2
Auto-sintonia de sistemas de bancos de dados

Este capítulo apresenta os conceitos relacionados à auto-sintonia de SGBDs abordadas nesta dissertação incluindo agentes de software e sintonia de desempenho. Também discute os desafios decorrentes do problema de auto-sintonia e as abordagens propostas. Finalmente, as considerações assumidas ao longo do trabalho e as motivações para a presente pesquisa serão expostas.

Uma introdução aos termos relacionados com desempenho pode ser consultada em [34], trabalho que contém um estado da arte em auto-sintonia de SGBDs relacionais.

2.1
Agentes

Embora o tema de agentes de software[1] venha sendo estudado pela área de Inteligência Artificial desde os anos 80 [48, 70] e pela área de Engenharia de Software nos últimos anos [19, 33], ainda não existe um consenso quanto às características que todo agente deve ter. Por enquanto, autonomia, reatividade, sociabilidade e pró-atividade são as mais aceitas [73]:

- Autonomia: capacidade de agir/reagir para atingir um objetivo sem a intervenção de humanos, tendo algum controle sobre suas atitudes;
- Reatividade: a partir de percepções sobre o que ocorre no ambiente possui a capacidade de responder a mudanças para que seus objetivos sejam atingidos;
- Pró-atividade: agir para manter a possibilidade de atingir seus objetivos, antecipando-se a mudanças no ambiente;
- Sociabilidade: capacidade de interagir com outros participantes do ambiente.

[1] Nesse trabalho serão chamados simplesmente de agentes

Os agentes podem ser classificados, entre outras formas, segundo a sua característica de agência: fraca ou forte [73]. Agentes de agência fraca são aqueles que possuem as características citadas acima. Os de agência forte, também chamados agentes inteligentes, deverão possuir a mais a capacidade de raciocínio (*rationality*), podendo ser atribuídas noções mentais como conhecimento, intenção e compromisso.

No contexto de agência forte outro atributo importante é o aprendizado. O aprendizado é essencial quando o agente tem um conhecimento incompleto do ambiente [48]. A idéia é aproveitar as experiências para melhorar a habilidade do agente atuar no futuro. O agente aprende durante sua interação com o ambiente e pela observação dos resultados do seu próprio processo de tomada de decisões. Existe uma forte relação entre este atributo e a autonomia: um sistema é autônomo na medida em que seu comportamento está determinado por sua própria experiência [48].

Agentes são encontrados com freqüência formando uma organização com outros agentes. Estas organizações são chamadas sistemas multi-agentes (*Multi-Agent Systems: MAS*) [73]. Um ponto crítico nos MAS é a interação entre os agentes. Os propósitos dos agentes podem ser conflitantes e devem, portanto, ser coordenados. A coordenação pode ser feita por um agente coordenador dedicado ou por uma comunicação ponto a ponto, onde os agentes interagem descentralizadamente. A cooperação é a coordenação entre agentes não antagônicos enquanto a negociação é a coordenação entre agentes competitivos ou "egoístas" [24]. Um MAS coerente é um sistema de agentes que se comporta como uma unidade. O problema de como manter a coerência do sistema sem um controle global explícito deve ser resolvido através da coordenação, junto com a definição de "compromissos sociais"[2], de forma a resolver os conflitos entre os interesses locais e os objetivos da sociedade. Existem várias soluções para esse problema, entre elas aquelas baseadas em mecanismos de mercado [24].

O desenho de um MAS abrange não somente o desenho de cada agente individual mas, também, os mecanismos de coordenação que lhes permitem relacionar-se dentro da sociedade para atingir seus objetivos. As características mais importantes destes sistemas são:

- cada agente tem informação incompleta e restrita das suas possibilidades;
- o controle do sistema é distribuído;

[2]compromissos de um agente com outro que permitem restringir o comportamento dos agentes dentro do sistema.

– os dados estão descentralizados;

– a computação é assíncrona.

A abordagem de um sistema multi-agentes se justifica nos casos em que a tarefa pode ser melhor resolvida por uma sociedade de múltiplos agentes que por um agente só. Dado que o uso de agentes pode introduzir uma sobrecarga desnecessária nas aplicações que não precisam desse paradigma, seu uso deve ser bem analisado em cada caso.

2.1.1 Agentes em SGBDs

A questão da construção de SGBDs baseados em agentes foi levantada em [40], onde são propostas as três arquiteturas para agentes em SGBDs apresentadas na figura 2.1.

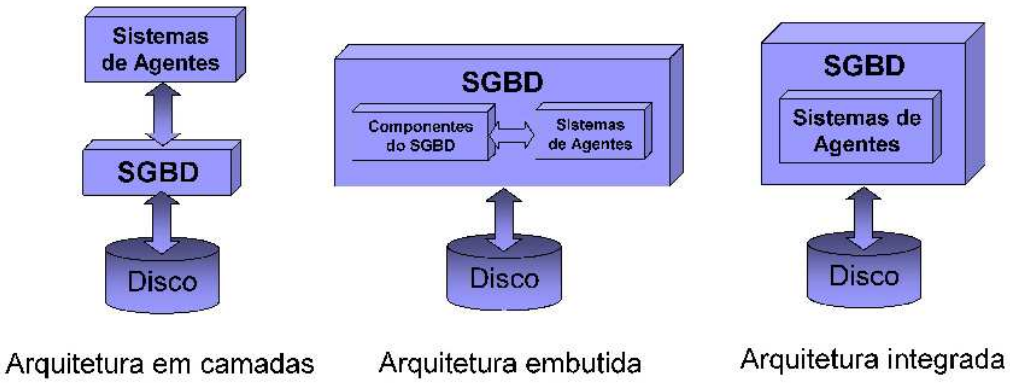

Figura 2.1: Arquiteturas de integração de agentes em SGBDs [66].

Estas são:

1. Arquitetura em camadas: Permite implementar agentes sobre SGBDs existentes sem ter praticamente que modificar o SGBD objetivo. A ação dos agentes é limitada pois devem acessar os componentes do SGBD via as interfaces oferecidas.

2. Arquitetura Embutida: Os agentes são embutidos dentro de SGBDs já construídos para aumentar e estender suas funcionalidades. Esta arquitetura tem limites impostos pelas funcionalidades oferecidas pelo SGBD.

3. Arquitetura Integrada: Os componentes do SGBD são implementados como subsistemas de agentes. A integração dos agentes no sistema

aporta mais flexibilidade e controle que nas outras arquiteturas, ao mesmo tempo que traz mais dificuldades de implementação.

Assim como os SGBDs baseados em agentes, os sistemas de bancos de dados ativos conseguem reagir a mudanças de estado sem intervenção humana [5]. Algumas vantagens podem ser citadas a favor da tecnologia de agentes, como a quase ilimitada persistência das intenções, a possibilidade de retornar o resultado das ações, a capacidade de controlar cenários dinâmicos em tempo real e o fato de que algoritmos complexos de raciocínio são mais facilmente implementados em arquiteturas de agentes. Estas razões sugerem o uso de agentes em lugar de sistemas de bancos de dados ativos para aplicações que exigem adaptabilidade, autonomia e inteligência como a auto-sintonia de SGBDs.

2.2 A arte da sintonia de desempenho

A sintonia de desempenho (*performance tuning*) consiste na realização de ajustes em um sistema (seja um SGBD ou um sistema de computação qualquer) para uma carga de trabalho específica visando obter um melhor desempenho das aplicações ao otimizar a utilização dos recursos computacionais disponíveis.

O *desempenho* pode ser definido como a eficiência com que um sistema atinge seus objetivos. Os sistemas podem fornecer determinados parâmetros de controle que permitam ajustar a alocação dos recursos e, como conseqüência, mudar o desempenho. O desempenho de um sistema pode ser caracterizado através da observação de três classes de métricas [7]:

1. Métricas de Configuração: são características que não mudam ajustando os parâmetros de controle, como a quantidade de memória e a velocidade da CPU.

2. Medidas de Desempenho ou Métricas de Nível de Serviço: As medidas mais usadas para avaliar o desempenho de um SGBD são a vazão e o tempo de resposta. A vazão (*throughput*) é uma medida de produtividade que corresponde à quantidade de trabalho executada pelo sistema durante um dado período de tempo. O tempo de resposta é o tempo de processamento das requisições submetidas ao sistema.

3. Métricas de carga de trabalho: designa todo o processamento de requisições submetidas ao sistema pelo usuário durante um dado

intervalo de tempo. Em [64] é proposto um conjunto de parâmetros para a caracterização da carga de trabalho em bancos de dados. Estes são: a localidade de acesso ou de referência, o comprimento das transações, a proporção de operações de escrita das transações, o nível de multiprogramação e a taxa de chegada das consultas.

A satisfação do usuário com o desempenho do sistema é que define o fim do processo de sintonia.

Chamada às vezes de arte, a sintonia de SGBDs ainda é uma tarefa que tem muito de empírico e requer, com freqüência, as habilidades e imaginação de administradores experimentados [49]. A dificuldade que essa tarefa apresenta tem causas técnicas e subjetivas. Entre as causas técnicas podemos mencionar o grande número de ajustes diferentes que podem resolver um dado problema de desempenho, as fortes inter-relações que existem entre os recursos, assim como entre os recursos e a carga de trabalho, e o escasso conhecimento que existe ainda sobre essas inter-relações [56]. Um exemplo da complexidade que as interferências entre as diferentes cargas de trabalho acrescentam à sintonia é descrito em [66]. É comparada a complexidade do problema do controle do nível de multiprogramação (*MultiProgramming Level - MPL*) para uma carga homogênea e para uma mistura de cargas, conforme mostrado na figura 2.2. Aqui é notável a influência da carga no valor ótimo do parâmetro, cuja determinação se torna mais complexa na mistura de cargas heterogêneas.

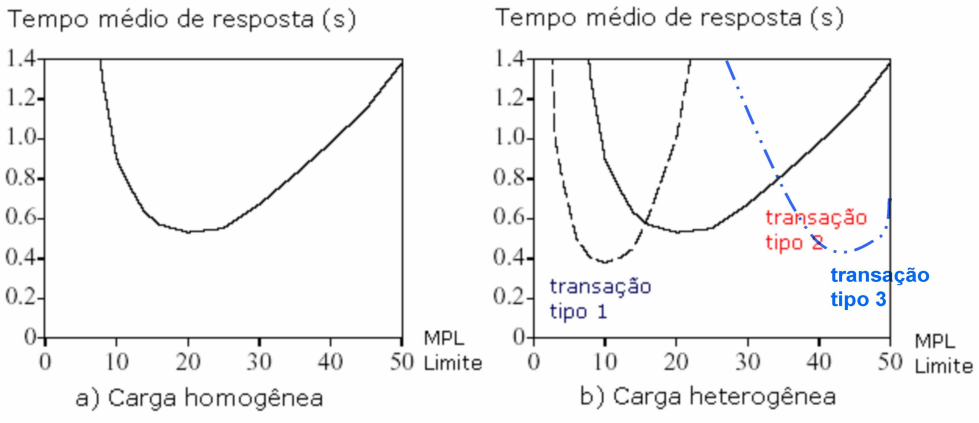

Figura 2.2: Desempenho da transação em função do MPL limite [66].

No primeiro caso, pode-se notar na figura que o valor ótimo para o parâmetro MPL é 20, pois é com esse valor que o tempo médio de resposta do sistema é menor para essa carga de trabalho. No casso da mistura de cargas é mais difícil a determinação do MPL ótimo. A escolha do valor

ótimo de uma carga como valor para o parâmetro MPL otimiza o tempo de resposta para essa carga, mas piora para as outras.

As causas subjetivas da complexidade da tarefa de sintonia se devem fundamentalmente ao fato de que a tarefa habitual do SGBD é oferecer serviços a seres humanos. Isso deriva, por exemplo, na dificuldade para definir prioridades no processamento de consultas dado que muitas vezes os critérios são basicamente empresariais, dificilmente definidos automaticamente.

O fator subjetivo na tarefa de sintonia é um poderoso voto contra a possibilidade da sua total automatização em sistemas de bancos de dados. Dessa forma, a idéia de integrar componentes de auto-sintonia em SGBDs é eliminar o máximo possível as tarefas rotineiras de modo que os DBAs (*DataBase Administrators*) possam se concentrar na verdadeira gerência dos dados.

O processo de sintonia depende fortemente do contexto da aplicação. Este inclui o tipo de aplicação, a relevância da informação gerenciada e a estabilidade do sistema. Aplicações de comércio eletrônico, por exemplo, demandam respeitar limites nos tempos de resposta de forma a evitar a desistência dos usuários.

A relevância da informação é determinante em cenários nos quais decisões de sintonia podem melhorar notavelmente o desempenho do banco colocando em risco, por outro lado, a integridade dos dados. Este é o caso, por exemplo, da opção que permite desabilitar *fsync* (*fsync-disabling*) no SGBD PostgreSQL, conforme descrito em [65]. Quando *fsync* está habilitado, é o *backend* do PostgreSQL que decide quando serão salvos (*flushed*) os dados a disco. Por outro lado, com *fsync* desabilitado é o sistema operacional que decide. As escritas em disco nesse caso serão menos freqüentes aumentando o desempenho do sistema de banco de dados. Mas poderá acontecer que dados já confirmados (*commited*) que estiverem ainda em memória sejam perdidos em caso de uma falha de *hardware*. O valor pré-determinado de *fsync* é habilitado. Em ambientes onde as falhas de *hardware* são improváveis e as informações não são críticas, desabilitar este parâmetro é uma opção a se levar em conta.

A postura de tentar resolver os problemas de desempenho depois destes terem se manifestado, também chamada de postura reativa, é cada vez mais difícil de se manter por causa do aumento das exigências dos sistemas atuais e das cargas a que estes são submetidos [55]. Vários trabalhos têm discutido a necessidade de passar de posturas reativas a pró-ativas, que se antecipem aos problemas de forma a tentar evitar que estes aconteçam.

Em [55] propõe-se uma metodologia chamada de *Total Performance Management - TPM* em que, após concluído o processo de sintonia, se executa uma etapa de detecção e alerta de problemas potenciais. Em [64] depois de concluídas as etapas que são chamadas no trabalho de diagnóstico e terapia (ou resolução de problemas) executa-se uma etapa de previsão de desempenho que estima a evolução da carga e permite determinar quando o sistema deverá ser ajustado novamente. A indústria de sistemas para gerência de sistemas de bancos de dados tem assumido em grande parte esta postura [34]. Por exemplo, a informação coletada pelas ferramentas da suíte de gerência de desempenho da Veritas Software [63] é consolidada e armazenada em um repositório chamado *Performance DataWarehouse*. Análises históricas sobre os dados armazenados nesse repositório permitem antecipar alertas ante problemas potenciais.

A complexidade que apresenta a tarefa de sintonia tem motivado o desenvolvimento de várias ferramentas de ajuda ao DBA na tomada de decisões de sintonia. Algumas delas, baseadas em Sistemas Especialistas (*Expert Systems*), Raciocínio Baseado em Casos (*Case Based Reasoning - CBR*) e Redes Neurais, são destacadas em [28]. O objetivo desses sistemas é apenas sugerir ações: não inclui a execução dessas ações.

2.3 Auto-Sintonia

Os sistemas com auto-sintonia devem ser capazes de escolher os valores ótimos dos parâmetros de forma a satisfazer os requisitos de desempenho previamente estabelecidos pelo administrador ou usuário do sistema. São tarefas de um sistema com auto-sintonia executar tanto processos que tenham por objetivo resolver problemas que prejudiquem ou que podem prejudicar a perspectiva do cliente como processos que visem melhorar o desempenho do sistema.

No caso particular de SGBDs, alguns administradores são contrários à idéia da auto-sintonia, dado que consideram que a maioria dos problemas desses sistemas se devem à implementação ineficiente das aplicações[3]. Certamente, um sistema de auto-sintonia pode não ser a melhor alternativa em todos os casos. Entretanto, podemos identificar algumas situações em que o seu resultado pode ser vantajoso:

[3] *The limits of self-tuning*, http://www.computerworld.com/news/2002/story/0,11280,73890,00.html

1. Dado que o desempenho do sistema depende intimamente da carga de trabalho, assim que esta mudar o sistema pode precisar ser sintonizado novamente. Por este motivo, a automatização do processo de sintonia é essencial se a freqüência com que a carga de trabalho do sistema varia for alta [18].

2. Em instalações cujas dimensões não justifiquem o alto custo de um DBA que gerencie a complexidade e requisitos de desempenho dos sistemas atuais [2, 66].

3. Novos paradigmas como a computação ubíqua (*Ubiquitous Computing*) [71] procuram isolar o usuário das particularidades do sistema. A auto-sintonia de SGBDs embutidos em dispositivos construídos seguindo esses paradigmas é uma necessidade.

Outro argumento contra a auto-sintonia é o fato de que a sua integração com o SGBD implica em uma carga adicional concorrendo pelos recursos do sistema. Uma decisão errada pode piorar o problema de desempenho e erros de programação ou concepção podem deixar um sistema, até então considerado robusto, fora de uso. A qualidade da implementação do sistema de auto-sintonia é crítica para o seu sucesso. Deve-se ter cuidado para não prejudicar o desempenho do SGBD, assim como conseguir diagnosticar os problemas mais significativos de desempenho que o sistema possa sofrer. Os requisitos que um sistema de auto-sintonia deve cumprir serão enumerados na seção 3.1.

Nesse trabalho se considerará que o sistema oferece uma interface mínima com o usuário. Na prática, no entanto, é desejável que o sistema de auto-sintonia ofereça ao administrador a possibilidade de intervir no processo de sintonia, monitorando-o ou decidindo quais ações serão executadas. Por este motivo, deveremos considerar como sistema com auto-sintonia aquele que possua a capacidade de executar ações sem intervenção humana, ainda quando essa possibilidade estiver disponível.

2.3.1 Considerações

1. A sintonia envolve a otimização da utilização dos recursos preexistentes no sistema. Nesse trabalho não serão consideradas soluções que impliquem acrescentar recursos de *hardware*.

2. Nesse trabalho será considerada como parte da atividade de sintonia de desempenho a resolução de problemas de deterioração de desempenho, bem como os processos que geram configurações iniciais ótimas dos parâmetros do SGBD.

3. Apesar de estarem muito relacionados, o conceito de sintonia é diferente da tarefa de planejamento de capacidade, que consiste em determinar a configuração de um sistema ainda inexistente de forma a suportar as cargas de trabalho estimadas.

4. A sintonia fim a fim[4] de um sistema requer a análise de todos os seus componentes, ou seja, tanto o sistema de banco de dados como o sistema operacional e a aplicação. A proposta de [55] é executar a sintonia procurando o gargalo em cada sub-sistema. Em seguida, estudar as interações entre eles e estreitar o espaço de busca voltando a atenção para o sub-sistema suspeito. Para os efeitos dessa dissertação, entretanto, não serão considerados casos que requeiram a sintonia da aplicação ou do sistema operacional, devido à elevada complexidade desse problema. A integração de todas as camadas da aplicação para uma completa gerência dos sistemas também é tratada em [29].

5. Para a definição dos valores esperados das métricas de desempenho pode-se assumir duas alternativas: uma é calcular o melhor valor possível para cada parâmetro com base nos recursos disponíveis. Outra é permitir ao administrador do sistema especificar níveis de serviço ou objetivos (*Service Level Agreements - SLAs*) que serão então mapeados pelo sistema para valores de parâmetros de controle. Nesse trabalho estaremos considerando somente a primeira opção, pois entendemos que a mesma abrange os problemas mais importantes relativos à auto-sintonia de sistemas de banco de dados, e não requer a construção de interfaces com o usuário, uma tarefa que está fora do escopo da dissertação.

2.3.2 Abordagem global à auto-sintonia

Uma abordagem freqüente ao problema da auto-sintonia consiste em usar algoritmos isolados, como os apresentados em [34]. Porém, interações tanto entre os componentes do sistema como entre as cargas de trabalho

[4] Aquela que abrange todos os subsistemas envolvidos na execução da aplicação

podem gerar deterioração do desempenho como conseqüência das próprias ações de auto-sintonia. Por outro lado, um problema de contenção em um recurso pode-se refletir em outro e ser erroneamente interpretado pelo mecanismo de auto-sintonia local que o controla.

Isto leva à idéia de um sistema de sintonia global [8, 11, 42, 66] diferente da idéia da sintonia como a combinação não coordenada de ajustes específicos para problemas ou componentes determinados. As interações entre os componentes podem fazer com que mudanças em um parâmetro afetem outros e devem por isso ser consideradas.

Por exemplo: colocando-se todas as páginas de índices em memória, melhora-se o desempenho de buscas indexadas. Igualmente, aumentando-se os *buffers* mantêm-se por mais tempo os dados na memória principal, o que diminui os acessos e a contenção de disco. Mas estas ações podem deixar pouca memória disponível para ordenação ou *hash* e em função disso, o otimizador de consultas poderia escolher planos de execução menos eficientes para determinadas consultas, comprometendo o desempenho do sistema.

2.4 Trabalhos relacionados

Trabalhos anteriores têm proposto a implementação de sistemas de auto-sintonia global para sistemas de computação em geral e de bancos de dados em particular [34]. Em [8] sugere-se um processo em que um agente único com visão global do sistema executa as funções de análise dos dados coletados pelos sensores e armazenados em um repositório comum. Como resultado dessa análise são acionados mecanismos para efetuar ações de controle.

Em [28] descreve-se a tecnologia dos ATS (*Automated Tuning Systems*), cuja operação coincide em essência com a proposta de [8]. Além dos comentários sobre os desafios técnicos apresentados pela construção desses sistemas, esse trabalho propõe resolver o problema da integração dos ATSs para garantir determinados níveis de desempenho fim-a-fim. Essa integração seria atingida através de um ATS coordenador capaz de balancear os requisitos de cada um dos ATSs do nível inferior dada sua visão global do problema, o que sugere a colaboração entre os ATSs.

Em [7] descreve-se a implementação de um sistema baseado em agentes para a auto-sintonia de sistemas genéricos. Esta generalidade é obtida através de um modelo do sistema controlado que pode ser reconstruído

assim que mudam as características do sistema controlado. Este modelo permite mapear os valores esperados de níveis de serviço em parâmetros de controle.

No domínio de aplicação de bancos de dados as propostas para a auto-sintonia de sistemas podem ser classificadas em dois grupos: aquelas que propõem sistemas auto-sintonizáveis por projeto e aquelas que objetivam a integração de componentes de auto-sintonia em bancos de dados já existentes [34]. Assim, a proposta descrita em [12] baseia-se na complexidade dos sistemas de bancos de dados atuais para propor a construção de um novo tipo de SGBD mais simples, baseado em componentes. Esta simplicidade facilitaria a modelagem e os ajustes. Atualmente está sendo desenvolvido um protótipo seguindo essa abordagem [23] mas ainda não constam na bibliografia sistemas deste tipo já prontos.

Representante da segunda vertente há o sistema apresentado em [39], chamado Quartermaster, para auto-sintonia global de SGBDs no âmbito de um sistema de comércio eletrônico. A arquitetura do Quartermaster segue a linha daquelas descritas em [8, 28]. Constitui-se de um grupo de módulos que são: o Monitor, o Planejador e o Controlador, os quais executam respectivamente as funções de monitoramento, diagnóstico e ação. O Quartermaster possui também módulos de interação com o usuário. Um repositório armazena os objetivos, as medições dos parâmetros de desempenho, regras e características da carga de trabalho, entre outros. Apesar da sua arquitetura lhe permitir executar ações independentes, o Quartermaster ainda deixa boa parte das decisões a cargo do administrador. O módulo do Planejador foi descrito em [9].

A partir da documentação do SGBD e o conhecimento dos especialistas, são criados um modelo de recursos, um modelo de carga de trabalho, regras de diagnóstico e finalmente uma árvore de diagnóstico. O processo de diagnóstico consiste em atravessar a árvore de diagnóstico. Ele produz um conjunto de recursos cujo ajuste é uma possível solução do problema de desempenho. A determinação do ajuste que oferecerá o maior desempenho é feita por algoritmos de auto-sintonia. A diferença de outros trabalhos, o diagnóstico leva em conta vários recursos do sistema a partir de um modelo de recursos que contém informação sobre os recursos e as relações entre eles. A partir desse modelo podem determinar-se quais serão os efeitos de um ajuste sobre os outros recursos e quais são os recursos que podem ter afetado o desempenho de um recurso dado.

Finalmente, em [11] aborda-se o tema da integração dos estudos existentes em auto-sintonia local dentro do contexto da proposta de uma

arquitetura baseada em agentes para a auto-sintonia de índices. O modelo de auto-sintonia global proposto inclui um repositório onde são armazenadas as operações executadas pelo agente de forma a evitar a execução de ações que no passado tivessem afetado o desempenho do sistema. O modelo prevê também a comunicação dos agentes de forma a garantir uma ação combinada em favor do sistema como um todo.

Todas estas propostas utilizam como mecanismo de controle o ciclo de controle fechado ou de realimentação (*feedback control loop*). O ciclo de realimentação permite controlar continuamente a saída do sistema em virtude do desconhecimento da função que relaciona os parâmetros de configuração e carga de trabalho com o desempenho do sistema [45].

As etapas em que deve ser dividida a auto-sintonia, por outro lado, variam ligeiramente de uma proposta para outra. Assim, em [28] considera-se que os passos a executar durante o processo de auto-sintonia devem ser detecção, diagnóstico, ação e avaliação. Por outro lado, em [8] é sugerido que o processo seja formado pelas etapas de definição de expectativas, monitoramento de desempenho, análise e ação. A proposta de [66] inclui somente três etapas, que são observação, predição e reação, diferente de [11], que identifica quatro etapas de um Processo de Auto-Sintonia Genérico:

1. Coleta de informações: Monitoramento da parte do sistema onde está sendo realizado a auto-sintonia.

2. Avaliação do sistema: Avaliação do desempenho baseado nas métricas relacionadas ao nível do sistema e detecção da necessidade de adaptações. A escolha da métrica é uma grande dificuldade dado que é muito dependente do caso em questão e as possíveis alternativas são ora numerosas ora muito complexas.

3. Enumeração de possíveis alterações: Listagem das possíveis alterações a serem realizadas quando é detectado que um componente não está respondendo adequadamente. É a etapa que gera a maior sobrecarga no sistema. São elaboradas várias alternativas calculando seus ganhos/custos. Trata-se de um processo geralmente bastante oneroso ao sistema e nem sempre a alternativa escolhida será a ótima.

4. Realização das alterações: alteração dos componentes do SGBD mediante o mecanismo de auto-sintonia.

No ciclo descrito em [30] como parte da arquitetura de computação autonômica são propostas quatro etapas que podem ser mapeadas dire-

tamente às etapas propostas em [11]. Estas são: monitoramento, análise, planejamento e ação.

O sistema é controlado através dos sensores (que coletam métricas de desempenho) e os efetuadores (que ajustam a alocação dos recursos). A operação do ciclo de controle transcorre como segue: o monitor coleta, filtra e consolida a informação captada pelos sensores. O analisador então executa algoritmos de correlação e modelagem para detectar ou predizer problemas de sintonia. Posteriormente, na etapa de planejamento, elabora-se um plano de ação objetivando atingir os níveis de desempenho esperados, plano este que é executado durante a etapa de ação. As particularidades de cada uma dessas etapas, assim como os desafios que apresentam as suas implementações, serão discutidos na seção 2.5.2.

2.4.1 Definição do problema da configuração ótima

A tarefa de auto-sintonia pode ser caracterizada como a otimização da utilização dos recursos presentes em um sistema para atingir um determinado objetivo com intervenção mínima do ser humano. A configuração de recursos é modificada através de parâmetros de controle. Chamamos de solução de um problema de sintonia de desempenho a escolha de um conjunto desses parâmetros (não necessariamente a combinação que ofereça o melhor desempenho, dado que existem limitações de tempo e consumo de recursos) que aplicados ao sistema gerem os melhores valores de níveis de serviço possíveis para uma carga de trabalho dada, ou valores que se aproximem a estes.

Em trabalhos anteriores [9, 64] a possibilidade de solução deste problema usando técnicas de otimização tem sido descartada. Outras soluções, como a geração de modelos matemáticos[5] são difíceis de aplicar devido à complexidade associada à geração do sistema de equações. Igualmente os modelos analíticos usualmente não são considerados pelas dificuldades da sua elaboração para um sistema tão complexo como é um SGBD.

Considerando o fato de que as métricas de nível de serviço não são independentes [18](podem inclusive ser contraditórias, como é o caso da vazão e o tempo de resposta em algumas situações) e dado o grande número de soluções possíveis (sendo que o espaço de busca corresponde a todos os valores possíveis dos parâmetros de sintonia acessíveis do sistema), o

[5] Um modelo matemático é formado por um conjunto de equações que tentam modelar o comportamento dinâmico do sistema.

problema pode ser formulado como um problema de Otimização Combinatória Multi-Objetivo (*Multi-Objective Combinatorial Optimization Problem - MOCO*[14]). A resolução do problema usando esta abordagem deve gerar um grupo de soluções eficientes ou Pareto-ótimas[6], as quais devem ser posteriormente avaliadas por um "juiz", que nesse caso seria o próprio sistema. Como resultado desta avaliação seria escolhida a solução mais conveniente, considerando-se o valor de cada métrica de nível de serviço na solução ponderado pela relevância da métrica para o sistema em particular. Acreditamos que essa abordagem oferece uma alternativa para a execução do processo de configuração inicial dos parâmetros de sintonia do SGBD.

2.5 Operação do sistema de auto-sintonia

Nessa seção descreveremos a operação do sistema de auto-sintonia.

O sistema executa dois tipos de processos diferentes: estes são configuração e ajuste. A seguir descrevemos em detalhes ambos os tipos de processos.

2.5.1 Configuração

Este processo consiste na determinação dos valores iniciais dos parâmetros de configuração que otimizem o desempenho do sistema e o ajuste dos parâmetros de controle até atingir esses valores. A operação é similar à do *Configuration Advisor* do DB2 [32], que oferece recomendações para a configuração destes sistema. Ele deve ser executado no início da operação do sistema de auto-sintonia e toda vez que o sistema observado tivesse mudado de tal forma que se justifiquem mudanças radicais. É altamente recomendável que um processo desse gênero seja executado off-line ou em horários de pouca atividade, pois os modelos podem incluir imprecisões que prejudiquem o desempenho.

A determinação dos parâmetros em [32] se baseia na modelagem analítica do sistema combinando informação relacionada com características do ambiente que são especificadas pelo usuário, as características do sistema e o conhecimento de especialistas em sintonia do DB2. Os resultados experimentais indicam que as soluções obtidas não melhoram aqueles obti-

[6] Não existe uma outra solução viável que melhore pelo menos um e não piore nenhum dos objetivos.

dos por especialistas em sintonia, mas ultrapassam de forma considerável o desempenho do sistema com os valores de configuração pré-determinados.

Após terminado o processo de configuração e com o sistema já servindo a cargas de trabalho reais, é possível iniciar o processo de auto-sintonia (comentado a seguir) que objetiva ajustar parâmetros de controle do SGBD de forma a melhorar o seu desempenho.

2.5.2 Processo de auto-sintonia genérico

Nessa seção são analisados brevemente os desafios para a implementação das distintas etapas do processo de auto-sintonia. Assumiremos aqui como etapas do processo a observação, a análise, o planejamento e a ação.

Observação

O monitoramento ou observação é a primeira e mais importante etapa do ciclo de controle [66]. Trata-se da captura de uma imagem do estado do sistema em intervalos que dependem da dinâmica de cada parâmetro.

Vários problemas comprometem esta etapa:

- Existe um compromisso entre o fato de que a observação deve ser pouco intrusiva para não interferir nos resultados nem sobrecarregar o sistema e que a freqüência de execução deva ser suficientemente alta para garantir a acurácia da medição e a reação imediata em caso de problemas de sintonia.
- A escolha das métricas que caracterizam o desempenho do sistema é um dos problemas fundamentais a resolver nesta etapa [66]. Devem ser preferivelmente independentes de outras variáveis como, por exemplo, o tempo de observação.
- Normalmente as informações coletadas são armazenadas para posterior análise em um repositório de histórico. Dependendo da freqüência de amostragem, do tipo e da quantidade de métricas observadas, o volume deste repositório pode tornar-se consideravelmente grande.

A coleta de dados pode efetuar-se usando uma das seguintes técnicas:

- Por eventos: A informação se coleta quando acontecem determinados eventos. Usualmente a instrumentação deste tipo de medição consiste em inserir código em lugares específicos do programa controlado.

Quando esses eventos acontecem com freqüência, pode ser introduzida uma grande sobrecarga no processo de medição. Dado que a freqüência dos eventos não pode ser controlada, a sobrecarga da medição se torna imprevisível, o que é uma das maiores desvantagens desse método.

– Por amostragem: Nesse modo, a informação é coletada em intervalos predefinidos especificados no inicio da sessão de monitoramento. A sobrecarga nesse caso depende de dois fatores: número de variáveis observadas e tamanho do intervalo de amostragem. Longos intervalos de amostragem geram baixas sobrecargas, mas a diminuição das amostras provoca por outro lado a redução da acurácia da medição. Comparado com o modo de eventos, fornece uma observação menos detalhada.

Devido ao fato de que os sensores usam os mesmos recursos que estão medindo, dependendo do nível de sobrecarga introduzido, podem ser obtidos dados errôneos. Detalhes de implementação podem provocar também leituras errôneas. Em [46] se reportam problemas de precisão dos dados causados pela inexatidão das ferramentas de linha de comandos do sistema operacional, que levaram os desenvolvedores do sistema a coletar os dados acessando diretamente o núcleo do sistema.

Os sistemas comerciais de gerência de desempenho oferecem soluções interessantes ao problema do monitoramento. Por exemplo, o Veritas Indepth para Oracle [63] coleta as informações diretamente da área de memória compartilhada (SGA). Estas informações são armazenadas em arquivos locais que são carregados em intervalos regulares em um repositório (*Performance Warehouse*) que pode estar situado em outro servidor. A idéia deste repositório é comum entre os trabalhos que propõem sistemas de auto-sintonia [11, 39] e entre as ferramentas de análise de desempenho [34], pois fornece dados históricos que permitem analisar situações passadas.

Análise

Um sistema de auto-sintonia deve procurar continuamente a existência de métricas cujos valores não satisfaçam as expectativas especificadas, indicando problemas de desempenho. A detecção de problemas de desempenho potenciais, que corresponde à abordagem pró-ativa, requer a implementação de algoritmos que, baseados em análises do repositório de dados históricos, consigam extrair informações que indiquem a necessidade da execução de ações corretivas. A escolha da técnica depende da disponibilidade e exatidão

dos dados históricos nos quais se baseia a análise, o horizonte da predição e dos padrões contidos nos dados [26]. Entre os padrões possíveis estão os de tendência, cíclico, sazonal e estacionário. O padrão de tendência reflete uma função que tende a crescer ou a decrescer em função do tempo, enquanto o estacionário exibe uma média constante. Tanto o padrão sazonal como o cíclico apresentam flutuações, a diferença entre um e outro está na periodicidade dessas flutuações.

Idealmente a detecção pró-ativa seria suficiente. No entanto, na prática é possível que alguns problemas sejam detectados somente depois de manifestarem-se.

Fatores decisivos para a efetividade desta etapa são, entre outros:

- a definição do intervalo admissível para cada parâmetro -dependendo dos limites, problemas podem não ser detectados ou, pelo contrário, gerar falsos alarmes-;
- a técnica escolhida para executar a análise -comumente a análise de limite. É muito importante que esta técnica seja capaz de detectar todos os problemas a partir de determinada severidade, ou a confiabilidade do sistema comprometeria a sua utilidade.

É possível que em um determinado SGBD o sistema de auto-sintonia não consiga resolver um problema de desempenho dado por causa das limitações de recursos ou por falta de mecanismos que permitam ajustar a configuração. Entretanto, a detecção destes problemas é crítica para o sucesso do sistema.

Um problema a resolver nessa etapa é saber distinguir se as mudanças na resposta do sistema se devem à mudanças no padrão de carga de trabalho ou à reação do SGBD a ajustes de sintonia. Esse problema se torna ainda mais crítico devido à demora da realimentação, pois dependendo do caso em particular a reação do sistema pode demorar um tempo indeterminado em manifestar-se. Esta é uma das causas pelas quais é preferível executar os ajustes off-line ou em horários de pouca carga de trabalho. No entanto, modelos de carga de trabalho podem ajudar nesse sentido, conforme proposto em [7]. Uma outra opção possível é utilizar SGBD teste. Nesse caso as modificações seriam testadas em réplicas do SGBD operacional submetidas à carga de trabalho real de forma a não afetar o trabalho do sistema.

Após a detecção de um problema de desempenho, providências deverão ser tomadas para a sua resolução. Segundo [18], os métodos baseados na remoção sucessiva de gargalos se demostraram úteis para melhorar o desempenho. Isso sugere que, ainda que sejam detectados vários problemas

de desempenho de uma vez, a resolução deve ser seqüencial pois a reação do sistema pode não eliminar os problemas restantes como também criar novos problemas. Os métodos para detectar e resolver esses problemas serão comentados a seguir.

Planejamento

Nessa etapa o problema é diagnosticado de forma a determinar suas verdadeiras causas. Em seguida são enumeradas as ações que podem corrigir o problema detectado.

O objetivo do algoritmo de diagnóstico é localizar a causa do problema em tempo mínimo. Existem inúmeras publicações e produtos comerciais relacionados com o diagnóstico de problemas de desempenho em sistemas de computação em geral e de SGBDs em particular. Em [34] são estudadas algumas ferramentas de gerência de desempenho de SGBDs. A respeito do diagnóstico, estas ferramentas conseguem sugerir recomendações para resolver problemas de desempenho baseadas em sistemas especialistas. De novo nessa etapa é importante a utilização do repositório, pois nele é mantido um registro de dados temporais que permitem correlacionar eventos gerados por diferentes módulos do sistema para detectar a verdadeira causa do problema. Em qualquer caso, o diagnóstico de desempenho em um SGBD é uma atividade inerentemente global, por causa das interações já discutidas.

A grande quantidade de métricas usualmente disponíveis nos SGBDs leva à sugestão de métodos que permitam estreitar o espaço de busca das variáveis relacionadas com a verdadeira causa do problema de desempenho. Em linha com esta abordagem é proposta em [36] uma hierarquia de produtores e consumidores que representam os componentes do SGBD, como mostra a figura 2.3. Os recursos primários ocupam o nível mais baixo da hierarquia como produtores e os processos ou usuários o mais alto, como consumidores. No nível intermediário encontram-se os subsistemas do SGBD que consomem recursos para servir aos consumidores do nível superior. Em cada ponto da hierarquia são feitas amostragens para checar o estado do sistema.

Esta representação facilita a compreensão das relações causa-efeito, dado que os efeitos do problema de sintonia podem manifestar-se em um componente diferente daquele que o está causando. Desta forma problemas surgidos nas leituras das amostragens podem ser relacionados com as causas.

Enquanto um processo de sintonia nesta etapa retornaria a causa do problema e, em alguns casos, possíveis soluções para resolvê-lo, a auto-

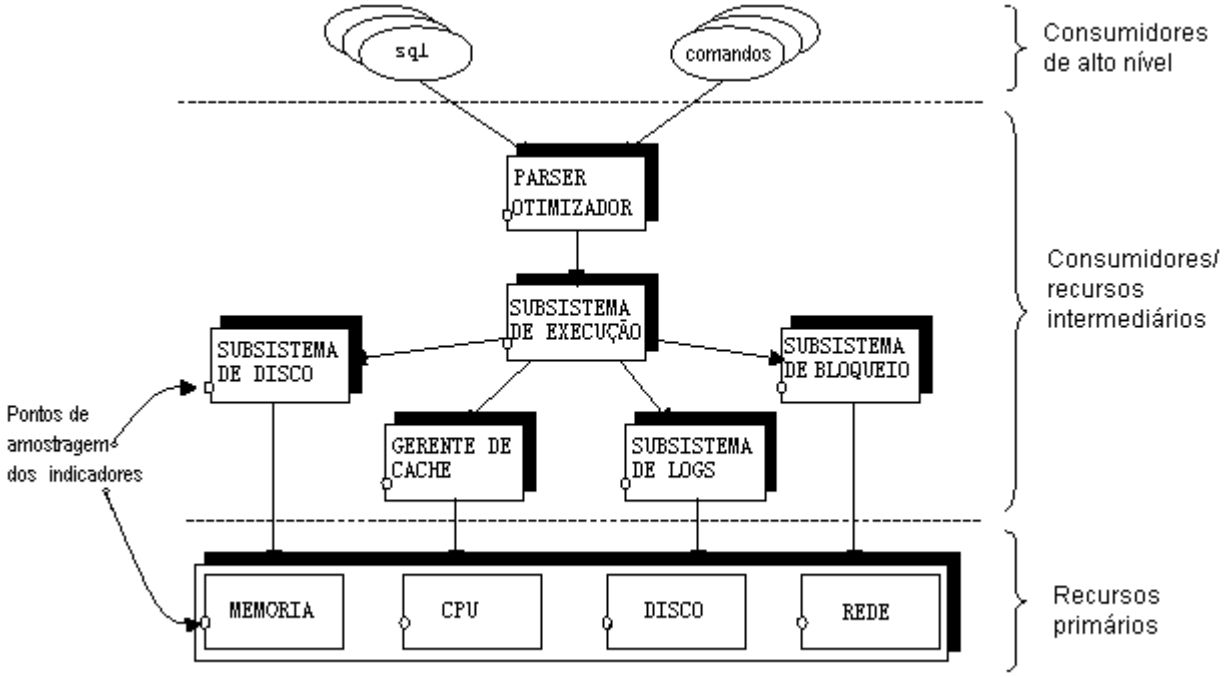

Figura 2.3: Cadeia de Consumo em SGBDs [36]

sintonia precisa determinar os ajustes que podem solucionar o problema de desempenho, assim como os novos valores que deverão ser atribuídos aos parâmetros de sintonia.

Qualquer que seja o método escolhido para a geração das possíveis soluções, após a enumeração de possíveis alterações deve haver uma avaliação para quantificar o desempenho que será obtido reconfigurando o sistema a partir de cada uma das possíveis soluções com a carga de trabalho atual. Testar as possíveis soluções diretamente sobre o sistema pode levar a instabilidade dado que a reação não é totalmente conhecida. Nesse caso os testes somente poderão ser executados em momentos de pouca carga de trabalho ou durante o processo de instalação. Ou então efetuando os ajustes progressivamente em intervalos pequenos como sugere o método de Ajuste Incremental [25]. De outra forma, uma representação ou modelo do sistema será necessária para poder avaliar o impacto das mudanças no sistema. Alguns problemas que podem apresentar as soluções propostas são enumerados em [64]:

- Relação custo-benefício inviável.
- Conflito: ações contraditórias definidas para um mesmo recurso que se anulariam mutuamente.

– Restrição: características do componente, carga ou ambiente em execução limitam a aplicação da ação (exemplo: limitações em tempo de reação ou memória disponível).

Ação

A etapa de ação é a última do ciclo de auto-sintonia. Embora a lista de ajustes já tenha sido definida na etapa anterior, algumas decisões deverão ainda ser tomadas.

Tomemos por exemplo o algoritmo proposto em [66] para o controle de carga orientado a conflitos de bloqueio. Nesse trabalho foi identificada uma métrica para a detecção de problemas de bloqueio, denominada razão de conflitos (*conflict ratio*), que mede a razão entre o número total de bloqueios obtidos e a quantidade deles possuídos por transações ativas. O valor crítico da razão de conflitos em que o sistema começa a sofrer *thrashing* de contenção de dados foi determinado experimentalmente (coincidindo com a determinação teórica desenvolvida em [59]). Este valor está em torno de 1.3, quase independentemente da carga de trabalho. Esta independência da carga, e o fato de ser um valor limite constante, são as características mais importantes dessa métrica pois não é necessário qualquer ajuste. Por outro lado, o caso do algoritmo para a execução das ações de controle é diferente. Ele se divide em dois processos: um processo de controle de admissão de novas transações e um de cancelamento de transações em execução. Várias opções podem ser assumidas na implementação destes métodos. Assim, uma solução ingênua pode ser colocar em uma fila todas as novas transações que chegarem ao sistema durante o tempo em que a variável razão de conflito permanecer em valores maiores de 1.3. Porém, tal decisão pode afetar desnecessariamente o desempenho do sistema no caso de reter transações que não precisem dos objetos bloqueados que estão provocando contenção de dados.

Outras abordagens para este problema requerem conhecimento dos objetos que serão afetados durante o processamento da transação. Da mesma forma, no processo de cancelamento existem vários critérios possíveis a assumir na escolha das transações vítimas. Estas podem ser escolhidas por número de bloqueios requisitados, por tempo de processamento, por custo, entre outras, sendo que algumas decisões podem ser boas para alguns casos e ruins para outros. Embora a métrica do problema seja independente da carga, o processo de sintonia acaba não o sendo por causa da relação da etapa de ação com a carga de trabalho.

Novos problemas de desempenho decorrentes da reação do sistema à execução de ações de auto-sintonia deverão ser detectados pelos mecanismos anteriormente discutidos, fechando-se assim o ciclo de realimentação. O fato de que a concorrência de vários processos de auto-sintonia dificulta esta detecção sugere a implementação de um mecanismo de escalonamento, que execute em nível de sistema um processo de cada vez, sendo que um processo de sintonia pode estar composto por uma ou várias ações. Algumas alternativas podem ser avaliadas, como por exemplo, a introdução de prioridades e o cancelamento de atividades que não sejam mais válidas quando é chegado o momento da execução.

A implementação desses processos é complexa e precisa de uma abstração que ofereça adaptabilidade e reatividade. Os agentes de software possuem características como pró-atividade e aprendizado [7, 33, 73] que podem ser úteis na implementação de sistemas de auto-sintonia. Estudos de agentes de software formando parte de SGBDs para aumentar sua flexibilidade e adaptabilidade foram realizados em [11, 40, 62].

2.5.3 Agentes para auto-sintonia

A construção de sistemas de auto-sintonia usando agentes foi abordada em [7, 11]. Em [11] foi usada a arquitetura proposta em [33], que apresenta um *framework* para o projeto e implementação de agentes de software. A arquitetura está organizada em camadas verticais e permite, com relativa simplicidade, a construção de agentes com capacidades pró-ativa e reativa. É possível mapear as etapas do processo de auto-sintonia proposto em [11] às camadas deste modelo, como mostra a figura 2.4. Por sua vez, essas etapas podem ser mapeadas diretamente com as quatro etapas propostas do processo de auto-sintonia nesse trabalho, que são observação, análise, planejamento e ação.

A arquitetura apresentada tem duas passagens: ocorre um fluxo da camada inferior à superior e outro em sentido contrário. A camada inferior monitora o estado do ambiente e atualiza as crenças, que ativam um procedimento na camada de raciocínio. Este procedimento pode gerar tanto planos a serem executados na camada de Ação como mensagens a serem preparadas e enviadas pela camada de Colaboração. As outras camadas tratam os problemas da tradução de mensagens e mobilidade do agente. Já no agente receptor o processo percorre o sentido contrário. Assim, a camada superior recebe as mensagens que são repassadas à camada de Raciocínio.

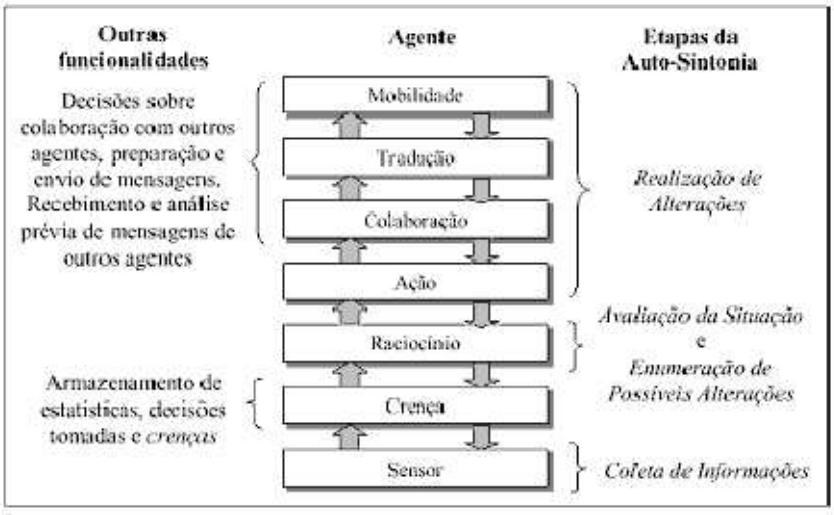

Figura 2.4: Etapas de um processo de auto-sintonia [11]

Esta pode ou não atualizar as crenças e/ou executar modificações na camada Sensor, assim como gerar novos planos. Nesta configuração, a percepção e a ação estarão envolvidas cada uma com uma camada no máximo.

Este trabalho se baseará nessa arquitetura de agentes. Outras arquiteturas podem ser revisadas em [73].

2.6 Motivação da pesquisa

O foco desta dissertação está na sintonia global de SGBDs, dado que existem situações em que ajustes independentes não são suficientes para garantir um bom desempenho. Acreditamos que a implementação de tais sistemas é factível, mesmo que não existam ainda SGBDs comerciais com auto-sintonia global [34]. O objetivo do trabalho é a automatização da tarefa dentro do possível e não a substituição total dos DBAs, o que de qualquer forma deve redundar em uma diminuição dos custos de manutenção dos SGBDs e no aumento da sua disponibilidade.

Seguimos a linha de trabalho do nosso grupo de pesquisa cujo interesse é a utilização de agentes de software para aumentar as funcionalidades dos sistemas de bancos de dados. Este trabalho em particular está aprofundando uma idéia colocada em [11] no contexto de uma proposta para um sistema de auto-sintonia de índices baseado em agentes, em que se menciona a forma em que o componente de auto-sintonia poderia inserir-se em um sistema formado por outros agentes de auto-sintonia.

O objetivo dessa pesquisa é criar um componente de auto-sintonia

global baseado em agentes que possa ser embutido dentro de um SGBD. Entra, portanto, dentro da classificação de sistemas de auto-sintonia por adaptação (veja [34]).

2.7 Conclusões

Nesse capítulo apresentamos alguns conceitos relacionados á nossa pesquisa, como são os agentes de software, assim como as considerações que serão assumidas ao longo do trabalho. Foi mostrado um panorama geral das abordagens atuais para a realização de sintonia e auto-sintonia de SGBDs. Finalizamos com as motivações que originaram o presente trabalho.

No próximo capítulo discutiremos possíveis abordagens para o problema da auto-sintonia de SGBDs usando agentes. Mostraremos os problemas que devem ser resolvidos e, finalmente, proporemos uma arquitetura para a auto-sintonia global de sistemas de bancos de dados usando agentes.