

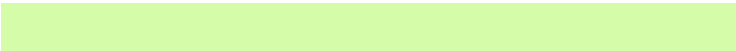


INE 5417

Engenharia de Software I

Prof^a. Patrícia Vilain

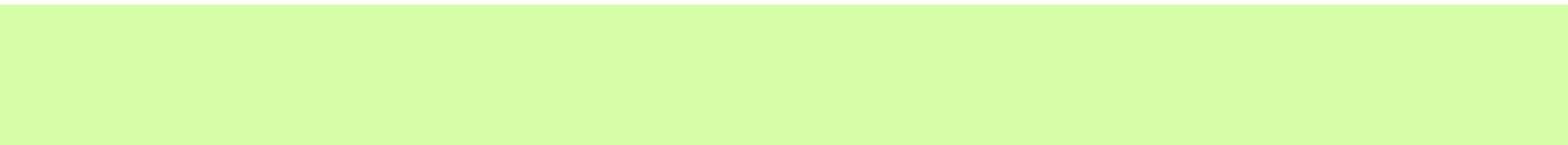

2011.2





Conteúdo

1. Introdução
2. Levantamento de Requisitos
3. Análise Orientada a Objetos
4. Projeto Orientado a Objetos
5. UML
6. Métodos Ágeis




Introdução

O que é Software?

Software \neq Programa de Computador

Software = conjunto de programas
+
arquivos de configuração
+
documentação do sistema
(descreve a estrutura do sistema)
+
documentação do usuário
(explica como usar o sistema)





Engenharia de Software

- A noção de Engenharia de Software surgiu em 1968 em uma conferência onde discutia-se que os softwares estavam ficando bem maiores e mais complexos que os precedentes.
- Os custos de hardware caíam e os custos de software subiam.
- Uma abordagem informal de desenvolvimento de software não era suficiente → Novas técnicas e métodos eram necessários para controlar a complexidade inerente dos grandes softwares.

O que é Engenharia de Software (ES)?


É uma disciplina de engenharia que está relacionada com todos os aspectos da produção de software, desde os estágios iniciais da especificação até a manutenção. [Sommerville]

- 
- processos técnicos de desenvolvimento de sw
 - atividades como gerenciamento de projetos de sw
 - desenvolvimento de ferramentas, métodos e teorias que dêem apoio à produção de sw.



Para que Engenharia de Software?

- Para solucionar o aumento da demanda de software.
- Para solucionar o aumento do custo.
- Para solucionar os problemas do software:
 - estimativas de prazo e de custo imprecisas;
 - qualidade não adequada;
 - manutenção;
 - complexidade (difícil entender um sw grande como um todo).



O que é Processo de Software?

Um processo de software é um conjunto de atividades e resultados que produzem um produto de software.


Processos de software diferentes organizam essas atividades de maneiras diferentes e são descritas em níveis de detalhes diferentes.

Processos de Software

Processos de Software são complexos e dependem do julgamento e criatividade humana



Tentativas de automatização do processo de software não tem muito sucesso



Etapas do Processo de Software

Apesar de existirem diferentes processos para o desenvolvimento de software, no geral, todos os processos apresentam as seguintes atividades:

Especificação: A funcionalidade do software e restrições da sua operação são definidas.

Projeto e Implementação: Converte a especificação do sistema em um sistema executável.

Validação: O software é validado para assegurar que faz o que o usuário deseja.

Evolução: O software deve evoluir para incluir as mudanças resultantes das novas necessidades do cliente.

Especificação do Software

Define quais os serviços que o sistema deve oferecer (requisitos funcionais) e as restrições sobre as operações e o desenvolvimento do sistema (requisitos não funcionais).

➡ Etapa crítica do processo de software.

Os requisitos são detalhados em dois níveis:

- para os usuários e clientes: uma especificação dos requisitos em alto nível;

➡ Levantamento (ou Análise) de Requisitos

- para os desenvolvedores: uma especificação detalhada do sistema.

➡ Análise

Análise x Projeto (Design)

Análise: Especifica o comportamento do sistema sem considerar um ambiente de implementação. Focaliza o **O QUE?**.

Análise OO: investigação dos objetos do domínio

Projeto: Especifica como o sistema deverá ser implementado. Focaliza o **COMO?**.

Projeto OO: Enfatiza a definição dos objetos do software e como eles colaboram, ao invés da sua implementação.

Exemplo

Passos de um Projeto (Project) OO

Jogo de Dados: um jogador joga dois dados. Se o total for 7, ele ganha, senão, perde.

Etapas:

1. Definição dos Casos de Uso
2. Definição de um Modelo de Domínio
3. Definição dos Diagramas de Interação
4. Definição dos Diagramas de Classes de Projeto

Exemplo - Levantamento de Requisitos

1. Definição dos Casos de Uso

Caso de Uso: estória de uso do sistema.

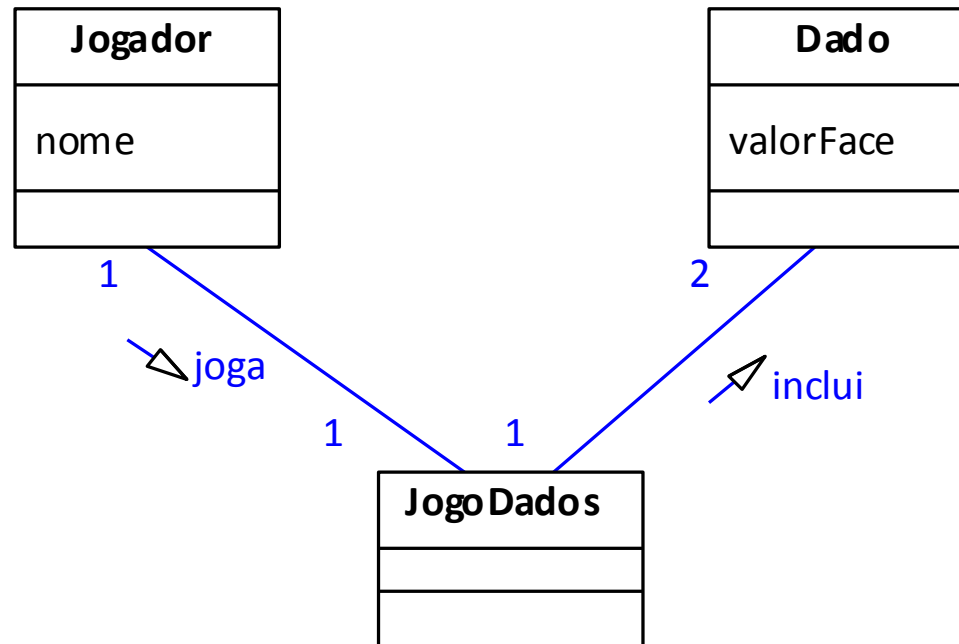
Caso de uso Jogar um jogo de dados:

O jogador pede para jogar os dados. O sistema apresenta o resultado: se a soma das faces dos dados for 7, ele ganha; senão, perde.

Exemplo - Análise

2. Definição de um Modelo de Domínio

Modelo do domínio: descreve os conceitos do mundo real, sob a perspectiva de objetos.

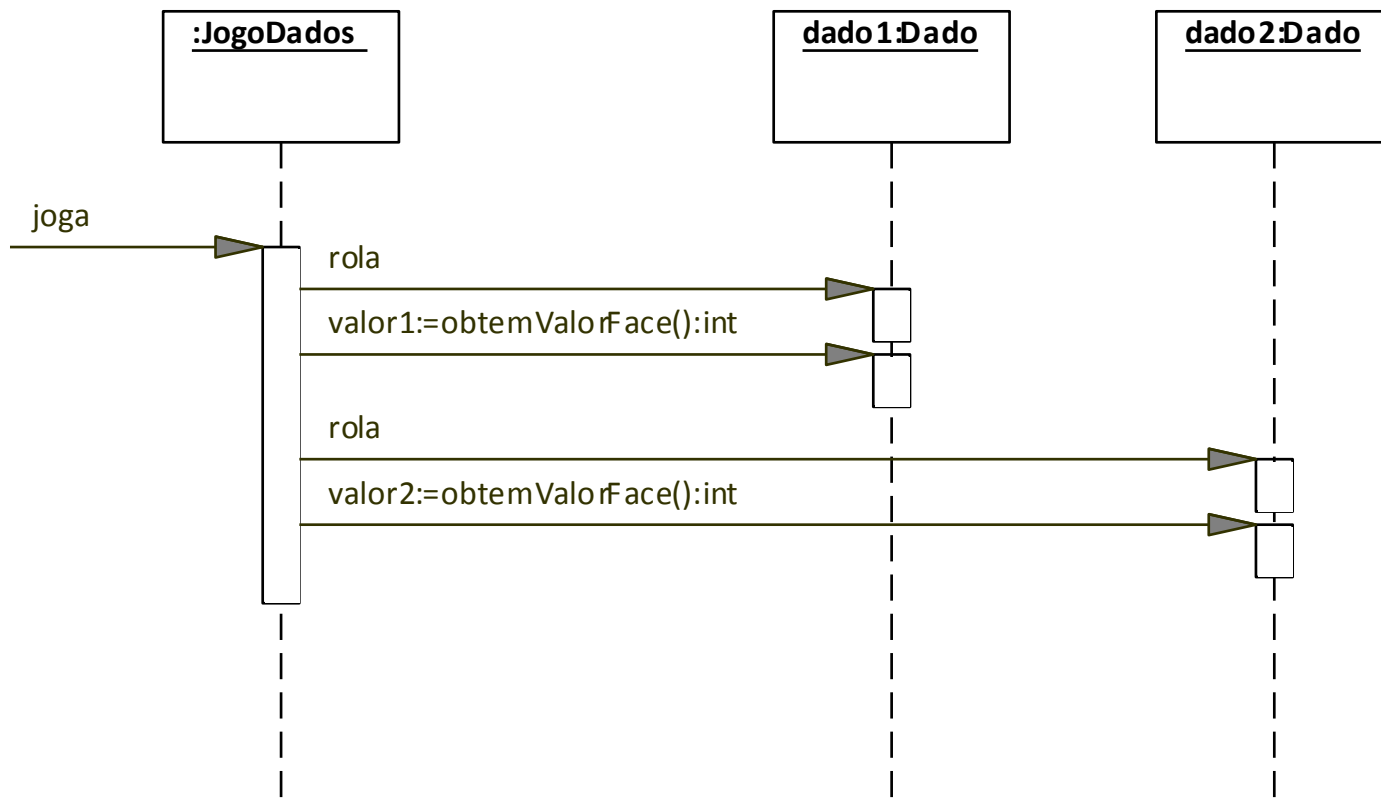


- Identificação dos conceitos, atributos e associações.

Exemplo - Projeto

3. Atribuição das Responsabilidades dos Objetos e Definição dos Diagramas de Interação (Visão dinâmica)

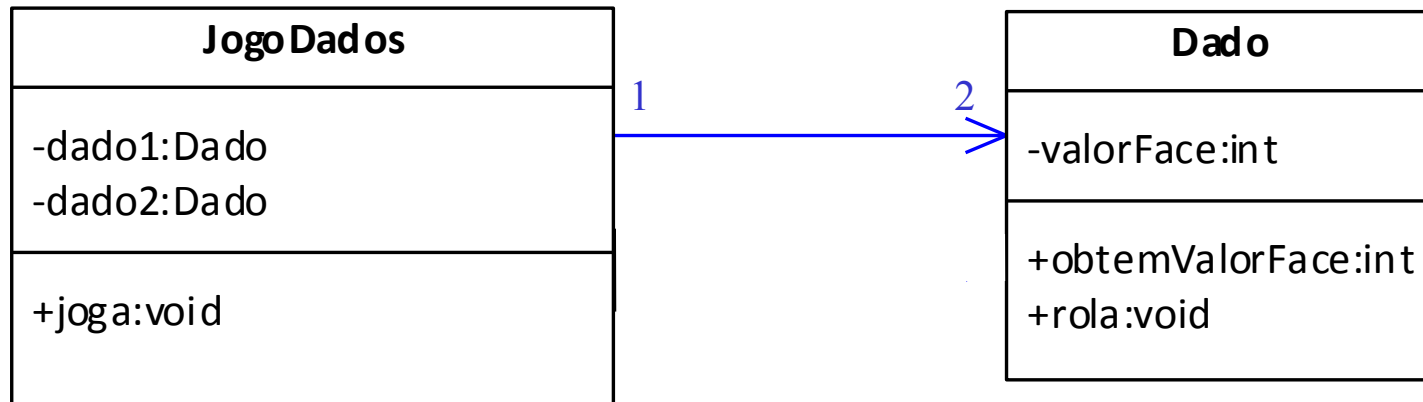
Diagrama de interação: mostra a troca de mens. entre os objetos de sw.




Exemplo - Design

4. Definição dos Diagramas de Classes de Projeto (Visão estática)

Diagramas de classes de projeto: descrição dos objetos de software.







Exemplo

... continuação:

- Implementação e Testes de Unidade
- Testes de Integração e Sistema
- Validação
- Evolução



UML

(Unified Modeling Language)

UML

⇒ A UML é uma linguagem visual para especificação, construção e documentação de artefatos de sistemas.

[OMG - Object Management Group]

⇒ A UML é uma notação diagramática.



Processo Unificado (Unified Process)


Processo Unificado

- ➡ É um processo genérico de software que pode ser especializado
- ➡ Utiliza a UML (Unified Modeling Language) como notação.
- ➡ Características:
 - Centrado na Arquitetura
 - Direcionado por Caso de Uso
 - Iterativo e Incremental

Processo Unificado

➡ Centrado na Arquitetura

- A arquitetura define os aspectos estáticos e dinâmicos mais significantes do sistema.
- A arquitetura é influenciada por:
 - plataforma do sw (e.g. arquitetura do computador, sistema operacional, SGBD, protocolos para comunicação);
 - blocos que serão reusáveis (e.g. framework para interface gráfica);
 - sistemas legados; requisitos não-funcionais, etc.



Processo Unificado

- ➡ Direcionado por Caso de Uso
- Os casos de uso descrevem a funcionalidade completa do sistema.
- Direcionam o design, implementação e teste.

Processo Unificado

➡ Iterativo e Incremental

- O desenvolvimento é dividido em mini-projetos. Cada mini-projeto é uma iteração que resulta em um incremento.

Vantagens do desenvolvimento iterativo:

- Os clientes não precisam esperar o desenvolvimento de todo o sistema (parte com os requisitos críticos já estará pronta).
- Feedback e envolvimento do usuário desde o início leva a um sistema que melhor satisfaz as necessidades dos stakeholders.
- Existe um baixo risco de que todo o projeto (project) falhe.

Fases do Processo Unificado

- Uma série de ciclos são repetidos durante a vida do sistema.
- Cada ciclo consiste de quatro fases:

Fase de Concepção: estudo rápido do sistema proposto a partir de uma visão do produto final. (Continuar ou não o projeto?)

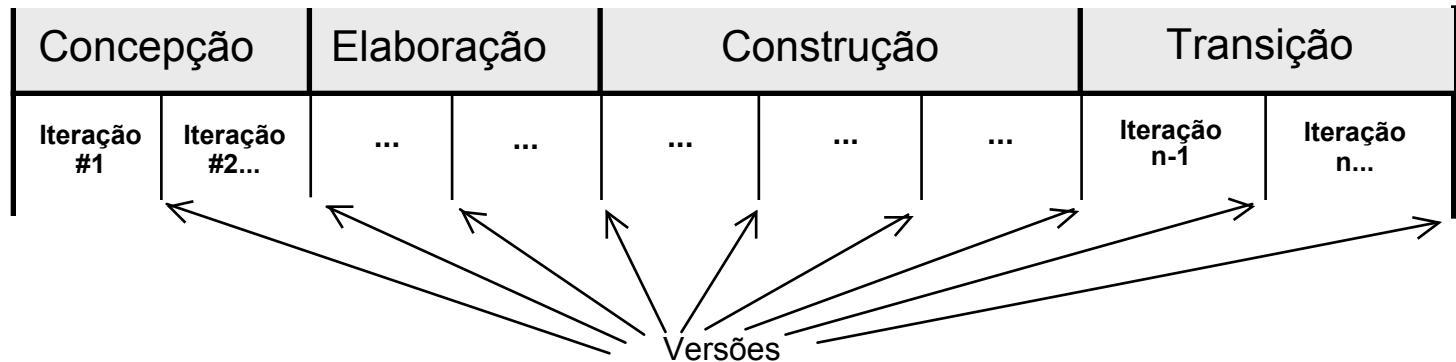
Fase de Elaboração: a maioria dos casos de uso são especificados detalhadamente e a arquitetura do sistema é projetada (modelo de casos de uso, modelo de análise, modelo de design, etc.).

Fase de Construção: o produto é construído e torna-se operacional.

Fase de Transição: o produto entra na fase de teste beta: um número reduzido de usuários experientes utilizam o produto e identificam defeitos e deficiências.

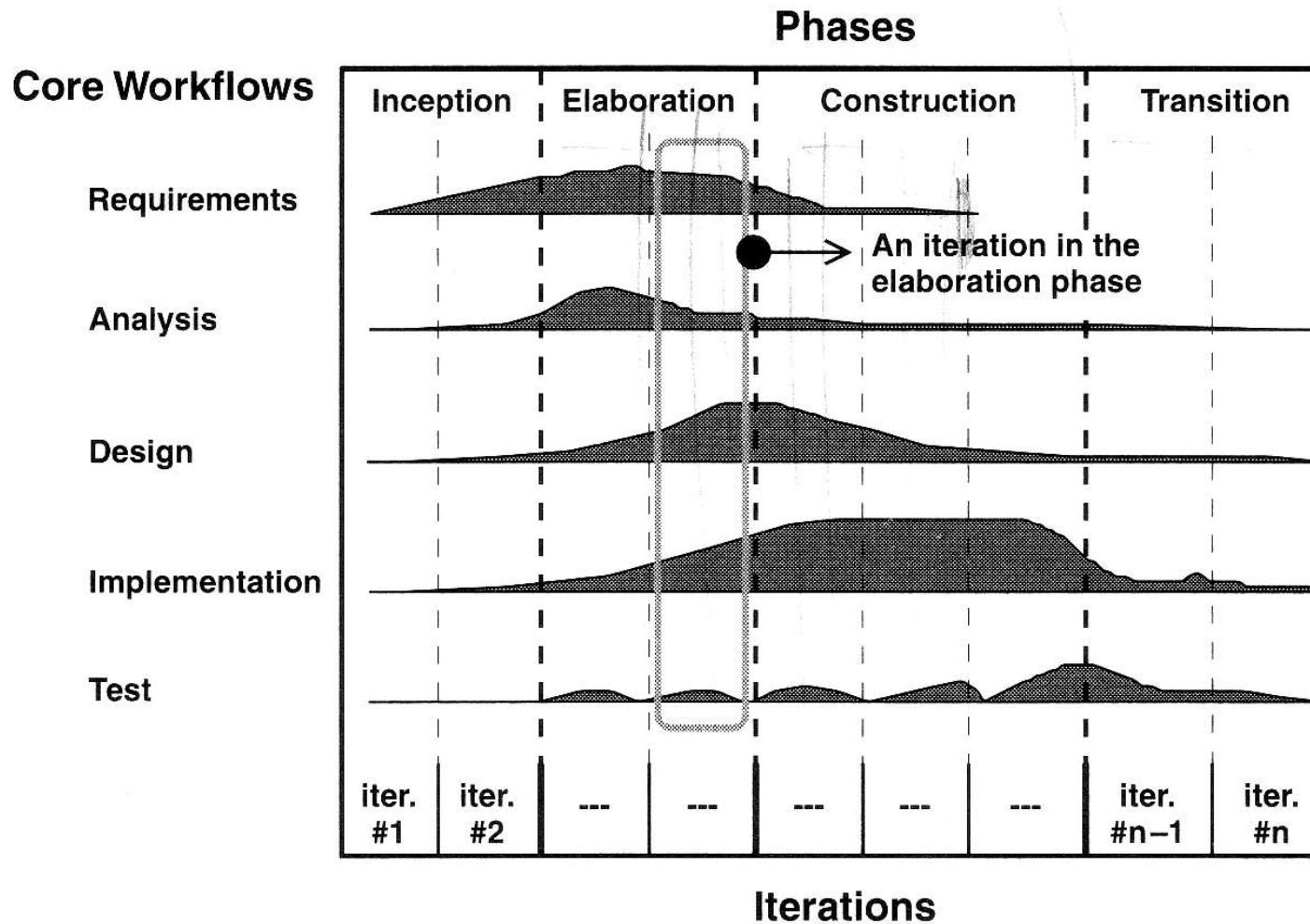
Fases do Processo Unificado

- Cada fase é subdividida em iterações.



- Cada iteração apresenta os workflows de requisitos, análise, design, implementação e testes.

Fases do Processo Unificado





Desenvolvimento Iterativo e Evolucionário

Desenvolvimento Iterativo e Evolucionário


➡ Desenvolvimento Iterativo:

O desenvolvimento é organizado em séries de pequenos mini-projetos de tamanho fixo, chamados de iterações.

Cada iteração inclui as atividades de análise, design, implementação e teste.

➡ Desenvolvimento Evolucionário:

O sistema cresce incrementalmente (evolui), a cada iteração.



Desenvolvimento Iterativo e Evolucionário

- ➡ A saída de uma iteração não é um protótipo experimental ou descartável (throw-away), ou seja, o desenvolvimento iterativo não é prototipação.
- ➡ A saída é um subconjunto do sistema final.



Métodos Ágeis

Métodos Ágeis

- ➡ Aplicam o desenvolvimento iterativo e evolucionário,
 - empregam o planejamento adaptativo,
 - promovem a entrega incremental,
 - e incluem outros valores e práticas que encorajam a agilidade (resposta rápida e flexível às mudanças).

Métodos Ágeis

➡ Valores do Manifesto Ágil

- Indivíduos e interações valem mais que processos e ferramentas.
 - Um software funcionando vale mais que uma documentação extensa.
 - A colaboração do cliente vale mais que a negociação de contrato.
 - Responder a mudanças vale mais que seguir um plano.
- ➔ Apesar dos itens da direita serem importantes, os itens da esquerda são considerados mais importantes que os itens da direita.

Métodos Ágeis

➡ Princípios Ágeis

1. A prioridade é satisfazer ao cliente através de entregas de software contínuas e freqüentes.
2. Receber bem as mudanças de requisitos, mesmo em uma fase avançada, dando aos clientes vantagens competitivas.
3. Entregar software em funcionamento com freqüência de algumas semanas ou meses, sempre na menor escala de tempo.
4. As equipes de negócio e de desenvolvimento devem trabalhar juntas diariamente durante todo o projeto (project).
5. Manter uma equipe motivada fornecendo ambiente, apoio e confiança necessários para a realização do trabalho.
6. A maneira mais eficiente da informação circular dentro da equipe é através de uma conversa face-a-face.

Métodos Ágeis

➡ Princípios Ágeis (continuação)

7. Ter o software funcionando é a melhor medida de progresso.
8. Processos ágeis promovem o desenvolvimento sustentável. Os financiadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante.
9. Atenção contínua a excelência técnica e a um bom projeto aumentam a agilidade.
10. Simplicidade é essencial.
11. As melhores arquiteturas, requisitos e projetos provêm de equipes organizadas.
12. Em intervalos regulares, a equipe deve refletir sobre como se tornar mais eficaz e então se ajustar e adaptar seu comportamento.




Estudos de Caso




Estudo de Caso 1: Sistema Ponto de Venda

- Aplicação computadorizada para uma loja de varejo.
- Objetivo: registrar compras e manipular pagamentos.
- Computador, scanner para código de barra e software para rodar o sistema.
- Interface com outras aplicações (calculadora de impostos, controle do estoque).
- Tolerante a falhas.
- Vários terminais com browser Web no lado do cliente (thin).
- Customizável.



Estudo de Caso 2: Banco Imobiliário

- Simulação do Banco Imobiliário: uma pessoa inicia o jogo e indica o número de jogadores que serão simulados, e então fica observando enquanto o jogo vai sendo jogado automaticamente.



Estudo de Caso 3: Sistema de Revisão

- Sistema de Revisão de Conferência: suporte ao processo de submissão, avaliação e seleção de artigos para uma conferência qualquer.
- Funções do Sistema:
 - Submissão de um artigo;
 - Atribuição de artigos aos avaliadores;
 - Entrada de uma revisão;
 - Escolha dos artigos aceitos e rejeitados.