4
Plano de Recuperação

Como pode ser observado na Seção 3.2, um projeto de um middleware para TVD deve considerar o fato que ele será embarcado em plataformas diversas e, portanto, que fará uso de diversas bibliotecas de terceiros para realização de tarefas específicas, tais como decodificação e exibição de diversos tipos de mídias, tratamento de eventos de entrada, entre outras. Essa peculiaridade pode fazer com que a confiabilidade do middleware seja comprometida e seja dependente de possíveis problemas que essas bibliotecas venham apresentar, como vazamentos de memória, comportamentos inesperados ou mesmo falhas críticas que, sem tratamento, podem levar a erros irrecuperáveis.

Outra particularidade habitualmente considerada nos projetos de middleware para TVD, e que também pode comprometer a confiabilidade, é o suporte à sua atualização em tempo de execução, ou evolução dinâmica.

Em todas as estruturas de dados necessárias para o plano de apresentação de aplicações interativas, definidas por Costa et al. (Costa, 2006a) em especial para o middleware Ginga, não houve a preocupação com o suporte a recuperação de falhas. Visando sanar essa lacuna, este capítulo propõe um plano de recuperação para os subsistemas Ginga-CC e Ginga-NCL do middleware Ginga. O plano de recuperação proposto tem por objetivo tornar resiliente não apenas o middleware em si, mas também a apresentação de aplicações interativas que esse middleware executa.

4.1.
Arquitetura

Com o objetivo de definir conjuntos destinados à aplicação de técnicas de recuperação, os módulos da arquitetura Ginga, apresentados na Seção 3.1, foram classificados da seguinte maneira: módulos de Risco, módulos de Apresentação, módulos de Controle e módulos de Recuperação.

Os módulos de Risco são aqueles que podem comprometer a confiabilidade de um middleware de TVD, por geralmente agregarem bibliotecas de terceiros ou serem atualizados no processo de evolução dinâmica do middleware. Os seguintes módulos compõem esse conjunto: Exibidores, Gerente Gráfico, Máquina Lua, Gerente de Contexto, Sintonizador, Transporte, Persistência, Processador de Dados e *Parser* XML.

Os módulos de Apresentação, por sua vez, são aqueles que controlam algum dos módulos de Risco e estão diretamente envolvidos na apresentação de conteúdo. Os módulos Adaptadores e Gerente de Leiaute fazem parte desse conjunto.

Os módulos de Controle são aqueles que controlam ou os módulos de Apresentação ou algum dos módulos de Risco. Os seguintes módulos fazem parte desse conjunto: Gerente de Evolução Dinâmica, Gerente de Bases Privadas, Conversores, Gerente de Exibidores, Gerente de Contexto NCL e Escalonador de Apresentação.

Finalmente, os módulos de Recuperação são aqueles responsáveis pela detecção, controle e recuperação de falhas de software. O módulo Gerente de Recuperação é o único módulo desse conjunto, de fato um super-módulo que, internamente, divide-se em outros módulos.

A Figura 5 ilustra como o conjunto dos módulos de Recuperação atuam sobre os outros conjuntos de módulos para criar o plano de recuperação a falhas. A arquitetura apresentada na Figura 5 é dividida de acordo com as técnicas de recuperação: proativa e reativa.

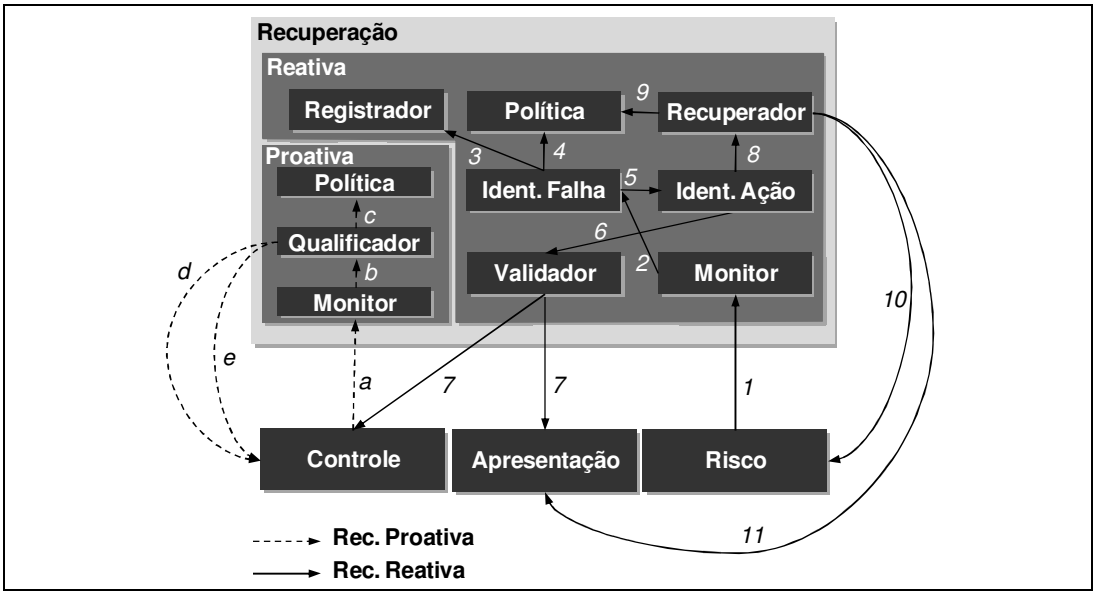

Figura 5 – Arquitetura Plano de Recuperação Ginga-NCL.

Na recuperação proativa, as técnicas de recuperação são aplicadas pelo componente Qualificador por meio de rejuvenescimento dos módulos de Controle. Quando um módulo de Controle é rejuvenescido, os módulos de Apresentação e os módulos de Risco, controlados por esse módulo de Controle, são, indiretamente, rejuvenescidos.

Para saber quando realizar o rejuvenescimento de cada módulo de Controle, sem comprometer a disponibilidade do sistema, foi definido um componente Monitor, que deve ser cadastrado como ouvinte do comportamento dos módulos de Controle. Quando alguma operação que possibilita rejuvenescimento é permitida nesses módulos, o Monitor é notificado (na Figura 5, seta "a"). O Monitor então repassa a informação para que o Qualificador avalie se há alguma ação de recuperação a ser realizada (na Figura 5, seta "b").

Na avaliação realizada pelo Qualificador, pode ser levada em consideração uma política de recuperação proativa especificada para o dispositivo receptor, através de uma consulta ao componente Política (na Figura 5, seta "c"). Por exemplo, um dispositivo receptor capaz de apresentar apenas uma aplicação interativa por vez deve possuir como política a existência de apenas um Formatador instanciado.

Antes de executar a ação de recuperação (na Figura 5, seta "e"), o Qualificador consulta o estado dos módulos de Controle para certificar-se que o rejuvenescimento pode ser realizado (na Figura 5, seta "d").

Na recuperação reativa, um Monitor de falhas foi definido para receber notificações de falhas ocorridas nos módulos de Risco (na Figura 5, seta "1"). Para isso, o Monitor é cadastrado como ouvinte de falhas dos módulos de Risco. Ao receber uma notificação, o Monitor repassa a informação de falha ao Identificador de Falha (na Figura 5, seta "2").

De posse das informações sobre a falha, o Identificador de Falha solicita ao Registrador que a falha seja registrada no sistema (na Figura 5, seta "3"). O Registrador de falhas retorna ao Identificador de Falha quantas vezes a falha ocorreu em um intervalo de tempo. Com essa informação, o Identificador de Falha consulta a Política de recuperação reativa para avaliar se a falha deve ser tratada (na Figura 5, seta "4"). Por exemplo, um dispositivo receptor pode definir como política que uma determinada falha não seja mais tratada após um determinado número de tentativas de recuperação. Caso o Identificador de Falha defina que a falha não seja tratada, nenhuma ação de recuperação será realizada. Caso contrário, o Identificador de Falha repassa as informações sobre a falha para o Identificador de Ação (na Figura 5, seta "5").

Ao receber as informações sobre a falha ocorrida, o Identificador de Ação avalia qual a ação de recuperação a ser realizada. O Identificador de Ação então solicita ao Validador que o estado do módulo a ser recuperado seja validado (na Figura 5, seta "6"). Para isso, o Validador consulta os módulos de Controle e de Apresentação, retornando as informações para o Identificador de Ação (na Figura 5, seta "7"). O estado retornado pelo Validador pode indicar que, apesar da falha ocorrida, o módulo que consiste na origem da falha realizou suas operações corretamente. Nesse caso, nenhuma ação de recuperação será realizada. Caso contrário, após definir a ação de recuperação e adquirir as informações necessárias para realizá-la, o Identificador de Ação aciona o Recuperador (na Figura 5, seta "8").

O Recuperador é responsável por garantir que a ação de recuperação recebida através do Identificador de Ação seja realizada. Antes de executar a ação de recuperação, no entanto, o Recuperador consulta a Política de recuperação reativa para determinar os requisitos da ação (na Figura 5, seta "9"). Um exemplo de requisito é o intervalo de tempo em que a ação de recuperação é válida. De posse de todas as informações necessárias, o Recuperador realiza a ação de recuperação sobre o módulo de Risco origem da falha (na Figura 5, seta "10") e,

em seguida, notifica o módulo de Apresentação, que controla esse módulo de Risco, que uma recuperação foi realizada (na Figura 5, seta “11”).

4.2. Implementação

O suporte ao plano de recuperação foi criado na versão 0.12.1 da implementação de referência do middleware Ginga (veja Seção 3.2). Nessa versão, dois módulos, denominados Gerente de Recuperação Proativa e Gerente de Recuperação Reativa, foram criados.

A implementação do componente Gerente de Recuperação Proativa, apresentado na Figura 6 como *ProactiveRecoveryManager*, embute as funcionalidades necessárias para aplicar as técnicas de recuperação proativa. Para isso, dois tipos de monitores foram implementados no componente.

O primeiro tipo de monitor observa o comportamento do componente Sintonizador, que é apresentado na Figura 6 como *Tuner*. Quando o *ProactiveRecoveryManager* é instanciado, ele cria o monitor, que implementa a interface *ITunerListener*, e o cadastra como ouvinte do componente *Tuner*, através da interface *ITuner*. O componente *Tuner* é adquirido pelo *ProactiveRecoveryManager* através do Gerente de Componentes, apresentado na Figura 6 como *ComponentManager*.

O componente *ComponentManager* (veja Seção 3.2) foi implementado utilizando o padrão de projeto *Singleton*. Sua instância única pode ser adquirida através da interface *IComponentManager*.

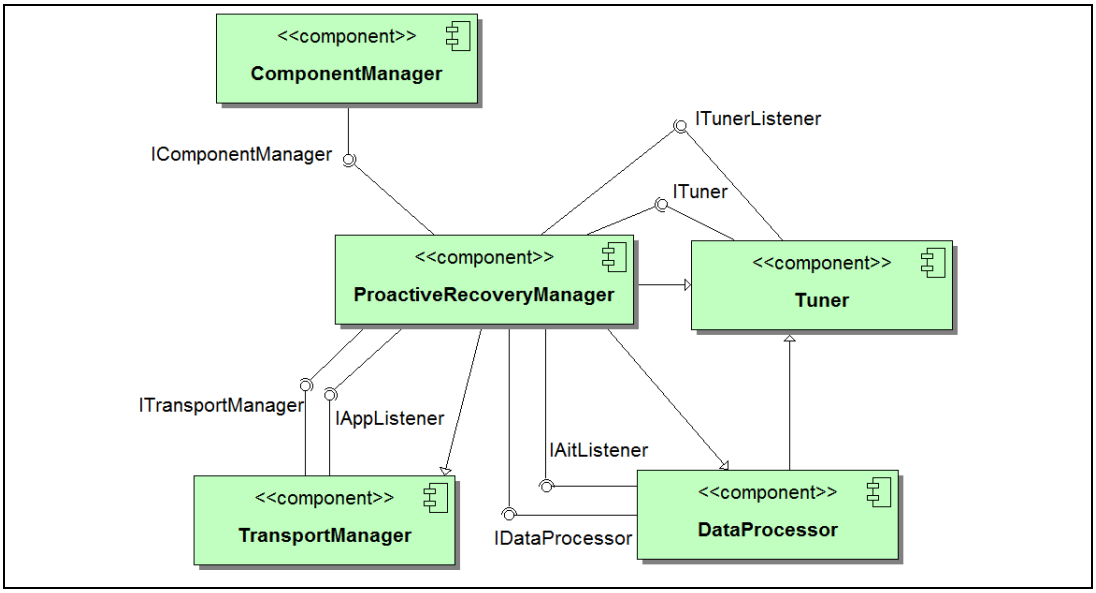

Figura 6 – Diagrama de Componentes para Recuperação Proativa.

A partir desse instante, quando um novo canal for sintonizado, o monitor será notificado, pelo componente *Tuner*, repassando a informação para que o qualificador implementado avalie qual ação a ser tomada. O qualificador faz com que todos os componentes do Núcleo Ginga sejam rejuvenescidos. Para isso, as funcionalidades do *ComponentManager* são novamente utilizadas, através da interface *IComponentManager*.

O segundo tipo de monitor observa as sinalizações sobre recepção de aplicações interativas. Para isso, o monitor foi implementado com uma herança múltipla (implementa a interface *IAITListener* e a interface *IAppListener*), permitindo-o receber notificações do componente Processador de Dados, apresentado na Figura 6 como *DataProcessor*; e do componente Transporte, apresentado na Figura 6 como *TransportManager*.

No método construtor do *ProactiveRecoveryManager* o monitor é criado e cadastrado como ouvinte do componente *DataProcessor*, através da interface *IDataProcessor*, e do componente *TransportManager*, através da interface *ITransportManager*. Os componentes *DataProcessor* e *TransportManager* são adquiridos pelo *ProactiveRecoveryManager* através da interface *IComponentManager* do componente *ComponentManager*.

Como resultado dessas operações, quando uma aplicação for sinalizada para ter sua apresentação iniciada, o qualificador do componente *ProactiveRecoveryManager* é acionado pelo monitor. O qualificador então avalia a notificação e instancia um novo Formatador para apresentar a aplicação.

Quando o usuário troca de canal, a apresentação da aplicação pode até ser finalizada e o Formatador destruído. Como ao trocar de canal o usuário pode retornar ao canal anterior, caso isso aconteça, podem ser entregues ao Formatador rejuvenescido estruturas conhecidas como cadeias temporais e HTG (Costa, 2006a), que oferecem suporte ao início ou retomada síncrona da apresentação. Na versão atual da implementação de referência, entretanto, o suporte ao HTG não está implementado. Assim, ao retornar ao canal anterior, a aplicação é reiniciada no Formatador rejuvenescido e não retomada.

As operações realizadas na troca de canal fazem com que o qualificador diferencie as notificações entre os dois tipos de monitores, de forma que os componentes do Ginga-CC sejam destruídos apenas após todos os Formatadores serem destruídos.

Finalmente, no *ProactiveRecoveryManager*, uma interface foi definida para consulta de políticas de recuperação proativa. A implementação dessa interface, no entanto, foi deixada para trabalhos futuros.

Para a implementação do Gerente de Recuperação Reativa, foi necessária uma refatoração (*refactoring*) dos componentes exibidores do Ginga-CC, fazendo com que as funcionalidades do middleware sejam exercidas por múltiplos processos em execução.

Para permitir maior controle nas notificações de sinais de processo e facilitar o embarque do código em diferentes plataformas, uma abstração de processo foi implementada. Essa abstração consiste no componente *Process*, apresentado na Figura 7. O componente *Process* é utilizado pelos exibidores (componente *Player* na Figura 7) para que a decodificação das mídias seja realizada em processos independentes. Assim, após a refatoração, um *Player* passou a implementar a interface *IProcess*.

Para receber as notificações de processo de um *Player*, o Gerente de Recuperação Reativa, apresentado na Figura 7 como *ReactiveRecoveryManager*, possui um monitor de falhas que implementa a interface *IProcessListener*. Quando um *Player* é criado, um novo monitor deve ser cadastrado como observador desse *Player*.

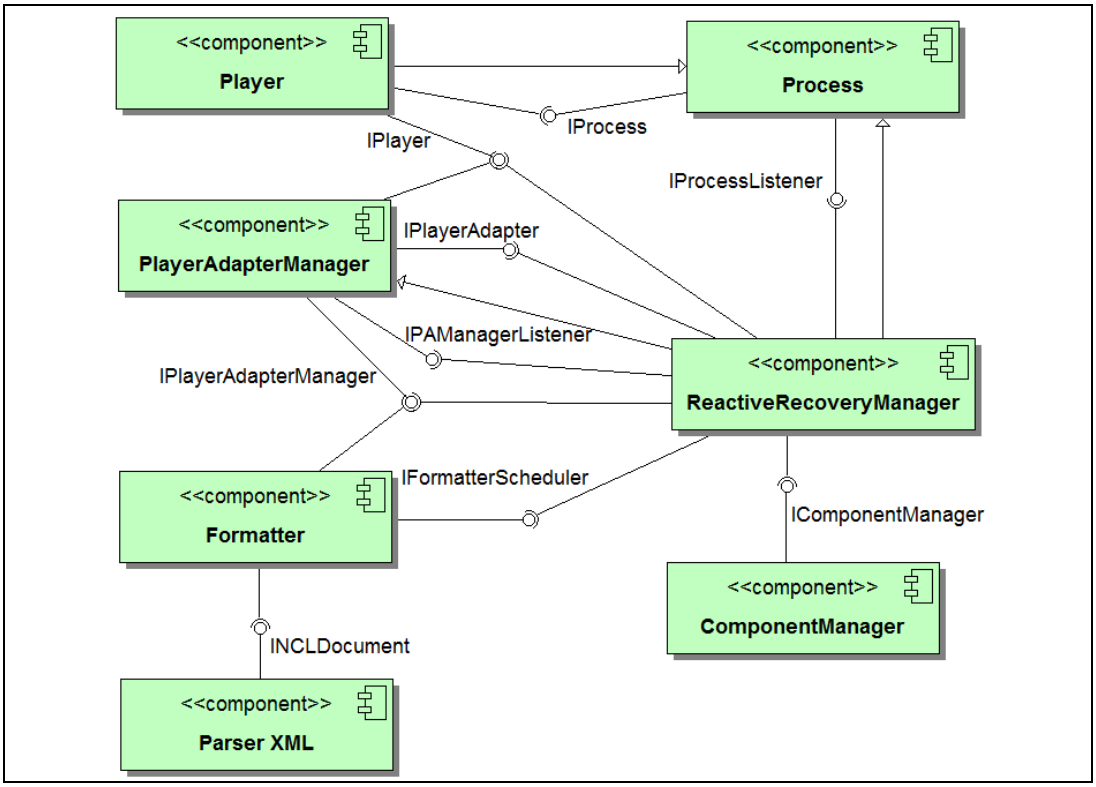

Figura 7 – Diagrama de Componentes para Recuperação Reativa.

Para realizar essa operação, o *RectiveRecoveryManager* possui uma fábrica de monitores que implementa a interface *IPAManager*. Ao ser instanciada, a fábrica cadastra-se como observadora do Gerente de Exibidores (componente *PlayerAdapterManager*) através da interface *IPlayerAdapterManager*. Para ter acesso ao *PlayerAdapterManager*, as funcionalidades do *ComponentManager* são utilizadas.

Quando cada exibidor é criado pelo *PlayerAdapterManager* uma notificação é gerada para a fábrica do *ReactiveRecoveryManager*. Através dessa notificação, a fábrica tem acesso ao novo exibidor. O exibidor implementa a interface *IPlayerAdapter*. Por meio dessa interface, a fábrica tem acesso ao Player, para cadastrar um novo monitor de falhas como observador desse *Player*. Esse cadastro é realizado através da interface *IPlayer*.

Ao receber uma notificação de falha, o monitor repassa essa notificação para o identificador de falha implementado. Esse identificador classifica as falhas conforme os tipos apresentados na coluna “Falha” da Tabela 3.

Falha	Ação
Conteúdo inexistente / corrompido	Ignorar relacionamentos especificados para o objeto (objeto não ocorre)
Fatal decodificação / renderização	Recuperar com o estado especificado
Especificação da Aplicação	Tratar exceção

Tabela 3 – Descritor de evento para comandos de edição NCL.

Assim que uma falha é classificada, ocorre seu registro e a consulta da política de recuperação reativa. O registrador de falha foi implementado para identificar univocamente uma falha e sua origem. A definição da política oferece apenas a opção de quantas falhas pode ocorrer e em qual período de tempo. Caso esses valores não sejam definidos, não é realizada a validação da política.

Após a consulta da política de recuperação, o identificador de falhas passa as informações ao identificador de ação implementado, que define a ação, de acordo com o tipo de falha, como apresentado na coluna "Ação" da Tabela 3.

Os mecanismos de recepção do conteúdo definidos no sistema de TVD devem possuir algoritmos para correção de falhas (Morris, 2005). A falha "conteúdo inexistente / corrompido", apresentada na Tabela 3, pode ocorrer por erros na geração do conteúdo na emissora como, por exemplo, uma aplicação interativa com uma imagem inexistente sendo referenciada ou mesmo uma imagem com conteúdo corrompido. A ação de recuperação definida para essa falha (veja Tabela 3) possui o objetivo de manter a consistência da apresentação (ABNT, 2009; ITU-T, 2009). Nesse caso, não existe a necessidade de validação do estado do objeto que notificou a falha, bem como não existe a necessidade do recuperador implementado consultar a política de recuperação reativa para executar a ação de recuperação.

Para garantir que os relacionamentos do objeto não sejam mais considerados na apresentação, o recuperador implementado utiliza a interface *IFormatterScheduler* do componente *FormatterSheduler*. Para ter acesso a esse componente, a interface *IComponentManager* do *ComponentManager* é utilizada.

Durante a decodificação e renderização de um objeto de mídia, a biblioteca responsável por essas operações pode apresentar comportamentos imprevistos que podem ocasionar a perda do processo. Nesse caso, a falha "fatal decodificação / renderização" é notificada.

Para realizar a ação de recuperação definida para essa falha (veja Tabela 3), é necessária uma consulta ao componente *FormatterScheduller* e *PlayerAdapterManager* sobre o estado de exibição a ser retomado. Considere, por exemplo, uma aplicação que deve exibir um objeto de áudio com duração de 30 segundos. Considere ainda que o exibidor de áudio envie uma notificação de falha "fatal de decodificação / renderização" no instante 15 segundos. Nesse caso, o estado retornado pela consulta ao *FormatterScheduller* e ao *PlayerAdapterManager* é o de ocorrendo em 15 segundos. Para garantir que a ação restaure o processo de forma apropriada, o recuperador cria um novo *Player* e utiliza as informações de estado extraídas para definir o estado em que o novo Player deverá iniciar sua exibição. Após, o recuperador utiliza a interface *IPlayerAdapterManager* para notificar a ação de recuperação, atualizando as referências do Player antigo para o novo *Player*.

Ao iniciar a apresentação de uma aplicação interativa não conforme as normas (ABNT, 2009; ITU-T, 2009), a falha "Especificação da Aplicação" deve ser notificada ao componente *ReactiveRecoveryManager*, para que o tratamento da falha seja realizado. No entanto, essa operação não é feita na versão atual da implementação, pois o interpretador de aplicações, componente *Parser* XML na Figura 7, centraliza todo o tratamento de exceções necessário para esse tipo de falha. Dependendo do comportamento da biblioteca que implementa as funcionalidades de interpretação da aplicação, uma integração do *Parser* XML com o *ReactiveRecoveryManager* pode ser necessária.

O Anexo IV apresenta casos de teste do plano de recuperação, considerando a injeção de falhas para medir o tempo necessário para ações de recuperação.