ASPECTOS DE EPISTEMOLOGIA DO SOFTWARE

Gilberto Câmara

Departamento de Processamento de Imagens - DPI
Instituto de Pesquisas Espaciais - INPE
Caixa Postal 515
12.201 - Sao José dos Campos - SP

RESUMO

Este trabalho exploratório procura analisar a natureza do processo de desenvolvimento de software, do ponto de vista metodológico. Enfase especial é dada a problemas epistemológicos, com uma apresentação de várias questões fundamentais. Inicialmente, é feita uma distinção entre as duas principais correntes de Engenharia de Software. As concepções de filosofia da linguagem são a seguir utilizadas para procurar indicar o alcance e as limitações do atual estado da arte de métodos analítico-formais. Ao final, são fornecidas indicações de caráter prático.

ABSTRACT

This paper explores the very nature of the software development process, from a methodological point of view. Special emphasis is placed on epistemic problems, as many fundamental questions are presented. First, a distinction between the two main schools of Software Engineering is presented. Concepts of Philosophy of Language are then used to indicate the paths and pitfalls of the current state-of-the-art formal methods. Finally, some practical consequences are derived.

1. PRELIMINARES

O desenvolvimento de métodos de Engenharia de Software tem sido feito, muitas vezes, deixando de lado a natureza intrínseca do assunto. e sem levar em conta os fundamentos epistemológicos do processo de desenvolvimento de software. A natureza instrumental da atividade de programar e a formação nem sempre completa dos profissionais da área contribuem para que se confunda o estudo de Engenharia de Software com a mera apresentação de técnicas de desenvolvimento de programas.

A transformação da Engenharia de Software em uma disciplina científica só poderá acontecer a partir do um conhecimento da sua essência. Assim, é importante lançar luz sobre os problemas conceituais envolvidos ao se desenvolver software. Tal é o objetivo deste trabalho, organizado em quatro itens:

- inicialmente. é traçada um visão geral das duas grandes abordagens do desenvolvimento de software: o pragmatismo (empirismo) e a escola analítico-formal;
- apresentam-se a seguir os pressupostos filosóficos de cada corrente;
- a natureza do desenvolvimento de software é então analisada, procurando verificar qual a contribuição (e os possíveis problemas) de cada abordagem;
- finalmente, algumas análises de ordem prática são feitas.

O trabalho deve ser visto como um conjunto de idéias preliminares sobre um problema amplo e tem apenas a pretensão de levantar questões de larga abrangência.

2. A GRANDE DICOTOMIA

É possivel identificar duas grandes abordagens para o problema do desenvolvimento de software: a teoria empírica, representada pela escola pragmática de Brooks (1975) e a corrente analítico-formal (Manna, 1974; Haeberer et alii., 1989). Esta dicotomia reflete antes diferenças de experiência e formação básica que um completo antagonismo.

No primeiro caso, o que se procura fazer é constatar os problemas presentes, ganhando "insight" e fornecendo instrumentos e conselhos de ordem prática. Típicas desta abordagem são estudos sobre técnicas de programação, organização de equipes e métricas. A escola pragmática parte da constatação objetiva que os requisitos externos de um sistema computacional são complexos, imperfeitamente formulados e sujeitos a mudanças. Um sistema de software tem profundas diferenças conceituais com computadores, prédios ou automóveis (Brooks, 1987): o número de estados possível é muitas ordens de grandeza superior ao do mais complexo hardware e as interações entre as partes são frequentemente não lineares.

A escola pragmática considera que a essência do software é sua complexidade e não é possível construir modelos abstratos que representem corretamente os requisitos informais de um sistema (Brooks, 1987).

A escola analítico-formal procura fundamentar teóricamente o problema, partindo da concepção (Hoare, 1969) de que "a programação de computadores é uma ciência exata, pois todas as propriedades de um programa e todas as consequências de sua execução podem, em princípio, obter-se a partir do seu próprio texto, por meio de raciocínio puramente dedutivo".

A escola analítico-formal concebe o processo de desenvolvimento de software como uma sequência de transformações. Dada uma proposição informal do problema, produz-se para uma especificação formal, a qual permite derivar rigorosamente um programa que o resolve (Haeberer et alii, 1989). A ênfase na especificação formal é central a esta abordagem. Uma parte importante do processo do desenvolvimento de software passa a ser traduzir fielmente uma verbalização informal para um formalismo rigoroso.

Ambas as concepções têm raízes filosóficas: a primeira, na tradição das ciências humanas e sociais; a segunda, na escola positivista da filosofia da ciência, cuja maior expressão foi o Círculo de Viena.

As bases conceituais de cada alternativa serão apresentadas brevemente na próxima seção. No que segue, o termo "paradigma" tem o significado utilizado por Kuhn (1962), e mais tarde esclarecido pelo mesmo autor (Kuhn, 1974) e também chamado de "matriz disciplinar": um conjunto característico de crenças e preconceitos praticados pelos membros de um corpo científico, num determinada época. A descrição é sumária e esquemática, realizada com o único objetivo de esclarecer problemas conceituais.

3. PARADIGMAS CONCEITUAIS EM FILOSOFIA DA CIÊNCIA

3.1 MÉTODO HIPOTÉTICO-DEDUTIVO

Em seu esquema clássico (o proposto pelo Círculo de Viena) o método hipotético-dedutivo procura incorporar o simbolismo da lógica matemática para formular um método geral aplicável a todo o conjunto das atividades científicas, cuja inspiração mais imediata foi a axiomatização da maior parte da matemática obtida por Whitehead e Russel. Em essência, esta escola propunha que fossem construídas teorias axiomáticas, formuladas com base na lógica matemática, para os vários campos do conhecimento.

Ao ser aplicado a um particular corpo científico, o método hipotético-dedutivo preconiza derivar leis teóricas (gerais) e obter previsões empíricas sobre comportamentos observáveis dos fenômenos a serem explicados. Este método, em sua forma original, baseia-se na possibilidade da separação entre sentenças **sintéticas** e **analíticas**. Os enunciados analíticos são aqueles cuja verdade ou falsidade pode ser verificada a partir de estreita equivalência entre seu significado e uma expressão lógica correspondente. Para o Círculo de Viena, uma teoria científica deve conter apenas enunciados analíticos, para que suas hipóteses possam ser comprovadas.

As propostas do Círculo de Viena (também chamadas de "Received View") tem sido debatidas e criticadas intensamente (veja-se, exemplo, Suppe, 1974). É atualmente aceita pela maior parte dos filósofos da ciência a idéia de que as propostas do Círculo de Viena são inadequadas para o conjunto de todas as teorias normalmente aceitas como científicas. Por exemplo, a teoria da evolução de Charles Darwin não pode ser encaixada no rígido esquema do Círculo de Viena. Dentre as várias propostas alternativas, uma das formulações mais gerais e menos restritivas é a proposta por Popper, em seus clássicos livros "*A Lógica da Pesquisa Científica*" (Popper, 1959) e "*Conjecturas and Refutações*" (Popper, 1965): "as teorias são conjecturas genuínas e são sérias tentativas de descobrir a verdade... embora não possamos saber, nem talvez nunca iremos saber, se são ou não corretas". Assim, uma teoria nunca pode ser confirmada, mas apenas **refutada**; as teorias são apenas conjecturas, e a ciência deve permitir sua proliferação tanto quanto possível, sujeitando uma grande variedade de teorias à falsificação empírica.

3.2 OS MÉTODOS DAS CIENCIAS HUMANAS

A perspectiva das ciências sociais e humanas é bastante distinta da abordagem do método hipótetico-dedutivo. Para citar um caso clássico, a psicanálise não pode ser entendida como ciência no sentido proposto por Popper, pois não é possível refutar a teoria de Freud. A epistemologia recente das ciências sociais aceita as propostas de Popper para as ciências "exatas" ou naturais e propõe uma visão distinta para o caso das ciências sociais (veja-se, por exemplo, as propostas de Habermas (1976) em "Conhecimento e Interesse").

Em todas as ciências sociais, a presença do homem como agente da história adiciona grande complexidade às disciplinas. O uso de mecanismos formais nas ciências humanas e sociais exige simplificações de tal ordem que muitas vezes não permite a correta representação dos problemas.

A dificuldade em produzir teorias genericamente aplicáveis faz com que muitos dos métodos em ciências sociais preconizem a busca de **fatos isolados** que expliquem por que se verificaram outros **fatos isolados**. (Para um exemplo desta perspectiva na análise histórica, veja-se Guinzburg, 1989). Em lugar dos formalismos da matemática, o paradigma aqui é o da solução de um caso criminal, histórico ou médico. Aqui, acontecimentos aparentemente desimportantes têm tanta importância para compreender os fenômenos quanto leis de natureza mais geral. A interpretação do fenômeno torna-se tão relevante quanto a informação em si. Neste caso, pode-se falar em métodos **indiciários**: o conhecimento é construído de maneira intuitiva, a partir de indícios recolhidos sobre cada caso particular.

Mesmo quando se utiliza uma visão de mundo globalizante e genérica (como o marxismo), o que se dispõe é um conjunto de conceitos globais, cuja aplicação prática depende de uma análise da realidade concreta (Não por acaso, Gramsci se referia ao marxismo como "a filosofia da práxis").

Deste modo, no caso de ciências humanas e sociais, o método hipotético-dedutivo não é geralmente aplicável, o que resulta, na prática, em uma plêiade de alternativas (pense-se no caso da economia ou da sociologia).

4. MODELOS CONCEITUAIS E SEUS PROBLEMAS

4.1 DA NATUREZA DO SOFTWARE

Que atividades são envolvidas no desenvolvimento de software? O software pode ser visto como uma ciência no sentido de Popper? ou será que os analistas de software trabalham preferencialmente com paradigmas divinatórios como no caso da medicina?

Prover respostas concretas a estas perguntas é extremamente difícil devido à natureza contraditória do desenvolvimento de software. De um lado, a difusão usual do conhecimento neste campo se processa de maneira indiciária: muitos dos livros utilizados ensinam técnicas que apresentam validade apenas em casos concretos e particulares ou

que formulam princípios gerais ("aprenda com exemplos de bons sistemas"). Por outro lado, todo programa de computador pode ser entendido como uma teoria ("conjectura") no sentido de Popper, pois pode produzir resultados empíricos, cuja observação poderá falsear a "teoria" (i.e. o programa). Esta dicotomia se reflete no fato de existirem duas grandes abordagens para o problema.

O fato dos problemas a serem resolvidos por um sistema computacional terem origem na realidade histórica e social concreta fornece ponderável argumento à escola pragmática. No entanto, deve-se reconhecer que a corrente formal da Engenharia de Software procura dar a esta disciplina uma fundamentação sólida, dentro da tradição das ciências exatas. No entanto, esta abordagem tem problemas epistemológicos ainda não completamente resolvidos, como se verá a seguir.

4.2 PROBLEMAS DO MODELO FORMAL

O objetivo geral dos modelos formais em Engenharia de Software é obter, a partir dos requisitos do sistema - expressos em linguagem natural - uma formalização que seja exatamente equivalente. Assim posto, este é um problema insolúvel, pois proposições expressas de maneira vaga normalmente são acompanhadas de descrições inexatas dos resultados esperados. Apenas uma especificação completamente formal é passível de ser validada, o que não é o caso mais geral no desenvolvimento de software.

A escola formal procura enfrentar este obstáculo ao usar a decomposição, onde o problema a ser resolvido é dividido em subproblemas. Espera-se que pelo menos um dos subproblemas seja verificável formalmente, isto é, possa ser expresso por *enunciados analíticamente determinados*, cuja veracidade ou falsidade seja passível de dedução lógica.

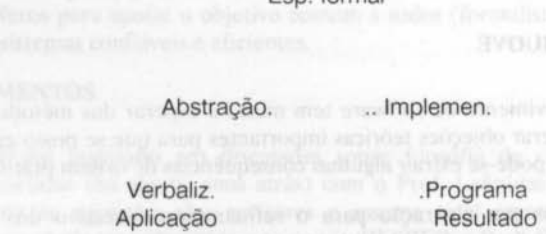
Os métodos formais baseiam-se, deste modo, na hipótese de que é possível traduzir (ainda que parcialmente) verbalizações para especificações formais. Tal hipótese está estreitamente ligada a um delicado problema metodológico, bastante abordado em filosofia da linguagem: **o problema do significado** (Kutschera, 1975), que corresponde à pergunta: "como podemos ter certeza de que duas sentenças são equivalentes ? ". No caso de Engenharia de Software, questiona-se: "É possível garantir que a especificação formal consegue ainda representar perfeitamente o problema original?".

Importantes correntes da filosofia da linguagem sustentam que é o problema do significado é insolúvel, sendo impossível garantir a **precisão linguística**; ou seja, nunca se poderia ter certeza de interpretar corretamente o significado de sentenças escritas em linguagem informal. Em seu trabalho "Investigações Filosóficas", Wittgenstein (1979) sustenta que a linguagem apenas nos fornece uma interpretação parcial da realidade, não existindo um sentido exato de uma frase, escondido na linguagem corrente, a ser determinado analíticamente. Como coloca Wittgenstein, " *o significado de uma palavra é seu uso na linguagem*".

Outro dos pilares da concepção formal é a possibilidade de distinguir entre sentenças analíticas e sintéticas. Este é também um aspecto questionável. Filósofos da linguagem

como Quine (1960 e 1962) argumentam que é impossível garantir tal distinção; isto implica que não se pode ter certeza que uma representação formal mantenha o mesmo significado da frase original, pois o significado das expressões linguisticas depende do contexto e da situação concreta.

Para exemplificar os problemas enfrentados pela escola formalista, considera-se aqui um paradigma dos modelos recentes para a formalização do processo de desenvolvimento de software: o "modelo PW" proposto por Lehman e analisado por Haeberer, Veloso e Baum (1989). Numa visão esquemática, o modelo PW propõe uma estrutura triangular invertida, com a especificação formal no ápice e a verbalização informal dos requisitos e o programa no outros vértices.



O modelo divide o processo de programar em duas fases distintas: a abstração (passar de uma descrição informal a uma especificação formal) e a implementação (gerar um programa que resolva o problema colocado formalmente).

O modelo PW propõe gerar uma especificação formal a partir de uma sequência de passos, onde cada passo sucessivo produziria uma especificação formal parcial, onde parte da descrição ainda é feita de maneira informal. Neste processo, usam-se as idéias de **decomposição por abstração** (Liskov e Guttag, 1986) para obter, a cada passo, especificações cada vez mais abstratas, que mantenham o significado dos requisitos informais (Haeberer et alii, 1989).

A decomposição por abstração é o mecanismo fundamental no modelo PW, procurando transformar um problema complexo em um conjunto de subproblemas simples. No entanto, o modelo PW (à semelhança de outras proposições) padece de um problema conceitual fundamental: como podemos garantir que as especificações formais parciais mantenham o mesmo poder descritivo da verbalização ?

Para procurar remover este obstáculo no caso do modelo PW, uma alternativa é gerar resultados observáveis para validar cada especificação parcial. Este procedimento enfrenta outra barreira metodológica: o **problema da validação**, já indicado por Popper: nunca se pode confirmar uma teoria (conjectura), mas apenas rechaçá-la. Assim, mesmo que uma especificação parcial produzisse resultados esperados, não há garantias quanto à sua correção. No máximo, pode-se utilizar os conceitos de Popper de que uma teoria está **corroborada** se tiver sido submetida a testes severos e sobreviver a eles.

As considerações acima não pretendem invalidar o uso do modelo PW ou de modelos semelhantes, mas indicar dificuldades conceituais. O que se procurou fazer foi indicar como teorias de filosofia da linguagem e da ciência questionam se - no caso mais geral - é possível produzir uma especificação formal (ainda que parcial) que corresponda perfeitamente ao significado de requisitos informais. A resposta a esta objeção fundamental é um dos desafios presentes para os estudiosos da abordagem formal.

Estes fatos abrem uma brecha para o uso de métodos "indutivos" ou "heurísticos". Uma consequência possível do "dilema de Popper" aplicado ao processo de desenvolvimento de software é que aspectos como seleção de dados para teste de programas (i.e. tentativas de refutá-los) assumem proporções essenciais, tanto do ponto de vista teórico, como do ponto de vista prático.

5. ...EPPUR SI MUOVE

O processo de desenvolvimento de software tem muito a esperar dos métodos formais, que ainda precisam superar objeções teóricas importantes para que se possa garantir sua viabilidade. No entanto, pode-se extrair algumas consequências de ordem prática.

O uso da decomposição por abstração para o refinamento sucessivo dos requisitos informais, reforça os paradigmas de tipos abstratos de dados e de programação orientada a objetos. O uso de abstrações de dados corresponde exatamente ao processo de derivação de problemas que podem ser combinados. (Esta idéia está também exposta em Veloso (1987)). A idéia dos métodos formais é que cada problema pode ser expresso em termos analíticos; a rigorosa expressão de problemas atômicos corresponde - a nível prático - à implementação de abstrações.

Isto indica que, dado um conjunto de requisitos, o método de abstração é uma abordagem geral o suficiente para ser aplicada a problemas de desenvolvimento de software de natureza distinta (veja-se, por exemplo, Câmara et alii, 1988).

O processo de refinamentos sucessivos da verbalização, que produz a cada passo descrições formais parciais, pode ser visto como equivalente aos métodos de construção de protótipos ("prototyping"). Considerando-se a visão popperiana de um programa (conjectura), um protótipo pode ser visto como uma conjectura parcial testável e refutável. Isto dá uma sustentação analítica ao fato de que o uso de protótipos tem se mostrado um método extremamente eficaz de desenvolvimento de software confiável. Construir um protótipo seria o correspondente informal a produzir uma especificação formal parcial, que pode auxiliar a concepção do software e testar a validade dos problemas parciais que compõem a solução desejada.

6. À GUISA DE CONCLUSÃO

O "status" do software o coloca numa posição sui-generis como ciência: trata-se de uma disciplina inexata, que combina aspectos práticos com possibilidades de formulação rigorosa. É precisamente este aspecto de inexatidão (ou quase-exatidão) que gera uma

divisão entre as escolas que se preocupam com o processo de desenvolvimento de software.

Todas as abordagens sobre o problema concordam que o essencial para se desenvolver programas corretos e confiáveis é que se possa produzir uma especificação verificável (e, se possível, formal) com o mesmo significado dos requisitos informais. O trabalho procura mostrar que existem dúvidas conceituais se isto é possível, no caso mais geral.

Este trabalho é exploratório e apenas apresenta alguns aspectos que uma epistemologia completa do software deverá considerar, incluindo temas de filosofia da linguagem e filosofia da ciência. Espera-se que ele possa auxiliar a todos quantos se precupam com questões metodológicas, pois o autor acredita que estudos nesta direção podem se mostrar frutíferos para apoiar o objetivo comum a todos (formalistas ou não): produzir programas e sistemas confiáveis e eficientes.

AGRADECIMENTOS

Este trabalho foi inspirado em discussões sobre filosofia da ciência e filosofia da linguagem mantidas (há alguns anos atrás) com o Prof. Leônidas Hegenberg. O autor agradece ainda às sugestões dos revisores e assume total responsabilidade pelo seu aproveitamento. Após ter submetido este para o III SBES, o Prof. Flávio Velasco (INPE) indicou-me "paper" recente, onde idéias semelhantes são apresentadas ("Program Verification: The Very Idea", de James Fetzer, Communications of the ACM, Setembro 1988).

BIBLIOGRAFIA

BROOKS,F. "*The Mythical Man-Month*". Reading, Addison-Wesley, 1975.

BROOKS,F. "No Silver Bullet: Essence and Accidents of Software Engeneering." *IEEE Computer*, vol. 20(4): 10-19, April 1987.

CAMARA_NETO,G.; OLIVEIRA, J.R.F.; YAMAGUCHI, F.Y.; VELASCO, F.R.D.; SOUZA, R.C.M. "*Tipos Abstratos de Dados em Computação Gráfica e Processamento de Imagens*". II Simpósio Brasileiro de Engenharia de Software, Canela, RS, outubro de 1988.

GINZBURG, C. "*Mitos, Emblemas, Sinais*". São Paulo, Companhia das Letras, 1989. (trad. do italiano "Miti emblemi spie").

HABERMAS,J. "*Connaissance et Intérêt*". Paris, Gallimard, 1976. (trad. do alemão Erkenntnis und Interesse, 1968).

HAEBERER,A.M.; VELOSO,P.A.S.; BAUM,G. "*Formalizacion del Processo de Dessarolo de Software*". Buenos Aires, EBAI,1989.

HOARE,C.A.R. "An Axiomatic Basis for Computer Programming". *Communications of the ACM*, vol. 12, no.5, pp.576-583, May 1969.

KUHN,T.S. "*The Structure of Scientific Revolutions*". Chicago, Univ. of Chicago Press, 1970.

KUHN,T.S. "Second Thoughts on Paradigms". In: SUPPE,F. (ed.) "*The Structure of Scientific Theories*". Urbana, Universty of Illinois Press, 1974, pp. 459-482.

KUTSCHERA,F. "*Philosophy of Language*". Dordrecht, D. Reidel, 1975.

LISKOV,B.; GUTTAG,J. "*Abstraction and Specification in Program Development*". Cambridge, MIT Press, 1986.

MANNA,Z. "*Mathematical Theory of Computation*". New York, McGraw Hill, 1974.

POPPER,K. "*The Logic of Scientific Discovery*". Basic Books, New York, 1959.

POPPER,K. "*Conjectures and Refutations: The Growth of Scientific Knowledge*". Basic Books, New York, 2nd ed., 1965.

QUINE, W.V.O. "*Word and Object*". Cambridge. MIT Press, 1960.

QUINE, W.V.O. "*From a Logical Point of View*". Cambridge, Harvard Press, 2nd. ed., 1962.

SUPPE,F. (ed.) "*The Structure of Scientific Theories*". Urbana, Universty of Illinois Press, 1974

VELOSO,P. "*Estruturação e Verificação de Programas com Tipos de Dados*". São Paulo, Edgard-Blucher, 1987.

WITTGENSTEIN,L. "*Investigações Filosóficas*". São Paulo, Abril, 1979. (Trad. do alemão Philosophische Unterschungen, 1953).