Ciência de Dados e I.A.
Escola de Matemática Aplicada
Fundação Getúlio Vargas

Engenharia de Requisitos

TCC

An Approach for Extracting UML Diagram from Object-Oriented Program Based on J2X

Aluno: Isabela Yabe
Orientador: Rafael de Pinho André
Escola de Matemática Aplicada, FGV/EMAp
Rio de Janeiro - RJ.

Rio de Janeiro, 2025

1 Revisão literária

Artigo revisado Zhang (2016):

A revisão tem o objetivo de compreender o estado da arte das abordagens de engenharia reversa que partem de código-fonte e produzem artefatos de alto nível, como diagramas UML. Para garantir uma análise sistemática e comparável entre diferentes propostas, foram definidas perguntas de pesquisa (*Research Questions — RQs*) que orientam a coleta e síntese dos dados extraídos dos estudos selecionados.

- **RQ1.** Em quais linguagens e domínios as abordagens que partem de código-fonte foram aplicadas?
- **RQ2.** Quais modelos/artefatos de alto nível são gerados?
- **RQ3.** Qual aspecto é privilegiado (estático, dinâmico, híbrido) e com qual objetivo (compreensão, redocumentação, migração, qualidade)?
- **RQ4.** Quais técnicas e transformações viabilizam a passagem do código para o modelo de alto nível?
- **RQ5.** Quais ferramentas/frameworks são utilizados?
- **RQ6.** Como as abordagens são validadas e com que qualidade prática?

2 RQ1. Em quais linguagens e domínios as abordagens que partem de código-fonte foram aplicadas?

A implementação e avaliação concentram-se na linguagem Java. Os autores testam a abordagem em cinco pequenos sistemas orientados a objetos de propósito geral: *Minesweeper*, *Blog*, *PayrollSys*, *eLib* e *myAlgsLib*. Esses casos evidenciam a aplicação em programas de escopo reduzido e natureza genérica, sem vínculo a um domínio específico.

Embora a experimentação esteja restrita ao ecossistema Java, o artigo destaca que o uso do intermediário J2X foi concebido para "isolar diferenças de linguagem" e permitir a extensão do método a outras linguagens.

3 RQ2. Quais modelos/artefatos de alto nível são gerados?

O método gera dois artefatos UML de alto nível: o diagrama de classes (estrutura) e o diagrama de sequência (comportamento de interação). O processo ocorre em três etapas: (i) transformação do código em representação J2X, (ii) extração estrutural para o diagrama de classes e (iii) análise de fluxo (OFG e CFG) para construir o diagrama de sequência. Assim, a cadeia de transformação é: *código* $\rightarrow$ *J2X* $\rightarrow$ *OFG/CFG* $\rightarrow$ *UML*.

4 RQ3. Qual aspecto é privilegiado (estático, dinâmico, híbrido) e com qual objetivo?

A abordagem é predominantemente Estático, operando sobre o código-fonte sem execução. O objetivo central é a Compreensão e Redocumentação de sistemas OO, elevando o nível de abstração por meio da geração de diagramas UML de classes e sequência.

Extensão proposta (minha proposta). Os autores reconhecem a possibilidade de um futuro híbrido (estático + dinâmico) para aumentar a acurácia. Como extensão, propomos um Híbrido que combine a análise estática com uma *análise de intenção* (o que o software está intencionado a fazer), inferida de indícios semânticos do código (nomes, contratos, comentários) e correlacionada a casos de uso. A hipótese é que a integração entre (Estático) e intenção de alto nível melhore a precisão na identificação de interações relevantes e reduza ambiguidades na geração de UML Sequência, mantendo o baixo custo de coleta (sem execução).

5 RQ4. Quais técnicas e transformações viabilizam a passagem do código para o modelo de alto nível?

O processo segue um pipeline de quatro fases principais:

1. **Código → AST → J2X (IR):** parsing e anotação semântica do código em linguagem intermediária XML (J2X), que padroniza elementos e isola diferenças de linguagem.

2. **J2X → metadados → UML Classe:** extração de classes, interfaces, métodos e campos, mapeando relações de generalização, implementação, associação e dependência.

3. **OFG (Object Flow Graph):** refinamento das relações por rastreamento de fluxos de objetos, reduzindo falsos positivos/negativos e tratando polimorfismo.

4. **J2X → CFG → (OFG + CFG) → UML Sequência:** Para o diagrama de sequência, o método modela chamadas e estruturas de controle em um *Control Flow Graph* (CFG) e o combina com o OFG a fim de identificar objetos participantes, mensagens e condições/loops. *Efeito prático:* mesmo sem execução, a combinação OFG+CFG preserva ordem/condição das interações, permitindo gerar *UML Sequência* com *lifelines*, *messages* e *fragments* (alt/opt/loop)

Síntese (RQ4). *Intermediário:* J2X (XML). *Mapeamentos:* estrutura (gen./impl./assoc./dep.). *Precisão:* OFG e CFG reduzem ambiguidades. *Saída:* UML Classe e Sequência.

6 RQ5. Quais ferramentas/frameworks são utilizados?

O ecossistema técnico inclui:

- **J2UML:** ferramenta desenvolvida pelos autores que orquestra a extração e geração dos diagramas UML.
- **JavaCC:** gerador de analisadores automáticos usado para construir a AST a partir do código-fonte.
- **DOM4J:** biblioteca para manipulação eficiente do XML que representa o modelo intermediário J2X.
- **J2X (DTD/XML):** linguagem intermediária padronizada para estruturar e armazenar informações semânticas do programa.

Os experimentos foram realizados em um ambiente Windows 32-bit, com 3GB de memória e CPU Intel Core 2 Duo a 2,93GHz, garantindo reprodutibilidade dos resultados.

Síntese (RQ5). *Ferramentas:* J2UML, JavaCC, DOM4J. *Representação:* J2X (DTD/XML). *Ambiente:* Windows 32-bit, CPU 2.93GHz, 3GB RAM.

7 RQ6. Como as abordagens são validadas e com que qualidade prática?

A validação foi conduzida experimentalmente em cinco sistemas orientados a objetos de pequeno porte. A ferramenta J2UML foi aplicada para extrair diagramas de classes, e os resultados foram comparados manualmente com um *ground truth* derivado do código-fonte. As métricas consideradas foram tempo de execução e acurácia da extração.

A acurácia foi calculada como a razão entre elementos corretamente extraídos e o total de elementos esperados, apresentando valores entre 96,4–100% para classes e 65,0–90,4% para relações. O desempenho foi considerado satisfatório, e os autores destacam que o uso de OFG melhora a precisão nas relações. Apesar disso, o estudo limita-se a casos didáticos, sem análise estatística ou avaliação em sistemas de larga escala.

Síntese (RQ6). *Desenho:* estudo experimental com 5 sistemas OO. *Métricas:* tempo e acurácia (classes e relações). *Resultados:* classes 96,4–100%; relações 65,0–90,4%. *Qualidade prática:* boa precisão estrutural; limitações de escala e avaliação.

Autores / Referência	Linguagem / Domínio	Modelo Gerado	Aspecto	Técnica / Transformação	Ferramenta / Framework	Validação / Estudo de Caso
Zhang (2016)	Java; pequenos sistemas OO (eLib, Minesweeper, Blog, PayrollSys, myAlgsLib)	UML Classe; UML Sequência	Estático — compreensão e redocumentação	Código→AST→J2X; sentenças simplif.→OFG; CFG+OFG→Sequência	J2X; JavaCC; DOM4J; J2X (DTD/XML)	5 casos pequenos; acurácia: classes 96,4–100%; relações 65,0–90,4%

Tabela 1: Resumo da abordagem MDRE baseada em J2X (Zhang (2016))

Referências

ZHANG, H. An Approach for Extracting UML Diagram from Object-Oriented Program Based on J2X. Versão inglesa. *In:* INTERNATIONAL Forum on Mechanical, Control and Automation (IFMCA 2016). Changchun, China: Atlantis Press, 2016. p. 266–276. Published in Advances in Engineering Research, vol. 113.