Tolerância a Faltas em Sistemas Multiagentes Multidimensionais

Luis P. Lampert[1], **Jomi F. Hübner**[1], **Maicon R. Zatelli** [1]

[1]Universidade Federal de Santa Catarina (UFSC)
Florianópolis – SC – Brasil

lp.lampert@gmail.com, jomi.hubner@ufsc.br, maicon.zatelli@ufsc.br

Resumo. *A autonomia presente nos Sistemas Multiagentes (SMA) é uma característica que contribui para que esse tipo de sistema seja suscetível a faltas. Com a finalidade de aumentar a confiabilidade, a tolerância a faltas aplicada a SMA é um tópico que vem ganhando destaque. Este trabalho tem como objetivo propor um modelo de tolerância a faltas para SMA que utiliza as abstrações fornecidas pela programação multidimensional (agentes, ambiente e organização). O modelo proposto adiciona a capacidade de monitoramento do estado dos agentes através da instrumentação do ambiente. Além disso, propõe a adição de agentes especializados, que têm a capacidade de monitorar e atuar nos casos de detecção de faltas no sistema.*

1. Introdução

A utilização de computadores é tão presente no cotidiano das pessoas que poucas vezes é questionada qual a confiabilidade dos serviços que esses sistemas oferecem. A ocorrência de falhas nesses sistemas pode ter consequências que variam de simples incomodações até eventuais catástrofes, com sérios prejuízos econômicos e até perda de vidas humanas. Nesse cenário, a *tolerância a faltas* (TF) pode ser descrita como sendo um meio para garantir que o sistema entregue o serviço que originalmente foi projetado, mesmo na presença de erros originados por faltas [Laprie 1992].

Segundo [Potiron et al. 2013], quando se fala de sistemas computacionais tradicionais, a TF é um tópico estudado a bastante tempo e essa base comum de conhecimento é amplamente compartilhada, o que facilita as especificações de sistemas, implementações de técnicas de TF e comparações entre diferentes abordagens. Entretanto, no caso dos SMA, não existem muitos trabalhos que analisam a TF e levam em consideração as particularidades da *autonomia*, que é a principal característica que distingue um SMA de um sistema tradicional. Assim, o estudo de abordagens que sejam adaptadas às especificidades dos SMAs mostra-se importante para o aprimoramento da confiabilidade dos SMAs, permitindo que esses sejam utilizados cada vez mais em aplicações que necessitam de uma maior robustez.

A fim de contribuir para o desenvolvimento da tecnologia de SMA, o presente trabalho apresenta uma proposta inicial de modelo de TF adaptado aos SMA que utilizam as abstrações de multidimensões (agente, ambiente e organização), apresentadas no trabalho de [Boissier et al. 2013]. Até onde se investigou, não foi encontrado trabalho que utilizasse um modelo de TF em SMA e considerasse outras dimensões além da dos agentes. O trabalho faz parte de uma dissertação em andamento e apresenta um estudo

das abordagens de tolerância a faltas existentes na Seção 2, propõe um modelo adaptado à programação multidimencional na Seção 3 e, por último, na Seção 4 faz algumas observações finais.

2. Tolerância a Faltas em Sistemas Multiagentes

Em sistemas computacionais o termo dependabilidade aparece com frequência, trata-se de uma tradução literal do termo em inglês *dependability*, e indica a qualidade do serviço oferecido pelo sistema. Quando a dependabilidade é deficiente, três efeitos podem surgir: a *falha*, o *erro* e a *falta*. Por definição, uma falha ocorre quando o sistema não produz a mesma entrega para o qual este foi especificado inicialmente. O erro é uma parte dos estados do sistema que podem levar a uma falha subsequente. O surgimento de um erro é uma indicação de que uma falta ocorreu no sistema [Laprie 1992]. Quando adicionamos métodos de tolerância a faltas ao sistema, estamos tentando prevenir a ocorrência das falhas através de ações que devem ser tomadas quando os erros são detectados, ou seja, os erros podem ser detectados e, nesse momento, as técnicas de tolerância a faltas devem entrar em ação para tentar evitar que falhas aconteçam.

Para o projeto de um sistema tolerante a faltas é imprescindível que o projetista conheça quais são as faltas que o sistema está suscetível. Entretanto, quando se trata de um SMA, esse tipo de previsão não pode ser feita pois os agentes do sistema possuem leis e objetivos próprios e, em muitos casos, o projetista não tem acesso ou conhecimento dessas informações [Potiron et al. 2013]. Essa *autonomia* é a característica fundamental dos SMA e deve ser levada em conta em qualquer projeto de tolerância a faltas.

Muitos trabalhos que abordam a TF em SMA acabam fazendo abordagens bastante específicas, normalmente lidando com apenas alguns modos de falta, que são os mais comuns para aquele sistema específico [Isong and Bekele 2013]. Os trabalhos de [Stanković et al. 2017] e [Isong and Bekele 2013] investigaram diversas abordagens existentes e listaram três principais características que as classificam: *tipo de detecção de faltas*, *protocolo de tolerância a faltas* e *nível de manipulação de falta*.

2.1. Detecção de Faltas

A detecção de faltas é uma característica chave para as abordagens de TF pois, sem uma detecção adequada, mesmo o melhor tipo de tratamento será ineficaz. As técnicas de detecção podem ser divididas em duas categorias: detecção por *batimento cardíaco* e através da *interação*. Na detecção por batimento cardíaco, as faltas em agentes são identificadas através da troca de mensagens periódicas de confirmação (*acknowledge*) entre agentes comuns e agentes monitores. Essa técnica possui uma implementação simples e permite uma arquitetura modular, porém acaba inundando o sistema com uma série de mensagens. Na detecção por interação, aproveita-se a relação já existente entre agentes, identificando possíveis erros através de ferramentas como *timeout* ou erros em retorno de chamadas (*callbacks*).

2.2. Protocolo de Tolerância a Faltas

O protocolo de TF define como será o procedimento padrão de ação na ocorrência de faltas no sistema. Um dos protocolos mais utilizado é chamado de *não bloqueante*, que permite o reinício de processos que falharam. Dentro dessa categoria, as técnicas que frequentemente aparecem são baseadas em *recuperação de estados* anteriores e na *replicação* de

agentes. A recuperação tenta restaurar o sistema a um estado anterior à ocorrência da falta, evitando assim que processos devam ser reiniciados de pontos iniciais muito distantes dos atuais. Essa técnica necessita que sejam feitos processos periódicos de gravação dos estados do sistema, conhecidos como *checkpoints*, o que acaba demandando custo computacional. A abordagem por replicação utiliza a ideia de manter copias de agentes no sistema, que podem substituir eventuais agentes em estado de erro.

2.3. Nível de Manipulação de Faltas

A última característica indica em qual nível deve ocorrer o gerenciamento das exceções e a recuperação das faltas: no nível dos agentes ou do SMA. Esse tipo do classificação também é abordada no trabalho de [Klein et al. 2003], que utiliza as denominações de abordagem "sobrevivente" e "cidadã".

Na abordagem sobrevivente, adota-se a ideia de que cada agente deve ser responsável por manter a sua própria existência, ou seja, a programação das técnicas de TF devem ser adicionadas juntamente ao código de cada agente. A vantagem dessa abordagem é a independência de plataforma, que permite uma melhor interoperabilidade entre sistemas. Entretanto, uma maior responsabilidade recai sobre os programadores, que devem coordenar o comportamento dos agentes na ocorrência de faltas, que são, muitas vezes, difíceis de prever. Essa adição de código defensivo nos agentes resulta em uma maior dificuldade para manutenção, prejudica o desempenho do sistema e aumenta as chances de ocorrência de novas faltas inseridas pelos programadores.

A abordagem cidadã estabelece que os processos TF devem ser gerenciados por uma entidade especializada e externa ao agente, habitualmente chamada de *sentinela*. Utilizando uma analogia similar a sociedade humana, a abordagem cidadã adota a ideia da criação de instituições que são reconhecidas pelos agentes (os cidadãos). Por exemplo a polícia em uma sociedade, essa organização oferece serviço de segurança aos cidadãos e, em troca, é reconhecida por eles como uma autoridade que tem permissão para atuar no estado de cada um deles, podendo, por exemplo, prender indivíduos que estejam causando problemas aos demais. Segundo [Klein et al. 2003], a principal vantagem neste tipo de abordagem é a noção de expertise para tratamento de faltas, simplificando assim a programação dos agentes "comuns", que ficam focados nos seus objetivos mais básicos. A principal desvantagem dessa abordagem é a redução da interoperabilidade entre sistemas devido à necessidade de utilizar plataformas específicas.

2.4. Trabalhos relacionados

Em seu trabalho, [Hägg 1997] introduziu o termo *sentinela*, uma classe especial de agente que tem a função de proteger algumas funcionalidades do sistema. Esse tipo de agente não pertence ao domínio da solução de problemas, ficando responsável por monitorar os agentes comuns do sistema e, se for detectado a ocorrência de alguma falta, o sentinela pode tomar ações corretivas, como escolher planos alternativos, excluir agentes, alterar parâmetros do sistema e reportar aos operadores humanos. O sentinela pode monitorar a comunicação dos agentes e interagir com eles, além de usar *timers* para detectar agentes mortos ou faltas em links de comunicação.

Em sua tese, [Díaz 2018] apresentou o eJason, um interpretador da linguagem de programação Jason em Erlang. Sua principal contribuição foi o projeto e implementação

deste ambiente de desenvolvimento para SMA que combina as sintaxes e semânticas do Jason, que é uma linguagem de programação para SMA que possui alto nível de abstração para programação de agentes BDI (*belief-desire-intention*), com as ferramentas de tolerância a faltas do Erlang, uma linguagem de programação com características que favorecem o desenvolvimento de aplicações concorrentes, distribuídas e com alta robustez. Entre as ferramentas de TF fornecidas pelo eJason, destacam-se dois tipos de relações que podem ser estabelecidas entre os agentes do sistema: a relação de *monitor* e de *supervisor*. A primeira relação permite que agentes monitores recebam informações sobre acontecimentos nos agentes monitorados. Já a segunda relação permite atuações especiais de controle do agente supervisor sobre os seus supervisionados.

3. Proposta de um Modelo de Tolerância a Faltas para Sistemas Multiagentes Multidimensionais

O modelo proposto neste trabalho faz uma analogia com um sistema de monitoramento de saúde remoto, onde existem equipamentos móveis que acompanham cada indivíduo, monitorando a sua saúde e enviando informações para equipes especializadas que podem agir em eventuais casos de emergência. Esses monitores possuem informações disponíveis em uma tela visível para quem tiver interesse em observa-lá, mas apenas integrantes das equipes especializadas têm acesso a informações sigilosas, habilidade de fazer diagnósticos a respeito da saúde do indivíduo e tomar ações corretivas.

Neste cenário, a proposta inicial de modelo é ilustrada na Figura 1, que representa o SMA com a adição das duas entidades básicas do serviço de TF: o artefato *monitor_de_saude*, instrumentado na dimensão do ambiente e os *agentes_de_saude*, que estão inseridos na dimensão dos agentes.

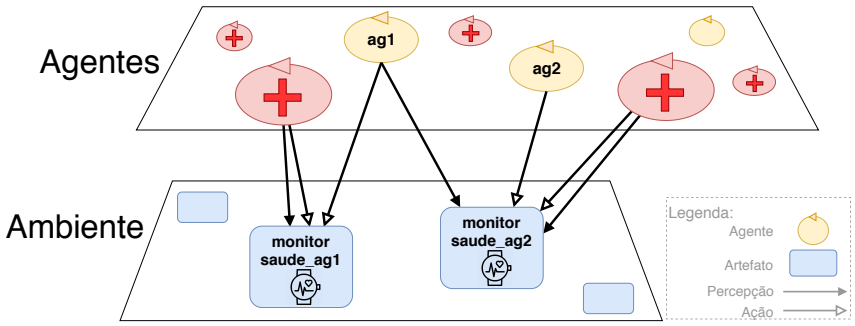

Figura 1. Modelo de TF proposto

Nesta configuração, cada agente do sistema pode ter um artefato monitor de saúde atrelado a si. O monitor mantém uma relação exclusiva com o agente, verificando periodicamente o seu estado de saúde. A saúde do agente é indicada no monitor em forma de batimento cardíaco e a ocorrência de faltas (morte do agente) aparece como a ausência de sinais vitais. Além do monitor cardíaco, outras informações podem ser verificadas nas propriedades observáveis do monitor de saúde, tais como os desejos e intenções dos agentes.

3.1. Exemplo ilustrativo

Para ilustrar o funcionamento do modelo proposto, propõem-se um SMA com três agentes principais: o agente `consumidor`, o agente `pizzaiolo` e o agente `entregador`.

O agente consumidor tem apenas um objetivo, que é `comer_pizza`, e para concretizar esse objetivo ele põe em prática o plano de `ligar_pizzaria` para fazer o seu pedido e fechar um contrato com o agente pizzaiolo. Na sequência, o agente pizzaiolo põe em prática o plano `fazer_pizza` e também `contratar_entrega` juntamente ao agente entregador que, ao ser contratado, executa as funções de `pegar_pizza` e depois `realizar_entrega`.

Durante a entrega da pizza, o agente entregador pode sofrer um acidente e falhar no seu objetivo de entregar o produto. Num sistema não tolerante a faltas o serviço contratado inicialmente pelo consumidor não será entregue e, nem ele, nem o pizzaiolo saberiam informar o que ocorreu com o sistema. Ao adaptar esse sistema ao modelo de TF, propõem-se que o agente entregador seja tolerante a faltas e, com isso, um artefato monitor de saúde é criado para monitorar o estado de saúde do entregador. Além disso, considera-se que, no momento em que o pizzaiolo contrata o entregador, ele passa a monitorar o artefato de saúde atrelado ao entregador, pois ele é diretamente interessado no serviço que este agente contratado está prestando. A Figura 2 ilustra o cenário proposto.

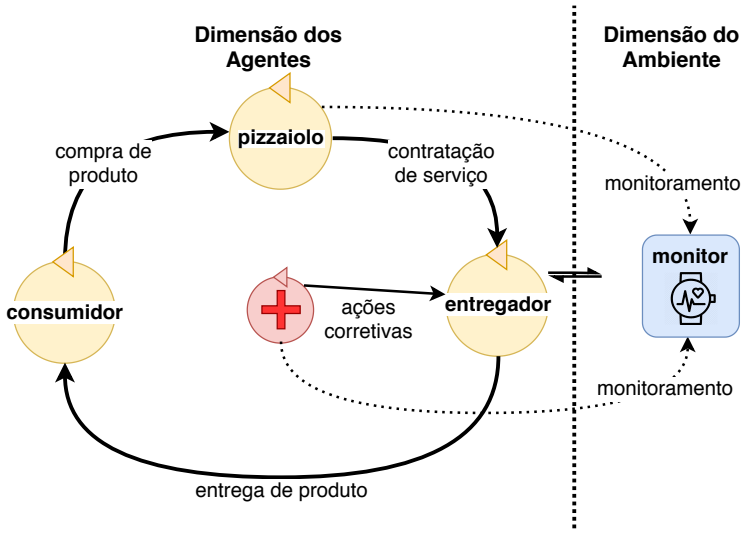

Figura 2. Cenário ilustrativo do SMA

Quando o modelo é implementado, alguns cenários se modificam. O monitor do entregador detecta problemas no batimento cardíaco e emite um aviso para um agente de saúde, que entra em ação e realiza um procedimento de reanimação. Nesse processo, uma cópia do estado mental do entregador é feita, elimina-se o agente em estado de erro e uma replica do agente é inserida no sistema. Caso o estado mental do novo agente não tenha as informações necessárias para executar o plano da entrega (i.e. o agente perdeu a pizza), pode ser executado um plano de contingência, onde o entregador combina de retornar à pizzaria para buscar novamente o pedido.

O agente pizzaiolo, por estar focado no artefato relacionado ao entregador, recebe sinais e mensagens com diversas informações e pode executar planos de contingência para o acidente como, por exemplo: entrar em contato com o cliente e avisar que a entrega do produto vai atrasar; se antecipar, produzindo uma nova pizza e contratando um novo serviço de entrega; aguardar a informação da reinserção do entregador no sistema e

solicitar que ele volte até a pizzaria para buscar o novo produto.

No final de todo processo, o entregador estará apto a finalizar o plano de entrega da pizza e, pela perspectiva do cliente, o serviço foi contratado e entregue corretamente, mesmo com ocorrência de uma falta durante a sua execução. Em muitos casos, o cliente nem ficará sabendo do ocorrido, pois sua preocupação está apenas com o produto final.

4. Conclusões e Trabalhos Futuros

Por se tratar de um trabalho de dissertação em andamento, essa primeira etapa é focada no estudo das abordagens de TF existentes e na elaboração de uma proposta inicial de modelo adaptado à programação multidimencional, que utiliza abstrações da dimensão dos agentes e do ambiente.

Há vários pontos que merecem discussão, um exemplo é como deve ser tratada a questão de informações sigilosas contidas nos artefatos monitores de saúde, nesse ponto é razoável pensar que devem ser criados níveis diferentes de acesso para monitoramento do próprio agente, dos demais agentes do sistema e dos agentes de saúde. Outro ponto de discussão diz respeito a como implementar as ações corretivas de maneira que o modelo fique o mais independe possível da plataforma de programação do SMA.

A sequência do trabalho prevê uma melhor discussão sobre esse e outros pontos do modelo, bem como a sua implementação e integração em uma plataforma de SMA multidimensional e a avaliação do modelo através de testes e análise resultados.

Referências

Boissier, O., Bordini, R. H., Hübner, J. F., Ricci, A., and Santi, A. (2013). Multi-agent oriented programming with JaCaMo. *Science of Computer Programming*, 78:747–761.

Díaz, Á. F. (2018). *eJason : a Framework for Distributed and Fault-tolerant Multi-Agent Systems*. PhD thesis.

Hägg, S. (1997). A sentinel approach to fault handling in multi-agent systems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1286:181–195.

Isong, B. E. and Bekele, E. (2013). A Systematic Review of Fault Tolerance in Mobile Agents. *American Journal of Software Engineering and Applications*, 2(5):111–124.

Klein, M., Rodriguez-Aguilar, J.-A., and Dellarocas, C. (2003). Using Domain-Independent Exception Handling Services to Enable Robust Open Multi-Agent Systems: The Case of Agent Death. *Autonomous Agents and Multi-Agent Systems*, 7(1/2):179–189.

Laprie, J. C. (1992). Dependability: Basic Concepts and Terminology. pages 3–245. Springer, Vienna.

Potiron, K., El Fallah Seghrouchni, A., and Taillibert, P. (2013). *From Fault Classification to Fault Tolerance for Multi-Agent Systems*. Number 9781447150459.

Stanković, R., Štula, M., and Maras, J. (2017). Evaluating fault tolerance approaches in multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 31(1):151–177.