"Sistema Distribuído" revisitado: a solução torna-se problema

Distribuir significa "dividir e distribuir em porções", implicando uma totalidade anterior capaz de ser dividido [*http://www.thefreedictionary.com/Distribute*d]. Paradoxalmente, o problema epistêmico com sistemas distribuıdas é que eles são concebidos, projetados e implementados em correspondência com a conotação comum de "Distribuído" [11, 8]. Assim, "Em computação distribuída, um programa é dividido em partes que são executados simultaneamente em vários computadores que se comunicam através de uma rede. [...] O principal objetivo de um sistema de computação distribuída é conectar usuários e recursos de forma transparente, aberta e escalável. [...] A computação distribuída implementa uma espécie de simultaneidade. Ela se relaciona tão estreitamente com programação simultânea que, às vezes, não são ensinados como disciplinas distintas [...] Se não for planejado de forma adequada, um sistema distribuído pode diminuir a confiabilidade geral " [*http: // en.wikipedia.org/wiki/Distributed_computing*].

"Distribuição", como "o ato de distribuição ou a condição de ser distribuído; repartido" [http://www.thefreedictionary.com/distribution], está relacionada à gestão de recursos (se algo é abundante, não há necessidade de repartir).

Resumindo, de forma simplista: existem quatro tipos de circunstâncias nas quais,, no âmbito do tratamento de TI nesse caso, a *distribuição*, em seu sentido convencional, pode auxiliar (exemplos entre parênteses):

a) ***No espaço***. Distribuição espacial é o tipo mais antigo e mais conhecido de repartição de recursos (componentes de equipamentos, as fases do processo, redes, terminais de cartão de crédito).

b) ***No tempo***. O time-sharing precede seu nome (métodos de aprendizagem nas escolas, delegar autoridade, ler em uma caminhada, são muito mais antigos do que os sistemas operacionais baseados em UNIX ou barramentos paralelos).

c) ***Na organização***. Qualquer organismo é baseado em ordem distribuída (corpo humano, empresa, estado). Para as empresas virtuais é uma necessidade.

d) ***Na solução de problemas***. "Dividir para conquistar" sempre foi uma importante estratégia para combater a complexidade, principalmente cognitiva (teorias mais reducionistas, a maioria das metodologias – como o algoritmo de Euclides para programação estruturada).

As dificuldades começaram quando "distribuído" e "paralelo" eram percebidos como que sinonimos do sintagma "computação distribuída". Além disso, a confusão epistêmica aumenta quando outros conceitos discutíveis (semanticamente antinômicas) geram antônimos ainda mais confusos: "sequencial" / "simultânea" (em vez de "paralelo"), ou "holístico" / "analítico" (em vez de "reducionista") . Um caso relevante está relacionado com a metáfora básica da ANN (RNA): apesar da mesma estrutura de rede neural distribuída em ambos os hemisférios cerebrais, um par de funções antônomas é indicado como "processamento algorítmico linear" versus "processamento algorítmico holistico" [http: // en .wikipedia.org / wiki / Cerebral_hemispheres].

Aqui, a relutância em aceitar a mudança de paradigma de que o processamento pode ser também não-algorítmico (mesmo no lado esquerdo do cérebro) gera um semipleonasmo (poderia um procedimento passo-a-passo ser não-linear?) e uma quase contradição (poderia um procedimento determinista ser holistico?).

No que refere ao *processamento*, a verdadeira oposição é "sequencial" / "paralelo", onde o paralelismo envolve a distribuição. Por exemplo, "em relação ao processo de aprendizagem como tal, com o prefixo "e-" ou não, o ponto de vista é que o aprendizado humano é melhor descrito pela abordagem de processamento de informações em psicologia cognitiva, de acordo com as idéias aprovadas em: "A maioria das teorias de

processamento de informação modernos são baseadas nas teorias do "aprender-fazendo", que implicam uma melhor aprendizagem devido à combinação de instrução abstrata e exemplos concretos". A aprendizagem deve ser considerada, tanto para seres humanos como para agentes artificiais, como um processo no qual a maior eficácia é alcançado através de uma mistura de modos operacionais simbólicos (como no hemisfério cerebral esquerdo) e sub-simbólicos (como no hemisfério cerebral direito).

Entretanto, a confusão se torna famtasiosa quando "concorrência" e "distribuição" são percebidos como conceitualmente próximos o suficiente para que as aplicações orientadas ao agente - concorrente por excelência - possam ser eficazes se eles são projetados usando mecanismos concebidos (e utilizados com sucesso) para sistemas distribuídos. Como esta é uma reivindicação central desse artigo, deve ser melhor elaborada.

Ao projetar sistemas distribuídos (os exemplos existentes são elucidativos), a distribuição em si é parte principal da solução, e não parte do problema. De fato, na maioria dos casos, o problema era um todo e, considerando-se que tal problema possa ser dividido em subproblemas, a totalidade é desagregada, os subproblemas são resolvidos e, finalmente, as soluções parciais são re-agregadas. No nível da programação, elas executam em paralelo e, ao acessar recursos compartilhados, precisa de *exclusão mútua*. Pelo contrário, as aplicações concebidas de acordo com o paradigma "Computing como Interação" (acima de todos os aplicativos baseados em agentes) implica pelo menos dois interagentes (o agente da interface e seu proprietário), evoluindo em paralelo, de forma autônoma, mas não de forma independente (uma vez que eles interagem como em qualquer processo de comunicação normal: informar, esperar, interromper um ao outor, etc.). Resumidamente, são threads concorrentes em um processo, e sua programação implica multithreading. O problema crucial de engenharia de software é que, enquanto uma API (*Application Programming Interface*) capaz de suportar multithreading abrange todos os requisitos para a exclusão mútua, o oposto é verdadeiro apenas em casos muito simples. Pior ainda, na maioria dos casos, a criação de aplicativos simultâneos com API destinados para sistemas distribuídos resulta em ineficácia grave. Assim, reconhecendo a diferença entre a distribuição e a concorrência é fundamental não apenas no nível epistemológico, mas também no nível de engenharia. Um passo relevante para reduzir as confusões foi a denominação diferenciada adotada em C# para uma instrução existente em Java, sem alterar sua semântica: "Synchronize" é agora "Lock", expressando o que ela realmente faz (reveladora, quase o oposto do que o seu nome anterior significava, uma vez que, para preservar a coerência, o paralelismo deve ser reduzido).

Terminologia

Epistêmico - trata da natureza, etapas e limites do conhecimento humano, especialmente nas relações que se estabelecem entre o sujeito e o objeto do conhecimento.

Sintagma - unidade sintática composta de um ou mais vocábulos que formam orações.

Holístico - relacionado à ideia de que as propriedades de um sistema complexo não podem ser explicadas apenas pela soma das propriedades dos seus componentes. Em um sistema holístico, *O todo é maior do que a simples soma das suas partes.*

Paradigma - representação de um padrão a ser seguido, um exemplo típico ou modelo de algo.

Pleonasmo - redundância numa expressão, propositada ou não, enfatizando-a.

Threads - sequências de etapas em um processo que podem ser executadas paralelamente