52 – Outubro de 2003

Software fechado. Um mal necessário?

Há muito tempo acompanhamos a evolução do software, dito por muitos, como o fator motivador para a evolução do hardware. Por conta disso, nossos equipamentos tornam-se obsoletos em curto espaço de tempo, não por não serem mais aptos a executar os antigos programas de edição de textos, ou planilhas eletrônicas, mas por serem incapazes de suportar com honestidade os últimos lançamentos de editoração eletrônica, manipulação de imagens e também, versões mais pesadas dos mesmos editores de texto.
Em paralelo à esta constante evolução binária, vimos surgir ainda timidamente, o conceito de software livre, ou software cujo código fonte fora desenvolvido por diversas pessoas, sem no entanto, pertencer à nenhuma delas. Nascia aí uma nova fase. Sua penetração, como ocorre com a maioria das iniciativas revolucionárias, se inicia no meio acadêmico. Foi um período longo até que o conceito ganhasse a grande mídia, atingisse o mercado através de versões de demonstração e começasse a ser comentado por respeitados profissionais de tecnologia da informação.
A sensação, nesta fase, era de que muitos acreditavam no seu potencial, falavam orgulhosamente sobre a quebra de paradigma que envolvia a alternativa ao software fechado ou comercial, mas não tinham coragem de entregar a operação de seus processos de negócio à um programa de computador sem dono, apesar de gratuito. Entretanto, iniciativas corajosas e nascidas lá no miolo dos ambientes de TI, foram o laboratório perfeito para demonstrar a competência que os tais softwares abertos tinham. Quando um vírus devastador escrito para a uma plataforma comercial dominante entrava em ação, o poder de proteção do software aberto se revelava por não ter sido afetado.

Em primeira análise, estaríamos diante de uma maravilha da engenharia de software. Contudo, muitas outras variáveis precisam ser avaliadas para se chegar à uma conclusão final. Fazendo uma breve reflexão, é possível perceber as características mais marcantes do software aberto. Sua gratuidade; a possibilidade de conhecer em detalhes sua engenharia e seu código; de interagir abertamente com este código e adaptá-lo aos requisitos do seu processo ou aplicação; a potencialização do conhecimento coletivo com a possibilidade de contar com milhões de programadores ao redor do mundo dispostos a incrementá-lo, corrigí-lo e otimizá-lo; e até a chance de criar seu próprio sistema, com o seu nome.
E foi o que aconteceu. Pequenas fábricas de software agarraram uma versão do produto aberto, como base estrutural, e iniciaram um trabalho de otimização de interface, suporte à dispositivos, serviços de rede, entre outros, fazendo-o assumir uma identidade própria. Tudo parecia com uma iniciativa franca de colaborar com a comunidade *opensource*, trabalhando pelo ideal de provar o potencial de um software aberto, transparente e revolucionariamente concorrente dos softwares fechados. Surgiram eventos grandiosos, órgãos governamentais assumiram a guerra contra os altos preços das licenças dos software comerciais, cópias aos milhares foram distribuídas e ficaram acessíveis aos aventureiros de plantão. Entretanto, sabemos que não se vive de vento e boas intenções. Por menores que sejam, os custos existiam e assim surgiu um modelo de suporte pago, ou seja, não havia

custo para instalar e utilizar o software, mas para receber um suporte especializado e específico para uma versão otimizada, era preciso desembolsar quantia em dinheiro.
Talvez o caráter colaborativo tenha sido o propulsor inicial, mas com o tempo, estas empresas de software começaram a perceber o diferencial que seus produtos abertos e agora otimizados tinham em relação aos demais e principalmente, em relação à estrutura crua da versão de base.

As empresas usuárias, por sua vez, saíram dos tímidos testes em servidores de baixa importância, para serviços e equipamentos *frontend*. Procuravam principalmente o conforto de utilizar um software que não gerasse despesa direta. Apesar dos pontos positivos inquestionáveis, alguns fantasmas rondavam esta iniciativa e provocavam sérios questionamentos: Qual a garantia de que este software será mantido pela equipe que participou de seu desenvolvimento? Qual a garantia de que todas as modificações implementadas foram bem documentadas para orientar uma iniciativa interna de aprimoramento? Qual a garantia de que novos serviços ou drivers de dispositivo poderão suportar minhas necessidades particulares? Qual a garantia de que sempre haverá suporte disponível e capaz de solucionar problemas em um software mantido por muitos programadores de forma distribuída? Qual a garantia de que um dia poderei formar uma equipe capaz de manter a evolução deste padrão de software que adotei? Qual a garantia de que poderá haver responsabilização legal por um defeito de software que provoque danos ao meu negócio? Qual a garantia de que eu estou usando a variante do software que melhor se adeqüa aos meus requisitos de portabilidade, interatividade e performance? Qual a garantia de que haverá continuidade evolutiva do software? Qual a garantia?

Essa é a questão. Num primeiro momento, não há garantia alguma. A informalidade que proporcionou o veloz crescimento do modelo de desenvolvimento participativo e distribuído, agora vira um vilão para os empresários tanto do setor privado quanto público. Ninguém está disposto a assumir risco pelo risco. Tem de haver sempre uma compensação, e será que a economia da licença é suficiente para compensar tanto risco? Muitos dizem que sim e confesso que podem estar certos, pois irá depender muito do modelo e natureza de cada negócio, além de sua composição tecnológica. Mas em linha gerais, podemos ver muito mais dúvidas do que certezas.

Buscando contornar esta situação, inconscientemente ou não, muitas empresas de software que se agarraram ao desenvolvimento aberto desde o início, tanto aperfeiçoaram suas versões que estas já ganharam status de software diferenciado. Têm nome e sobrenome, suporte especializado próprio, manual de instruções, documentação bem trabalhada, equipe de desenvolvimento, treinamentos abertos, e claro, cobram por tudo isso como se uma licença fosse. Em contrapartida, oferecem maior conforto ao cliente e a tão questionada "garantia". Então, não estariam praticando o modelo de software comercial? Talvez não na íntegra, por estar baseado em um núcleo aberto e acessível, mas certamente muitos dos benefícios iniciais passam a ser questionáveis.

É. Realmente a questão não é das mais fáceis. Os softwares fechados também tem muitos problemas. A relação de dependência silenciosamente imposta por uma plataforma obscura e embalada em caixa preta é desconfortável. Os altos custos de aquisição, implementação, configuração e manutenção são, muitas vezes, proibitivos. Sem contar o custo de

propriedade (TCO) indesejável, que ganha cada vez mais participação no orçamento de TI das empresas.
Mas e os pontos positivos? E a forte estrutura que mantém tudo em dinamismo? E a garantia de ter a quem reclamar? E a responsabilidade pela continuidade do software? E os altos índices de interatividade e integração com novos dispositivos e serviços? E o suporte profissional para resolução de problemas?

Eu estou certo de que temos mesmo que viver este momento. Ele é saudável. O fato de existirem alternativas nos faz mais pró-ativo, mais questionador, e nos torna um agente de mudança e de melhoria para ambos. Sim, deve haver uma opção mais adequada. Não uma para todos, mas talvez uma opção para cada grupo de perfil de empresa, ou quem sabe, um modelo híbrido que não coloque todos os ovos em uma mesma cesta. Um modelo de risco distribuído que equilibre a operação da empresa nos dois pilares e a faça usufruir dos benefícios das duas opções.

Esta conclusão não me faz um pessoa contrária ao modelo aberto de desenvolvimento de software, nem tão pouco, um defensor do capitalismo tecnológico que prioriza o lucro à qualquer custo, porém, nem sempre o que é brilhante no conceito o é na prática. Ainda há muito o que evoluir, experimentar, errar e acertar. Enquanto isso, depois do quase empate, o software fechado parece um mal necessário para evitar surpresas desagradáveis que um gerente responsável detestaria vivenciar.

Marcos Sêmola *é Consultor Sênior em Gestão de Segurança da Informação, Professor de Segurança da Informação da FGV – Fundação Getúlio Vargas, MBA em Tecnologia Aplicada, Bacharel em Ciência da Computação e autor do livro Gestão da Segurança da Informação – uma visão executiva, Ed.Campus 2003. Atualmente é Consulting Practice Manager da Schlumberger Worldwide IT Partner.*
marcos@semola.com.br