Desenvolvimento de um Microprocessador 8086 na Arquitetura RISC

Dênis Araújo da Silva
Engenharia da Computação
Universidade Federal de Itajubá
silvadenisaraujo@gmail.com

Marcos Aurélio F. de A. Costa
Engenharia da Computação
Universidade Federal de Itajubá
macaufreitas@gmail.com

Maurílio Pereira Coutinho
Orientador
Universidade Federal de Itajubá
maurilio.coutinho@gmail.com

Robson Luiz Moreno
Co-Orientador
Universidade Federal de Itajubá
moreno@unifei.edu.br

Abstract

The Reduced Instruction Set Computers (RISC) are receiving much attention in the last few years. The main concept of this architecture allows a fast execution in nowadays machines. Once we have fast memories, it isn't necessary to avoid accessing it, so short instructions which do a simple task is the better option. This way, this paper introduce the job of converting a CISC machine (iAPX8086) in a RISC one. The whole process consist in analyse and reduce the instruction set, after this some adjustments need to be done, like adapter all the instructions in order to have all of them in the same size.

Introdução

A arquitetura CISC possui um conjunto de instruções extenso, onde cada instrução é capaz de executar uma tarefa bastante complexa. Tal formato requer instruções de tamanhos variados e que necessitam de uma diferente quantidade de ciclos de clock para serem executadas. Esta definição é válida quando se trabalha com memórias de baixa velocidade, onde cada acesso é custoso, e se torna necessário evitar acessos à memória de modo a reduzir o tempo de execução. Porém, atualmente as memórias são de alto desempenho, o que torna desnecessário o intuito de evitar acessá-las.

Por sua vez, a arquitetura RISC possui um conjunto de instruções reduzido, onde cada instrução é capaz de realizar um simples tarefa. Há instruções específicas para acessar a memória, de modo a permitir que a seguinte regra possa ser respeitada: A cada ciclo de clock uma nova instrução deve começar a ser executada. Nesta arquitetura todas as instruções possuem o mesmo tamanho e levam a mesma quantidade de cliclos de clock para serem executadas, com excessão das instruções que acessam a memória. Desta forma, este formato é interessante para o hardware disponível atualmente.

Desenvolvimento

??) apresenta uma implementação de um sistema médico para gerenciamento de imagens digitais denominado PACS (Picture Archiving and Communication System). O PACS, em conjunto com os Sistemas de Informação em Radiologia (RIS) e de Informação Hospitalar (HIS), formam a base para um serviço de Radiologia *Filmless*.

As imagens são obtidas de modalidades de imagens médicas digitais, como Ressonância Magnética Nuclear (RMN) e Tomografia Computadorizada (TC), em formato DICOM (Digital Imaging and Communication in Medicine) 3.0, indexadas, armazenadas e vinculadas ao RIS do Hospital para posterior visualização.

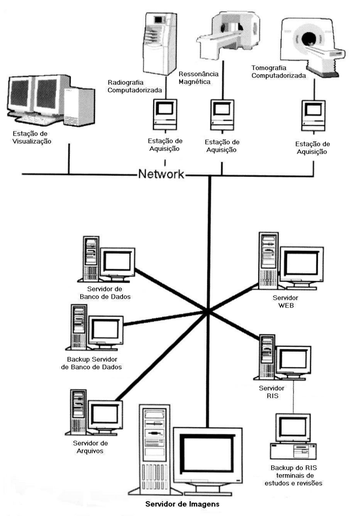

Figura 1: Arquitetura do sistema PACS proposto.

Este sistema foi concebido e desenvolvido dentre um escopo bem-definido. No entanto, há diversos requisitos de negócio em ambientes médicos que necessitam de manipular as informações obtidas pelo sistema apresentado. De forma a realizar integrações com tal sistema de forma escalável e padronizável, é proposta a utilização de SOA.

Segundo **??**), Arquitetura Orientada a Serviços (SOA - Service-Oriented Architecture) é um paradigma para a compreensão e manutenção de processos de negócio que abrangem sistemas gran-des. É baseado em três conceitos principais: serviço, interoperabilidade e baixo acoplamento.

- **Serviço**: pedaço auto-contido de uma funcionalidade de negócio, seja esta funcionalidade simples ou complexa. Serviços desempenham o papel de estruturar sistemas distribuídos baseados em abstrações de regras e processos de negócios.
- **Interoperabilidade**: capacidade de interação entre sistemas diferentes, baseada em infra-estruturas que distribuem e oferecem ser-viços a tais sistemas, utilizando diferentes plataformas e tecnologias.
- **Baixo acoplamento**: minimização de dependências entre sistemas, de tal forma que modificações e falhas não sejam muito impactantes. Com isso, ganha-se em flexibilidade, escalabilidade e tolerância a falhas.

??) apresenta o conceito de serviço como (idealmente) uma funcionalidade de negócio auto-contida que não mantém estado, que aceita uma ou mais requisições e retorna uma ou mais respostas através de uma interface bem definida e padronizada. Servi-ços podem também realizar unidades de trabalho discretas como editar e processar uma transação. Serviços não devem depender do estado de outras funções ou processos. A tecnologia utilizada para prover o serviço, como linguagens de programação, não faz parte desta definição.

Um conceito-chave em SOA é BPM, acrônimo que diz respeito a dois conceitos distintos:

1. **Business Process Management (BPM)**: termo geral que diz respeito a todas as atividades relacionadas ao gerenciamento e melhoria de processos de negócio.
2. **Business Process Modeling (BPM)**: termo que diz respeito à modelagem de processos de negócio e partes destes.

Para implementar BPM, é fundamental a utilização de uma linguagem padronizada para a modelagem e execução de processos de negócio em ferramentas e *engines*. Uma linguagem referência para isto é BPEL (Business Process Execution Language), que está rapidamente tornando-se o padrão para projeto e execução de processos de negócio.

Conceitualmente, uma linguagem como BPEL é uma linguagem XML (Extensible Markup Language) para descrever fluxos de negócio e seqüências, que dizem respeito a serviços. Há elementos de linguagem para a chamada de serviços, respostas de processos, e manipulação de variáveis de processo, estruturas de controle e erros.

O domínio do problema em questão remete à necessidade da obtenção de diagnóstico de mamografia de um paciente, dado que as informações sobre o paciente em questão estão armazenadas em um sistema HIS, e as imagens obtidas em exames estão em um sistema RIS. Neste contexto, propõe-se uma solução baseada em BPM, onde um processo realizará buscas em ambos sistemas HIS e RIS, coletará os resultados desta busca e realizará a junção destes, obtendo um diagnóstico adequado.

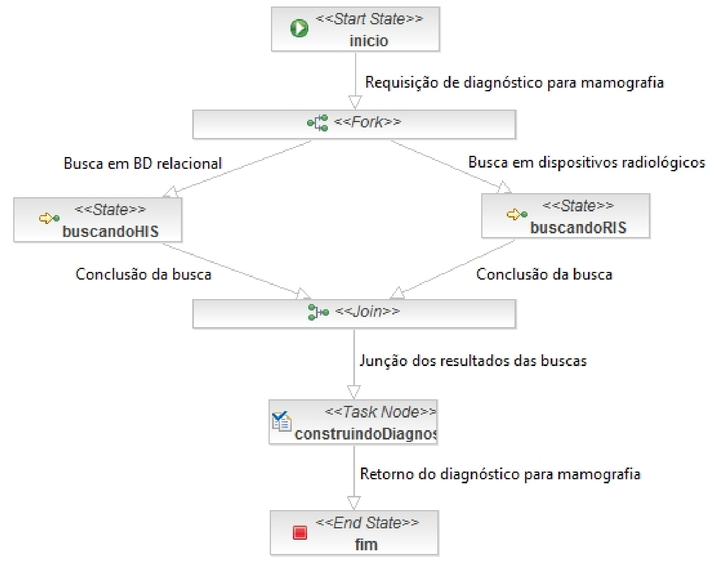

Figura 2: Modelo BPM de diagnóstico de mamografia usando jPDL, uma linguagem para BPM.

Como já foi apresentado, SOA é um conceito amplo e abstrato, já um Web Service é uma das implementações possíveis deste conceito. De acordo com a W3C, um Web Service é uma aplicação de software identificada por uma URI (Uniform Resource Identifier), cujas interfaces públicas e contratos são capazes de serem definidos, descritos e descobertos por artefatos XML; e ainda suporta diretamente interações com outras aplicações utilizando mensagens baseadas em XML, via protocolos baseados na internet. Entre as principais vantagens de um Web Service estão:

- **Interface abstrata**: os Web Services fornecem uma interface abstrata para acesso aos métodos disponibilizados, ocultando detalhes de implementação do usuário do serviço;
- **Semântica acompanha dados**: Ao invés de trafegarem somente os dados, a comunicação entre o servidor e o cliente carrega consigo metadados;
- **Portabilidade**: Por se tratar de um padrão aberto, baseado em XML, garante-se a portabilidade das mensagens mesmo sob diferentes plataformas e linguagens de programação;
- **Segurança**: Opcionalmente, as informações trafegadas podem ser criptografadas;
- **Utilização de recursos**: Os Web Services são sistemas não invasivos, pois não consomem recursos de comunicação enquanto em estado de espera.

Desta forma, propõe-se o uso de Web Services para a implementação de PACS, uma vez que suas tarefas de operação podem ser subdivididas em serviços, que podem ser utilizados por diferentes aplicações de software dentro do mesmo hospital, tornando sua adaptação mais ágil ao modelo de trabalho do hospital.

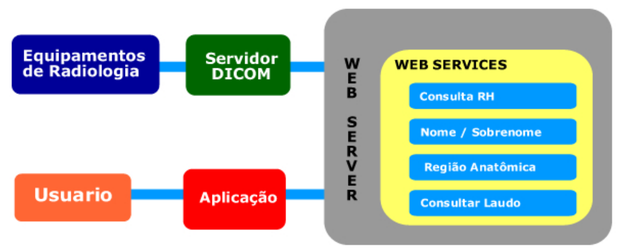

Figura 3: Funcionamento de um Web Service na implementação de um PACS.

Como já foi dito anteriormente, SOA não constitu uma tecnologia mas sim um conceito de como desenvolver sistemas que sejam facilmente interligáveis e como interligar sistemas previamente existentes. Com esse objetivo, foram criadas várias tecnologias para que sistemas desenvolvidos utilizando tecnologias diferentes pudessem comunicar-se entre si. Entre essas tecnologias podemos citar CORBA, RMI e Web Services, porém tais tecnologias usadas de forma isolada ainda apresentavam falhas, algumas por não darem suporte a diversas tecnologias, outras por serem difíceis de implementar ou até mesmo por não possuírem uma especificação que fosse seguida sempre. O maior problema dessas tecnologias era que elas só ligavam um sistema a outro, e sempre que fosse necessária a ligação de vários sistemas elas geravam uma arquitetura muito complexa para se manter, e com o surgimento de novos sistemas a arquitetura tenderia a ficar ainda mais complexa, o que as tornaria inviáveis para o sistema proposto.

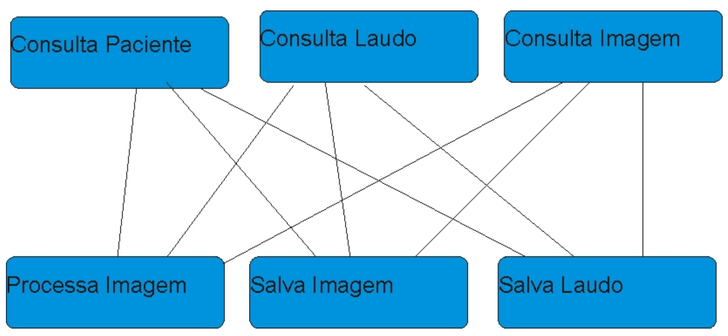

Figura 4: Arquitetura do sistema antes da implantação de um ESB.

Com esse problema, surge a necessidade de um centralização de requisições para os serviços, e nesse contexto entra o Enterprise Service Bus (ESB), que surge como o cerne de uma Arquitetura Orientada a Serviços, pois ele torna-se o responsável pela parte de segurança do sistema, roteamento de requisições e a publicação dos serviços, tornando uma arquitetura mais "limpa", onde todos os serviços são publicados no barramento, o que torna o sistema mais expansível. Desta forma, a cada surgimento de um novo sistema, só será necessário publicá-lo como novos serviços no barramento, não necessitando a alteração dos outros serviços pra que possam integrar-se com o novo sistema.

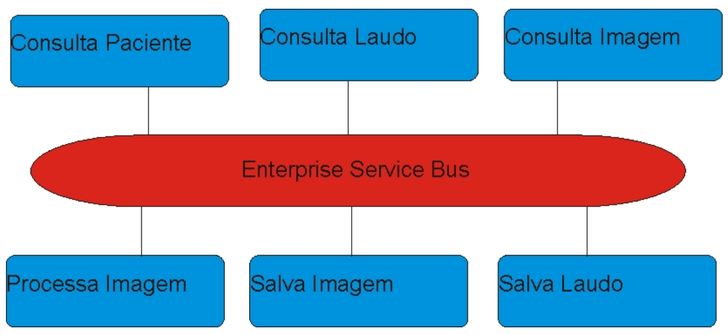

Figura 5: Arquitetura do sistema após a implantação de um ESB.

Conclusão

Conforme descrito neste documento, a adaptação de uma máquina CISC para RISC é um trabalho válido, onde se tem uma redução da quantidade de instruções disponíveis e a diminuição no tempo de execução de cada uma.

O microprocessador resultante deste trabalho é uma máquina eficaz e de simples programação. Estão disponíveis as principais instruções necessárias, com as quais é possível escrever programas que executam as mais diversas tarefas. O desempenho destes programas é de certa forma incrementado e o resultado obtido é considerado satisfatório.

Uma boa comparação a ser feita é a análise do código do sistema operacional DOS da Microsoft. Onde as instruções mais utilizadas são as que constam na máquina desenvolvida neste documento.