definição e Planejamento da Análise: são definidos os detalhes da análise que será realizada e seus objetivos, possibilitando que os artigos sejam selecionados para análise com base nestes objetivos. Após isso, os parâmetros iniciais são definidos, tais como, limitações, expectativas e metodologia aplicada na análise. É utilizada a abordagem baseada em GQM para a definição dos objetivos de análise mencionados nesta etapa;
- Avaliação de Qualidade: nessa etapa os artigos são detalhadamente avaliados, considerando os critérios de avaliação definidos em uma lista de verificação dos fatores que necessitam ser avaliados para cada estudo;

- Extração de Informações: permite a criação de resultados a partir das informações disponíveis nos dados ou relatórios extraídos de estudos primários, tendo como objetivo criar um conjunto de resultados a partir das informações disponíveis. Foi desenvolvida a ferramenta InfoESE para subsidiar a coleta das informações dos estudos primários;
- Análise de Resultados: compreende as atividades de formalização dos resultados e interpretação dos resultados, tendo como objetivo permitir a análise dos resultados obtidos dos artigos, juntamente com as informações de contexto deles. A ferramenta escolhida para exploração hierárquica da base de resultados foi a "*Treemap*".

Durante todo o processo as anotações são registradas e formalizadas para a composição da base de conhecimentos. O insumo para esse processo são as questões disponíveis sobre determinado tema dentro do contexto de Engenharia de Software Experimental e a saída, ao final do processo sistemático proposto pela SecESE são as conclusões, permitindo a criação de uma base estruturada de resultados e informações de contexto desses resultados que viabiliza a expansão dessa base de resultados através do seu reuso.

2.2.1 Revisão Sistemática (RS)

A ESBE pode fornecer mecanismos necessários para auxiliar os profissionais na adoção de adequadas tecnologias, evitando as inadequadas. Com isto busca-se a aplicação de melhores práticas e procedimentos para a realização das tarefas inerentes à área.

Segundo Dyba *et al.* (2005), as evidências existentes sobre uma determinada tecnologia, na ESBE, são agrupadas em cinco etapas, sendo:

1. Transformar o problema ou necessidade de informação em uma questão de pesquisa;
2. Pesquisar na literatura por melhores evidências disponíveis para responder às perguntas;
3. Avaliar criticamente as evidências, quanto a sua validade, impacto e aplicabilidade;

4. Integrar as evidências avaliadas à prática da ES;
5. Avaliar o desempenho das etapas anteriores e buscar formas de melhorá-los.

Geralmente as etapas 2 e 3 acima mencionadas são realizadas através de uma RS, que é um dos principais métodos empregado pela ESBE.

Tendo as suas raízes oriundas da área médica, a RS é uma abordagem de revisão da literatura que vem sendo aplicada recentemente em diversas áreas da comunidade científica da computação. Este fato deve-se à sua capacidade explícita de prover um estudo detalhado e abrangente sobre o assunto ao qual se apresenta um interesse de pesquisa.

Na atualidade, considerando o grande número de produções científicas sobre um determinado tema, RS apresentam-se como uma boa oportunidade para a captação, reconhecimento e possibilidade de síntese das evidências científicas. Os resultados adquiridos mostram-se abrangentes e não tendenciosos. Este fato deve-se à RS ser um estudo secundário e como tal, depende dos estudos primários conduzidos para poder agregar evidências e construir conhecimento (Biolchini e Travassos, 2007; Dyba, Dingsoyr, Hanssen, 2007; Kitchenham e Charters, 2007).

O planejamento e execução de uma RS devem ser um processo cuidadoso para assegurar a validade de seus resultados, visto que a qualidade dos estudos primários interfere na qualidade de estudos secundários (Biolchini e Travassos, 2007; Dyba, Dingsoyr, Hanssen, 2007; Kitchenham e Charters, 2007).

Considerando as declarações de Kitchenham (2004), a RS é uma abordagem que busca identificar, avaliar e interpretar todas as pesquisas relevantes disponíveis relacionadas com uma questão de pesquisa, um tópico ou um fenômeno de interesse. Existem algumas razões que levam à necessidade de realizar RS na literatura, dentre elas: sumarizar evidências existentes sobre um determinado fenômeno, identificar lacunas existentes na pesquisa atual, fornecer um arcabouço para posicionar novas pesquisas, apoiar a geração de novas hipóteses (Kitchenham, 2004).

De uma forma geral a RS é uma abordagem rigorosa proposta para:

- Identificar os estudos sobre um tema em questão, aplicando métodos explícitos e sistematizados de busca;
- Avaliar a qualidade e validade desses estudos, assim como sua aplicabilidade no contexto onde as mudanças serão implantadas. O intuito é selecionar os estudos que fornecerão as evidências científicas e, disponibilizar a sua síntese, com vistas a facilitar as descobertas.

Cada um desses momentos é planejado no protocolo da RS considerando critérios que os validam, para minimizar o viés e outorgar qualidade à metodologia. Os procedimentos desenvolvidos em cada momento devem ser registrados para possibilitar que a RS seja reproduzida e conferida por outros pesquisadores.

Se compararmos a RS com uma revisão bibliográfica feita de maneira ad hoc, pode-se perceber o quão mais demorado e trabalhoso o primeiro se apresenta (Montebelo, 2007). Isto pode ser explicado pelo fato da RS envolver a execução de passos definidos e sistemáticos, conforme mencionado anteriormente.

Outras características são mencionadas por Kitchenham (2004) para diferenciar uma revisão convencional da literatura de uma RS:

- Existe a definição de um protocolo de avaliação que especifica a questão de pesquisa a ser abordada e os métodos que serão utilizados para realizar a revisão.
- São baseadas em uma estratégia de busca como parte do protocolo.
- Essa estratégia de busca é explicitamente documentada para que os leitores possam auditar ou replicar.
- Existem critérios de inclusão e exclusão explícitos para avaliar os estudos que farão parte da revisão.
- Definição de critérios de qualidade que permitam avaliar os estudos que farão parte da revisão.
- A revisão sistemática é um pré-requisito para a meta-análise quantitativa.

Ainda considerando Kitchenham (2004), a RS apresenta três fases: Planejamento, Execução e Documentação da Revisão, descritas na Tabela 3.

Tabela 3 Objetivos e etapas de cada fase da RS extraído de (Kitchenham, 2004)

Fase	**Descrição**	**Resumo**
Planejamento	Definir o objetivo e planejar a RS.	- Identificar a necessidade da RS; - Definir os objetivos da pesquisa; - Definir o protocolo da revisão; - Validar o protocolo da revisão.
Execução	Executar o planejamento feito no protocolo, buscar estudos primários e selecionar os estudos primários para serem sintetizados.	- Identificar pesquisas relevantes; - Selecionar os estudos primários de acordo com os critérios de inclusão e exclusão estabelecidos; - Avaliar a qualidade dos estudos de acordo com os critérios estabelecidos; - Extrair informações dos estudos primários selecionados; - Sintetizar dados dos estudos primários selecionados.
Documentação	Documentar a revisão, gerando o	- Escrever Relatório da Revisão;

da Revisão	relatório da revisão que contém a síntese das informações dos estudos primários que atendem ao propósito da revisão.	- Validar Relatório.

Esta forma sistemática de realizar uma RS tenta evitar as principais limitações de uma revisão informal. Como exemplo podemos citar a introdução de eventuais vieses na escolha de documentos e extração de informações, ou a dificuldade de repetição do estudo pela falta de documentação do seu protocolo de execução (Cruzes, 2007). Desta forma é utilizada uma metodologia rigorosa, confiável e passível de auditagem (Kitchenham, 2004).

A primeira fase do processo de condução para uma revisão sistemática é o planejamento da revisão e nela pode-se citar a construção do protocolo como o grande cerne da questão. O protocolo de uma RS definido garante que a mesma seja conduzida com o mesmo rigor sistemático das pesquisas científicas, aumentando assim, a sua confiabilidade e possibilidade de repetição por outros pesquisadores.

Os componentes do protocolo que precisam ser definidos nesta etapa são: a(s) questão(ões) de pesquisa, os critérios de inclusão/exclusão, as estratégias para buscar as pesquisas, como as pesquisas serão avaliadas criticamente, a coleta e síntese dos dados.

A formulação adequada das questões de pesquisa, considerando os objetivos da mesma apresenta importante relevância quando estamos tratando de uma RS, principalmente porque proporciona o direcionamento para a execução das demais atividades relativas ao processo. Uma pergunta mal formulada pode comprometer todo o processo de uma RS e seus resultados.

Segundo Counsell (1997), as perguntas guiam a revisão, pois definem quais serão os estudos incluídos, quais serão as estratégias adotadas para identificar os estudos e quais serão os dados que necessitam ser coletados de cada estudo identificado.

Para que a busca dos estudos se torne uma atividade possível é imprescindível que se defina as bases de dados e a string de busca que serão trabalhados.

Segundo Hamer e Gill (2005), a utilização de uma estratégia ampla de busca dos estudos consiste em uma procura nas bases eletrônicas de dados, envolvendo também a busca manual em periódicos, as referências listadas nos estudos identificados, contato com os pesquisadores e, se possível, até mesmo o encontro de material não publicado.

Ao adotar as bases eletrônicas de dados que serão utilizadas na busca dos estudos deve haver uma preocupação e cuidado com a necessidade de buscar os estudos em mais de uma base de dados. Isto também implica na necessidade de habilidade na forma correta de procurar em cada uma dela (Hamer e Gill, 2005), considerando as suas particularidades.

A utilidade de uma RS depende em grande parte da qualidade dos estudos analisados. Para Hamer e Gill (2005), a avaliação da qualidade é uma etapa da RS onde todos os estudos selecionados são avaliados com rigor metodológico. O propósito é averiguar se os métodos e resultados das pesquisas são suficientemente válidos para serem considerados.

Algumas ferramentas podem ser utilizadas para auxiliar as fases definidas pela execução de uma RS, suportando todos os passos descritos anteriormente e ainda permitindo a geração de relatórios sobre o tópico pesquisado. Um exemplo seria a ferramenta StArt (Fabbri, 2012; Zamboni *et al.*, 2010) utilizada para a RS mencionada nesta dissertação.

No contexto da RS é importante salientar também que a mesma pode estar ancorada em pesquisas qualitativas ou quantitativas, dependendo do objetivo do estudo e das perguntas que nortearão a pesquisa, conforme representação da Figura 3.

Na Figura 3, podemos perceber que:

- Para a abordagem quantitativa, caso seja necessária uma síntese com análise estatística é indicado a realização de uma revisão sistemática com meta-análise. Já para os casos em que não seja necessária uma síntese estatística, é recomendada a revisão sistemática descritiva;
- Para a abordagem quantitativa e qualitativa em conjunto é necessária uma abordagem integrativa para amparar as duas necessidades explicitadas;
- Para a abordagem qualitativa, foco desta dissertação, a realização de síntese é recomendada. E a escolha do método que será utilizado depende dos objetivos atribuídos à revisão sistemática realizada.

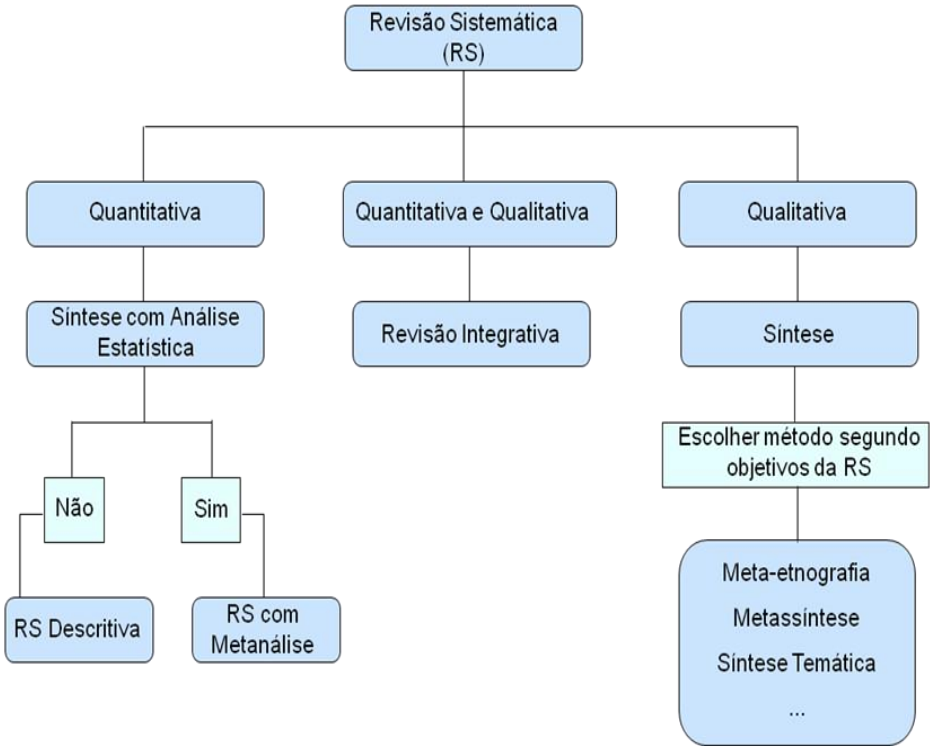


Figura 3 Métodos para síntese de evidência científica adaptado de (De-La-Torre-Ugarte *et al.*, 2011)

O foco da pesquisa quantitativa é descrever variáveis quanto à tendência central, dispersão e frequência (Richardson. 2008), enquanto a preocupação da pesquisa qualitativa é a necessidade de entendimento do contexto no qual um fenômeno ocorre, considerando a descrição dos comportamentos percebidos. Estes dois tipos de pesquisa são diferentes, mas não excludentes, podendo ser associadas quando o objetivo é realizar uma revisão integrativa, por exemplo (Richardson. 2008).

Na Revisão Sistemática qualitativa, observa-se a diversidade dos métodos que podem ser aplicados, permitindo a síntese dos resultados obtidos. A despeito de cada método apresentar as suas particularidades, eles podem ser complementares ou mesmo apresentar características que se justapõem. Há métodos que priorizam a construção ou explicação de teorias e aqueles voltados a descrever um determinado fenômeno (Counsell, 1997). Alguns exemplos de métodos são: Metassíntese (Paterson, 2001), Metaetnografia (Campbell *et al.*, 2003; Noblit e Hare, 1988) e Síntese Temática (Cruzes e Dyba, 2011a; Cruzes e Dyba, 2011b).

A escolha do método está intrinsecamente ligada aos objetivos aos quais se pretende atingir com o estudo, ou seja, o julgamento de valor atribuído a um determinado

método deve estar diretamente relacionado à sua capacidade de nos aproximar da realidade pesquisada.

Geralmente quando é realizada uma RS com o foco quantitativo, a documentação da revisão gera uma síntese que pode ser descritiva ou por meta-análise. Já quando o foco da RS é qualitativo, é interessante a adoção de um método que permita a síntese dos dados com um foco interpretativo.

Desta forma, parece fazer sentido ampliar ao máximo as fontes de busca para as RS com foco quantitativo, enquanto que nas RS com foco qualitativo, é importante a execução de uma seleção detalhada das fontes imprescindíveis ou mais relacionadas à temática de estudo (De-La-Torre-Ugarte *et al.*, 2011). Para a abordagem qualitativa, é importante ajustar e dimensionar a capacidade de análise do investigador à quantidade de artigos disponíveis sobre essa temática (De-La-Torre-Ugarte *et al.*, 2011). Este cuidado deve-se ao fato de que o número elevado de artigos dificulta o aprofundamento da análise, podendo constituir-se em ameaça na validação da RS qualitativa (Sandelowski e Barroso, 2007).

Segundo Petersen *et al.* (2008), com o amadurecimento de uma área de pesquisa, o número de estudos e resultados relacionados tende a crescer. Sendo assim, torna-se importante a realização de RS para que o acesso a estes estudos seja mais bem direcionado, aumentando a sistematização das buscas.

A literatura diferencia alguns tipos de RS:

- Revisões Sistemáticas convencionais (Petticrew, Roberts, 2006): agregam resultados sobre a eficácia de um determinado tratamento, intervenção ou tecnologia. Apresenta-se como um dos principais métodos de pesquisa da ESBE, trabalhando com estratégia de pesquisa e critérios de aceitação que permitem que as evidências pertinentes sejam consideradas de forma sistemática e transparente.
- Mapeamento Sistemático (Arksey, O'Malley, 2005): também conhecido como Estudo de Escopo, objetiva identificar todas as pesquisas relacionadas a um tópico específico, promovendo uma visão geral sobre determinada área de pesquisa, ou seja, mapear uma determinada área de pesquisa de forma a descobrir o que está sendo estudado e o que precisa ser estudado.

2.2.2 Mapeamento Sistemático (MS)

O MS é um tipo de revisão sistemática. Nele uma revisão mais ampla dos estudos primários é realizada em busca de identificar quais evidências estão disponíveis. Lacunas no conjunto os estudos primários também são identificadas com o objetivo de direcionar o foco de revisões sistemáticas futuras e identificar áreas onde mais estudos primários precisam ser conduzidos (Kitchenham, 2004).

O estudo de MS fornece uma visão geral de uma área de pesquisa. Através dele podemos identificar a quantidade, os tipos de pesquisas realizadas, os resultados disponíveis, além das frequências de publicações ao longo do tempo, percebendo tendências (Petersen, 2008).

De acordo com Kitcheham e Charters (2007), as principais diferenças entre uma RS convencional e um MS são as seguintes:

- Geralmente, MS são guiados por múltiplas questões de pesquisa, apresentando-se de forma mais amplas, ou seja, mais exploratória;
- As questões de pesquisa de uma RS são mais focadas do que as questões de pesquisa de um MS. Isto facilita que um número maior de estudos seja considerado durante a realização de MS, mas isso não gera um problema, dado o foco de um MS que é ter uma ampla cobertura sobre a área de pesquisa;
- O processo de extração de dados para um MS é muito mais amplo do que do processo de uma RS e pode ser chamado de “Classificação” ou “Estágio de categorização”. O objetivo deste processo é classificar os artigos com riqueza de detalhes que se mostrem suficientes para responder às questões de pesquisa com foco exploratório, sem permitir que se torne um processo muito extenso;
- Na etapa de análise em um MS é feita uma síntese dos dados para que as questões de pesquisa trabalhadas possam ser respondidas. Não é muito comum que técnicas de análise profundas sejam aplicadas, como por exemplo, meta-análise ou síntese narrativa. Gráficos são gerados das distribuições dos estudos por tipo de classificação, resultados mais focados em contagens e resumos;

- A divulgação dos resultados de um MS pode ser mais limitada do que uma RS; limitada aos que requisitam o trabalho e a publicações acadêmicas, com o objetivo de influenciar o rumo das pesquisas primárias.

Para Askey e O'Malley (2005) a execução de um MS, dentre outras questões, pode estar relacionada à identificação de uma necessidade em realizar uma revisão sistemática convencional. Nestes casos, um mapeamento preliminar da literatura pode ser realizado para identificar a viabilidade ou relevância na execução da RS. No entanto é importante perceber que identificar lacunas na literatura através da realização de um MS não é necessariamente identificar lacunas na pesquisa em si. Ou seja, identificar se uma pesquisa tem qualidade pobre, uma vez que a avaliação da qualidade não é o principal foco de um estudo deste tipo.

Segundo Dyba *et al.* (2005) revisões sistemáticas e mapeamentos sistemático possuem em comum as seguintes características:

1. Identificação da necessidade de executar uma revisão sistemática;
2. Elaboração de um protocolo formal de pesquisa;
3. Busca abrangente e exaustiva por estudos primários;
4. Avaliação de qualidade dos estudos considerados;
5. Identificação dos dados necessários para responder às questões de pesquisa;
6. Extração dos dados;
7. Resumo e síntese dos resultados dos estudos (meta-análise);
8. Interpretação dos resultados, auxiliando a analisar a sua aplicabilidade;
9. Escrita do relatório.

O protocolo detalhado de pesquisa deve ser previamente definido para evitar a possibilidade de o pesquisador enviesar a pesquisa e possibilitar a replicação do estudo por outros pesquisadores. Na definição deste protocolo de pesquisa, as questões também devem ser definidas. Uma vez produzido o protocolo, o MS pode ser iniciado.

Além dos passos citados anteriormente é interessante ressaltar a possibilidade de complementação dos resultados obtidos através de uma busca manual do material relevante para o tema em estudo, considerando os principais anais de conferências e periódicos. Esta busca manual pode favorecer na ampliação da cobertura da RS, bem como a descobertas de estudos relevantes que não usam os termos da *string* de busca.

Segundo Kitchenham e Charters (2007), a etapa final do MS é a criação e apresentação de mapas contendo as evidências encontradas e que caracterizam a área de

pesquisa, permitindo uma avaliação mais sistemática da mesma. Os resultados são apresentados em um nível de granularidade mais elevado, de forma a responder às questões de pesquisa trabalhadas. Desta forma, descobrimos os *gaps* de temas e conceitos não trabalhados dentro da área pesquisada, favorecendo o direcionamento do foco de futuras RS e áreas necessitadas da realização de mais estudos primários.

2.3 CODIFICAÇÃO

A técnica de codificação conhecida como *Coding* é muito utilizada na etapa de análise e interpretação de resultados, tendo aplicação comum logo após a realização das coletas de experimentos primários.

Nesta dissertação utilizamos a referida técnica na etapa de codificação dos dados prevista no método proposto por Cruzes e Dyba (2011a).

Para Seaman (1999) esta técnica auxilia a preparação de dados qualitativos para serem analisados quantitativamente.

Segundo Corbin e Strauss (2008), esta técnica oferece um procedimento analítico, por meio do qual os dados qualitativos são divididos, conceitualizados e integrados dando possibilidade à formação de teorias.

Cruzes e Dyba (2011a) identificam uma etapa do seu método proposto para realização de síntese temática em ES onde a codificação é realizada. Para estes autores, códigos são rótulos descritivos aplicados a segmentos de texto para cada estudo, mas não se resumem a isto, visto que uma codificação bem feita requer uma noção clara do contexto em que descobertas são feitas.

Na técnica *Coding*, os trechos dos textos em análise são rotulados e combinados, de acordo com a ideia presente, o tema e categoria, permitindo que futuramente essas informações relacionadas a determinados assuntos possam ser recuperadas (cruzes e Dyba, 2011a). Essa organização e categorização facilitam as buscas e comparações das informações existentes nos artigos selecionados, permitindo tanto que padrões possam ser identificados, quanto que ideias possam ser refutadas frente às divergências encontradas.

Saldaña (2008) divide o processo de codificação em dois ciclos básicos, sendo eles:

1. Primeiro ciclo: os códigos são atribuídos a pedaços de textos selecionados nos artigos;
2. Segundo ciclo: trabalham com o resultado da codificação encontrada do primeiro ciclo.

Para cada ciclo inúmeras abordagens são apresentadas com o intuito de facilitar a aplicabilidade do processo de codificação.

A abordagem de codificação descritiva, por exemplo, propõe o resumo em uma palavra ou frase curta o tema básico de uma passagem de dados qualitativos. O método categoriza dados em um nível básico para proporcionar ao pesquisador uma maior compreensão do estudo. Aplica-se a todos os estudos qualitativos, principalmente etnografia que lida muito com dados longos e complexos de entrevistas.

A Tabela 4 apresenta exemplos de codificação para melhor entendimento desta técnica. Sendo assim, um dos códigos apresentados poderia ser considerado pelo pesquisador para codificar o bloco de texto em questão.

Tabela 4 Exemplo de codificação extraído de (Saldaña, 2008)

Texto	**Possibilidade de códigos atribuídos**
Simplesmente não há lugar neste país para imigrantes ilegais. Cerque-os e os envie de volta.	- Questões de imigração - Xenofobia

Geralmente os métodos de pesquisa qualitativa utilizam essa técnica de codificação após a etapa de coleta dos dados, considerando as vantagens mencionadas acima. Se considerarmos o grande volume de artigos que pode constituir a etapa inicial de coleta, percebe-se a importância da adoção de uma técnica que apresente uma forma mais sistemática para a análise dos dados.

Exatamente por essa questão, essa técnica, quando realizada de forma totalmente manual, acaba por exigir muito do pesquisador, dependendo do tamanho da amostra a ser trabalhada. Desta forma, a adoção de um processo sistemático, pautado em revisões constantes para tentar dirimir as inconsistências e possíveis falhas passa a ser o ponto chave para a confiabilidade dos resultados alcançados (Saldaña, 2008).

2.4 SÍNTESE TEMÁTICA

Conforme exposto anteriormente, existe uma série de métodos que podem ser aplicados às Revisões Sistemáticas para a realização de sínteses qualitativas, como por exemplo, a metaetnografia, metassíntese, a análise temática e a própria síntese temática.

Síntese temática é o termo criado por Thomas e Harden (2008) para referenciar o método definido pelos autores com base na análise temática. A intenção dos autores era auxiliar o suporte teórico, propondo um método mais claro e preciso, composto por três passos sistemáticos. Para tanto, desenvolveram uma abordagem que combinava e adaptava a metaetnografia e grounded theory.

Para as definições do método, os autores aplicaram seus estudos em uma avaliação das facilidades e obstáculos inerentes à alimentação saudável de crianças. Neste estudo, os autores identificavam códigos livres relacionados às constatações que obtiveram e as organizaram em temas, através de comparações constantes. Inicialmente estes temas tinham caráter mais descritivo e, posteriormente, eram repensados de maneira mais analítica.

Em 2011, Cruzes e Dyba analisaram o método proposto por Thomas e Harden e criaram o método de síntese temática focado em ES, contendo cinco passos sistemáticos. Para estes autores, o termo Síntese é comumente utilizado para descrever uma família de métodos para sumarização, integração, combinação e comparação de resultados de diferentes estudos primários que estão interessados em questões ou tópicos relacionados às pesquisas (Cruzes e Dyba, 2011b). Estes geram estudos secundários que podem agregar resultados novos não percebidos nos estudos primários de forma independente, auxiliando na identificação de teorias e possíveis generalizações.

A Figura 4 apresenta uma síntese dos cinco passos propostos por Cruzes e Dyba (2011a).

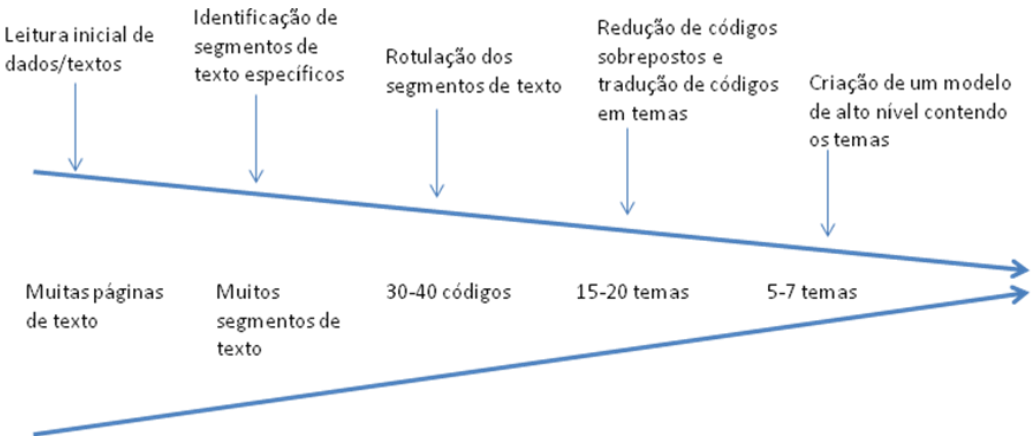


Figura 4 Método de síntese temática adaptado de (Creswell, 2007; Cruzes e Dyba, 2011)

A Figura 4 apresenta a relação entre os passos que são propostos e os resultados que são obtidos a cada passo. Na leitura inicial dos textos, pode-se perceber que muitas páginas de texto são lidas na medida em que o método conduz a uma leitura completa dos artigos selecionados. Para identificação de segmentos de texto específicos, o resultado são alguns segmentos de texto extraídos. O passo seguinte menciona a rotulação dos segmentos de texto, resultando em códigos livres. Para a tradução dos códigos em temas, os códigos sobrepostos são reduzidos, gerando os temas. O último passo refere-se à criação do modelo de mais alto nível contendo os temas já tratados que representam a amostra analisada.

Uma das etapas essenciais de qualquer trabalho científico é a avaliação dos resultados descobertos (Richardson. 2008). Através das descobertas são facilmente extraídas as contribuições e limitações.

Considerando Cruzes e Dyba (2011b), pode-se mencionar que estudos relacionados a ES são frequentemente muito heterogêneos, o que dificulta somente a aplicação de relações estatísticas, comprovadas por vezes pelo uso de meta-análise. Portanto, o uso de métodos de síntese qualitativa e métodos mistos apresentam-se como uma solução para um melhor entendimento da área.

Segundo Cruzes e Dyba (2011a), a síntese temática é uma abordagem que utiliza os princípios da análise temática frequentemente utilizada para identificação, análise e identificação de padrões (temas) dentro de dados observados em uma pesquisa qualitativa primária.

Uma das possíveis saídas de um processo de síntese temática é o Mapa Temático. O mapa temático é um modelo contendo os temas de mais alto nível, os subtemas e seus respectivos códigos que permite a visualização de uma forma mais gráfica, possibilitando a

exploração das relações entre os temas identificados. Além do mapa temático, uma síntese temática pode gerar uma taxonomia, uma visão mais descritiva dos temas encontrados ou mesmo uma teoria relacionada ao assunto principal tratado na síntese.

Detalhes do método de síntese temática adotado como referência para esta dissertação serão apresentados no Capítulo 3.

2.5 ABORDAGEM GQM

GQM (*Goal*, *Question*, *Metrics*) é uma técnica baseada em métricas de software promovida por Basili e disponível na Encyclopaedia of Software Engineering (Van Solingen, 2002). Esta técnica inclui a definição de elementos que precisam ser analisados em relação ao objetivo do estudo, às questões relacionadas e às métricas, favorecendo que a mesma seja utilizada e/ou adaptada para o uso em propósitos distintos no processo científico (Jedlitschka, Ciolkowski, Pfahl, 2008). Para este trabalho, esta técnica é importante por ter sido utilizada na codificação dos objetivos dos artigos analisados na síntese temática realizada.

Como exemplo da utilização na abordagem GQM para o contexto científico, pode-se citar a descrição do objetivo da pesquisa ou, considerando Wohlin *et al.* (2000), a "Definição do Experimento". Considerando que este objetivo de pesquisa deve ser o mais coerente possível, a aplicação da técnica apresenta-se como auxilio para o sucesso desta tarefa (Van Solingen, 2002).

Essa técnica inclui alguns elementos a serem preenchidos, conforme mencionado no Quadro 1:

Analyze <…> for the purpose of <…> with respect to their <…> from the point of view of the <…> in the context of <…>.

Quadro 1 Template GQM extraído de (Jedlitschka, Ciolkowski, Pfahl, 2008)

Onde:

- *Analyze*: apresenta o foco principal do estudo que está sendo analisado;
- *For the purpose of*: referencia o propósito que embasa o "analyze" para o contexto deste estudo;

- *With respect to their*: explicita os principais objetos do estudo analisado;
- *From the point of*: especifica sob qual ponto de vista considerado pelo estudo que está sendo analisado;
- *In the context of*: auxilia no mapeamento do contexto ao qual o estudo se refere.

Um exemplo de uma adaptação para aplicação desta técnica foi extraído para que possa ilustrar melhor e promover o entendimento mais amplo (Jedlitschka, Ciolkowski, Pfahl, 2008):

- Analyze perspective-based reading and ad hoc reading techniques;
- For the purpose of evaluation;
- With respect to their effectiveness;
- From the viewpoint of potential users;
- In the context of the software engineering class at the University.

Sjøberg (2005) realizou um estudo com a finalidade de caracterizar os tópicos dos experimentos, seus participantes e as tarefas executadas durante o experimento. Considerando os resultados obtidos neste estudo chega-se à conclusão de que existe uma fragilidade clara que demonstra que os artigos de Engenharia de Software Experimental são geralmente vagos. Ou seja, os artigos desta área não se apresentam de forma sistemática. Outro fator mencionado no estudo foi o de que não existe uma terminologia consistente nos relatos dos experimentos.

Diante dos resultados encontrados pelo estudo pode-se compreender que nem sempre a caracterização dos estudos e experimentos vai nos permitir uma visão tão clara a respeito dos aspectos necessários ao mapeamento da técnica GQM, dificultando a aplicação da técnica. Estes fatores são limitações dos estudos e desfavorecem, por exemplo, a realização de sínteses em estudos secundários (Sjøberg, 2005).

2.6 DESIGN DE SOFTWARE ORIENTADO A OBJETOS

Segundo Sommerville *et al.* (1995), o *design* orientado a objetos (OO) é uma estratégia de design na qual os engenheiros de sistemas pensam em termos de coisas ao invés

de operações ou funções. A execução do sistema é feita por meio da interação de objetos que proveem operações e informações.

A qualidade do design de software OO representa um impacto relevante no processo de desenvolvimento de sistemas. Em função deste impacto alguns pesquisadores têm se preocupado em definir técnicas para apoiar a definição de design de qualidade e mecanismos de detecção.

Basicamente, os autores desta linha de pesquisa propõem estratégias para identificação de aspectos no código que tenham quebrado os princípios do paradigma de orientação a objetos. Em um dos primeiros livros sobre o tema, Riel (1996) usou sua experiência para discutir problemas comuns observados. Ele abordou o uso indevido de relacionamento entre as classes, herança, a relação de contenção de classes e atributos. O autor também apresentou heurísticas que podiam ser utilizar na tentativa de evitar os problemas mencionados.

2.6.1 Code Smells

Termos como "*design flaws*" (Riel, 1996), "*code smell*" (Fowler, 1999), "*disharmony*" (Lanza, Marinescu, 2005), anomalias de código (Macia *et al.*, 2012b) são frequentemente utilizados para definir potenciais problemas de *design*. Neste trabalho, foi adotado o termo "*code smells*", ou simplesmente "*smells*", para designar este tipo de problemas de *design*.

O termo *code smell* foi designado por Fowler (1999) para descrever os potenciais problemas de *design*. O autor apresentou vinte e dois tipos de *code smells*, a partir de uma discussão informal, apresentando técnicas possíveis de refatoração para eliminá-los. Segundo o autor, é importante melhorar a estrutura interna (*design*), evitando problemas futuros, especialmente com manutenção. Como *design* pode-se entender questões relacionadas à organização das classes do sistema e como elas se relacionam entre si. O autor apresentou várias técnicas de refatoração, permitindo a difusão na aplicabilidade das mesmas.

Segundo Marinescu (2001), existem problemas na detecção manual de *code smells* tratada por Fowler. Estes problemas remetem a questões como: quantidade de tempo gasto, ausência de repetição e ausência de escalabilidade (Marinescu, 2001). Devido a estas

questões, Marinescu (2001) trouxe uma proposta baseada em métricas para a detecção de *code smells*. A intenção da utilização das métricas era adotar um mecanismo para avaliar e controlar a qualidade do *design*. A dificuldade inerente à utilização destas métricas na identificação de possíveis problemas de *design* está relacionada à dificuldade de obter interpretações adequadas. Esta dificuldade termina por limitar a relevância dos resultados obtidos com a aplicabilidade das métricas de software.

Para tratar estas limitações das métricas de software, Lanza e Marinescu (2005) propuseram mecanismo para formular regras baseadas em métricas que capturam desvios em relação aos princípios e heurísticas do bom *design* OO, ou seja, code *smells*. Marinescu (2004) chama este mecanismo de *estratégias de detecção.*

As estratégias de detecção ajudam os desenvolvedores de software a detectar e localizar *code smells.*

Lanza e Marinescu (2005) adotaram uma estratégia de detecção de *smells* baseada em métricas e limites. Para definir suas regras, os autores utilizam os resultados e informações disponibilizadas por Riel (1996) e Fowler (1999). A partir deles, Lanza e Marinescu identificaram métricas que poderiam ser afetadas por *code smells*. Em seguida, eles definiram os limites.

Um exemplo do trabalho de Lanza e Marinescu (2005) seria relacionado ao *code smells* conhecido como *God Class.* Este se refere a classes que tendem a centralizar a inteligência dos sistemas. Lanza e Marinescu elegeram as métricas ATFD (*Acess to Foreign Data*), WMC (*Weighted Contagem Method*) e TCC (*Tight Class Cohesion*) para auxiliar na identificação deste *smell*.

Onde:

- ATFD: conta o número de atributos de outras classes acessados, diretamente ou utilizando métodos de acesso, pela classe avaliada;
- WMC: é a soma das complexidades de todos os métodos de uma classe;
- TCC: conta o número relativo de métodos de uma classe que acessam pelo menos um atributo comum.

A Figura 5 apresenta a estratégia de detecção para o exemplo de *god class* citado anteriormente. Na Figura pode-se perceber que uma classe é considerada como *god* class se apresenta:

- A métrica ATFD maior do que poucas outras classes; e
- A métrica WMC muito alta; e

- A métrica TCC menos que um terço das outras classes.

Os valores-limites são previamente definidos de acordo com outros atributos do sistema, como tamanho, por exemplo. Se as métricas e os limites são bem definidos, o *code smell* pode ser automaticamente detectado. A Figura 5 apresenta de forma resumida o que pretendemos mostrar na estratégia de detecção de *God Class*.

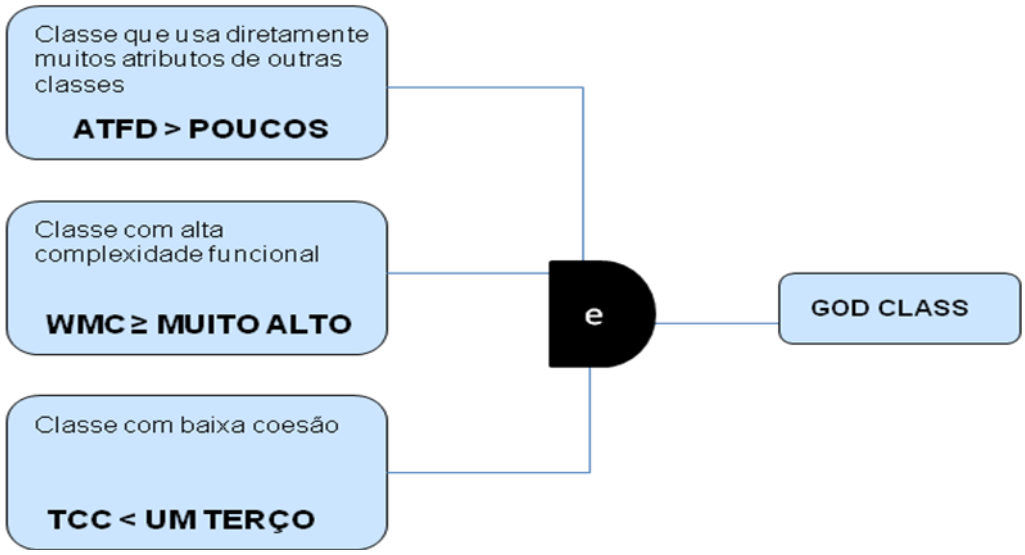


Figura 5 Estratégia de Detecção de God Class adaptado de (Lanza, Marinescu, 2005)

O trabalho Lanza e Marinescu (2005) é base de muitas pesquisas e ferramentas focadas na detecção automática de *smells*.

2.7 CONCLUSÃO DO CAPÍTULO

Este capítulo apresentou conceitos, definições e características da pesquisa qualitativa, focando em síntese temática e nas diversas técnicas e abordagens que podem ser utilizadas em conjunto para atingir o objetivo de um trabalho com este foco. Como a aplicação prática do método proposto por esta dissertação terá como tema central *code smells*, alguns conceitos básicos relacionados ao tema foram citados neste Capítulo.

No Capítulo seguinte apresentaremos o método de síntese temática proposto pelos autores Cruzes e Dyba, trazendo também uma das contribuições deste trabalho que são os

Checklists sistemáticos. Estes foram criados a partir das recomendações dos autores para possibilitar a visão prática, auxiliando o entendimento e norteando a aplicação do método.

Capítulo 3

Este Capítulo apresenta o método para elaboração de síntese temática em ES proposto por Cruzes e Dyba (2011a). Os autores definem o que deve ser feito e em raros casos, como na criação do modelo de alto nível, sugerem como deve ser feito.

3 MÉTODO DE SÍNTESE TEMÁTICA PARA ENGENHARIA DE SOFTWARE

O método de síntese temática proposto como resultado do trabalho de Cruzes e Dyba apresenta um conjunto extenso de recomendações em forma textual e um checklist, que indica alguns aspectos práticos para realização de cada etapa. Com esta forma de apresentação, a aplicação do método requer que a utilização do checklist seja complementada com as informações textuais, onde as recomendações estão distribuídas.

Neste Capítulo, propomos uma nova forma de apresentação dos resultados e recomendações realizadas por Cruzes e Dyba. Para tanto, o método foi entendido e sistematizado para a criação de um novo checklist com formato de tabela. Este novo checklist geral proposto no Capítulo visa simplificar a aplicação sistemática do método, sendo utilizado nos Capítulos 4 e 5 desta dissertação.

3.1 MÉTODO DE SÍNTESE TEMÁTICA

O método de síntese temática proposto por Cruzes e Dyba (2011a) foi definido para a área de ES em cinco passos, permitindo que pesquisadores e estudiosos realizem sínteses de uma forma mais sistemática.

A Tabela 5 apresenta um guia para a utilização do método, definido por Cruzes e Dyba. Nesta, cada um dos cinco passos sistemáticos é individualmente explicitado e um

checklist contendo as principais questões que precisam ser garantidas pelo pesquisador, por etapa, é apresentado. A finalidade deste *checklist* é apoiar a execução do método, aumentar a sua confiabilidade e permitir que outros pesquisadores possam repeti-lo.

Tabela 5 Etapas do Método de Síntese Temática e Checklist extraído de (Cruzes e Dyba, 2011a)

Etapa	**Descrição**	***Checklist***
Extração de Dados	Extrair dados de estudos primários, incluindo informação bibliográfica, objetivos, contexto e resultados.	1. Todos os artigos foram lidos para facilitar a imersão no contexto e nos dados? 2. Foram identificados segmentos de texto que ajudam a identificar os objetivos do artigo? 3. Detalhes da publicação, descrições de contexto, e resultados foram extraídos dos artigos selecionados? 4. Outro pesquisador verificou a extração?
Codificação de Dados	Identificar e codificar conceitos interessantes, categorias, descobertas e resultados de uma forma sistemática em todo o conjunto de dados.	5. Os segmentos importantes do texto como conceitos, categorias, descobertas e resultados foram rotulados e codificados? 6. A codificação foi feita em um nível apropriado para as questões de pesquisa? 7. Lista de códigos iniciais com definições e frequências foram criadas e controladas por outro pesquisador? 8. Verificações de consistência ou de confiabilidade entre pesquisadores foram realizados para estabelecer a credibilidade da codificação? 9. Existem ligações claras e evidentes entre o texto e os códigos?
Tradução de Códigos em Temas	Traduzir códigos em temas, subtemas e temas mais abrangentes.	10. Os temas foram criados a partir de uma visão geral, inclusiva e compreensiva dos códigos de todos os artigos? 11. Códigos sobrepostos foram reduzidos e os códigos restantes foram traduzidos em temas? 12. Os temas foram verificados uns contra os outros e foi feito batimento dos mesmos com os dados dos artigos originais? 13. Os temas são coerentes, consistentes e distintos?
Criação do Modelo de Alto Nível	Explorar relações entre temas e criar um modelo com os temas mais abrangentes.	14. Existem descobertas interpretadas que fazem mais sentido do que se fosse somente parafraseada ou descrita? 15. Os temas e as relações entre os temas foram comparados e verificados com as conclusões dos estudos primários? 16. Existe uma descrição clara das relações entre os temas mais abrangentes? 17. Foi criado um modelo para apresentar as relações existentes entre os temas mais abrangentes?
Avaliação da Confiabilidade da Síntese	Avaliar a confiabilidade das interpretações que antecederam a síntese temática.	18. As suposições sobre uma abordagem específica para a síntese temática foram claramente explicadas? 19. Existe um bom ajuste entre o que é reivindicado e o que a evidência mostra? 20. A linguagem e os conceitos utilizados na síntese são consistentes? 21. As questões de pesquisa foram respondidas com base na evidência da síntese temática?

3.2 RESUMO DAS ETAPAS DE UMA SÍNTESE TEMÁTICA

Nesta Seção os cinco passos considerados são explicitados em um maior nível de detalhes. Ao final de cada passo apresentado será disponibilizado um *checklist* em formato de tabela sumarizando as recomendações apresentadas por Cruzes e Dyba.

Tanto as recomendações dispostas ao longo do texto pelos autores, quanto o *checklist* por eles proposto foi utilizado como insumo para a sumarização do *checklist* proposto nesta dissertação. Esta sumarização será a base para as discussões a respeito da aplicabilidade prática do método de síntese temática.

3.2.1 Extração de Dados

Esta etapa aparece como o primeiro passo do método de síntese temática. Segundo Cruzes e Dyba (2011a), trata-se de uma etapa fundamental da RS, na qual os dados provenientes dos estudos primários são obtidos de forma explícita e coerente de acordo com a estratégia de extração definida.

Os autores recomendam que haja uma leitura completa dos artigos candidatos à extração, dada a importância do aprofundamento e necessidade de imersão no conteúdo. Esta leitura pode remeter os pesquisadores à necessidade de revisar o protocolo definido para a revisão sistemática.

A extração dos dados foi adaptada da técnica proposta por Cruzes *et al.* (2007), para explorar evidências de uma RS. Esta técnica provê subsídio para que o pesquisador siga um procedimento sistemático para a identificação de informações de contexto e resultados dos artigos selecionados.

Cruzes e Dyba (2011a) propõem um arcabouço de extração contendo dimensões e tipos de dados relevantes ao método de síntese. Apresentamos este arcabouço na Figura 6. Esta apresenta as dimensões consideradas, os tipos de dados relevantes para cada dimensão e a relação existente entre estas dimensões. Para auxiliar na interpretação, pode-se entender que

uma publicação está associada a um contexto específico e um contexto pode estar associado a um ou mais resultados apresentados pelos artigos.

Figura 6 Arcabouço de Extração adaptado de (Cruzes e Dyba, 2011a)

Cruzes e Dyba (2011a) mencionam que durante a leitura inicial, detalhes importantes que sejam percebidos nos artigos selecionados já devem ser destacados ou transferidos para uma ferramenta de análise qualitativa de dados.

Os autores comentam que os detalhes da publicação são geralmente simples de serem extraídos. O objetivo dos artigos é mencionado como um dado da dimensão publicação que pode não apresentar-se de forma clara. Para esses casos é necessária a realização de um trabalho mais analítico para identificação dos objetivos.

Para o contexto, os autores ressaltam a importância para que o pesquisador foque nas informações que o ajudarão na compreensão e interpretação dos resultados do estudo em questão.

Considerando a extração dos resultados dos artigos, as recomendações são relacionadas à observação das principais fontes das informações, que são as seguintes seções: as que descrevem os resultados, as que apresentam a análise dos resultados, discussão e conclusões. Figuras e tabelas também devem ser consideradas como fontes de informação sobre os resultados.

Um conjunto de perguntas é proposto para auxiliar na identificação de declarações que podem ser consideradas como constatações (Cruzes e Dyba, 2011a):

- Indica resultados das medições?
- Resumem os dados brutos?
- Destacam alguma característica específica dos dados brutos?
- Fornecem *insights* sobre tabelas ou figuras apresentadas?
- Resumem os resultados das análises?

- Pode ser usado para responder às questões de pesquisa apresentadas?
- Refletem os principais resultados do estudo?

Os autores alertam que a extração de informações de contexto são as mais desafiadoras, visto que alguns artigos não fornecem dados suficientes sobre os projetos e as conclusões dos estudos.

A última recomendação é que, sempre que possível, a extração seja realizada por dois ou vários pesquisadores, de forma independente. Caso dois pesquisadores trabalhem nas extrações de forma independente, seus resultados devem ser comparados buscando resolver as discordâncias por meio de consenso ou definição oriunda de um terceiro pesquisador adicional e independente (Cruzes e Dyba, 2011a).

Resumo das recomendações para a etapa de extração

A partir do resumo analítico realizado sobre esta etapa, extraímos as recomendações apresentadas por Cruzes e Dyba (2011a) em forma de itens sumarizados. A Tabela 6 apresenta o *checklist* contendo as recomendações/aspectos relevantes detectados nesta análise. Na primeira coluna, destacamos as recomendações. Na segunda coluna, apresentamos tópicos relacionados à recomendação principal, quando aplicável.

Tabela 6 Checklist da Etapa de Extração de Dados

Recomendações/Aspectos Relevantes	**Subitens**
1. Leitura completa dos artigos para facilitar imersão no contexto e dados	1.1. Identificação e destaque de segmentos de texto importantes
2. Arcabouço contendo conjuntos de dados para nortear extração dos dados	2.1. Verificar se os objetivos estão claros 2.2. Caso não estejam, realizar trabalho de análise para identificação dos mesmos (objetivos) 2.3. Detalhes da publicação, descrições de contexto, e resultados foram extraídos dos artigos selecionados
3. As principais fontes das informações devem ser buscadas nas seções: que descrevem os resultados, que apresentam a análise dos resultados, discussão e conclusões	Não se aplica
4. Figuras e tabelas devem ser consideradas	Não se aplica
5. Perguntas sugeridas devem ser empregadas quando da identificação de uma declaração que pode ser considerada como uma constatação	Não se aplica
6. Extração do contexto é mais desafiadora	6.1. Ausência de informações completas atrapalha a extração do contexto
7. Sempre que possível, a extração deve ser realizada por dois ou vários pesquisadores, de forma	7.1. Verificação da extração por parte de outro pesquisador

independente	7.2. Discordâncias devem ser tratadas por consenso ou definição oriunda de um terceiro pesquisador adicional e independente

3.2.2 Codificação de Dados

Para Cruzes e Dyba (2011a), os códigos devem ser vistos como rótulos descritivos, aplicados aos segmentos de texto extraídos de cada estudo para lhe conferir um significado específico. Sendo assim, os autores deixam claro que o processo de codificar é muito mais do que apenas a inserção de um rótulo, pois envolve análise e organização dos dados contidos em cada estudo primário da RS. Para tanto, os autores reportam que deve ser considerada a exigência de conhecimento claro do contexto do estudo e do que realmente é relevante para o assunto que está sendo trabalhado.

Os dados são primeiramente extraídos para depois serem codificados. Neste contexto, os autores explicitam que no momento da realização da extração, o pesquisador deve extrair todos os segmentos de texto que exemplificam a mesma ideia, já permitindo a organização e o agrupamento de dados semelhantes em categorias. É salientada uma recomendação relacionada à RS que prevê que é mais indicado trabalhar com a codificação de blocos de texto e este fato referencia o início da identificação dos temas.

Há ainda a recomendação de que a codificação seja refinada com a categorização dos temas e comparações constantes, considerando o conhecimento dos pesquisadores a respeito do assunto em questão.

Baseando-se nas questões de pesquisa, o pesquisador foca nas descobertas isoladas ou na combinação de descobertas e informações dos estudos primários.

São apresentadas, pelos autores, três abordagens distintas para a condução da codificação. Sendo elas:

- Dedutiva: envolve a geração de uma lista inicial contendo códigos previamente formatados originários da teoria, questões de pesquisa, hipóteses, áreas problemáticas e/ou variáveis chave. Estes códigos preliminares podem ajudar os pesquisadores a integrar os conceitos já bem

conhecidos na literatura, mas sempre com o cuidado de evitar forçar dados dos estudos analisados apenas para cobrir as categorias pré-existentes;

- Indutiva: os códigos surgem para o pesquisador e são atribuídos após uma análise dos dados extraídos, sem uma listagem prévia dos possíveis códigos que podem ser trabalhados para um determinado conceito. Esses códigos são refinados à medida que novos dados são inseridos e analisados. Para verificar se um código foi adequadamente atribuído, o pesquisador pode fazer uso da comparação dos segmentos de texto extraídos com a finalidade de avaliar e decidir se estes refletem o mesmo conceito. Usando este método de comparação constante, os pesquisadores refinam as dimensões dos códigos existentes e podem identificar novos códigos;
- Integrada: encontra-se entre a abordagem indutiva (de baixo para cima) e a abordagem dedutiva (lista preliminar de códigos). Esta abordagem prevê a criação de um esquema geral de códigos sem um conteúdo específico, mas que aponta para domínios gerais nos quais os códigos propriamente ditos poderão ser trabalhados de forma indutiva. Este esquema geral ajuda o pesquisador a pensar em categorias macro, nas quais os códigos serão trabalhados.

Os autores alertam pata alguns problemas típicos que podem surgir durante a codificação:

- Codificação ser feita em um nível muito geral;
- Identificar o que você quer ver e não o que o texto está dizendo.
- Codificação fora do contexto.

Para evitar estes problemas é recomendado que os segmentos de texto importantes como conceitos, categorias, descobertas e resultados sejam rotulados e codificados durante a execução do método (2011a). Finalizando as recomendações dos autores para este passo, a codificação deve ser revisada para validação dos códigos identificados, evitando redundâncias e possibilitando a contextualização dos mesmos para o assunto em que se está trabalhando.

Resumo das recomendações para a etapa de codificação

A Tabela 7 apresenta o *checklist* contendo as recomendações/aspectos relevantes para a etapa de codificação.

Tabela 7 Checklist da Etapa de Codificação de Dados

Recomendações/Aspectos Relevantes	Subitens
8. Todos os segmentos de texto que exemplifiquem a mesma ideia devem ser extraídos	Não se aplica
9. A categorização dos temas acontece em comparações constantes, considerando o conhecimento dos pesquisadores	Não se aplica
10. Para a codificação em RS é mais indicado trabalhar blocos de texto	10.1. Este fato referencia o início da identificação dos temas
11. A adoção de uma abordagem para condução da codificação é importante: dedutiva, indutiva ou integrada	11.1. Lista de códigos iniciais com definições e frequências deve ser criada e controlada por outro pesquisador
12. Os segmentos importantes do texto, como conceitos, categorias, descobertas e resultados, devem ser rotulados e codificados	Não se aplica
13. A codificação deve ser feita em um nível apropriado para as questões de pesquisa	Não se aplica
14. Verificações de consistência e/ou de confiabilidade entre pesquisadores devem ser realizados para estabelecer a credibilidade da codificação	14.1. Verificar se existem ligações claras e evidentes entre o texto e os códigos

3.2.3 Tradução de Códigos em Temas

Cruzes e Dyba (2011a) explicam que este passo se refere ao processo de considerar como os códigos identificados no passo anterior podem se combinar com a intenção de gerar temas mais abrangentes para o assunto que está sendo abordado. Os autores recomendam que todos os códigos possíveis já sejam de conhecimento dos pesquisadores para se iniciar este passo. Mencionam ainda que códigos sobrepostos devem ser tratados, evitando redundâncias e permitindo o início da identificação dos temas.

Os autores ressaltam que no início do método existe o distanciamento do pesquisador em relação ao texto e que, desta forma, à medida que o método evolui, o nível de abstração aumenta e torna-se possível a generalização dos temas. Para os autores, os códigos identificados devem ser analisados frente a essa necessidade de tradução em temas, formando um ciclo que permite a criação de novos códigos, novos temas, agrupamentos e reclassificações com o intuito de representar melhor os resultados encontrados. Sendo assim, este passo só termina quando o pesquisador esgota as possibilidades de temas (Cruzes e Dyba, 2011a).

O uso de representações visuais é recomendado pelos autores para auxiliar no entendimento e classificação dos diferentes códigos em temas. Alguns exemplos de representações visuais são mapas mentais e mapas temáticos, dentre outros. Na Figura 7, é apresentado um exemplo de mapa temático. Nesta Figura, pode-se perceber a existência de um tópico central, norteando as pesquisas em busca de descobertas. As relações entre os temas e temas de mais alto nível são explicitadas. Também é possível perceber relações secundárias que envolvem temas ligados a outro tema de mais alto nível, ao qual o mesmo não está diretamente relacionado. Esta visão gráfica auxilia os pesquisadores envolvidos a entenderem os resultados obtidos com a síntese, facilitando também a compreensão dos interessados no assunto, que façam uso da mesma.

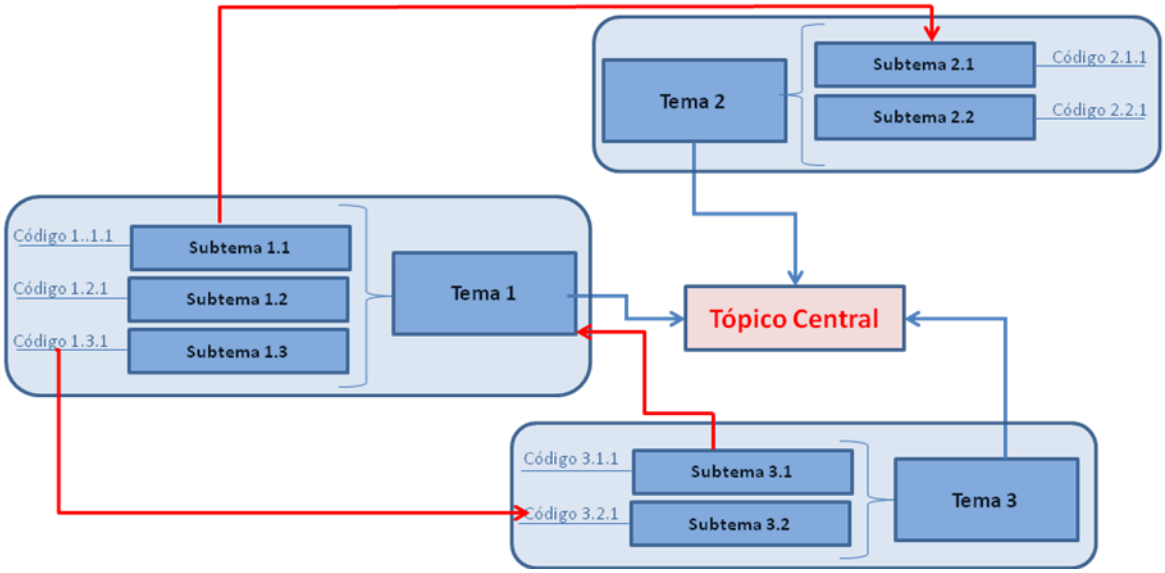


Figura 7 Exemplo de Mapa Temático adaptado de (Cruzes e Dyba, 2011a)

Outra recomendação é de que os temas sejam verificados uns contra os outros e com os dados extraídos dos artigos.

Resumo das recomendações para a etapa de tradução de código em temas

A Tabela 8 apresenta o *checklist* contendo as recomendações/aspectos relevantes para a etapa de tradução de código em temas.

Tabela 8 Checklist da Etapa de Tradução de Códigos em Temas

Recomendações/Aspectos Relevantes	Subitens
15. Os temas devem ser criados a partir de uma visão geral, inclusiva e compreensiva dos códigos de todos os artigos	Não se aplica.
16. Códigos sobrepostos devem ser reduzidos e os códigos restantes traduzidos em temas	Não se aplica.
17. Os temas devem ser verificados uns contra os outros e com os dados dos artigos originais	Não se aplica.
18. Utilização de representações visuais para auxiliar entendimento e classificação dos diferentes códigos em temas e temas mais abrangentes	Não se aplica.

3.2.4 Criação do Modelo de Alto Nível

Cruzes e Dyba (2011a) explicitam que neste passo é importante voltar-se para as questões originais de pesquisa e os interesses teóricos que as sustentam. O objetivo deste passo, segundo os autores, é avaliar os temas descobertos e trabalhados nas etapas anteriores. Esta revisão tem a finalidade de identificar a representatividade dos temas e suas relações. Caso haja necessidade, os temas são reclassificados de uma forma que fiquem mais bem representados (Cruzes e Dyba, 2011a).

Seguindo a linha de raciocínio dos autores, as saídas deste passo podem ser uma visão mais gráfica dos temas e suas relações, uma visão mais descritiva, uma taxonomia ou mesmo uma teoria. Cada uma apresenta sua contribuição quando se pretende fazer uma síntese temática. Eles também mencionam que as questões de pesquisa ajudam a discernir qual a melhor saída para a síntese em questão e que o modelo de alto nível reflete a exploração e interpretação feita a respeito dos temas identificados, fenômenos observados e das relações existentes entre eles.

Os autores percebem e mencionam que existe uma heterogeneidade muito grande nos estudos realizados em ES, seja pela diversidade de descobertas realizadas, pela diversidade de métodos aplicados, de ferramentas ou populações. Por isto, o contexto em que as descobertas dos artigos foram encontradas deve ser levado em consideração no momento em que se pretende criar um modelo que os represente.

Para os casos em que os estudos primários relatam as relações entre as descobertas e o contexto dos estudos, os autores explicitam a importância de comparar e contrastar estas informações, favorecendo as generalizações e síntese. Já para os casos em que o contexto dos estudos não seja apresentado de forma explícita, é recomendada a utilização dos dados previamente extraídos dos estudos primários para identificar as relações entre as descobertas e aspectos-chave dos estudos, comparando-as e contrastando-as. Cruzes e Dyba concordam com Popay *et al.* (2006) quando eles mencionam que este passo apresenta-se demorado e crítico, mas mostra-se necessário para a qualidade da síntese temática.

Alguns passos são sugeridos por Cruzes e Dyba (2011a) para se chegar ao modelo de alto nível. São eles:

1. Revise o modelo de alto nível da última etapa, pegando cada ramo por vez, descrevendo o seu conteúdo com os resultados e o contexto das informações e criando temas de ordem superior;
2. Identifique as ligações entre os temas de mais alto nível e as provas que fundamentam e contexto desta evidência;
3. Explore as conexões com a teoria e pesquisas prévias, contextualize novamente, defina e ainda refine os temas de mais alto nível;
4. Crie um modelo, taxonomia, mapa temático, ou teoria relacionada aos temas de mais alto nível e as suas evidências.

Resumo das recomendações para a etapa de criação do modelo de alto nível

A Tabela 9 apresenta o *checklist* contendo as recomendações/aspectos relevantes para a etapa de criação do modelo de alto nível.

Tabela 9 Checklist da Etapa de Criação do Modelo de Alto Nível

Recomendações/Aspectos Relevantes	**Subitens**
19. Descobertas interpretadas devem fazer mais sentido do que se fosse somente parafraseada ou descrita	Não se aplica
20. Temas e relações entre eles devem ser comparados e verificados com as conclusões dos estudos primários	Não se aplica
21. Deve existir uma descrição clara das relações entre os temas mais abrangentes	Não se aplica
22. As saídas podem ser uma visão mais gráfica dos temas e suas relações, uma visão mais descritiva, uma taxonomia ou mesmo uma teoria	22.1. Deve ser criado um modelo para apresentar as relações existentes entre os temas e temas mais abrangentes 22.2. Revise o modelo de alto nível da última

	etapa, pegando cada ramo por vez, descrevendo o seu conteúdo com os resultados e o contexto das informações e criando temas de ordem superior 22.3. Identifique as ligações entre os temas de mais alto nível e as provas que fundamentam e contexto desta evidência 22.4. Explore as conexões com a teoria e pesquisas prévias, contextualize novamente, defina e ainda refine os temas de mais alto nível
23. O contexto em que as descobertas foram encontradas deve ser considerado	Não se aplica

3.2.5 Avaliação da Confiabilidade da Síntese

Para Cruzes e Dyba (2011a), os resultados da pesquisa devem ser tão confiáveis quanto seja possível e toda pesquisa deve ser avaliada em relação aos métodos utilizados. Sendo assim, para os autores, a avaliação da confiabilidade da síntese é um passo dependente da qualidade dos estudos primários e da quantidade de evidências identificadas.

Alguns fatores são reportados pelos autores como responsáveis por influenciar a confiabilidade da síntese. Dentre eles, os autores destacam a qualidade metodológica de estudos primários considerados na RS e os métodos utilizados na síntese, por exemplo, as medidas tomadas para minimizar o viés.

Conceitos na pesquisa qualitativa que podem ser utilizados para descrever melhor os aspectos relacionados à confiabilidade da síntese são descritos pelos autores. Sendo eles:

- Credibilidade: lida com o foco da pesquisa e refere-se à confiança que pode ser colocada na forma como os dados e processos de análise endereçam o foco pretendido. A credibilidade de uma síntese temática relaciona-se desde o método adotado para a RS conduzida, a etapa de extração, considerando as partes do texto que refletem as questões que estão sendo buscadas e analisadas, bem como a clareza e consistência na definição dos temas e a sua correlação com os códigos identificados.
- Confirmabilidade: preocupa-se com a forma como os dados extraídos são codificados e ordenados. Além disso, considera se pesquisadores e

especialistas concordam com a maneira como estes dados foram codificados e classificados.

- Confiança: preocupa-se com a estabilidade dos dados, o grau de alterações realizadas nos dados ao longo do tempo, e alterações feitas nas decisões do pesquisador durante a síntese. É importante estabelecer uma "trilha de auditoria" sobre o processo realizado, garantindo a rastreabilidade das informações do artigo até a síntese propriamente dita.
- Transmissibilidade: refere-se à possibilidade de transferência dos resultados para outras configurações ou grupos. Os autores podem emitir sua opinião a respeito da possibilidade de transferência, mas é o leitor que define se os dados são transferíveis para outros contextos.

Além dos itens já mencionados que afetam a confiabilidade de uma síntese temática, os autores citam ainda que, para melhorar a transmissibilidade, é importante uma apresentação rica dos resultados com as citações apropriadas.

Para Cruzes e Dyba (2011a), a confiabilidade das interpretações feitas em uma síntese temática está diretamente relacionada com a explanação dos argumentos considerados. Considerando a possibilidade de várias interpretações distintas e a inexistência de um único significado correto, os autores acreditam que a confiabilidade se pauta na forma como os resultados são apresentados, permitindo ao leitor que olhe para interpretações alternativas.

Resumo das recomendações para a etapa de avaliação da confiabilidade da síntese

A Tabela 10 apresenta o *checklist* contendo as recomendações/aspectos relevantes para a etapa de avaliação da confiabilidade da síntese.

Tabela 10 Checklist da Etapa de Avaliação da Confiabilidade da Síntese

Recomendações/Aspectos Relevantes	**Subitem**
24. Considerar os conceitos de credibilidade, confirmabilidade, confiança, transmissibilidade em assuntos relacionados à confiabilidade da síntese	Não se aplica
25. Explanar os argumentos que conduzem as interpretações feitas em uma síntese temática	25.1. As suposições sobre uma abordagem específica para a síntese temática devem ser claramente explicadas 25.2. Deve existir um bom ajuste entre o que é reivindicado e o que a evidência mostra
26. A linguagem e os conceitos utilizados na síntese devem ser consistentes	Não se aplica

27. As questões de pesquisa devem ser respondidas com base na evidência da síntese temática	Não se aplica

3.3 CONCLUSÃO DO CAPÍTULO

Neste Capítulo sistematizamos as recomendações propostas por Cruzes e Dyba. Os *Checklists* contendo a sumarização destas recomendações encontram-se dispostos ao final de cada etapa apresentada e contribuíram tanto para o refinamento do método, quanto para a aplicação prática da síntese temática constantes nos Capítulos 4 e 5, respectivamente. Todas as recomendações dos autores foram numeradas de forma sequencial, considerando cada etapa do método de síntese temática proposto, para a composição destes *Checklists*.

No Capítulo seguinte será apresentado o método de síntese temática refinado. Conseguimos fazer esta proposição, considerando o contexto ao qual aplicamos a nossa pesquisa (estudantes de mestrado e doutorado), a resposta fornecida a cada uma das recomendações apresentadas por Cruzes e Dyba (as perguntas estão sumarizadas neste Capítulo e as respostas foram consideradas no Capítulo 4) e da execução prática que será apresentada no Capítulo 5.

Capítulo 4

Este capítulo apresenta as considerações práticas para a realização de síntese temática em ES. Para tanto, discutimos a sumarização das recomendações e aspectos relevantes apresentados no Capítulo anterior frente à prática realizada.

4 REFINAMENTO DO MÉTODO DE SÍNTESE TEMÁTICA PARA ES

Cada uma das questões sumarizadas no Capítulo 3 será aqui discutida com a visão prática que obtivemos com a execução do método. A numeração sequencial apresentada neste Capítulo, portanto, refere-se à numeração das respostas correspondentes às questões de mesmo número, apresentadas nos Checklists que compõem o Capítulo 3 desta dissertação.

O refinamento do método ocorreu com base nestas recomendações sumarizadas e na experiência prática, obtida com a execução do método, descrita neste Capítulo.

Todos os comentários aqui apresentados são contribuições originais desta dissertação e referem-se à aplicação prática do método proposto por Cruzes e Dyba (2011a), considerando o contexto no qual o trabalho foi realizado. Conforme mencionado no item 1.3 desta dissertação, o trabalho foi realizado por pesquisadores da UFBA que, inicialmente não possuíam experiência no método adotado. Esta foi sendo adquirida à medida que o trabalho evoluía e as atividades eram executadas.

4.1 EXTRAÇÃO DE DADOS

O Capítulo 3 apresenta as Tabelas de 6 a 10, contendo a sumarização das questões propostas por Cruzes e Dyba (2011a). Estas Tabelas apresentam-se sistematizadas sob a forma de Checklists que ajudam no entendimento e aplicabilidade do método. Neste item,

demonstraremos como seguimos cada item contido nestes Checklists, seguindo a numeração sequencial que foi dada a cada questão anteriormente apresentada.

1 - Leitura completa dos artigos para facilitar imersão no contexto e dados:

Nos passos apresentados por Cruzes e Dyba (2011a) é claramente mencionada a importância da leitura completa dos artigos selecionados como uma forma de tentar melhorar a imersão dos pesquisadores no contexto. Consideramos que esta imersão deve ocorrer a partir das etapas iniciais da RS e não somente como parte da etapa de extração de dados da síntese temática. Em função da necessidade de promover um nivelamento no tema central adotado pelos pesquisadores envolvidos, recomendamos algumas medidas a serem tomadas desde o início da filtragem, antes mesmo da extração:

a) Propomos que, antes mesmo de iniciar a remoção dos artigos baseada na leitura do título e abstract, encontro(s) seja(m) realizado(s) com a participação de todos os pesquisadores a fim de promover o nivelamento. Nossa estratégia foi selecionar aleatoriamente cinquenta artigos, realizando leitura em conjunto de todos os artigos (título e resumo). Nesta reunião, os critérios de exclusão adotados foram equalizados, garantindo a compreensão de todos os objetivos traçados para o trabalho. Dúvidas foram mitigadas desde o início.

b) Foi realizada leitura completa dos artigos logo após a etapa de filtragem da RS, visando ratificar a permanência dos mesmos. Neste caso, as discussões consideraram a consonância com os objetivos e questões de pesquisa da RS. Ressaltamos que esta iniciativa extrapolou os ganhos esperados, culminando também em uma maior imersão dos pesquisadores no contexto e nos dados.

c) Outra questão relevante de ser mencionada e que impacta a imersão dos pesquisadores no tema é relacionada ao fato das questões de pesquisa serem refinadas ao longo do método. Apesar de ser comum, especialmente no contexto que estamos considerando em nossa pesquisa, isto aumenta as dificuldades de compreensão clara dos objetivos. Esta dificuldade ocorre especialmente pelo fato de haver pesquisadores menos experientes executando o trabalho. Desta forma, propomos reuniões de discussão nas fases de filtragem, visto que as mesmas já proporcionam alguma imersão no conteúdo.

d) Ainda sobre este tópico, sentimos a necessidade de trabalhar com pequenos grupos dos artigos. Cada grupo seguiu o ciclo de vida adotado no método de síntese temática, favorecendo o amadurecimento dos pesquisadores em relação ao método adotado e ao tema

central. Considerando esta abordagem, os artigos foram separados em etapas com a finalidade de rodar ciclos de extração e codificação, como pode ser percebido na Figura 8. Desta forma, as demais etapas complementavam as descobertas e temas, através de comparações constantes. Apesar de não existir um claro critério definido para a divisão dos artigos em grupos menores neste trabalho, acreditamos que, para trabalhos futuros, estes critérios podem auxiliar na condução da estratégia da extração, equalizando o trabalho realizado.

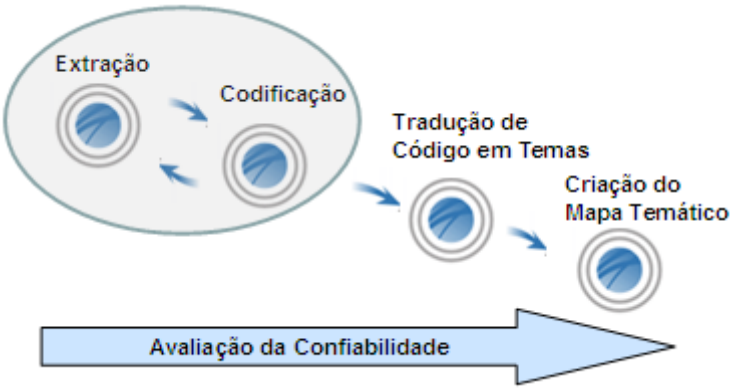


Figura 8 Ciclo de vida adotado no método de síntese temática

2- Arcabouço contendo dimensões (conjuntos de dados) para a extração dos dados:

A nossa recomendação é que as dimensões presentes no arcabouço de extração estão diretamente relacionadas às questões de pesquisa que norteiam a RS. Portanto, para cada nova RS realizada, o arcabouço sugerido pelos autores Cruzes e Dyba deve ser analisado e adaptado, segundo os objetivos e questões de pesquisa da RS.

Considerando a RS realizada para *code smells*, utilizada como base para execução da síntese temática deste trabalho, nossa principal observação é relacionada ao tipo de dado da dimensão chamada de publicação, apresentado com o título de "Objetivos". Na prática tivemos dificuldade de utilizar a extração simples de objetivos para que houvesse entendimento pleno dos propósitos dos artigos. Identificamos inerente a este dado, tanto uma importância relevante em relação ao entendimento completo do que se pretende buscar com os estudos, quanto à existência de outros dados que subsidiavam o entendimento do objetivo dos artigos. Desta forma, adotamos Objetivo como uma nova dimensão composta por seus tipos de dados que favoreçam o aprofundamento no foco dos estudos, melhorando o entendimento nas análises posteriores em comparação com os contextos analisados para

escrita da síntese. Dentre outros dados inerentes à esta dimensão, ressaltamos a consideração das questões de pesquisa/hipóteses dos artigos trabalhados, pois entendemos que estes dados mostraram-se úteis como insumos para a realização da etapa subsequente de codificação dos objetivos, facilitando as comparações entre os artigos. A extração das questões de pesquisa/hipóteses também contribuiu para a identificação da "força da evidência" dos resultados que seriam extraídos para cada artigo. Buscávamos identificar o foco do artigo, ou seja, o que estava sendo apresentado pelos autores, através da análise em conjunto com as respectivas questões de pesquisa/hipóteses. Este fato contribuiu de forma significativa na identificação dos resultados relevantes para o estudo, favorecendo a extração correta e abrangente.

Esta necessidade de apresentar maior confiabilidade na síntese e comparações mais assertivas durante toda a execução do método, nos fez repensar como a dimensão Objetivo poderia auxiliar neste processo de comparações contínuas.

Na Figura 9 pode-se perceber a divisão de duas partes para facilitar o entendimento da análise que se pretende apresentar. Na parte A da Figura é apresentado o arcabouço proposto por Cruzes e Dyba (2011a). Na parte B da Figura apresentamos a nossa adaptação do arcabouço.

Para tentar atingir nossas necessidades de extração, identificamos que a dimensão objetivo precisava conter: ID (identificador do artigo na ferramenta StArt), título (título do artigo), objetivo (este tratado por Cruzes e Dyba, 2011), questões de pesquisa/hipóteses e observações que nos ajudassem a compreender estes objetivos principais dos artigos selecionados (todos extraídos dos artigos oriundos da RS). O ID e o título foram necessários apenas para manter a rastreabilidade com o artigo que estávamos trabalhando, visto que na Planilha de Extração/Mapeamento só constavam os dados extraídos das dimensões Objetivo e Resultado. As demais dimensões foram trabalhadas diretamente na ferramenta desenvolvida em CakePHP.

Na dimensão publicação não consideramos relevante a extração inteira do resumo do artigo e seus tópicos, mas os resumos dos artigos apresentavam dados que foram utilizados nas extrações. Não identificamos, na prática, retorno significativo que justificasse este trabalho, até mesmo pelo fato da extração ser feita de forma manual e do resumo já constar na ferramenta StArt.

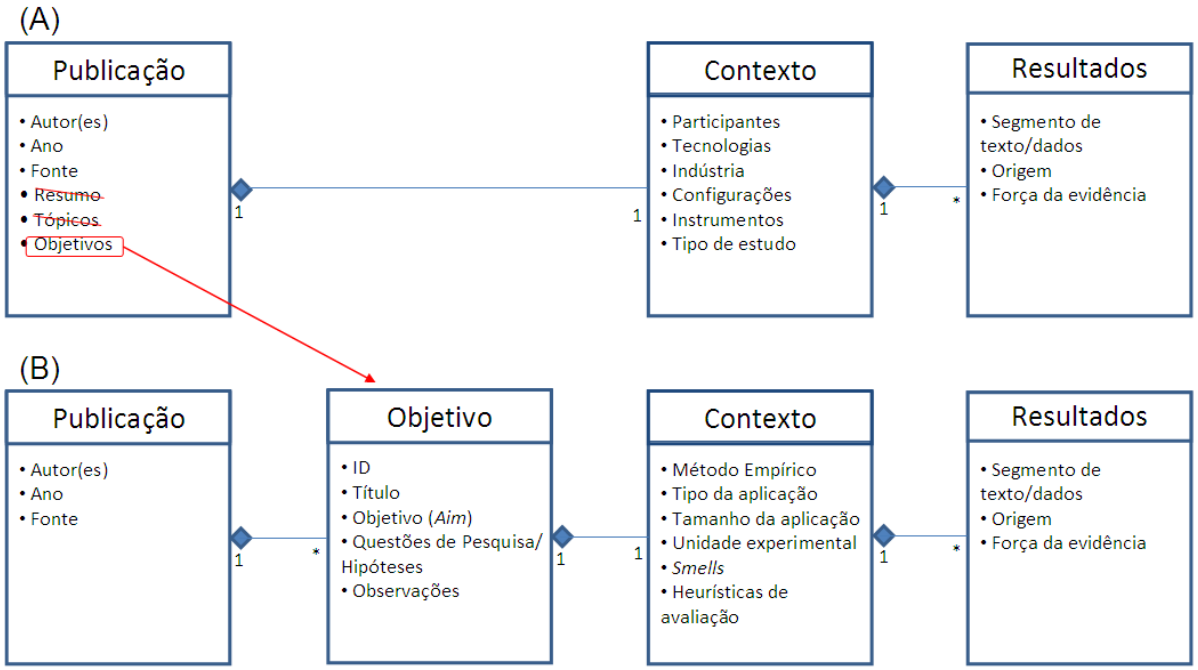


Figura 9 Comparativo dos Arcabouços de Extração

Se analisarmos a dimensão Contexto, também considerando as questões de pesquisa da RS que norteou a execução da síntese temática, percebemos a adaptação que se apresentou necessária, considerando nos dados do Contexto, as especificidades da área que estava sendo analisada. A intenção era nos aproximar da realidade de *code smells* e poder responder às questões de pesquisa da RS com as análises oriundas desta extração.

Na Tabela 11 apresentamos como ficou um pouco diferente nossa dimensão contexto em relação à dimensão proposta por Cruzes e Dyba (2011a).

Tabela 11 Análise das dimensões Contexto

Dimensão Contexto – Cruzes e Dyba	**Dimensão Contexto – Método Refinado**
Participantes	Unidade experimental
Tecnologia	Não se aplica para *code smells*
Indústria	Não se aplica para *code smells*
Configurações	Tipo da Aplicação Tamanho da Aplicação
Instrumentos	*Smell* Heurística de avaliação
Tipos de Estudo	Método empírico

Desta forma, a extração dos dados foi realizada seguindo uma adaptação das definições do trabalho de Cruzes e Dyba (2011a), onde os seguintes tipos de dados foram extraídos para as respectivas dimensões apresentadas:

- Publicação: Autor(es), ano, fonte;
- Objetivos: ID, título, objetivo, questões de pesquisa/hipóteses, observações;
- Contexto: Método empírico, tipo de aplicação, tamanho da aplicação, unidade experimental (que envolvia os participantes), *smells*, heurística de avaliação;
- Resultados: segmentos de texto/dados (texto extraído dos resultados apresentados ao longo de todo o artigo), origem, força da evidência.

3- As principais fontes das informações devem ser buscadas nas seções: que descrevem os resultados, que apresentam a análise dos resultados, discussão e conclusões:

Tratamos como principais fontes de informações as seguintes dimensões: objetivo, contexto e resultados.

A Tabela 12 apresenta os tipos de dados extraídos dos estudos primários para mapear a dimensão "objetivo" e as principais seções dos artigos em que há maior incidência de cada dado extraído.

Tabela 12 Dados extraídos e seções onde foram encontrados

Dados	Seções
ID	Representa apenas um id para o artigo, não tendo sido extraído dos artigos e sim fornecido pela ferramenta StArt
Título	Título
Objetivo	Resumo, Introdução, Conclusão
Questões de Pesquisa	Introdução, Questões de Pesquisa, Planejamento Experimental, Discussão
Hipóteses	Definição do Estudo, *Design*

Para a extração dos dados do contexto, as principais seções são: as que trazem a explicação do *design* experimental, que apresentam a análise dos resultados e as discussões.

Já para os resultados, propomos considerar: Resumo, Introdução, Resultados, Discussão e Conclusão. Esta forma de extração aperfeiçoa e sistematiza a etapa de extração ajudando, inclusive, na avaliação da qualidade dos estudos primários selecionados.

Este resultado, de uma maneira geral, reforça o que foi mencionado pelos autores no Capítulo 3. A comprovação das seções dos artigos em que havia maior incidência de cada dado extraído ocorreu já o início do método, ou seja, na primeira etapa do ciclo de

extração/codificação. Esta comprovação foi um aprendizado que norteou as buscas nas etapas subsequentes, tornando a execução do método mais ágil.

4- Figuras e tabelas devem ser consideradas na extração:

As figuras e tabelas existentes nos estudos primários selecionados para a síntese temática são úteis para o entendimento dos artigos, principalmente no que tange ao seu contexto e seus resultados. Extrair figuras e tabelas quando se está extraindo de forma manual, complica o método. Para os artigos trabalhados nesta dissertação, não identificamos informações contidas em figuras e tabelas que não houvessem se apresentado também de forma textual. Consideramos que todo o texto extraído para suprir as necessidades das dimensões trabalhadas no arcabouço da extração foi suficiente para a condução do método e realização da síntese. Não notamos qualquer tipo de dificuldade na condução do método ou mesmo qualquer perda por não termos considerado figuras e tabelas como fontes de dados a serem extraídas. Isto não quer dizer que, para outros trabalhos, as mesmas não devam ser consideradas como passíveis de extração.

5- Perguntas sugeridas devem ser empregadas quando da identificação de uma declaração que pode ser considerada como uma constatação:

As perguntas sugeridas pelos autores, tais como: “Resumem os resultados das análises?” ou “Pode ser usado para responder às questões de pesquisa apresentadas?” devem ser amplamente utilizadas. Estas perguntas auxiliam durante os momentos de dúvidas e/ou divergências entre os pesquisadores envolvidos, principalmente em relação à extração dos resultados. Consideramos a pergunta a seguir como uma das mais relevantes: “Refletem os principais resultados do estudo?”. Muitas descobertas são sinalizadas pelos artigos, mas nem todas refletem os principais resultados do estudo, o que facilita a compreensão dos envolvidos acerca do que deve ser extraído nos estudos.

Uma consideração importante que realizamos é relacionada à diferenciação do que é apenas uma suposição dos autores e do que é efetivamente uma constatação, ou seja, um resultado do estudo. A decisão sobre como tratar as conjecturas depende dos objetivos do estudo e da avaliação dos recursos (especialmente tempo e força de trabalho) disponíveis. Em nosso caso, apesar das conjecturas terem sido extraídas e marcadas em cor diferente na Planilha de Extração/Mapeamento, após consenso, não foi feita codificação destas conjecturas, apenas das constatações efetivas dos artigos.

6- Extração do contexto é mais desafiadora:

Como mencionado pelos autores, constatamos que a extração do contexto é mesmo uma tarefa mais desafiadora e também atribuímos esta observação ao fato de não existirem padrões na apresentação dos artigos. Decorre disto a falta de determinadas informações consideradas relevantes. Por exemplo, nem sempre os artigos apresentam de forma clara os participantes nos estudos ou mesmo as heurísticas adotadas para avaliação dos *smells*. Para estes exemplos, tanto a extração das informações quanto a generalização dos achados fica prejudicada, mas não chega a inviabilizar a execução do trabalho.

7- Sempre que possível, a extração deve ocorrer de forma independente por dois ou vários pesquisadores, comparando os resultados:

No contexto que realizamos nosso trabalho, é comum não haver disponibilidade de diversos pesquisadores experientes executando com rigor todas as etapas de uma síntese temática. Considerando esta inexperiência, propomos que ao invés de ocorrer de forma independente, que a extração seja realizada em diferentes etapas, conforme as considerações a seguir:

- Marcações devem ser realizadas diretamente no corpo dos artigos em relação aos conjuntos de dados que precisam ser extraídos.
- Após esta etapa, um dos pesquisadores pode realizar a transcrição destas informações para a Planilha de Extração/Mapeamento. Nesta transcrição, o primeiro nível de verificação é realizado com a finalidade de tentar confirmar a relevância das informações marcadas que serão extraídas para a planilha.
- Após as extrações na planilha, o especialista era envolvido para um segundo nível de verificação.

Dessa forma, mesmo com um pouco mais de esforço inicial, tentávamos manter a assertividade das informações consideradas e a confiabilidade da síntese que estava sendo desenvolvida. As verificações, seguindo estas recomendações, não são negligenciadas. Além disso, reuniões de consenso devem ser realizadas quando são percebidas discordâncias. Esta estratégia auxiliou no amadurecimento dos envolvidos em diferentes níveis: pesquisadores marcando os artigos compreendem melhor os dados que devem ser considerados; pesquisadores envolvidos na montagem da Planilha de Extração compreendem melhor a identificação, seleção dos dados e imersão no tema da síntese em questão.

Uma forma de minimizar o esforço desta etapa é estabelecer uma hierarquia para a obtenção do consenso. Neste trabalho, propomos a estrutura hierárquica representada na Figura 10. Representamos entre parênteses a quantidade de pesquisadores assumindo cada um dos papéis para este trabalho. Sempre que existiam discordâncias entre os pesquisadores, um especialista no tema era envolvido. Caso o especialista no tema estivesse atuando em conjunto com os pesquisadores, ou o consenso não fosse alcançado, um pesquisador independente era envolvido.

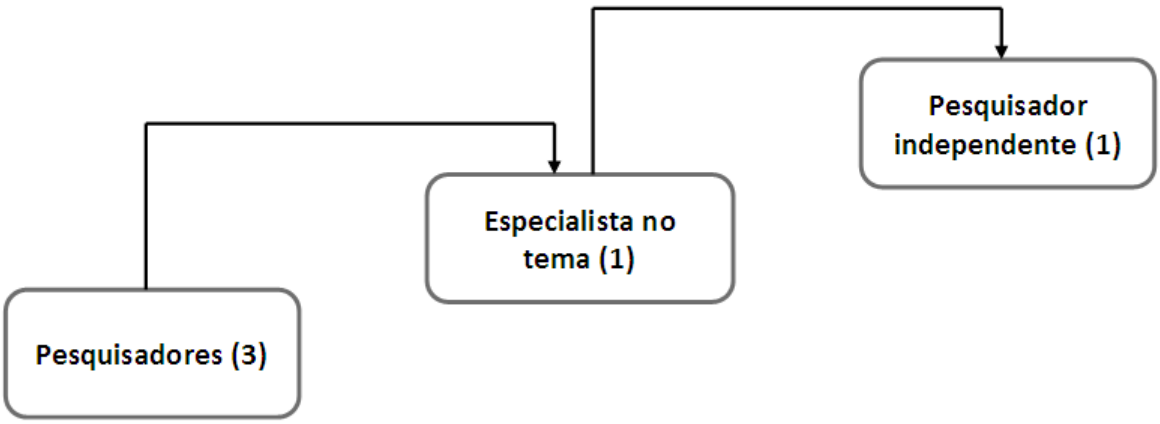


Figura 10 Hierarquia recomendada para o tratamento de discordâncias

4.2 CODIFICAÇÃO DOS DADOS

8- Todos os segmentos de texto que exemplifiquem a mesma ideia devem ser extraídos:

Apesar desta recomendação aparecer no passo de codificação para Cruzes e Dyba (2011a), a mesma foi utilizada na prática durante o passo de extração. O objetivo de um artigo tipicamente aparece no resumo, introdução e mesmo na conclusão. Da mesma forma, um resultado, por exemplo, também pode ser reportado no resumo, na introdução, nos resultados, na discussão e também na conclusão.

Diante desta dispersão dos dados, adotamos para este trabalho, a recomendação de extrair todos os segmentos de texto que exemplificavam a mesma ideia.

A Planilha de Extração/Mapeamento apresenta todos estes segmentos de texto que exemplificam a mesma ideia, considerando o conjunto de dados que precisa ser extraído.

Outra questão importante, dado o caráter manual do trabalho está relacionada ao fato da rastreabilidade entre os segmentos de texto extraídos nos artigos e sua transcrição para a Planilha de Extração/Mapeamento ter sido mantida. A rastreabilidade também deve ser garantida entre as etapas de codificação e definição do mapa temático. Desta forma, uma rastreabilidade bidirecional foi criada (conseguimos a partir dos dados extraídos chegar até o resultado e do resultado contido no Mapa Temático, voltar até a extração), favorecendo a avaliação de confiabilidade da síntese. A rastreabilidade considerada para esta dissertação será explicitada na Seção 5.2.3, questão 12 desta dissertação.

9- A categorização dos temas acontece em comparações constantes, considerando o conhecimento dos pesquisadores:

A presença do especialista no tema foi fundamental para a condução da síntese temática. No contexto em que este trabalho se insere, é muito comum que os pesquisadores envolvidos apresentem diferentes níveis de conhecimento sobre o tema. Isso ocorre mesmo com toda imersão propiciada nas fases iniciais do método. O envolvimento e conhecimento do especialista favoreceram as categorizações dos subtemas e temas e identificação das relações, além de ter consolidado a formação de pesquisadores menos experientes sobre o tema.

10- Para a codificação em RS é mais indicado trabalhar blocos de texto:

Utilizamos a extração de blocos de texto que continham os dados inerentes às dimensões (publicação, objetivo, contexto e resultados) para realizar a codificação. A codificação foi realizada para a dimensão objetivo e para a dimensão resultados. Porém todas as dimensões utilizadas na extração apresentaram-se importantes para que esta codificação fosse mais assertiva.

A recomendação de utilizar blocos de texto para a extração parece pertinente não apenas para o caso de codificação em RS, mas para os casos de codificação em diversos outros contextos de pesquisa. Trabalhos que se baseiam em evidências extraídas de entrevista são outro exemplo de que a codificação de blocos de texto facilita na identificação de temas relacionados. O método de codificação escolhido para nortear esta etapa foi com base neste fato de que era necessário trabalhar com blocos de texto.

11- A adoção de uma abordagem para condução da codificação é importante: dedutiva, indutiva ou integrada:

A abordagem indutiva foi adotada na condução da codificação. Este fato se deu porque não se mostrava estratégico, frente às questões de pesquisa da RS, qualquer forma que restringisse as percepções dos pesquisadores envolvidos. Desta forma, não foi criada lista de códigos iniciais com definições e frequências. Não foi identificado de imediato nenhum tipo de prejuízo ao método pela inexistência da referida lista. A abordagem também se mostrou adequada com relação à consolidação dos conhecimentos por parte de pesquisadores menos experientes.

Com a maturidade adquirida ao longo do método, percebemos que a abordagem integrada era uma opção interessante de ser adotada para este trabalho.

12- Os segmentos importantes do texto como conceitos, categorias, descobertas e resultados devem ser rotulados e codificados e 13- A codificação deve ser feita em um nível apropriado para as questões de pesquisa:

Na preparação para a extração já havíamos definido os conjuntos de dados que seriam extraídos e trabalhados durante a codificação. Toda decisão foi embasada pelas questões de pesquisa documentadas para a RS. Foram trabalhadas as seguintes dimensões: publicação, objetivo, contexto e resultados. Conceitos e categorias não se mostraram interessantes para atender às questões de pesquisa definidas, mas foram imprescindíveis para o entendimento geral dos artigos e da área que estava sendo explorada. Desta forma não foram extraídos e codificados, mas entendidos e analisados junto ao contexto em que se apresentavam, favorecendo o aumento do nível de entendimento dos pesquisadores envolvidos.

Vale ressaltar que, pelo método adotado, códigos sobrepostos (relacionado à questão 16) já foram reduzidos neste passo da codificação e não no passo seguinte de tradução dos códigos em temas, como sugerem Cruzes e Dyba (2011a).

14- Verificações de consistência e de confiabilidade entre pesquisadores devem ser realizados para estabelecer a credibilidade da codificação:

A preocupação com as verificações de consistência e com a confiabilidade foram garantidas pelo fato de terem sido realizadas revisões da codificação por outro pesquisador que não o que codificou. O revisor, neste caso, foi o especialista no tema. A verificação deve

checar, dentre outras coisas, se existem ligações claras e evidentes entre os segmentos de texto extraído dos artigos e os códigos que foram atribuídos. Além disso, a relevância dos achados para o tema central da RS também deve ser verificada. Para os casos de discordância, reuniões são necessárias para discussão e devidos esclarecimentos. Se após a reunião, a dúvida persiste, um terceiro pesquisador independente deve ser consultado. Esta estratégia baseia-se na estrutura hierárquica de trabalho pré-definida, que permitiu otimização dos esforços dentro do contexto da pesquisa.

Como já mencionado, a rastreabilidade foi uma preocupação presente durante toda a realização do método. Esta não é uma preocupação apresentada de forma clara no método proposto por Cruzes e Dyba (2011a).

4.3 TRADUÇÃO DE CÓDIGOS EM TEMAS

15- Os temas devem ser criados a partir de uma visão geral, inclusiva e compreensiva dos códigos de todos os artigos:

Essa recomendação é parte natural do método incremental e iterativo que propomos para os passos de Extração e Codificação. Apenas após a codificação da última etapa de trabalho, deve-se iniciar a definição dos temas. Sendo assim, é possível adquirir uma visão geral, inclusiva e compreensiva mencionada pelo método de síntese temática. Esta visão adquirida agrega maior nível de entendimento dos envolvidos e amadurecimento no tema, favorecendo as análises e discussões para a criação da síntese temática.

16- Códigos sobrepostos devem ser reduzidos e os códigos restantes traduzidos em temas:

Com a recomendação de que todas as ideias sejam extraídas, é esperado que existam segmentos de texto redundantes. Durante o passo de codificação, quando os segmentos de texto são analisados e rotulados, já existe uma preocupação em codificar reduzindo-os. Com essa estratégia, minimizamos a existência de códigos sobrepostos a serem reduzidos nos próximos passos executados. Neste passo do método, apenas revisamos os códigos buscando identificar a necessidade de tratar algum código que ainda pudesse ser

reduzido. Neste ponto, dada a maturidade adquirida com a execução do método, a avaliação dos códigos sobrepostos foi bastante simplificada.

17- Os temas devem ser verificados uns contra os outros e deve ser feito batimento dos mesmos com os dados dos artigos originais:

Como as extrações são inicialmente transcrições de segmentos de texto dos artigos e estas devem ser realizadas e revisadas por pesquisadores independentes, este batimento mostrou-se necessário apenas durante a definição do Mapa Temático.

A verificação dos temas uns contra os outros e os batimentos dos mesmos com as extrações feitas na Planilha de Extração/Mapeamento atendem essa recomendação dentro do nosso contexto, pois consolidam a formação dos pesquisadores envolvidos e minimizam o esforço de verificação. Entendemos que esta atividade é suficiente para garantir uma consistência e coerência entre as descobertas dos artigos e os temas identificados.

18- Utilização de representações visuais para auxiliar entendimento e classificação dos diferentes códigos em subtemas e temas:

Na prática, este passo deve ser realizado junto à criação do mapa temático. Acreditamos que esta junção torna o método mais rápido e simples. Para um melhor aproveitamento do método, considerando a execução de todos os passos de maneira manual, um identificador para cada código gerado deve ser criado, mantendo a rastreabilidade do mesmo no artigo com o mapa temático criado. Explicitaremos as demais considerações nas seções 4.4 e 4.5, mas a tradução dos códigos em temas ocorre durante a criação do mapa temático, como já foi mencionado.

4.4 CRIAÇÃO DO MODELO DE ALTO NÍVEL

19- Descobertas interpretadas devem fazer mais sentido do que se fosse somente parafraseada ou descrita:

Durante a realização do método, todo o cuidado deve ser mantido para evitar que a interpretação das descobertas altere o sentido dado pelo autor. Como trabalhamos com a transcrição dos segmentos de textos dos artigos para a Planilha de Extração/Mapeamento,

evitamos que este problema ocorresse nos passos iniciais da síntese temática. Contextualizando esta questão para o passo de criação do modelo de alto nível, podemos mencionar nossas interpretações para a escrita da síntese temática, citando o exemplo relacionado à concordância na detecção de *smells* (*agreement on detection*). A descrição completa da síntese, contendo mais exemplos relacionados a esta questão de interpretação, encontra-se no Apêndice A desta dissertação.

A Figura 11 apresenta a análise feita a respeito deste subtema, onde podemos interpretar que o nível de concordância na detecção de *smells* dentre os artigos analisados é baixo.

> ***Agreement on detection* (Concordância na detecção).** Seis dos vinte e seis artigos mencionam a respeito de acordo na detecção de *code smells* dentre os participantes. A grande maioria destes artigos está contida no conjunto de estudos que abordam fatores que afetam a detecção de *code smells*, ou seja, no subtema *Factors affecting detection* (apenas o artigo [2] trabalhado neste subtema não aparece no subtema *Factors affecting detection*). Devido a este fato, as características dos estudos são muito semelhantes: i) quatro realizaram experiências controladas, e dois realizaram *surveys*; ii) os estudos utilizaram sistemas de pequeno, médio e grande porte e; iii) usaram sistemas open source, comerciais e construídos; iv) quatro estudos tiveram como participantes profissionais, enquanto os outros dois estudos usaram alunos como participantes; e v) todos os estudos adotaram inspeção código como a tarefa a ser realizada durante os seus respectivos experimentos. Com exceção do artigo [84], que foi um *survey* questionando os participantes sobre a extensa variedade de *smells*, todos os demais estudos investigaram o nível de concordância/acordo sobre um pequeno conjunto de *smells* (três, no máximo, para cada artigo). Somente o artigo [86] descobriu que há concordância parcial entre os participantes, sendo que todos os demais encontraram baixa concordância. Os artigos [48][2] explicitam realmente como baixo nível de concordância na detecção de *smells*, enquanto que o artigo [83] chega a conclusão igual, mas apresentou a ideia de uma forma diferente. Para este estudo, os autores afirmaram que os participantes tiveram diferentes percepções e julgamentos durante a identificação de *code smells*. O mesmo aconteceu com o artigo [84], onde seus autores descobriram que os desenvolvedores não têm uma única opinião sobre o "*smelliness*" do código fonte.

Figura 11 Análise sobre a concordância na detecção de *smells* extraída do Apêndice A

20- Temas e relações entre eles devem ser comparados e verificados com as conclusões dos estudos primários:

As comparações constantes fazem parte do método de síntese temática e são importantes para que possamos refinar os temas e entender as relações entre eles. Os temas e relações devem ser comparados durante todo o passo de criação do modelo de alto nível, reorganizando sempre que necessário, para criar uma visão gráfica coerente e concisa das conclusões apresentadas pelos estudos primários. Na Planilha de Extração/Mapeamento já há uma preocupação em extrair as descobertas e conclusões apresentadas pelos artigos, contribuindo mais uma vez para que estas comparações constantes pudessem ser realizadas com o auxílio do referido instrumento. Dessa forma, esta recomendação pode ser considerada como tendo sido incorporada às etapas anteriores, o que mais uma vez minimiza o esforço de voltar aos estudos primários selecionados para realizar análises e comparações.

21- Deve existir uma descrição clara das relações entre os temas mais abrangentes:

Para a descoberta das relações entre os temas e entre os temas e seus respectivos subtemas, considerando o nosso contexto, foi imprescindível a participação do especialista no tema. A sua presença forneceu maior segurança para que as relações percebidas pelos pesquisadores envolvidos fossem efetivamente validadas. Reuniões para discussão sobre a definição destas relações também foram realizadas, possibilitando a consolidação do conhecimento de pesquisadores menos experientes.

22- As saídas podem ser uma visão mais gráfica dos temas e suas relações, uma visão mais descritiva, uma taxonomia ou mesmo uma teoria:

Esta recomendação depende fortemente da natureza da pesquisa a ser realizada. Recomendamos, entretanto, não separar a criação do mapa temático da atividade anterior de tradução dos códigos em tema. Ambas devem ser realizadas em conjunto visando otimização do esforço, obtida com nossa experiência prática.

Para a nossa experiência prática, geramos o Mapa Temático (visão gráfica) e uma visão descritiva, ambos devidamente representados pelo Apêndice A desta dissertação.

23- O contexto em que as descobertas foram encontradas deve ser considerado:

Recomendamos o uso de um arcabouço para a definição do contexto e os objetivos dos estudos primários. Pensar no que é importante para o estudo em questão representa uma boa prática e pode implicar na qualidade dos resultados alcançados. Para a

escrita da visão descritiva da síntese consideramos o contexto em que cada resultado foi obtido, detectando quando era possível generalizar um resultado ou não.

4.5 AVALIAÇÃO DA CONFIABILIDADE DA SÍNTESE

24- Considerar os conceitos de credibilidade, confirmabilidade, confiança, transmissibilidade em assuntos relacionados à confiabilidade da síntese:

Os conceitos apresentados são considerados na avaliação da confiabilidade da síntese. Na Seção 4.5.1 apresentamos a nossa proposta para um *Checklist* de Avaliação de Confiabilidade da Síntese que adota estes principais conceitos.

25- Explanar os argumentos que conduzem as interpretações feitas em uma síntese temática:

Para a síntese temática, todas as interpretações feitas devem ser expressas por colocações de um ou mais autores, respaldando-as. No Apêndice A, na análise da visão descritiva, houve uma preocupação em explicar as descobertas considerando as colocações expostas pelos autores em seus respectivos trabalhos. Este fato permite que não seja transferido para a síntese um julgamento pessoal ou subjetivo do que está sendo trabalhado.

26- A linguagem e os conceitos utilizados na síntese devem ser consistentes:

A presença do especialista no tema foi fundamental para que esta recomendação fosse atendida. As verificações constantes feitas pelo especialista são um passo importante para a garantia desta consistência, especialmente pela presença de pesquisadores menos experientes nas fases iniciais da síntese, considerando o nosso contexto.

27- As questões de pesquisa devem ser respondidas com base na evidência da síntese temática:

Apesar desta recomendação aparecer de forma clara no trabalho de Cruzes e Dyba (2011a), entendemos que isso é um processo natural de qualquer estudo empírico. Como tal, não identificamos a necessidade de maiores recomendações sobre o assunto. Ressaltando apenas que esta foi uma preocupação constante do trabalho realizado.

4.5.1 Checklist de Avaliação de Confiabilidade da Síntese

Cruzes e Dyba (2011a) apresentam conceitos para a realização da avaliação da confiabilidade da síntese. Com base nestes conceitos, propomos um conjunto de perguntas complementares que podem ser utilizadas como *checklist* para simplificar a execução da avaliação da confiabilidade da síntese. Esta contribuição transcende o refinamento que propomos nesta dissertação, uma vez que é aplicável a qualquer contexto em que uma síntese seja realizada.

As questões relacionadas ao conceito de credibilidade foram consideradas por Cruzes e Dyba. Os demais conceitos foram descritas pelos autores, mas não houve proposição de questões inerentes aos mesmos. Aproveitamos estas descrições conceituais dos autores para propor os demais itens dos *checklist*, vinculados aos conceitos de: confirmabilidade, confiança e transmissibilidade.

A Tabela 13 apresenta estes principais conceitos e questões utilizados como base para a avaliação da confiabilidade da síntese realizada.

Tabela 13 Checklist de Avaliação de Confiabilidade adaptado de (cruzes e Dyba, 2011a)

Critério	Pergunta
Credibilidade	1. Os argumentos que conduzem as interpretações feitas na síntese temática foram explicitados? 2. As suposições sobre uma abordagem específica para a síntese temática foram claramente explicadas? 3. Existe um bom ajuste entre o que é reivindicado e o que a evidência mostra? 4. As questões de pesquisa foram respondidas com base na evidência da síntese temática? 5. A linguagem e os conceitos utilizados na síntese são consistentes?
Confirmabilidade	6. Pesquisadores e especialistas concordam com a maneira que estes dados foram codificados e classificados?
Confiança	7. Foi estabelecida e mantida a rastreabilidade das informações do artigo até a síntese?
Transmissibilidade	8. Os resultados obtidos podem ser transferidos para outros contextos?

4.6 REFINAMENTO DO MÉTODO

Nas questões tratadas nas seções 4.1 a 4.5, apresentamos as considerações práticas identificadas durante a execução do método de síntese temática proposto por Cruzes e Dyba (2011a). Percebemos neste contexto novas recomendações, frente ao contexto adotado para esta dissertação. Ou seja, um refinamento do método mostra-se interessante se o contexto a trabalhá-lo envolve disponibilidade de poucos pesquisadores e se estes se encontram em níveis diferentes de compreensão do tema central adotado. Para tanto, propomos a Tabela 14, contendo o refinamento do método apresentado de forma sistematizada, já considerando o *Checklist* de Avaliação de Confiabilidade da Síntese apresentado na Seção 4.5.1.

O principal objetivo da apresentação do refinamento do método de síntese temática proposto na Tabela 14 é auxiliar a execução prática por grupos que apresentem similaridade ao contexto aqui apresentado.

Separamos, em negrito na Tabela 14, as contribuições desta dissertação para o método refinado.

Tabela 14 Refinamento do Método de Síntese Temática

O que fazer?	Como fazer?
Extração de Dados	**1. Definir a adoção ou não de uma ferramenta de extração;** **1.1. Caso a decisão seja de realização da extração e codificação de forma manual é necessário garantir a rastreabilidade das decisões tomadas, extrações realizadas, codificações e identificação de temas.** **1.2. Para extrações manuais, instrumentos de extração devem ser planejados e gerados para tornar o processo mais consistente, diminuindo a possibilidade de retrabalhos.** **2. Realizar encontros iniciais para nivelar o conhecimento, equalizando as decisões tomadas para a realização do trabalho (tema central, critérios de inclusão, exclusão, alinhamento das expectativas em relação às questões de pesquisa);** **3. Avaliar a necessidade de um pesquisador especialista no tema central adotado para a síntese temática. Caso os pesquisadores não detenham domínio do tema central para o qual a RS foi realizada, recomenda-se a participação de um especialista, principalmente para promover as verificações dos passos apresentados, aumentando a confiabilidade da síntese;** 4. Ler todos os artigos selecionados, buscando imersão no contexto; 5. Analisar e definir o arcabouço de extração; 5.1. O arcabouço de extração sugere as possíveis dimensões que poderão ser codificadas no item seguinte. 5.2. É recomendado que o arcabouço de extração considere, no mínimo, as seguintes dimensões: publicação, contexto e resultados. **6. Analisar e dividir os artigos selecionados para extração, considerando a execução de etapas iterativas de extração e codificação. Recomenda-se a adoção de critérios para esta divisão em etapas;** 7. Extrair os dados dos estudos selecionados, observando na primeira etapa as principais seções onde as fontes de informação são comumente encontradas; **7.1. Detalhes das dimensões consideradas no arcabouço de extrações devem ser extraídos dos artigos selecionados;** **7.2. Diferenciar durante a extração o que é apenas suposição dos autores do que efetivamente é contribuição dos artigos;** **7.3. As principais seções onde as fontes de informações são comumente encontradas tendem auxiliar nas próximas etapas de extração, visto que norteiam na maioria dos casos, onde encontrar as informações que estão sendo consideradas na extração. Além disso, reduzem o esforço de leitura, sem prescindir da**

	sistematização do processo. 7.4. As seguintes perguntas podem auxiliar na identificação de uma constatação: • Indica resultados das medições? • Resumem os dados brutos? • Destacam alguma característica específica dos dados brutos? • Fornecem insights sobre tabelas ou figuras apresentadas? • Resumem os resultados das análises? • Pode ser usado para responder às questões de pesquisa apresentadas? • Refletem os principais resultados do estudo? 7.5. Todos os segmentos de texto que exemplifiquem a mesma ideia devem ser extraídos. 7.6. Sempre que possível, a extração deve ocorrer de forma independente por dois ou vários pesquisadores, comparando os resultados; **7.7. Manter a rastreabilidade entre as informações contidas nos artigos e os segmentos de texto extraídos, caso o processo seja sem o apoio de ferramentas de extração.** 8. Verificar a extração. **8.1. A Verificação deve ser feita por um pesquisador diferente daquele que realizou a extração dos dados.** **8.2. Na verificação é importante garantir que os itens relevantes para as dimensões consideradas foram extraídos.** **8.3. É importante combinar com os demais envolvidos na síntese/verificação, os passos que serão adotados na execução das verificações, alinhando os critérios de verificação adotados.**
Codificação de Dados	**9. Definir as dimensões que serão codificadas;** **9.1. Como dito no passo anterior, as dimensões que serão codificadas devem estar contidas no arcabouço de extração.** 10. Definir os métodos de codificação que serão adotados, considerando a natureza da dimensão e sua representatividade nas análises; 11. Definir a abordagem para realização da codificação (dedutiva, indutiva, integrada); 11.1. A abordagem integrada é uma opção interessante de ser adotada, pois concilia a preocupação de não ser restritiva com o aproveitamento do conhecimento extenso de algum especialista que esteja participando do método. No entanto, deve-se considerar a experiência dos pesquisadores envolvidos e o estágio da pesquisa. **12. Manter a rastreabilidade na codificação.**

	13. Codificar os conjuntos de dados, realizando comparações constantes, considerando o conhecimento dos pesquisadores e as questões de pesquisa. Esta codificação deve considerar a rastreabilidade definida para a síntese temática. Caso sejam identificados códigos sobrepostos, os mesmos já podem começar a ser tratados; 13.1. Para a codificação em RS é mais indicado trabalhar blocos de texto; 13.2. A codificação requer cuidados, principalmente para os casos em que se trabalha com blocos de textos e /ou entrevistas. Esta ressalva torna-se importante para que não seja empregada subjetividade dos pesquisadores nas descobertas e informações passadas pelos autores dos artigos selecionados; 13.3. Os segmentos importantes do texto como conceitos, categorias e demais informações relacionadas com as dimensões consideradas devem ser rotulados e codificados; 13.4. A codificação deve ser feita em um nível apropriado para as questões de pesquisa consideradas para a síntese temática. 14. Verificar a codificação. 14.1. A Verificação deve ser feita por outro pesquisador que não o que fez a codificação dos dados. **14.2. Na verificação é importante garantir a consistência, aumentando a confiabilidade e permitindo aumento da credibilidade.** 14.3. Devem existir ligações claras entre o texto e os códigos identificados. **14.4. É importante combinar com os demais envolvidos na síntese/verificação, os passos que serão adotados na execução das verificações, alinhando os demais critérios de verificação que venham a ser adotados.**
Tradução de Códigos em Temas	15. Analisar se foi adquirida uma visão geral, inclusiva e compreensiva de todos os artigos; **15.1. Com a divisão dos artigos em etapas iterativas de extração e codificação, consegue-se melhorar a visão geral, inclusiva e compreensiva de todos os artigos.** 15.2. Esta visão geral, inclusiva e compreensiva é importante para o sucesso deste passo, evitando retrabalhos. 16. Traduzir os códigos em temas, considerando que: 16.1. Códigos sobrepostos, que não tenham sido tratados até este passo, devem ser reduzidos; 16.2. É importante separar os códigos que apresentem relações entre si utilizando um instrumento que melhore a visibilidade para esta análise; 16.3. Análise dos códigos agrupados com a finalidade de tentar identificar um termo que possa abrangê-los (temas); 16.4. Uma vez identificados, os temas devem ser verificados uns contra os outros; 16.5. Os temas devem ser verificados com os dados dos artigos originais; **16.6. Caso as extrações tenham sido transcrições de segmentos de texto dos artigos, a verificação mencionada para os temas pode ser feita em relação a estas extrações, não sendo imprescindível retornar aos artigos**

	originais para esta atividade. **16.7. A rastreabilidade deve ser mantida, entre os códigos gerados, temas e subtemas encontrados.** 17. Verificar a tradução dos códigos em temas. 17.1. A Verificação deve ser feita por outro pesquisador que não o que fez a tradução dos códigos em temas. 17.2. Na verificação deve-se tentar garantir que os temas são coerentes, consistentes e distintos. **17.3. É importante combinar com os demais envolvidos na síntese/verificação, os passos que serão adotados na execução das verificações, alinhando os demais critérios de verificação que venham a ser adotados.**
Criação do Modelo de Alto Nível	18. Analisar, explorar e comparar as relações existentes entre os temas; 19. Comparar as relações existentes entre os temas com as conclusões dos artigos; 20. Criar um modelo para representar os temas e as relações existentes entre os temas; 20.1. As saídas podem ser uma visão mais gráfica dos temas e suas relações (mapa temático), uma visão mais descritiva, uma taxonomia ou mesmo uma teoria. **20.2. A rastreabilidade deve ser mantida de forma que pelo item do modelo de alto nível criado seja possível chegar ao código e, respectivamente, ao segmento de texto extraído do qual o mesmo foi originado.** 21. Descrever de forma clara as relações existentes entre os temas; 21.1. O contexto em que as descobertas foram encontradas deve ser considerado. Portanto, as dimensões que serão codificadas auxiliam as comparações e consequentemente, auxiliam a criação dos modelos que representam os temas e suas relações. 21.2. Revise o modelo de alto nível da última etapa, pegando cada ramo por vez, descrevendo o seu conteúdo com os resultados e o contexto das informações e criando temas de ordem superior. 21.3. Identifique as ligações entre os temas de mais alto nível e as provas que fundamentam e contexto desta evidência. 21.4. Explore as conexões com a teoria e pesquisas prévias, contextualize novamente, defina e ainda refine os temas de mais alto nível. 22. Verificar a criação do modelo de alto nível. 22.1. Esta deve ser feita por outro pesquisador ou por especialista no assunto. 22.2. Na verificação deve-se tentar garantir que o modelo de alto nível gerado reflete as relações encontradas durante a realização do método. **22.3. É importante combinar com os demais envolvidos na síntese/verificação, os passos que serão adotados na execução das verificações, alinhando os demais critérios de verificação que venham a ser adotados.**

<table>
<tr><td rowspan="4">Avaliação da Confiabilidade da Síntese</td><td colspan="2">23. Realizar a avaliação da confiabilidade da síntese.
23.1. Para a realização da confiabilidade da síntese, preocupar-se com os conceitos e questões que compõem o checklist de avaliação da confiabilidade da síntese, sendo eles:</td></tr>
<tr><td>Credibilidade</td><td>1. Os argumentos que conduzem às interpretações feitas na síntese temática foram explicitados?
2. As suposições sobre uma abordagem específica para a síntese temática foram claramente explicadas?
3. Existe um bom ajuste entre o que é reivindicado e o que a evidência mostra?
4. As questões de pesquisa foram respondidas com base na evidência da síntese temática?
5. A linguagem e os conceitos utilizados na síntese são consistentes?</td></tr>
<tr><td>Confirmabilidade</td><td>6. Pesquisadores e especialistas concordam com a maneira que estes dados foram codificados e classificados?</td></tr>
<tr><td>Confiança</td><td>7. Foi estabelecida e mantida a rastreabilidade das informações do artigo até a síntese?</td></tr>
<tr><td></td><td>Transmissibilidade</td><td>8. Os resultados obtidos podem ser transferidos para outros contextos?</td></tr>
</table>

4.7 CONCLUSÃO DO CAPÍTULO

Neste Capítulo foram apresentadas as respostas às recomendações apresentadas por Cruzes e Dyba (2011a), em forma de *Checklist*, no Capítulo 3 e o método refinado de síntese temática, gerado como contribuição deste trabalho. A numeração sequencial das respostas apresentadas neste Capítulo, portanto, refere-se à numeração das questões apresentadas no *Checklist* que compõe o Capítulo 3 desta dissertação.

No Capítulo 5 será apresentada a aplicação do método refinado proposto neste Capítulo, considerando o tema central *Code Smells*. Sendo assim, para o Capítulo 5, a numeração sequencial disposta referencia as questões apresentadas na Tabela 14 que contém o método refinado, proposta desta dissertação, provando que o mesmo foi seguido.

Capítulo 5

Este Capítulo apresenta a aplicação do método de síntese temática refinado, proposto no Capítulo 4 desta dissertação. Resultados obtidos com esta prática subsidiaram a proposição do método refinado, apresentado no Capítulo anterior e vice-versa.

5 APLICAÇÃO DO MÉTODO REFINADO

À medida que o método refinado proposto nesta dissertação era executado na prática, observações sobre a sua aplicabilidade eram identificadas. Estas observações serviram de insumo para a definição das recomendações apresentadas no refinamento do método. Enquanto as diretrizes surgiam, retroalimentavam a execução do método, formando um processo incremental.

O foco principal deste Capítulo será a aplicabilidade prática do método refinado, com a explicação de alguns exemplos que elucidem a sua aplicação. Os resultados da síntese temática propriamente dita, bem como os detalhes da RS realizada podem ser obtidos no Apêndice A e Anexo 1, respectivamente, desta dissertação.

5.1 ESTRUTURAÇÃO PARA REALIZAÇÃO DA SÍNTESE TEMÁTICA

O método adotado para a realização da síntese temática ocorreu conforme ilustração da Figura 12. A Figura apresenta uma adaptação dos cinco passos considerados por Cruzes e Dyba (2011a). Nos passos de Extração dos Dados e Codificação, não identificamos necessidade de comentários, mas no passo de Tradução do Código em Temas, achamos que ficaria mais compreensível dividir nas subetapas de Identificação de Subtemas e Identificação de Temas para deixar o método mais sistemático, representando fielmente os passos

realizados na nossa prática. Após este passo vem a Criação do Modelo de Alto Nível e, por fim, a escrita da síntese temática propriamente dita. Optamos na nossa prática pela criação do Mapa Temático e pela visão descritiva da síntese para o tema central adotado. Ambos podem ser encontrados no Apêndice A desta dissertação.

O passo de Avaliação da Confiabilidade da Síntese ocorre durante toda a execução do método, através das diversas verificações que são realizadas ao final de cada passo. Ainda neste passo ocorreu a aplicação do *Checklist* de Avaliação da Confiabilidade, após a escrita da síntese temática. Um maior detalhamento de cada passo será apresentado a partir da Seção 5.2 desta dissertação.

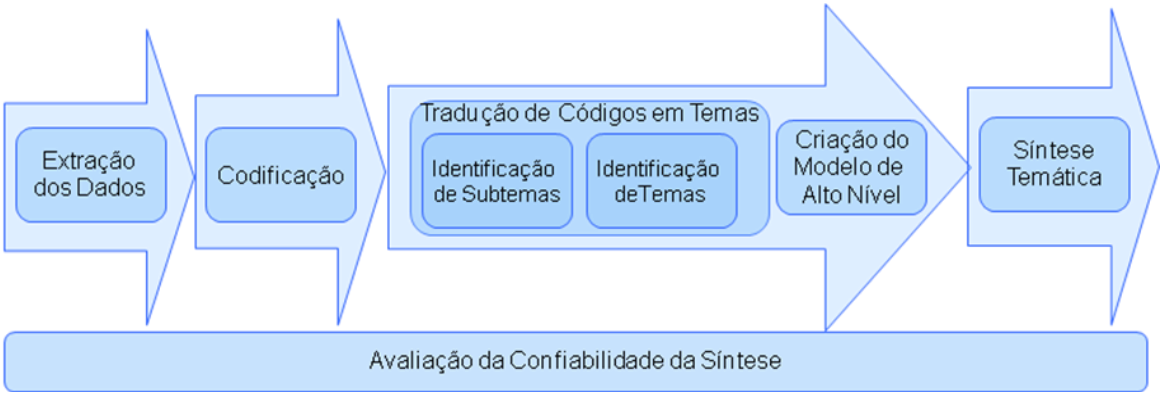


Figura 12 Método Geral da Síntese Temática

5.1.1 Participantes

Além da participação da autora desta dissertação e do especialista no tema, mais dois pesquisados da UFBA, também alunos de mestrado, contribuíram para o cumprimento das metas, totalizando quatro colaboradores envolvidos.

A participação dos dois pesquisadores da UFBA ocorreu mais fortemente na etapa de filtragem. Estes pesquisadores também auxiliaram na extração de dados, realizando as marcações em uma parte dos artigos considerados para a realização da síntese. Estas marcações ocorreram basicamente para a etapa 03 e foram revisadas pela autora desta dissertação. A autora desta dissertação ficou responsável pela organização de todos os dados

marcados nos artigos selecionados para a Planilha de Extração/Mapeamento. Quando não havia consenso nesta etapa, o especialista no tema era acionado.

Foi adotada a redundância de verificação neste contexto, por acreditar que isso poderia contribuir para o aumento da confiabilidade do método. Sendo assim, tudo era produzido e ao final de cada etapa era verificado pelo especialista no tema, mesmo não havendo discordâncias.

Já para as etapas em que os pesquisadores da UFBA não participaram (da transcrição dos dados extraídos para a Planilha em diante), a autora desta dissertação contou com a verificação do especialista no tema e um terceiro pesquisador independente foi envolvido para os casos de discordâncias.

5.2 ETAPAS APLICADAS NO MÉTODO DE SÍNTESE TEMÁTICA REFINADO

Nesta Seção, apresentamos a aplicação de cada uma das recomendações propostas a partir do método de síntese temática refinado que é proposto nesta dissertação.

Para iniciar o entendimento, é importante aclarar que uma das recomendações dos autores Cruzes e Dyba (2011a) para o passo de Extração dos Dados foi realizada, na prática, na fase inicial de filtragem da RS. A atividade de leitura aprofundada dos artigos ocorreu na etapa de filtragem, anterior ao início dos passos da síntese temática. Devido a este fato, iniciaremos a apresentação da nossa aplicação prática do método a partir desta etapa específica da RS, prevista no refinamento realizado. Mesmo descrevendo desta maneira, apenas com a finalidade de tentar melhorar a didática da apresentação da nossa prática, não iremos alterar os passos propostos para a execução do método. Concordamos que esta atividade de leitura, sendo executada antes ou depois, deve estar finalizada até o início do passo de Extração dos Dados.

Após a apresentação da etapa de Filtragem, discutimos os passos de Extração, Codificação, Tradução dos Códigos em Temas e, por fim, a Avaliação da Confiabilidade da Síntese, conforme método refinado adotado.

O objetivo é sintetizar os resultados obtidos pelos estudos empíricos que investigam o impacto na adoção do conceito de *code smell* no desenvolvimento de software. *Code smells* são apresentados como uma forma de identificar problemas de *design* de código

e estes, potencialmente, podem gerar problemas de evolução, principalmente durante a manutenção (Fowler, 1999; Lanza, Marinescu, 2005). Os estudos empíricos que avaliam o efeito do *smell* têm apresentado resultados inconsistentes (Marinescu e Marinescu, 2011; Pope, Mays, 1995; Yamashita e Moonen, 2013b; Yamashita e Moonen, 2013c). A síntese temática realizada no âmbito desta dissertação visa entender estas diferenças entre os resultados encontrados pelos artigos. Dado este fato, foi necessário sintetizar os resultados dos experimentos que envolviam pessoas avaliando *smells* ou experimentos correlacionando *smells* com atributos ligados à evolução de *software*, como número de erros e esforço gasto com manutenção. Desta forma, tornou-se possível um maior aprofundamento a respeito do tema em questão.

5.2.1 Filtragem

Durante a fase de filtragem utilizamos a ferramenta StArt (Fabbri, 2012; Zamboni *et al.*, 2010). Esta ferramenta serviu como apoio para a realização da RS e foi útil, principalmente na organização dos registros, conduzindo a um racional dos artigos considerados e dos excluídos.

Durante toda a filtragem, cada estudo foi avaliado por dois pesquisadores, de forma independente. Em caso de desacordo, o artigo era mantido até a próxima fase, onde a leitura mais aprofundada eliminaria a dúvida.

Na fase inicial existiam 2.130 artigos. Um resumo do método de filtragem acima explicitado pode ser verificado na Figura 13.

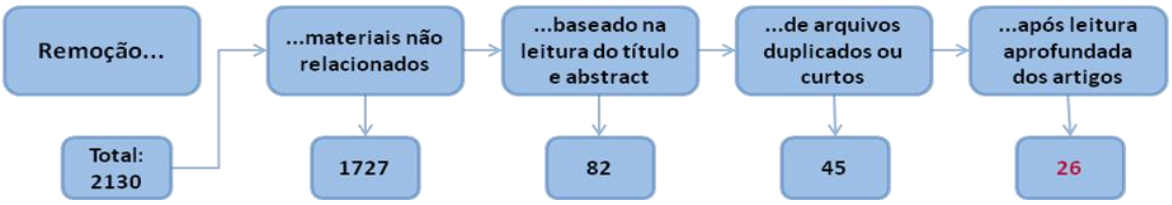


Figura 13 Método de Filtragem

1º Remoção de material não relacionado. Todo o material duplicado e não relacionado foi desconsiderado da amostra dos artigos selecionados para a síntese temática.

Em sua grande maioria, os artigos excluídos eram resumos ou trabalho de conferências, mas também foram removidas dissertações de mestrado, teses de doutorado e alguns documentos editoriais. Esta foi a única atividade realizada apenas por um pesquisador, porque a observação destas características era muito simples.

Desta forma, dos 2.130 artigos iniciais, chega-se ao final deste passo com 1.727 artigos.

2º Remoção baseada na leitura do título e abstract dos artigos. Antes de iniciar este passo foram escolhidos aleatoriamente cinquenta artigos. Todos os pesquisadores participaram desta atividade com a finalidade de equalizar os critérios de exclusão, garantindo a compreensão de todos com relação aos objetivos do trabalho. Apenas dois dos cinquenta artigos inicialmente escolhidos apresentaram algum tipo de divergência dentre os pesquisadores, sendo resolvidas em discussão entre os mesmos. Após esta etapa, os artigos restantes foram avaliados de forma independente por dois pesquisadores. Durante a execução desta etapa de filtragem, a leitura em conjunto e discussões em grupo foram utilizadas para mitigar as dúvidas. Mesmo realizando as mitigações de dúvidas durante este passo, chegou-se a um número de noventa e oito artigos que apresentavam discordância em relação à sua aceitabilidade. Isto representava 5,8% do total de 1.677 artigos (foram trabalhados inicialmente 50 artigos, como já foi dito, mais esses 1.677, resultando no total de 1.727 artigos) restantes para serem trabalhados neste passo. Uma avaliação rigorosa ocorreu entre o grupo com a finalidade de solucionar as discordâncias, resultando em oitenta e dois artigos a serem trabalhados no próximo passo.

3º Remoção de artigos resumidos ou duplicados. Mesmo após a execução do 1º Passo, onde artigos duplicados foram excluídos, ainda foram encontrados alguns remanescentes neste passo. A justificativa para os artigos duplicados persistirem até este passo é o fato de alguns apresentarem os títulos e subtítulos escritos de maneiras diferentes em cada base pesquisada. Enquanto um artigo era escrito normalmente em uma base, em outra ele apresentava caracteres como “dois pontos” ou mesmo “aspas” que os diferenciava na ferramenta adotada.

Os artigos resumidos também foram desconsiderados a partir deste passo, visto que foi percebida uma descrição insuficiente do experimento para atingir os objetivos da

síntese temática. Dos artigos trabalhados durante este passo, restaram 45 para o próximo passo.

4º Remoção após leitura aprofundada dos artigos. Todos os artigos resultantes da etapa anterior foram lidos em profundidade com a finalidade de avaliar a sua permanência dentre os artigos que iriam compor a síntese temática. Os quarenta e cinco artigos foram divididos entre os participantes e durante todo este passo ocorreram discussões entre o grupo, especialmente relacionadas com dúvidas sobre as descobertas dos artigos e sobre o impacto do uso de *smells* na prática do desenvolvimento de *sofware*. As discussões em grupo foram muito úteis para alinhar a visão em relação aos artigos, possibilitando, inclusive, ajustes nos critérios de exclusão. Por exemplo, o critério de exclusão "Artigos que são resumo de outros estudos" só foi definido após a leitura completa dos artigos e compreensão da configuração experimental, facilitando o entendimento da existência de um artigo que sumarizava os resultados de outros estudos.

Após a leitura aprofundada dos artigos e suas devidas exclusões, restaram 26 estudos primários. Uma leitura completa foi realizada com o intuito de ratificar a permanência dos mesmos frente aos critérios de exclusão. Para isto foi considerado como critério de permanência a consonância com os objetivos e questões de pesquisa. Após esta leitura foi confirmada a permanência dos 26 artigos como estudos primários considerados na síntese temática.

Devido ao fato da ferramenta StArt (Fabbri, 2012; Zamboni *et al.*, 2010) não ser colaborativa e não ter uma versão disponível para web, buscamos uma forma de evitar retrabalho e inconsistências com a utilização da mesma para quatro pesquisadores. A estratégia adotada foi deixar apenas um pesquisador com a responsabilidade de manipular a mesma, atualizando as suas informações sempre que necessário. A ferramenta foi utilizada para cada um dos passos mencionados acima. Ao final de cada passo, o arquivo .start era enviado para verificação e acompanhamento dos demais envolvidos, tentando garantir a consistências das informações trabalhadas.

5.2.2 Extração de Dados

O objetivo é tornar o processo mais transparente, evidenciando as dificuldades enfrentadas na prática e as estratégias adotadas para minimizar estas dificuldades. Para tanto, a mesma numeração apresentada na Tabela 14 foi adotada nesta subseção, mantendo a coerência entre teoria proposta e prática realizada.

1. Definir a adoção ou não de uma ferramenta de extração.

Apesar de mencionar que o uso de ferramentas de análise qualitativa auxilia o método, principalmente na etapa de codificação, Cruzes e Dyba (2011a) não sugerem a utilização de uma ferramenta específica e nem mesmo excluem a possibilidade de realização do método de maneira manual. Pesquisamos algumas referências passadas pelos autores para a codificação e estes também não desconsideram a realização de forma manual.

Sendo assim, para a realização da síntese temática não foi adotada ferramenta de extração, conforme justificativas apresentadas na Seção 1.3 desta dissertação. A realização manual de todo o processo de extração exigiu especial atenção na definição da rastreabilidade entre as extrações, códigos, subtemas e temas, tentando evitar inconsistências e retrabalhos. Esta rastreabilidade será apresentada na questão 7 e em um maior nível de detalhes na questão 12.

Uma vez que optamos pela realização de extração e codificação manual, instrumentos foram gerados com a finalidade de tornar o processo mais consistente, diminuindo a possibilidade de retrabalhos e melhorando a confiabilidade da síntese temática.

Os instrumentos geralmente são usados para assegurar que todos os dados relevantes para o método sejam considerados, minimizando o risco de problemas com a transcrição, garantindo maior precisão na checagem e servindo como registro. Cruzes e Dyba (2011a) não mencionam nada a respeito da adoção de instrumentos para extração, visto que não se aprofundam em “como” os passos devem ser executados.

Em um primeiro momento, para auxiliar a extração dos dados, foram utilizados dois instrumentos:

a) Planilha de Extração/Mapeamento

Planilha criada no Excel com a finalidade de facilitar seu reuso para todas as etapas previstas de extração. Foi utilizada para armazenamento inicial das informações extraídas dos estudos primários selecionados para a síntese temática em questão.

A Figura 14 apresenta a Planilha adotada nesta dissertação, especificamente a aba de Levantamento Inicial. Nesta aba focamos nas informações relevantes de extração e codificação dos objetivos dos artigos.

Os campos da planilha responsáveis pela extração dos objetivos são:

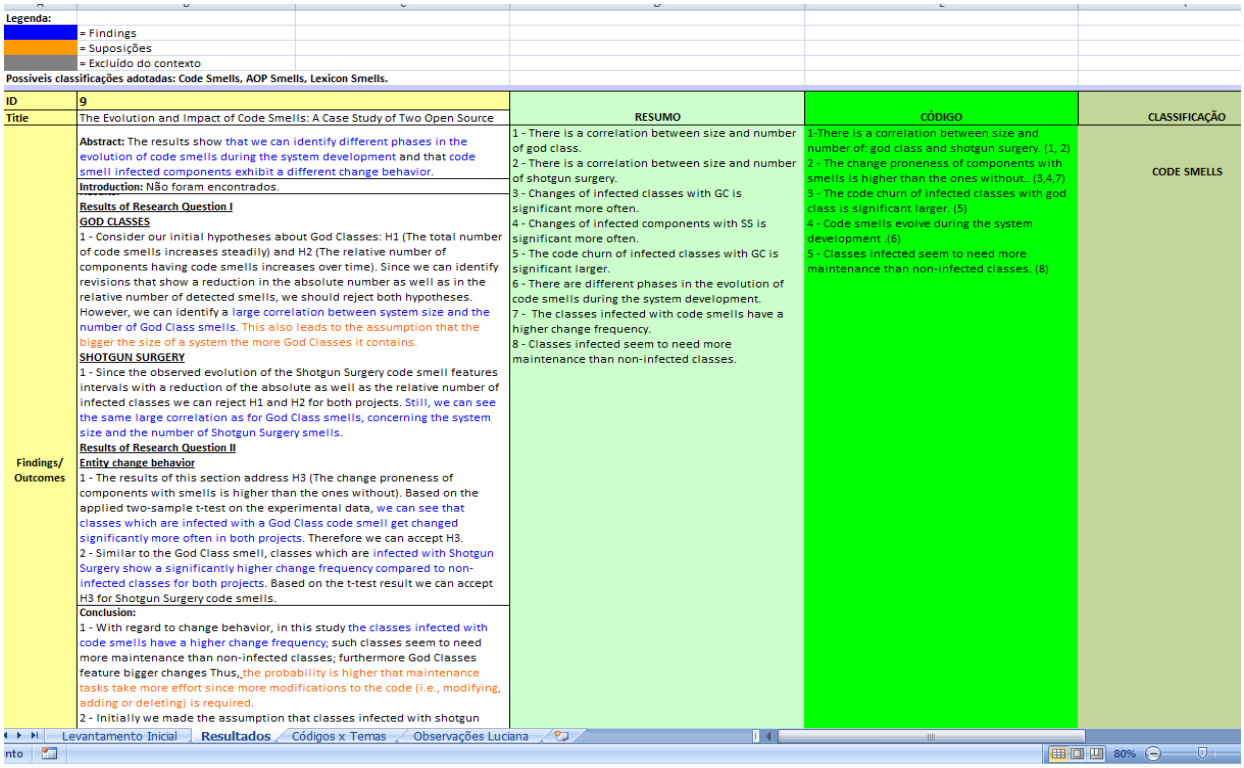
- ID: representa o ID do StArt;
- *Title*: representa o título do artigo:
- *Aim*: representa o foco do artigo;
- *Research Questions/Hypothesis*: representam as questões de pesquisa e/ou hipóteses adotadas pelos artigos selecionados;
- *Method*: representa o método adotado pelo artigo para a representação dos resultados obtidos;
- GQM: representado por um subconjunto das informações que será explicado na questão 13 deste Capítulo. Na linha GQM representa o mapeamento inicial do GQM com base nas transcrições dos segmentos de texto dos artigos. Na segunda coluna, também denominada de "GQM", está a codificação propriamente dita para os objetivos dos artigos.
- Observação: representa as observações importantes encontradas nos artigos e que poderiam auxiliar a extração, codificação, ou mesmo o entendimento dos pesquisadores envolvidos.

	ID	9	GQM
	Title	The Evolution and Impact of Code Smells: A Case Study of Two Open Source Systems	
	Aim	This study focuses on the evolution of code smells within a system and their impact on the change behavior (change frequency and size). The study investigates two code smells, God Class and Shotgun Surgery, by analyzing the historical data over several years of development of two large scale open source systems. We want to know how the number of code smells changes over a long period of development time. We want to investigate if software components associated with smells show a different change frequency than components without smells and if changes to classes with smells force successively larger changes	Analyze (evolution of code) for the purpose of (investigation) with respect to their (God class evolution and Shotgun Surgery evolution) Using (automatic detection) Analyze (evolution of code) for the purpose of (correlation) with respect to their (number of god class AND increases over time) Using (automatic detection) Analyze (evolution of code) for the purpose of (correlation) with respect to their (number of shotgun surgery AND increases over time) Using (automatic detection) Analyze (evolution of code) for the purpose of (correlation) with respect to their (change proneness and god class) Using (automatic detection) Analyze (evolution of code) for the purpose of (correlation) with respect to their (code churn and god class) Using (automatic detection) Analyze (evolution of code) for the purpose of (correlation) with respect to their (change proneness and shotgun surgery) Using (automatic detection) Analyze (evolution of code) for the purpose of (correlation) with respect to their (code churn and shotgun surgery) Using (automatic detection)
	Research Questions/ Hypothesis	Hypothesis One hypothesis is that the bad smells increases over time, because as the software changes are performed, the software degrades. Another hypothesis is that the classes infected by the code smells suffer more and larger changes than the ones that are non-infected. Research Question The paper addresses questions related to the evolution of the code: How do code smells evolve over time and how do code smells influence the change behavior of the infected system elements? 1 - The first research question addresses the evolution of code smells in a software project. H1: The total number of code smells increases steadily. H2: The relative number of components having code smells increases over time. 2 - The second question of interest aims at investigating if we can find evidence that code smells effect component development in terms of frequency and size of changes. H3: The change proneness of components with smells is higher than the ones without. H4: The code churn of changes in infected classes is significantly larger than the code churn on noninfected classes.	
	Method	The study investigates two code smells, God Class and Shotgun Surgery, by analyzing the historical data over several years of development of two large scale open source systems. The detection of code smells in the evolution of those systems was performed by the application of an automated approach using detection strategies. Within the scope of this study we analyzed the historical data of two major Open-Source-Projects, Apache Lucene2 and Apache Xerces 2 J3. The CodeVizard provides functionality to gather and mine data from source code repositories (i.e. Subversion5) and hence the past evolution of the stored systems. The tool also offers various visualizations for examining the infected classes and their change history.	
GQM	Object	Analyzing the historical data over several years of development of two large scale open source systems. The detection of code smells in the evolution of those systems was performed by the application of an automated approach using detection strategies.	
GQM	Purpose	1 - Investigation 2 - Correlation	
GQM	With respect to their	1 - If the bad smells increases over time, because as the software changes are performed, the software can be degraded. 2 - If the classes infected by the code smells suffer more and larger changes than the ones that are non-	
	Observation	This was an exploratory study. The study object are two major Open-Source-Projects, Apache Lucene2 and Apache Xerces 2 (Large Systems). Code smells are design flaws in object-oriented designs that may lead to maintainability issues in the further evolution of the software system. The research questions stated above require an analysis of the evolution of software systems.	

Levantamento Inicial | Resultados | Códigos x Temas | Observações Luciana

Figura 14 Planilha de Extração/Mapeamento – aba Levantamento Inicial

Já a Figura 15 apresenta a aba de Resultados. Nesta aba focamos nas informações relevantes de extração e codificação dos resultados encontrados nos artigos selecionados.



Legenda:
= Findings
= Suposições
= Excluído do contexto
Possíveis classificações adotadas: Code Smells, AOP Smells, Lexicon Smells.

ID	9	RESUMO	CÓDIGO	CLASSIFICAÇÃO
Title	The Evolution and Impact of Code Smells: A Case Study of Two Open Source			
Findings/ Outcomes	**Abstract:** The results show that we can identify different phases in the evolution of code smells during the system development and that code smell infected components exhibit a different change behavior. **Introduction:** Não foram encontrados. **Results of Research Question I** **GOD CLASSES** 1 - Consider our initial hypotheses about God Classes: H1 (The total number of code smells increases steadily) and H2 (The relative number of components having code smells increases over time). Since we can identify revisions that show a reduction in the absolute number as well as in the relative number of detected smells, we should reject both hypotheses. However, we can identify a large correlation between system size and the number of God Class smells. This also leads to the assumption that the bigger the size of a system the more God Classes it contains. **SHOTGUN SURGERY** 1 - Since the observed evolution of the Shotgun Surgery code smell features intervals with a reduction of the absolute as well as the relative number of infected classes we can reject H1 and H2 for both projects. Still, we can see the same large correlation as for God Class smells, concerning the system size and the number of Shotgun Surgery smells. **Results of Research Question II** **Entity change behavior** 1 - The results of this section address H3 (The change proneness of components with smells is higher than the ones without). Based on the applied two-sample t-test on the experimental data, we can see that classes which are infected with a God Class code smell get changed significantly more often in both projects. Therefore we can accept H3. 2 - Similar to the God Class smell, classes which are infected with Shotgun Surgery show a significantly higher change frequency compared to non-infected classes for both projects. Based on the t-test result we can accept H3 for Shotgun Surgery code smells. **Conclusion:** 1 - With regard to change behavior, in this study the classes infected with code smells have a higher change frequency; such classes seem to need more maintenance than non-infected classes; furthermore God Classes feature bigger changes Thus, the probability is higher that maintenance tasks take more effort since more modifications to the code (i.e., modifying, adding or deleting) is required. 2 - Initially we made the assumption that classes infected with shotgun	1 - There is a correlation between size and number of god class. 2 - There is a correlation between size and number of shotgun surgery. 3 - Changes of infected classes with GC is significant more often. 4 - Changes of infected components with SS is significant more often. 5 - The code churn of infected classes with GC is significant larger. 6 - There are different phases in the evolution of code smells during the system development. 7 - The classes infected with code smells have a higher change frequency. 8 - Classes infected seem to need more maintenance than non-infected classes.	1-There is a correlation between size and number of: god class and shotgun surgery. (1, 2) 2 - The change proneness of components with smells is higher than the ones without.. (3,4,7) 3 - The code churn of infected classes with god class is significant larger. (5) 4 - Code smells evolve during the system development .(6) 5 - Classes infected seem to need more maintenance than non-infected classes. (8)	CODE SMELLS

Levantamento Inicial | Resultados | Códigos x Temas | Observações Luciana

Figura 15 Planilha de Extração/Mapeamento – aba Resultados

Na aba Resultados pode-se perceber a existência de quatro campos principais, sendo eles:

- *Findings/Outcomes*: são os segmentos de texto completo, contendo as descobertas encontradas durante a leitura e análise dos 26 artigos selecionados para a síntese temática na área de *Code Smells*. Corresponde a todo o segmento de texto extraído *ipsis litteris* como apresentado no artigo;
- Resumo: um resumo do campo Findings/Outcomes, que realmente referencia uma descoberta do artigo;
- Código: apresenta o código mapeado para representar um ou mais itens contidos no resumo;
- Classificação: campo usado para confirmar se o assunto principal do artigo tratava mesmo de *code smells* e não de outros tipos de *smells* (como *smells* orientados a aspectos, por exemplo).

Todos os segmentos de texto relevantes dos estudos primários eram transcritos para esta planilha, sendo os mesmos organizados pelos tópicos em que apareciam nos artigos (*abstract*, introdução, discussão, conclusão, dentre outros). Neste momento foram considerados todos os resultados e conjecturas dos autores, mas com diferenciação de cor

entre eles. Isso foi feito porque neste momento ainda não estava claro se as suposições seriam consideradas na síntese.

Na segunda etapa da extração, foi realizada uma reunião para avaliar a representatividade destas conjecturas para o objetivo que se pretendia atingir com a síntese temática. Decidimos que as mesmas não seriam consideradas. Mesmo não sendo consideradas, as conjecturas foram importantes para entendimento das ideias dos autores a respeito dos assuntos trabalhados.

b) Ferramenta

Foi desenvolvida uma ferramenta para armazenamento dos dados que contribuiu para o cruzamento das informações, auxiliando as análises e, porteriormente, a escrita da visão descritiva da síntese temática. Este cruzamento poderia ter sido feito de maneira manual, mas preferimos aproveitar a oportunidade para melhorar a confiabilidade dos cruzamentos.

A arquitetura definida foi a MVC (*model-view-controller*), sendo o software composto por visões de entrada e um SGBD, utilizando a linguagem SQL.

A Figura 16 apresenta a ferramenta desenvolvida. Podemos perceber a existência do seguinte *menu*:

- *Authors*: permite o cadastramento dos autores dos artigos selecionados;
- *Codes*: permite o cadastramento dos códigos identificados para os artigos;
- GQM: permite o cadastramento dos GQM de cada artigo;
- *Authors for a paper*: permitia a vinculação dos autores com os artigos selecionados;
- *Papers*: permitia o cadastramento de um artigo. Para tanto as seguintes informações eram necessárias: ID do StArt, ano de publicação, fonte da publicação, tipo do estudo, tipo de sistema utilizado no estudo, tamanho do sistema utilizado no estudo, participantes dos estudos;
- *Sources*: permitia o cadastramento das possíveis fontes de publicação para posterior vinculação no cadastro do artigo;
- *Theme*: permitia o cadastramento dos possíveis tipos de classificação dos estudos (*aspects smells, inter-smells, lexicon bad smells*);
- *Smell*: permitia o cadastramento de todos os *smells* analisados nos artigos selecionados;
- *Smells for a paper:* permitia a vinculação dos *smells* analisados em um artigo.

Figura 16 Ferramenta desenvolvida em CakePHP

Foram realizados *selects* na base de dados da ferramenta para possibilitar os cruzamentos de informações, favorecendo as análises.

Não identificamos relevância na descrição das questões técnicas que envolveram a ferramenta desenvolvida para cruzamento de informações extraídas. Acreditamos que seu uso foi muito pontual e que a mesma não teve uma importância imprescindível para a realização de todo o trabalho.

2. Realizar encontros iniciais para nivelar o conhecimento, equalizando as decisões tomadas para a realização do trabalho (tema central, critérios de inclusão, exclusão, alinhamento das expectativas em relação às questões de pesquisa).

Os encontros iniciais realizados na fase de filtragem foram importantes para nivelar o conhecimento, equalizando as decisões tomadas para a realização do trabalho (discutimos sobre o tema central adotado para a síntese, critérios de inclusão e exclusão, bem como alinhamos as expectativas em relação às questões de pesquisa que seriam trabalhadas).

3. Avaliar a necessidade de um pesquisador especialista no tema central adotado para a síntese temática.

A participação do especialista em *code smell* apresentou-se como uma necessidade para a síntese temática em questão, dado o nível de conhecimento no tema central dos envolvidos na execução do método. A participação do especialista também foi vista como uma oportunidade, já que havia uma necessidade particular do mesmo com os resultados da síntese temática para o seu trabalho de doutoramento na UFBA. Esta necessidade específica evidenciou a relevância de considerar a realização de pesquisas executadas por pesquisadores menos experientes. Como contexto de nossa pesquisa. Estas necessidades concomitantes favoreceram o foco deste trabalho no método proposto por Cruzes e Dyba (2011a).

4. Ler todos os artigos selecionados, buscando imersão no contexto.

Nos passos apresentados por Cruzes e Dyba (2011a) é claramente mencionada a importância da leitura completa dos artigos selecionados. Durante a execução do método, percebemos que a leitura mencionada pelos autores foi necessária desde a fase de filtragem. Esta percepção deu-se por duas questões, sendo elas:

- Era importante garantir que os artigos realmente atendiam aos critérios de inclusão e exclusão, evitando retrabalhos no método;
- Estes retrabalhos poderiam existir frente à ausência de nivelamento sobre o tema centrado adotado para a síntese temática e frente à ausência de domínio do método adotado para a realização desta síntese.

A leitura completa foi feita apenas pela autora desta dissertação e pelo especialista no tema. Percebemos que a execução da leitura completa para todos os pesquisadores envolvidos é relevante com a finalidade de uniformizar o nível de compreensão e percepção a respeito do assunto principal abordado nas questões de pesquisa da RS. Isso não influenciou negativamente a execução deste trabalho, pois os outros dois pesquisadores que não fizeram a leitura completa dos artigos não foram envolvidos em todos os passos da síntese temática, tendo suas participações restritas até a fase de Extração. Consideramos que esta observação também evidencia a importância do contexto observado em nossa pesquisa, onde há poucos recursos disponíveis para execução da pesquisa, especialmente considerando a força de trabalho e diferentes níveis de experiência dos pesquisadores envolvidos.

5. Analisar e definir o arcabouço de extração.

Tendo como base as questões de pesquisa da RS e o tema central da síntese temática, foram identificadas as dimensões que deveriam nortear a extração. Como dimensão deve-se entender os conjuntos de dados definidos pelo arcabouço de extração utilizado na síntese temática, representado pela Figura 17. Nesta Figura, podemos perceber a existência das quatro dimensões adotadas nesta dissertação, sendo elas: publicação, objetivo, contexto e resultados. Cada uma das dimensões apresenta os seus dados relacionados que norteiam as extrações dos artigos selecionados. As relações entre as dimensões também se encontram expressas na Figura. Uma publicação pode ter um ou vários objetivos. Um objetivo apresenta um contexto. E um contexto pode findar em um ou vários resultados.

A comparação entre os dois arcabouços mencionados encontra-se definida no Capítulo 4, Figura 9.

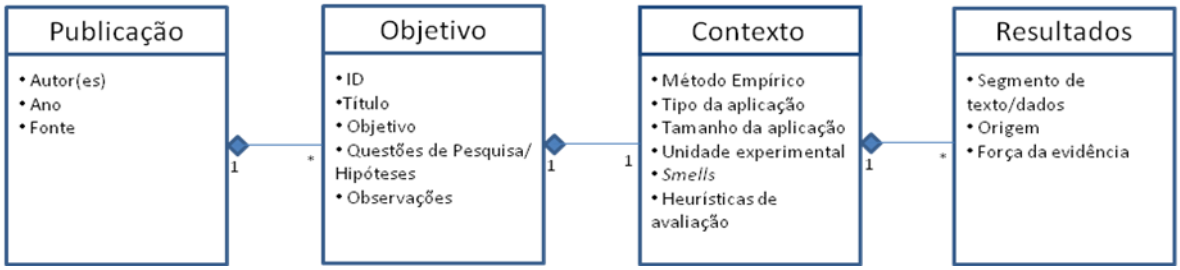


Figura 17 Arcabouço de Extração adaptado de (Cruzes e Dyba, 2011a)

Todos os segmentos de texto que exemplificavam a mesma ideia para as dimensões acima mencionadas foram extraídos. Desde o primeiro momento da extração das informações foram criadas estratégias de rastreabilidade, dado o caráter manual, favorecendo a confiabilidade da síntese temática.

6. Analisar e dividir os artigos selecionados para extração, considerando a execução de etapas iterativas de extração e codificação.

Para que este passo da extração ocorresse da melhor forma possível, os vinte e seis artigos selecionados como insumo foram divididos em etapas, tentando gerar um balanceamento para não desequilibrar a realização do trabalho. Isto evitou que um grande volume de dados fosse trabalhado, permitindo alinhamento de ideias dos pesquisadores sobre a extração de informações úteis para a síntese temática.

No total foram realizadas cinco etapas de extração/codificação. Na primeira etapa foram trabalhados os artigos que o especialista sinalizou como prováveis de fornecer maior aprofundamento no tema central abordado. Por isto a primeira etapa apresentou uma maior quantidade de artigos. Para as demais etapas, a escolha dos artigos foi aleatória, tentando apenas não ultrapassar o limite de seis artigos por etapa. Identificamos que iniciar a primeira etapa com uma menor quantidade de artigos poderia ter facilitado a execução do trabalho, pois as dificuldades e falta de maturidade no método são mais evidentes no início de sua execução. Acreditamos na possibilidade de haver melhores resultados se a separação dos artigos por etapa seguir critérios bem definidos. Um exemplo de critério seria em relação ao grau de relevância das descobertas identificadas na leitura realizada ao final do passo de filtragem.

Para justificar a divisão dos artigos em etapas, apresentamos algumas justificativas:

- O primeiro fato está relacionado ao grande número de artigos selecionados para trabalhar na síntese temática;
- Outro fato é a necessidade de buscar uma visão geral sobre o que havia de mais interessante dentre os artigos que seriam trabalhados, facilitando o processo de amadurecimento dos pesquisadores menos experientes. Isto é especificamente importante na adoção da abordagem de codificação indutiva. Neste caso, os artigos mais representativos trazem ganhos na medida em que apresentam termos e percepções de possíveis temas relevantes para o assunto principal que estava sendo trabalhado.

Na Tabela 15 pode-se perceber a distribuição dos artigos trabalhados em cada etapa da extração.

Tabela 15 Etapas de extração e seus respectivos artigos

Etapa	ID START	Total
01	9, 26, 38, 62, 63, 148, 278, 697	08
02	10, 210, 260, 276, 277	05
03	60, 334, 419, 429, 461, 823	06
04	427, 522, 526, 747, 1509	05
05	1941, 2000	02
TOTAL DE ARTIGOS		**26**

Cruzes e Dyba (2011a) mencionam a possibilidade de iteratividade entre os passos por eles apresentados para a realização de síntese em ES. Durante a prática deste trabalho foi percebida a necessidade de realização dos passos apresentados considerando o ciclo de vida do método como sendo incremental e iterativo. Este fato permitiu rodá-lo em pequenas partes que iam trazendo maior entendimento sobre o assunto principal e, principalmente, sobre as possíveis melhorias a serem realizadas no ciclo seguinte do método adotado.

O detalhamento das informações relacionadas aos artigos selecionados para a síntese temática pode ser analisado em detalhes na Tabela 16.

Tabela 16 Detalhes da extração, seus respectivos artigos e referências

Etapa	ID START	Referência	Título	Total Etapa
01	9	(Olbrich, 2009)	*The Evolution and Impact of Code Smells: A Case Study of Two Open Source Systems*	08
	26	(Schumacher, 2010)	*Building empirical support for automated code smell detection*	

	38	(Yamashita e Moonen, 2013b)	*Exploring the Impact of Inter-smell Relations on Software Maintainability: An Empirical Study*	
	62	(Yamashita, Counsell, 2013)	*Code smells as system-level indicators of maintainability: An empirical study*	
	63	(Yamashita e Moonen, 2013c)	*To what extent can maintenance problems be predicted by code smell detection? An empirical study*	
	148	(Sjøberg, 2013)	*Quantifying the Effect of Code Smells on Maintenance Effort*	
	278	(Macia *et al.*, 2012a)	*Are Automatically-detected Code Anomalies Relevant to Architectural Modularity?: An Exploratory Analysis of Evolving Systems*	
	697	(Li, Shatnawi, 2007)	*An empirical study of the bad smells and class error probability in the post-release object-oriented system evolution*	
02	10	(Santos e Mendonça, 2014)	*An exploratory study to investigate the impact of conceptualization in god class detection*	05
	210	(Yamashita, Moonen, 2012)	*Do Code Smells Reflect Important Maintainability Aspects?*	
	260	(Pope e Mays, 1995)	*Clones: what is that smell?*	
	276	(Macia *et al.*, 2012b)	*On the Relevance of Code Anomalies for Identifying Architecture Degradation Symptoms*	
	277	(Peters, Zaidman, 2012)	*Evaluating the Lifespan of Code Smells Using Software Repository Mining*	
03	60	(Zazworka *et al.*, 2011)	*Investigating the Impact of Design Debt on Software Quality*	06
	334	(Marinescu e Marinescu, 2011)	*Are the clients of flawed classes (also) defect prone?*	
	419	(Olbrich *et al.*, 2010)	*Are all code smells harmful? A study of God Classes and Brain Classes in the evolution of three open source systems*	
	429	(Chatzigeorgiou e Manakos, 2010)	*Investigating the Evolution of Bad Smells in Object-Oriented Code*	
	461	(D'Ambros, Bacchielli e Lanza, 2010)	*On the Impact of Design Flaws on Software Defects*	
	823	(Mantyla, Vanhanen, Lassenius, 2005)	*Bad Smells Humans as Code Critics*	
04	427	(Carneiro *et al.*, 2010)	*Identifying code smells with multiple concern views*	05
	522	(Khomh, Penta, Gueheneuc, 2009)	*An Exploratory Study of the Impact of Code Smells on Software Change-proneness*	
	526	(Vauche, 2009)	*Tracking Design Smells: Lessons from a Study of God Classes*	
	747	(Mantyla, Lassenius, 2006)	*Subjective evaluation of software evolvability using code smells: An empirical study*	
	1509	(Mantyla, 2005)	*An experiment on subjective evolvability evaluation of object-oriented software: explaining factors and interrater agreement*	
05	1941	(Yamashita e Moonen, 2013a)	*Do developers care about code smells? An exploratory survey*	02
	2000	(Fontana, 2013)	*Investigating the Impact of Code Smells on System's Quality: An Empirical Study on Systems of Different Application Domains*	
TOTAL DE ARTIGOS				**26**

Durante o passo de extração foram identificados três artigos que, apesar de estarem adequados aos critérios de inclusão e exclusão, tratavam de particularidades de *smells* que não era o foco pretendido nesta síntese temática. Dois tratavam especificamente de *code smells* relacionados a aspectos (Macia, 2011a; Macia *et al.*, 2011b) e um tratava de *lexicon code smells* (Abebe *et al.*, 2011).

Apesar de terem chegado até esta fase, estes três artigos foram considerados fora do escopo pretendido. Mesmo assim, esta decisão foi levada para discussão com o pesquisador independente. Como estes artigos compunham a última etapa prevista de extração/codificação, ao final desta decisão de exclusão, a etapa ficou apenas com dois artigos válidos.

7. Extrair os dados dos estudos selecionados, observando na primeira etapa as principais seções onde as fontes de informação são comumente encontradas.

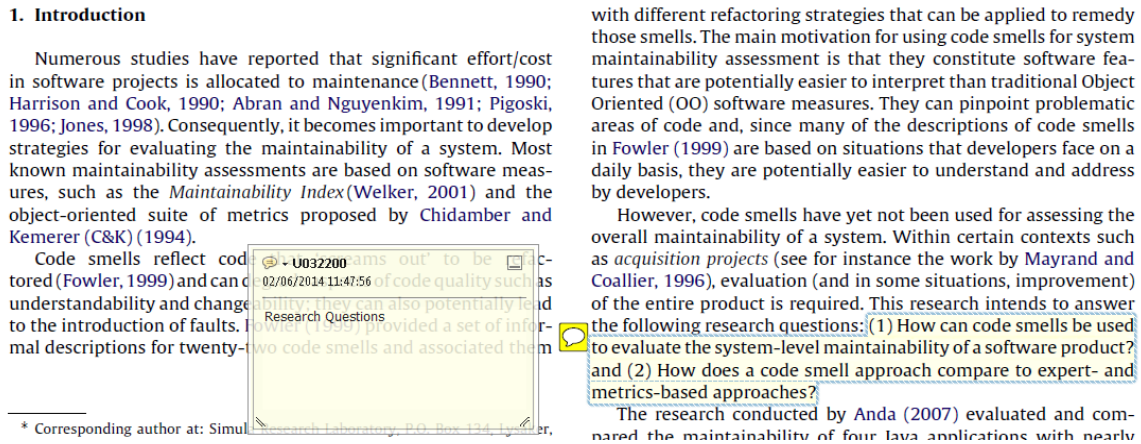
Toda a extração envolveu duas etapas importantes:

- Marcação dos dados nos artigos com comentários: para esta etapa três pesquisadores realizavam a busca e marcação das informações relevantes. Depois, o especialista no tema avaliava se havia alguma discordância. Sempre que necessário eram realizadas reuniões para discutir e se chegar a um consenso sobre a discordância. As seções apresentadas por Cruzes e Dyba (2011a) como sendo as mais comuns na identificação das informações contidas nas dimensões consideradas foram úteis e ajudaram na etapa de extração.
- Extração das informações para compor a Planilha de Extração/Mapeamento e a ferramenta desenvolvida: para esta etapa apenas dois pesquisadores trabalharam, sendo a autora desta dissertação e o especialista no tema, no papel de revisor. Todas as dimensões extraídas, tais como publicação, objetivo, contexto e resultados foram verificados pelo especialista no tema. Quando qualquer discordância era encontrada, a planilha era marcada. Depois ocorria uma reunião para atingir um consenso. Caso esse consenso não chegasse, um pesquisador independente era envolvido.

Vale ressaltar que a marcação dos dados com os devidos comentários nos artigos se deu como uma forma de garantir a validação das informações extraídas, na tentativa de aumentar o nível de confiabilidade do método. Estes comentários acrescidos às marcações

feitas nos artigos referenciavam exatamente o item da Planilha de Extração/Mapeamento para o qual a informação era transcrita.

Como exemplo de rastreabilidade para garantir a validação das informações, podemos citar o caso do artigo trabalhado (Yamashita e Counsell, 2013). Para este artigo, a Figura 18 apresenta as marcações realizadas no corpo do mesmo que referenciam as questões de pesquisa apresentadas.


1. Introduction

Numerous studies have reported that significant effort/cost in software projects is allocated to maintenance (Bennett, 1990; Harrison and Cook, 1990; Abran and Nguyenkim, 1991; Pigoski, 1996; Jones, 1998). Consequently, it becomes important to develop strategies for evaluating the maintainability of a system. Most known maintainability assessments are based on software measures, such as the *Maintainability Index* (Welker, 2001) and the object-oriented suite of metrics proposed by Chidamber and Kemerer (C&K) (1994).

Code smells reflect cod... ...ams out' to be ...c-tored (Fowler, 1999) and can c... ...of code quality such as understandability and change... ...bility; they can also potentially lead to the introduction of faults. ...owler (1999) provided a set of infor-mal descriptions for twenty-t...wo code smells and associated them

U032200
02/06/2014 11:47:56
Research Questions

with different refactoring strategies that can be applied to remedy those smells. The main motivation for using code smells for system maintainability assessment is that they constitute software features that are potentially easier to interpret than traditional Object Oriented (OO) software measures. They can pinpoint problematic areas of code and, since many of the descriptions of code smells in Fowler (1999) are based on situations that developers face on a daily basis, they are potentially easier to understand and address by developers.

However, code smells have yet not been used for assessing the overall maintainability of a system. Within certain contexts such as *acquisition projects* (see for instance the work by Mayrand and Coallier, 1996), evaluation (and in some situations, improvement) of the entire product is required. This research intends to answer the following research questions: (1) How can code smells be used to evaluate the system-level maintainability of a software product? and (2) How does a code smell approach compare to expert- and metrics-based approaches?

The research conducted by Anda (2007) evaluated and compared the maintainability of four Java applications with nearly

* Corresponding author at: Simul... ...Research Laboratory, P.O. Box 134, ...er,

Figura 18 Marcação das Questões de Pesquisa no corpo do artigo de Yamashita e Counsell (2013)

Estas mesmas questões de pesquisa apresentam-se na Figura 19, transcritas para o item *Research Question/Hypothesis* da Planilha de Extração/Mapeamento.

ID	62
Title	Code smells as system-level indicators of maintainability: An empirical study
Aim	The research in this paper investigates the potential of code smells to reflect system-level indicators of maintainability.
Research Questions/ Hypothesis	(1) How can code smells be used to evaluate the system-level maintainability of a software product? (2) How does a code smell approach compare to expert- and metrics-based approaches?

Figura 19 Questões de Pesquisa do artigo de Yamashita e Counsell (2013) transcritas

Outro exemplo completo de rastreabilidade será fornecido na Seção 5.2.3, questão 12 desta dissertação.

Inicialmente as informações extraídas foram utilizadas para o preenchimento da Planilha de Extração/Mapeamento e posteriormente as mesmas foram migradas para a ferramenta desenvolvida, sendo devidamente complementadas pelas informações de contexto.

As estratégias adotadas para manter a rastreabilidade foram:

- Marcações nos segmentos de texto dos artigos que foram extraídos, considerando informações referentes às quatro dimensões consideradas (publicação, objetivo, contexto e resultados); Transcrição dos segmentos de

texto marcados nos artigos para os devidos campos da Planilha de Extração/Mapeamento;

- Marcações de segmentos de texto na Planilha de Extração/Mapeamento dos itens que foram considerados importantes para se chegar aos resumos dos resultados encontrados nos artigos;
- Numeração atribuída aos textos que resumiam os resultados encontrados na Planilha de Extração/Mapeamento;
- Numeração atribuída ao código descritivo encontrado com a análise dos resumos;
- Vinculação entre as numerações atribuídas aos códigos e aos resumos que os originaram, demonstrando claramente a relação entre os mesmos;
- Organização dos temas e subtemas no Mapa Temático considerando estas numerações atribuídas aos códigos na Planilha de Extração/Mapeamento.

Apresentaremos, a seguir, a extração de cada uma das quatro dimensões que compõem o arcabouço de extração.

Publicação

As informações relacionadas à Publicação foram extraídas diretamente para a ferramenta desenvolvida, complementando as demais informações oriundas da Planilha de Extração/Mapeamento. As buscas nas bases eletrônicas de dados foram feitas considerando o período compreendido entre os anos de 2002 a 2013. Foi considerado para este trabalho que as informações "autor (es)", "ano" e "fonte" seriam dados demográficos.

A distribuição dos artigos pelas respectivas fontes, a distribuição dos artigos por ano, por autor, a caracterização dos artigos, bem como suas análises, podem ser encontradas na Seção 6 do Anexo 1 desta dissertação.

Os dados que compõem esta dimensão não apresentaram dificuldades para a extração. Não foram detectadas seções específicas em que os mesmos são encontrados.

Objetivo

Como já descrito, o Objetivo foi dissociado para uma dimensão específica, dada a sua relevância e influência direta nos resultados encontrados pelos artigos. Durante a leitura completa para imersão nos artigos selecionados, percebemos que a relação *objetivo x contexto x resultados* apresentava forte correlação. Era necessário obter o máximo de entendimento sobre cada uma destas dimensões, com a finalidade de tentar facilitar a realização da síntese temática.

A extração da dimensão Objetivo compreendeu a identificação dos seguintes dados nos artigos selecionados: ID do Start, título, foco, questões de pesquisa/hipóteses, observações. O termo "*aim*" adotado nas Planilhas de Extração/Mapeamento foi traduzido como sendo "foco" dos artigos. Este fato se deu considerando a necessidade de diferenciação do mesmo em relação ao termo "objetivo". Este aqui sendo utilizado para denominar uma dimensão trabalhada no arcabouço de extração.

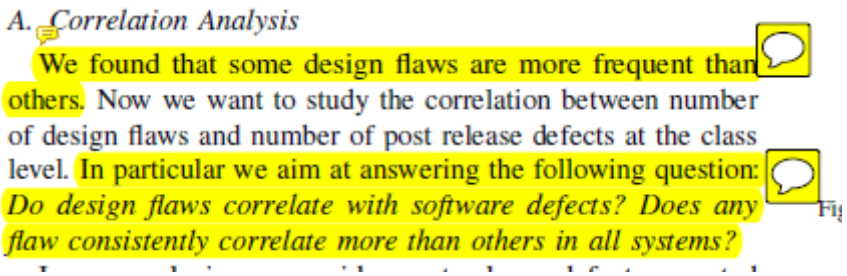
Com base nestas informações extraídas, o item 5.2.3 desta dissertação irá apresentar a utilização da abordagem baseada em GQM para a codificação desta dimensão.

Não foram encontradas dificuldades de extração dos segmentos de texto que ajudavam a identificar claramente o foco dos artigos selecionados, como se pode perceber no exemplo do artigo (Schumacher, 2010):

> **Abstract**: "The study investigates the way professional software developers detect god class code smells, then compares these results to automatic classification."
> **Introduction**: "Our study aims to improve the understanding how humans utilize the concept of code smells in improving software quality. We investigate how expert developers detect god classes and whether it is a repeatable process (e.g., whether there is agreement among multiple judges). Based on observations of real developers and real systems, we can begin to formulate the symptoms that let humans identify a god class as such."
> **Conclusion**: "The purpose of this thesis was to find empirical support for the detection of code smells."

Alguns artigos apresentam uma estruturação que não ajuda na detecção e extração de todos os dados necessários, por não os trazer de forma explícita ou organizada. Algumas características dos estudos empíricos não estão centralizadas em um tópico estruturado, apresentando-se soltas pelo texto do artigo, sendo necessário mais tempo para encontrá-las e conectá-las. Um exemplo para este caso poderia ser refletido no artigo que faz um estudo de correlação para analisar o impacto de *design flaws* nos defeitos de *software* (D'Ambros, Bacchielli e Lanza, 2010), onde estas as informações não se apresentam de forma estruturada.

A Figura 20 apresenta um exemplo extraído deste artigo para a Seção IV denominada "*Experiments*". Nesta figura podemos perceber que na primeira marcação o autor nos apresenta um resultado encontrado em seu artigo e na segunda marcação podemos perceber as questões de pesquisa que o artigo trabalha. Não foram consideradas seções mais estruturadas para que houvesse uma representação clara do estudo realizado. Desta forma, era importante ler com cautela e tentar identificar os dados que necessitavam ser extraídos.

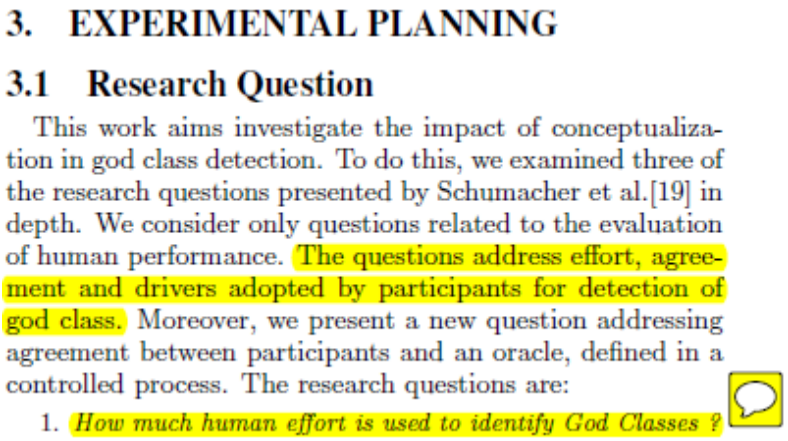


A. Correlation Analysis

We found that some design flaws are more frequent than others. Now we want to study the correlation between number of design flaws and number of post release defects at the class level. In particular we aim at answering the following question: *Do design flaws correlate with software defects? Does any flaw consistently correlate more than others in all systems?*

Figura 20 Extração do artigo de D'Ambros, Bacchielli e Lanza (2010) que representa falta de estruturação

Os artigos que apresentavam questões de pesquisa estruturadas tratavam a Seção "*Discussion*" relacionando as discussões às questões de pesquisa apresentadas anteriormente na Seção *Research Question*. Nas Figuras 21 e 22 podemos notar o exemplo do artigo (Santos e Mendonça, 2014) que promoveu um estudo exploratório para investigar o impacto da conceitualização da detecção de *God Class*.



3. EXPERIMENTAL PLANNING

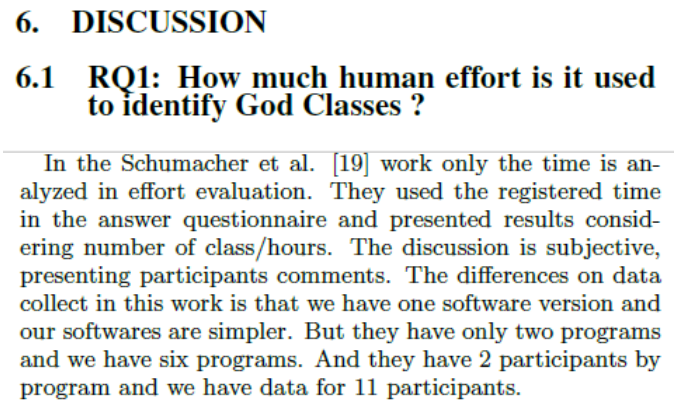
3.1 Research Question

This work aims investigate the impact of conceptualization in god class detection. To do this, we examined three of the research questions presented by Schumacher et al.[19] in depth. We consider only questions related to the evaluation of human performance. The questions address effort, agreement and drivers adopted by participants for detection of god class. Moreover, we present a new question addressing agreement between participants and an oracle, defined in a controlled process. The research questions are:

1. *How much human effort is used to identify God Classes ?*

Figura 21 Organização do artigo de Santos e Mendonça (2014) – Questões de Pesquisa

Esse artigo apresenta suas questões de pesquisa descritas na respectiva Seção, conforme apresentado na Figura 21. Apresenta também na Seção "*Discussion*" os resultados encontrados em função de cada questão de pesquisa abordada, conforme pode ser percebido na Figura 22.



6. DISCUSSION

6.1 RQ1: How much human effort is it used to identify God Classes ?

In the Schumacher et al. [19] work only the time is analyzed in effort evaluation. They used the registered time in the answer questionnaire and presented results considering number of class/hours. The discussion is subjective, presenting participants comments. The differences on data collect in this work is that we have one software version and our softwares are simpler. But they have only two programs and we have six programs. And they have 2 participants by program and we have data for 11 participants.

Figura 22 Organização do artigo de Santos e Mendonça (2014) – Trecho da discussão dos resultados

Cruzes e Dyba (2011a) mencionam que é comum na área de ES enfrentar alguma dificuldade para extração dos objetivos de alguns artigos. Neste caso, consideramos que este objetivo mencionado pelos autores seria compatível com o que denominamos "foco". Especificamente para o item "foco" isso não se confirmou para este trabalho. Porém algumas dificuldades se apresentaram ao longo do método, como por exemplo, o artigo de Yamashita e Moonen (2013b) não apresenta explicitamente suas hipóteses/questões de pesquisa. Estas consideradas importantes para subsidiar a codificação, considerando o método GQM adotado. Este método será explicado na Seção 5.2.3 desta dissertação.

Diante desta dificuldade, apenas como exemplo do que foi enfrentado, foi realizada uma reunião para tentar identificar um segmento do texto que apresentasse algo relevante que pudesse ser considerado para nortear esse quesito para o artigo em questão. O seguinte segmento de texto foi nomeado como um item importante que poderia ser considerado para nortear as questões dependentes:

> "… *Based on these observations, we conjectured that interaction effects between co-located smells (i.e., smells located in the same artifact) can intensify problems caused by individual code smells or lead to additional, unforeseen maintenance issues*." (Yamashita e Moonen, 2013, p. 01, grifo nosso).

Contexto

A extração desta dimensão compreendeu a identificação dos seguintes dados nos artigos selecionados: método, tipo da aplicação, tamanho da aplicação, unidade experimental, *smells*, heurísticas de avaliação.

Para o método encontramos basicamente experimentos controlados *in vitro*, experimentos controlados *in vivo*, estudos de caso, estudos de correlação e *surveys*. O que

denominamos de estudos de correlação, para este trabalho, foram os estudos em que os repositórios eram avaliados, para analisar a relação dos *smells* com determinadas características de desenvolvimento.

Para o tipo de aplicação encontramos: *open source*, comercial e outras construídas especificamente para os experimentos. Algumas também não devidamente especificadas pelos artigos.

Para o tipo de aplicação encontramos: pequena, média e grande, sendo que alguns artigos também não especificavam claramente.

Para unidade experimental encontramos: profissionais, estudantes de graduação, estudantes de pós-graduação. Percebemos que apenas os estudos de correlação não especificaram participação humana nos experimentos.

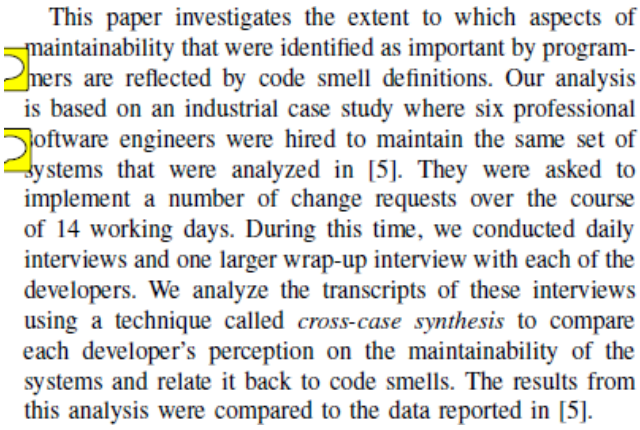
Foram detectados os seguintes *smells*: AbstractClass, Alternative Classes with Different Interfaces, ChildClass, ClassGlobalVariable, ClassOneMethod, ComplexClassOnly, Composition Bloat, ControllerClass, Data Class, Data Clump, Dead Code, Divergent Change, Duplicate Pointcut, Duplicated code, Feature envy, FewMethods, FieldPrivate, FieldPublic, Forced Join Point, FunctionClass, God Aspect, God class, God Method, God Pointcut, HasChildren, Inappropriate Intimacy, Incomplete Library Class, Interface Segregation Principle Violation, Large class, LargeClassOnly, Lazy Class, Long Method, Long Parameter List, LowCohesionOnly, ManyAttributes, Message Chains, MethodNoParameter, Middle Man, Misplaced Class, MultipleInterface, NoInheritance, NoPolymorphism, NotAbstract, NotComplex, OneChildClass, Parallel Inheritance Hierarchies, ParentClassProvidesProtected, Primitive Obsession, RareOverriding, Redundant Pointcut, Refused Bequest, Shotgun surgery, Small Class, Speculative Generality, State Checking, Switch statements, Temporary Field, Temporary variable used for several purposes, TwoInheritance, Use interface instead of implementation.

Para as heurísticas de avaliação consideradas, detectamos: Marinescu (Thesis, 2002), avaliação humana, Lanza and Marinescu (2005), heurística criada pelos próprios autores (para código duplicado), N. Tsantalis, 2010, F. Khomh, S. Vaucher, Y.-G. Gueheneuc (2009).

As informações relacionadas ao Contexto foram extraídas diretamente para a ferramenta desenvolvida, complementando as demais informações oriundas da Planilha de Extração/Mapeamento.

Enfrentamos dois problemas em relação à extração dos dados de contexto, sendo eles:

- Alguns artigos não mencionam explicitamente a adoção de um determinado tipo de método ou heurística de avaliação adotada. Com a leitura e análise realizadas, em reuniões de consenso, estes foram classificados, considerando as suas características. Como exemplo, podemos citar: o artigo de Carneiro *et al.* (2010) que ao analisarmos percebemos tratar-se de um experimento controlado *in vitro.*
- Outros artigos chegam a mencionar explicitamente a adoção de um determinado tipo de método. Porém, com a leitura e análise realizadas, em reuniões de consenso, estes foram reclassificados considerando as suas características. Um exemplo relacionado a este item ocorreu com o artigo de Yamashita e Moonen (2012). O artigo menciona tratar-se de um estudo de caso, mas ao analisarmos o mesmo, percebemos tratar-se de um experimento controlado *in vivo*. Esta reclassificação mostrou-se necessária, dado o argumento: estudo ter sido realizado na empresa enquanto o software era desenvolvido, conforme Figura 23.



This paper investigates the extent to which aspects of maintainability that were identified as important by programmers are reflected by code smell definitions. Our analysis is based on an industrial case study where six professional software engineers were hired to maintain the same set of systems that were analyzed in [5]. They were asked to implement a number of change requests over the course of 14 working days. During this time, we conducted daily interviews and one larger wrap-up interview with each of the developers. We analyze the transcripts of these interviews using a technique called *cross-case synthesis* to compare each developer's perception on the maintainability of the systems and relate it back to code smells. The results from this analysis were compared to the data reported in [5].

Figura 23 Classificação do artigo de Yamashita e Moonen (2012) – Trecho da Introdução

- Devido à ausência de uma estruturação de alguns artigos, encontramos dificuldades na extração de informações que se apresentavam dispersas sem uma lógica que facilitasse a detecção das mesmas com base nos tópicos apresentados nos estudos.

Não foram detectadas seções específicas em que os mesmos dados extraídos do contexto são mais facilmente encontrados.

Resultados

Para a extração dos resultados dos artigos ou "*findings*", não temos dados associados como nas demais dimensões. Sendo assim, é importante entender o objetivo do artigo (em relação à dimensão mesmo) e seu contexto, facilitando a identificação dos resultados aos quais os artigos se referem.

Existia uma preocupação de extrair as partes completas do texto com a finalidade de prover as informações suficientes para que fosse possível deixar visíveis as descobertas dos estudos. Esta recomendação foi tratada como algo importante por Floersch *et al.* (2010), sendo seguida nesta dissertação.

Em alguns casos foi necessário copiar um parágrafo inteiro para conseguir extrair a ideia principal do achado para o artigo.

Apresentaremos um exemplo do artigo (Macia *et al.*, 2012a) , onde foi extraída a parte dos resultados apresentados nas seções "*Abstract*" e "*Study Results*":

> ***Abstrac****t: "The outcome of our evaluation suggests that many of the code anomalies detected by the employed strategies were not related to architectural problems. Even worse, over 50% of the anomalies not observed by the employed techniques (false negatives) were found to be correlated with architectural problems."*
> ***Study results:*** *"In general, our analysis reveals that detection strategies are inaccurate in identifying architecturally-relevant code anomalies. Specifically, most of the automatically-detected code anomalies were not associated with architectural modularity problems, leading to many false positives. In general, the average of the automatically-detected code anomalies represented about 45% (or less) of the total number of code anomaly related to architectural modularity problems.*
> **…**
> *Even worse, many of the code anomalies harmful to architectural modularity problems were not automatically detected by strategies, leading to a high rate of false negatives. About 55% or more of the non automatically-detected code anomalies were related to architectural modularity problems. These results indicate that detection strategies seem to have a tendency to send developers in wrong* directions when addressing code anomaly related to architectural modularity problem." (Macia *et al.*, 2012, p. 01).

A partir da informação textual extraída, as constatações dos estudos eram resumidas. Para detectar os resultados em meio à quantidade de segmentos de texto extraídos, eram aplicadas as questões de pesquisa/hipóteses no intuito de facilitar esta identificação. Com as extrações finalizadas na Planilha de Extração/Mapeamento para uma determinada etapa, iniciava-se o passo de codificação dos dados desta etapa, formando assim um processo cíclico e incremental de extração e codificação.

Os resultados foram encontrados pulverizados e repetidos ao longo dos artigos selecionados. A maioria dos resultados relevantes estava descrita na introdução, na Seção "*Discussion*", "*Results*" e na conclusão.

8. Verificar a extração.

A verificação das etapas está diretamente relacionada à confiabilidade da síntese e foi adquirida da seguinte forma:

- Durante a execução houve verificação realizada pelo especialista em relação ao que era filtrado ou extraído;
- Sugestões e comentários eram registrados pelo verificador com o intuito de prover a concordância;
- Para os casos de discordância, uma reunião era realizada para discussão e devidos esclarecimentos;
- Para os casos em que não se chegasse a um consenso, mesmo após a reunião, um terceiro pesquisador independente era consultado.

Os critérios de verificação para este passo foram relacionados à necessidade de garantir que os principais dados relacionados com cada uma das dimensões consideradas no arcabouço de extração foram extraídos e que a rastreabilidade tinha sido mantida entre a extração e os segmentos de texto do artigo.

Esta forma de garantir a confiabilidade dos dados que seriam utilizados na síntese temática se estendeu para toda a aplicação do método neste trabalho.

Apesar de explicitar que a realização de verificações era uma atividade importante para a realização do método, Cruzes e Dyba não se aprofundam em como esta verificação poderia ser feita. Desta forma, foi necessário que os passos acima mencionados fossem combinados entre os pesquisadores participantes, favorecendo a prática. Vale ressaltar que a estratégia adotada otimizou o esforço para a realização da atividade, considerando a disponibilidade dos envolvidos: um pesquisador realizava a tarefa de forma independente; outro pesquisador verificava o resultado e realizava comentários; e em uma etapa posterior requerendo reuniões presenciais, o consenso era alcançado.

5.2.3 Codificação de Dados

A partir desta etapa, os pesquisadores envolvidos foram a autora desta dissertação, o especialista no tema, como revisor, e o terceiro pesquisador independente acionado em caso de dúvidas e/ou conflitos.

9. Definir as dimensões que serão codificadas.

O passo de codificação foi realizado para as dimensões Objetivo e Resultado. Estas são dimensões apresentadas em forma textual, em partes diferentes do texto, conforme discutido anteriormente. Não havia justificativa para a codificação das dimensões Publicação e Contexto, visto que seus dados apresentavam-se autossuficientes e claros. Mesmo não tendo sido codificada, a dimensão Contexto foi amplamente utilizada na codificação das demais dimensões e na identificação da relevância das informações extraídas.

10. Definir os métodos de codificação que serão adotados, considerando a natureza da dimensão e sua representatividade nas análises.

A codificação para estas duas dimensões (Objetivo e Resultado) não ocorreu de maneira padrão, visto a heterogeneidade dos tipos de dados inerentes a cada uma e sua representatividade nas análises posteriores.

Cruzes e Dyba (2011a) explicitam que a análise temática não especifica uma técnica de codificação e o próprio artigo também não apresenta detalhe de técnicas que poderiam ser utilizadas para a realização da etapa de codificação. O que o artigo traz é a referência a algumas leituras nas quais várias técnicas são apresentadas.

Para realização prática da codificação foi necessário analisar as técnicas existentes e adotar as que mais se adequavam à realidade das dimensões analisadas e do objetivo da síntese temática. Uma leitura das indicações propostas foi realizada para obtenção de maior conhecimento das técnicas existentes. Estas leituras possibilitaram a definição e adoção das técnicas que pareciam estar mais apropriadas à síntese.

Após reuniões de análise e discussão entre os pesquisadores envolvidos nesta etapa, ficou definido:

- Adaptação do método GQM para codificação da dimensão Objetivo. Buscava-se com esta adoção obter um maior entendimento e simplificar as

informações relacionadas aos objetivos dos artigos. Esta adoção precisava ser assertiva para que pudesse colaborar com as análises posteriores dos artigos, sendo um passo construtivo para a síntese temática;

- Utilização da ideia proposta por Saldaña (2008) ao descrever o método chamado de codificação descritiva para a dimensão Resultado. A adoção deste método deu-se por alguns fatores. Um deles foi pelo fato do mesmo ser de simples aplicabilidade e indicado para iniciantes, que era o nosso caso. Outro fator foi relacionado à questão do mesmo ser apropriado para quase todos os estudos qualitativos.

Ambas as codificações foram realizadas de forma manual e cíclica (não linear), tendo seus resultados justificados na Planilha de Extração/Mapeamento e alguns exemplos apresentados nesta dissertação.

11. Definir a abordagem para realização da codificação (dedutiva, indutiva, integrada).

O método de Cruzes e Dyba (2011a) deixa clara a necessidade de criação de uma lista prévia com definições e frequências dos códigos, independentemente da abordagem adotada para a realização do trabalho. Esta necessidade aparece no *Checklist* proposto pelos autores no Capítulo 3, mas não é mencionada de forma mais detalhada no texto dos autores para que permitisse uma melhor compreensão do momento da sua geração e da necessidade. Sendo assim, alinhamos que a presença do termo "prévia" se referindo à lista em questão poderia significar que a mesma deve ser gerada antes da identificação dos códigos.

Os autores mencionam também a necessidade de adoção de uma abordagem para condução da codificação: indutiva, dedutiva ou integrada, não obrigando a adoção de nenhuma das três.

Diante do exposto, adotamos a abordagem indutiva, onde os códigos surgem e são atribuídos após uma análise dos dados extraídos. Isto é, sem uma listagem prévia dos possíveis códigos que podem ser trabalhados para um determinado conceito. Esta forma de trabalhar estaria mais associada à abordagem dedutiva. Devido ao fato de adotarmos a abordagem indutiva e de termos alinhado o entendimento de que a lista prévia seria mais relevante para a abordagem dedutiva, não fizemos esta lista para a síntese temática gerada para esta dissertação.

Para explicitar melhor nossa adoção pela abordagem indutiva, seguem as principais questões avaliadas:

- A participação de um revisor que era pesquisador da área de *code smells*, aqui representado como um especialista da área;
- O fato de não se mostrar estratégico para os resultados alcançados, considerando a amplitude proposta pelas questões de pesquisa, adotar qualquer medida que restringisse as percepções dos pesquisadores envolvidos.

Assim, os temas e subtemas foram surgindo desde a primeira etapa do trabalho, sendo complementados nas etapas posteriores. Assim, a abordagem dedutiva e, por conseguinte, a integrada foi descartada no início da etapa de codificação.

Como a abordagem adotada foi a indutiva, a lista de códigos iniciais com definições e frequências não foi discutida. Como resultado de não atender a esta recomendação fornecida pelos autores, não foi identificado nenhum tipo de prejuízo ao método. Esta questão pode representar uma necessidade futura de estudar e entender a fundo a obrigatoriedade desta lista inicial quando a abordagem adotada é a indutiva.

Com a maturidade adquirida ao longo do método, percebemos que a abordagem integrada era uma opção interessante de ser adotada para este trabalho. A mesma poderia conciliar a preocupação de não ser restritiva, com o aproveitamento do conhecimento do especialista no tema *code smells,* como participante de todo o método.

12. Manter a rastreabilidade na codificação.

Foi adotada uma rastreabilidade simples para a etapa de codificação. Um número sequencial foi atribuído a cada resumo gerado a partir das extrações. Quando da identificação dos códigos, estes também receberam numerações sequenciais. As numerações relacionadas com os resumos que auxiliaram na identificação dos códigos foram registradas entre parênteses. Esta rastreabilidade mostra-se importante para aclarar o processo de codificação e criação do mapa temático, melhorando a confiabilidade da síntese temática.

Adotamos o padrão apresentado no Quadro 2 para garantir a rastreabilidade e melhorar a confiabilidade da síntese gerada. Este padrão permite que seja possível identificar a etapa em que o artigo foi trabalhado, qual o ID do StArt vinculado ao mesmo, qual o código vinculado ao resultado, possibilitando identificar o tema e subtema no Mapa Temático.

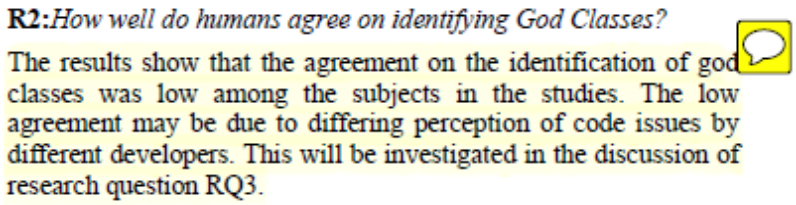
ETAPA.ID.CÓDIGO

Quadro 2 Rastreabilidade mantida para criação do Modelo de Alto Nível

Onde:

- ETAPA: é a etapa na qual o artigo foi trabalhado na extração/codificação. Para este trabalho foram adotadas 05 etapas, conforme mencionado na Tabela 15;
- ID: é o identificador do artigo gerado pela ferramenta StArt;
- CÓDIGO: é o número sequencial da codificação realizada para um determinado resultado.

Para elucidar melhor a utilização deste padrão de rastreabilidade, apresentamos um exemplo em que a rastreabilidade completa pode ser acompanhada. O artigo escolhido para retratar esta rastreabilidade foi o de Schumacher (2010), vinculado ao ID 26 da ferramenta StArt. No corpo do artigo foram feitas marcações nos segmentos de texto que identificavam um possível resultado, considerando as questões de pesquisa/hipóteses definidas para o artigo em questão. Na Figura 24 é apresentada a marcação da questão de pesquisa de número R2 para o artigo em questão, extraída da Seção "*Discussion*".


R2:*How well do humans agree on identifying God Classes?*

The results show that the agreement on the identification of god classes was low among the subjects in the studies. The low agreement may be due to differing perception of code issues by different developers. This will be investigated in the discussion of research question RQ3.

Figura 24 Marcação da Questão de Pesquisa no artigo de Schumacher (2010)

A cada marcação feita no corpo do artigo foi atribuído um comentário com o título correspondente à marcação feita. Neste exemplo, o balão amarelo apresenta um comentário que demonstra tratar-se de um resultado do artigo.

Na Figura 25 apresentamos a Planilha de Extração/Mapeamento contendo a transcrição desta marcação no local indicado pelo retângulo vermelho. Este exemplo remete à aba Resultados da Planilha, onde cada resultado encontrado nos artigos era transcrito para a devida Seção em que foi encontrado nos artigos. Existia a preocupação em manter a coerência e rastreabilidade para garantir a confiabilidade do processo manual.

ID	26
Title	Building Empirical Support for Automated Code Smell Detection
Findings/ Outcomes	The results show that, even though the subjects perceive detecting god classes as an easy task, the agreement for the classification is low. Misplaced methods are a strong driver for letting subjects identify god classes as such. Earlier proposed metric-based detection approaches performed well compared to the human classification. These results lead to the conclusion that an automated metric-based pre-selection decreases the effort spent on manual code inspections. Automatic detection accompanied by a manual review increases
	Introduction: Não foram encontrados.
	Results: 1 - All subjects were able to identify classes they felt were god classes. 2 - Due to the apparent low agreement between the subjects, it is important to identify the reasoning behind the subjects' classification of god classes. 3 - In addition, the agreement on the identified issues between the subjects for the two studies was calculated using Cohen's Kappa. Kappa for project A is 37% and 39% for project B. This indicates "fair agreement" between the subjects [5]. The agreement for individual issues was calculated as well. For misplaced methods, the agreement is 46% for project A and 47% for project B. This results in ``moderate agreement'' between the subjects for the misplaced method issue. 4 - Therefore, the hypothesis that the likelihood of a change of a line of code in a god class is significantly higher than in a non-god class has to be rejected.
	Discussion: 1 - The results show that the subjects in the two studies did not find the task of identifying god classes to be challenging. We believe that this is in large part due to the way subjects were introduced to god classes.
	2 - The results show that the agreement on the identification of god classes was low among the subjects in the studies. The low agreement may be due to differing perception of code issues by different developers. This will be investigated in the discussion of research question RQ3.

Figura 25 Transcrição do segmento de texto do artigo de Schumacher (2010)

Durante a fase de codificação, esta transcrição foi analisada e trabalhada, respeitando a ideia original dos artigos. Numerações consistentes foram fornecidas para manter esta rastreabilidade. Os números que aparecem entre parênteses na coluna "Código" são os que referenciam os números da coluna "Resumo" que originaram os códigos. Como podemos observar na Figura 26, o código 2, do artigo de Schumacher (2010), trabalhado na etapa 1 de extração/codificação foi gerado a partir dos resumos de números 2, 6, 10 e 19 feitos para as transcrições realizadas.

RESUMO	CÓDIGO
1 - The subjects perceive detecting god classes as an easy task. 2 - The agreement on GC classification is low. 3 - Automated metric-based pre-selection decreases the effort spent on manual code inspections. 4 - Automatic detection accompanied by a manual review increases the overall confidence in the results of metric-based classifiers. 5 - All subjects were able to identify classes they felt were god classes.	1 - The subjects perceive detecting god classes as an easy task. (1, 5, 9) 2 - The agreement on GC classification and detection is low. (2, 6, 10, 19) 3 - Automatic detection accompanied by a manual review increases the overall confidence in the results of metric-based classifiers. (4) 4 - God class don't require a higher maintainability effort than non-code smells. (17 , 21)

Figura 26 Rastreabilidade na codificação do artigo de Schumacher (2010)

Esta rastreabilidade é considerada na criação do Mapa Temático, onde o ID deste código utilizado para compor o mapa dentro do tema. Na Figura 27, apresentamos um exemplo para o tema "*Human Aspects*" e subtema "*Agreement on detection*". Na Figura 27, o ID 1.26.2 representa que o artigo pertencia à etapa 1 de extração/codificação, que o ID do StArt do artigo era o 26, e que o código vinculado ao resultado em questão era o de número 2 na Planilha de Extração/Mapeamento.

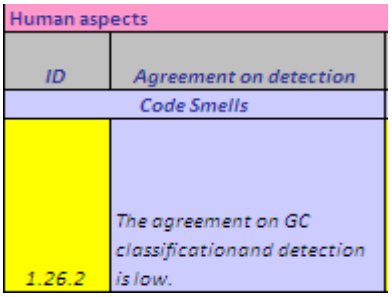


Figura 27 Rastreabilidade no Mapa Temático para o artigo de Schumacher (2010)

A organização dos dados desde a etapa de extração e codificação favoreceu a sua utilização nos passos seguintes. Principalmente na criação do Mapa Temático, pois as referências seguiam o padrão mencionado acima. Com isto, a verificação do especialista também ficava mais simples de ser realizada, favorecendo a confiabilidade da síntese.

13. Codificar os conjuntos de dados, realizando comparações constantes, considerando o conhecimento dos pesquisadores e as questões de pesquisa.

Nesta questão será apresentada a codificação de cada uma das duas dimensões em que a codificação se faz necessária para atender às necessidades da síntese temática. A codificação foi realizada considerando o bloco de texto extraído no passo anterior para a Planilha de Extração/Mapeamento e foi realizada considerando as questões de pesquisa explicitadas para a RS.

Objetivo

Para Wohlin *et al.* (2000), a "Definição do Experimento", ou seja, a descrição do objetivo ao qual a pesquisa encontra-se pautada, deve ser o mais coerente possível. Para este trabalho foram coletadas na Planilha de Extração/Mapeamento as informações relacionadas aos objetivos dos artigos. Dada a grande quantidade de informações e considerando a

necessidade de realizar análises futuras com estes dados, adotamos uma técnica que favorecesse a fiel codificação do objetivo, uniformizando o seu entendimento e facilitando a utilização do mesmo durante as análises necessárias para a condução da síntese temática. Desta forma, para a dimensão Objetivo foi utilizada a técnica GQM (*Goal, Question, Metrics*) (Van Solingen, 2002). Esta técnica inclui alguns elementos a serem preenchidos, conforme mencionado na Seção 2.3 desta dissertação.

Considerando as necessidades específicas do trabalho em questão, adaptamos o modelo GQM para a codificação dos objetivos dos artigos selecionados. Para atender à nossa necessidade retiramos as seguintes questões: "*from the point of view* of the" e "*in the context of*". Acrescentamos, no entanto, o item "*using*" para deixar claro se o artigo usava detecção automática ou não.

O Quadro 3 apresenta os elementos utilizados para a codificação dos objetivos dos artigos.

Analyze <...> for the purpose of <...> with respect to their <...> using <...>

Quadro 3 Elementos utilizado do GQM adaptado de (Thomas e Harden, 2008)

Onde:

- *Analyze*: apresenta o foco principal do artigo;
- *For the purpose of*: referencia o propósito que embasa o "*analyze*";
- *With respect to their*: explicita os principais objetos de estudo do artigo;
- *Using*: aclara a forma de instrumentação utilizada para cada arquivo.

A justificativa para a adaptação acima mencionada deve-se ao fato de que as informações de contexto utilizadas na síntese eram bastante amplas. Além disso, a intenção inicial era somente entender o que os artigos faziam, a fim de realizar melhor a filtragem, amadurecendo a visão a respeito dos possíveis temas que seriam trabalhados.

Todos os objetivos dos artigos foram codificados.

Na Figura 28, apresentamos um exemplo dos dados extraídos para compreender o objetivo do artigo e sua respectiva codificação utilizando o método GQM adaptado. As marcações em vermelho apresentadas no lado esquerdo da Figura representam partes do texto que foram extraídas para gerar o GQM utilizado na síntese.

			GQM
ID		38	
Title		Exploring the Impact of Inter-smell Relations on Software Maintainability: An Empirical	
Aim		We empirically investigate the interactions amongst 12 code smells and analyze how those interactions relate to maintenance problems. The aim of the study was to explore situations where maintenance problems occurred due to the interaction of several code smells.	
Research Questions/ Hypothesis		Questões de Pesquisa não explicitadas no texto. Based on these observations, we conjectured that interaction effects between co-located smells (i.e., smells located in the same artifact) can intensify problems caused by individual code smells or lead to additional, unforeseen maintenance issues.	Analyze (maintenance activity) for The purpose (correlation) with Respect to their (maintenance problems and inter-smells) Using (automatic detection)
GQM	Object	Twelve code smells were automatically identified in the original "pre-maintenance" versions of the systems, using Borland Together and InCode. Identification of Problems and Problematic - Therefore, we needed to identify the artifacts that caused problems during maintenance and to record the nature of the problems caused by these artifacts. Therefore, we needed to identify the artifacts that caused problems during maintenance and to record the nature of the problems caused by these artifacts. Analysis Technique - This analysis is based on the explanation building technique [37] and aims to determine the extent to which the presence of a single code smell or several code smells contributed to the problems experienced by the developers during maintenance (análise qualitativa).	
	Purpose	Correlation	
	With respect to their	1 - inter-smell 2 - maintenance problems and artifacts	
Observation		It is based on an industrial case study in which six professional software engineers were hired to maintain four medium-sized Java systems with equivalent functionality but dissimilar designs for a period of up to four weeks. Professional developers were hired for a period of four weeks to implement change requests on four medium-sized Java csystems with known smells.	

Figura 28 Extração e codificação do Objetivo do artigo de Yamashita e Moonen (2013b)

Na Figura 29 apresentamos o mesmo exemplo da codificação do objetivo pata o artigo de Yamashita e Moonen (2013b), de forma a permitir uma melhor visibilidade do padrão adotado.

Analyze (maintenance activity) for
The purpose of (correlation) with
Respect to their (maintenance problems and code smells)
Using (automatic detection)

Figura 29 Exemplo do método GQM – codificação da dimensão "Objetivo"

Para a maioria dos artigos selecionados não foi necessário retornar ao texto original para realizar a codificação, sendo os itens extraídos suficientes para a atividade. Apenas para os casos em que havia discordância com o revisor, o texto original precisou ser revisitado com o intuito de sanar as dúvidas e buscar o entendimento correto do contexto geral para facilitar a concordância.

Resultado

Para a codificação dos resultados foi adotada a técnica de codificação descritiva proposta por Saldanã (2008). Como já mencionado na questão 10 desta dissertação, a justificativa para adoção desta técnica foi a sua aplicabilidade simples e indicação para iniciantes.

Antes do início da codificação dos resultados, avaliamos a relação entre os resultados encontrados em cada artigo e seus respectivos mapeamentos GQM. A intenção foi identificar se existiam lacunas entre estas informações. Com esta validação foi possível verificar se algum resultado relevante havia sido desconsiderado. Com o uso do GQM foi possível identificar se os objetivos definidos nos artigos estavam contemplados na Planilha de Extração/Mapeamento. Nos casos em que percebemos inconsistência quanto dos resultados extraídos com relação aos objetivos, relemos os artigos para buscar os resultados ausentes no processo de extração.

Como já foi mencionado anteriormente, para esta etapa relacionada aos resultados, foram inicialmente extraídas para a Planilha de Extração/Mapeamento todas as partes do artigo que apresentavam os resultados encontrados pelos artigos. Para o preenchimento da coluna "Resumo" estes segmentos de texto eram trabalhados com o intuito de definir uma frase que os representasse. Ainda não nos preocupávamos com códigos sobrepostos neste momento, focando apenas no resumo dos segmentos de texto extraídos.

A Figura 30 apresenta os segmentos de texto extraídos para os resultados encontrados no artigo (Olbrich *et al.*, 2009). As marcações nos textos extraídos, separando o que era resultado (em azul) do que era conjectura dos autores (em laranja), auxiliou na identificação do que era relevante compor a coluna "Resumo".

ID	9	RESUMO
Title	The Evolution and Impact of Code Smells: A Case Study of Two Open Source Systems	1 - There is a correlation between size and number of god class. 2 - There is a correlation between size and number of shotgun surgery. 3 - Changes of infected classes with GC is significant more often. 4 - Changes of infected components with SS is significant more often. 5 - The code churn of infected classes with GC is significant larger. 6 - There are different phases in the evolution of code smells during the system development. 7 - The classes infected with code smells have a higher change frequency. 8 - Classes infected seem to need more maintenance than non-infected classes.
Findings/ Outcomes	**Abstract:** The results show that we can identify different phases in the evolution of code smells during the system development and that code smell infected components exhibit a different change behavior. **Introduction:** Não foram encontrados. **Results of Research Question I** **GOD CLASSES** 1 - Consider our initial hypotheses about God Classes: H1 (The total number of code smells increases steadily) and H2 (The relative number of components having code smells increases over time). Since we can identify revisions that show a reduction in the absolute number as well as in the relative number of detected smells, we should reject both hypotheses. However, we can identify a large correlation between system size and the number of God Class smells. This also leads to the assumption that the bigger the size of a system the more God Classes it contains. **SHOTGUN SURGERY** 1 - Since the observed evolution of the Shotgun Surgery code smell features intervals with a reduction of the absolute as well as the relative number of infected classes we can reject H1 and H2 for both projects. Still, we can see the same large correlation as for God Class smells, concerning the system size and the number of Shotgun Surgery smells. **Results of Research Question II** **Entity change behavior** 1 - The results of this section address H3 (The change proneness of components with smells is higher than the ones without). Based on the applied two-sample t-test on the experimental data, we can see that classes which are infected with a God Class code smell get changed significantly more often in both projects. Therefore we can accept H3. 2 - Similar to the God Class smell, classes which are infected with Shotgun Surgery show a significantly higher change frequency compared to non-infected classes for both projects. Based on the t-test result we can accept H3 for Shotgun Surgery code smells. **Entity churn comparison** 1 - This section contains the results regarding H4 (The size of changes of infected classes is significant larger than the size of changes on non-infected classes). The two sample ttests on the experimental data shows that the size of changes is significantly larger for classes which **Conclusion:** 1 - With regard to change behavior, in this study the classes infected with code smells have a higher change frequency; such classes seem to need more maintenance than non-infected classes; furthermore God Classes feature bigger changes Thus, the probability is higher that maintenance tasks take more effort since more modifications to the code (i.e., modifying, adding or deleting) is required. 2 - Initially we made the assumption that classes infected with shotgun surgery would have a lower change frequency since people would be hesitant to touch them because this implies	

Figura 30 Passo intermediário da codificação para o artigo de Olbrich *et al.* (2009)

Depois de fazer os resumos, eram gerados os códigos propriamente ditos. Neste momento nos preocupávamos em não gerar códigos sobrepostos. Por este motivo, um determinado código pode estar associado a mais de um item da coluna "Resumo", conforme apresentado na Figura 31.

RESUMO	CÓDIGO
1 - There is a correlation between size and number of god class. 2 - There is a correlation between size and number of shotgun surgery. 3 - Changes of infected classes with GC is significant more often. 4 - Changes of infected components with SS is significant more often. 5 - The code churn of infected classes with GC is significant larger. 6 - There are different phases in the evolution of code smells during the system development. 7 - The classes infected with code smells have a higher change frequency. 8 - Classes infected seem to need more maintenance than non-infected classes.	1-There is a correlation between size and number of: god class and shotgun surgery. (1, 2) 2 - The change proneness of components with smells is higher than the ones without.. (3,4,7) 3 - The code churn of infected classes with god class is significant larger. (5) 4 - Code smells evolve during the system development .(6) 5 - Classes infected seem to need more maintenance than non-infected classes. (8)

Figura 31 Codificação para o artigo de Olbrich *et al.* (2009)

Pode-se observar na Figura 31 que, para cada item da coluna “Resumo”, foi atribuído um número sequencial que referencia o resultado. Este registro contribui para a rastreabilidade entre os resumos e os códigos. Cada resultado também possui um número sequencial atribuído, referenciado ao final e entre parênteses. Estes números correspondem à numeração atribuída aos itens da coluna “Resumo” que o originou.

Nesta etapa de codificação não havia preocupação com os temas propriamente ditos, entendendo que esses seriam naturalmente percebidos no passo seguinte de tradução dos códigos em temas.

Houve preocupação em trabalhar todos os textos que apresentavam a mesma ideia, extraídos muitas vezes de seções distintas, para que existisse apenas um código que os representasse. A intenção foi evitar a duplicidade de informações, gerando um método mais sistemático.

Diante do exposto, podemos ver que a codificação seguiu livre, mas havia uma preocupação desde a execução deste passo de criar os códigos de maneira sistemática e organizada, evitando códigos sobrepostos.

Cruzes e Dyba (2011a) apresentam no método proposto esta codificação mais livre e sem preocupações em organizá-los e torná-los distintos quanto aos seus significados. Foi percebida esta diferença entre a teoria e a prática, talvez pela inexperiência dos pesquisadores envolvidos com as técnicas de codificação, receando lacunas futuras e/ou retrabalhos. Nossa forma de condução nos auxiliou a criar uma estratégia estruturada e sistemática de atuação, contribuindo para que chegássemos a resultados de uma forma em que todos se sentissem mais seguros.

Uma dificuldade considerada pelos autores e ratificada pela execução prática do método foi conseguir chegar a códigos dos textos extraídos sem deixar que a ideia original do autor se perdesse. Ou seja, foi necessário trabalhar na codificação sem julgamentos pessoais ou tendências, focando na escrita de frases com conteúdos atômicos e códigos distintos entre si. A forma encontrada para trabalhar melhor esta questão foi a realização de verificações constantes com o especialista em *code smell* e discussões sempre que as dúvidas permaneciam ou que se fazia necessários chegar a um consenso sobre o assunto.

Por vezes, o resumo apresentava esta atomicidade mencionada e era aplicado como código para os resultados em questão. Outras vezes era necessário entender o resumo, seu contexto, seu objetivo principal e depois chegar a uma codificação que o melhor representasse.

14. Verificar a codificação.

A verificação das etapas está diretamente relacionada à confiabilidade da síntese e foi adquirida da seguinte forma:

- Durante a execução houve verificação realizada pelo especialista em relação ao que era codificado;
- Sugestões e comentários eram registrados pelo verificador com o intuito de prover a concordância;
- Para os casos de discordância, uma reunião era realizada para discussão e devidos esclarecimentos;
- Para os casos em que não se chegasse a um consenso, mesmo após a reunião, um terceiro pesquisador independente era consultado.

Os critérios de verificação para este passo foram relacionados à necessidade de garantir:

- Consistência entre a linguagem e os conceitos utilizados;
- Ligações claras e evidentes entre os segmentos de texto extraído dos artigos e os códigos que foram atribuídos;
- Relevância dos resultados para o tema central *code smells* e para o foco que estava sendo dado ao estudo;
- Rastreabilidade entre as extrações e os códigos gerados.

Ao final da etapa de codificação, as informações coletadas foram cadastradas por um pesquisador na ferramenta desenvolvida, para posterior condução das análises. As informações relativas à publicação e contexto dos artigos selecionados também foram cadastradas na ferramenta. Houve verificação de todos os cadastros, procurando evitar inconsistências.

5.2.4 Tradução de códigos em temas e criação do modelo de alto nível

15. Analisar se foi adquirida uma visão geral, inclusiva e compreensiva de todos os artigos.

A divisão dos artigos em etapas iterativas de extração e codificação ajudou na aquisição de uma visão geral, inclusiva e compreensiva de todos os artigos. Esta visão apresenta-se importante para a tradução dos códigos em temas. O especialista teve participação relevante na aquisição da visão geral, pois foi através de seus conhecimentos explicitados que podemos nivelar os entendimentos dos pesquisadores envolvidos em relação aos principais pontos abordados pelos artigos selecionados.

16. Traduzir códigos em Temas.

Para um melhor aproveitamento desta etapa do trabalho, foi adotado o conceito de tema mencionado por Cruzes e Dyba (2011a). Os autores definem o tema como uma entidade abstrata que captura e unifica a natureza básica da experiência contida em um todo. Simplificando este entendimento para a aplicabilidade prática neste trabalho, pode-se dizer que um tema é a representação de um conjunto de códigos que apresentam alguma interligação na essência das suas informações.

Para o passo de criar o modelo de mais alto nível, Cruzes e Dyba (2011a) além de apresentar recomendações gerais, indicam de forma detalhada como executá-lo. Desta forma, nossa contribuição neste passo foi mínima, mas o refinamento proposto nesta dissertação reflete de forma completa a discussão proposta pelos autores.

Na prática, o passo de tradução do código em temas, apesar de ser um passo anterior à criação do modelo de alto nível propriamente dito, foi realizado em paralelo. Basicamente optamos por gerar um artefato em Excel com o nome Mapa Temático e nele atuamos durante os passos de Tradução do Código em Temas e Criação do Modelo de Alto Nível. Neste caso, o modelo de alto nível criado foi o próprio Mapa Temático.

Para preenchimento inicial do Mapa Temático adotamos a seguinte estratégia:

- Cada código existente na Planilha de Extração/Mapeamento era adicionado ao artefato. Para tanto, a similaridade entre estes códigos era considerada para que definíssemos a coluna em que cada um seria colocado. A intenção era que códigos com significados próximos ficassem na mesma coluna.
- Fizemos isso com todos os códigos de um mesmo artigo até que fosse possível olhar para as colunas, analisar os códigos existentes e identificar um rótulo que melhor os representasse. Interessante que sobre um mesmo aspecto existiam concordâncias e discordâncias entre os artigos. Este assunto foi capturado na escrita da síntese temática.

- Para os artigos seguintes, além de acrescentar os códigos relacionados às colunas já existentes, criávamos novas colunas caso as existentes não os representassem. Mais uma vez analisávamos esta nova coluna identificada e atribuíamos um rótulo. Quando acabávamos de acrescentar um artigo ao Mapa Temático inicial, sempre revisávamos o rótulo atribuído na busca de um termo que pudesse abrangê-los;
- Estes passos se repetiram até que todos os artigos tivessem sido considerados. Neste momento, códigos sobrepostos que não haviam sido tratados em tempo de codificação foram reduzidos, fazendo-nos ajustar o método para manter a coerência e rastreabilidade.

Desta forma, o Mapa Temático foi preenchido através da análise e organização dos códigos de todos os artigos e suas respectivas nomenclaturas (considerando o padrão referido acima). A rastreabilidade foi uma preocupação constante, tendo sido explicitado um exemplo completo da adoção da mesma na questão 15 desta Seção.

Cruzes e Dyba (2011a) mencionam que os códigos sobrepostos sejam reduzidos e que, ao final deste processo, os temas sejam definidos. Para a forma como este método foi conduzido, os códigos já tinham sido reduzidos desde o momento da codificação propriamente dita. Os resumos trabalhados como método intermediário para a codificação, já haviam passado por este filtro, não sendo representativa a quantidade de códigos sobrepostos que foram aqui trabalhados.

Depois de definidos os temas, uma análise foi realizada através de comparações entre os mesmos. O objetivo foi identificar se os temas poderiam ser agrupados, ou se poderiam virar subtemas dentro de um tema maior e mais abrangente que os representasse. Foi importante essa visão geral, inclusiva e compreensiva dos códigos de todos os artigos para que não houvesse retrabalho e para sistematizar os assuntos na mente antes de começar a pensar nos temas. Este passo é mencionado por Cruzes e Dyba (2011a) como algo importante e necessário.

Outra questão importante de ser mencionada diz respeito à verificação dos temas finais encontrados com as extrações realizadas. Apesar de Cruzes e Dyba (2011a) afirmarem que os temas devem ser verificados com os dados dos artigos originais, não realizamos este trabalho em função das extrações realizadas. Uma justificativa para a adoção desta adaptação deve-se ao fato de que as extrações são transcrições fiéis aos artigos originais. Acreditamos

que cumprimos com o objetivo do método de forma eficiente, poupando tempo para realização da síntese.

A Figura 32 expõe de forma gráfica e iterativa os níveis de interpretação adotados para este trabalho de síntese temática, considerando a abordagem incremental e iterativa do método adotado.

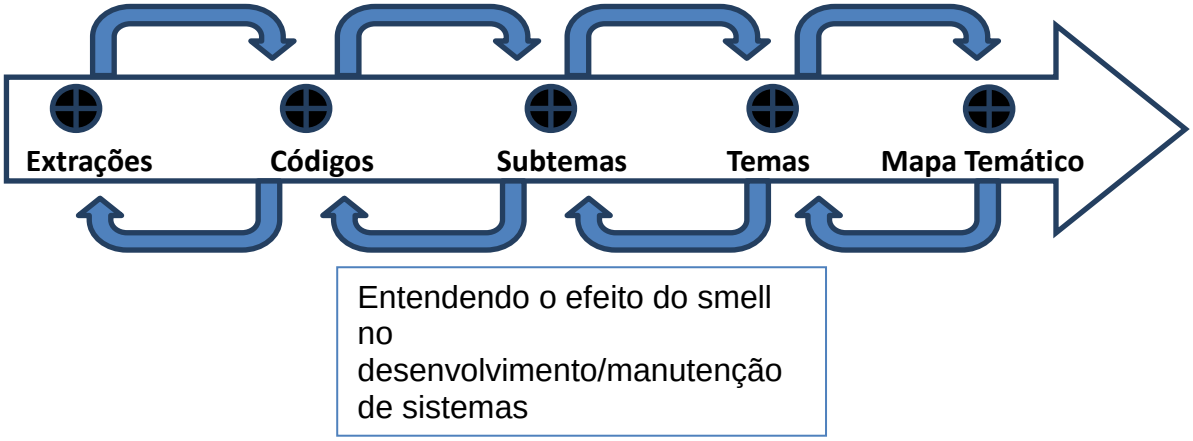


Figura 32 Níveis de interpretação da síntese temática adaptado de (Cruzes e Dyba, 2011a)

Durante todo o método, foi mantida uma coluna no Mapa Temático de dúvida, contendo os códigos para os quais não era possível mapear, de imediato, os temas. Para estes casos, uma reunião foi realizada com o intuito de levantar as discussões e análises e possibilitar os devidos ajustes no mapa temático deixando-o completo e coeso. A participação do especialista no tema foi importante para conseguir realizar de forma assertiva a inclusão dos códigos em que existiam dúvidas.

São exemplos de códigos que não foram encaixados logo no início nos temas identificados:

1. 1.26.1 - *The subjects perceive detecting god classes as an easy task. (Detection difficulty)*
2. 1.26.8 - *Lack of cohesion and complexity are identified as issues that let the human subjects identify a class as a god class. (Decision drivers)*
3. 3.60.5 - *There isn't a linear relationship between class size and change likelihood that justifies LOC-normalization. (Smell Density)*
4. 4.1509.3 - *Simple code smells and source code metrics have a relationship. (Metrics versus smell)*

Entre parênteses, apresentamos o respectivo código ao qual o mesmo foi encaixado após a realização da reunião com o especialista no tema.

OBS: A questão 17 (Verificar a tradução dos códigos em temas) será apresentada em conjunto com a questão 22 (Verificar a criação do modelo de alto nível). Esta forma de apresentação deu-se pelo fato dos passos de Tradução dos Códigos em Temas e Criação do Modelo de Alto Nível terem sido executados quase que simultaneamente. Os motivos estão explicitados na questão 22.

18. Analisar, explorar e comparar as relações existentes entre os temas.

Na medida em que novos temas eram percebidos para a composição do Mapa Temático, iniciava-se uma análise nos seus códigos encontrados até o momento. O objetivo era tentar identificar se existiam relações entre os temas já identificados. Marcações foram feitas para viabilizar a descrição de forma clara das relações identificadas entre os temas. O mapeamento destas relações pode ser identificado na questão 21 desta Seção.

19. Comparar as relações existentes entre os temas com as conclusões dos artigos.

Durante o processo de análise para criação do Mapa Temático, é importante checar se os temas identificados para a síntese temática apresentam consistência em relação às conclusões apresentadas pelos artigos selecionados. Como as conclusões dos artigos refletem a maioria dos resultados encontrados pelos mesmos, voltamos às conclusões e fizemos esta checagem. Não foram identificadas discrepâncias nesta comparação que justificasse a alteração de algum tema identificado para melhor representar as conclusões dos artigos. Esta questão também fornece subsídio para que o modelo mencionado na questão seguinte possa ser representado.

20. Criar um modelo para representar os temas e as relações existentes entre os temas.

A estratégia adotada para preenchimento inicial do Mapa Temático, apresentada na questão 16 (Traduzir os códigos em temas) desta Seção, repetiu-se até a completa finalização dos códigos de todos os artigos selecionados, chegando ao resultado representado na Figura 33. O Mapa Temático completo gerado para esta etapa encontra-se no Apêndice A desta dissertação. Na Figura 33, apresentamos uma visão simplificada do assunto central trabalhado com os seus principais temas e subtemas encontrados.

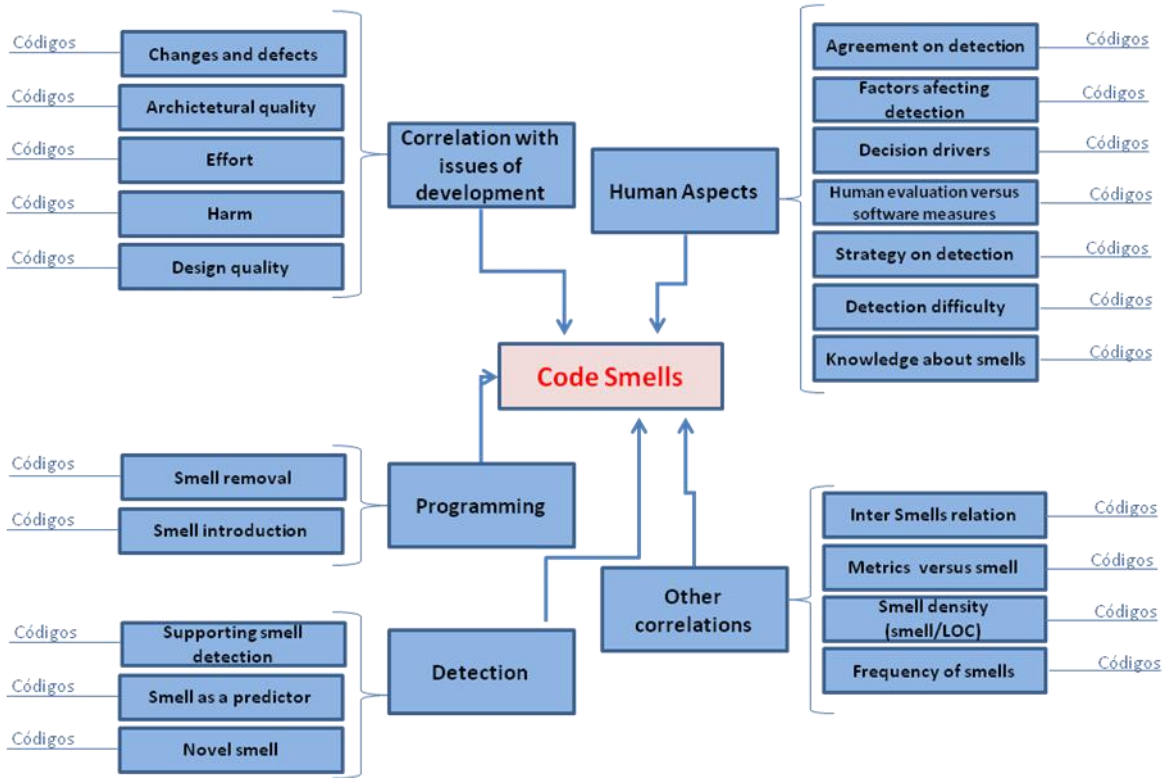


Figura 33 Visão simplificada do Mapa Temático

Pode-se perceber na Figura a existência de cinco temas ligados ao tema central *code smells*. Cada um destes temas apresenta subtemas vinculados. As análises e comparações constantes, a partir dos códigos mapeados de cada artigo, foram de fundamental importância para atingir este resultado do Mapa Temático e das visões de relação entre os temas trabalhados.

Com o resultado obtido após a criação do mapa temático para o tema central da RS realizada, os temas e as relações entre os mesmos foram comparados e verificados com as conclusões apresentadas pelos estudos primários. Esta verificação se deu através da participação de um pesquisador, checando discrepância em relação aos achados. Nenhuma discrepância foi detectada.

Analisando um pouco os temas mais abrangentes, pode-se perceber que:

- *Programming*: este tema está relacionado com o momento em que os *smells* são introduzidos no código e são removidos;
- *Human Aspects*: este tema trata basicamente das questões humanas que influenciam a detecção de *smells*, englobando todas as dificuldades, conhecimentos, a relação com as métricas, os drivers e principalmente o acordo na detecção dos *smells*;

- *Corretation with issues of development*: este tema aborda a correlação existente entre *smells* e as questões relacionadas ao desenvolvimento (o esforço mapeado como subtema neste tema exibe claramente questões associadas a esforço de manutenção ou inspeção de sistemas e não a esforço de detecção de *smells*. Caso fosse esforço de detecção, este subtema seria mais bem representado pelo tema Detection, apresentado a seguir);
- *Detection*: este tema aborda a necessidade de suporte para detecção de *smells*, o fato do *smell* ser considerado ou não como um indicador de problema no código e, para um caso particular, até a detecção de um novo tipo de smell não apresentado na literatura existente;
- *Other correlations*: tema que aborda as correlações entre *smells* e outras questões consideradas relevantes para os autores dos artigos selecionados, mas não consideradas como o foco do trabalho que foi realizado para a RS em questão, pelo fato da mesma considerar nas suas questões de pesquisa, exclusivamente os assuntos que estão relacionados ao efeito dos *smells* no desenvolvimento/manutenção de sistemas. Portanto, estes não fizeram parte das análises apresentadas neste trabalho.

21. Descrever de forma clara as relações existentes entre os temas.

A correlação entre os temas encontrados e relacionados ao tema central da RS foi analisada através dos códigos existentes no mapa temático, sendo apresentada na Figura 34.

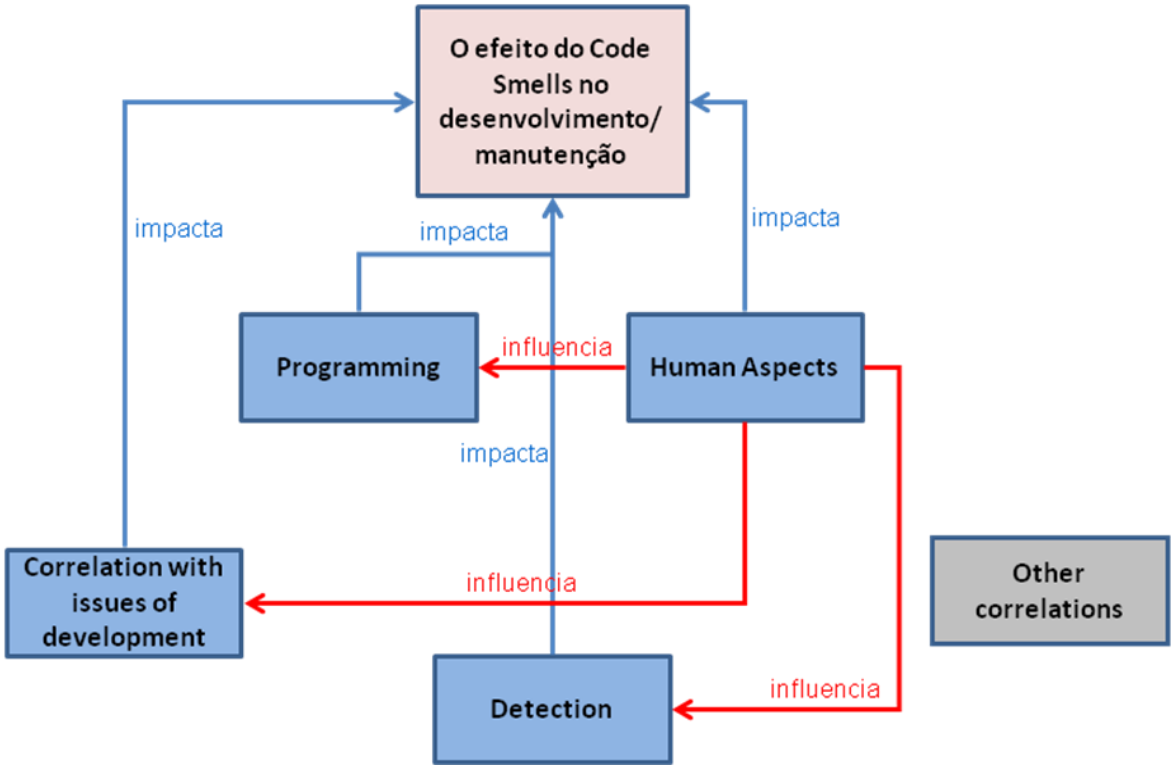


Figura 34 Relação entre os temas encontrados

O tema *Other Correlations*, como já mencionado anteriormente, não fez parte das análises para a RS e também não fará parte da síntese temática para o tema *Code Smells* e nem da análise da relação entre os temas, considerando o seu foco no efeito dos mesmos no desenvolvimento/manutenção de sistemas. Os poucos códigos que estavam neste tema e que apresentam relação com o foco acima mencionado, foram classificados também entre os subtemas de outros temas que melhor o representassem. Como exemplo pode ser citado o código 1.63.6 (*Inconsistent design can be identified by detecting a combination of code smells*) que aparece no subtema *Inter Smells relation* do Tema *Other correlations* e foi reclassificado para o subtema *Design quality*, no tema *Correlation with issues of development* para que pudesse ser trabalhado.

Através da análise da Figura 34, pode-se perceber que existe uma relação entre os temas identificados e o foco apresentado nas questões de pesquisa para *code smells*. Este foco, como já mencionado nesta dissertação no Capítulo 1, permeia na análise do efeito do *smell* no desenvolvimento/manutenção de sistemas.

Através da análise da Figura ainda podemos concluir que os aspectos humanos tratados pelos artigos selecionados influenciam os demais temas encontrados.

Na Figura 35 ilustramos algumas relações encontradas, que comprovam que os aspectos humanos influenciam cada um dos temas mencionados.

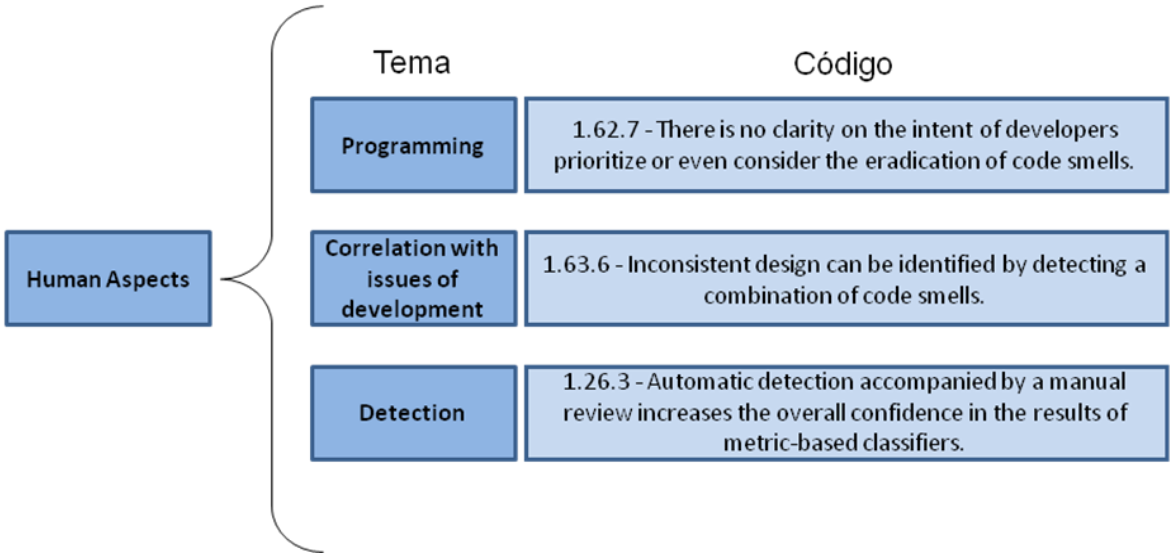


Figura 35 Representação da influência dos aspectos humanos nos demais temas

Para o código 1.63.6 a relação não aparece tão claramente especificada, mas sabe-se que as inconsistências de projeto são geradas pelos analistas, ou seja, pessoas envolvidas no projeto, dado o seu entendimento e até mesmo os seus conhecimentos na área. Já para o código 1.26.3, a necessidade de complementar detecção automática com manual é considerada e isso remete ao fato de ter uma pessoa realizando a detecção manual, que por sua vez, implica no entendimento da pessoa, seus conhecimentos na área e etc.

17 e 22. Verificar a tradução dos códigos em temas e Verificar a criação do modelo de alto nível.

A verificação das etapas está diretamente relacionada à confiabilidade da síntese e foi adquirida da seguinte forma:

- Durante a execução houve verificação realizada pelo especialista em relação aos temas identificados e ao modelo de alto nível gerado;
- Sugestões e comentários eram registrados pelo verificador com o intuito de prover a concordância;
- Para os casos de discordância, uma reunião era realizada para discussão e devidos esclarecimentos;

- Para os casos em que não se chegasse a um consenso, mesmo após a reunião, um terceiro pesquisador independente era consultado.

Como os passos de Tradução de Códigos em Temas e Criação do Modelo de Alto Nível foram passos que ocorreram em paralelo e de forma incremental, a verificação foi realizada de forma conjunta.

A verificação checava dentre outras coisas:

- A aplicabilidade prática das delimitações feitas nos temas e subtemas, evitando a redundância e falta de clareza. Ajustes significavam um melhor refinamento dos subtemas/temas, considerando a relevância dos mesmos para o tópico central: *code smells*;
- A coerência dos temas encontrados, consistência e distinção entre os mesmos;
- Se o modelo de alto nível gerado reflete as relações encontradas durante a realização do método.

5.2.5 Avaliação da Confiabilidade da Síntese

23. Realizar a avaliação da confiabilidade da síntese.

Todas as etapas intermediárias de verificação contribuíram para o aumento do nível de confiabilidade da síntese temática. A linguagem e os conceitos utilizados na síntese foram avaliados durante as verificações.

A participação de um especialista no tema nas verificações foi considerada um quesito importante para melhorar a confiabilidade da síntese temática. Desde o início do método, o mesmo atuou retirando as dúvidas e revisando os passos realizados. Isto contribuiu para o nivelamento geral dos conhecimentos à respeito do tema e para aumentar a confiança dos envolvidos em relação aos resultados que estavam sendo encontrados.

As conjecturas e definições adotadas foram explicitadas ao longo deste Capítulo, que apresentou toda a execução do método. Um exemplo claro citado anteriormente foi o artigo de Yamashita e Moonen (2013b) que não apresentava explicitamente suas hipóteses/questões de pesquisa, que para este trabalho apresentavam-se como itens importantes para subsidiar a definição da codificação para o método GQM e para nortear a

relevância dos resultados apresentados nos artigos. Para este caso, a reunião foi realizada e os envolvidos chegaram a um consenso do que deveria ser considerado para este item.

Foi criado um *checklist* contendo os principais conceitos e questões apresentados por Cruzes e Dyba (2011a) para sistematizar a avaliação da confiabilidade da síntese.

O objetivo foi garantir que os principais conceitos e suas questões tenham sido considerados ao longo da execução do método, diminuindo a subjetividade desta avaliação.

Este *checklist* foi gerado para aplicação em contextos similares ao trabalhado no grupo SoftVis. Após o resultado obtido acreditamos que o mesmo possa ser utilizado na verificação da avaliação da confiabilidade em outros contextos. A justificativa para tal afirmação deve-se ao fato de termos considerado as teorias genéricas para ES apresentadas pelos autores Cruzes e Dyba (2011a). A utilização menos restrita do mesmo pode ratificar o nível de confiança dos resultados obtidos a partir da aplicação do mesmo.

Os conceitos e as questões adotadas para avaliação da confiabilidade da síntese foram:

a) Conceito: Credibilidade

1. Os argumentos que conduzem às interpretações feitas na síntese temática foram explicitados?

A interpretação dos argumentos apresentados nos artigos foi adquirida a partir das revisões realizadas em todas as etapas. Este tópico teve importância destacada em função dos artigos terem sido escritos em inglês, língua não comum à dos pesquisadores envolvidos.

2. As suposições sobre uma abordagem específica para a síntese temática foram claramente explicadas?

Para a apresentação da síntese, utilizamos segmentos dos artigos para fundamentar nossas interpretações a respeito dos resultados encontrados pelos autores de cada artigo. Um exemplo seria a suposição de baixa concordância na detecção de smells, oriunda de alguns resultados individuais que tornavam esta afirmativa possível.

3. Existe um bom ajuste entre o que é reivindicado e o que a evidência mostra?

Em nosso caso, este item está intrinsecamente relacionado com a apresentação de nossas suposições (item 2). Uma vez que nossas suposições se basearam em transcrições de partes dos artigos, esta relação é direta.

4. As questões de pesquisa foram respondidas com base na evidência da síntese temática?

Sim, as questões de pesquisa foram respondidas. Foram identificados os temas que representam os principais aspectos abordados nos estudos sobre os efeitos de *code smells e a* semelhança/distância entre os contextos analisados dos estudos que investigam os efeitos de *smells* e o nível de convergência dos resultados dos artigos foram considerados para a criação do Mapa Temático.

5. A linguagem e os conceitos utilizados na síntese são consistentes?

Desde a etapa de extração, realizamos um esforço no sentido de convergir sobre uso de termos. Além disso, as revisões reforçaram a consistência.

b) Conceito: Confirmabilidade

6. Pesquisadores e especialistas concordam com a maneira que estes dados foram codificados e classificados?

Este tópico também foi obtido através das revisões.

c) Conceito: Confiança

7. Foi estabelecida e mantida a rastreabilidade das informações do artigo até a síntese?

Durante toda a execução do método houve a preocupação em manter a rastreabilidade das informações extraídas, codificadas e mesmo das decisões tomadas. A revisão realizada ao final de cada passo verificava, dentre outras coisas, se esta rastreabilidade estava garantida e consistente.

d) Conceito: Transmissibilidade

8. Os resultados obtidos podem ser transferidos para outros contextos?

Em uma avaliação superficial, podemos argumentar que a observação de *code smells* como indicativo de potenciais problemas de código pode ser considerada como um contraponto para a análise da qualidade do *design* de software. Esta é uma discussão que está latente, mas que requer uma análise mais aprofundada.

Podemos concluir nesta Seção, que todas as preocupações transmitidas por Cruzes e Dyba (2011a) para que houvesse uma maior confiabilidade da síntese foram consideradas e seguidas nesta dissertação.

5.3 CONCLUSÃO DO CAPÍTULO

Neste Capítulo foi apresentada a aplicação prática do método de síntese temática para o tema *Code Smells*. No Apêndice A constam os resultados obtidos com a síntese temática e no Anexo 1 consta o artigo relacionado a estes resultados, trazendo também toda a contextualização inicial do processo da revisão sistemática realizada para o tema *Code Smells*. Este artigo encontra-se no formato original e encontra-se em fase de revisão.

No Capítulo 6 será apresentada a conclusão do trabalho, reportando as lições aprendidas, principais contribuições, limitações, publicações e trabalhos futuros.

Capítulo

6

Este capítulo apresenta a conclusão desta tese de mestrado, juntamente com as suas contribuições. Também são apresentados os trabalhos futuros, as limitações e lições aprendidas.

6 CONCLUSÃO

Este trabalho tratou a carência de suporte teórico para o método de síntese temática em ES, considerando o contexto acadêmico em que as pesquisas científicas ocorrem dentro das universidades. Nestes casos, as pesquisas são comumente realizadas por alunos de mestrado e doutorado, orientados por um professor pesquisador. Neste contexto, os conhecimentos dos pesquisadores envolvidos se apresentam em diversos níveis, tanto sobre os métodos e abordagens que podem ser utilizados, quanto em relação aos temas centrais que podem ser explorados nas pesquisas. O objetivo principal do trabalho foi ampliar o suporte necessário para aplicação de métodos qualitativos em ES, mais especificamente síntese temática. Para tanto, subsídios foram fornecidos, contemplando recomendações sobre o que deve ser considerado quando uma síntese temática for realizada, principalmente em contexto acadêmico.

O trabalho foi realizado seguindo uma sequência lógica de atividades definidas no início da pesquisa que envolveu: (i) levantamento bibliográfico; (ii) sumarização das diretrizes propostas por Cruzes e Dyba (2011a); (iii) definição e criação de suporte ferramental; (iv) realização da síntese temática para *code smell*; (v) refinamento do método de síntese temática.

O restante deste capítulo aborda na Seção 6.1, as lições aprendidas coletadas durante a realização da pesquisa; na Seção 6.2, as principais contribuições do trabalho; na Seção 6.3, algumas limitações; e na Seção 6.4, a apresentação de uma agenda de publicações com a proposta de trabalhos futuros que podem ser realizados para aperfeiçoar o método de síntese temática proposto nesta pesquisa.

6.1 CONTRIBUIÇÕES

A principal contribuição deste trabalho é o refinamento do método de síntese temática proposto por Cruzes e Dyba (2011a). O método refinado define uma série de orientações adicionais que os pesquisadores precisam se preocupar quando trabalham com uma abordagem qualitativa, especificamente síntese temática. Estas orientações mostram-se necessárias, principalmente, para adoção no contexto acadêmico, onde pesquisadores estão sendo formados durante a realização das pesquisas.

Como contribuições secundárias, o trabalho deixa a sumarização do método proposto por Cruzes e Dyba (2011a), incluindo a criação do *Checklist* de Confiabilidade da Síntese. Esta sumarização apresenta-se em formato de tabela e facilita a adoção do método, uma vez que busca sistematizar a apresentação das recomendações passadas pelos autores em seu trabalho.

Além da sumarização, este trabalho desenvolveu uma síntese temática tendo como tema central a área de *code smells*. A realização desta síntese ocorreu em paralelo com o refinamento do método e foi imprescindível para que este fosse refinado. Através da realização prática desta síntese, as recomendações de Cruzes e Dyba (2011a) puderam ser complementadas pelas experiências empíricas adquiridas com a realização desta síntese e percepção das necessidades. Todo este processo norteou a escrita do método de síntese temática refinado, considerando o contexto acadêmico reportado nesta dissertação.

6.2 LIMITAÇÕES

O método de síntese temática adotado como base para a realização do trabalho não especificava como os passos deveriam ser realizados. Para enfrentar esta dificuldade, tivemos que focar no aprendizado empírico que era absorvido durante a execução prática da síntese, em conjunto com alguns direcionamentos seguidos de outras fontes bibliográficas.

Como limitações percebidas durante o trabalho apresentado nesta dissertação, podemos citar:

- Ausência de adoção de uma ferramenta de análise qualitativa que facilitasse a tarefa de extrair e analisar os dados, mantendo uma rastreabilidade automática e diminuindo o trabalho e a complexidade inerente. Muita energia faz-se necessária para tornar os dados sistematicamente comparáveis dentro de um contexto, sem a geração de retrabalhos e releituras dos artigos;
- A impossibilidade de determinadas generalizações de resultados frente à própria natureza da pesquisa qualitativa que se apresenta específica para um determinado contexto, tempo e grupo de participantes. Para este trabalho foi enfrentada a dificuldade de generalizar quando vários *smells* eram analisados nos artigos;
- Ausência de um padrão estrutural para alguns dos artigos em ES selecionados, apresentando muitas vezes a ausência de informações importantes que ajudam no entendimento do contexto para o qual o estudo foi realizado. Informações relacionadas aos participantes, heurísticas adotadas, tamanho das aplicações adotadas e até mesmo as questões de pesquisa eram negligenciados. Para estes exemplos citados, mesmo que não chegue a inviabilizar o método, tanto a extração das informações quanto a generalização dos achados ficam prejudicadas;
- Ausência de conhecimentos sobre o tema *code smells* por parte da autora desta dissertação, sendo de fundamental importância a participação de um especialista no assunto para a realização da síntese temática, validando as extrações, códigos, subtemas e temas encontrados, tentando além de garantir a confiabilidade do método, garantir a consistência e coerência dos achados considerados relevantes.

6.3 LIÇÕES APRENDIDAS

6.3.1 Delimitações das Questões de Pesquisa

O tema principal adotado para a realização da síntese temática foi *code smells*, mas nem todas as vertentes associadas a este tema interessavam ao foco do trabalho. Existiam quatro pesquisadores envolvidos na seleção dos estudos primários. Uma das questões de pesquisa mencionava quais seriam os aspectos (ou seja, os temas) que estavam sendo estudados nas investigações sobre o efeito de *smells*, mas não especificava claramente que seriam apenas os efeitos de *smells* no desenvolvimento/manutenção de sistemas. Este fato possibilitou que se chegasse à fase de extração com três artigos que trabalhavam vertentes diferentes. Dois tratavam de *code smells* relacionados a aspectos e um tratava de *lexicon code smells*. Como o foco principal do trabalho era o efeito do *code smells* no desenvolvimento/manutenção de software, foi feita uma reunião entre os pesquisadores, onde este entendimento foi fechado, sendo realizada uma análise sobre a exclusão dos mesmos. Após a reunião houve concordância que estes não faziam parte do escopo que deveria ser considerado para a síntese temática. Delimitar melhor as questões de pesquisa, já no início do trabalho, poderia ter evitado este tipo de problema enfrentado.

6.3.2 Suporte Ferramental

A decisão por um método suportado por ferramentas ou manual é um passo importante. Pensamos que pelo fato do nosso processo ser manual, fazia-se necessário que instrumentos de coleta/extração fossem criados para facilitar a realização da síntese temática.

Aprendemos que no início do estudo deve-se planejar e gastar o tempo que for necessário para a construção destes instrumentos. Neste trabalho, embasados pelas questões de pesquisa definidas, foi criada inicialmente a Planilha de Extração/Mapeamento para abarcar a coleta/extração/codificação das dimensões definidas no arcabouço de extração. No

entanto, não consideramos a dimensão Contexto nesta Planilha. Desta forma, o sistema desenvolvido em PHP que havia sido planejado apenas com a função de auxiliar nas análises e resultados, dado a sua capacidade de fazer os devidos cruzamentos nos dados coletados/extraídos, precisou ser utilizado também como um instrumento de coleta para os dados relacionados ao contexto (*empirical method, type application, size of application, experimental unit, smells, evaluations' heuristics*).

6.3.3 Quantidade de artigos selecionados

A quantidade de artigos selecionados, após a etapa de buscas e seleção de estudos primários, deve ser observada. Esta observação irá direcionar a forma com que o trabalho (espécie de ciclo de vida) será realizado. Busca-se com isto aumentar a confiabilidade e a produtividade da pesquisa.

Aprendemos que, quando o processo de seleção resulta em uma quantidade razoável de artigos, é recomendado adotar uma abordagem incremental e iterativa. Dividir os artigos em etapas pode ser uma solução viável para facilitar a imersão e evitar retrabalhos. Iterações são, neste contexto, entendidas como os passos que precisam ser seguidos dentro do método proposto. Incrementos devem ser entendidos como os crescimentos dos resultados que vão sendo obtidos com a aplicação do método em pequenas partes fracionadas dos artigos selecionados.

6.3.4 Realização da síntese em um método iterativo e incremental

Durante a execução do método foi notório o crescimento do conhecimento a respeito do tema e do próprio método que estava sendo adotado. A escolha do ciclo de vida adotado, no caso, incremental e iterativo, teve uma contribuição relevante para esta construção do conhecimento, tornando-se uma lição aprendida que vale ser compartilhada.

6.4 TRABALHOS FUTUROS

Como trabalho futuro, podemos destacar algumas atividades que podem ser realizadas com a finalidade de aperfeiçoar o método de síntese temática refinado para esta dissertação. Sendo elas:

- Outra aplicação prática do método apresentado nesta dissertação para a área de ES, em um contexto semelhante;
- Ajuste do método refinado nesta dissertação através de oportunidades de melhoria percebidas na aplicação em outro tema dentro de ES, considerando o contexto acadêmico apresentado nesta dissertação;
- Análise da usabilidade e confiabilidade do método em outros contextos, diferentes do apresentado nesta dissertação.

Está prevista a publicação futura do artigo relacionado ao trabalho desenvolvido nesta dissertação de mestrado, apresentado como Anexo 1. O artigo está em fase de revisão.

REFERÊNCIAS

ABEBE, S. L. *et al.* The Effect of Lexicon Bad Smells on Concept Location in Source Code. **Source Code Analysis and Manipulation (SCAM). 11th IEEE International Working Conference on (2011a)**, 2011. 125–134.

ALVES, N. S. R. *et al.* Towards an Ontology of Terms on Technical Debt. **In: the Sixth International Workshop on Managing Technical Debt**, Victoria, British Columbia, 2014.

ALVES, N. S. R. *et al.* Identification and Management of Technical Debt: A Systematic Mapping Study. **Submitted to Information and Software Technology**, 2015.

ARKSEY, H.; O'MALLEY, L. Scoping studies: towards a methodological framework. **International Journal of Social Research Methodology**, Fevereiro 2005. 19–32.

BARROSO, J. *et al.* The Challenges of Searching for and Retrieving Qualitative Studies. **West J Nurs Res.**, v. 25, n. 02, p. 153-178, Março 2003.

BASILI, G. *et al.* Packaging researcher experience to assist replication of experiments. **ISERN Meeting**, Sydney, 1996. 3-6.

BIOLCHINI, J.; TRAVASSOS, G. H. Revisões sistemáticas aplicadas a engenharia de software. **In XXI SBES - Brazilian Symposium on Software Engineering**, João Pessoa, PB, Brasil, 2007.

CAMPBELL, D. T.; FISKE, D. W. Convergent and discriminant validation by the multitrait-multimethod matrix. **Psychological Bulletin**, n. 56, p. 81-105, 1959.

CAMPBELL, R. *et al.* Evaluating meta-ethnography: a synthesis of qualitative research on lay experiences of diabetes and diabetes care. **Soc Sci Med**, 2003. 671-684.

CAMPOS, M. M. Pesquisa participante: possibilidades para o estudo da escola. **Cadernos de Pesquisa**, n. 49, p. 63-66, 1984.

CARNEIRO, G. *et al.* **Identifying code smells with multiple concern views**. In: Proceddings of the 1th Brazilian Conference on Software: Theory and Practice (CBSOFT). [S.l.]: [s.n.]. 2010. p. 128-137.

CASSEL, C.; SYMON, G. **Qualitative Methods in organizational research**. Londres: Sage Publications, 1994.

CHATZIGEORGIOU, A.; MANAKOS, A. Investigating the evolution of bad smells in object-oriented code. **In: Proceddings of the 7th International Conference on the Quality of Information and Communications Technology (QUATIC)**, 2010. 106-115.

COOK, T. D.; REICHARDT, C. S. (Eds.). **QUALITATIVE AND QUANTITATIVE METHODS IN EVALUATION RESEARCH**. SAGE PUBLICATIONS, INC; UNITED STATES OF AMERICA: [s.n.], v. 1, 1979.

CORBIN, J.; STRAUSS, A. **Basics of qualitative research:** Grounded theory procedures and techniques. 3ª Edição. ed. Thousand Oaks, Califórnia: SAGE Publications, Inc., 2008.

COUNSELL, C. Formulating questions and locating primary studies for inclusion in systematic reviews. **Ann Intern Med**, Setembro 1997. 380-719.

CRESWELL, J. W. **Educational Research:** Planning, Conducting, and Evaluating Quantitative and Qualitative Research. 3º Edição. ed. [S.l.]: Prentice Hall, 2007.

CRUZES, D. S. Análise Secundária de Estudos Experimentais em Engenharia de Software. **Dissertação de Mestrado. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação**, Campinas, São Paulo, 2007.

CRUZES, D. S. *et al.* Extracting Information from Experimental Software Engineering Papers. **In: Proceedings SCCC'07**, 2007. 105– 114.

CRUZES, D. S.; DYBA, T. Recommended Steps for Thematic Synthesis in Software Engineering. **In: Proceedings of the 2011 International Symposium on Empirical Software Engineering and Measurement**, 2011a. 275-284.

CRUZES, D. S.; DYBÅ, T. Research synthesis in software engineering: A tertiary study. **Information and Software Technology 53**, 2011b. 440-455.

DA SILVA, F. Q. *et al.* Six years of systematic literature reviews in software engineering: An updated tertiary study. **Information and Software Technology**, v. 53, n. 9, p. 899-913, 2011.

DALFOVO, M. S.; LANA, R. A.; SILVEIRA, A. Métodos quantitativos e qualitativos: um resgate teórico. **Revista Interdisciplinar Científica Aplicada**, Blumenau, v. 2, n. 4, p. 01-13, 2008.

D'AMBROS, M.; BACCHELLI, A.; LANZA, M. On the impact of design flaws on software defects. **Iin: Proceddings of the 10th International Conference on Quality Software (QSIC)**, 2010. 23-31.

DE-LA-TORRE-UGARTE, M. C.; GUANILO, R. F. T.; BERTOLOZZI, M. R. Revisão sistemática: noções gerais. **Revista da Escola de Enfermagem**, São Paulo, v. 45, n. 5, p. 1260-1266, 2011. Disponivel em: <http://www.scielo.br/pdf/reeusp/v45n5/v45n5a33.pdf>.

DUFFY, M. E. Methodological triangulation: a vehicle for merging quantitative and qualitative research methods. **In Journal of Nursing Scholarship**, v. 19, n. 3, p. 130-133, 1987.

DYBA, T.; DINGSOYR, T.; HANSSEN, G. K. Applying systematic reviews to diverse study types: An experience report. **In Proceedings of the First International Symposium on Empirical Software Engineering and Measurement, ESEM, IEEE Computer Society**, Washington, DC, USA, 2007. 225–234.

DYBA, T.; KITCHENHAM, B.; JøRGENSEN, M. Evidence-based Software Engineering for Practitioners. **IEEE Software**, v. 22 (1), p. 58-65, Janeiro 2005.

FABBRI, S. *et al.* Managing literature reviews information through visualization. **In International Conference on Enterprise Information Systems.14th. ICEIS**, Breslávia, Polônia, Junho 2012.

FLOERSCH, J. *et al.* Integrating Thematic, Grounded Theory and Narrative Analysis: A Case Study of Adolescent Psychotropic Treatment. **Qualitative Social Work**, v. 9, p. 407-425, Setembro 2010.

FONTANA, F. *et al.* **Investigating the impact of code smells on system's quality:** An empirical study on systems of different application domains. In: Proceddings of the 29th IEEE International Conference on Software Maintenance (ICSM). [S.l.]: [s.n.]. 2013. p. 260-269.

FOWLER, M. **Refactoring:** improving the design of existing code. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999.

GODOY, A. S. Introdução à pesquisa qualitativa e suas possibilidades. **In: Revista de Administração de Empresas**, v. 35, n. 2, p. 57-63, Mar/Abril 1995.

HAMER, S.; GILL, C. **Achieving evidence-based practice:** a handbook for practitioners. 2º Edição. ed. Londres: Baillière Tindall, 2005.

JEDLITSCHKA, A.; CIOLKOWSKI, M.; PFAHL, D. Reporting Experiments in Software Engineering. In: SHULL, F.; SINGER, J.; SJøBERG, D. I. K. **Guide to Advanced Empirical Software Engineering**. London: Kluwer Academic Publishers, 2008. Cap. Capítulo 8, p. 201-228.

JICK, T. D. Mixing qualitative and quantitative methods: Triangulation in action. **In Administrative Science Quarterly**, v. 24, n. 4, p. 602-611, Dezembro 1979.

KHOMH, F.; PENTA, M. D.; GUEHENEUC, Y.-G. **An exploratory study of the impact of code smells on software change-proneness**. In: Proceddings of the 16th Working Conference on Reverse Engineering (WCRE). [S.l.]: [s.n.]. 2009. p. 75-84.

KITCHENHAM, B. A. **Procedures for Performing Systematic Reviews**. Technical Report TR/SE-0401. Keele University, and Technical Report 0400011T.1, NICTA. 2004.

KITCHENHAM, B. A.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering. **Technical Report EBSE 2007-001**, Keele University and Durham University Joint Report, Julho 2007.

KITCHENHAM, B. A.; DYBA, T.; JøRGENSEN, M. **Evidence-based Software Engineering**. In Proceddings ICSE'04. Edimburgo, Escócia: [s.n.]. Maio, 2014. p. 273-281.

KITCHENHAM, B. *et al.* Systematic literature reviews in software engineering. A tertiary study. **Information and Software Technology**, v. 52, n. 8, 2010.

KITCHENHAM, B.; DYBA, T. . J. M. Evidence-based Software Engineering. **In: Proceedings of the 26th International Conference on Software Engineering (ICSE'04)**, 2004.

LANZA, M.; MARINESCU, R. **Object-Oriented Metrics in Practice**. [S.l.]: Springer-Verlag New York, Inc., 2005.

LI, W.; SHATNAWI, R. An empirical study of the bad smells and class error probability in the post-release object-oriented system evolution. **Journal of Systems and Software**, v. 80, n. 7, p. 1120-1128, 2007.

MACIA, I. *et al.* On the impact of aspect-oriented code smells on architecture modularity: An exploratory study. **In: Proceedings of the 5th Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS)**, 2011a. 41-50.

MACIA, I. *et al.* Are Automatically-Detected Code Anomalies Relevant to Architectural Modularity? An Exploratory Analysis of Evolving Systems. **In Proceddings of the 11th Annual International Conference on Aspect-oriented Software Development (AOSD)**, Potsdam, Germany, Março 2012a. 167-178.

MACIA, I. *et al.* On the relevance of Code Anomalies for identifying architecture degradation symptoms. **In: Proceedings of the 16th European Conference on Software Maintenance and Reengineering (CSMR)**, 2012b. 277-286.

MACIA, I.; GARCIA, A.; VON STAA, A. An exploratory study of code smells in evolving aspect-oriented systems. **In: Proceedings of the 10th International Conference on Aspect-oriented Software Development (AOSD)**, 2011b. 203-214.

MANTYLA, M. An experiment on subjective evolvability evaluation of objectoriented software: explaining factors and interrater agreement. **In: Proceedings of the 4th International Symposium on Empirical Software Engineering (ISESE)**, 2005.

MANTYLA, M. V.; VANHANEN, J.; LASSENIUS, C. Bad smells humans as code critics. **In: Proceedings of the 20th IEEE International Conference on Software Maintenance (ICSM)**, 2004. 399-408.
MANTYLA, M.; LASSENIUS, C. Subjective evaluation of software evolvability using code smells: An empirical study. **Empirical Software Engineering**, v. 11, n. 3, p. 395-431, 2006.

MARINESCU, R. **Detecting Design Flaws via Metrics in Object-Oriented Systems**. 39th International Conference and Exhibition on Technology Of Object-oriented Languages And Systems. Santa Barbara, CA: [s.n.]. 2001. p. 173-182.

MARINESCU, R. **Detection Strategies:** Metric Based Rules for Detecting Design Flaws. IEEE International Conference on Software Maintenance (ICSM). [S.l.]: [s.n.]. 2004. p. 350-359.

MARINESCU, R.; MARINESCU, C. **Are the clients of flawed classes (also) defect prone?** In: Proceddings of the 11th IEEE International Working Conference on Source Code Analysis and Manipulation (SCAM). [S.l.]: [s.n.]. 2011. p. 65-74.

MENDES, T. S. *et al.* VisMinerTD – An Open Source Tool to Support the Monitoring of the Technical Debt Evolution using Software Visualization. **In: 17th International Conference on Enterprise Information Systems (ICEIS)**, Barcelona, Espanha, 2015.

MONTEBELO, R. P. *et al.* **SRAT (Systematic Review Automatic Tool) - uma ferramenta computacional de apoio à Revisão Sistemática**. Fundação de Ensino Eurípedes Soares da Rocha. In Experimental Software Engineering Latin American Workshop, p. 13-22. 2007.

MORSE, J. Approaches to qualitative-quantitative methodological triangulation. **Nursing Research**, v. 40, n. 1, p. 120-132, 1991.

NGUYEN-DUC, A.; CRUZES, D. S.; CONRADI, R. The impact of global dispersion on coordination, team performance and software quality a systematic literature review. **Information and Software Technology 57**, 2015. 277-294.

NOBLIT, G. W.; HARE, R. D. **Meta-Ethnography:** Synthesizing qualitative studies. [S.l.]: Newbury Park: Sage, 1988.

NOVAIS, R. *et al.* On the proactive and interactive visualization for feature evolution comprehension: An industrial investigation. **In: 34th International Conference on Software Engineering (ICSE)**, Zurich, 2012. 1044-1053.

NOVAIS, R. L. *et al.* Software evolution visualization: A systematic mapping study. **Information and Software Technology**, 2013. 1860-1883.

OLBRICH, S. *et al.* The Evolution and Impact of Code Smells: A Case Study of Two Open Source Systems. **In 3rd International Symposium on Empirical Software Engineering and Measurement (ESEM)**, 2009. 390-400.

OLBRICH, S. M.; CRUZES, D. S.; SJBERG, D. I. K. Are all code smells harmful? a study of god classes and brain classes in the evolution of three open source systems. **In Proceddings of the 26th International Conference on Software Maintenance (ICSM)**, 2010. 1-10.

PASSOS, C. Understanding the belief systems behind software engineering practice: Studies on evidence-based practices in an industrial setting. **Tese (doutorado) – Universidade Federal da Bahia, Instituto de Matemática, Universidade Salvador, Universidade Estadual de Feira de Santana**, Salvador, Bahia, 2014.

PASSOS, C. *et al.* Challenges of applying ethnography to study software practices. **In: ESEM 2012, Lund. Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement**, New York: ACM Press. v. 1, 2012. 9-18.

PATERSON, B. L. *et al.* **Meta-Study of Qualitative Health Research:** A Practical Guide to Meta-Analisys and Meta-Synthesis. Thousand Oaks, California: Sage Publications, Inc., v. III, 2001.

PETERS, R.; ZAIDMAN, A. **Evaluating the lifespan of code smells using software repository mining**. In: Proceddings of the 16th European Conference on Software Maintenance and Reengineering (CSMR). [S.l.]: [s.n.]. 2012. p. 411-416.

PETERSEN, K. *et al.* Systematic mapping studies in software engineering. **In 12th International Conference on Evaluation and Assessment in Software Engineering**, Austrália, 2008. 71-80.

PETTICREW, M.; ROBERTS, H. **Systematic Reviews in the Social Sciences:** A Practical Guide. [S.l.]: Blackwell Publishing, 2006.

POPAY, J. *et al.* **Guidance on the conduct of narrative synthesis in systematic reviews**. [S.l.]: A product from the ESRC Methods Programme (Institute of Health Research: ESRC Methods Program, Lancaster)., 2006.

POPE, C.; MAYS, N. Reaching the parts oyher methods cannot reach: an introduction to qualitative methods in health and health service research. **In British Medical Journal**, nº 311 1995. 42-45.

RAHMAN, F.; BIRD, C.; DEVANBU, P. **Clones:** What is that smell? Iin Working Conf. Mining Softw. Repositories (MSR). [S.l.]: [s.n.]. 2010. p. 72-81.

RICHARDSON, R. J. **Pesquisa Social - Métodos e Técnicas**. 3ª edição. ed. São Paulo: Atlas, 2008.

RIEL, A. J. **Object-Oriented Design Heuristics**. 1ª Edição. ed. Boston, USA: Addison-Wesley Longman Publishing Co., Inc., 1996.

SALDAÑA, J. **The Coding Manual for Qualitative Researchers**. Londres: Sage Publications, 2008.

SANDELOWSKI, M.; BARROSO, J. **Handbook for synthesizing qualitative research**. Nova York: Springer Publishing Company Inc., 2007.

SANTOS, J. A. *et al.* The problem of conceptualization in god class detection: agreement, strategies and decision drivers. **Journal of Software Engineering Research and Development (JSERD)**, v. 2, n. 11, p. 1–33, 2014.

SANTOS, J. A. M.; MENDONÇA, M. G. Identifying strategies on god class detection in two controlled experiments. **In: Proceedings of the 26th International Conference on Software Engineering and Knowledge Engineering (SEKE)**, 2014. 244–249.

SANTOS, J. A. M.; MENDONÇA, M. G. Exploring Decision Drivers on God Class Detection in Three Controlled Experiments. **In: Proceedings ACM/SIGAPP Symposium On Applied Computing**, Salamanca, Espanha, 2015. 1472-1479.

SANTOS, J. A. M.; MENDONÇA, M. G.; SILVA, C. V. A. An exploratory study to investigate the impact of conceptualization in god class detection. **In: 17th International Conference on Evaluation and Assessment in Software Engineering (EASE)**, Porto de Galinhas -PE, Brazil, 2013. 48-59.

SCHUMACHER, J. *et al.* Building Empirical Support for Automated Code Smell Detection. **In Proceddings of the 4th International Symposium on Empirical Software Engineering and Measurement (ESEM)**, Bolzano-Bozen, Itália, Setembro 2010. 1-10.

SEAMAN, C. B. Qualitative Methods in Empirical Studies of Software Engineering. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, v. 25, n. 4, p. 557-572, julho 1999. ISSN [doi>10.1109/32.799955].

SILVA, F. Q. B. *et al.* **Using meta-ethnography to synthesize research:** A worked example of the relations between personality and software team processes. In: ACM / IEEE International Symposium on Empirical Software Engineering and Measurement. [S.l.]: [s.n.]. 2013.

SJøBERG, D. I. K. *et al.* Quantifying the Effect of Code Smells on Maintenance Effort. **IEEE Transactions on Software Engineering 39 (8)**, Agosto 2013. 1144-1156.

SJOEBERG, D. I. K. *et al.* A survey of controlled experiments in software engineering. **IEEE Trans. Software Engineering**, n. 31, p. 733-753, Setembro 2005.

SOARES, H. F. *et al.* Investigating the Link between User Stories and Documentation Debt on Software Projects. **In the 12th International Conference on Information Technology : New Generations**, Las Vegas, EUA, 2015.

SOMMERVILLE, I.; MASERA, L.; DEMARIA, C. Practical Guidelines for Ada Reuse in an Industrial Environment. **In: Proceddings 2nd Symposium on Software Quality**, Florence. Heidelberg, 1995.

THOMAS, J.; HARDEN, A. Methods for the thematic synthesis of qualitative research in systematic reviews. **BMC Med Res Methodol**, v. 8, n. 45, 2008.

VAN SOLINGEN, R. *et al.* Goal Question Metric (GQM) Approach. In: SOLINGEN, V. **Encyclopedia of Software Engineering**. [S.l.]: John Wiley & Sons, v. II, 2002. p. 578-583.

VAUCHE, S. *et al.* **Tracking design smells:** Lessons from a study of god classes. In: Proceddings of the 16th Working Conference on Reverse Engineering (WCRE). [S.l.]: [s.n.]. 2009. p. 145-154.

WOHLIN, C. *et al.* **Experimentation in Software Engineering – An Introduction**. Boston: Kluwer Academic Publishers, 2000. ISBN 0-7923-8682-5.

YAMASHITA, A.; COUNSELL, S. Code smells as system-level indicators of maintainability: An empirical study. **Journal of Systems and Software 86 (10)**, 2013. 2639-2653.

YAMASHITA, A.; MOONEN, L. Do code smells reflect important maintainability aspects? **In Proceddings of the 28th the International Conference on Software Maintenance (ICSM)**, 2012. 306-315.

YAMASHITA, A.; MOONEN, L. **Do developers care about code smells? an exploratory survey**. In: Proceddings of the 20th Working Conference on Reverse Engineering (WCRE). [S.l.]: [s.n.]. 2013. p. 242-251.

YAMASHITA, A.; MOONEN, L. Exploring the Impact of Inter-smell Relations on Software Maintainability: An Empirical Study. **In Proceddings of the 35th International Conference on Software Engineering (ICSE)**, San Francisco, CA, USA, 2013. 682-691.

YAMASHITA, A.; MOONEN, L. To what extent can maintenance problems be predicted by code smell detection? – An empirical study. **Information and Software Technology 55 (12)**, 2013. 2223-2242.

ZAMBONI, A. B. *et al.* **StArt Uma Ferramenta Computacional de Apoio à Revisão Sistemática**. In: Brazilian Conference on Software: Theory and Practice (CBSoft) - Tools session. UFBA. Salvador, Bahia: [s.n.]. 2010.

ZAZWORKA, N. *et al.* Investigating the Impact of Design Debt on Software Quality. **In Proceddings of the 2nd Workshop on Managing Technical Debt (MTD)**, Waikiki, Honolulu, HI, USA, 2011. 17-23.

APÊNDICE A – RESULTADOS DA SÍNTESE TEMÁTICA PARA CODE SMELLS

A.1 TEMAS IDENTIFICADOS

Foram encontrados cinco temas associados ao tema central da RS realizada. Inicialmente os códigos foram agrupados quanto à sua semelhança, dando origem aos subtemas. Após análises e comparações constantes estes subtemas foram revistos e reagrupados em temas que o representassem.

A Tabela 17 apresenta alguns dos resultados extraídos dos artigos que auxiliam na demonstração e exemplificação do significado dos temas e subtemas encontrados.

Tabela 17 Temas e Subtemas

Tema	Subtema	Exemplos de Resultados
Correlation with issues of development	*Changes and defects*	*Changes of infected components with ISP Violation is significant larger.*
	Effort	*God class don't require a higher maintainability e ort than non-code smells.*
	Design quality	*Code smells often related to immature design and implementation.*
	Architectural quality	*There is a correlation between code anomaly and problems in architectural design.*
	Harm	*Presence of God class and Brain class may be beneficial to a system.*
Human aspects	*Factors a affecting detection*	*Demographics aspects of the developers affect the code smell evaluations.*
	Agreement on detection	*The agreement on god class detection is low.*
	Human evaluation versus software measures	*The developers' evaluations on code smells do not correlate with related source code metrics.*
	Decision drivers	*Misplaced methods is the strongest driver for letting the subjects classify a class as a god class.*
	Knowledge about smells	*There is not strong understanding about code smells and practical application of these concepts.*
	Strategy on detection	*Participant with optimistic style investigate code smell candidates using more combinations of views when compared to participants with more conservative styles.*
	Detection difficulty	*The subjects perceive detecting god classes as an easy task.*
Programming	*Smell removal*	*The developers resolve code smells for opportunistic reasons.*
	Smell introduction	*The developers who revised most often also introduced the most smells.*
Detection	*Supporting smell*	*Developers need support to enhance detection of code*

	detection	*anomaly.*
	Smell as a predictor	*Expert judgment is most accurate method for assessing system maintainability.*
	Novel Smell	*A new smell was propose: Multiple Inheritance simulation.*
Other correlations	*Metrics versus smells*	*Code smells are strongly influenced by size (LOC).*
	Smell density	*Code smell density is likely to be inaccurate when comparing size differing systems.*
	Frequency of smells	*Some design flaws are more frequent than others.*
	Inter-smell relation	*There is interaction between code smells.*

Para cada tema/subtema, existe uma quantidade de códigos associadas que foram trabalhados e extraídos dos artigos selecionados. Na Tabela 18 apresentamos as relações dos subtemas com os temas e a quantidade de artigos que subsidiaram as descobertas.

Tabela 18 Distribuição dos artigos por Temas e Subtemas – sem referência aos artigos

Tema	**Subtema**	**Quantidade de artigos por subtema**	**Quantidade de artigos por tema**
Correlation with issues of development	*Changes and defects*	11	17
	Effort	4	
	Design quality	3	
	Architectural quality	2	
	Harm	2	
Human aspects	*Factors affecting detection*	7	7
	Agreement on detection	6	
	Human evaluation versus software measures	2	
	Decision drivers	1	
	Knowledge about smells	1	
	Strategy on detection	1	
	Detection difficulty	1	
Programming	*Smell removal*	7	7
	Smell introduction	4	
Detection	*Supporting smell detection*	6	12
	Smell as a predictor	5	
	Novel Smell	1	
Other correlations	*Metrics versus smells*	6	10
	Smell density	3	
	Frequency of smells	3	
	Inter-smell relation	2	

A quantidade de artigos por tema não é a soma da quantidade de artigos por subtema, visto que um mesmo artigo pode tratar de mais de um subtema. Um exemplo para entender melhor esta questão seria o fato dos artigos de Khomh *et al.* (2009) e de Yamashita e Moonen (2013c) estarem relacionados ao subtema *Changes and defects* e *Design quality*, fazendo com

que os mesmos sejam contabilizados em duplicidade no subtema, porém contabilizados distintivamente para os temas.

A Tabela 19 apresenta detalhes da categorização dos artigos por subtemas encontrados.

Tabela 19 Distribuição dos artigos por Temas e Subtemas – com referência aos artigos

Tema	Subtema	ID START	Referência
Correlation with issues of development	*Changes and defects*	9, 38, 60, 63, 210, 260, 334, 429, 461, 522, 697	(Chatzigeorgiou e Manakos, 2010), (D'Ambros *et al.*, 2010), (Khomh *et al.*, 2009), (Li e Shatnawi, 2007), (Marinescu e Marinescu, 2011), (Olbrich, 2009), (Rahman *et al.*, 2010), (Yamashita e Moonen, 2012), (Yamashita e Moonen, 2013b), (Yamashita e Moonen, 2013c), (Zazworka *et al.*, 2011)
	Effort	10, 26, 148, 429	(Chatzigeorgiou e Manakos, 2010), (Santos e Mendonça, 2014), (Schumacher, 2010), (Silva, 2013)
	Design quality	38, 63, 522	(Khomh *et al.*, 2009), (Yamashita e Moonen, 2013b), (Yamashita e Moonen, 2013c)
	Architectural quality	276, 278	(Macia, 2012a), (Macia, 2012b)
	Harm	60, 419	(Olbrich *et al.*, 2010), (Zazworka *et al.*, 2011)
Human aspects	*Factors affecting detection*	10, 427, 747, 823, 1509, 1941, 2000	(Carneiro, 2010), (Mantyla, 2005), (Mantyla *et al.*, 2004), (Mantyla e Lassenius, 2006), (Santos e Mendonça, 2014)
	Agreement on detection	10, 26, 427, 747, 823, 1509	(Carneiro, 2010), (Mantyla, 2005), (Mantyla *et al.*, 2004), (Mantyla e Lassenius, 2006), (Santos e Mendonça, 2014), (Schumacher, 2010)
	Human evaluation versus software measures	747, 823	(Mantyla *et al.*, 2004), (Mantyla e Lassenius, 2006)
	Decision drivers	26	(Schumacher, 2010)
	Knowledge about smells	1941	(Yamashita e Moonen, 2013a)
	Strategy on detection	427	(Carneiro, 2010)
	Detection difficulty	26	(Schumacher, 2010)
Programming	*Smell removal*	62, 148, 276, 277, 429, 526, 1509	(Chatzigeorgiou e Manakos, 2010), (Macia, 2012b), (Mantyla, 2005), (Peters e Zaidman, 2012), (Silva, 2013), (Vauche, 2009), (Yamashita e Counsell, 2013)
	Smell introduction	148, 276, 429, 526	(Chatzigeorgiou e Manakos, 2010), (Macia, 2012b), (Silva, 2013), (Vauche, 2009)
Detection	*Supporting smell detection*	26, 276, 427, 747, 1941, 2000	(Carneiro, 2010), (Fontana *et al.*, 2013), (Macia, 2012b), (Mantyla e Lassenius, 2006), (Schumacher, 2010), (Yamashita e Moonen, 2013a)
	Smell as a predictor	62, 63, 210, 522, 823	(Khomh *et al.*, 2009), (Mantyla *et al.*, 2004), (Yamashita e Counsell, 2013), (Yamashita e Moonen, 2012),

			(Yamashita e Moonen, 2013c)
	Novel Smell	63	(Yamashita e Moonen, 2013c)
Other correlations	*Metrics versus smells*	9, 26, 62, 63, 1509, 2000	(Fontana *et al.*, 2013), (Olbrich, 2009), (Schumacher, 2010), (Yamashita e Counsell, 2013), (Yamashita e Moonen, 2013a), (Yamashita e Moonen, 2013c)
	Smell density	60, 62, 419	(Olbrich *et al.*, 2010), (Yamashita e Counsell, 2013), (Zazworka *et al.*, 2011)
	Frequency of smells	419, 461, 2000	(D'Ambros *et al.*, 2010), (Fontana *et al.*, 2013), (Olbrich *et al.*, 2010)
	Inter-smell relation	38, 63	(Yamashita e Moonen, 2013b), (Yamashita e Moonen, 2013c)

A maioria dos artigos trata de assuntos relacionados à correlação existente entre *smells* e questões de desenvolvimento de sistemas, seguido da preocupação com a detecção de *code smells*. Apenas um artigo mencionou ter encontrado um novo *code smell* (Yamashita e Moonen, 2013c).

A.2. PRINCIPAIS RESULTADOS

A.2.1 Síntese Temática sobre Code Smells

Neste item serão apresentados os principais resultados obtidos com a realização da síntese temática para o tema central *Code Smells*, representando a visão descritiva obtida com a execução do trabalho desta dissertação. Estes resultados foram o foco principal do artigo escrito sobre o tema e que se encontra no Anexo 1 desta dissertação. O método adotado para obtenção destes resultados foi o refinamento apresentado no Capítulo 4 desta dissertação.

Para chegar aos resultados aqui explicitados foi necessário trabalhar as informações contidas no arcabouço de extrações, contendo as seguintes dimensões: Publicação, Objetivo, Contexto e Resultados.

Todas as informações extraídas que compunham a Planilha de Extração/Mapeamento, o Mapa Temático e as informações cadastradas na ferramenta desenvolvida foram comparadas e analisadas. Esta comparação deu-se com a ajuda de alguns

"*selects*" gerados no banco de dados da aplicação desenvolvida que permitiam cruzamento de dados e aumento da visibilidade da amplitude dos resultados. Este processo poderia ter sido feito de maneira manual, mas acreditamos que o uso da ferramenta desenvolvida nos possibilitou uma maior confiabilidade dos resultados encontrados.

De uma forma geral pode-se perceber que os estudos foram realizados em contextos muito amplos, dificultando as possibilidades de generalizações dos resultados encontrados.

Correlation with issues of development

Este foi o tema com o maior número de artigos relacionados. Houve dezessete estudos correlacionando *smells* a aspectos relacionados com o desenvolvimento de software.

Dentro deste tema foram encontrados cinco subtemas: *changes and defects, effort, design quality, architectural quality, harm*. As análises mais interessantes para a síntese temática foram relacionadas com os subtemas: *changes and defects, effort* e *architectural quality* e serão apresentadas a seguir.

Changes and defects **(Mudanças e defeitos).** Por mudanças, foram consideradas as atividades de manutenção ou de refatoração mencionadas nos artigos. Mudanças e Defeitos foi o principal subtema de todas as categorias, apresentando onze artigos relacionados. Oito destes eram estudos de correlação e três eram experiências realizadas em um ambiente in vivo controlado, com base na mesma configuração experimental. Os artigos avaliaram os sistemas de origem comercial ou open source, de tamanhos médios e grandes. Devido ao seu foco em evolução, eles adotaram a detecção automática. Apenas os experimentos controlados adotaram a avaliação humana, bem como a detecção automática. No total, os estudos que analisavam a correlação de *smells* com mudanças/defeitos apresentados nos sistemas, trabalharam com cinquenta e três *smells*, o que dificulta uma possível generalização das descobertas sobre um tipo específico de *smell*.

Apesar das dificuldades de generalizações, uma discussão geral foi possível.

Foram encontradas algumas conclusões divergentes nos artigos analisados para este item. Em alguns casos, percebe-se nitidamente que a adoção do conceito de *smell* é considerada uma prática útil no desenvolvimento de software. Porém ao analisar outros estudos, justamente o oposto é explicitado.

A Tabela 20 apresenta alguns desses resultados com a finalidade de auxiliar a identificar com maior clareza as convergências/divergências encontradas.

Tabela 20 Convergências e Divergências encontradas entre os artigos

Convergentes		Divergentes	
Referência	**Resultado**	**Referência**	**Resultado**
(Olbrich, 2009)	*The change proneness of components with smells is higher than the ones without.*	(Yamashita e Moonen, 2013b)	*There are smells that are not associated with maintenance problems.*
(Zazworka *et al.*, 2011)	*God classes impacts maintainability.*	Yamashita e Moonen (2013c)	*The role of code smells on the overall system maintainability is relatively minor*
(Yamashita e Moonen, 2012)	*There is a correlation between maintainability factors and code smell.*	(Rahman *et al.*, 2010)	*There is no correlation between clones and bugs.*
(Khomh *et al.*, 2009)	*Classes with code smells are more change-prone than others.*	(Marinescu e Marinescu, 2011)	*Taken in isolation, classes exhibiting the identity disharmonies design flaws do not have an increased likelihood to exhibit defects that classes which do not reveal design flaws.*
(Li e Shatnawi, 2007)	*Some bad smells are associated significantly with class error proneness.*	(D'Ambros *et al.*, 2010)	*Design flaws can't be considered more harmful with respect to software defects.*

Na coluna da esquerda podem ser percebidos alguns exemplos de artigos que apresentam descobertas que convergem, ou seja, consideraram em suas conclusões *code smell* como um conceito útil para ser adotado no desenvolvimento de *software*. Já a coluna da direita apresenta alguns exemplos contrários, ou seja, que divergem em relação a estes achados.

Sendo assim, não foi encontrado nenhum respaldo que nos permitisse algum nível de generalização para este subtema.

***Effort* (Esforço).** Este se apresentou como o segundo subtema dentro do tema principal em quantidade de artigos que o referencia. Quatro dos vinte e seis artigos selecionados abordavam questões relacionadas a este subtema. Dois destes artigos identificaram uma correlação negativa entre *smells* e esforço de atividades relacionadas a manutenção de sistemas. O artigo de Silva (2013) descobriu que, após ajustar o tamanho do arquivo e o número de mudanças, *code smells* não se apresentam significativamente associados ao aumento de esforço. Os autores consideram que o tamanho do arquivo e o número de alterações (*code churn*) representam melhor uma possível relação com a variação de esforço do que *code smells* propriamente dito. Seguindo nossa análise, o artigo de Schumacher (2010) também afirma que o smell *God Class* não exige uma maior capacidade de esforço de manutenção que as classes que não apresentam *code smells*. Apesar de encontrar correlações negativas entre *smells* e esforço em atividades de manutenção, é

importante considerar que o número de estudos em consonância sobre este aspecto é baixa (apenas dois). Os outros dois artigos neste subtema abordam outros aspectos relacionados a esforço. O artigo de Santos e Mendonça (2014) abordou o impacto das características de software para o esforço de detecção de *God Class*. Os autores consideram que as características de software não impactam o esforço de detecção de *God Class*. Além disso, o artigo de Chatzigeorgiou e Manakos (2010) propõe uma estratégia de aplicação de esforço. Para os seus autores, o esforço de manutenção deve priorizar o *smell Long Method* sobre os demais *code smells*.

Architectural quality **(Qualidade Arquitetural).** Este subtema apresentou alguns resultados interessantes, apesar do fato de que apenas dois estudos o abordaram. Existiu uma clara convergência e consistência entre os resultados apresentados pelos estudos. Ambos os artigos (Macia, 2012a; Macia, 2012b) identificaram problemas nas estratégias de detecção *smells* pela arquitetura. O artigo de Macia (2012b) descobriu que existe uma correlação entre anomalias de código e degradação de arquitetura. No entanto, apesar de notar alguma correlação entre *smells* e degradação arquitetural, os autores notaram que não existe uma anomalia de código específica que seja mais relevante como indicador desta degradação. Apresentando também uma correlação negativa entre *smells* e qualidade arquitetural, o artigo de Macia (2012a) descobriu que estratégias de detecção apresentam uma tendência em direcionar os desenvolvedores de forma equivocada quando endereçam anomalias de código a problemas de modularidade arquitetural.

Human Aspects

Este tema estava presente em sete dos vinte e seis artigos selecionados para a RS.

Foram identificados sete subtemas relacionados: *factors a affecting detection, agreement on detection, human evaluation versus software measures, decision drivers, knowledge about smells, strategy on detection, detection difficulty*. Como anteriormente mencionado, discutimos apenas os subtemas mais relevantes.

Factors affecting detection **(Fatores que afetam a detecção).** Todos os estudos que abordam aspectos humanos apresentam conclusões sobre este subtema, sendo que diferentes aspectos de contexto são considerados. Três artigos realizam experimentos controlados, três realizaram *surveys* e um apresentou um estudo de correlação. Os artigos utilizaram pequenas, médias e grandes aplicações, considerando sistemas comerciais, *open source* e mesmo sistemas construídos especificamente para o trabalho em questão. Quatro

destes artigos utilizaram profissionais como participantes, dois foram realizados com estudantes e apenas o estudo de correlação não usou participantes humanos. Exceto para o estudo de correlação, os demais artigos basearam sua análise sobre a atividade de inspeção de código. Analisando os resultados dos estudos deste subtema, foi identificada uma tendência de impacto dos dados demográficos sobre a avaliação humana de *smells*. O artigo de Mantyla *et al.* (2004) descobriu que os dados demográficos (conhecimentos, papel desempenhado, experiência de trabalho) pareciam explicar algumas das variações nas avaliações realizadas para *smells*. Esta conclusão apresenta-se semelhante à constatação apresentada pelo artigo de Mantyla e Lassenius (2006), onde existe a afirmação de que aspectos demográficos dos desenvolvedores afetam as avaliações de *code smells*. Os artigos de Carneiro (2010) e de Santos e Mendonça (2014) mencionam que as características humanas e os conhecimentos adquiridos em uma versão influenciam a avaliação humana em relação a *code smells*. O artigo de Yamashita e Moonen (2013a) considera o conhecimento sobre *smells* como um fator que afeta a avaliação humana.

Para os autores, quanto mais se conhece sobre *code smells*, mais preocupados ficam com a presença nos mesmos. Apenas o artigo de Mantyla (2005) não encontrou forte evidência do impacto de dados demográficos sobre a avaliação humana para *smells*, contatando que os dados demográficos não foram fortes evidências para a decisão de refatoração durante a avaliação de *code smells*. Outro fator abordado foi o domínio da aplicação. O artigo de Fontana *et al.* (2013) considerou o domínio da aplicação como um fator importante que deve ser incluído nas estratégias de detecção de *code smells*. Vale ressaltar que este foi o único estudo que menciona fatores que afetam a detecção de *code smells* e que não estão relacionados com dados demográficos.

Agreement on detection **(Concordância na detecção).** Seis dos vinte e seis artigos mencionam a respeito de acordo na detecção de *code smells* dentre os participantes. A grande maioria destes artigos está contida no conjunto de estudos que abordam fatores que afetam a detecção de *code smells*, ou seja, no subtema *Factors affecting detection* (apenas o artigo de Schumacher (2010) trabalhado neste subtema não aparece no subtema *Factors affecting detection*).

Devido a este fato, as características dos estudos são muito semelhantes: i) quatro realizaram experiências controladas, e dois realizaram *surveys*; ii) os estudos utilizaram sistemas de pequeno, médio e grande porte e; iii) usaram sistemas *open source*, comerciais e construídos especificamente para os experimentos; iv) quatro estudos tiveram como

participantes profissionais, enquanto os outros dois estudos usaram alunos como participantes; e v) todos os estudos adotaram inspeção código como a tarefa a ser realizada durante os seus respectivos experimentos. Com exceção do artigo de Mantyla e Lassenius (2006), que foi um *survey* questionando os participantes sobre a extensa variedade de *smells*, todos os demais estudos investigaram o nível de concordância/acordo sobre um pequeno conjunto de *smells* (três, no máximo, para cada artigo). Somente o artigo de Mantyla (2005) descobriu que há concordância parcial entre os participantes, sendo que todos os demais encontraram baixa concordância. Os artigos de Santos e Mendonça (2014) e de Schumacher (2010) explicitam realmente como baixo nível de concordância na detecção de *smells*, enquanto que o artigo de Carneiro (2010) chega a conclusão igual, mas apresentou a ideia de uma forma diferente. Para este estudo, os autores afirmaram que os participantes tiveram diferentes percepções e julgamentos durante a identificação de *code smells*. O mesmo aconteceu com o artigo de Mantyla e Lassenius (2006), onde seus autores descobriram que os desenvolvedores não têm uma única opinião sobre o "*smelliness*" do código fonte.

***Human evaluation versus software measures* (Avaliação humana *versus* medidas de software).** Este subtema foi abordado pelos artigos de Mantyla *et al.* (2004) e Mantyla e Lassenius (2006) que adotaram *survey* como método experimental. Ambos os estudos são baseados no mesmo sistema comercial, contando com a participação de profissionais nas pesquisas, incluindo desenvolvedores de software orientado a objeto. Os artigos mencionam como descoberta o fato das avaliações de *smells* por parte dos desenvolvedores não se correlacionarem com as métricas de software utilizadas.

Programming

Sete artigos abordaram aspectos relacionados com a programação de *smells*. Foram identificados dois subtemas relacionados: *smell removal*, *smell introduction*.

***Smell removal* (Remoção de smell).** Todos os sete artigos abordaram este subtema. Quatro deles são estudos de correlação e três deles são experimentos controlados, sendo dois deles com base na mesma configuração experimental. Os artigos relacionados a este subtema observaram as atividades de manutenção, os repositórios de software e atividades de inspeção de código. Dezenove *smells* foram trabalhados por esses estudos, o que tornou impraticável a generalização dos resultados encontrados. Apesar das dificuldades em generalizar resultados, observou-se que a remoção do smell não é sistematizada. A prova disto são os resultados apresentados por alguns desses estudos. O artigo de Yamashita e Counsell

(2013), por exemplo, descobriu que não há clareza sobre a intenção dos desenvolvedores em priorizar ou mesmo considerar a erradicação de *code smells*; o artigo de Peters e Zaidman (2012) descobriu que os desenvolvedores resolvem *code smells* por razões oportunas; e o artigo de Chatzigeorgiou e Manakos (2010) descobriu que os desenvolvedores realizam refatorações rotineiras com base na sua percepção subjetiva de áreas de código problemáticas em vez de aplicar soluções para identificados estes problemas.

***Smell introduction* (Introdução de *smell*).** Sobre a introdução de *smells* pode-se mencionar que, em alguns casos, na tentativa de remover determinados *smells*, outros *smells* são introduzidos ao código. Este fato foi reportado pelo artigo de Vauche (2009), onde os autores consideram que a correção de *God Class* em uma determinada classe do código pode mover o problema para uma classe diferente. O artigo de Silva (2013) descobriu que os desenvolvedores que revisam o código na maioria das vezes também introduzem mais *smells*. Os artigos de Chatzigeorgiou e Manakos (2010) e de Macia (2012b) apresentam discussões sobre quando *smells* são introduzidos. O artigo de Macia (2012b) descobriu que as anomalias de código induzem a anomalias arquiteturais, assim como o artigo de Chatzigeorgiou e Manakos (2010) descobriu que *smells* são introduzidos durante a concepção/implementação do código.

Detection

Dez artigos abordaram aspectos relacionados à detecção de *smells* em códigos. Foram identificados três subtemas relacionados: *supporting smell detection, smell as a predictor, novel Smell.*

***Smell as a predictor* (*Smell* como sinalizador de problemas).** Alguns estudos compararam estratégias de detecção de *smells* com outros sinalizadores de possíveis problemas no desenvolvimento de software. Cinco artigos abordaram este subtema. Dois experimentos controlados, dois estudos de correlação e um *survey*. Os dois experimentos controlados utilizaram a mesma configuração experimental. Quatro artigos utilizaram sistemas comerciais, e um artigo utilizou sistema open source. O tamanho dos sistemas variou de médio a grande. Os participantes eram profissionais, com exceção de um dos estudos de correlação, onde não foi considerada a avaliação humana. Mais uma vez, o principal problema em generalizar conhecimentos relacionados com as conclusões destes estudos é o elevado número de *smells* analisados, fechando um total de quarenta e oito *smells*. Além disso, outras questões foram consideradas. O artigo de Macia (2012a) focava em problemas de arquitetura

(estratégias de detecção são imprecisos na identificação de anomalias de código relacionadas a arquitetura); o artigo de Yamashita e Counsell (2013) abordou a relação entre as estratégias de detecção de *smells* e pareceres de peritos, apresentando resultados de diferentes perspectivas. Foi apresentando o julgamento dos peritos como sendo o método mais preciso para avaliar a manutenibilidade do sistema e foi mencionado que a abordagem de *code smells* pura pode detectar falhas de projeto que tenham sido negligenciadas por certos especialistas. Sendo assim, os autores concluem que a melhor opção para tentar garantir avaliações mais completas relacionadas à manutenção, seria a combinação de medidas de software com o julgamento dos peritos, não sendo suficiente apenas utilizar o conceito de *smell* para estas avaliações.

Outra perspectiva foi apresentada pelo artigo de Yamashita e Moonen (2012), onde os autores consideram que *code smells* podem abranger um espectro mais heterogêneo de fatores que não métricas de software e julgamento dos peritos individualmente. No entanto, eles entendem que a detecção de *smells* precisa ser complementada com abordagens alternativas, como a análise semântica e inspeção manual, por endereçar fatores de manutenção. Da mesma forma, o artigo de Khomh *et al.* (2009) considera *smells* como um sinalizador interessante de problemas no código, mas não como a estratégia mais interessante. Os autores deste artigo declararam que *smells* podem fornecer recomendações aos desenvolvedores mais fáceis de entender do que as métricas, mas não chegam a substituí-las. Principalmente quando pensamos na capacidade das métricas de tratar a propensão a mudanças (*change-proneness)* e gerar modelos de previsão.

***Supporting smell detection* (Apoiando na detecção de *smell*).** Quatro artigos abordaram a adoção de algum tipo de apoio para a detecção de *smell*. O contexto era bem definido, considerando médios e grandes sistemas, *open source* e comerciais; e usando profissionais como participantes. Todos eles utilizaram atividades de inspeção de código para a realização dos estudos. Dois deles realizaram experimentos controlados, um deles realizou um artigo de correlação e o outro adotou o *survey* como método empírico. Todos eles consideraram relevante, ou pelo menos, interessante, o uso de algum tipo de apoio complementar na detecção de *smells*. Foram destaques nestes artigos a definição de um processo (Schumacher, 2010), os recursos visuais adotados em Carneiro (2010), ferramentas para a detecção automática (Macia, 2012b) ou métricas de código fonte (Mantyla e Lassenius, 2006) como apoio complementar para a detecção de *smells*.

Other Correlations

Como não havia sido traçado previamente nenhum fator limitante, todos os resultados encontrados nos vinte e seis artigos selecionados foram extraídos e trabalhados.

Alguns destes resultados não estavam diretamente relacionados com o efeito do *smell* no desenvolvimento/manutenção de sistemas, o que ocasionou, obviamente, a tradução em códigos e temas com a mesma característica. Devido à ausência deste fator limitante, os mesmos foram extraídos, mas em reunião com o especialista em *code smells*, ficou decido que, considerando as questões de pesquisa e abordagem deste trabalho, os mesmos não seriam detalhados por não apresentam fatores significativos sobre o efeito do *smells*, foco da RS para a qual será realizada a síntese temática.

Dentro deste tema, os seguintes subtemas foram encontrados: *metrics versus smells* (estudos correlacionando *smells* e medidas de software), *smell density* (estudos sobre a relação entre o smell/linha de código), *frequency of smells*, *inter-smell relation*.

Na Tabela 21 são resumidos os resultados por tema e suas respectivas provas/evidências que conduziu a interpretação dos mesmos. Com isto busca-se aclarar o método utilizado tentando transmitir de forma transparente as considerações feitas ao longo desta dissertação. É importante notar que os resultados estão relacionados com o atual conhecimento sobre o efeito de *code smells*.

Tabela 21 Principais resultados por tema e suas respectivas evidências

Tema	Resultado	Evidências
Correlation with issues of development	*Smell concepts do not support evaluation of design quality, in practice.*	*i) divergent findings correlating smells and changes and defects; and common findings presenting negative correlations between.* *ii) smells and effort on maintenance activities; and* *iii) smells and architectural quality.*
Human aspects	*Human evaluation of smells should not be trusted.*	*i) all studies addressing the topic found low agreement on human evaluation; and studies tend agree that.* *ii) human evaluation of smells does not correlate with metrics; and* *iii) demographic data impacts smell detection.*
Programming	*There is not a systematized process in order to avoid/remove smells.*	*Convergence on findings of studies addressing smell removal and introduction.*
Detection	*The current smell detection strategies are not consistent with practical problems of the software development*	*i) the studies on the topic consider that some complement is necessary on smell detection; and* *ii) the most of studies have shown that the adoption of smells as a predictor is not trusted, not only considering human evaluation, but also using automatic detection heuristics.*

A.2.2 Mapa Temático

O mapa temático é o resumo gráfico de toda a condução do método aplicado. Este documento apresenta grande responsabilidade no que tange à apresentação sistemática dos resultados encontrados neste trabalho.

A análise e discussão apresentadas nos itens 1 e 2 deste Apêndice, como já citado anteriormente, também foram baseadas na observação deste Mapa Temático.

As Tabelas 22 a 32 representam uma visão do Mapa Temático criado para a síntese temática que norteou a aplicação prática do método refinado e proposto nesta dissertação. A visão apresenta várias partes do Mapa para uma melhor organização de sua apresentação, visto que existiam muitos temas, subtemas e respectivos códigos. A versão original utilizada foi criada na ferramenta Excel. No título das Tabelas mencionamos o tema ao qual os dados pertencem e os respectivos subtemas. Nas Tabelas separamos os códigos nas colunas de cada subtema aos quais os mesmos estão relacionados. Caso um código estivesse ligado a mais de um subtema, ele deveria ser repetido nas colunas específicas dos subtemas em questão.

Tabela 22 Mapa Temático - Tema: *Correlation with issues of development* – Subtemas: *Archictetural Quality* e *Harm*

Archictetural Quality		***Harm***	
1.278.1	The detection strategies are inaccurate in identifying architecturally-relevant code anomalies.	*3.419.1*	*The presence of God and Brain Classes is not necessarily harmful.*
1.278.2	The detection strategies are more effective in systems where architecture conformance is more strictly enforced in the code.	*3.419.4*	*Presence of God class and Brain class may be benefical to a system.*
1.278.3	Architecturally-relevant code anomalies often occurred in code elements responsible for implementing different architectural elements.	*3.60.3*	*God classes have been confirmed to have the technical debt property.*
1.278.4	The detection strategies seem to have a tendency to send developers in wrong directions when addressing code anomalies related to architectural modularity problem.		
1.278.5	The imperfection of the detection strategies (related to impact on architectural problems) is not simply related to specific thresholds or combinations of particular measures.		
1.278.6	The imperfection of the detection strategies is not simply related to specific thresholds or combinations of particular measures		
2.276.1	Refactoring strategies do not contribute to remove architectural-relevant code anomaly.		
2.276.2	There is a correlation between code anomaly and problems in architectural design.		
2.276.3	There is a correlation between code anomaly and architectural degradation.		
2.276.5	There is no specific code anomaly more relevant as indicator os architecture degradation.		
2.276.7	Architecturally-relevant code anomalies are left in the code when the system evolves.		

Tabela 23 Mapa Temático - Tema: *Correlation with issues of development* – Subtema: *Changes and Defects*

Changes and Defects			
1.38.3	*There are smells that are not associated with maintenance problems.*	*2.210.1*	*There is a correlation between maintanability factors and code smell.*
1.38.4	*Feature Envy and God Method togheter were associated with time-consuming changes.*	*2.260.1*	*There is no correlation between clones and bugs.*
1.38.5	*Changes of infected componentes with ISP Violation is significant larger*	*2.260.2*	*There is no evidence that clones are bad smells.*
1.63.1	*The role of code smells on the overall system maintainability is relatively minor.*	*3.334.1*	*Taken in isolation, classes exhibiting the identity disharmonies design flaws do not have an increased likelihood to exhibit defects that classes which do not reveal design flaws.*
1.63.12	*The overlap between problems related to code smells and problems related to artifacts associated to code smells can explain causes of inconsistencies in empirical studies about maintainability.*	*3.334.2*	*Classes exhibiting the identity disharmonies design flaws used by their clients shows the increase of likelihood for the clients to exhibit defects, especially for the post-release defects.*
1.63.13	*Some difficulties on maintenance activities are associated to a combination of smells and other characteristics, whereas others were not associated code smells at all.*	*3.429.1*	*The number of design problems (smells) increases as the system evolves.*
1.63.2	*Code smells can help to explain and potentially identify some specific difficulties occurred during maintenance beforehand.*	*3.461.2*	*Design flaws cant be considered more harmful with respect to software defects.*
1.63.7	*Feature Envy (which indicates efferent coupling) or ISP Violation (which indicates afferent coupling) can help to identify cross-cuting concerns situations.*	*3.461.3*	*In some systems there is no design flaw which strongly correlates with defects.*
1.697.1	*Some bad smells are associated significantly with class error proneness.*	*3.461.4*	*Some systems are characterized by one flaw is correlated with defects much more frequently than all the others.*
1.697.2	*The bad smells will not reveal all classes that contain errors.*	*3.60.1*	*The change likelihood of god classes is higher than the change likelihood of non-god classes.*
1.697.3	*Shotgun Surgery (and also God class and God method, in less intensity level) was the most bad smell that were associated with all severity levels of errors in all releases.*	*3.60.2*	*The defect likelihood of god classes is higher than the defect likelihood of non-god classes.*
1.9.2	*The change proneness of components with smells is higher than the ones without..*	*3.60.4*	*God classes impacts maintainability.*
1.9.3	*The code churn of infected classes with god class is significant larger.*	*4.522.1*	*Classes with code smells are more change-prone than others.*
1.9.4	*Code smells evolve during the system development.*	*4.522.2*	*Specific smells are more correlated than others to change-proneness.*
1.9.5	*Classes infected seem to need more maintenance than non-infected classes.*		

Tabela 24 Mapa Temático - Tema: *Correlation with issues of development* – Subtemas: *Effort* e *Design Quality*

Effort		***Design Quality***	
1.148.1	*Without any adjustments for file size and the number of changes smells were associated with more effort than files without these smells. (1.148.2)*	*1.38.7*	*The presence of a code smell may intensify or spread the effects of bad/limited design choices throughout the system.*
1.148.2	*After adjustments for file size and the number of changes code smells are not significantly associated with increased effort. (1.148.1)(1.148.3)(1.148.4)*	*4.522.3*	*Code smells often related to immature design and implementation.*
1.148.3	*File size and the number of changes explained almost all of the modeled variation in effort. (1.148.2)*	*1.63.6*	*Inconsistent design can be identified by detecting a combination of code smells.*
1.148.4	*There is a correlation between effort and code churn. (1.148.2)*		
1.148.6	*Effort may decrease by reducing file size and improving work processes to reduce the number of revisions, instead refactoring.*		
1.26.4	*God class don't require a higher maintainability effort than non-code smells.*		
2.10.3	*Software characteristics don't impact effort on god class detection.*		
3.429.4	*Maintenance effort should prioritize Long Method smells over other smells.*		

Tabela 25 Mapa Temático - Tema: *Human Aspects* – Subtemas: *Factors Affecting Detection* e *Decision Drivers*

Factors Affecting Detection		***Decision Drivers***	
2.10.1	*Human characteristic impact agreement, effort and drivers' choice on god class detection.*	*1.26.6*	*``Moderate agreement'' among the subjects detecting GC , for the misplaced method issue (also called driver).*
3.823.3	*Demographic data (knowledge, role, work experience) seemed to explain some of the variances in smell evaluations.*	*1.26.7*	*Misplaced methods is the strongest driver for letting the subjects classify a class as a god class.*
4.427.4	*The knowledge acquired in one version effectively supported the identification of related code smells in other versions.*	*1.26.8*	*Lack of cohesion and complexity are identified as issues that let the human subjects identify a class as a god class.*
4.747.2	*Demographics aspects of the developers affect the code smell evaluations.*		
4.747.5	*Developers with better knowledge of the module evaluated that there is more of the Lazy Class smell that is difficult to spot.*		
4.1509.2	*The demographics were not useful predictors neither for evaluating code smells nor the refactoring decision. (and Erradication)*		
5.1941.2	*The more details people actually know about code smells, the more concerned they are by the presence of smells in their code.*		
5.2000.2	*The domain of an analyzed system is an important factor that should be included in the detection strategies for various code smells*		

5.2000.6	*Smells exhibit similar behavior in different software domains.*	

Tabela 26 Mapa Temático - Tema: *Human Aspects* – Subtemas: *Human evaluation versus software measures* e *Agreement on detection*

Human evaluation versus software measures		**Agreement on detection**	
3.823.2	*Developers' evaluations of the smells do not correlate with the used source code metrics.*	*1.26.2*	*The agreement on GC classificationand detection is low.*
3.823.5	*Developers' evaluations on the Large Class smell seem to be conflicting, when compared to large class measures.*	*2.10.2*	*The agreement on god class detection is low.*
4.747.3	*The developers' evaluations on code smells dont correlate with related source code metrics.*	*3.823.1*	*Software developers show low agreement about "smelliness" of the source code.*
4.747.6	*The developers' evaluations of the smells correlated better with the metrics for smells that are simple and easy to spot.*	*4.427.2*	*The participants had different perceptions and judgments during the code smells identification.*
		4.747.1	*Developers have not a uniform opinion on the "smelliness'' of the source code.*
		4.1509.1	*There is partial concordance among the evaluators in all evaluations.*

Tabela 27 Mapa Temático - Tema: *Human Aspects* – Subtemas: *Knowledge about smells, Detection difficulty* e *Strategy on detection*

Knowledge about smells		**Detection difficulty**		**Strategy on detection**	
5.1941.1	*There is not strong understanding about code smells and practical application of these concepts.*	*1.26.1*	*The subjects perceive detecting god classes as an easy task.*	*4.427.3*	*Participant with optimistic style investigate code smell candidates using more combinations of views when compared to participants with more conservative styles.*
5.1941.3	*Duplicated code was by far the most mentioned smell.*				
5.1941.6	*Respondents who were extremely or moderately concerned about smell benefits gave as rationale reasons like product evolvability, end-product quality, and developer productivity*				

Tabela 28 Mapa Temático - Tema: *Programming* – Subtemas: *Smell removal* e *Smell introduction*

Smell removal		**Smell introduction**	
1.148.5	*The developers who revised most often also introduced the most smells.*	*1.148.5*	*The developers who revised most often also introduced the most smells.*
1.62.7	*There is no clarity on the intent of developers prioritize or even consider the eradication of code smells.*	*2.276.6*	*Early code anomalies induces architectural anomalies.*
2.276.7	*Architecturally-relevant code anomalies are left in the code when the system evolves.*	*3.429.2*	*Smells are introduced during the intial design/implementation.*
2.277.1	*The developers resolve code smells for opportunistic reasons.*	*4.526.1*	*A large proportion of GCs seems to be introduced as a conscious design decision by developers.*
2.277.2	*Code smells of early revisions are resolved whithin a few revisions.*	*4.526.4*	*Changes can result in the degradation of GCs.*
3.429.3	*Designers perform refactorings routinely based on their subjective perception of problematic code areas rather than applying them as solutions to identified design problems.*	*4.526.3*	*The correction of a GC may also move the problem to a different class*
4.526.2	*Specific maintenance activities can eliminate GCs when they are accidents.*		
4.526.3	*The correction of a GC may also move the problem to a different class*		
4.1509.2	*The demographics were not useful predictors neither for evaluating code smells nor the refactoring decision.*		

Tabela 29 Mapa Temático - Tema: *Detection* – Subtemas: *Supporting smell detection* e *Novel smell*

Supporting smell detection		**Novel smell**	
1.26.3	*Automatic detection accompanied by a manual review increases the overall confidence in the results of metric-based classifiers.*	*1.63.8*	*A new smell was propose: Multiple Inheritance simulation.*
1.26.5	*If provided with a suitable process, humans can detect code smells in an effective fashion.*		
2.276.4	*Developers need support to enhance detection of code anomaly.*		
4.427.1	*The use of the visual representation of concerns in the multiple views approach provided useful means to visually spot some kinds of code smells more than others.*		
4.747.4	*Using source code metrics in conjunction with human evaluations is likely to be the best alternative.*		
5.1941.4	*Software professionals who are interested on code smells and anti-patterns expressed a need for better support, including tools, during the software evolution cycle.*		
5.1941.5	*Respondents indicated that they use technical blogs, programmer forums, colleagues and industry seminars as their main sources of information.*		
5.2000.3	*Accuracy of code smells detection rules used by the tools can be improved by exploiting knowledge about both the domain and the design of a system.*		

Tabela 30 Mapa Temático - Tema: *Detection* – Subtemas: *Smell as a predictor*

Smell as a predictor			
1.278.1	*The detection strategies are inaccurate in identifying architecturally-relevant code anomalies.*	*1.63.4*	*A combination of different code analysis techniques and tools is needed in order to achieve a more comprehensive evaluations of maintainability.*
1.278.3	*The detection strategies are more effective in systems where architecture conformance is more strictly enforced in the code.*	*2.210.2*	*Code smells would need to be complemented with alternative approaches such as semantic analysis and manual inspection for address maintainability factors*
1.278.5	*The detection strategies seem to have a tendency to send developers in wrong directions when addressing code anomalies related to architectural modularity problem.*	*2.210.3*	*Code smells can cover a more heterogeneous spectrum of factors than software metrics and expert judgment individually.*
1.62.4	*Expert judgment is most accurate method for assessing system maintainability.*	*3.823.4*	*The use of the smells for internal software quality assessment should be questioned*
1.62.5	*Code smells approach can detect design flaws that may be overlooked by experts.*	*4.522.4*	*Smells can provide to developers recommendations easier to understand than what metric profiles can do.*
1.62.6	*Software measures and expert judgment addressed different aspects of maintainability in a system, and combining them can lead to more complete evaluations of maintainability.*	*4.522.5*	*Smells are not replacement to metrics in the ability of building change-proneness or fault-proneness prediction models*
1.63.3	*Complementary approaches (besides smells) are needed to achieve more comprehensive assessments of maintainability.*		

Tabela 31 Mapa Temático - Tema: *Other Correlations* – Subtemas: *Inter Smells relation*, *Smell density* e *Frequency of smells*

Inter Smells relation		**Smell density (smell/LOC)**		**Frequency of smells**	
1.38.1	*There is an interaction between code smells.*	*1.62.2*	*Code smell density is likely to be inaccurate when comparing size differing systems.*	*3.419.2*	*The larger systems have a larger proportion of God and Brain Classes.*
1.38.2	*There is a correlation between inter-smells smells and maintenance problems.*	*1.62.3*	*Code smell density distinguishes maintainability across similar sized systems.*	*3.461.1*	*Some design flaws are more frequent than others.*
1.38.6	*Studies into the effects of inter-smell relations are a topic that deserves more attention.*	*3.419.3*	*Size normalizing impacts evaluation of change frequency of God class and Brain class.*	*3.461.5*	*Feature envy is the most frequent design flaw, but it is not the most correlated with software defects.*
1.63.10	*Observing combinations of code smells could be useful to discriminate instances of God Classes that are potentialy problematic, against instances of no problematic god class.*	*3.60.5*	*There isn't a linear relationship between class size and change likelihood that justifies LOC-normalization.*	*5.2000.1*	*Some smells are more prevalent than others.*
1.63.13	*Some difficulties on maintenance activities are associated to a combination of smells and other characteristics, whereas others were not associated code smells at all.*				
1.63.5	*Interaction effects amongst collocated smells and coupled smells should be taken into account during analysis.*				
1.63.6	*Inconsistent design can be identified by detecting a combination of code smells.*				

Tabela 32 Mapa Temático - Tema: *Other Correlations* – Subtemas: *Metrics versus smell*

Metrics versus smell			
1.26.9	*Automated metric-based pre-selection of smells decreases the effort spent on manual code inspection.*	*1.9.1*	*There is a correlation between size and number of: god class and shotgun surgery.*
1.62.1	*Code smells are strongly influenced by size (LOC).*	*4.1509.3*	*Simple code smells and source code metrics have a relationship.*
1.63.11	*Wide coupling displayed Feature Envy or ISP Violation and to some extent Shotgun Surgery.*	*5.2000.4*	*There is correlation between several code smells and selected metrics.*
1.63.9	*Large size exhibited either the God Class smell or the God Method smell and in most cases a combination of both smells.*	*5.2000.5*	*The strength of correlation between code smells and metrics varies with respect to the domain*

A.3. DISCUSSÃO

A.3.1 Sobre os Resultados Percebidos para Code Smells

Com base nos estudos analisados e resultados encontrados, torna-se clara a existência de alguns desafios que precisam ser trabalhados para que torne possível a efetiva melhora na avaliação do efeito de *smells* no desenvolvimento/manutenção de software. Estes desafios devem ser enfrentados para entender melhor por que alguns estudos têm mostrado que o conceito de *smells* não se correlaciona com problemas de desenvolvimento de software, como esperado a partir de discussões teóricas. Os desafios são:

Agrupar *smells* de acordo com a sua natureza. A primeira observação feita foi relacionada ao grande número de *smells* abordados nos estudos. Isso aconteceu em todos os temas identificados, tornando a generalização das descobertas mais difícil. Parece necessário compreender melhor a natureza dos *smells*, agrupando-os de acordo com os interesses de análise. Apenas Mantyla e Lassenius (2006) reportam um tipo de agrupamento de *smells*, porém este ainda não foi fortemente validado por outros estudos.

Delinear aspectos contextuais. Em alguns casos, a dificuldade em generalizar os resultados deu-se devido aos aspectos relacionados ao contexto amplo que foi considerado nos estudos. Isto ocorreu, por exemplo, na análise de estudos abordando aspectos humanos.

Enquanto alguns estudos adotaram software de pequeno porte, outros estudos adotaram software de grande porte, dificultando a generalização. O mesmo ocorreu para os participantes nos estudos. Enquanto alguns estudos foram realizados com alunos, outros estudos foram realizados com os profissionais. Quando estes aspectos contexto estiverem bem delineados, problemas reportados anteriormente que dificultam a generalização poderiam ser evitados. Um exemplo claro para ilustrar este item seria a quantidade de *smells* trabalhados no contexto do subtema *Change and Defects*. Este poderia ser um ponto de partida para

compreender o efeito de *smells* considerando um contexto específico. No entanto, mostra-se notória a necessidade de mais estudos para delinear melhor o contexto relacionado aos efeitos do smell.

Diminuir a subjetividade relacionada à avaliação de *Smells*. A maioria dos estudos que abordam avaliação humana (comparando com métricas, ferramentas de detecção automática, ou investigando acordo sobre a detecção de *smells*) mostrou que a avaliação humana não deve ser algo confiável. Em geral, estes estudos concentraram-se em *smells* específicos, principalmente *god class*. Note-se que a avaliação de *god class* é mais fácil do que a avaliação de outros *smells*, como *feature envy* ou *shotgun surgery*, uma vez que envolve a observação de um menor número de classes (isto está relacionado com a investigação da natureza dos *smells* proposta como o primeiro desafio). Isso reforça a afirmação de que a avaliação humana não deve ser confiável, mas também levanta dificuldades em realizar estudos na área. Diante do exposto, duas estratégias podem ser seguidas: i) aprofundar a avaliação dos fatores humanos que afetam a detecção de *smells*; e ii) avaliar os aspectos cognitivos na detecção de *smells*, que está relacionado com a compreensão do programa, e requer conhecimento, tanto em ciência da computação e psicologia cognitiva. Esta questão é chamada de problema conceituação de *code smells* e vem sendo discutida (Santos, 2014; Santos e Mendonça, 2014; Santos e Mendonça, 2015).

Aumentar o número de estudos na área. Alguns subtemas foram abordados em apenas um pequeno número de estudos. Por exemplo, a qualidade arquitetural foi abordada apenas por dois estudos. Esforço foi abordado por quatro estudos, mas apenas dois focados em correlacionar esforço e atividades de manutenção. Isso também aconteceu com outros subtemas. A necessidade de estudos mais empíricos na área também é reportada em outros estudos (Mantyla *et al.*, 2004; Schumacher, 2010; Silva, 2013).

ANEXO 1 – CODE SMELL EFFECT: A THEMATIC SYNTHESIS

Code smell effect: a thematic synthesis

José Amancio M. Santos[a], Luciana Carla Lins Prates[c], Rogeres Santos do Nascimento[c], Mydiã Falcão Freitas[c], Manoel Gomes Mendonça[b,c]

[a]*Technology Department State University of Feira de Santana, Bahia, Brazil*
[b]*Fraunhofer Project Center for Software & Systems Engineering at Federal University of Bahia, Brazil*
[c]*Computer Science Department, Federal University of Bahia, Bahia, Brazil*

Abstract

Context: *Code smell* is a term commonly used to describe potential problems in the design of software. The concept is well accepted by the software engineering community. However, some studies have presented divergent findings about the practical usefulness of the code smell concept. These studies, taken in isolation, do not represent the knowledge about the subject. The area lacks an overall common understanding of the concept. **Objective**: To synthesize current knowledge related to the practical usefulness of the smell concept. We based this synthesis on empirical studies investigating how smells impact the software development, which we call the *code smell effect*. **Method**: A systematic review about the smell effect is carried out. We extracted, coded and grouped the findings of the primary studies in themes, considering the context, also extracted from these studies. We adopted thematic analysis as the main method of synthesis. **Result**: We noted that the smell concept does not support the evaluation of quality design in practice. There is no strong evidence correlating smells and some important software development attributes, such as effort in maintenance. We also noted that primary studies converge in pointing out that agreement on smell detection is low and demographic data, such as developers' experience, significantly impacts the evaluation of smell. Consequently, we conclude that human evaluation of smells should not be trusted. **Conclusion**: In order to improve future analysis on the subject, we consider that the area needs to better outline: i) factors affecting human evaluation of code smells; ii) contextual aspects; and iii) the nature of different types of code smells, grouping them according to relevant characteristics.

Keywords: Code smell, systematic review, thematic synthesis

1. Introduction

Quality of object-oriented design has been widely addressed in Software Engineering. Terms such as "design flaws" [1], "code smell" [2] and "disharmony" [3] are widely used to define potential design problems. In this paper, we adopt the term *code smell*, or simply *smell*, to refer to design problems.

The smell concept is well accepted as indicative of potential design problems. However, some empirical studies have refuted this idea, under certain circumstances [4, 5, 6]. For example, Sjøberg et al [4] investigated the relationship between smells and maintenance effort. They noted that none of the investigated code smells were significantly associated with increased maintenance effort. Macia et al. [5] investigated the relationship between code smells and problems that occur with an evolving system's architecture. In their study, they noted that many of the detected smells were not related to architectural problems. In [7], Yamashita summarizes a wide analysis based on the same experimental setup used by Sjøberg et al. [4]. One of her findings is that *"aggregated code smells are not so good indicators of system-level maintainability"*.

Email addresses: zeamancio@ecomp.uefs.br (José Amancio M. Santos), llins@dcc.ufba.br (Luciana Carla Lins Prates), rogeres19@gmail.com (Rogeres Santos do Nascimento), mydiaff@dcc.ufba.br (Mydiã Falcão Freitas), mgmendonca@dcc.ufba.br (Manoel Gomes Mendonça)

In fact, there is no strong evidence linking code smells with problems arising from bad design. Or at least, the question is not yet well understood. We highlight some claims reinforcing this idea. Zhang et al. [8] claimed that *"...we do not know whether using code bad smells to target code improvement is effective"*. Sjøberg et al. [4] claimed that *"the present focus on bad design as operationalized by code smells may be misdirected"*. The reasons for this has not been well explored. This is a complex task because it could be related to: i) comprehension of cognitive aspects on the human perception of smells; ii) technical and demographic aspects affecting human perception of smells; iii) investigation of the quality and context aspects considered in empirical studies; and iv) effectiveness of the smell detection heuristics adopted in the studies.

The area lacks discussions about why empirical studies fail to prove the well accepted idea that smells represent potential design problems. In this work, we attempt to synthesize current knowledge based on the empirical studies addressing the practical adoption of smells. Our synthesis reveals a gap between the theory and the empirical results. We aim to discuss three main aspects: i) how the empirical studies investigate the practical use of smells; ii) identification of the context aspects considered by the empirical studies; and iii) the convergence/divergence of the findings of these studies.

To do this, we carried out a systematic review (SR) as pro-

posed by Kitchenham [9]. One of the main difficulties of our SR is that findings are presented as qualitative data in papers. In order achieve our objective, we had to code the findings and to group them according to the context aspects. From the synthesis methods available [10], such as thematic analysis, grounded theory or meta-ethnography, we adopted the thematic analysis, following recommendations proposed by Cruzes and Dybå [11]. According to them, "thematic analysis is often used for identifying, analyzing and reporting patters (themes) within data in primary qualitative research".

We highlight the two main findings of our SR. The first is that the smell concept does not support the evaluation of design quality in practice. Our evidence is the divergent results from the primary studies of our SR. Moreover, the studies did not find correlations between smells and effort in maintenance activities, and between smells and architectural quality. The other important finding was that human evaluation of smells should not be trusted. In this case, the evidence is the studies showing that agreement on human evaluation of smells is low. They also highlight the relevance of demographic data, such as developers' experience, on smell detection.

It is important to note that our findings are related to current knowledge on the topic. According to our investigations, it is not possible to establish a categorical correlation between smells and software development issues nowadays. However, we identified some challenges in order to improve future analyses on the subject. We consider that studies in the area need to better outline i) factors affecting human evaluation of smells; ii) contextual aspects; and iii) the nature of different types of smells, grouping them according to relevant characteristics.

The structure of the rest of this paper is as follows. Section 2 introduces main concepts as the basis of our work, highlighting the method we adopted and the code smell as the central concept of our work. Section 3 presents prior empirical studies addressing code smell. Section 4 and 5 present the protocol and the data extraction process of our SR. Section 6 and 7 present results and a discussion of them. Section 8 discusses the threats to the validity of the SR. Lastly, Section 9 presents our conclusions and proposes future works.

2. Background

In this section, we characterize the empirical method that we adopted by briefly describing systematic reviews and thematic synthesis. We also present code smell as the central concept of our work.

2.1. Systematic reviews and thematic synthesis

2.1.1. Systematic review

A SR is "a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest" [9]. Kitchenham et al. [12] showed that SR is increasingly being used by the software engineering community. The method was originally proposed to support evidence-based medicine and it was adapted to software engineering research, mainly by Kitchenham and Charters [9]. Brereton et al. [13] discuss the process of adaption to software engineering.

In their technical report, Kitchenham and Charters [9] present a guideline with the steps for SR adapted for software engineering. Wohlin et al. summarize these steps. We present them as follows:

1. Planning
   - *Identification of the need for a review*. A SR originates from aiming to understand the state-of-art in an area.
   - *Specifying the research questions*. The research questions steer the researchers in the identification of primary studies, the extraction of data from the studies, and the analysis.
   - *Developing a review protocol*. The review protocol defines the procedures for the SR, involving research questions, search strategy for primary studies, data extraction strategies and other issues.
2. Conducting the review
   - *Identification of research*. The main activity in this step involves specifying search terms and applying them to databases. Other aspects that might be considered are manual searches in journals and proceedings and the use of "snowballing" [15].
   - *Selection of primary studies*. This involves the definition of inclusion/exclusion criteria and a strategy of selection which might consider the reading of titles and abstracts or a more thorough analysis of other sections, such as methodology or conclusions.
   - *Study quality assessment*. There is no systematized strategy proposed for quality assessments of primary studies. However, checklists are the most practical strategy used.
   - *Data extraction and monitoring*. The definition of the data extraction form to collect the information needed from the primary study reports.
   - *Data synthesis*. This may involve quantitative and/or qualitative analysis. The most advanced form of data synthesis for quantitative data is meta-analysis. For qualitative analysis, different approaches are presented [16]: thematic analysis, narrative synthesis, comparative analysis, case survey, meta-ethnography, meta-analysis (based on statistical methods to integrate quantitative data from several cases) and scoping analysis. We will discuss thematic analysis as the method we adopt in our SR in the next subsection.
3. Reporting the Review
   - *Observing the audience*. The discussion about reporting the review involves observation of the audience. For practitioners, different publishing sources might be used, such as practitioner-oriented journals

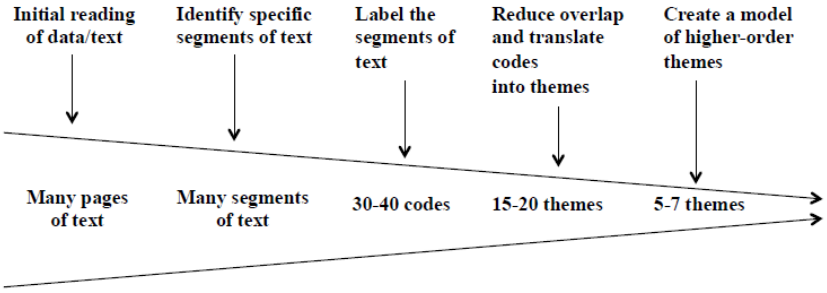


Figure 1: Thematic synthesis process extracted from [11] (Cruzes and Dyba adapted the figure from [17])

and magazines, posters or web-pages. For the academic audience, the detailed reporting of procedures is required.

2.1.2. Thematic synthesis

As previously discussed we adopted thematic synthesis as presented by Cruzes and Dybå [11]. They define thematic synthesis from the thematic analysis method as follows:

> "Thematic analysis is an approach that is often used for identifying, analyzing, and reporting patterns (themes) within data in primary qualitative research. 'Thematic synthesis' draws on the principles of thematic analysis and identifies the recurring themes or issues from multiple studies, interprets and explains these themes and draws conclusions in systematic reviews."

Cruzes and Dybå conceptualize the thematic synthesis analysis in five steps summarized in Figure 1.

The steps are detailed and an extensive set of recommendations are presented:

1. *Data extraction*. This involves extraction of data from primary studies, including bibliographical information, aims, context and results.
2. *Data coding*. This involves systematic identification and coding of interesting concepts, categories, findings and results.
3. *Translation of codes into themes*. This might consider themes, sub-themes and high-order themes.
4. *Creation of a model of higher-order themes*. This discusses exploring the relationship between themes and/or sub-themes.
5. *Assessing the trustworthiness of the synthesis*. This involves the interpretation process leading to the thematic synthesis.

2.2. A brief history of the code smell concept

The main concept we address in this work is code smell. As discussed in Section 1, we are using the code smell as a "design flaw" and " disharmony" indistinctly. All these terms are used to refer to design problems. Basically, the authors propose strategies for the identification of aspects in the code that break the principles of the object-oriented paradigm. In one of the first books on the topic, Riel [1] used his experience to discuss common problems observed. He addressed the misuse of relationships between classes, inheritance, the containment relationship from classes and attributes, and others. After each discussion, he presented heuristics to avoid the problems. For example, related to the misuse of multiple inheritance, one of Riel's suggestions is *"if you have an example of multiple inheritance in your design, assume you have made a mistake and then prove otherwise"*.

Another book in the area was by Fowler [2]. He focused on discussions about refactoring. He defined refactoring as *"the process of changing a software system in such a way that it does not alter the external behavior of the code yet improves its internal structure"*. The main idea is improving the internal structure (design), avoiding future problems, especially with regard to maintenance. He named and presented refactoring techniques for many situations, such as "extract method", "move method", "replace data value with object", and many others. Due to the fact that his presentation was based on a step-by-step format, including pieces of code in his examples, in some cases it is possible to apply the techniques automatically

However, the most interesting discussion proposed by Fowler is related to *when we apply refactoring?*. Rather than *how we apply refactoring?* Fowler himself classifies *how to apply refactoring* as a simple problem and *when to apply refactoring* as a "not so cut-and-dried" problem. He suggested that until then, a "vague notion of programming aesthetics" had been commonly proposed and he wanted "something a bit more solid". Then, he coined the term code smell to describe potential design problems. He presented 22 bad smells, from an informal discussion. He also recommends the techniques of refactoring to eliminate code smells.

In both Riel [1] and Fowler's [2] books, briefly discussed above, the characterization of code smells is subjective. This was one of the motivations for Lanza and Marinescu's [3] work. Their strategy of code smell detection is based on metrics and thresholds. To define their rules, Lanza and Marinescu based on the informal definitions of Riel and Fowler, identifying metrics that could be affected by the code smell. Then, they defined the thresholds and finally, they composed the metrics and thresholds.

For example, the code smell god class refers to classes that tend to centralize the intelligence of the systems [1]. Lanza

and Marinescu elected the metrics ATFD (Access To Foreign Data), WMC (Weighted Method Count) and TCC (Tight Class Cohesion)[1]. The composition is presented in the Figure 2. The figure shows a class is a god class if it has ATFD > *FEW*, and WMC ≥ *VERY HIGH* and TCC < *ONE THIRD*. The threshold values *FEW* and *VERY HIGH* are previously defined according to other attributes of the system, such as size. If the metrics and thresholds are well defined, then the code smell can be automatically detected.

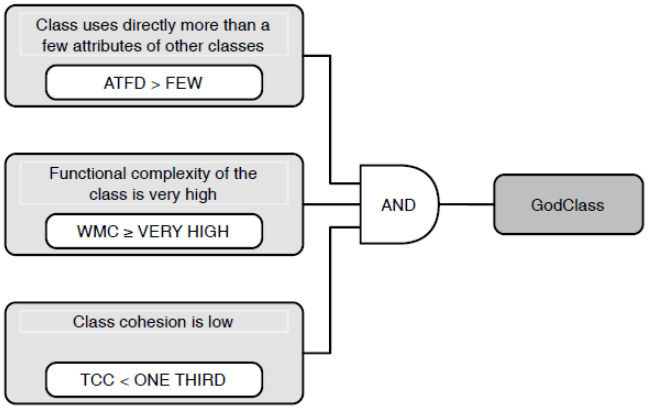


Figure 2: Detection strategy for god class [3]

The work of Lanza and Marinescu is the basis of much research and many tools which focus on the automatic detection of smells.

3. Related work

In this section, we present the types of empirical studies addressing smells. Then, we present some studies revealing problems in the evaluation of the practical use of smells, in order to define the scope of our SR well.

3.1. Types of empirical studies addressing smells

As the use of the code smell concept has become widespread, empirical studies have been carried out to help understand its effect. In our literature review, we found three types of studies:

1. **Correlation studies**. Studies trying to establish a correlation between a code smell and some quality attribute of the software, such as number of bugs or number of modifications in classes. They commonly evaluate the impact of smells on data extracted from software repositories [6, 18, 19, 20].
2. **Role of human**. This type of study investigates factors related to human aspects and their effect on code smell detection. They involve humans in tasks related to code smell detection, and they analyze agreement, decision drivers and inconsistencies in their answers [21, 22, 23, 24].
3. **Tool assessment**. This type of study mainly focuses on automatic detection tools [23, 25] or software visualization [26, 27, 28, 29]. Tools for code smell detection are based on metrics which can produce imprecise results due to inconsistent heuristics, as noted by [30]. Software visualization has emerged as an alternative to address smell detection. Software visualization is the use of visual paradigms (visual resources, graphic design or animation) to facilitate both the human understanding and effective use of software [31]. Software visualization tools are also based on metrics. However, visual resources combined with metrics may help humans to identify design problems.

In our SR, we are interested in studies evaluating how smells affect software development, which we call the *smell effect*. As a result, we only consider the correlation studies because their results represent knowledge on the application of smells in practice, and the studies addressing the role of humans because they demonstrate problems in the evaluation of smells. We disregarded tool assessment studies, because their findings are related to the effectiveness of the tools. For us this is not related to smell effect evaluations.

3.1.1. Secondary study on code smell

The only secondary study that we found on the subject is a mapping study by Zhang et al [8]. One of their motivations was the absence of knowledge about the effectiveness of the code smell concept to target code improvement, which is similar to our motivations. They reviewed 319 papers from 2000 to 2009 and analyzed 39 of those in detail. Despite having similar aims, the studies have different characteristics. While Zhang et al. performed a systematic mapping study, performing a broad, but not particularly deeply analysis, we did the opposite: a more focused and deeper analysis. We highlight differences in the data sources, search terms and analysis strategy. Zhang et al. adopted specific journals as sources. The only search engine they used was IEEE Xplorer. We adopted two well-known publishers (IEEE Xplorer is one of them) and two well-known indexers. Related to search terms, Zhang et al. limited their study to the concept of code smell. They did not adopt similar terms, such as “anomaly”, “design flaw” or “disharmony”. However, they adopted names of some specific smells, which we did not adopt. A further difference in search terms is that we considered only empirical studies focusing on the smell effect. When Zhang et al. presented their mapping study, they did not discuss the smell effect at length, rather, they presented a broad mapping of the area. As for the analysis strategy, they based their study on quantitative data, while we performed a thematic synthesis which focused on qualitative data. We detail our study protocol later on.

Zhang et al. examined four research questions:

1. *Which Code Bad Smells have attracted the most research attention?* For this question, the answer was *duplicated code*. They found this code smell in 21 papers. They also noted that some smells have received little attention from researchers.

[1] We do not detail the metrics because this is beyond the scope of our work.

2. *What are the aims of studies on Code Bad Smells?* They identified that most studies focus on identifying code smells as opposed to investigating their impact. As the study addressed broad aspects of code smell, they selected papers addressing topics as wide as tools, refactoring, and heuristics for code smell detection. As a result, they do not examine any of these topics in depth.
3. *What methods have been used to study Code Bad Smells?* They suggested that the focus has been on objective studies and there are few subjective studies. They consider this a gap in code smell research.
4. *What evidence is there that Code Bad Smells indicate problems in code?* Their findings indicate that the impact of code smell is not well understood. They noted that researchers concentrate on investigating how to identify Code Bad Smells rather than examining their impact in practice.

3.1.2. Other empirical studies

In this section, we present some empirical works showing inconsistencies on smell effect evaluation in order to reinforce the context of our SR. Of these, only the work of Palombra et al. [32] is not a primary study in our SR because it is outside the range of years that we considered.

Mäntylä and Lassenius [22] and Mäntylä et al. [33] investigated agreement and the impact of demographic data, such as experience of developers, on smell detection by humans. Moreover, they compared the results of human evaluation with metric-based heuristics. They defined the heuristics by themselves because research on heuristics for smell detection was incipient at the time. Using a survey, they asked participants about 23 smells and used a scale from one (lack) to seven (large presence) to evaluate the presence of smells in a piece of code. They received 12 completed questionnaires from 18 sent to developers in a small software company. They found that some demographic variables, such as developer experience, partly explain the variation. Regarding the correlation of human evaluation and the metric-based heuristics, they analyzed only four smells: *large class*, *long method*, *long parameter list*, and *duplicate code*. For *long method* and *long parameter list*, human evaluation correlated well with the metrics. For *large class* and *duplicate code*, they did not perceive the correlation as well. We highlight two main findings bringing up the perception of difficulties on the practical adoption of smell concepts in software development. In one of the findings the authors affirm that: "*the use of smells for code evaluation purposes is hard due to conflicting perceptions of different evaluators*". The authors also consider that "*the use of the smells for internal software quality assessment should be questioned*".

Olbrich et al. [6] investigated the influence of two smells (*god class* and *brain class*) on the frequency of defects. They analyzed historical data, adopting Lanza and Marinescu's detection strategies [3], from three open-source software systems. They found that without taking the class size into account, there is a higher change frequency, change size, and defects rate in *god* and *brain classes* than in other classes. However, when these smells were normalized with respect to size, they had smaller values for change frequency, change size, and weighted defect rate than other classes had. Based on these findings, they concluded that in specific cases the presence of these smells might be beneficial to a software system.

Sjøberg et al. [4] presented a controlled study in which six professionals were hired to maintain four systems for 14 days. All the systems were developed based on the same specification, as a result of a tender sent by Simula Labs. It was a medium-size web-based information system to keep track of their empirical studies. Their aim was to quantify the relationship between code smells and maintenance effort. They focused on 12 code smells and how they affect maintenance. The heuristics for smell detection adopted were the ones presented by [3] and by [34]. One of the main findings of the study was that *"the 12 smells appear to be superfluous for explaining maintenance effort"*. In this context, if we consider that one of the main effects of bad design is the impact on maintenance activities, the presence of code smells might not represent bad design. The authors stated that "*the present focus in research and industry on 'bad design' as operationalized by code smells may be misdirected*".

Palombra et al. [32] investigated if what developers believe to be a problem is actually a problem. They manually identified instances of 12 smells in three open source systems. Then they sent a questionnaire to potential participants, asking about code snippets affected and not affected by smells. When the answer was positive, the participant was asked to explain the problem and assess its level of severity. Three types of participants were considered in the study: i) graduate students; ii) industrial developers and; iii) the developers of the systems themselves. They received answers from 15 graduate students, nine industrial developers, and ten of the original developers. Their findings show that some smells are generally not perceived as a design problem: *class data should be private*, *middle man*, *long parameter list*, *lazy class*, and *inappropriate intimacy*. They also noted "*instances of a bad smell may or may not represent a problem based on the 'intensity' of the problem*".

4. Review protocol

In this section, we present the definitions of the research questions, data sources, search terms, inclusion/exclusion criteria and quality assessments used for the identification of relevant primary studies of our SR.

4.1. Research questions

As previously discussed, our SR focuses on synthesizing current knowledge on how the smell concept has been empirically investigated. We defined the following research questions in order to steer our research:

1. What aspects (themes) are studied in smell effect investigations?
2. How similar/different is the context of studies investigating smell effect?
3. Are the findings converging?

4.2. Data sources, search terms and range of years

We adopted electronic data sources (EDS) in our SR. The EDS and search terms were defined from an iterative process, by refining terms related to our aims. We also considered the range of years and the volume of papers retrieved in different EDS. Both the choice of EDS and the definition of search terms were based on our previous ad-hoc literature review and on our experience on the topics. The process consisted of three meetings with four researchers in the area.

At the first meeting, one of the researchers (the first author) presented results of a preliminary search, considering two ranges: from 2006 to 2013, and from 2002 to 2013). After discussion, terms and EDS were added/removed. At the second meeting new search results were presented. Here we decided, for example, to use *"code anomaly"* and *"design anomaly"* as some of our terms, instead *anomaly*, in order to minimize the number of empirical studies from other engineering subjects, such as chemistry and electronics. We also decided to use the term *controlled* rather than *"controlled experiment"* because we did not notice a significant impact on the results . At the third meeting, we evaluated the new search, refining some minor points and defining the final set of EDS, the search terms and range of years of our SR

The final range of years that we considered was from 2002 to 2013 because we noticed that the number of retrieved papers was feasible. Moreover, we considered twelve years as a reasonable range of years to cover discussions about the smell effect. We detail the choices of EDS and search terms below.

4.2.1. Data sources

Secondary studies in software engineering have adopted an extensive set of EDS, such as IEEE Xplore, ACM Digital library (ACM DL), Springer Link, Google Scholar, ISI Web of Science, Science Direct, Scopus, Kluwer Online. In order to guarantee a broad range of potential primary studies, we decided to select well accepted EDS by the software engineering community, considering coverage, advanced features and exportability. From the meetings described previously, we decided on two well-known publishers[2], which were IEEE Explore and ACM DL; and two relevant indexers[3], which were Scopus and Science Direct.

Using the advanced resources from each EDS, the search was done on meta data (title, abstract and keywords). This decision was based on our experience, the volume of papers retrieved and recommendations by Dieste and Pádua [35]. The IEEE Xplorer can filter by type of source so we filtered for journals and conferences. In the ACM DL, we filtered the search for journals, proceedings and transactions. Finally, in Socpus and Science Direct, we filtered the results for knowledge sub-area. In Science Direct, we chose Computer Science and Engineering. In Scopus, we adopted the same sub-areas, adding Multidisciplinary.

[2]http://ieeexplore.ieee.org/Xplore/home.jsp and http://dl.acm.org/

[3]http://www.scopus.com/ and http://www.sciencedirect.com/

4.2.2. Search terms

We considered terms from two main topics. At first, we search for studies describing potential problems in the design of software. As previously discussed, important books coined the well-known terms *smell*, *design flaw* and *disharmony*. We also considered the use of the term *anomaly*, used in some works in the area[5, 29, 36, 37, 38, 39]. The other main topic considered was related to the definition of empirical methods adopted by the experimental software engineering community. Besides the most common methods, such as *controlled experiment* or *survey*, we considered the methods presented by Easterbrook et al. [40] as the main empirical methods adopted in software engineering. From the Easterbrook et al. work, we added the methods *action research* and *ethnography* in our search.

We considered the specific syntax of each EDS to compose terms and to consider the plural (for some EDS we had to specify the plural, in others it was not necessary). A summary of the final terms, using quotes to compose terms and connectives *AND* and *OR* follows:

> (smell OR "design flaw" OR disharmony OR "code anomaly" OR "design anomaly")
>
> AND
>
> (experiment OR empirical OR survey OR ethnography OR "action research" OR "exploratory analysis" OR study OR controlled)

Table 1 shows the number studies retrieved by each EDS.

Table 1: Number of studies retrieved by the EDS

Years	IEEE Xplore	ACM DL	ScienceDirect	Scopus
2002- 2013	1079	71	70	910

4.3. Inclusion/exclusion criteria

We searched for papers presenting empirical studies focusing on the smell effect, which are correlation studies or studies evaluating human aspects in smell detection. As inclusion criteria, we only included papers:

- *written in English*,
- *from journals or conferences*. We did not consider gray literature, such as technical reports or PhD theses,
- *from software engineering*. Once we adopted some generic terms, we noticed that many papers were related to other engineering subjects. Many papers addressed smells in the sense of odors.
- *addressing smell effect*. An example of a paper excluded because it was not in this classification was a paper investigating component modularity. The authors claimed that this was related to smell, but they did not investigate the smell effect.

The exclusion criteria were refined during all the research. First, we analyzed a sample of the retrieved papers in order to align the perception of all researchers involved in the SR. Then, we iteratively refined the exclusion criteria. The main difficulty was with papers addressing smells, but not the smell effect. We removed a paper if it:

- *focused on the evaluation of automatic detection heuristics*. Their findings were normally related to the quality of their heuristics in comparison with another strategy of smell detection.
- *focused on refactoring issues*. Some papers evaluated refactoring tools or strategies. We also found papers investigating types of refactoring used.
- *focused on visualization tools*. In these cases, the findings point to the effectiveness of the tools. Due to this, we considered them outside the scope of our SR.
- *focused on metrics for smells*. Also in these cases, the findings are not related to the smell effect, but to the quality of metrics.
- *is a short paper*. This exclusion criterion was proposed after in-depth reading. We considered that the constraint of space makes a consistent presentation of empirical results difficult. We defined short paper as a paper with fewer than five pages.
- *did not focus on smells for code design*. We found papers using the term smell, but not related to code design. In two cases, we found papers addressing the aspect smell; and in one case, we found a paper addressing lexicon smell. As the nature of these smells is different, we disregarded these studies from our SR.
- *is a summary of other studies*. We defined this exclusion criterion because we found one paper summarizing findings of other five studies based on the same experimental setup [7]. Once we had accepted the other five works as primary studies of our SR, we removed it. We discuss this issue later.

4.4. Quality evaluation

Following principles of good practices for conducting SR [9], we defined quality assessments of the primary studies according to our aims. As we were interested in the current empirical knowledge on the smell effect, we had to consider a wide variety of context aspects, evaluating different empirical methods and setups. Due to this, we considered the quality based on the presentation of the aim, experimental setup, results, discussions and limitation of the potential primary studies of our SR. We adapted our general criteria from [41, 42]. Both works mapped the main elements in reporting controlled experiments.

To align the evaluation of researchers, we defined a checklists based on [43, 44, 45] and drew up a set of questions that we considered relevant to describe the study. Table 2 shows these questions. Each researcher considered all these questions in the quality evaluation of the studies. In cases of doubt, at least one different researcher evaluated the study and after discussion a consensus about the quality of the study was reached.

Table 2: Quality evaluation (extracted/adapted from [44, 45])

Problem statement
Q1. Is research objective sufficiently explained and well-motivated?
Research design
Q2. Are the sample and experimental units described?
Q3. Is the design of the experiment described?
Data collection
Q4. Are the data collection procedures and measures described?
Q5. Are the data analysis procedures defined?
Data analysis
Q6. Are the results of the study adequately described?
Conclusion
Q7. Are the findings of study clearly stated?
Q8. Does the paper discuss limitations or validity?

5. Filtering and data extraction processes

The reliability of the filtering and data extraction processes was enhanced by pair-review and group discussion in case of disagreement. For all cases, a consensus was reached by group discussion, which occurred in all phases of the filtering process.

5.1. The filtering process

The filtering process was supported by *StArt*[4], a support tool for SR. The tool helped us in all phases of the filtering process. We saved the retrieved results for each EDS in *.bibtex* format. Then, we imported the files in the *StArt* tool. The total number of retrieved material was 2130. Throughout the filtering process, each study was reviewed by two researchers independently. Four reviewers were involved in this phase. If some disagreement arose, the paper remained until the in-depth reading phase. Figure 3 shows the different phases of the filtering process and the final number of studies for each phase.

5.1.1. Removal of unrelated material

From the 2130 items of retrieved material, we found a set of duplicated and unrelated material which we removed. In most cases, the unrelated material was abstracts and conferences proceedings, but we also removed some Masters and PhD theses, as well as some editorial documents. This was carried out by one researcher as it was a straightforward task. After removing all unrelated material, there were 1727 papers remaining.

5.1.2. Exclusion based on title and abstract reading

Before starting the filtering based on title and abstract reading, we randomly sampled 50 papers in order to agree collectively on the exclusion criteria. All the researchers performed the activity for these papers. Then we met to discuss all 50 papers. There were only a few disagreements (2 out of 50). Moreover, we refined the exclusion criteria. After a group discussion, a consensus was reached.

[4] http://lapes.dc.ufscar.br/tools/start_tool

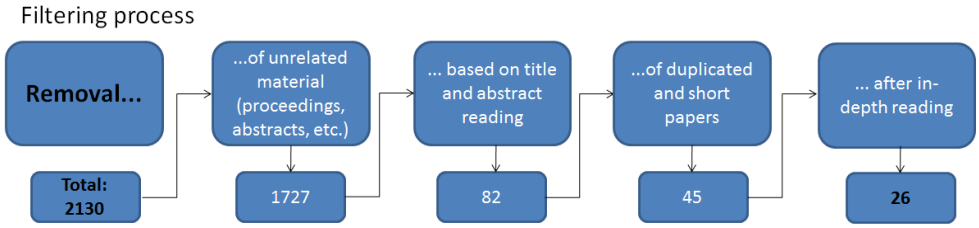


Figure 3: Filtering process

After this. the other papers were independently evaluated by two researchers. During this phase, superficial reading and group discussion were used to mitigate doubts, which were mainly related to papers addressing smell, but not its effect on software development. For example, we found papers addressing automatic detection of smells from different perspectives: i) the papers adopted automatic detection heuristics in order to establish some type of correlation between smells and software attributes; or ii) the papers proposed and evaluated an automatic detection heuristics, presenting findings related to the heuristics. For us, the former perspective is related to the smell effect, while the latter is not. Despite mitigating doubts by discussion in this phase, there were 98 disagreements, representing 5.8%, from the total of 1677 (1727 - 50) papers. After this phase, there were 82 papers remaining.

5.1.3. Removal of duplicated and short papers

In this phase, we downloaded all 82 papers. As EDS uses different special characters, some papers were duplicated. They had the same title, but they were written in a different way, especially in cases where a colon or quotation marks were used. In this phase, we also decided to remove short papers, because we noted that the empirical description was insufficient for our aims: we understood it unfair to consider the findings of complete and short papers in the same way. At the end of this phase, there were 45 papers remaining.

5.1.4. Removal after in-depth reading

During the in-depth reading, there were a lot of group discussions, specially related to doubts about findings on the smell effect, as previously discussed. The group discussions were very useful to align the view of the type of papers we were interested in and to adjust the exclusion criteria. After in-depth reading, we chose the 26 primary studies considered in our SR.

5.2. Data extraction

We followed two main trails in the data extraction process we adopted in our SR. The first is the extraction of context aspects, which are mainly quantitative data. The second is the extraction of primary studies findings, which were the basis of the thematic synthesis. In this case, this is textual information, i.e. qualitative data. Figure 4 shows the type of data that we extracted adapted from Cruzes and Dybå [11]. We discuss them below.

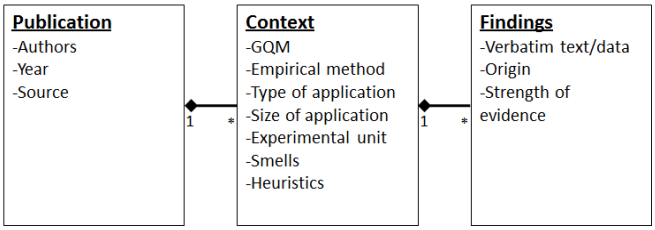


Figure 4: Data extraction (adapted from [11])

5.2.1. Context aspects

The data extraction of context aspects was done by one researcher and checked by another researcher. To make the checking process simpler, the researcher extracting the data marked the document, inserting comments in parts of the text where he/she found the information. In order to maintain the data, we developed a simple piece of software based on MVC (model-view-controller) architecture. The software is composed of input-views and a DBMS (database management system), supporting SQL language.

We consider the two left boxes in Figure 4 as the relevant context aspect for our SR. The first box describes authors, year and sources as the demographic data.

The other elements were detailed in the context description (the middle box in the Figure 4). The first data type is GQM. We adapted the GQM (Goal/Question/Metrics) template [46, 14] to capture the description of the paper objective. The GQM includes elements to be filled in as shown below.

Analyze < ... >
for the purpose of < ... >
with respect to their < ... >
from the point of view of the < ... >
in the context of < ... >

We extracted the three first elements (*analyze < ... >, for the purpose of < ... > and with respect to their < ... >*). The other data types that we used to characterize the studies are detailed in Table 3. For the sake of simplicity, we present only values that we found in the table.

5.2.2. From primary study findings to a thematic map

In this section, we describe the process that we adopted to produce the thematic map from the findings of the primary studies. Two researchers were mainly involved in this phase. A third researcher participated when consensus was not reached, which occurred in a few cases. During all the extraction and

Table 3: Data extracted from the context aspects

Data type	Values
Type of method	correlation studies, controlled experiment (in-vivo), controlled experiment (in-vitro) and survey
Type of application	open source systems, commercial, constructed and unspecified
Size	small, medium, large and unknown
Experimental unit	undergraduate, graduate, professional and unknown
Type of smell	from known catalogs, such as Fowler [2] and Lanza and Marinescu [3]
Heuristic for detection	free text for description of the heuristic or the tool

Table 4: Example of extraction and coding of findings for the S13 study

Finding's extraction	Code (extraction number)
1 - many of the code anomalies detected by the employed strategies were not related to architectural problems	1 - detection strategies are inaccurate in identifying architecturally-relevant code anomalies (1,2)
2 - detection strategies are inaccurate in identifying architecturally-relevant code anomalies	
3 - most of the automatically-detected code anomalies were not associated with architectural modularity problems, leading to many false positives	2 - the detection strategies seem to have a tendency to send developers in wrong directions when addressing code anomalies related to architectural modularity problem (3,4)
4 - detection strategies seem to have a tendency to send developers in wrong directions when addressing code anomalies related to architectural modularity problem	

production of the thematic map, we used double-checking. One of the researchers extracted and coded the data. A second researcher checked the results, returning comments and suggestions to the first researcher who then re-checked the results using the comments and suggestions. If they disagreed, they met to discuss. If some doubt persisted, a third researcher was consulted.

The process ran iteratively, avoiding extensive volume of data and aligning the researchers' ideas about extraction of useful information for the SR. In fact, the learning acquired in the iterations strengthened the process of data extraction, coding and making the thematic map. There were five iterations. In the first iteration, we chose eight of the most relevant primary studies. It was the most time-consuming iteration. For the other iterations, we sampled studies randomly.

For each iteration we used a spreadsheet to maintain the data. At first, we copied textual parts of the studies where we found some finding. We focused on the sections "abstract", "introduction", "result", "discussion" and "conclusion", but we read the whole paper. We copied textual information containing sufficient information of context to make the findings of the studies clear. In some cases, it was necessary to copy whole paragraphs. Below, we show a simple example for the study with ID S13 (we will present the studies and their IDs in Section 6). We show part of the sections "Abstract" and "Study results" of the S13 study, highlighting the most relevant information:

> **Abstract**: "The outcome of our evaluation suggests that ***many of the code anomalies detected by the employed strategies were not related to architectural problems***. Even worse, over 50% of the anomalies not observed by the employed techniques (false negatives) were found to be correlated with architectural problems."

> **Study results**: "In general, our analysis reveals that ***detection strategies are inaccurate in identifying architecturally-relevant code anomalies***. Specifically, ***most of the automatically-detected code anomalies were not associated with architectural modularity problems, leading to many false positives***. In general, the average of the automatically-detected code anomalies represented about 45% (or less) of the total number of code anomalies related to architectural modularity problems.
> ....
> Even worse, many of the code anomalies harmful to architectural modularity problems were not automatically detected by strategies, leading to a high rate of false negatives. About 55% or more of the non automatically-detected code anomalies were related to architectural modularity problems. These results indicate that ***detection strategies seem to have a tendency to send developers in wrong directions when addressing code anomalies related to architectural modularity problem***."

From the textual information, we extracted what we considered relevant as the findings of the studies. After that, we coded the findings, summarizing them in a sentence. Table 4 shows an example of the extraction and codes for the S13 study. The left column shows the findings that we extracted from the text above. The right column shows the code and the number of the related finding, inside brackets. For the sake of simplicity, we used an example where the coding was very simple, because we adopted the same sentence written in the study. In some cases, the coding required making a new sentence summarizing the finding.

We defined the thematic map for our SR grouping the codes according to themes and sub-themes that we identified. For the example in Table 4, we classified the codes 1 and 2 into the theme "correlation with issues of development", sub-theme "architectural quality", and into the theme "detection", sub-theme "smell as a predictor". We present the complete thematic map in Appendix A and the main results in Section 6.

6. Results

In this section, we present the results of our SR. First, we present the primary studies. Then, we present the demographic data, the contextual characterization of the studies and the thematic map.

Table 5 presents the primary studies that we selected in our SR. The table shows the ID, bibliographic reference, title and year. It is important to note that some different studies present results as part of the same experimental setup. The most evident case happened for the S4, S6, S7, S8 and S9 studies. The same happened for the S22 and S23 studies. For these two cases, the studies have the same experimental setup and at least one common author. Other studies have a similar, but not identical, set of object software. These are the S1 and S15 studies, and the S19 and S20 studies. For the S1 and S15 studies, there

Table 5: The SLR's primary studies

ID	Reference	Title	Year
S1	[19]	The evolution and impact of code smells: A case study of two open source systems	2009
S2	[47]	An exploratory study to investigate the impact of conceptualization in god class detection	2013
S3	[23]	Building empirical support for automated code smell detection	2010
S4	[20]	Exploring the Impact of Inter-smell Relations on Software Maintainability: An Empirical Study	2013
S5	[48]	Investigating the Impact of Design Debt on Software Quality	2011
S6	[49]	Code smells as system-level indicators of maintainability: An empirical study	2013
S7	[50]	To what extent can maintenance problems be predicted by code smell detection? An empirical study	2013
S8	[4]	Quantifying the Effect of Code Smells on Maintenance Effort	2013
S9	[24]	Do Code Smells Reflect Important Maintainability Aspects?	2012
S10	[51]	Clones: what is that smell?	2012
S11	[39]	On the Relevance of Code Anomalies for Identifying Architecture Degradation Symptoms	2012
S12	[52]	Evaluating the Lifespan of Code Smells Using Software Repository Mining	2012
S13	[5]	Are Automatically-detected Code Anomalies Relevant to Architectural Modularity?: An Exploratory Analysis of Evolving Systems	2012
S14	[53]	Are the clients of flawed classes (also) defect prone?	2011
S15	[6]	Are all code smells harmful? A study of God Classes and Brain Classes in the evolution of three open source systems	2010
S16	[29]	Identifying code smells with multiple concern views	2010
S17	[54]	Investigating the Evolution of Bad Smells in Object-Oriented Code	2010
S18	[55]	On the Impact of Design Flaws on Software Defects	2010
S19	[56]	An Exploratory Study of the Impact of Code Smells on Software Change-proneness	2009
S20	[57]	Tracking Design Smells: Lessons from a Study of God Classes	2009
S21	[18]	An empirical study of the bad smells and class error probability in the post-release object-oriented system evolution	2007
S22	[22]	Subjective evaluation of software evolvability using code smells: An empirical study	2006
S23	[33]	Bad Smells Humans as Code Critics	2004
S24	[21]	An experiment on subjective evolvability evaluation of object-oriented software: explaining factors and interrater agreement	2005
S25	[58]	Do developers care about code smells? An exploratory survey	2013
S26	[59]	Investigating the Impact of Code Smells on System's Quality: An Empirical Study on Systems of Different Application Domains	2013

are two common authors. We considered all these cases independent studies because they present findings from different perspectives on smell effect, despite sharing the same (or a similar) experimental setup.

6.1. Demographic

We considered source, year and authors as the demographic data.

6.1.1. Source's distribution

Table 6 shows the distribution of the studies by sources. The first observation is that most were published at conferences. Only six studies were published in journals. ICSM and WCRE were the conferences with the highest number of studies (four and three out of 26, respectively). Other conferences with two studies were ESEM and CSMR. Considering the journals, there were two studies in the JSS and EMSE. The other sources had only one study published.

6.1.2. Distribution of studies by year

Figure 5 shows the distribution of the studies by year. The number of studies addressing the smell effect has increased since 2009 showing that researchers have dedicated more attention to the topic recently. On the other hand, we consider that the number of studies still remains small. The years with the highest number of studies are 2010 (six) and 2013 (seven).

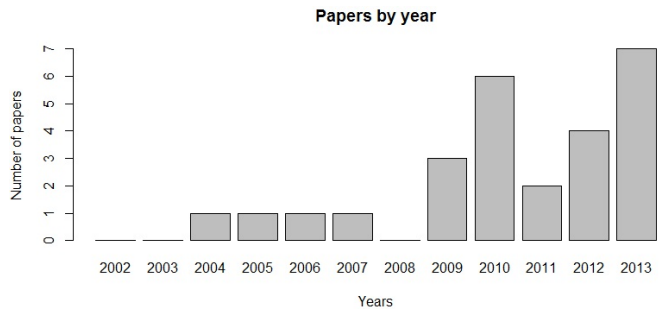


Figure 5: Distribution of papers by year

6.1.3. Distribution of studies by authors

Table 7 shows the distribution of studies by authors. Some authors appear in different studies using the same experimental setup. We have previously discussed these cases.

Table 7: Distribution of papers by authors

Author	Number of papers
Aiko Yamashita	6
Leon Moonen	4
Alessandro Garcia	3
Mika V. Mäntylä	3
Nico Zazworka	3
Arndt von Staa	2
Carolyn Seaman	2
Casper Lassenius	2
Dag Sjøberg	2
Daniela S. Cruzes	2
Forrest Shull	2
Foutse Khomh	2
Isela Macia Bertran	2
Manoel G. de Mendoncça	2
Michele Shaw	2
Steffen Olbrich	2
Yann-Gaël Guéhéneuc	2
Other 40 authors	1

6.2. Characterization of the studies

In this section, we mapped the context attributes of the studies. Table 8 shows the nature of the studies. For most cases (16 out 26), the smell effect was mainly investigated from software repositories. The total number of studies is 27 because the S3 study investigated software repositories and code inspection activity. The maintenance activity was considered in five studies, but all them based on the same experimental setup. Therefore, there was only one experimental setup which was the basis for five studies addressing the smell effect from the observations of developers performing software maintenance activities. The last observation related to the nature of the studies is that the S25 study is a survey in which questions addressed the knowledge of developers. The authors did not ask the participants to analyze any artifact or to perform some activity. Due to this, we classified this as the only study based on "Developer knowledge".

Table 6: Distribution of empirical studies focusing on the smell effect: 2002-2013

Journal/conference proceeding	Type	Number	Percent
International Conference on Software Maintenance (ICSM)	Conference	4	15.3
Working Conference on Reverse Engineering (WCRE)	Conference	3	11.5
Journal of Systems and Software (JSS)	Journal	2	7.7
International Symposium on Empirical Software Engineering and Measurement (ESEM)	Conference	2	7.7
Empirical Software Engineering (EMSE)	Journal	2	7.7
European Conference on Software Maintenance and Reengineering (CSMR)	Conference	2	7.7
International Symposium on Empirical Software Engineering (ISESE)	Conference	1	3.8
Aspect-Oriented Software Development conference (AOSD)	Conference	1	3.8
International Working Conference on Source Code Analysis Manipulation (SCAM)	Conference	1	3.8
Information and Software Technology (IST)	Journal	1	3.8
Brazilian Symposium on Software Engineering (SBES)	Conference	1	3.8
IEEE Transactions on Software Engineering	Journal	1	3.8
International Conference on the Quality of Information and Communications Technology (QUATIC)	Conference	1	3.8
International Conference on Evaluation and Assessment in Software Engineering (EASE)	Conference	1	3.8
International Conference on Quality Software (QSIC)	Conference	1	3.8
International Conference on Software Engineering (ICSE)	Conference	1	3.8
International Workshop on Managing Technical Debt (MTD)	Conference	1	3.8
Total		26	100

Table 8: Nature of studies

Nature of studies	Number of studies
Correlation over software repositories	16
Code inspection activity	5
Maintenance activity	5
Developers knowledge	1
Total	27

Table 9 shows the experimental methods adopted, and the studies by method. Most cases (14 out 26) were correlation studies, which is in accordance with the analysis of software repositories highlighted above. Another observation is related to the controlled experiments using an in-vivo setting. Five of the six studies were based on the same experimental setup, where authors observed developers performing maintenance activities. It can be argued that the empirical method adopted was a *case study*. We classified the method as controlled experiments because the authors controlled relevant factors, such as the definition of the maintenance tasks. This was highlighted in the S7 study.

Table 9: Experimental methods

Method	Number of studies	Studies
Controlled experiment (in-vivo)	6	S3, S4, S6, S7, S8, S9
Controlled experiment (in-vitro)	3	S2, S16, S24
Correlation study	14	S1, S5, S10, S11, S12, S13, S14, S15, S17, S18, S19, S20, S21, S26
Survey	3	S22, S23, S25
Total	26	

Figure 6 shows the number of smells by study. The controlled experiments carried out in an in-vitro setting investigated a small number of smells (≤ 3). These are S1, S16 and S24 studies. On the other hand, five controlled experiments carried out in an in-vivo setting investigated a higher number of smells (≥ 12). These are S4, S6, S7, S8 and S9. Note that this does not indicate that the participants had knowledge or were trained about a higher number of smells. Instead, this happened because in the experimental setup the authors had direct contact with the participants, interviewing them or using a think aloud protocol as the method of data collection. In the S4 study, for example, the authors automatically identified the smells and asked the participants if they agreed with the automatic results. In the S9 study, they extracted relevant sentences from the interviews and compared with the definitions of smells.

The surveys evaluated a low (S23 addressed 3 smells) and high (S22 study addressed 22 smells) number of smells. The same happened for the 14 correlation studies: the number of smells addressed significantly varied.

The S25 study did not address any specific smell and therefore it does not appear in Figure 6.

We also mapped the type of smells addressed by the studies. Table 10 shows them. As we can see, the *god class* and *feature envy* smells were the most commonly addressed: 18 and 16 studies considered them, respectively. Another observation is that there were 63 smells in total, which we consider a high number. We also noted that the S13 study considered some aspect smells. We disregarded these.

Table 10: Smells addressed

Smell	Number of papers	Percent (%)
God class	18	28.6
Feature envy	16	25.4
Shotgun surgery	12	19.0
Data Class	10	15.9
Duplicated code	9	14.3
Long Parameter List	9	14.3
Refused Bequest	8	12.7
Long Method	7	11.1
Misplaced Class	7	11.1
Data Clump	6	9.5
God Method	6	9.5
Divergent Change	5	7.9
Interface Segregation Principle Violation	5	7.9
Large class	4	6.3
Message Chains	4	6.3
Temporary variable used for several purposes	4	6.3
Use interface instead of implementation	4	6.3
Brain class	3	4.8
Speculative Generality	3	4.8
Alternative Classes with Different Interfaces	2	3.2
Brain Method	2	3.2
Inappropriate Intimacy	2	3.2
Intensive Coupling	2	3.2
Lazy Class	2	3.2
Switch statements	2	3.2
Temporary Field	2	3.2
Other 37 smells	1	1.6
Total of smells = 63		

6.3. Thematic map

In this section, we present the thematic map produced in our SR. We grouped the findings of the studies into five themes. For each theme, we identified sub-themes. Table 11 lists them, presenting an example of the findings to clarify the meaning of the themes and sub-themes.

Table 12 shows the distribution of studies by themes and sub-themes. It is possible to note that the number of studies by theme

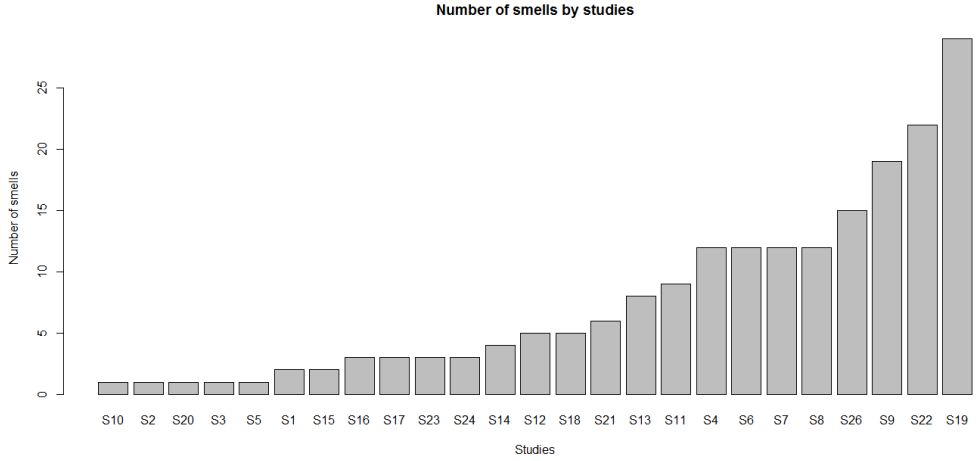


Figure 6: Number of types of smells by studies

Table 11: Themes and sub-themes

Theme	Sub-theme	Example of findings
Correlation with issues of development	Changes and defects	*Changes of infected components with ISP Violation is significant larger*
	Effort	*God class don't require a higher maintainability effort than non-code smells*
	Design quality	*Code smells often related to immature design and implementation*
	Architectural quality	*There is a correlation between code anomaly and problems in architectural design*
	Harm	*Presence of God class and Brain class may be beneficial to a system*
Human aspects	Factors affecting detection	*Demographics aspects of the developers affect the code smell evaluations*
	Agreement on detection	*The agreement on god class detection is low*
	Human evaluation versus software measures	*The developers' evaluations on code smells do not correlate with related source code metrics*
	Decision drivers	*Misplaced methods is the strongest driver for letting the subjects classify a class as a god class*
	Knowledge about smells	*There is not strong understanding about code smells and practical application of these concepts*
	Strategy on detection	*Participant with optimistic style investigate code smell candidates using more combinations of views when compared to participants with more conservative styles*
	Detection difficulty	*The subjects perceive detecting god classes as an easy task*
Programming	Smell removal	*The developers resolve code smells for opportunistic reasons*
	Smell introduction	*The developers who revised most often also introduced the most smells*
Detection	Supporting smell detection	*Developers need support to enhance detection of code anomaly*
	Smell as a predictor	*Expert judgment is most accurate method for assessing system maintainability*
	Novel smell	*A new smell was propose: Multiple Inheritance simulation*
Other correlations	Metrics versus smells	*Code smells are strongly influenced by size (LOC)*
	Smell density	*Code smell density is likely to be inaccurate when comparing size differing systems*
	Frequency of smells	*Some design flaws are more frequent than others*
	Inter-smell relation	*There is interaction between code smells*

(column *#studies by theme*) can be lower than the total number of studies by sub-theme (column *#studies by sub-theme*). For example, for the first theme (*correlation with issues of development*), the total number of studies by theme is 17, and the total number of studies by sub-theme is 22 (11+4+3+2+2). This happened because some studies present findings about different sub-themes. We detail the themes and the main sub-themes results as follows.

6.3.1. Correlation with issues of development

This was the theme with the highest number of studies. There were 17 studies correlating smells and some aspect related to the software development. We identified five sub-themes: *changes and defects*, *effort*, *design quality*, *architectural quality* and *harmfulness*. The most interesting analysis that we did was related to the sub-themes *changes and defects*, *effort* and *architectural quality*.

Changes and defects. By changes, we considered maintenance or refactoring activities. *Changes and defects* was the main sub-theme, with 11 studies. Eight of them are correlation studies. Three of them are controlled experiments carried out in an in-vivo setting, based on the same experimental setup. The studies evaluated medium and large commercial or open source systems. Due to their focus on evolution, they adopted automatic detection. Only the controlled experiments adopted human evaluation as well as automatic detection. In total, the studies correlating smells and changes/defects investigated 53 smells, which makes it difficult to generalize from the findings about a specific type of smell.

Despite the difficulties of generalizations, an overall discussion was possible. We found some divergent findings. On reading some studies the adoption of smell concept is considered a useful practice in software development, but on reading other studies the opposite is considered. Table 13 shows some of these divergent findings side-by-side. In the left column, we show studies and findings indicating which we called a *positive correlation*. This is a study that presents a positive correlation if its findings consider smell as a useful concept to be adopted in software development. For the opposite cases, we called them *negative correlations* (the right column in the table).

Effort. This is the second sub-theme in a number of studies addressing correlations between smells and software development attributes. There were four studies addressing this sub-theme. Two of them identify a negative correlation between smells and effort in maintenance

Table 12: Distribution of studies by themes and sub-themes

Theme	Sub-theme	#studies by sub-theme	#studies by theme
Correlation with issues of development	Changes and defects	11	17
	Effort	4	
	Design quality	3	
	Architectural quality	2	
	Harm	2	
Human aspects	Factors affecting detection	7	7
	Agreement on detection	6	
	Human evaluation versus software measures	2	
	Decision drivers	1	
	Knowledge about smells	1	
	Strategy of detection	1	
	Detection difficulty	1	
Programming	Smell removal	7	7
	Smell introduction	4	
Detection	Supporting smell detection	6	12
	Smell as a predictor	5	
	Novel smell	1	
Other correlations	Metrics versus smells	6	10
	Smell density	3	
	Frequency of smells	3	
	Inter-smell relation	2	

Table 13: Positive versus negative correlations between smells and changes/defects

Positive correlations		Negative correlations	
Study	**Finding**	**Study**	**Finding**
S1	*the change proneness of components with smells is higher than the ones without*	S4	*there are smells that are not associated with maintenance problems*
S5	*god classes impacts maintainability*	S7	*the role of code smells on the overall system maintainability is relatively minor*
S9	*there is a correlation between maintainability factors and code smell*	S10	*there is no correlation between clones and bugs*
S19	*classes with code smells are more change-prone than others*	S14	*taken in isolation, classes exhibiting the identity disharmonies design flaws do not have an increased likelihood to exhibit defects that classes which do not reveal design flaws*
S21	*some bad smells are associated significantly with class error proneness*	S18	*design flaws cant be considered more harmful with respect to software defects*

activities. The S8 study found that *after adjustments for file size and the number of changes, code smells are not significantly associated with increased effort*. The authors consider that file size and the number of changes (code churn) are a better predictor of variations in effort than code smells. In the same direction, the S3 study claims that the smell *god class do not require a higher maintainability effort than non-code smells*. Despite finding negative correlations between smells and effort in maintenance activities, we have to consider that the number of studies agreeing on this aspect is low (only two).

The other two studies in this sub-theme addressed other aspects related to effort. The S2 study addressed the impact of software characteristics on god class detection effort. They consider that *the software characteristic does not impact effort in god class detection*. Furthermore, the S17 study proposes a strategy on applying effort. For their authors, *maintenance effort should prioritize Long Method smells over other smells*.

Architectural quality. This sub-theme also presented some interesting results, despite the fact that only two studies addressed it. We noted that they were consistent. Both S11 and S13 studies identified problems in the detection strategies of smells from the architectural perspective.

The S11 study found that *there is a correlation between code anomaly and architectural degradation*. However, despite noting some correlation between smells and architectural degradation, the authors noted that *there is no specific code anomaly more relevant as indicator of architecture degradation*. Also presenting a negative correlation between smells and architectural quality, the S13 study found that *the detection strategies seem to have a tendency to send developers in wrong directions when addressing code anomalies related to architectural modularity problem*.

6.3.2. Human aspects

There were seven studies investigating this theme. We identified seven sub-themes being addressed by these studies: *factors affecting detection*, *agreement on detection*, *human evaluation versus software measures*, *decision drivers*, *knowledge about smells*, *strategy of detection* and *detection difficulty*. As previously, we discuss only the main sub-themes.

Factors affecting detection. All studies addressing human aspects presented findings on this sub-theme, and many different aspects of the context were considered. Three studies carried out controlled experiments, three carried out surveys, and one performed a correlation study. The studies used small, medium and large systems; and they used open source, commercial and constructed systems. Four of these studies used professionals as participant subjects, two used students, and the correlation study did not use human participants. Except for the correlation study, the studies based their analysis on the code inspection activity.

Analyzing findings of studies in this sub-theme, we also identified a trend which is the impact of demographic data on the human evaluation of smells. The S23 study found that *demographic data (knowledge, role, work experience) seemed to explain some of the variances in smell evaluations*. This is similar to the finding presented by the S22 study: *demographics aspects of the developers affect the code smell evaluations*. The S2 and S16 studies found that human evaluation of smells is affected by *human characteristics* and *knowledge acquired in one version*. The S25 study considered knowledge about smells as a factor affecting human evaluation. For the authors, *the more details people actually know about code smells, the more concerned they are by the presence of smells*. Only one study (S24) did not find strong evidence of the impact of demographic data on the human evaluation of smells. It found that *the demographics were not useful predictors*

neither for evaluating code smells nor the refactoring decision.

Another factor addressed was the software domain. The S26 study considered the software domain as *an important factor that should be included in the detection strategies for various code smells*. It was the only study addressing factors affecting smell detection that are not related to demographic data.

Agreement on detection. Six studies (out of 26) addressed agreement among participants. All of them are contained in the set of studies addressing factors affecting smell detection. Due to this, the characteristics of the studies are very similar: i) four carried out controlled experiments, and two carried out surveys; ii) the studies used small, medium and large systems; iii) they used open source, commercial and constructed systems; iv) four studies had professionals as participants, while the other two studies used students as participants; and v) all studies adopted code inspection as the task during the experiments themselves. Except for the S22 study, which was a survey asking participants for an extensive variety of smells, all other studies investigated agreement on a small set of smells (three, at the most, for each study).

We also identified a trend here. Only the S24 study found that *there is partial concordance among the evaluators in all evaluations*. All others found low agreement. This was affirmed in the S2 and S3 studies. The S16 study found the same idea, but it presented the idea in a different way. The authors stated that *participants had different perceptions and judgments during the code smell identification*. The same happened with the S22 study. Their authors found that *developers have not a uniform opinion on the 'smelliness' of the source code*.

Human evaluation versus software measures. This sub-theme was addressed by the S22 and S23 studies, which adopted survey as experimental method. Both studies are based on the same commercial system, with professional participants, including developers of the object software. They found that *developers' evaluations of the smells do not correlate with the used source code metrics*.

6.3.3. Programming

Seven studies addressed aspects related to the smell programing. We identified two sub-themes being addressed by them: *smell removal* and *smell introduction*.

Smell removal. All seven studies addressed this sub-theme. Four of them are correlation studies and three of them are controlled experiments (two of them based on the same experimental setting). They observed maintenance activities, software repositories and code inspection activity. Nineteen smells were addressed by these studies, which made it impractical to generalize.

Despite the difficulties in generalizing findings, we noted that smell removal is not systematized. The evidence for this is the findings presented by some of these studies. The S6 study, for example, found that *there is no clarity on the intent of developers prioritize or even consider the eradication of code smells*; the S12 study found that *the developers resolve code smells for opportunistic reasons*; and the S17 study found that *designers perform refactorings routinely based on their subjective perception of problematic code areas rather than applying them as solutions to identified design problems*.

Smell introduction. About the introduction of smells we noted that in some cases attempting to remove smells might introduce smells. This was observed by the S20 study. The authors consider that *the correction of a god class may also move the problem to a different class*. The S8 study found that *the developers who revised most often also introduced the most smells*. The S11 and S17 studies presented discussions about when smells are introduced. The S11 study found that *early code anomalies induces architectural anomalies*, and the S17 study found that *smells are introduced during the initial design/implementation*.

6.3.4. Detection

Ten studies addressed aspects related to smell detection. We identified three sub-themes on this topic: *smell as a predictor*, *supporting smell detection* and *novel smell*.

Smell as a predictor. Some studies compared strategies of smell detection with other predictors of potential problems in software development. Five studies addressed this sub-theme. Two controlled experiments, two correlation studies and one survey. The two controlled experiments were based on the same experimental setup. Four studies used commercial systems, and one study used open source systems. The size of the systems varied from medium to large. The participant subjects were professionals, except for one of the correlation studies, where human evaluation was not considered.

Again, the main problem in generalizing knowledge related to the findings of these studies is the high number of smells addressed (48 in total). Moreover, many topics were considered. The S13 study focused on architectural problems (*detection strategies are inaccurate in identifying architecturally relevant code anomalies*); the S6 study addressed the relationship between smell detection strategies and expert judgment, presenting findings from different perspectives, such as *expert judgment is most accurate method for assessing system maintainability* and *code smells approach can detect design flaws that may be overlooked by experts*. For the authors of the S6 study, the combination of software measures and expert judgment can lead to more complete evaluations of maintainability than the smell concept.

Another perspective was presented by the S9 study. The authors consider that code smell *can cover a more heterogeneous spectrum of factors than software metrics and expert judgment individually*. However, they understand that smell detection needs to be *complemented with alternative approaches, such as semantic analysis and manual inspection, for address maintainability factors*. In the same way, the S19 study considers smells as an interesting predictor, but not as the most interesting strategy. They declared that *smells can provide to developers recommendations easier to understand than what metric profiles can do*, but *smells are not replacement to metrics in the ability of building change-proneness or fault-proneness prediction models*.

Supporting smell detection. Four studies addressed the adoption of some type of support for smell detection. The context was well defined, considering medium to large commercial and open source systems; and using professionals as participants of the studies. All of them investigated the code inspection activity. Two of them carried out controlled experiments, one of them performed a correlation study, and one adopted a survey as its empirical method. All them consider relevant, or at least, interesting, the use of some type of complementary support on smell detection. They highlighted the definition of a process (S3), visual resources (S16), tools for automatic detection (S11) or source code metrics (S22) as complementary support for smell detection.

6.3.5. Other correlations

We identified some topics not directly related to smell effect. We classified them under the follows sub-themes. *Metrics and smells* (studies correlating smells and software measures); *smell density* (studies discussing the relationship between smell/line of code); *inter-smell relation*; and *frequency of smells*. For the sake of simplicity, we do not detail these sub-themes because we consider that they do not impact the smell effect significantly.

Table 14: Main findings-by-theme and evidences

Theme:	Correlation with issues of development
Finding:	*smell concepts do not support evaluation of design quality, in practice*
Evidences:	i) divergent findings correlating smells and changes and defects; and common findings presenting negative correlations between ii) smells and effort on maintenance activities, and iii) smells and architectural quality
Theme:	Human aspects
Finding:	*human evaluation of smells should not be trusted*
Evidences:	i) all studies addressing the topic found low agreement on human evaluation; and studies tend agree that: ii) human evaluation of smells does not correlate with metrics, and iii) demographic data impacts smell detection
Theme:	Programming
Finding:	*there is not a systematized process in order to avoid/remove smells*
Evidences:	convergence on findings of studies addressing smell removal and introduction
Theme:	Detection
Finding:	*the current smell detection strategies are not consistent with practical problems of the software development*
Evidences:	i) the studies on the topic consider that some complement is necessary on smell detection; and ii) the most of studies have shown that the adoption of smells as a predictor is not trusted, not only considering human evaluation, but also using automatic detection heuristics

7. Discussion

In this section, we discuss our findings based on the results previously presented. We also propose some challenges for the community in order to enhance future discussions in the area. Table 14 summarizes our findings-by-theme and the evidence that led us to them. It is important to note that the findings are related to the current knowledge about the smell effect. We expect that these findings will be affected by new studies on the topics, especially by studies facing the challenges that we discuss below.

7.1. Challenges

We identified some challenges in order to improve the evaluation of the smell effect on software development. These challenges have to be faced to better understand why some studies have shown that smells concepts do not correlate with problems of software development, as expected from theoretic discussions. The challenges are:

1. *Grouping smells according to their nature*. The first observation that we made was related to the large number of smells addressed in the studies. This happened for all the themes that we identified, making it difficult to generalize from our findings. From our point of view, it is necessary to understand the nature of the smells, grouping them according to the interests of analysis. We found only one work grouping smells [22]. However, it has not yet been strongly validated by other studies
2. *Outlining context aspects*. In some cases, the difficulties in generalizing findings were related to the wide context aspects considered. This occurred, for example, in the analysis of studies addressing human aspects. While some studies adopted small software, other studies adopted large software. The same for participant subjects. While some studies adopted students, other studies were carried out with professionals. When these context aspects were well outlined, the problems were related to the high number of smells addressed, such as in the case of the sub-theme *changes and defects*, or the small number of studies, such as in the case of the sub-theme *effort*. We consider that we identified an initial starting point to understand the smell effect in a specific context. However, we understand that more studies in the area are necessary to better outline the context of studies addressing the smell effect.
3. *Understanding the relevance of subjectivity in smell evaluation*. Most studies addressing human evaluation (comparing with metrics, automatic detection tools, or investigating agreement on smell detection) showed that human evaluation should not be trusted. In general, these studies focused on specific smells, mainly god class. Note that the evaluation of god class is easier than the evaluation of other smells, such as feature envy or shotgun surgery, because it involves the observation of a smaller number of classes (this is related to the investigation of the nature of smells that we proposed as the first challenge). This reinforces our belief that human evaluation should not be trusted, but it also raises difficulties in studies in the area. We consider that two strategies can be followed: i) deepen the evaluation of human factors affecting smell detection; and ii) evaluate cognitive aspects in smell detection, which is related to program comprehension, and requires knowledge both in computer science and cognitive psychology, as observed by Maletic [60]. We have called this issue the *code smell conceptualization problem* and we have discussed the problem in [47, 61, 62].
4. *Increasing the number of studies in the area*. Some sub-themes were addressed in just a small number of studies. For example, the architectural quality was addressed by only two studies. Effort was addressed by four studies, but only two focused on correlating effort and maintenance activities. This also happened for other sub-themes. The need for more empirical studies in the area is also highlighted in [4, 8, 23, 33].

8. Threats to validity

In this section, we discuss some factors that might have biased our SR. We highlight the search and selection of the primary studies, and the data extraction and synthesis processes.

8.1. Search of primary studies

A common threat of SR is finding all the relevant studies [45, 12]. We consider two threats here. First, the search for relevant studies might be biased by the choice of the EDS. We adopted four EDS, but we understand that this number could be higher. Despite this, we adopted well known EDS, including two relevant indexers, which were Scopus and Science Direct. We based our choices on our experience, discussions among researchers of our research group and other relevant secondary studies. ACM and IEEE, for example, were adopted by all secondary studies that we found in Software Engineering, such as [43, 45, 12, 63]. Scopus and Science Direct were also adopted by at least one of these studies. We also selected some relevant studies that we found in our ad-hoc review and checked if they were returned by one of the EDS that we chose. For all cases, the studies were found in at least one of the EDS. Because of this, we are confident that our choices were comprehensive, despite the absence of some EDS.

Another threat related to the search for primary studies is the definition of the search terms. In our case, we have to consider terms related to empirical methods and to smell concepts. One of the problems in our case is that we did not use names of specific smells. During the group discussions, we understood that, even using specific names of smells in the title, some context discussion linking the type smell with the theory would be necessary. Due to this, we consider this a weak threat to our work. We also checked the search terms looking for them in some studies that we had found in our previous ad-hoc review.

8.2. Selection of primary studies

We have to consider the threat of removing some relevant primary study in the selection phase. To mitigate this bias all studies generating doubts remained until the in-depth reading phase, when at least two researchers read and discussed the study. Due to this, in the in-depth reading phase 19 studies were removed, representing about 42% of the 45 input studies of this phase.

8.3. Data extraction and thematic synthesis

We used double-checking in order to minimize bias in the data extraction, including in the extraction of qualitative data used in the thematic synthesis. Despite understanding that bias related to the subjectivity is part of qualitative syntheses [64], we mitigated it by checking and re-checking the extraction. All the data extraction was done by one researcher, checked by the other and re-checked by the first researcher. When doubts remained, a third researcher was consulted.

9. Conclusion and future works

In this work we presented a systematic review (SR) to synthesize current knowledge about how the smell concept impacts software development practices, namely the smell effect. The SR focused on two types of empirical studies: i) studies correlating smells with some aspect of software development, such as effort in maintenance activities, and ii) studies investigating the human role in smell detection, which are studies investigating agreement, factors and decision drivers affecting human evaluation of smells. We selected 26 papers as primary studies of our SR.

The SR is based on a thematic synthesis [11], a technique to identify, analyze and report patterns from qualitative research. In our case, we coded and grouped textual information, which is how findings are mainly presented in empirical studies.

One of the findings of our SR is that the smell concept does not support the evaluation of design quality in practice. We considered the set of divergent findings correlating smells with issues of software development as evidence that the area lacks an understanding of the smell effect. Another finding is that human evaluation of smells should not be trusted. This is because we found a trend in primary studies showing that the agreement on smell detection is low. Moreover, the studies found that demographic data, such as developers' experience, significantly impacts smell evaluation. We also highlight that we did not find evidence that the current strategies of smell detection are consistent with practical problems of software development.

From these findings, we identified some challenges for the area. We understand that the area needs to better understand the nature of each smell, in order to group them according to their nature. This will enable generalizations about specific groups of smells to be made, which was not possible in our SR. Another challenge is to deepen the comprehension of subjectivity in smell evaluation. We think that it is necessary to better understand how human factors affect smell detection. We called this issue *the code smell conceptualization problem* and we discussed this in [47, 61, 62]. We believe that in depth discussions about the code smell conceptualization problem will help the area to systematize the practical adoption of smell concepts.

As future works, we plan to face these challenges. We intend to evaluate catalogs of smells, understanding their nature and which aspects reflect the potential design problems. Then, we will classify the smells in order to group findings that we have found in empirical studies on the topic. We also intent to deepen investigations about factors affecting human evaluation of smell in order to better understand reasons of low agreement on smell detection. We consider that this understanding will help to enhance decisions about the practical adoption of the smell concept.

Acknowledgement

This work was partially supported by SECTI-BA (Bureau of Science and Technology of Bahia) - Fraunhofer Project Center for Software & Systems Eng. agreement 2012/001.

Appendix A. Thematic map

In this appendix, we present the complete thematic map of our SR. We grouped the themes in different tables because of the extensive volume of information. Table A.15 shows the thematic map for the theme "correlation with issues of development", sub-theme "changes and defects". The columns ID in the table are formed by the ID of the study and the number of the finding, which we identified sequentially during the extraction process. The other columns describe the sub-theme showing their codes and forming the thematic map. All the other following tables have the same structure.

Table A.16 shows the thematic map for the theme "correlation with issues of development", sub-themes "architectural quality" and "harm". Table A.17 shows the thematic map for the theme "correlation with issues of development", sub-themes "effort" and "design quality". Table A.18 shows the thematic map for the theme "human aspects", sub-themes "factors affecting detection", "decision drivers" and "human evaluation versus software measures". Table A.19 shows the thematic map for the theme "human aspects", sub-themes "agreement on detection", "strategy of detection", "detection difficulty" and "knowledge about smells". Table A.20 shows the thematic map for the theme "programming", sub-themes "smell introduction" and "smell introduction". Table A.21 shows the thematic map for the theme "detection", sub-themes "supporting smell detection", "smell as a predictor" and "novel smell". Table A.22 shows the thematic map for the

theme “other correlations”, sub-themes “inter-smell relation”, “metrics versus smells”, “smell density” and “frequency of smells”.

Table A.16: Thematic map for the theme “correlation with issues of development”, sub-themes “architectural quality” and “harm”

ID	Architectural	ID	Harm
S13.1	The detection strategies are inaccurate in identifying architecturally-relevant code anomalies.	S15.1	The presence of God and Brain Classes is not necessarily harmful.
S13.2	The detection strategies are more effective in systems where architecture conformance is more strictly enforced in the code.	S15.4	Presence of God class and Brain class may be benefical to a system.
S13.3	Architecturally-relevant code anomalies often occurred in code elements responsible for implementing different architectural elements.	S5.3	God classes have been confirmed to have the technical debt property.
S13.5	The detection strategies seem to have a tendency to send developers in wrong directions when addressing code anomalies related to architectural modularity problem.		
S13.6	The imperfection of the detection strategies (related to impact on architectural problems) is not simply related to specific thresholds or combinations of particular measures.		
S11.1	Refactoring strategies do not contribute to remove architectural-relevant code anomaly.		
S11.2	There is a correlation between code anomaly and problems in architectural design.		
S11.3	There is a correlation between code anomaly and architectural degradation.		
S11.5	There is no specific code anomaly more relevant as indicator os architecture degradation.		
S11.7	Architecturally-relevant code anomalies are left in the code when the system evolves.		

Table A.15: Thematic map for the theme “correlation with issues of development”, sub-theme “changes and deffects”

ID	Changes and defects	ID	Changes and defects (cont. I)	ID	Changes and defects (cont. II)
S4.3	There are smells that are not associated with maintenance problems.	S21.3	Shotgun Surgery (and also God class and God method, in less intensity level) was the most bad smell that were associated with all severity levels of errors in all releases.	S17.1	The number of design problems (smells) increases as the system evolves.
S4.4	Feature Envy and God Method together were associated with time-consuming changes.	S1.2	The change proneness of components with smells is higher than the ones without..	S18.2	Design flaws cant be considered more harmful with respect to software defects.
S4.5	Changes of infected components with ISP Violation is significant larger	S1.3	The code churn of infected classes with god class is significant larger.	S18.3	In some systems there is no design flaw which strongly correlates with defects.
S7.1	The role of code smells on the overall system maintainability is relatively minor.	S1.4	Code smells evolve during the system development.	S18.4	Some systems are characterized by one flaw is correlated with defects much more frequently than all the others.
S7.12	The overlap between problems related to code smells and problems related to artifacts associated to code smells can explain causes of inconsistencies in empirical studies about maintainability.	S1.5	Classes infected seem to need more maintenance than non-infected classes.	S5.1	The change likelihood of god classes is higher than the change likelihood of non-god classes.
S7.13	Some difficulties on maintenance activities are associated to a combination of smells and other characteristics, whereas others were not associated code smells at all.	S9.1	There is a correlation between maintainability factors and code smell.	S5.2	The defect likelihood of god classes is higher than the defect likelihood of non-god classes.
S7.2	Code smells can help to explain and potentially identify some specific difficulties occurred during maintenance beforehand.	S10.1	There is no correlation between clones and bugs.	S5.4	God classes impacts maintainability.
S7.7	Feature Envy (which indicates efferent coupling) or ISP Violation (which indicates afferent coupling) can help to identify cross-cutting concerns situations.	S10.2	There is no evidence that clones are bad smells.	S19.1	Classes with code smells are more change-prone than others.
S21.1	Some bad smells are associated significantly with class error proneness.	S14.1	Taken in isolation, classes exhibiting the identity disharmonies design flaws do not have an increased likelihood to exhibit defects that classes which do not reveal design flaws.	S19.2	Specific smells are more correlated than others to change-proneness.
S21.2	The bad smells will not reveal all classes that contain errors.	S14.2	Classes exhibiting the identity disharmonies design flaws used by their clients shows the increase of likelihood for the clients to exhibit defects, especially for the post-release defects.		

Table A.17: Thematic map for the theme “correlation with issues of development”, sub-themes “effort” and “design quality”

Correlation with issues of development			
ID	**Effort**	**ID**	**Design quality**
S8.1	Without any adjustments for file size and the number of changes smells were associated with more effort than files without these smells. (S8.2)	S4.7	The presence of a code smell may intensify or spread the effects of bad/limited design choices throughout the system.
S8.2	After adjustments for file size and the number of changes code smells are not significantly associated with increased effort. (S8.1)(S8.3)(S8.4)	S19.3	Code smells often related to immature design and implementation.
S8.3	File size and the number of changes explained almost all of the modeled variation in effort. (S8.2)	S7.6	Inconsistent design can be identified by detecting a combination of code smells.
S8.4	There is a correlation between effort and code churn. (S8.2)		
S8.6	Effort may decrease by reducing file size and improving work processes to reduce the number of revisions, instead refactoring.		
S3.4	God class don’t require a higher maintainability effort than non-code smells.		
S2.3	Software characteristics don’t impact effort on god class detection.		
S17.4	Maintenance effort should prioritize Long Method smells over other smells.		

Table A.18: Thematic map for the theme “human aspects”, sub-themes “factors affecting detection”, “decision drivers” and “human evaluation versus software measures”

ID	**Factors affecting detection**	**ID**	**Decision drivers**	**ID**	**Human evaluation versus software measures**
S2.1	Human characteristic impact agreement, effort and drivers’ choice on god class detection.	S3.6	“Moderate agreement” among the subjects detecting GC , for the misplaced method issue (also called driver).	S23.2	Developers evaluations of the smells do not correlate with the used source code metrics.
S23.3	Demographic data (knowledge, role, work experience) seemed to explain some of the variances in smell evaluations.	S3.7	Misplaced methods is the strongest driver for letting the subjects classify a class as a god class.	S23.5	Developers evaluations on the Large Class smell seem to be conflicting, when compared to large class measures.
S16.4	The knowledge acquired in one version effectively supported the identification of related code smells in other versions.	S3.8	Lack of cohesion and complexity are identified as issues that let the human subjects identify a class as a god class.	S22.3	The developers evaluations on code smells dont correlate with related source code metrics.
S22.2	Demographics aspects of the developers affect the code smell evaluations.			S22.6	The developers evaluations of the smells correlated better with the metrics for smells that are simple and easy to spot.
S22.5	Developers with better knowledge of the module evaluated that there is more of the Lazy Class smell that is difficult to spot.				
S24.2	The demographics were not useful predictors neither for evaluating code smells nor the refactoring decision. (and Erradication)				
S25.2	The more details people actually know about code smells, the more concerned they are by the presence of smells in their code.				
S26.2	The domain of an analyzed system is an important factor that should be included in the detection strategies for various code smells				
S26.6	Smells exhibit similar behavior in different software domains.				

Table A.20: Thematic map for the theme “programming”, sub-themes “smell introduction” and “smell introduction”

ID	Smell removal	ID	Smell introduction
S8.5	The developers who revised most often also introduced the most smells.	S8.5	The developers who revised most often also introduced the most smells.
S6.7	There is no clarity on the intent of developers prioritize or even consider the eradication of code smells.	S11.6	Early code anomalies induces architectural anomalies.
S11.7	Architecturally-relevant code anomalies are left in the code when the system evolves.	S17.2	Smells are introduced during the initial design/implementation.
S12.1	The developers resolve code smells for opportunistic reasons.	S20.1	A large proportion of GCs seems to be introduced as a conscious design decision by developers.
S12.2	Code smells of early revisions are resolved within a few revisions.	S20.4	Changes can result in the degradation of GCs.
S17.3	Designers perform refactorings routinely based on their subjective perception of problematic code areas rather than applying them as solutions to identified design problems.	S20.3	The correction of a GC may also move the problem to a different class
S20.2	Specific maintenance activities can eliminate GCs when they are accidents.		
S20.3	The correction of a GC may also move the problem to a different class		
S24.2	The demographics were not useful predictors neither for evaluating code smells nor the refactoring decision.		

Table A.19: Thematic map for the theme “human aspects”, sub-themes “agreement on detection”, “strategy of detection”, “detection difficulty” and “knowledge about smells”

ID	Agreement on detection	ID	Strategy of detection	ID	Detection difficulty	ID	Knowledge about smells
S3.2	The agreement on GC classification and detection is low.	S16.3	Participant with optimistic style investigate code smell candidates using more combinations of views when compared to participants with more conservative styles.	S3.1	The subjects perceive detecting god classes as an easy task.	S25.1	There is not strong understanding about code smells and practical application of these concepts.
S2.2	The agreement on god class detection is low.					S25.3	Duplicated code was by far the most mentioned smell.
S23.1	Software developers show low agreement about smelliness of the source code.					S25.6	Respondents who were extremely or moderately concerned about smell benefits gave as rationale reasons like product evolvability, end-product quality, and developer productivity
S16.2	The participants had different perceptions and judgments during the code smells identification.						
S22.1	Developers have not a uniform opinion on the ”smelliness of the source code.						
S24.1	There is partial concordance among the evaluators in all evaluations.						

Table A.21: Thematic map for the theme “detection”, sub-themes “supporting smell detection”, “smell as a predictor” and “novel smell”

ID	Supporting smell detection	ID	Smell as a predictor	ID	Novel smell
S3.3	Automatic detection accompanied by a manual review increases the overall confidence in the results of metric-based classifiers.	S13.1	The detection strategies are inaccurate in identifying architecturally-relevant code anomalies.	S7.8	A new smell was propose: Multiple Inheritance simulation.
S3.5	If provided with a suitable process, humans can detect code smells in an effective fashion.	S13.3	The detection strategies are more effective in systems where architecture conformance is more strictly enforced in the code.		
S11.4	Developers need support to enhance detection of code anomaly.	S13.5	The detection strategies seem to have a tendency to send developers in wrong directions when addressing code anomalies related to architectural modularity problem.		
S16.1	The use of the visual representation of concerns in the multiple views approach provided useful means to visually spot some kinds of code smells more than others.	S6.4	Expert judgment is most accurate method for assessing system maintainability.		
S22.4	Using source code metrics in conjunction with human evaluations is likely to be the best alternative.	S6.5	Code smells approach can detect design flaws that may be overlooked by experts.		
S25.4	Software professionals who are interested on code smells and anti-patterns expressed a need for better support, including tools, during the software evolution cycle.	S6.6	Software measures and expert judgment addressed different aspects of maintainability in a system, and combining them can lead to more complete evaluations of maintainability.		
S25.5	Respondents indicated that they use technical blogs, programmer forums, colleagues and industry seminars as their main sources of information.	S9.2	Code smells would need to be complemented with alternative approaches such as semantic analysis and manual inspection for address maintainability factors		
S26.3	Accuracy of code smells detection rules used by the tools can be improved by exploiting knowledge about both the domain and the design of a system.	S9.3	Code smells can cover a more heterogeneous spectrum of factors than software metrics and expert judgment individually.		
		S23.4	The use of the smells for internal software quality assessment should be questioned		
		S19.4	Smells can provide to developers recommendations easier to understand than what metric profiles can do.		
		S19.5	Smells are not replacement to metrics in the ability of building change-proneness or fault-proneness prediction models		

Table A.22: Thematic map for the theme “other correlations”, sub-themes “inter-smell relation”, “metrics versus smells”, “smell density” and “frequency of smells”

Other correlations							
ID	Inter-smell relation	ID	Metrics versus smells	ID	Smell density	ID	Frequency of smells
S4.1	There is interaction between code smells.	S3.9	Automated metric-based pre-selection of smells decreases the effort spent on manual code inspection.	S6.2	Code smell density is likely to be inaccurate when comparing size differing systems.	S15.2	The larger systems have a larger proportion of God and Brain Classes.
S4.2	There is a correlation between inter-smells smells and maintenance problems.	S6.1	Code smells are strongly influenced by size (LOC).	S6.3	Code smell density distinguishes maintainability across similar sized systems.	S18.1	Some design flaws are more frequent than others.
S4.6	Studies into the effects of inter-smell relations are a topic that deserves more attention.	S7.11	Wide coupling displayed Feature Envy or ISP Violation and to some extent Shotgun Surgery.	S15.3	Size normalizing impacts evaluation of change frequency of God class and Brain class.	S18.5	Feature envy is the most frequent design flaw, but it is not the most correlated with software defects.
S7.10	Observing combinations of code smells could be useful to discriminate instances of God Classes that are potentially problematic, against instances of no problematic god class.	S7.9	Large size exhibited either the God Class smell or the God Method smell and in most cases a combination of both smells.	S5.5	There isn’t a linear relationship between class size and change likelihood that justifies LOC-normalization.	S26.1	Some smells are more prevalent than others
S7.13	Some difficulties on maintenance activities are associated to a combination of smells and other characteristics, whereas others were not associated code smells at all.	S1.1	There is a correlation between size and number of: god class and shotgun surgery.				
S7.5	Interaction effects amongst collocated smells and coupled smells should be taken into account during analysis.	S24.3	Simple code smells and source code metrics have a relationship.				
S7.6	Inconsistent design can be identified by detecting a combination of code smells.	S26.4	There is correlation between several code smells and selected metrics.				
		S26.5	The strength of correlation between code smells and metrics varies with respect to the domain				

References

[1] A. J. Riel, Object-Oriented Design Heuristics, 1st Edition, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1996.

[2] M. Fowler, Refactoring: improving the design of existing code, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[3] M. Lanza, R. Marinescu, S. Ducasse, Object-Oriented Metrics in Practice, Springer-Verlag New York, Inc., 2005.

[4] D. Sjøberg, A. Yamashita, B. Anda, A. Mockus, T. Dyba, Quantifying the effect of code smells on maintenance effort, IEEE Transactions on Software Engineering 39 (8) (2013) 1144–1156.

[5] I. Macia, J. Garcia, D. Popescu, A. Garcia, N. Medvidovic, A. von Staa, Are automatically-detected code anomalies relevant to architectural modularity?: An exploratory analysis of evolving systems, in: Proc. of the 11th Annual International Conference on Aspect-oriented Software Development (AOSD), 2012, pp. 167–178.

[6] S. M. Olbrich, D. S. Cruzes, D. I. K. Sjøberg, Are all code smells harmful? a study of god classes and brain classes in the evolution of three open source systems, in: Proc. of the 26th International Conference on Software Maintenance (ICSM), 2010, pp. 1–10.

[7] A. Yamashita, How good are code smells for evaluating software maintainability? results from a comparative case study, in: 29th IEEE International Conference on Software Maintenance (ICSM), 2013, pp. 566–571.

[8] M. Zhang, T. Hall, N. Baddoo, Code bad smells: A review of current knowledge, Software Maintenance and Evolution: Research and Practice 23 (3) (2011) 179–202.

[9] B. Kitchenham, S. Charters, Guidelines for performing systematic literature reviews in software engineering (version 2.3), Ebse technical report ebse-2007-01, Keele University and Durham University (2007).

[10] M. Dixon-Woods, S. Agarwal, B. Young, D. Jones, A. Sutton, Integrative approaches to qualitative and quantitative evidence, Health Development Agency (2004) 1–35.

[11] D. Cruzes, T. Dybå, Recommended steps for thematic synthesis in software engineering, in: Proc. of the 5th International Symposium on Empirical Software Engineering and Measurement (ESEM), 2011, pp. 275–284.

[12] B. Kitchenham, R. Pretorius, D. Budgen, O. Pearl Brereton, M. Turner, M. Niazi, S. Linkman, Systematic literature reviews in software engineering - a tertiary study, Information and Software Technology 52 (8) (2010) 792–805.

[13] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, M. Khalil, Lessons from applying the systematic literature review process within the software engineering domain, Journal of Systems and Software 80 (4) (2007) 571–583.

[14] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, A. Wesslén, Experimentation in Software Engineering, Springer Berlin Heidelberg, 2012.

[15] M. Skoglund, P. Runeson, Reference-based search strategies in systematic reviews, in: Proceedings of the 13th International Conference on Evaluation and Assessment in Software Engineering (EASE), 2009, pp. 31–40.

[16] D. Cruzes, T. Dybå, Research synthesis in software engineering: A tertiary study, Information and Software Technology 53 (5) (2011) 440 – 455.

[17] J. Creswell, Educational Research: Planning, Conducting, and Evaluating Quantitative and Qualitative Research, Merrill, 2005.

[18] W. Li, R. Shatnawi, An empirical study of the bad smells and class error probability in the post-release object-oriented system evolution, Journal of Systems and Software 80 (7) (2007) 1120–1128.

[19] S. Olbrich, D. Cruzes, V. Basili, N. Zazworka, The evolution and impact of code smells: A case study of two open source systems, in: 3rd International Symposium on Empirical Software Engineering and Measurement (ESEM), 2009, pp. 390–400.

[20] A. Yamashita, L. Moonen, Exploring the impact of inter-smell relations on software maintainability: An empirical study, in: Proceddings of the 35th International Conference on Software Engineering (ICSE), 2013, pp. 682–691.

[21] M. Mäntylä, An experiment on subjective evolvability evaluation of object-oriented software: explaining factors and interrater agreement, in: Proceedings of the 4th International Symposium on Empirical Software Engineering (ISESE), 2005.

[22] M. Mäntylä, C. Lassenius, Subjective evaluation of software evolvability using code smells: An empirical study, Empirical Software Engineering 11 (3) (2006) 395–431.

[23] J. Schumacher, N. Zazworka, F. Shull, C. Seaman, M. Shaw, Building empirical support for automated code smell detection, in: Proc. of the 4th International Symposium on Empirical Software Engineering and Measurement (ESEM), 2010, pp. 1–10.

[24] L. Moonen, A. Yamashita, Do code smells reflect important maintainability aspects?, in: Proc. of 28th the International Conference on Software Maintenance (ICSM), 2012, pp. 306–315.

[25] M. V. Mäntylä, C. Lassenius, Drivers for software refactoring decisions, in: Proc. of 5th International Symposium on Empirical Software Engineering (ISESE), 2006, pp. 297–306.

[26] F. Simon, F. Steinbruckner, C. Lewerentz, Metrics based refactoring, in: Proc. of 5th European Conference on Software Maintenance and Reengineering (CSMR), 2001, pp. 30–38.

[27] C. Parnin, C. Görg, O. Nnadi, A catalogue of lightweight visualizations to support code smell inspection, in: Proc. of the 4th international Smposium on Software visualization (SOFTVIS), 2008, pp. 77–86.

[28] E. Murphy-Hill, A. P. Black, An interactive ambient visualization for code smells, in: Proc. of the 5th International Symposium on Software visualization (SOFTVIS), 2010, pp. 5–14.

[29] G. Carneiro, M. Silva, L. Maia, E. Figueiredo, C. Sant'Anna, A. Garcia, M. Mendonça, Identifying code smells with multiple concern views, in: Proc. of the 1th Brazilian Conference on Software: Theory and Practice (CBSOFT), 2010, pp. 128–137.

[30] D. Rapu, S. Ducasse, T. Girba, R. Marinescu, Using history information to improve design flaws detection, in: Proc. of 8th European Conference on Software Maintenance and Reengineering (CSRM), 2004, pp. 223–232.

[31] B. A. Price, R. M. Baecker, I. S. Small, An introduction to software visualization, in: J. Stasko, J. Dominique, M. Brown, B. Price (Eds.), Software Visualization, MIT Press, London, England, 1998, pp. 4–26.

[32] F. Palomba, G. Bavota, M. D. Penta, R. Oliveto, A. D. Lucia, Do they really smell bad? a study on developers' perception of bad code smells, in: Proc. of the 30th IEEE International Conference on Software Maintenance and Evolution (ICSME), 2014.

[33] M. V. Mantyla, J. Vanhanen, C. Lassenius, Bad smells humans as code critics, in: Proc. of the 20th IEEE International Conference on Software Maintenance (ICSM), 2004, pp. 399–408.

[34] R. Marinescu, Measurement and quality in objectoriented design, Ph.D. thesis, Politehnica University of Timisoara (2002).

[35] O. Dieste, A. G. Padua, Developing search strategies for detecting relevant experiments for systematic reviews, in: Proc. of the 1th ESEM, 2007, pp. 215–224.

[36] K. Dhambri, H. Sahraoui, P. Poulin, Visual detection of design anomalies, in: Proceeding of the 12th European Conference on Software Maintenance and Reengineering (CSMR), 2008, pp. 279–283.

[37] I. Macia, A. Garcia, A. von Staa, J. Garcia, N. Medvidovic, On the impact of aspect-oriented code smells on architecture modularity: An exploratory study, in: Proc. of the 5th Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS), 2011, pp. 41–50.

[38] I. Macia, A. Garcia, A. von Staa, An exploratory study of code smells in evolving aspect-oriented systems, in: Proc. of the 10th International Conference on Aspect-oriented Software Development (AOSD), 2011, pp. 203–214.

[39] I. Macia, R. Arcoverde, A. Garcia, C. Chavez, A. von Staa, On the relevance of code anomalies for identifying architecture degradation symptoms, in: Proc. of the 16th European Conference on Software Maintenance and Reengineering (CSMR), 2012, pp. 277–286.

[40] S. Easterbrook, J. Singer, M.-A. Storey, D. Damian, Selecting empirical methods for software engineering research, in: F. Shull, J. Singer, D. Sjøberg (Eds.), Guide to Advanced Empirical Software Engineering, Springer London, 2008, pp. 285–311.

[41] D. I. K. Sjøberg, J. E. Hannay, O. Hansen, V. By Kampenes, A. Karahasanovic, N.-K. Liborg, A. C. Rekdal, A survey of controlled experiments in software engineering, IEEE Transaction on Software Engineering 31 (9) (2005) 733–753.

[42] A. Jedlitschka, M. Ciolkowski, D. Pfahl, Reporting experiments in software engineering, in: F. Shull, J. Singer, D. I. K. Sjberg (Eds.), Guide to Advanced Empirical Software Engineering, Springer London, 2008, pp. 201–228.

[43] T. Dybå, T. Dingsøyr, Empirical studies of agile software development: A systematic review, Information and Software Technology 50 (910) (2008)

833 – 859.

[44] B. Kitchenham, D. I. K. Sjøberg, O. P. Brereton, D. Budgen, T. Dybå, M. Host, D. Pfahl, P. Runeson, Can we evaluate the quality of software engineering experiments?, in: Proc. of the 4th International Symposium on Empirical Software Engineering and Measurement (ESEM), 2010, pp. 2:1–2:8.

[45] A. Nguyen-Duc, D. S. Cruzes, R. Conradi, The impact of global dispersion on coordination, team performance and software quality a systematic literature review, Information and Software Technology 57 (0) (2015) 277 – 294.

[46] C. G. R. H. Basili, V.R., Goal question metric paradigm, in: Encyclopedia of Software Engineering, 1994, pp. 528–532.

[47] J. A. Santos, M. Mendonça, C. Silva, An exploratory study to investigate the impact of conceptualization in god class detection, in: Proc. of the 17th International Conference on Evaluation and Assessment in Software Engineering (EASE), 2013, pp. 48–59.

[48] N. Zazworka, M. A. Shaw, F. Shull, C. Seaman, Investigating the impact of design debt on software quality, in: Proc. of the 2nd Workshop on Managing Technical Debt (MTD), 2011, pp. 17–23.

[49] A. Yamashita, S. Counsell, Code smells as system-level indicators of maintainability: An empirical study, Journal of Systems and Software 86 (10) (2013) 2639 – 2653.

[50] A. Yamashita, L. Moonen, To what extent can maintenance problems be predicted by code smell detection? an empirical study, Information and Software Technology 55 (12) (2013) 2223 – 2242.

[51] F. Rahman, C. Bird, P. Devanbu, Clones: what is that smell?, Empirical Software Engineering 17 (4-5) (2012) 503–530.

[52] R. Peters, A. Zaidman, Evaluating the lifespan of code smells using software repository mining, in: Proc.of the 16th European Conference on Software Maintenance and Reengineering (CSMR), 2012, pp. 411–416.

[53] R. Marinescu, C. Marinescu, Are the clients of flawed classes (also) defect prone?, in: Proc. of the 11th IEEE International Working Conference on Source Code Analysis and Manipulation (SCAM), 2011, pp. 65–74.

[54] A. Chatzigeorgiou, A. Manakos, Investigating the evolution of bad smells in object-oriented code, in: Proc. of the 7th International Conference on the Quality of Information and Communications Technology (QUATIC), 2010, pp. 106–115.

[55] M. D'Ambros, A. Bacchelli, M. Lanza, On the impact of design flaws on software defects, in: Proc. of the 10th International Conference on Quality Software (QSIC), 2010, pp. 23–31.

[56] F. Khomh, M. Di Penta, Y.-G. Guéhéneuc, An exploratory study of the impact of code smells on software change-proneness, in: Proc. of the 16th Working Conference on Reverse Engineering (WCRE)., 2009, pp. 75–84.

[57] S. Vauche, F. Khomh, N. Moha, Y.-G. Guéhéneuc, Tracking design smells: Lessons from a study of god classes, in: Proc. of the 16th Working Conference on Reverse Engineering (WCRE)., 2009, pp. 145–154.

[58] A. Yamashita, L. Moonen, Do developers care about code smells? an exploratory survey, in: Proc. of the 20th Working Conference on Reverse Engineering (WCRE), 2013, pp. 242–251.

[59] F. Fontana, V. Ferme, A. Marino, B. Walter, P. Martenka, Investigating the impact of code smells on system's quality: An empirical study on systems of different application domains, in: Proc. of the 29th IEEE International Conference on Software Maintenance (ICSM), 2013, pp. 260–269.

[60] H. K. Jonathan I. Maletic, Expressiveness and effectiveness of program comprehension: Thoughts on future research directions, in: Frontiers of Software Maintenance (FoSM), 2008, pp. 31–37.

[61] J. A. Santos, M. G. de Mendonça, C. P. dos Santos, R. L. Novais, The problem of conceptualization in god class detection: agreement, strategies and decision drivers, Journal of Software Engineering Research and Development (JSERD) 2 (11) (2014) 1–33.

[62] J. A. Santos, M. Mendonça, Identifying strategies on god class detection in two controlled experiments, in: Proc. of the 26th International Conference on Software Engineering and Knowledge Engineering (SEKE), 2014, pp. 244–249.

[63] P. A. da Mota Silveira Neto, I. do Carmo Machado, J. D. McGregor, E. S. de Almeida, S. R. de Lemos Meira, A systematic mapping study of software product lines testing, Information and Software Technology 53 (5) (2011) 407 – 423.

[64] C. Cassell, G. Symon, Qualitative methods in organizational research: a practical guide, Sage, 1994.