

# Parte Subjetiva

Rodrigo di Lorenzo Lopes

21 de novembro de 2007



# Sumário

<b>1</b>	<b>Desafios e Frustrações</b>	<b>1</b>
1.1	Descasamento de impedância Objeto-Relacional . . . . .	1
1.2	Refatoração do banco de dados e integração contínua . . . . .	2
1.3	Passos maiores que a perna . . . . .	2
1.4	Melhorar estimativas . . . . .	3
1.5	Descasamento de impedância MARC - Colméia . . . . .	3
1.6	Alguns problemas mais complicados . . . . .	4
1.7	Deslizes . . . . .	4
<b>2</b>	<b>Disciplinas</b>	<b>5</b>
2.1	MAC110 . . . . .	5
2.2	MAT111 e MAT121 . . . . .	5
2.3	MAC211 . . . . .	5
2.4	MAC323 . . . . .	5
2.5	MAC413 . . . . .	6
2.6	MAC342 . . . . .	6
<b>3</b>	<b>Próximos Passos</b>	<b>7</b>
<b>4</b>	<b>Agradecimentos</b>	<b>7</b>

## 1 Desafios e Frustrações

A implementação do Zumbido foi uma serie de desafios, êxitos e frustrações. Até as tarefas mais fáceis provaram ser difíceis depois de uma olhada mais atenta. Erramos sucessivamente na estimativa do esforço. Baseamos nossas estimativas em várias suposições que se mostraram erradas ou incompletas. Isso tudo tornou o caminho mais difícil. Mas, olhando para trás, para todas as disciplinas concluídas, ou para os lados e nossos colegas passando pelas mesmas dificuldades, sabemos que não é hora de parar e que devemos aprender o quanto antes com nossos erros. Esse capítulo descreve esses erros.

### 1.1 Descasamento de impedância Objeto-Relacional

Pensar que um sistema resolve o problema de “descasamento de impedância” do paradigma banco-relacional com a programação orientada a objetos é uma suposição muito forte (e não deveria ser assumida *a priori*). Em outras palavras: não pense que o sistema em que vai trabalhar fornece as classes



adequadas para resolver seus problemas de negócio e para gravar dados num banco relacional, se depende disso para fornecer uma estimativa realística.

Para deixar claro como essa disparidade pode atrapalhar sua vida e como atrapalhou a nossa: primeiro, estimamos que a gravação de um exemplar na base de dados seria uma tarefa bem simples, considerando que o sistema já gravava informações de exemplar (as tabelas já estavam criadas e os objetos de acesso ao banco já estavam implementados). Procuramos, então, identificar uma árvore de objetos que utilizaríamos para guardar as informações provenientes da análise sintática do MARC (como: Exemplar, Autor, Série, Obra) e suas interrelações. Depois, procuramos no sistema por uma hierarquia de classes semelhante, mas não encontramos. Em vez disso, o projeto dispunha de objetos parecidos com as assinaturas das *Stored Procedures*, contendo uma lista de tipos primitivos. Teríamos que resolver o problema de converter a hierarquia da análise sintática em chamadas em uma lista de tipos primitivos.

## **1.2 Refatoração do banco de dados e integração contínua**

Uma experiência difícil para a equipe foi uma série de alterações que o modelo de banco de dados sofreu. A questão foi traumática porque as informações dos objetos do banco de dados são fixas no sistema (por exemplo: o nome das colunas e tabelas). Dessa forma, se o nome de uma coluna é alterado (ou a coluna é removida) um programa que depende das informações dessa coluna (seja para leitura ou escrita) deixará de funcionar.

Como o projeto Zumbido foi pautado na metodologia da Programação eXtrema (XP), os testes unitários logo apontavam que o sistema havia parado de funcionar. Então, gastávamos parte do nosso tempo tentando ajustar o sistema à nova situação do banco de dados. Esse problema causou um certo mal estar. Chegaram a propor a criação de um banco de dados para desenvolvimento, o que foi recusado pela maioria dos membros.

Hoje, percebemos que se não tivéssemos nos esforçado continuamente para a perfeita integração com o banco, teríamos desenvolvido algo muito longe da realidade atual do modelo de dados e tido muito trabalho para adaptar o código.

## **1.3 Passos maiores que a perna**

Durante o curso de XP, tínhamos pouco tempo para implementar o sistema que iríamos terminar vários meses depois. A “correria” fez com que



deixássemos certas decisões nas mãos de indivíduos da equipe, ao invés do consenso do grupo. Isso pesou negativamente contra a arquitetura do sistema, que tivemos que refazer em seguida.

Para ilustrar, tomemos a implementação que fizemos do leitor do MARC. Como o arquivo MARC não era 100% compatível com a biblioteca Marc4j, tivemos que refazer parte da biblioteca. No dia que tentamos implementar um remendo no Marc4j, não conseguimos reconstruí-lo e desconsideramos a idéia de fazer alterações. Fizemos uma cópia da classe com nosso remendo para o diretório de arquivos fonte do nosso projeto. Meses depois, ao analisar o Marc4j, constatei que o motivo pelo qual não conseguimos construir o projeto era a ausência de uma biblioteca que se encontrava numa pasta de nome muito estranho: “lib”.

Mas, por que a programação pareada não funcionou? A programação pareada às vezes não funciona quando indivíduos estão com medo. Medo de dizer: “Pára tudo, porque eu não estou entendendo mais nada” - e suas variações. Isso ilustra bem porque a Coragem é um dos quatro valores de XP. É preciso coragem para admitir que estamos atrasados ou que não estamos compreendendo o processo. Ou para reexplicar ao colega tudo que já está cansado de saber. E para, friamente, se afastar da situação de perigo ou constrangedora sem dar passos maiores que a perna.

## **1.4 Melhorar estimativas**

Quanto tempo se leva para criar um programa que lê um arquivo num formato desconhecido e grava esses dados num banco de dados? Em se tratando de estimativas de software, errar para mais é um dos menores problemas. Precisamos ser pessimistas. Mas é difícil saber quando estamos sendo pessimistas e quando estamos sendo exagerados.

Que tal uma reformulação da pergunta: Quanto tempo se leva para criar um programa que lê um arquivo binário com mil diferentes marcadores de campos num sistema de banco de dados com mais de trinta tabelas onde cada campo possui vários subcampos com regras e valores distintos?

## **1.5 Descasamento de impedância MARC - Colméia**

Não é apenas a complexidade e o tamanho desses dois modelos que dificultam a integração. Existe um problema de “incompatibilidade” entre o modelo do padrão MARC e o do banco de dados do Colméia que vão muito além da disposição física da informação (banco de dados e sistema de arquivos, registros em tabelas e registros num arquivo MARC). Explicando algumas diferenças:



- Os subcampos do MARC são diferentes de linhas ou colunas no banco. Exemplo: no MARC, os subcampos de Língua especificam a função da língua no exemplar. Essa informação não está disponível em nenhum lugar do banco de dados.
- Algumas entidades não possuem nem mesmo um correspondente parcial do MARC para a base (e vice-versa).
- O tamanho máximo dos campos do MARC são arbitrários, especificados em cada registro. Já o tamanho das colunas na base de dados são fixados nas definições das tabelas no banco de dados. As informações de data, local, não são guardados num formato padronizado no MARC.

## 1.6 Alguns problemas mais complicados

À medida que estudamos os cadastros bibliográficos, percebemos que vários tratamentos mais complicados poderiam ser implementados.

Ex1: Tratamento quanto ao reconhecimento de autores. John Smith publicou 20 mil artigos? Será que existiu um autor tão prolífero assim? Alvaro Palhares é homônimo de Álvaro G. B. Palhares?

Ex2: Tratamento de topônimos. Mogi Mirim não é a mesma cidade que Moji Mirim? Formosa é o antigo nome de Ilhabela?

Nesses casos, uma forma de verificar a consistência do cadastro é consultar fontes oficiais, como a Biblioteca do Congresso Americano (para recuperar as informações mais atualizadas do autor e topônimo utilizados em seus registros), ou consultar um catálogo de autoridades e procurar, numa lista, a opção que mais se aproxima do dado procurado (ex: utilizando a *distância de Levenshtein*).

Todos esses problemas foram ignorados dado a brevidade desse projeto e o esforço necessário para implementar tais soluções.

## 1.7 Deslizes

Podemos dizer que o maior problema que a equipe inicial sofreu foi a falta de Comunicação. Esse também foi o valor com o qual mais tivemos problemas no segundo semestre. Por conta disso, e por falta de disponibilidade de tempo, algumas práticas de XP ficaram comprometidas, em especial:

- Processo de planejamento;
- Ciclos curtos de lançamentos;



## 2 Disciplinas

Considero a grade do curso de ciência da computação da Universidade de São Paulo muito completa além de bem flexível. Completa porque permite que o aluno desenvolva sistemas de informação e realize uma análise formal do funcionamento de sistemas. Flexível pela quantidade e variedade de optativas disponíveis e pela ordem pela qual o aluno pode cursar as matérias (praticamente na ordem em que achar melhor, a menos da exigência de alguns poucos pré-requisitos).

### 2.1 MAC110

Cursei a disciplina (Introdução à Computação), ministrada pelo professor Arnaldo Mandel, quando ainda cursava o Bacharelado em Matemática. Foi uma excelente apresentação ao instituto e a primeira vez que programei de forma pareada e enfrentei problemas interessantes como o EP da ilha dos Lobo Coelhos onde fizemos uma simulação de uma ilha povoada de lobos e coelhos. Esse mesmo problema foi apresentado no curso de Laboratório de Matemática Aplicada que mostrou, em mais detalhes, o efeito do Caos.

### 2.2 MAT111 e MAT121

Agradeço aqui o excelente curso introdutório de cálculo, ministrado pela professora Zara Issa Abud que abriu os horizontes para outros cursos do instituto, como os demais cursos de cálculo e os de estatística.

### 2.3 MAC211

Foi uma disciplina extremamente divertida em que escrevi um programa para resolver equações de segundo grau em *Assembly*. Conheci as expressões regulares, a programação orientada a objetos do *Perl*, as mágicas do Gubi, e os joguinhos em modo texto (também do *Perl*).

### 2.4 MAC323

O curso de estrutura de dados ministrado pelo professor Carlos Eduardo Ferreira em JAVA é da mais alta valia para esse projeto. A escolha das estruturas aplicadas são pautadas pelos conceitos ensinados na disciplina. Sua aplicação se estende por várias outras disciplinas (alguns exemplos):

- banco de dados - armazenamento dos índices das tabelas.



- sistemas operacionais - estrutura dos processos, sistema de arquivos.
- programação orientada a objetos - além do uso prático das estruturas de dados, ainda serve como excelente exemplo para a modularização.
- análise de algoritmos - prévia introdução à teoria da complexidade computacional.

## 2.5 MAC413

Também de grande utilização no projeto, o curso nos faz refletir sobre o uso de padrões e anti-padrões, além de introduzir vários conceitos e tecnologias para programação orientada a objetos e apresentar exercícios de “*Design*”.

## 2.6 MAC342

Além de nos introduzir ao projeto, a parte prática do curso, ministrado pelo professor Fabio Kon está sedimentada no projeto desenvolvido. Além disso, o conteúdo teórico, serve de guia para reflexão. É possível relacionar os êxitos e falhas do projeto com nossos acertos e erros em implementar a Programação eXtrema (na seção desafios e frustrações faço menção direta aos conceitos de XP).

Tentamos aplicar ao máximo os conceitos aprendidos na disciplina de programação eXtrema. Em especial, tivemos grande sucesso em aplicar as seguintes práticas:

- Soluções simples;
- Integração contínua;
- Programação pareada;
- Desenvolvimento com testes *a priori*;
- Padrão de codificação;
- Propriedade coletiva do código;



### 3 Próximos Passos

O Zumbido pode reduzir o descasamento de impedância do seu modelo de dados com o MARC. Isso permitiria fornecer mais informações ao agente solicitante (seja ela uma biblioteca, por exemplo) que deseje consultar o acervo da biblioteca do IME e, ao mesmo, permitiria ao sistema “assimilar” mais informações de outras bases acessadas através do MARC.

Por outro lado, o projeto Zumbido também poderia implementar outros protocolos como Z39.50 para comunicação. Uma vez que o protocolo esteja implementado, é possível implementar outros protocolos mais modernos como SRU (*Search/Retrieve via URL*) ou SRW (*Search/Retrieve Web Service*) que utilizam a mesma linguagem de busca (*Common Query Language*) e estão aos poucos substituindo o padrão Z39.50 pré-*Web*.

### 4 Agradecimentos

Agradecimentos a minha esposa, meus filhos, meus pais, meus amigos e meus colegas por perdoar meus atrasos, ausências e faltas, e por sorrirem nos dias cinzas e nublados.