UML: Diagrama de Seqüência, Diagrama de Atividades, Diagrama de Estado

Diagrama de Atividades

Quando se realiza um caso de uso, deve-se definir os fluxos de eventos que o representam. Geralmente, um caso de uso é composto de um fluxo de eventos principal, também chamado de fluxo ótimo, e dos fluxos alternativos, conhecidos também como fluxos de erro ou fluxos excepcionais. A partir de todos os fluxos de eventos de um caso de uso, pode-se construir o diagrama de atividades que representa o caso de uso.

Para ilustrar a construção de um diagrama de atividades, pode-se considerar o caso de uso "Sacar dinheiro", que integraria o software de um caixa 24h. "Sacar dinheiro" poderia ser composto dos seguintes fluxos de eventos:

- Fluxo principal
    1. o usuário solicita o saque do dinheiro;
    2. o terminal pede que o cartão seja inserido;
    3. o usuário insere o cartão;
    4. o terminal lê o cartão, verifica que o cartão é válido e solicita a senha;
    5. o usuário digita a senha;
    6. o terminal avalia a senha e constata a sua validade. O terminal solicita a digitação da quantia a ser sacada;
    7. o usuário digita a quantia desejada;
    8. o terminal verifica a disponibilidade de saldo, autoriza o saque, libera o valor solicitado e imprime o recibo de saque.
- Fluxos alternativos
    - Cartão inválido
        1. o usuário solicita o saque do dinheiro;
        2. o terminal pede que o cartão seja inserido;
        3. o usuário insere o cartão;
        4. o terminal lê o cartão, determina que este é inválido e retorna ao passo 2.
    - Senha incorreta
        1. o usuário solicita o saque do dinheiro;
        2. o terminal pede que o cartão seja inserido;
        3. o usuário insere o cartão;
        4. o terminal lê o cartão, verifica que o cartão é válido e solicita a senha;
        5. o usuário digita a senha;
        6. o terminal avalia a senha e verifica que ela não é válida. Solicita, então, que o usuário digite a senha novamente. Após a terceira tentativa, o sistema bloqueia o cartão.
    - Saldo insuficiente
        1. o usuário solicita o saque do dinheiro;
        2. o terminal pede que o cartão seja inserido;
        3. o usuário insere o cartão;

4. o terminal lê o cartão, verifica que o cartão é válido e solicita a senha;
5. o usuário digita a senha;
6. o terminal avalia a senha e constata a sua validade. O terminal solicita a digitação da quantia a ser sacada;
7. o usuário digita a quantia desejada;
8. o terminal verifica que não há saldo suficiente e solicita um novo valor de acordo com o saldo existente.

O diagrama de atividades para os fluxos de eventos do caso de uso “Sacar dinheiro” pode ser visto na figura 1.

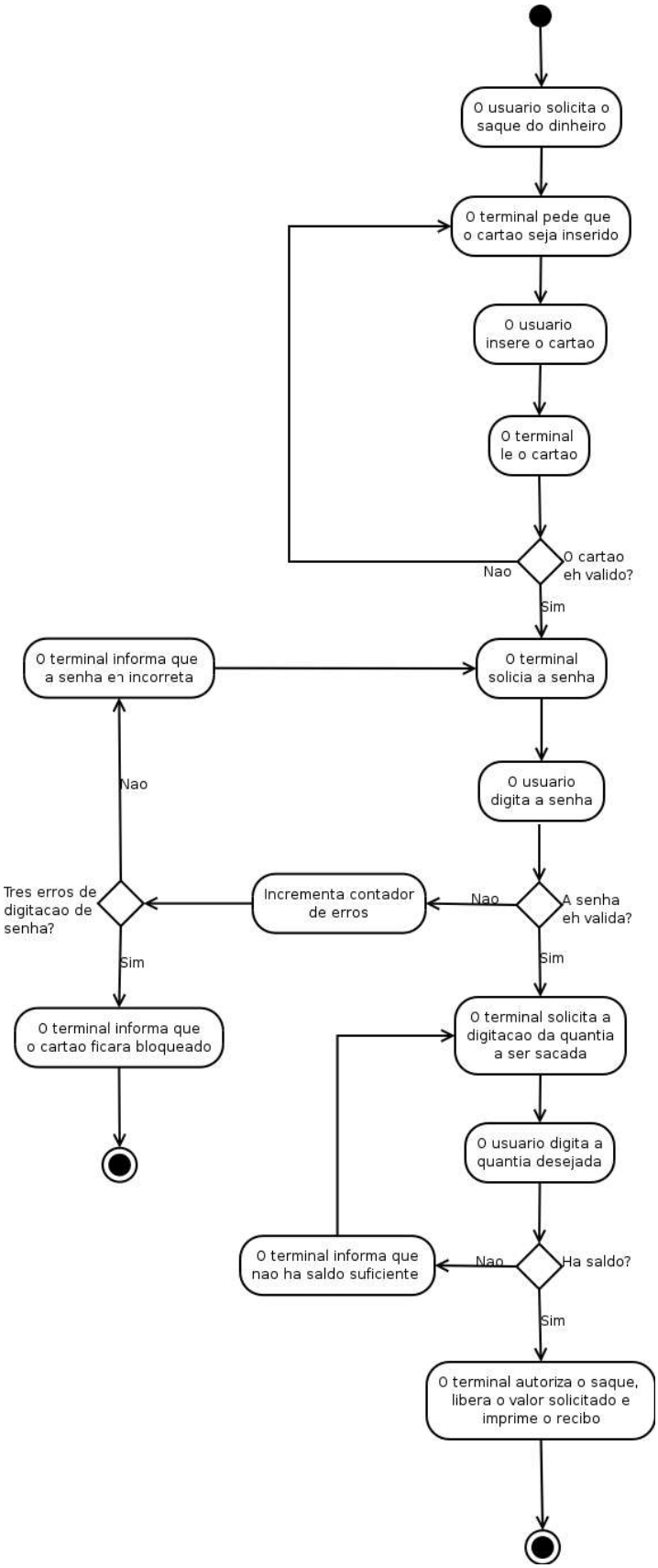

Fig. 1: Diagrama de atividades para o caso de uso "Sacar dinheiro".

Diagrama de Seqüência

O diagrama de seqüência, como o próprio nome diz, apresenta uma seqüência de eventos que determina o comportamento de um caso de uso. Na parte superior do diagrama são apresentados os atores e as classes de análise e, partindo destes, na vertical, são desenhadas as linhas de vida dos objetos.

No diagrama de seqüência mostra-se a interação entre objetos com a preocupação de documentar os métodos executados ao longo do tempo. Um diagrama de seqüência possui duas dimensões: vertical, representando o tempo; e horizontal, representando os diferentes objetos.

De forma geral, para cada caso de uso, constrói-se um diagrama de seqüência principal e alguns diagramas de seqüência complementares. O diagrama principal descreve a seqüência normal de comunicação entre os objetos e os diagramas complementares descrevem as seqüências de tratamento de erros e exceções.

A UML considera dois tipos de mensagens trocadas entre objetos:

1. Mensagens síncronas: o objeto que enviou a mensagem aguarda a conclusão do processamento da mensagem feito pelo objeto de destino, para então prosseguir seu fluxo de execução, ou seja, existe um sincronismo rígido entre os dois objetos. A notação UML para uma mensagem síncrona é a de um segmento de reta com uma seta cheia;
2. Mensagens assíncronas: o objeto de origem envia a mensagem e prossegue seu processamento independentemente do tratamento da mensagem feito no destino. De forma geral, todas as comunicações entre atores e objetos são feitas através de mensagens assíncronas. A notação UML para uma mensagem assíncrona é a de um segmento de reta com uma meia seta.

Para o caso de uso "Cadastrar Aluno", apresentado na figura 2, percebeu-se que eram necessárias duas classes de fronteira ("CIntSecretária" e "CIntSGBD"), uma classe de controle ("CctrlCadastrarAluno") e uma classe de entidade ("CAluno").

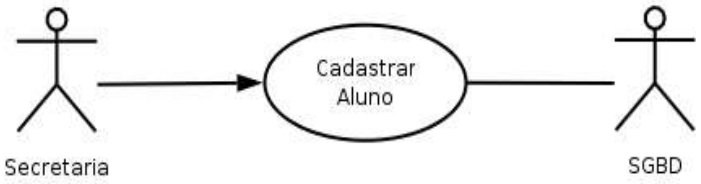

Fig. 2: Caso de uso Cadastrar Aluno.

A figura 3 apresenta as classes necessárias à realização do caso de uso "Cadastrar Aluno".

Fig. 3: Classes necessárias à realização do caso de uso Cadastrar Aluno.

O diagrama de seqüência para o caso de uso “Cadastrar aluno” poderia ser aquele apresentado na figura 4.

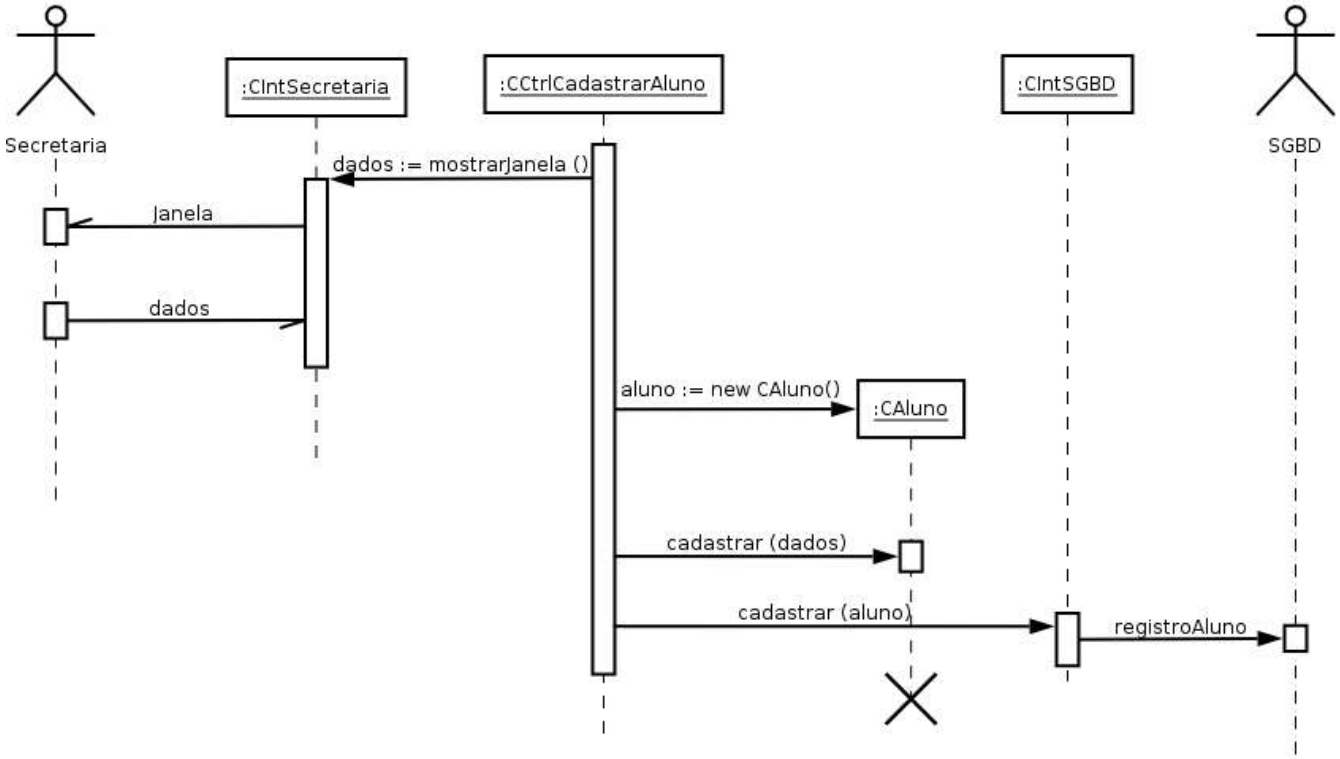

Fig. 4: Diagrama de seqüência relativo ao caso de uso "Cadastrar aluno".

Diagrama de Estados

Os diagramas de estado permitem especificar a dinâmica de um sistema e, para isso, devem reunir o comportamento completo de uma classe em todos os casos de uso, ou seja, deve-se fazer um levantamento de todos os caso de uso onde esta classe participa. Desta forma, o diagrama de estados é uma descrição global do comportamento dos objetos desta classe em todo o sistema.

Os estados são representados graficamente por elipses ou retângulos com bordas arredondadas e as transições são representadas por segmentos de retas dirigidos. Um estado pode ser compreendido como um momento na vida de um objeto, onde este se encontra em uma determinada situação. Já a transição de estados é o avanço que um objeto faz ao passar de um estado para outro.

Os nomes dos estados geralmente são identificados com verbos no gerúndio (p.ex.: “Calculando valor”) ou no particípio (p.ex: “Dados recebidos”), sendo que aqueles estados com nomes no particípio geralmente podem ser eliminados pois são opcionais.

Uma transição de estados pode possuir um rótulo com a sintaxe “Evento (argumentos) [Condição]/

Ação", onde "Evento" indica o nome de um sinal recebido pelo objeto e que habilita a transição, "argumentos" são os valores recebidos junto com o evento, "[Condição]" é uma expressão lógica que deve ser verdadeira para que a transição ocorra e "/Ação" indica um cálculo que é executado durante a transição.

A figura 5 ilustra um diagrama de estados com cinco estados e quatro transições.

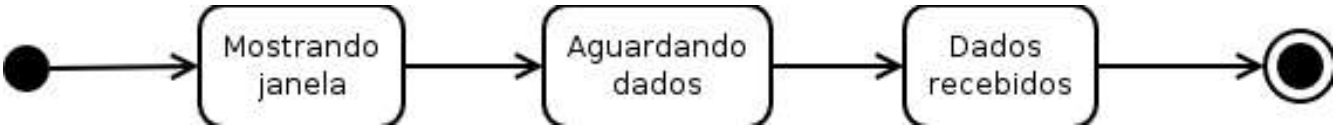

Fig. 5: Exemplo de diagrama de estados.

Um laço é representado por um encadeamento cíclico de estados e transições contendo alguma forma de controle sobre a repetição dos ciclos. Na figura 6, por exemplo, enquanto a variável de controle "x" tiver valor menor ou igual a 10, os estados "Mostrando janela", "Aguardando dados" e "Dados recebidos" suceder-se-ão.

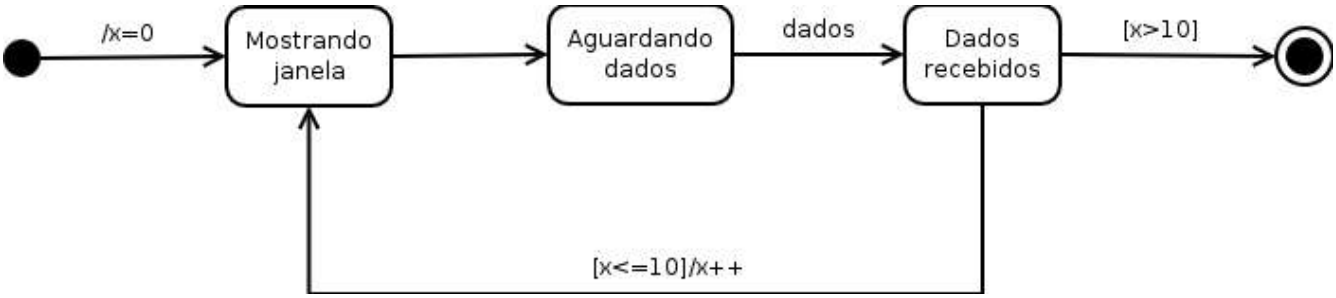

Fig. 6: Exemplo de diagrama de estados com laço (repetição).

Uma cláusula de envio é uma ação de envio de mensagem do objeto que se está modelando para algum outro objeto. As mensagens podem ser síncronas (funções) ou assíncronas (eventos do mouse, por exemplo). A notação para cláusula de envio é "^nome_do_objeto.nome_da_mensagem" ou "^nome_da_classe.nome_da_mensagem", quando o nome do objeto não for conhecido. A figura 7 mostra o diagrama de estados da classe "CCtrlCadastrarAluno" correspondente ao diagrama de seqüência da figura 4. Nota-se, por exemplo, o envio da mensagem "mostrarJanela()" a um objeto da classe "CIntSecretária" onde se aguarda, como resposta, os dados referentes ao aluno que se quer cadastrar (cláusula de envio "/dados := ^CintSecretária.mostrarJanela()").

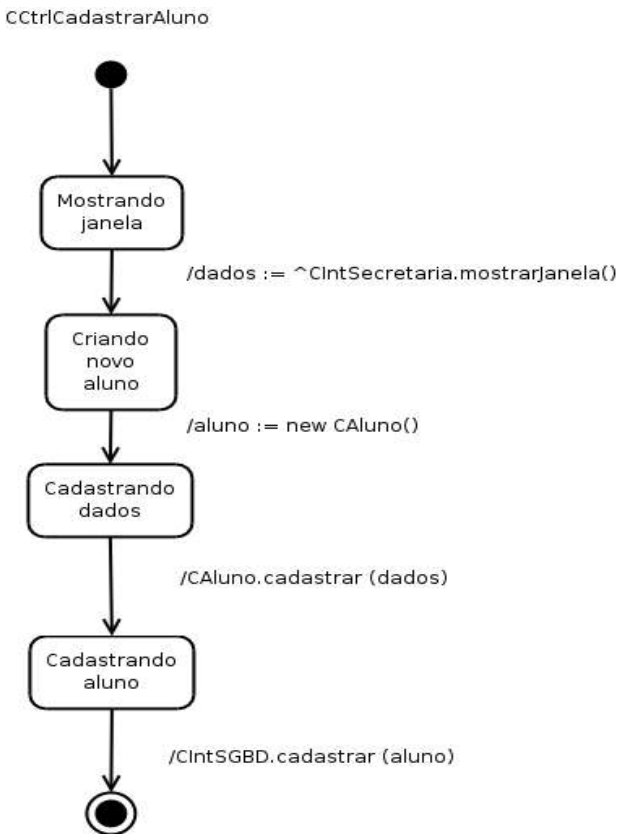

Fig. 7: Exemplo de diagrama de estados contendo cláusulas de envio.

De um modo geral, do ponto de vista dos diagramas de estado, as classes podem ser classificadas em "clientes" e "servidoras". As classes clientes são aquelas que solicitam serviços às classes servidoras e recebem os dados produzidos pelas últimas. Um exemplo de classe cliente é a "CCtrlCadastrarAluno" mostrada na figura 7 (as classes de controle são, em geral, classes clientes). As classes servidoras são aquelas que fornecem serviços às clientes (classes de entidade e de fronteira são servidoras).

Na figura 8, pode-se ver o diagrama de estados para a classe servidora "CIntSecretária".

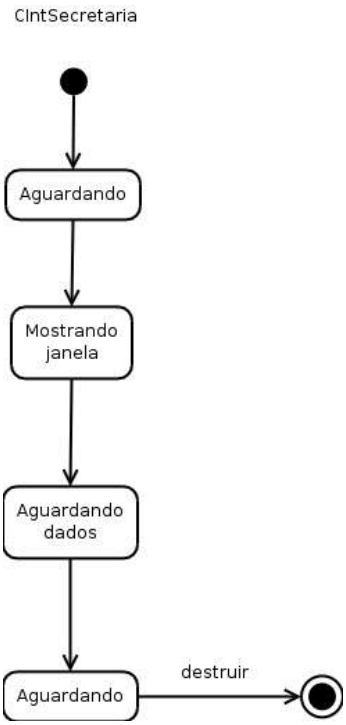

Fig. 8: Diagrama de estados para a classe CIntSecretária.

Bibliografia

TONSIG, Sérgio Luiz. **Engenharia de Software, Análise e Projeto de Sistemas**. Editora Futura, São Paulo, Brasil: 2003.

STADZISZ, Paulo Cézar. **Projeto de Software Utilizando a UML**. UTFPR, Departamento de Informárica: 2002.

CARDOSO, Caíque. **UML na Prática, do Problema ao Sistema**. Editora Ciência Moderna Ltda. Rio de Janeiro, Brasil: 2003.