Adaptabilidade e Confiabilidade em Ambientes Dinâmicos: a Abordagem Quasi-Síncrona

Carlos R. Almeida, *Membe, IEEE*

***Resumo*— A proliferação de novas infra-estruturas de comunicação e computação cria o potencial, e o desejo, de construir, nesse ambiente, aplicações com requisitos cada vez mais exigentes no que respeita à confiabilidade e ao tempo-real. No entanto, o carácter dinâmico e o tipo de sincronismo normalmente presente nesse tipo de infra-estruturas (fraco sincronismo) tornam difícil a materialização desse potencial. A obtenção de sistemas com uma relação custo-eficácia aceitável implica a existência de uma arquitectura com um elevado grau de flexibilidade e capacidade de adaptação. Essa adaptabilidade necessita de ser efectuada de uma forma segura e atempada de modo a não comprometer os objectivos aplicacionais. A *abordagem quasi-síncrona* permite lidar com esta situação de uma forma eficaz. Restringindo a uma pequena parte do sistema os requisitos de sincronismo, conseguem-se obter componentes síncronos susceptíveis de serem utilizados no controlo e validação do funcionamento dos restantes componentes. A detecção atempada de falhas temporais e a disseminação periódica de informação de controlo permitem criar um ambiente de suporte ao desenvolvimento de aplicações, capaz de oferecer as características de adaptabilidade e confiabilidade desejadas.**

***Palavras-chave*— Adaptabilidade de QdS, Comunicação em Grupo Tempo-Real, Sistemas Quasi-Síncronos**

I. Introdução e Motivação

Com a divulgação das redes de computadores, bem como o desenvolvimento e difusão de novas tecnologias de redes de comunicação, criou-se o potencial para a existência de um número crescente de novas aplicações com carácter distribuído e possuindo requisitos tanto de tolerância a faltas como de tempo-real. Contudo, a materialização desse potencial acarreta a resolução de problemas intrínsecos a este novo tipo de ambiente, onde, de uma forma geral, existem fracas garantias de sincronismo. Para além disso, as próprias aplicações podem também apresentar características dinâmicas, ou a necessidade de coexistir com outras aplicações que não são de tempo-real. Todos estes cenários tornam difícil (se não mesmo impossível ou demasiado caro) um controlo absoluto da carga do sistema. Nestas situações, em que é difícil avaliar um cenário de pior-caso, oferecer características tempo-real torna-se uma tarefa bastante complexa.

Num ambiente dinâmico, não é possível, ou pelo menos é demasiado cara, a utilização duma política de recursos adequados (*resource adequacy*) de forma a garantir todos os prazos. Por outro lado, a ausência de garantias torna impossível o funcionamento de aplicações que possuam simultaneamente requisitos de segurança (*safety*) e de pontualidade (*timeliness*).

Mesmo que não se pretenda suportar aplicações críticas de tempo-real estrito (*hard real-time*), pois isso seria praticamente impossível, ou extremamente caro, o suporte confiável de aplicações com requisitos temporais pode exigir mais do que uma simples política de melhor-esforço. Detecções atempadas de eventuais falhas temporais, e mecanismos de adaptação de forma segura e coerente, são factores que podem determinar a possibilidade, ou não, de suportar essas aplicações neste tipo de ambientes. Para além disso, este suporte deve oferecer uma relação custo-eficácia que seja aceitável para as aplicações em causa. Uma correcta escolha da arquitectura e abordagem a utilizar é fundamental para a obtenção desses objectivos.

Neste trabalho, é apresentada uma abordagem para lidar com este problema. Embora não sendo garantidos todos os prazos, são fornecidos os mecanismos para permitir obter segurança de forma atempada. Isto permite que novas classes de aplicações tempo-real possam executar-se, de uma forma compensadora, em cenários que lhes estavam anteriormente vedados.

Este artigo está organizado da seguinte forma: na próxima secção, são apresentadas as características fundamentais de um sistema quasi-síncrono e quais os principais problemas a resolver. Na Secção III, descreve-se a abordagem quasi-síncrona explicando como esta pode ser usada para obter flexibilidade e adaptabilidade de uma forma segura e atempada. Nas secções seguintes (Secção IV, Secção V e Secção VI), tendo como base a abordagem quasi-síncrona, são apresentadas várias contribuições complementares para a resolução do problema proposto. Mais concretamente, a utilização de replicação activa e hierarquia de grupos para lidar com falhas temporais, o ajuste dinâmico de forma segura e atempada de parâmetros associados à Qualidade-de-Serviço (QdS), a obtenção de um conjunto de QdS com base em componentes. Na descrição destas contribuições, são dados apontadores para algum trabalho prévio publicado em conferências internacionais. O artigo termina com as conclusões e algumas referências a trabalho relacionado.

Manuscrito recebido em Setembro de 2005. Este trabalho foi parcialmente suportado pela FCT, através do Projecto POSC/EIA/56041/2004 (DARIO).

C. Almeida integra o Departamento de Engenharia Electrotécnica e Computadores, AC-Computadores, Instituto Superior Técnico - Universidade Técnica de Lisboa, Av. Rovisco Pais - 1049-001 Lisboa - Portugal (email: cra@comp.ist.utl.pt).

II. Sistemas Quasi-Síncronos e Principais Problemas a Resolver

Os problemas associados à fiabilidade e ao tempo-real eram tradicionalmente tratados considerando sistemas síncronos, i.e. sistemas onde os limites temporais (na velocidade dos processadores, nos atrasos na transmissão das mensagens, na taxa de desvio dos relógios, etc.) são conhecidos [1],[2]. No entanto, a sua resolução torna-se mais complexa em ambientes dinâmicos e não completamente especificados, como os que são alvo da classe de aplicações que estamos a considerar. Na maior parte do trabalho publicado, este tipo de ambientes era tratado de acordo com pressupostos assíncronos, o que, do ponto de vista do tempo-real, é manifestamente de pouca utilidade.

Neste trabalho, é abordado o problema de se ter comunicação em grupo com características tempo-real num sistema que não é completamente síncrono (a que chamamos *quasi-síncrono* [3]). Por *quasi-síncrono*, entende-se que os limites anteriormente referidos (velocidade de processamento, atrasos na transmissão das mensagens e desvios nos relógios) existem, mas alguns ou todos eles não são conhecidos com precisão, ou os seus valores estão tão longe do caso normal que na prática são usados valores diferentes, mais próximos do caso normal. Em ambos os casos, isso significa que existe uma probabilidade não nula de que os valores escolhidos não sejam correctos (ver Figura 1). Neste tipo de sistemas, garantias temporais mais apertadas só são possíveis para um conjunto restrito de tarefas e mensagens de mais alta prioridade.

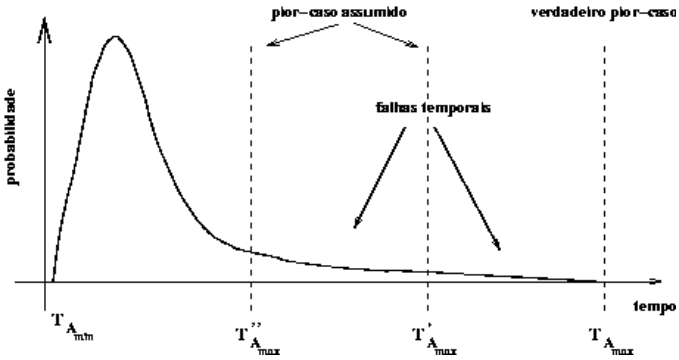

Fig. 1. Função de distribuição para a duração T_A de uma actividade genérica (processamento ou comunicação) num sistema quasi-síncrono.

A situação descrita atrás, relativamente à incerteza na identificação do cenário mais desfavorável (pior-caso), coloca os seguintes problemas, que têm sido objecto de estudo no nosso trabalho de investigação:

- Na tentativa de usar o verdadeiro pior-caso, é-se levado a escolher um valor muito elevado que está longe do caso normal. Se o protocolo de comunicação utilizar o valor correspondente a esse pior-caso como forma de obter alguma das suas propriedades (como, por exemplo, ordem), isso implicará uma baixa eficiência. Uma possível forma de tentar obter algumas optimizações é usar um protocolo de entrega antecipada [4]. A ideia base é utilizar mensagens de confirmação de baixo custo para tentar obter uma decisão antes do tempo correspondente ao pior-caso.
- Tendo em consideração que o pior-caso está longe do caso normal, a aplicação pode ter interesse em adoptar uma solução de compromisso entre pontualidade e falhas temporais através do ajuste do valor que é considerado como pior-caso (ver Figura 1). Isso levanta o problema do tratamento das falhas temporais e da capacidade de obter acordo de uma forma atempada. De modo a resolver este problema, é necessário um mecanismo adicional -- um serviço de detecção de falhas temporais. Este serviço é utilizado como um oráculo pelos protocolos de comunicação de forma a conseguirem obter segurança num tempo limitado.
- A solução proposta para resolver o problema referido no ponto anterior não resolve o problema da pontualidade *per se*. Consegue obter-se segurança (*safety*) de uma forma atempada, mas não uma pontualidade absoluta. No entanto, este resultado, embora não resolvendo todos os problemas, é útil para algumas aplicações tempo-real. A aplicação é notificada acerca da ocorrência de falhas temporais e é ela que é suposto tomar a decisão final acerca de como tratar a situação. Assim, é desejável que existam diversos protocolos com diferentes qualidades de serviço (QdS) que possam ser seleccionados pelas aplicações de modo a estas adaptarem as suas necessidades ao tipo de ambiente envolvente. Aspectos destas QdS são, por exemplo, as propriedades ordem, acordo e pontualidade associadas aos protocolos de comunicação em grupo.

III. Flexibilidade e Adaptabilidade de Forma Segura e Atempada

Nestas situações de incerteza relativamente a um cenário de pior-caso, é necessário adoptar uma solução de compromisso entre pontualidade e falhas temporais de modo a obter um sistema cujo custo seja compensador. A ideia é tentar tirar o máximo partido de um determinado ambiente, ao mesmo tempo em que se tenta satisfazer os requisitos fundamentais da aplicação. As aplicações devem poder seleccionar as qualidades de serviço desejadas e adaptarem-se sempre que o sistema se desloca de um determinado *envelope de funcionamento* para outro.

Um aspecto importante a resolver consiste na detecção atempada das situações em que não é possível cumprir todos os objectivos dentro do tempo limite especificado. Estas situações podem requerer um tratamento especial que pode variar de aplicação para aplicação. Se o sistema oferecer suporte ao nível da detecção destas situações e facilidades de reconfiguração, será então possível obter aplicações que melhor se adaptem a um determinado ambiente e consigam atingir objectivos que de outra forma seriam impossíveis de

satisfazer. Desta forma, será possível oferecer confiabilidade dentro de um determinado envelope de funcionamento, uma característica importante e que distingue a nossa aproximação das aproximações normalmente utilizadas em sistemas de tempo-real lato (*soft real-time*).

De modo a obter execuções que sejam simultaneamente correctas do ponto de vista lógico e atempadas, torna-se imprescindível a existência de um detector de falhas capaz de detectar falhas temporais. Tal como no sistema Isis [5] foi utilizado um detector de falhas para contornar o problema da impossibilidade de obter consenso num sistema assíncrono na presença de falhas [6], aqui pretende-se obter um detector de falhas que permita contornar alguns dos problemas associados a *pontualidade vs. segurança (timeliness vs. safety)*. É, pois, necessário um serviço de detecção de falhas temporais cuja execução seja efectuada num tempo limitado e conhecido, pois esta delimitação do tempo de execução é fundamental num sistema tempo-real [7].

Como é impossível obter consenso num sistema assíncrono na presença de falhas [6], teremos de utilizar canais de comunicação especiais (síncronos) para a realização deste serviço. No entanto, a restante comunicação poderá eventualmente utilizar canais menos exigentes, de acordo com as disponibilidades, funcionando desse modo de acordo com uma aproximação mais probabilista que é validada pelo recurso ao serviço de detecção de falhas.

Assim, combina-se a existência de partes síncronas e assíncronas num mesmo sistema, apenas exigindo as características síncronas numa parte restrita do mesmo. Esta solução -- *Abordagem Quasi-Síncrona* -- permite alargar o domínio de aplicabilidade de sistemas tempo-real. A arquitectura da abordagem quasi-síncrona está representada na Figura 2.

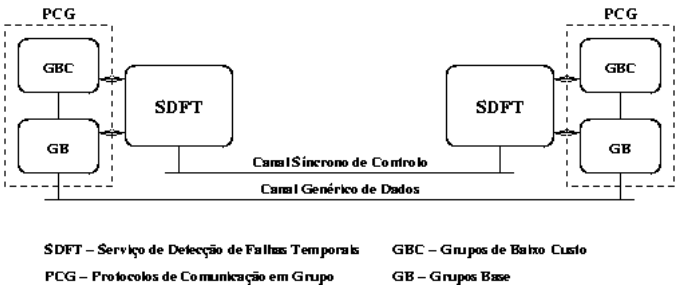

Fig. 2. Arquitectura da abordagem quasi-síncrona.

O Serviço de Detecção de Falhas Temporais (SDFT) constitui um componente de particular importância, uma vez que é ele que permite fornecer informação de controlo de forma atempada, e validar o correcto funcionamento dos restantes componentes do sistema. Funcionará como um oráculo que é, por exemplo, utilizado pelos protocolos de comunicação em grupo para obterem a informação necessária à satisfação das propriedades desejadas, de acordo com a qualidade de serviço seleccionada.

Em relação à necessidade de utilizar canais de comunicação especiais para o serviço de detecção de falhas, isso é realizável num grande número de situações, uma vez que o volume de informação trocada é reduzido (informação de controlo). Não sendo necessário um débito muito elevado, torna-se mais fácil obter um canal com melhores características temporais. Além disso, a possibilidade de existência de canais de comunicação privilegiados é contemplada em algumas redes de comunicação como no caso do Token-BUS [8], da rede ATM [9], ou através de mecanismos de reserva de recursos.

A arquitectura base da abordagem quasi-síncrona pode ser explorada de forma a obter uma infra-estrutura susceptível de ser utilizada em diversos cenários onde a capacidade de adaptação, de forma segura e atempada, é de extrema importância. Nas secções seguintes, são apresentados vários aspectos complementares que contribuem para esse objectivo.

IV. Hierarquia de Grupos e Replicação Activa

Uma possível forma de lidar com falhas temporais consiste na utilização de replicação activa. Desde que possa ser assumido um modo de falha independente, a existência de mais do que um componente a fornecer um determinado serviço pode permitir que esse serviço continue a ser oferecido de forma atempada, mesmo quando alguns desses componentes são afectados por falhas temporais. Obviamente, a coordenação dos vários componentes envolvidos, sobretudo quando existe a necessidade de efectuar operações de actualização, exige cuidados particulares de forma a manter a coerência do sistema.

A utilização de protocolos de comunicação em grupo associada à existência de um serviço de filiação é algo de extrema importância neste tipo de cenários. No entanto, de forma a manter uma visão coerente do estado global do sistema, é necessário que a informação de controlo possa ser disseminada entre os vários elementos do grupo de forma segura e atempada. A abordagem quasi-síncrona com o seu serviço de detecção de falhas temporais permite resolver este problema.

Para além disso, de forma a melhor gerir as situações em que de facto algum membro do grupo sofre uma falha temporal, o recurso a uma estrutura hierárquica na gestão dos grupos permite lidar com as falhas temporais de um modo mais eficiente. Os grupos estão organizados em dois níveis: grupos base (GB) e grupos de baixo custo (GBC). Os grupos base têm um comportamento mais tradicional, ficando associados às falhas por paragem, enquanto os grupos de baixo custo, localizados a um nível superior, lidam directamente com as falhas temporais. Deste modo, é possível garantir que uma eventual situação de incoerência temporária entre os vários elementos do grupo possa ser controlada evitando situações de contaminação. Isso é conseguido através da remoção do elemento faltoso do grupo de baixo custo. A sua exclusão, feita de forma atempada e coerente por todos os elementos do grupo, força um comportamento de *falha silenciosa* impedindo assim que a existência de um estado de incoerência seja propagado a outras partes do

sistema (ver Figura 3).

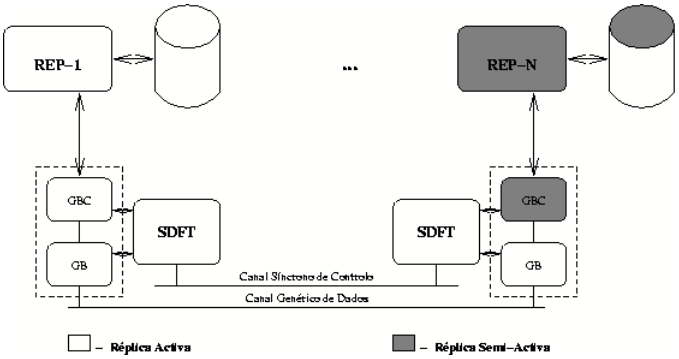

Fig. 3. Uso do SDFT e GBC numa base de dados tempo-real distribuída.

A remoção atempada e coerente do elemento faltoso do grupo faz com que seja possível não comprometer as propriedades de segurança e pontualidade dos restantes elementos do grupo. Esta remoção, consequência de uma falha temporal, apenas é feita ao nível do grupo de baixo custo. O elemento continuará a pertencer ao grupo base e poderá continuar a receber as mensagens atrasadas de forma a permitir uma recuperação mais rápida. Quando atingir um estado coerente, será reinserido no grupo de nível superior. Tudo isto é feito com o auxílio do serviço de detecção de falhas de forma a garantir a coerência destas operações.

Com este tipo de arquitectura, e uma escolha adequada do número de réplicas, é possível utilizar uma política mais agressiva na escolha do limiar que define a ocorrência de uma falha temporal, e assim obter melhorias significativas na pontualidade do sistema, uma vez que não é necessário esperar por eventuais elementos atrasados.

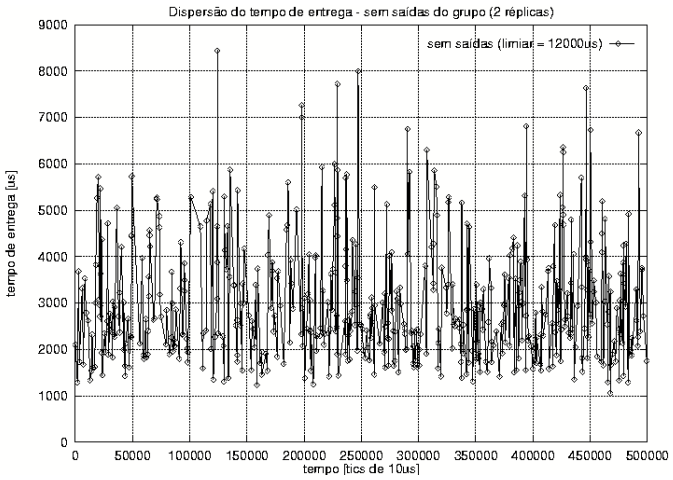

Fig. 4. Replicação activa com actualizações (dispersão do tempo de entrega). Uso do SDFT e GBC. Sem saídas do grupo (limiar = 12000 μs, 2 réplicas).

Uma descrição mais pormenorizada da utilização de grupos de baixo custo e replicação activa, com apresentação e discussão de alguns resultados, é feita em [10]. A título de exemplo, nas Figuras 4 e 5 mostra-se o que acontece nessas situações em que se reduz o limiar mas se aumenta o número de réplicas e se força a exclusão dos elementos atrasados.

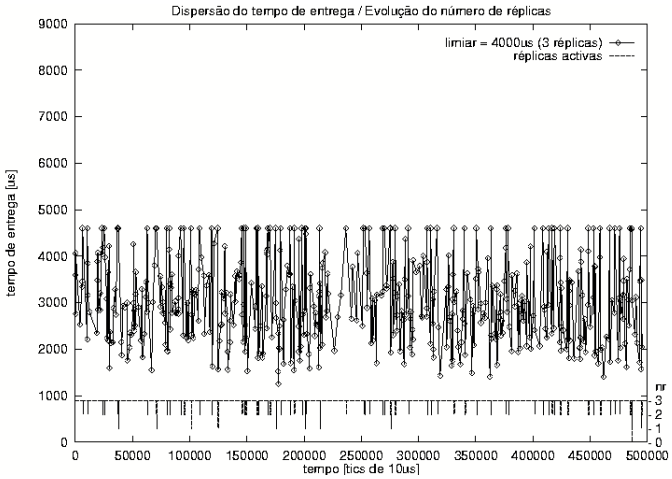

Fig. 5. Replicação activa com actualizações (dispersão do tempo de entrega). Uso do SDFT e GBC. Saídas e entradas no grupo (limiar = 4000 μs, 3 réplicas).

V. Ajuste de Parâmetros de Forma Dinâmica

A utilização de replicação activa para mascarar as falhas temporais, em conjunto com os grupos de baixo custo, é uma abordagem interessante que permite melhorar a pontualidade. No entanto, a sua cobertura está condicionada ao pressuposto de independência de falhas, e, se o limiar usado para decidir da ocorrência de uma falha temporal for demasiado apertado, existe uma probabilidade não nula de todos os elementos do grupo falharem. Isto pode acontecer no caso em que as condições do ambiente de funcionamento se degradam e o valor escolhido para limiar é demasiado optimista. Em situações destas, é de toda a conveniência existir a possibilidade de ajustar esse limiar de forma dinâmica. Isto corresponde a uma degradação da QdS oferecida (pontualidade), mas a alternativa seria a eventual falha do sistema, pelo que, para algumas aplicações, justifica-se esta adaptabilidade dinâmica da QdS. No exemplo de aplicação apresentado na Secção IV (e [10]), relativo a uma base de dados tempo-real distribuída e replicada, a utilização de um limiar demasiado optimista pode implicar a falha da aplicação. Todos os elementos do grupo podem ser excluídos, e o serviço deixaria de estar disponível temporariamente. O ajuste dinâmico do limiar, embora degradando a pontualidade, pode permitir a continuidade do serviço.

No entanto, o ajuste dinâmico de parâmetros relativos à QdS oferecida necessita de ser feito de forma coerente por todos os elementos do grupo. A actualização de um determinado parâmetro não pode ser feita de forma completamente arbitrária, a nível local. Terá de haver um acordo com os restantes elementos do grupo, ter em atenção eventuais mensagens já enviadas e ainda não entregues, e, dependendo de se estar a aumentar ou a diminuir o parâmetro, assegurar que as fases de transição são realizadas de forma segura e atempada.

A informação de controlo necessária a estas adaptações é disseminada pelo SDFT, mas é preciso que os protocolos de

comunicação em grupo realizem estas operações de forma correcta, para não criarem situações de incoerência. Algum trabalho relativamente a este assunto é apresentado em [11], onde se analisam as fases de transição associadas ao aumento ou diminuição do limiar que determina a ocorrência de falhas temporais. Nomeadamente, a alteração desse limiar tem de ser efectuada de forma a garantir que, relativamente a uma dada mensagem em curso, não existam situações em que alguns elementos do grupo utilizem o valor antigo para tomar a decisão, enquanto outros elementos do grupo utilizam o novo valor.

VI. Arquitectura Modular para a Concretização de Diversas QdS

Um outro aspecto importante no tipo de abordagem que aqui é feito tem que ver com as diversas qualidades de serviço (QdS) a oferecer pelo sistema às aplicações. Para o tipo de aplicações alvo considerado, é importante que estas tenham a capacidade de se adaptarem ao ambiente envolvente.

Quando, devido à ocorrência de faltas ou a situações de sobrecarga, não é possível continuar a garantir uma determinada QdS, pode ser vantajoso comutar, de uma forma controlada, para uma outra QdS menos exigente do ponto de vista de recursos, e desta forma permitir que a aplicação continue a efectuar progressos ainda que de um modo degradado (*graceful degradation*).

Para suportar este tipo de funcionamento, o sistema deve oferecer diversas QdS susceptíveis de serem seleccionadas pelas aplicações, bem como possuir mecanismos que permitam detectar e notificar as situações em que essa comutação de QdS é *aconselhável*. Este tipo de funcionamento corresponde à existência de diversos envelopes de trabalho. Uma dada aplicação executar-se-á normalmente num determinado envelope de trabalho, mas poderá, temporariamente, transitar por outros envelopes de acordo com a evolução experimentada pelo sistema, considerado na sua globalidade.

Neste tipo de situações, a adaptabilidade requerida pode ser mais do que o simples ajuste de parâmetros, é conveniente existir a possibilidade de seleccionar as propriedades que constituem a QdS (por exemplo, acordo, ordem, pontualidade). Se, em vez da aplicação apresentada nas secções anteriores (base de dados replicada), tivermos uma aplicação menos exigente do ponto de vista de QdS, em que alguma das propriedades do protocolo de comunicação não seja fundamental, ou que seja dispensável temporariamente, seria útil poder dispor de um conjunto diversificado de protocolos, e, deste modo, permitir uma melhor adaptação a um determinado cenário operacional.

De modo a permitir uma maior flexibilidade na construção de um conjunto de protocolos com diversas QdS, é útil ter diversos módulos (componentes), correspondendo a cada um deles uma das propriedades base (acordo, ordem, pontualidade) que definem a QdS desejada. Dependendo das necessidades da aplicação e da importância relativa destas propriedades, pode fazer-se uma combinação dos componentes apropriados, de forma a obter a QdS desejada. Basicamente, cada um destes componentes base tem a seguinte funcionalidade:

pontualidade: descartar as mensagens que falharam o limite temporal.

ordem: colocar em fila de espera as mensagens até poder ser garantida a ordem desejada.

acordo: colocar em fila de espera as mensagens até que o acordo seja satisfeito.

Cada componente funciona como um ``filtro'' que, para além de receber e entregar mensagens de dados, pode ser parametrizado, receber informação de controlo e fornecer informação de controlo (notificações) (Figura 6). As mensagens que não satisfazem a propriedade especificada são descartadas (ou podem ser recolhidas para uma actualização mais rápida). A parametrização permite a especificação de um limite temporal para esperar pela obtenção da respectiva propriedade relativamente a uma dada mensagem. Isto permite evitar potenciais atrasos não delimitados na obtenção de uma decisão. Se isso não constituir um problema, um valor ``infinito'' pode ser usado como parâmetro. Existe também uma entrada de ``suspender'' que pode ser usada pelo protocolo de filiação para suspender esse membro (excluí-lo do grupo para evitar problemas de contaminação).

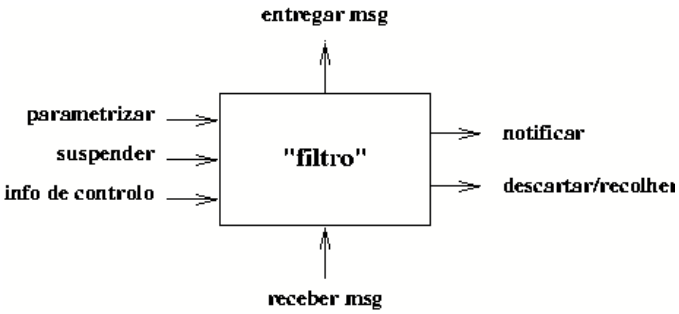

Fig. 6. Arquitectura de um componente.

A informação de controlo é obtida e disseminada utilizando o componente Filiação e os serviços do SDFT (abordagem quasi-síncrona). Deste modo, essa informação fica disponível em cada nodo / participante de uma forma atempada. Isto é importante para assegurar a segurança de forma atempada. A incorporação dos componentes relativos à QdS na arquitectura Quasi-Síncrona encontra-se representada na Figura 7. Um determinado componente será utilizado, ou não, dependendo da QdS pretendida para o protocolo de comunicação em grupo.

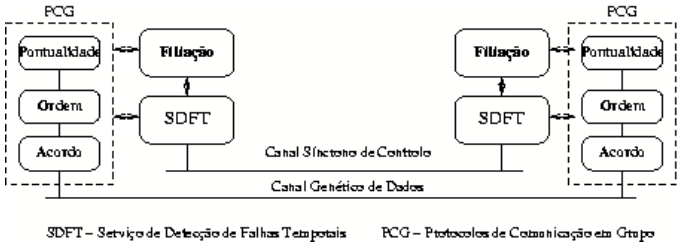

Fig. 7. Componentes QdS na arquitectura Quasi-Síncrona.

O modo como os vários componentes são combinados pode não ser indiferente. A sua ordem dependerá da importância atribuída a cada uma das propriedades pretendidas. Alguns parâmetros poderão ser propagados de um componente para os outros. Por exemplo, se o componente ``pontualidade'' estiver a ser utilizado, o valor especificado como limiar para consideração de falhas temporais poderá ser propagado aos restantes componentes como tempo máximo de espera pela obtenção da respectiva propriedade.

Algumas considerações adicionais sobre a utilização de componentes na obtenção de QdS podem ser encontradas em [12] e [13].

VII. Conclusões e Algum Trabalho Relacionado

As novas infra-estruturas de comunicação apresentam um ambiente dinâmico onde existem algumas incertezas no domínio temporal. Neste tipo de ambiente, é muito difícil suportar aplicações que tenham requisitos de confiabilidade e tempo-real. No entanto, existe uma procura crescente deste tipo de aplicações. A melhor solução para lidar com este problema, de uma forma compensadora, consiste em fornecer alguma capacidade de adaptação. Mesmo que não seja possível, num determinado momento, assegurar o serviço desejado de uma forma completa, poder escolher de uma forma segura e atempada a melhor QdS que se ajusta a um determinado cenário, é algo de extrema importância.

A abordagem quasi-síncrona, em que uma pequena parte do sistema, com características síncronas, é usada para controlar e validar os restantes componentes do sistema global, fornece os mecanismos para se atingir esse objectivo.

A utilização de replicação activa e uma estrutura hierárquica na gestão de grupos permite lidar de uma forma mais ``agressiva'' com as falhas temporais, facilitando o seu mascaramento, dentro de determinadas condições.

Quando a evolução do sistema se afasta significativamente de um determinado ponto de equilíbrio, a necessidade de preservar a cobertura pode implicar a conveniência em ajustar determinados parâmetros de forma dinâmica. De forma a preservar a coerência do sistema, esse ajuste deve ser feito de forma segura e atempada. A abordagem quasi-síncrona dá-nos os meios para conseguir esse objectivo.

No caso de ambientes em que as variações podem ser mais significativas, um simples ajuste de parâmetros pode não ser suficiente. A existência de um conjunto de diversas QdS, facilmente disponíveis para serem utilizadas pelas aplicações, constitui algo de interessante nesse tipo de ambiente. A possibilidade de obter esse conjunto de QdS com base em componentes associados às propriedades base que formam a QdS revela-se uma mais-valia que oferece um grande grau de liberdade na criação de uma infra-estrutura extremamente flexível.

A adaptabilidade e flexibilidade obtidas de forma segura é algo de extrema importância para lidar com os requisitos de tempo-real e tolerância a faltas por parte das aplicações emergentes nas novas infra-estruturas de comunicação. A arquitectura aqui apresentada cria as condições para suportar, neste tipo de ambientes, novas classes de aplicações que de outro modo não seriam possíveis.

A. Algum Trabalho Relacionado

Uma outra linha de investigação relacionada com o controlo da QdS em sistemas tempo-real adaptativos é aquela que tenta aplicar a teoria da área do controlo a este problema. Um exemplo desse trabalho é apresentado em [14], em que é usado controlo por realimentação (*feedback control*) para conseguir as desejadas respostas dinâmicas.

A adaptabilidade da QdS é um assunto que tem recebido alguma atenção nos últimos anos [15]. No entanto, a maior parte do trabalho existente preocupa-se mais com aplicações multimedia, lidando com adaptações ao nível dos atrasos e débitos, mas sem grande preocupação com os aspectos relativos à segurança e pontualidade dos procedimentos de adaptação. No nosso caso, suportamos protocolos de comunicação em grupo onde é importante que todos os elementos do grupo tenham uma visão coerente do sistema. Isto implica que a adaptação da QdS tenha de ser feita de uma forma segura e atempada. O termo ``QdS'' também é por nós usado num sentido mais genérico do que aquele que é normalmente utilizado no contexto das aplicações multimedia. Para nós, uma dada QdS pode ser definida com base em propriedades dos protocolos de comunicação em grupo (ordem, acordo e pontualidade).

No que diz respeito a modelos de sincronismo, existe um outro trabalho que, do ponto de vista de objectivos, apresenta algumas semelhanças com o nosso trabalho. Esse trabalho é baseado num modelo designado por *Timed Asynchronous* [16]. Esse modelo de sistema assume que os vários processos têm acesso a relógios locais que, embora não estando sincronizados, apresentam uma taxa de desvio limitada. O modelo de falhas considerado assume que os processos podem falhar por paragem (*crash*) ou experimentar falhas temporais (*performance failures*) e que relativamente às mensagens podem existir omissões ou também falhas temporais. O modelo não considera nenhum limite relativamente à taxa de falhas. Sobre este modelo, é construído um serviço de datagramas que tem conhecimento das falhas (*fail-aware*). Essa informação pode ser utilizada por outros módulos de nível superior para construir, por exemplo, um serviço de filiação que permite definir partições lógicas. Assim, é possível obter um sistema que tem conhecimento das falhas e que permite construir aplicações

com um estado de paragem seguro (*fail-safe*) [17].

Esse trabalho, que em termos de objectivos apresenta alguma afinidade com o trabalho aqui apresentado (construção de aplicações tempo-real seguras em ambientes não completamente síncronos), é baseado num modelo (*Timed Asynchronous*) que, no ``espectro" síncrono-assíncrono, está mais próximo do assíncrono do que o nosso modelo *Quasi-Síncrono*. Desta forma, no modelo Quasi-Síncrono, consegue-se obter qualidades de serviço mais exigentes do que as que são possíveis de obter usando o modelo *Timed Asynchronous*. Ambos os modelos permitem a construção de aplicações com uma paragem segura. No entanto, o modelo Quasi-Síncrono permite uma maior flexibilidade na reconfiguração do sistema. Devido à informação de controlo, que é fornecida pelo SDFT de forma atempada, é possível chegar a um acordo antes de decidir de que modo essa reconfiguração deve ser feita.

A existência de membros do grupo que se auto-excluem aquando da detecção de falha também é usada em [18]. No entanto, nesse trabalho, não é usada uma estrutura hierárquica para os grupos, e os membros que se auto-excluem fazem-no parando (*crash*). Uma outra diferença diz respeito à possibilidade de, no nosso caso, se poder chegar a um acordo antes de tomar a decisão de sair, ou não, do grupo.

Mais recentemente, houve também algum trabalho tentando unificar os modelos *Timed Asynchronous* e Quasi-Síncrono [19]. Apresentado como uma extensão ao modelo Quasi-Síncrono, é utilizada uma abordagem semelhante: uma parte síncrona do sistema é usada para controlar e fornecer serviços à outra parte.

VIII. Referências

[1] F. Cristian, H. Aghili, R. Strong, and D. Dolev, "Atomic broadcast: From simple message diffusion to byzantine agreement," IBM," Technical Report RJ-5244, July 1986, an early version appeared in Proc. of the Fifteenth International Symposium on Fault-Tolerant Computing, Michigan, June 1985.

[2] H. Kopetz, A. Damm, C. Koza, M. Mulazzani, W. Schwabl, C. Senft, and R. Zainlinger, "Distributed Fault-Tolerant Real-Time Systems: The Mars Approach," *IEEE Micro*, pp. 25-41, Feb. 1989.

[3] P. Veríssimo and C. Almeida, "Quasi-synchronism: a step away from the traditional fault-tolerant real-time system models," *Bulletin of the Technical Committee on Operating Systems and Application Environments (TCOS), IEEE Computer Society*, vol. 7, no. 4, pp. 35-39, Winter 1995.

[4] C. Almeida and P. Veríssimo, "An adaptive real-time group communication protocol," in *Proceedings of the First IEEE Workshop on Factory Communication Systems*, Leysin, Switzerland, Oct. 1995.

[5] A. M. Ricciardi and K. P. Birman, "Using process groups to implement failure detection in asynchronous environemnts," in *Proceedings of the 10th ACM Symposium on Principles of Distributed Computing*, Aug. 1991, pp. 341-351.

[6] M. J. Fischer, N. A. Lynch, and M. S. Paterson, "Impossibility of distributed consensus with one faulty process," *Journal of the Association for Computing Machinery*, vol. 32, no. 2, pp. 374-382, Apr. 1985.

[7] C. Almeida and P. Veríssimo, "Timing failure detection and real-time group communication in quasi-synchronous systems," in *Proceedings of the 8th Euromicro Workshop on Real-Time Systems*, L' Aquila, Italy, June 1996.

[8] *ISO DIS 8802/4-85, Token Passing Bus Access Method*, 1985.

[9] M. de Prycker, *Asynchronous Transfer Mode: Solution For Broadband ISDN (Third Edition)*. Prentice Hall, 1995, no. ISBN 0-13-342171-6.

[10] C. Almeida and P. Veríssimo, "Using light-weight groups to handle timing failures in quasi-synchronous systems," in *Proceedings of the 19th IEEE Real-Time Systems Symposium*, Madrid, Spain, Dec. 1998.

[11] C. Almeida, "Dynamic QoS adaptability in quasi-synchronous systems," in *Proceedings of the 14th IASTED International Conference on Parallel and Distributed Computing and Systems - PDCS2002*. Cambridge, USA: IASTED, Nov. 2002, pp. 691-696.

[12] C. Almeida, "Handling QoS in a dynamic real-time environment," in *Proceedings of the 8th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems - WORDS 2003*. Guadalajara, Mexico: IEEE, Jan. 2003.

[13] C. M. R. Almeida, "Component based QoS for real-time group communications," in *Proceedings of the IASTED International Conference on Parallel and Distributed Computing and Networks - PDCN2004*. Innsbruck, Austria: IASTED, Feb. 2004.

[14] C. Lu, J. A. Stankovic, T. F. Abdelzaher, G. Tao, S. H. Son, and M. Marley, "Performance specifications and metrics for adaptive real-time systems," in *Proceedings of the 21st IEEE Real-Time Systems Symposium*, Orlando, Florida, USA, Dec. 2000.

[15] C. Diot, "Adaptive applications and qos guaranties," in *IEEE Multimedia Networking*, Aizu (Japan), Sept. 1995, pp. 27-29.

[16] F. Cristian and C. Fetzer, "The timed asynchronous system model," in *Proceedings of 28th Annual International Symposium on Fault-Tolerant Computing*, Munich, Germany, June 1998.

[17] C. Fetzer and F. Cristian, "Fail-awareness: An approach to construct fail-safe applications," in *Proceedings of 27th Annual International Symposium on Fault-Tolerant Computing*, Seattle, Washington, USA, June 1997.

[18] T. Abdelzaher, A. Shaikh, F. Jahanian, and K. Shin, "RT-CAST: Lightweight multicast for real-time pro cess groups," in *Proceedings of Real-Time Technology and Applications Symposium*. Boston, MA: IEEE, June 1996.

[19] P. Veríssimo, A. Casimiro, and C. Fetzer, "The Timely Computing Base: Timely actions in the presence of uncertain timeliness," in *Proceedings of the International Conference on Dependable Systems and Networks*, New York, USA, June 2000.

IX. Biografia

Carlos R. Almeida (M'96) Carlos Almeida licenciou-se em Engenharia Electrotécnica e Computadores no Instituto Superior Técnico, Universidade Técnica de Lisboa, em 1984. Em 1989 e 1999, completou, respectivamente, o Mestrado e o Doutoramento em Engenharia Electrotécnica e Computadores nessa mesma Universidade. Desde 1985 que lecciona no Instituto Superior Técnico no Departamento de Engenharia Electrotécnica e Computadores, Área Científica de Computadores, onde é actualmente Professor Auxiliar. Foi investigador no INESC de 1984 a 1996. Em 1990-1991, efectuou um estágio junto da empresa francesa Chorus Systemes, e de 1991 a 1993 efectuou um estágio junto da Universidade de Cornell, EUA, no grupo ISIS (Computer Science Department). Tem participado em diversos projectos nacionais e internacionais. Os seus principais interesses de investigação são: sistemas tempo-real, sistemas embebidos, sistemas distribuídos, tolerância a faltas e redes de sensores.