3
Arquitetura

Neste capítulo serão sugeridos um conjunto de requisitos que devem ser satisfeitos pelos sistemas de auto-sintonia e serão discutidas possíveis arquiteturas para sua construção usando agentes.

3.1
Requisitos

A meta fundamental do sistema de auto-sintonia global é garantir que o SGBD controlado mantenha um bom desempenho. Para satisfazer esse objetivo é proposto nesta dissertação que o sistema deva cumprir um grupo de requisitos, a saber:

1. A correção na detecção e diagnóstico de problemas de desempenho crítico[1] que o SGBD possa sofrer. Nesses casos os problemas devem ser corrigidos imediatamente se for possível (se não for possível o administrador do sistema deverá ser notificado).

2. Não tornar o SGBD indisponível. É necessária uma cuidadosa implementação do sistema de forma a evitar que erros de programação ou ações do sistema de auto-sintonia possam deixar o SGBD indisponível.

3. Não introduzir instabilidade no SGBD. Este requisito se aplica a ajustes *on-line* e se refere a evitar oscilações e outros tipos de comportamento errôneo que podem ser causados pelos ajustes.

4. Não alterar os serviços do SGBD. Por exemplo, o sistema de auto-sintonia não pode fazer com que o SGBD comece a devolver dados aproximados para aumentar o desempenho. Qualquer decisão desse tipo cabe apenas ao administrador.

[1] situações em que esteja em risco a disponibilidade do SGBD ou em que o desempenho seja inadmissível sob a ótica dos usuários

5. Garantir que a sobrecarga provocada pelo sistema de auto-sintonia seja inferior ao benefício gerado.

Têm sido identificados também um grupo de requisitos desejáveis:

1. Ser pró-ativo. O sistema deverá ser capaz de detectar e corrigir situações que indiquem futuros problemas de deterioração de desempenho.

2. Requerer do administrador do sistema a mínima intervenção possível.

3. A interação com o administrador deve ser a alto nível, de forma que ele não precisará dominar detalhes da arquitetura interna do SGBD para interagir com o sistema de auto-sintonia.

4. Comportar-se melhor que a simples união de mecanismos de auto-sintonia isolados.

Os trabalhos existentes em auto-sintonia global satisfazem parcialmente esses requisitos. Assim, a tendência à pró-atividade está presente tanto nos sistemas comerciais [63, 72] como nos trabalhos acadêmicos [11, 55, 61, 64]. Por outro lado, os detalhes de implementação do sistema que devem ser seguidos para manter a disponibilidade do SGBD não têm sido muito comentados. O fato de que o administrador é que toma as decisões finais afeta a correção imediata de problemas críticos. A intenção de requerer do administrador do sistema uma intervenção mínima e a nível de objetivos do negócio é também uma tendência atual [34]. Estudos sobre como limitar a sobrecarga gerada pelo sistemas de auto-sintonia ficaram como trabalhos futuros em [8, 9], pelo que concluímos que não é ainda um problema resolvido. Outro problema igualmente não resolvido identificado em [9] (relacionado com o requisito 8) é como garantir que o processo de diagnóstico alcance o desempenho ótimo do SGBD. O requisito de se comportar melhor que a simples união de componentes isolados não é válido em trabalhos anteriores, pois nenhuma das propostas revisadas trata o problema da combinação de componentes de auto-sintonia dentro do SGBD.

Os requisitos propostos sugerem alguns princípios para a construção de sistemas desse gênero, tanto no que se refere ao processo de sintonia como à integração dos agentes com o sistema.

Dado que as interações no sistema fazem com que as reações às ações de sintonia não sejam totalmente previsíveis o sistema de auto-sintonia precisa de um mecanismo que permita controlá-las. A prática tem demonstrado

a efetividade do *ciclo de controle de realimentação* como método de controle do processo de auto-sintonia [18, 28, 66]. Combinado com políticas conservadoras de ajuste, esse método pode garantir a satisfação do requisito número 3.

A etapa de planejamento deverá conferir o custo da operação, de forma que se ele for maior que o ganho a ação não deverá ser executada, satisfazendo o requisito de baixa sobrecarga (requisito 5). Às ações podem ser associados custos, de forma que será possível escolher a de menor custo entre um grupo de ações candidatas. Nesse sentido também pode ser levada em conta a idéia de priorizar as consultas mais importantes e controlar as métricas seguindo uma hierarquia, como proposto em [8].

Para cumprir o requisito de manter a estabilidade do sistema, testes *on-line* devem ser evitados exceto em momentos de pouca carga de trabalho. Pode ser criado, para cada sistema, um modelo que descreva a resposta para uma configuração e uma carga de trabalho determinadas. Este modelo pode ser gerado de várias formas. Uma delas é construí-lo com redes neurais como proposto em [7], outra pode ser extraí-lo em forma de regras [9] a partir dos dados coletados pelos sensores, armazenados no repositório de desempenho como tuplas que relacionam grupos de valores dos parâmetros de sintonia e as respectivas métricas de desempenho resultantes para cada carga de trabalho. Inicialmente os dados podem ser gerados usando uma carga padrão como TPC-C [58] e modificando os controles de sintonia para obter a resposta do sistema para diferentes configurações.

A obtenção das métricas de desempenho deve minimizar o consumo de recursos de sistema para não comprometer a sua correção nem afetar o sistema observado.

Objetivando reagir ante manifestações de problemas de desempenho (requisito 8), deverão coletar-se continuamente amostras dos parâmetros que caracterizam o desempenho do sistema. Estas métricas devem ser cuidadosamente escolhidas como discutido no capítulo anterior. Os dados coletados devem ser checados periodicamente para detectar violações das políticas de sintonia (veja a seção 2.5.2). A detecção de tais violações devem gerar um processo para o diagnóstico do problema e ajuste de parâmetros de controle. Esse processo deverá continuar até que sejam satisfeitas as políticas de sintonia ou então que não seja possível executar ações que resolvam o problema de desempenho. O sistema de auto-sintonia deve incluir, entre as possíveis soluções dos problemas de falta de recursos, a opção de desligar-se para liberar seus recursos em favor do SGBD (requisito 2).

As amostragens devem ser pouco intrusivas para não sobrecarregar o sistema (requisito 5). Os dados coletados serão armazenados em repositórios de desempenho a fim de permitir a detecção de padrões que indiquem problemas que estão por surgir (requisito 1). Nesse repositório são também mantidas outras informações, tais como as políticas ou objetivos de sintonia. A definição destas políticas permite ao administrador definir as suas necessidades em termos de políticas da empresa, afastando-o dos detalhes do sistema (requisito 3).

A construção de um sistema global não se justifica se a simples combinação de mecanismos isolados de auto-sintonia for mais benéfica. Nesse sentido, medidas deverão ser tomadas como, por exemplo, limitar a sobrecarga causada pelos mecanismos de coordenação (requisito 4).

A seguir discutiremos à integração dos agentes com o sistema. Nesse sentido, é preciso desenvolver uma arquitetura que faça com que as ações dos agentes convirjam com os objetivos globais do sistema de auto-sintonia. Um sistema coerente pode ser construído através de um controle global ou da cooperação entre seus componentes(veja a seção 2.1). Essas duas abordagens são discutidas nas seções 3.3 e 3.4.

3.2 Propostas de Arquitetura

Uma abordagem para o desenvolvimento de uma arquitetura para a construção de sistemas de auto-sintonia é a integração de módulos que implementem resultados de estudos sobre auto-sintonia local, sendo que cada módulo pode ser constituído por um ou mais agentes. No desenvolvimento dessa arquitetura devem ser destacados vários detalhes importantes:

1. Nos sistemas comerciais existem usualmente uma grande quantidade de parâmetros de ajuste. Por isso, não é uma boa solução armazenar todas as medidas que descrevem a reação do sistema considerando todos os parâmetros de ajuste para cada uma das cargas de trabalho possíveis.

2. Existem estudos sobre auto-sintonia local para quase todos os componentes de um SGBD. Alguns deles têm sido testados em sistemas reais e inclusive implementados em sistemas comerciais [34]. Esses estudos, em geral, descrevem o processo de auto-sintonia de um determinado aspecto, desde a detecção até o ajuste. Entretanto, eles fazem determinadas considerações com respeito ao estado do SGBD [49] que podem

ser inválidas em determinado momento, ainda que se possa esperar que essas situações não aconteçam na maioria dos casos.

3. A maior parte das propostas não leva em conta os outros componentes do sistema, inclusive aqueles que podem causar um desvio similar nas métricas de sintonia. Isto implica que a integração destas propostas dentro do mesmo sistema pode ter efeitos inesperados. Por exemplo, em [66] se descreve a situação em que uma carga mal balanceada entre os discos provoca demoras em alguns acessos que podem fazer com que aumente a duração de alguns bloqueios. Se a contenção de dados chegar a um nível crítico, o mecanismo proposto nesse trabalho A.2 reagiria detendo a admissão de novas transações no sistema e cancelando algumas das transações em curso, causando a sobrecarga da CPU por causa do processamento causado pelo reinício das transações.

4. Conforme concluído em [16] nem sempre a combinação dos resultados oferecidos pelos algoritmos de análise de um processo de auto-sintonia é a melhor solução no nível global.

5. A complexidade de alguns processos de diagnóstico e ajuste justifica sua execução separada do trabalho do sistema, em um processo concorrente. Este é o caso da proposta em [50], na qual um processo de coleta de consultas SQL submetidas deve ser executado em paralelo a um processo de escalonamento de ações de criação e destruição de índices e a outro de análise. Enquanto o processo de coleta é executado por um agente dentro do sistema, os restantes executam fora do sistema a fim de satisfazer o requisito de baixa sobrecarga (requisito 5).

6. Os mecanismos que integram a etapa de ação de um componente de auto-sintonia local podem ser úteis aos outros componentes do sistema. É o caso do algoritmo de controle de admissão que forma parte do mecanismo de controle de contenção de dados. Ele pode ser usado por outro mecanismo que precise do controle do número de transações concorrentes no sistema.

7. Conforme discutido na seção 2.5.2, o reconhecimento da causa da deterioração de desempenho é facilitada se os processos de sintonia são executados um de cada vez, separados por intervalos que dependem do tempo de reação do sistema ao ajuste em particular, e do intervalo de monitoramento.

Dessas constatações podemos concluir que pode resultar em benefício a implementação de algumas das propostas de auto-sintonia local dentro de um sistema de auto-sintonia global. Entretanto, a introdução dessas propostas requer uma arquitetura que garanta o caráter social que devem ter o diagnóstico e o escalonamento das ações a executar (veja a seção 2.5.2). Essa é a motivação da nossa proposta de arquitetura para auto-sintonia global.

Essa arquitetura consiste basicamente em um grupo de agentes que controlam os diversos módulos do SGBD e em uma base de conhecimento que armazena toda a informação relativa ao estado do SGBD e as ações executadas pelos agentes, assim como as interações entre os componentes do SGBD (regras, conflitos) e os objetivos dos usuários. Os acessos á base de conhecimento podem ser executados através de um Agente de Histórico. Outros agentes podem ser incorporados à infra-estrutura do sistema, como será discutido nas seções seguintes. A combinação do trabalho dos agentes permite não só coordenar o trabalho como também aproveitar os recursos oferecidos por cada um deles, como seria o caso do módulo de coleta de consultas proposto em [50] que seria útil para que o sistema tivesse conhecimento da carga de trabalho submetida.

O relacionamento entre os agentes na arquitetura proposta dependerá do grau de distribuição do controle do sistema. Desta forma podem ser construídas versões decorrentes dessa magnitude que vão desde a abordagem totalmente centralizada até a totalmente distribuída.

Em cada caso é importante para o sucesso de um sistema que siga quaisquer destas abordagens, que os agentes que o compõem cumpram determinados compromissos sociais (aqueles que estabelecem os agentes entre si dentro da sociedade). Para os efeitos dessa proposta em particular, consideramos que os agentes que integram o sistema devem cumprir os seguintes compromissos:

- Os agentes são honestos. Não cabem no sistema atitudes como a de ocultar informação, informar dados errôneos propositadamente ou aceitar tarefas que não serão executadas.
- Os agentes formam parte de uma sociedade e não podem executar de forma independente tarefas inerentemente sociais.

3.3 Abordagem centralizada

A abordagem centralizada proposta é bem semelhante à forma como é executado o processo de sintonia no mundo real. Ou seja, o administrador recebe informações de problemas de desempenho no SGBD e efetua uma análise para resolvê-los. Essa análise baseia-se nas métricas coletadas, na documentação do sistema, na experiência do administrador e, possivelmente, em alguma ferramenta oferecida pelo sistema ou por terceiros. Com essas informações o administrador determina as causas prováveis do problema e começa a executar ajustes, enquanto observa a reação do sistema para desfazê-los no caso de mostrarem-se negativos para o desempenho do sistema. A abordagem centralizada apresentada na figura 3.1 permite executar esta metodologia substituindo o administrador por um agente de software.

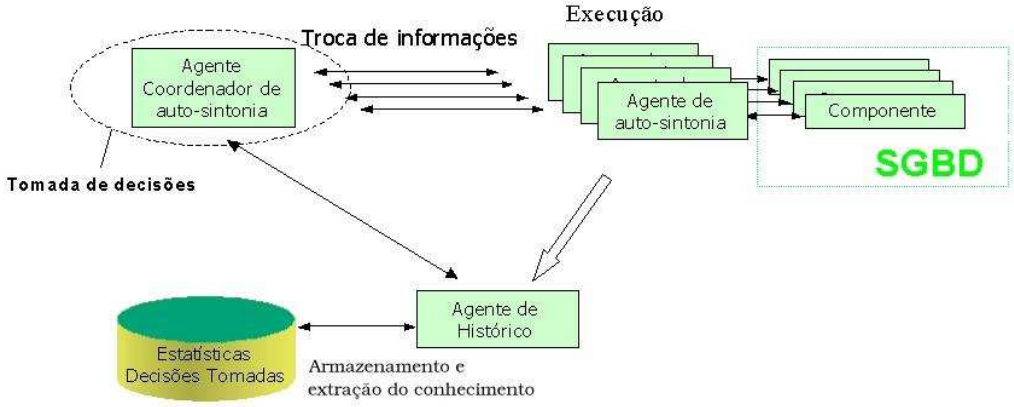

Figura 3.1: Abordagem centralizada para auto-sintonia global

Nessa arquitetura o processo consiste em um agente que comanda a execução de ações de sintonia quando recebe alguma notificação dos sensores de que alguma métrica não está respeitando os valores esperados. Nessa abordagem os agentes que estão relacionados diretamente com os recursos são meramente sensores e efetuadores, comprometendo seriamente sua autonomia a tal ponto que estes podem ser substituídos por objetos. O agente central é o único que domina a visão global do sistema e portanto é quem pode determinar qual é a verdadeira causa do problema de desempenho, dado que um sintoma pode ter diferentes causas. Isto implica que o único agente do sistema com poder para comandar a execução de ações é o agente coordenador, coincidindo com a proposta de [8]. Para efetuar as análises, ele precisará dominar o significado de cada variável, assim como particu-

laridades dos mecanismos disponíveis no sistema que permitam resolver o problema.

Problemas presentes nessa arquitetura, como a dificuldade de expansão, a falta de autonomia dos agentes e a dependência do agente coordenador nos levam a uma arquitetura onde o controle e os dados estejam mais distribuídos entre todos os agentes componentes do sistema, como veremos a seguir.

3.4 Abordagem distribuída

Na abordagem distribuída o controle (e o conhecimento) se encontram distribuídos pelo sistema. Na figura 3.1 apresenta-se uma proposta que segue a abordagem distribuída para auto-sintonia global de SGBDs.

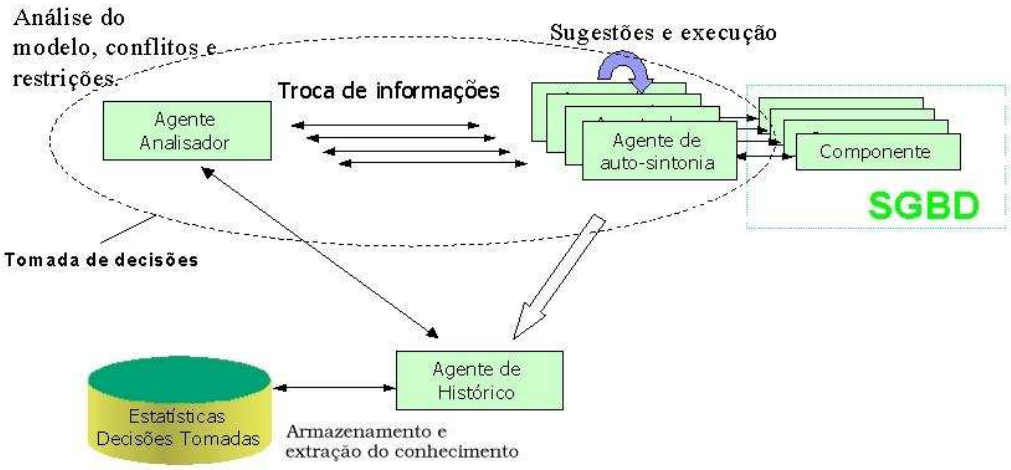

Figura 3.2: Arquitetura distribuída para auto-sintonia global

Essa arquitetura é formada por um agente analisador, agentes componentes e o agente de histórico que controla a base de conhecimento. Os agentes componentes contêm mecanismos que permitem variar os parâmetros de sintonia e os oferecem como serviços aos outros agentes. Nessa variante as funções de análise se distribuem pelo sistema, de modo que os agentes componentes tomam parte também na geração das soluções. A resolução distribuída de um problema de desempenho é descrita a seguir:

1. Durante a etapa de monitoramento cada sensor coleta as métricas de desempenho.

2. Os dados coletados são ora enviados ao agente de histórico, que oferece um serviço de registro com esse propósito, ora mantidos pelo próprio agente, sendo que no último caso algum mecanismo deverá existir que permita aos outros agentes ter acesso a esses dados. O fato de que existe uma grande quantidade dessas métricas indica que não é possível trafegá-las livremente dentro do sistema.

3. Os dados coletados são comparados com as expectativas. Se for detectado um problema de desempenho, um processo de sintonia deve ser iniciado.

4. A etapa de planejamento objetiva a determinação da causa do problema e a elaboração de um plano de ações (ou intenção) que deve resolvê-lo. As decisões do agente analisador tem a maior prioridade, pois ele tem acesso às informações contidas na base de conhecimento relativas a restrições e conflitos. Os agentes componentes relacionados com o problema podem elaborar propostas que podem ser eleitas desde que não violem essas restrições.

5. As ações enumeradas no plano resultado são escalonadas para ser executadas por agentes com as devidas capacidades. As solicitações de execução de ações são feitas em forma de requisições por serviços. A execução de cada tarefa está reservada ao agente que sugeriu a solução selecionada para execução, mas este pode delegá-la a qualquer agente que ofereça o serviço. Um dos protocolos mais usados para alocação de tarefas é o *Contract Net*[73].

Esta metodologia pode ser melhor compreendida através de um exemplo. Durante a etapa de monitoramento um sensor coleta dados de vazão média no sistema, que são enviados pelo agente para o repositório. No caso em que o valor médio foi menor que o valor esperado para esta métrica, uma mensagem de alarme é enviada ao sistema. O agente analisador envia uma mensagem anunciando um problema de "vazão baixa", a partir da qual os agentes relacionados com o problema reconhecem que irá começar uma sessão de planejamento.

Cada agente convocado pela mensagem pode oferecer a sua solução para o problema. Um agente implementado seguindo a proposta de [25] irá diminuir o MPL assim que a vazão descer, e começará aumentá-lo quando a queda da vazão parar. O agente encarregado do controle de páginas em *buffer* irá sugerir a troca de estratégia de substituição se for detectado uma varredura seqüencial muito longa. Um agente de índices pode sugerir a

destruição de um índice que está causando um elevado nível de bloqueios. E o agente analisador, segundo o modelo, irá sugerir a diminuição do MPL ou detenção temporal da aceitação de novas transações. Nesse caso somente as propostas de controle de MPL entram em conflito. Considerando que não tenham sido encontradas situações similares no histórico que provocassem deterioração do desempenho, todas as propostas serão executadas exceto aquela de controle de MPL que entra em conflito com a proposta do agente analisador. Estas ações serão escalonadas para execução seqüencial.

Quanto a qual será o grupo de agentes que receberá a mensagem de convocação, a mensagem pode ser um *broadcast* a todos os agentes do sistema multi-agentes. Nesse caso a vantagem é que não se precisa saber de antemão a lista de agentes relacionados com cada problema, enquanto a desvantagem é a sobrecarga desnecessária que representa no que diz respeito às comunicações. Uma possibilidade mais atraente pode ser aproveitar o processo de inscrição do agente na entrada no sistema multi-agentes, para registrar também os problemas que está capacitado para resolver.

Os resultados obtidos por cada agente envolvido no problema podem ser diferentes. O intercâmbio de resultados entre os agentes oferece todas as soluções disponíveis que o sistema de agentes pode acionar para resolver um determinado problema. Cada agente formula o resultado de acordo com o seu próprio algoritmo e a sua visão do sistema. Embora essa solução implique em um esforço inútil para os agentes cujas soluções não serão escolhidas, o dinamismo que oferece introduz vantagens tais como os agentes poderem entrar no sistema no meio da execução sem precisar de reconfiguração e experimentarem alternativas não previamente consideradas.

Os custos de sobrecarga que implica esta alternativa podem ser reduzidos concentrando a negociação em um repositório comum como o oferecido nas arquiteturas de *blackboards*. Um *blackboard* é um repositório compartilhado que permite intercâmbios indiretos de informação entre fontes independentes. Durante a resolução de um problema os especialistas trabalham cooperativamente, acrescentando suas contribuições ao *blackboard* toda vez que tem informação suficiente para fazê-lo, em um processo que continua até o problema ser resolvido [24]. Esta opção permite também que o agente não precise conhecer a quem notificar as decisões. Por outro lado, o *blackboard* precisa de um elemento de coordenação que introduz novamente problemas da arquitetura centralizada. No entanto, a vantagem é que ao pertencer à infra-estrutura do sistema, este elemento não precisa ser alterado toda vez que se precisa acrescentar um novo algoritmo, ao contrário do que acontece na arquitetura centralizada. Outra vantagem está em que esses elementos

são padronizados e disponíveis, minimizando o esforço de programação e a possibilidade de erros.

Outra possibilidade é utilizar o método de Criação-Destruição (*Create-Destruction*). Segundo esse método, um grupo de agentes, cujas funções são construtores ou destrutores, trabalham em um problema. Os construtores incorporam novas soluções ao conjunto, enquanto os destrutores eliminam soluções. A eliminação de soluções que não deveriam ter sido criadas deve melhorar a qualidade da solução. Nesse caso o analisador seria basicamente um destrutor, pois baseado nas conclusões extraídas da base de conhecimento pode determinar se alguma solução proposta pode danificar o desempenho.

A vantagem dessa proposta distribuída está no fato, já mencionado, de que existem situações em que os mecanismos de auto-sintonia não oferecem a melhor solução mas a experiência da sua utilização em sistemas comerciais e os resultados de estudos podem mostrar sua eficácia em boa parte das situações. Entretanto, o processo de diagnóstico e planejamento precisa que as decisões do agente analisador sejam sempre as mais prioritárias e que os compromissos sociais colocados sejam cumpridos.

Uma abordagem que pode ser considerada intermediária entre a centralizada e a distribuída consiste em fazer com que o agente analisador avalie as soluções extraídas da base de conhecimento e as propostas oferecidas pelos agentes. Esta variante não é totalmente distribuída e novamente é sensível a falhas no agente analisador, mas permite um grau mais forte de agência que a versão centralizada sem afetar a coerência do sistema além de simplificar o processo de negociação.

A abordagem distribuída supõe algumas vantagens sobre a centralizada:

1. Maior capacidade de resposta dado que existe mais de um agente atendendo as notificações e é possível a comunicação entre grupos de agentes em separado.

2. Permite implementar métodos de sintonia distintos em componentes separados, de forma que estes podem ser acrescentados e modificados sem alterar o sistema. No caso da arquitetura centralizada o código do componente mais sensível do sistema (o agente coordenador) deveria ser modificado.

3. Permite o reuso dos resultados de estudos existentes sobre auto-sintonia local. Este fato, além de aliviar o trabalho do agente central,

representa uma notável diminuição do esforço de programação do sistema e facilita a sua escalabilidade.

4. Facilita a introdução progressiva de novas funções e o trabalho em conjunto de distintos programadores.

5. Diminui a probabilidade de falhas do sistema.

6. Aumenta a facilidade na introdução de novos agentes, cujas propostas são avaliadas assim que ocorre algum problema com os recursos relacionados com o agente.

Por outro lado, podem ser citadas algumas desvantagens da abordagem distribuída:

1. Maior complexidade.

2. Os agentes precisam ter conhecimento dos outros agentes.

3. O conhecimento no sistema está disperso. As ações dos agentes devem ser coordenadas pois a solução depende sempre do contexto global que envolve todas as variáveis observadas.

Requisito	Resolvido	Comentários
Correção na detecção e diagnóstico de problemas de desempenho crítico	Não	Não existe até o momento uma forma de comprovar que o sistema escolhe a melhor solução.
Não tornar o SGBD indisponível.	Não	Esse aspecto depende da implementação
Não introduzir instabilidade no SGBD.	Sim	É possível introduzir nas crenças do agente os limites para as modificações do sistema. Da mesma forma a detecção de ciclos de modificações deve ser codificada como parte do raciocínio do agente coordenador na abordagem centralizada, ou do sistema na distribuída
Não alterar os serviços do SGBD.	Não	Esse aspecto não pode ser resolvido pela infra-estrutura do sistema. Em arquiteturas de integração como a embutida ou a integrada o acesso às funções do SGBD somente pode ser limitado por este. Fica pelo implementador levar em conta esse tipo de limitações
Garantir que a sobrecarga provocada pelo sistema de auto-sintonia seja inferior ao benefício gerado	Sim	No caso de serem detectadas pioras no desempenho decorrentes da execução de ações de auto-sintonia, estas devem ser desfeitas pelo próprio sistema. Ações cujo custo ultrapasse um certo limite deverão ser adiadas o impedidas de serem executadas. Idealmente, um cálculo de custo benefício poderia resolver este problema, se bem este nem sempre pode ser determinado previamente.
Ser pró-ativo	Sim	A introdução de agentes facilita a execução de algoritmos de predição que detectem problemas potenciais e iniciem processos de resolução destes. O aprendizado é uma característica importante nessa tarefa.
Requerer do administrador do sistema a mínima intervenção possível	Sim	Às soluções que impliquem interação com o administrador deve ser alocado um custo alto
A interação com o administrador deve ser a alto nível	Sim	Na base de conhecimento podem ser armazenados mapeamentos entre os níveis
Comportar-se melhor que a simples união de mecanismos de auto-sintonia isolados	Sim	A arquitetura pretende corrigir os problemas decorrentes das interações entre os mecanismos de auto-sintonia local

Tabela 3.1: Satisfação dos requisitos propostos para um sistema de auto-sintonia global de SGBDs na arquitetura.

A forma em que a arquitetura proposta satisfaz os requisitos de auto-sintonia colocados em 3.1 é mostrada na tabela 3.1

3.5 Agentes na arquitetura

Entre as vantagens que apresenta a introdução de agentes no projeto e implementação dessas arquitetura está a facilidade de integração de novos mecanismos de auto-sintonia. Estes podem ser acrescentados progressiva e independentemente um do outro, sendo a sua inter-relação gerenciada pelo sistema. Esta é uma grande diferença entre esse trabalho e propostas anteriores, em que os sistemas de auto-sintonia têm sido sempre tratados centralizadamente dentro do contexto do SGBD.

A arquitetura oferece variantes para a integração do sistema de auto-sintonia com o SGBD. No caso, dada a indisponibilidade de SGBDs construídos seguindo a arquitetura integrada (veja a seção 2.1.1), as opções se limitam às arquiteturas embutidas e em camadas. A arquitetura embutida possibilita uma implementação mais eficiente, tanto pela possibilidade de aproveitar componentes do SGBD como por poder acessar diretamente as funções e variáveis necessárias para efetuar o monitoramento e a execução de ações. De outro lado, complica a implementação pois é necessário entender o funcionamento interno do banco de dados e, possivelmente, alterar algumas das suas estruturas. A arquitetura em camadas, por outro lado, utiliza as interfaces que dispõe o sistema para uso público, em geral bem documentadas e mais fáceis de utilizar. As desvantagens desta se baseiam fundamentalmente em critérios de eficiência.

Ambas as arquiteturas permitem integrar mecanismos de auto-sintonia conhecidos, sendo que na abordagem centralizada seria preciso modificar o código do agente coordenador.

3.6 Conclusões

Nesse capítulo foram apresentados os requisitos que deve satisfazer um sistema para auto-sintonia de SGBDs. Duas abordagens de arquitetura (centralizada e distribuída) foram propostas, e comentadas as vantagens e desvantagens de cada uma, assim como detalhes da forma de operação.

Em seguida iremos descrever os detalhes da implementação do sistema. O protótipo implementado foi baseado na arquitetura centralizada pois

a implementação da arquitetura distribuída de forma completa não seria possível devido aos limites de tempo de desenvolvimento dessa dissertação.