1ªReunião do Grupo de Software 26/05/2010

Resumo do projeto: O grupo foi contratado pelo cliente para a manutenção, refatoração e adição de novas funcionalidades no produto pré-pronto deles, um jogo chamado Asgard Defender.
O grupo pretende adotar as práticas de Extreme Programming para que possa dar transparência no andar do processo, enviando versões rapidamente com pequenos incrementos na espera de rápido feedback do cliente.
Cliente: GUILDJ

A reunião será dividida em duas partes, uma deles só com a equipe para a discussão de padrões que serão adotados para facilitar a compreensão de todos ao decorrer do projeto, e a outra parte com a presença do cliente para o processo de elicitação de necessidades para o sistema.

Tempo estimado para cada fase da reunião:
* Decisões de padronização de código e boas práticas: 15min
* Elicitação de novas necessidades: 20min

Decisões que serão adotadas durante o decorrer do projeto:

*As reuniões serão feitas em pé, seguindo o padrão XP de reuniões curtas e objetivas;
*Uso de SVN, Subversions e GoogleCode para a sincronia de projetos, feita de maneira online no espaço da GUILDJ;
*Instalar o plug-in SVN no Eclipse ou Net-Beans;
*Criar conta Google para obter acesso ao GoogleCode ;
Como será feita a postagem de novas versões com novas funcionalidades?

Cada dupla será responsável pela solução de um problema apresentado pelo cliente, ao final da resolução uma nova versão deve ser postada no GoogleCode no espaço do GUILDJ com o nome padronizado, que será explicado mais a frente . O código deve vir documentado sobre problemas enfrentados e como foi solucionado, dado que o cliente deseja fornecer o jogo como material de estudos.

Padronização de código:

*Abrir e fechar de blocos : todo método, construtor ou sequência encadeada em blocos deve abrir as chaves na frente, e não na quebra de linha, e deve ter a distancia de UM espaço. E seu fechamento deve ser alinhado com sua origem.

Exemplo Correto:

```
umMetodo() {

}
```

Exemplo Incorreto:

```
umMetodo()          umMetodo(){
{
}                   }
```

*Os métodos e as variáveis devem ter nomes detalhados sobre a sua funcionalidade e devem ser precedidos de comentários que o expliquem, e no caso de lógicas complexas, explicar como é feito.

Exemplo Correto:

```
/*Comentários sobre o método, e sua funcionalidade.
 *Numa determinada linha, tais operações são realizadas
 *para que o problema fosse resolvido
*/
umMetodo() {
}
```

Exemplo Incorreto:

```
umMetodo{    <- Metodo sem comentário
}
```

*As variáveis devem ser declaradas no topo da classe, antes de todo e qualquer método ou construtor, e as variáveis devem ser declaradas uma por linha, e não agrupadas numa única linha e separadas por vírgulas.

Exemplo Correto:

```
public UmaClasse {
        int umaVariavel;
        int outraVariavel;
}
```

Exemplo Incorreto:

```
public UmaClasse {
        int umaVariavel;
        public umMetodo() { . . .}
        int outraVariavel;
}
```

*Passada e recepção de argumentos separados por espaço.

Exemplo Correto:

```
public umMetodo(int raio, int x, int y) {. . .}        umMetodo(raio, x, y);
```

Exemplo Incorreto:

```
public umMetodo(int raio,int x,int y) {. . .}        umMetodo(raio,x,y);
```

*Blocos de repetição ou decisão de uma única linha devem abrir e fechar chaves.

Exemplo Correto:

```
for(i = 0; i < 10; i++) {
        System.out.print("\n");
}
```

```
if("Brasil".equals("Brazil")) {
        System.out.print("Hexa!!!");
}
```

Exemplo Incorreto:

```
for(i = 0; i < 10; i++)
        System.out.print("\n");
```

```
if("Brasil".equals("Brazil"))
        System.out.print("Hexa!!!");
```

*Padronização dos nomes de entrada referente a arquivos externos, como imagens, sons, textos, etc, para que o arquivo possa ser substituído sem que o código precise ser alterado.

Exemplo Correto:

```
thorImageName = "ThorIcon.jpg" ;
 explosionSoundName = "explosion.mid";
```

Exemplo Incorreto:

```
thorImageName = "jff01.jpg"
explosionSoudName = "mid01.mid";
```

*Ao completar de uma semana um arquivo executável (.jar) deve ser entregue ao cliente para avaliação, seguindo o padrão de sub-versão: AsgardDefender**SXVXX.** Onde o SX significa de qual semana é, e o VXX quantas versões foram criadas.

Reunião com o Cliente

Problema: Ajuste de Colisão
Detalhamento: Os tiros estão atravessando a animação dos invasores em áreas que eles deveriam ser atingidos.
Responsáveis: Adriano e Marcus
Nível de Prioridade: Médio (2)

Problema: Melhorar a mesclagem da imagem com o background.
Detalhamento: As imagens apresentam um suave contorno branco que causa uma má impressão de sobreposição com o fundo.
Responsáveis: Vinicius e Regis
Nível de Prioridade: Baixo (1)

Problema: Troca de GIF para SpriteSheet
Detalhamento: Atualmente a animação é feita por um GIF que fica sendo transmitido, sem controle de como a animação irá se comportar, a troca deve ser feita para animação programada através de SpriteSheet.
Responsáveis: Vinicius e Regis
Nível de Prioridade: Baixo (1)

Problema: Adicionar sons aos eventos
Detalhamento: Atualmente o jogo conta somente com uma musica de fundo, devem ser adicionados sons de disparo, colisão entre outros eventos do jogo.
Responsáveis: Vinicius e Regis
Nível de Prioridade: Baixo (1)

Problema: Adicionar tiro nas quatro direções
Detalhamento: Atualmente o jogo conta somente com disparos para cima. Deve ser adicionados tiros que possam assumir as quatro direções de movimento, em função da direção do personagem.
Responsável: Regis
Nível de Prioridade: Alto (3)

Problema: Mover por KEY-PRESSED
Detalhamento: Atualmente o personagem se move constantemente para a ultima direção aplicada pela tecla, mudar para que ele se mova somente enquanto o botão é pressionado.
Responsável: Regis
Nível de Prioridade: Alto (3)

Problema: Adicionar funcionalidade aos botões da tela inicial que ainda não funcionam
Detalhamento: O menu inicial possui dois botões que ainda não estão funcionando, Ranking e Créditos.
Responsáveis: Adriano e Marcus
Nível de Prioridade: Alto (3)

Problema: Gerenciamento de painéis
Detalhamento: Adicionar transição de painéis através de eventos gerados por botões ou fim de jogo.
Responsáveis: Adriano e Marcus
Nível de Prioridade: Alto (3)

Problema: Criar poder especial do Thor
Detalhamento: Adicionar a habilidade especial para o personagem, o qual será um pulso de energia partindo do Thor, afetando uma área e dando mais dano que um tiro normal.
Responsáveis: Carlos e Regis
Nível de Prioridade: Baixo (1)

Problema: Remover Invasores Zig-Zag que travam nos cantos
Detalhamento: Raramente alguns invasores ficam presos nos cantos da tela, o que é um erro e deve ser removido.
Responsáveis: Carlos e Regis
Nível de Prioridade: Alto (3)

Problema: Adicionar tiro ao boss
Detalhamento: Foi pedido que o boss possa atirar periodicamente.
Responsáveis: Carlos e Regis
Nível de Prioridade: Baixo (1)