

WILLE MARCEL LIMA MALHEIRO

SOFTWARE LIVRE, BAZAR E COMPUTAÇÃO GRÁFICA:  
O CASO DO BLENDER

São Cristóvão  
2008

WILLE MARCEL LIMA MALHEIRO

SOFTWARE LIVRE, BAZAR E COMPUTAÇÃO GRÁFICA:  
O CASO DO BLENDER

Monografia apresentada ao curso de Comunicação Social – habilitação em Radialismo da Universidade Federal de Sergipe, como requisito parcial para obtenção do título de bacharel.

Orientador: Prof. Msc. Jean Fábio Borba Cerqueira.

São Cristóvão

2008

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL  
UNIVERSIDADE FEDERAL DE SERGIPE

Malheiro, Wille Marcel Lima

M249s      Software livre, bazar e computação gráfica : o caso do Blender /  
Wille Marcel Lima Malheiro. – São Cristóvão, SE, 2008.  
85f. : il.

Monografia (Bacharelado em Comunicação Social – Habilitação  
em Radialismo) – Departamento de Artes e Comunicação Social,  
Centro de Educação e Ciências Humanas, Universidade Federal  
de Sergipe, São Cristóvão, SE, 2008.

Orientador: Prof. Msc. Jean Fábio Borba Cerqueira.

1. Comunicação social. 2. Comunicação de massa. 3. Software  
livre. 4. Computação gráfica. 5. Animação digital. I. Título.

CDU 659.3::654.19::004.4

WILLE MARCEL LIMA MALHEIRO

SOFTWARE LIVRE, BAZAR E COMPUTAÇÃO GRÁFICA:  
O CASO DO BLENDER

Monografia apresentada ao curso de Comunicação Social – habilitação em Radialismo da Universidade Federal de Sergipe, como requisito parcial para obtenção do título de bacharel.

Aprovada em 04 de abril de 2008.

**BANCA EXAMINADORA**

**Prof. Msc. Jean Fábio Borba Cerqueira**

Departamento de Artes e Comunicação / UFS

**Prof. Dr. Leonardo Nogueira Matos**

Departamento de Computação / UFS

**Profa. Dra. Lilian Cristina Monteiro França**

Departamento de Artes e Comunicação / UFS

## RESUMO

Esta monografia discute a evolução técnica do Blender, um software livre voltado para a modelagem e animação tridimensional e para a criação de *game engines*. Desta forma, são objeto de análise os principais fatores que acreditamos ter exercido influência sobre o processo de desenvolvimento deste software. Além disso, são examinados os recursos oferecidos pelo Blender, de forma a avaliar se estes atendem as necessidades atuais de um filme e/ou vídeo de animação digital.

**Palavras-chave:** software livre, animação, computação gráfica.

# Sumário

<b>Introdução.....</b>	<b>9</b>
<b>1. Software Livre: liberdade do conhecimento e colaboração.....</b>	<b>11</b>
1.1 O contexto social e tecnológico do surgimento do Movimento Software Livre	11
1.1.1 As transformações no hardware.....	11
1.1.2 O surgimento das redes de computadores e do sistema Unix.....	13
1.1.3 A cultura hacker e a cultura tecnomeritocrática.....	15
1.2 Software livre: a ideologia por detrás do código.....	18
1.2.1 Do GNU ao GNU/Linux.....	18
1.2.2 As licenças livres.....	20
1.2.3 As quatro liberdades.....	21
1.2.4 As motivações do software livre.....	21
1.3 Ideologia e código fonte.....	22
1.4 O método de desenvolvimento do software livre.....	26
1.4.1 Os modelos Catedral e Bazar: inovações trazidas por Linus Torvalds.....	27
1.4.2 O ciclo de vida de um projeto de software livre.....	30
1.4.3 Estratégias de financiamento de software livre.....	30
1.5 Penetração e diversidade do software livre.....	32
<b>2. Ambiente digital de produção de animação.....</b>	<b>35</b>
2.1 Um breve resgate histórico da animação.....	35
2.2 Caracterização e conceituação da animação.....	42
2.3 Animação digital.....	43
2.3.1 A evolução da computação gráfica.....	44
2.3.2 O processo de animação digital.....	45
2.3.3 O ambiente de animação tridimensional.....	47
2.3.3.1 Modelagem.....	47
2.3.3.2 Iluminação.....	51

2.3.3.3 Textura.....	52
2.3.3.4 Animação.....	53
2.3.3.5 Renderização.....	55
2.3.4 Os principais sistemas comerciais de animação 3D.....	56
<b>3. Blender: descrição de recursos e modelo de desenvolvimento.....</b>	<b>59</b>
3.1 Do surgimento ao código livre.....	59
3.2 Os recursos do Blender.....	60
3.2.1 Ferramentas de modelagem.....	60
3.2.2 Iluminação e renderização.....	63
3.2.3 Animação.....	65
3.3 Análise dos recursos presentes no vídeo “Elephants Dream” .....	66
3.4 A avaliação dos usuários do Blender.....	70
3.5 A comunidade do Blender.....	72
3.5.1 As estratégias de financiamento adotadas pela comunidade do Blender	74
3.5.2 Contribuições dos “open projects” .....	75
3.5.2 As estratégias adotadas pela comunidade do Blender em relação às de outros softwares.....	76
3.5.3 A participação dos usuários do Blender na evolução do software.....	78
<b>Considerações Finais.....</b>	<b>80</b>
<b>Bibliografia Utilizada.....</b>	<b>82</b>
<b>Apêndices.....</b>	<b>85</b>

## Lista de Figuras

Fig. 2.1 – Modelo de lanterna mágica fabricado em 1900 na França.....	36
Fig. 2.2 – As duas faces de um taumatoscópio, um estroboscópio e um zootroscópio.....	37
Fig. 2.3 – Mickey, personagem dos estúdios Disney.....	40
Fig. 2.4 – Patolino, personagem da Warner Brothers.....	41
Fig. 2.5 – O Diabo da Tasmânia e Pernalonga, personagens da Warner Brothers....	41
Fig. 2.6 – Personagem sendo animado através de controle por esqueleto e keyframes.....	46
Fig. 2.7 – Pingüim sendo modelado a partir de uma esfera.....	48
Fig. 2.8 – Operação booleana de união entre um cubo e uma esfera.....	49
Fig. 2.9 Extrusão de uma forma ao longo de um eixo.....	49
Fig. 2.10 – Objeto produzido pela técnica de revolução.....	50
Fig. 2.11 – Modelo produzido por seção transversal serial.....	50
Fig. 3.1 – Exemplo de curva Bezier.....	61
Fig. 3.2 – Exemplo de curva do tipo NURBS.....	62
Fig. 3.3 – Sistema de partículas e material Halo.....	62
Fig. 3.4 – Demonstração da ferramenta Ambient Occlusion.....	64
Fig. 3.5 – Simulação de vento produzida com o uso da ferramenta Force Fields.....	66
Fig. 3.6 Personagem visto através do reflexo na água.....	67
Fig. 3.7 Uso do sistema de partículas e efeitos atmosféricos.....	68
Fig. 3.8 Utilização da iluminação global e do recurso Lamp.....	69
Fig. 3.9 Uso do sistema de simulação de física real e da cinemática inversa.....	69

## Introdução

Os filmes de animação tridimensional, desde alguns anos, já representam uma parcela importante da produção audiovisual. O fascínio que a animação exerce sobre o público é confirmado pelas estatísticas, que revelam uma presença cada vez mais significativa de animações nas telas de TV's e cinemas de todo o mundo. No ano de 2006, por exemplo, entre os dez filmes mais vistos nos cinemas estadunidenses, três são animações 3D<sup>1</sup>.

A produção de animação tridimensional, bem como grande parcela da produção audiovisual, utiliza atualmente uma série de tecnologias digitais, as quais são capazes de prover um maior realismo e de agilizar o processo produtivo. Entre estas tecnologias, incluem-se os softwares de computação gráfica, dos quais depende toda a produção digital de animação.

Sendo assim, nos últimos anos, um software vem se destacando no cenário de modelagem e animação 3D. O nome dele é Blender e, além de chamar atenção pelos recursos que oferece e pelas produções que têm sido realizadas com o uso dele, este tem despertado a curiosidade de muitos pelo fato de ser um software livre, o qual é distribuído gratuitamente e oferece uma série de liberdades. O trabalho de pesquisa que será apresentado nas próximas páginas procura analisar quais os principais recursos oferecidos pelo Blender, de acordo com as atuais necessidades envolvidas na produção de um filme de animação digital. Além disso, serão investigados os fatores que fizeram o Blender alcançar o atual nível de qualidade técnica.

Para iniciar, é importante destacar que o termo software livre começou a ser difundido a partir da década de 1980. Ao contrário do que à primeira vista este termo deixa transparecer, a liberdade não se refere apenas à gratuidade do software, mas sim à liberdade de acessar e fazer modificações no código fonte dos mesmos.

A preocupação daqueles que iniciaram a difusão do conceito de software livre está ligada à importância que os computadores vinham assumindo dentro da sociedade e ao risco ao qual estamos sujeitos quando não dispomos das informações de como os softwares funcionam. Além das liberdades oferecidas pelo software livre, o método de desenvolvimento destes softwares, denominado por Eric Raymond de “bazar”, mostrou-se bastante eficiente. Softwares produzidos de forma

1 Dados da página: <http://www.boxofficemojo.com/yearly/chart/?yr=2006&p=.htm>

aberta e colaborativa têm se mostrado como alternativas de alta qualidade técnica e conquistado usuários e colaboradores por todo o mundo.

Após a consolidação do sistema operacional GNU/Linux e de outros softwares livres em algumas áreas da informática, como a de servidores de internet, estes também passam a se estabelecer como opção de qualidade técnica em outras áreas, como a de aplicativos de escritório e navegadores de internet.

Nesse sentido, o Blender pode ser considerado um marco do estabelecimento do software livre na área de computação gráfica, em especial na parte de modelagem e animação 3D. Curiosamente, o Blender inicialmente era um software proprietário, porém com o falência da empresa que o desenvolvia, este foi disponibilizado como software livre no final do ano de 2002. Desde então, uma grande comunidade se formou em torno do Blender, dando continuidade ao desenvolvimento e promovendo a utilização deste.

Neste contexto, a análise da evolução técnica do Blender e dos fatores que influenciaram esse processo pode representar uma importante contribuição para o entendimento do potencial do método “bazar” de desenvolvimento de software livre. Além disso, pode representar uma contribuição também para a organização de outras comunidades de desenvolvimento de software livre.

Sendo assim, optamos por dividir este trabalho de pesquisa em três partes, além desta introdução. No capítulo um, serão abordadas as questões referentes ao software livre, desde as origens deste fenômeno até as mudanças que este provocou na forma de se desenvolver softwares. Por sua vez, o segundo capítulo vai tratar dos aspectos envolvidos na produção de animação por computador, com o enfoque na animação tridimensional, para a qual o Blender é voltado. Ambos os capítulos foram elaborados a partir de pesquisa bibliográfica acerca das considerações teóricas envolvidas no estudo.

Por fim, o capítulo três procura descrever os recursos técnicos oferecidos pelo Blender e os principais fatores que exerceram influência sobre o processo de desenvolvimento deste software. Para alcançar tais objetivos, utilizamos de pesquisa bibliográfica, consulta à documentação do software e das ações realizadas pela sua comunidade, e também realizamos uma pesquisa de campo, a qual nos mostrou a avaliação que os usuários fazem do software e o índice de colaboração destes com a comunidade.

## **1. Software Livre: liberdade do conhecimento e colaboração**

O software livre nunca recebeu tanta atenção como no momento atual. O modelo de desenvolvimento adotado pelos projetos de software livre tem demonstrado ser de grande eficiência e a qualidade dos softwares tem conquistado um grande número de usuários com os mais diversos interesses, reconfigurando, assim, a indústria de software mundial. Além disso, o software livre traz consigo uma ideologia de liberdade e de colaboração em rede e estabelece novos paradigmas na relação entre as pessoas e a informática.

### **1.1 O contexto social e tecnológico do surgimento do Movimento Software Livre**

O termo software livre passou a ser empregado na década de 80, após Richard Stallman ter iniciado o desenvolvimento do projeto GNU, uma proposta de sistema operacional que garantisse a desenvolvedores e usuários a liberdade no uso do software e no acesso ao código-fonte destes para depuração e possíveis modificações. Porém, antes de abordarmos detalhadamente as características do Movimento Software Livre, torna-se necessário procurar entender o contexto sócio-tecnológico em que este surge. Desta forma, trataremos nos tópicos seguintes dos componentes tecnológicos e culturais que influenciaram o surgimento deste movimento, a exemplo das redes de computadores e da cultura hacker.

#### **1.1.1 As transformações no hardware**

Na década de 60, o mercado de computadores era dominado pelos *mainframes*, computadores de grande porte utilizados apenas por empresas, instituições governamentais, universidades e centros de pesquisa (RANGEL, 1999). No entanto, essa década também é marcada pelo início da produção dos minicomputadores, considerados como a terceira geração de computadores. Essa geração foi caracterizada por possuir vários transistores integrados em um único circuito, diminuindo consideravelmente o tamanho das máquinas e o preço destas.

A empresa Digital Equipment Corporation (DEC) foi a primeira a iniciar a produção de computadores menores e mais baratos (RANGEL, 1999). A série PDP,

lançada pela DEC, também tinha como característica o fato de possibilitar a interatividade do usuário com a máquina, pois “agora os usuários podiam emitir seus comandos e ver as respostas imediatamente, no console” (RANGEL, 1999, p. 31).

Porém, é na década de 70 que foram efetuadas as transformações mais significativas no contexto tecnológico. Em 1971, a Intel lança o primeiro microprocessador da história, batizado de 4004, iniciando a quarta geração de computadores. O 4004 trazia em um único circuito toda a lógica necessária para se construir um computador, possibilitando que este se tornasse ainda menor e mais potente (RANGEL, 1999).

A partir de então foi iniciado um ritmo bastante veloz de evolução dos processadores, o qual ficou conhecido pela “Lei de Moore”, segundo a qual, a cada 18 meses, o poder de processamento dos computadores torna-se duas vezes maior e o custo deste permanece inalterado (RANGEL, 1999).

Após o desenvolvimento do microprocessador, o próximo passo para a popularização da informática foi o desenvolvimento do computador pessoal. Segundo Rangel (1999), o Altair foi o primeiro dos computadores pessoais e possuía uma arquitetura aberta e flexível. O Altair possuía uma série de conectores internos que possibilitava a instalação de placas de circuitos eletrônicos adicionais. E a especificação técnica desses conectores era pública, permitindo a qualquer empresa ou pessoa fabricar placas adicionais (RANGEL, 1999). O autor também considera essa “arquitetura aberta” do Altair como um dos fatores responsáveis pela multiplicação dos computadores pessoais.

Nos anos seguintes, outras empresas como a Apple e a IBM também lançaram seus modelos de computador pessoal, consolidando essa tecnologia. As especificações dos conectores tornadas públicas possibilitaram que os computadores pudessem ser facilmente montados a partir de conectores produzidos por diversas empresas. Quanto à “Lei de Moore”, esta continua válida até os dias atuais. Além do aumento do poder de processamento, a tendência à miniaturização dos componentes continua em evidência.

Em relação ao software, convém destacar que inicialmente cada modelo de computador necessitava de um sistema operacional específico<sup>2</sup>. Essa situação só

---

2 O sistema operacional é um programa ou conjunto destes com a função de fazer a intermediação entre o usuário e o hardware, alocando os recursos (memória, disco, periféricos, etc.) para cada um dos aplicativos que estão sendo utilizados no computador no momento. Fonte: [http://pt.wikipedia.org/wiki/Sistema\\_operativo](http://pt.wikipedia.org/wiki/Sistema_operativo)

começou a mudar com a advento dos computadores pessoais, quando a arquitetura de hardware começou a ser padronizada, possibilitando assim que um só sistema pudesse ser utilizado em diversas máquinas com a mesma arquitetura.

### **1.1.2 O surgimento das redes de computadores e do sistema Unix**

As redes de computadores e o movimento software livre mantêm uma relação bastante próxima. A cultura hacker e os princípios de colaboração influenciaram o processo de desenvolvimento da internet, ao mesmo tempo em que esta possibilitou o contato e a colaboração entre hackers de todo o mundo, contribuindo decisivamente para o crescimento do movimento software livre.

No ano de 1969, foi colocada em funcionamento a Arpanet, rede que é considerada como o ponto de origem da internet. O objetivo inicial da Arpanet, conforme atestam autores como Castells (2003) e Rangel (1999), era conectar os computadores de diversas universidades e centros de pesquisa estadunidenses para que estes pudessem compartilhar seus recursos de pesquisas entre si.

A Arpanet foi projetada utilizando-se da tecnologia de comutação de pacotes e do conceito de comunicação distribuída. Segundo essa concepção de rede, as mensagens a serem enviadas pela rede são divididas em pacotes – pequenos grupos de dados –, os quais são endereçados e repassados de computador em computador até o seu destino final.

Assim, “O itinerário específico que cada pacote percorreria seria irrelevante; o importante é que o modelo garantia que todos os pacotes chegariam a seus destinos e seriam reagrupados, reconstituindo a mensagem original” (RANGEL, 1999, p. 34). Com isso, garantia-se também a descentralização da rede, já que cada nó atuava ao mesmo tempo como emissor e receptor de dados e estava interligado com os outros nós por caminhos redundantes. Assim, conforme afirma Rangel (1999), o desligamento de um nó da rede não comprometia o desempenho desta.

Após o desenvolvimento das tecnologias que tornaram possível a Arpanet, o passo seguinte foi possibilitar a conexão da Arpanet com outras redes. “Isso introduziu um novo conceito: uma rede de redes” (CASTELLS, 2003, p. 14). Castells afirma que, para possibilitar que as diversas redes se comunicassem entre si foi necessário o desenvolvimento de protocolos de comunicação padronizados. Nasce então, no ano de 1973, o protocolo TCP, que mais tarde foi dividido em duas partes

originando o protocolo TCP/IP, o qual é utilizado até hoje nas redes que compõem a internet.

Castells (2003) defende que a Internet nasceu da intersecção de interesses científicos, militares e da cultura libertária. Segundo este autor, apesar da Arpanet ter sido originada no Departamento de Defesa dos Estados Unidos, os objetivos militares não foram prioridade para o projeto. Os três princípios nos quais, ainda hoje, são baseados o funcionamento da internet – “estrutura de rede descentralizada, poder computacional distribuído através dos nós da rede e redundância de funções na rede para diminuir o risco de desconexão” (CASTELLS, 2003, p. 20) – foram pensados com o objetivo de fazer com que o sistema de comunicação militar pudesse resistir a um ataque nuclear. São esses princípios, no entanto, que permitiram à internet escapar do controle de governos e ser uma rede onde a liberdade de expressão e pensamento puderam aflorar.

Além disso, Castells (2003) afirma que o vínculo primordial entre as diversas expressões culturais envolvidas no desenvolvimento tecnológico da internet é a abertura e a livre modificação do código-fonte dos softwares necessários para o funcionamento da rede. Nesse sentido, são muitos os softwares de código-fonte aberto que estiveram envolvidos no desenvolvimento da internet, o Unix e os protocolos UUCP inicialmente eram abertos, assim como o protocolo TCP/IP e o navegador Mosaic<sup>3</sup> que sempre foram abertos. Posteriormente, quando a internet já estava se tornando mais popular e acessível, vieram o Netscape Navigator, o sistema operacional GNU/Linux e o programa servidor Apache.

A rápida difusão dos protocolos de comunicação entre computadores não teria ocorrido sem a distribuição aberta, gratuita de software e o uso cooperativo de recursos que se tornou o código de conduta dos primeiros hackers. [...] O software de fonte aberta, portanto, é a característica tecnológica crucial no desenvolvimento da Internet. E essa abertura é culturalmente determinada (CASTELLS, 2003, p. 25-36).

Por outro lado, foi utilizando o potencial ilimitado de conexão da rede que o movimento software livre ganhou força. As redes de computadores facilitaram a distribuição de softwares (e de seus respectivos códigos-fonte) e o contato entre desenvolvedores. Silveira (2003) destaca que a internet é um espaço essencialmente colaborativo e que o Movimento Software Livre é a expressão autêntica do uso do potencial de interação da internet. Além disso, é “[...] o grande

<sup>3</sup> O Mosaic, lançado no início de 1993, foi um dos primeiros navegadores web a possibilitar a visualização de recursos gráficos.

modelo para a consolidação de soluções compartilhadas diante de questões complexas, a partir da interação multiétnica, multinacional e multicultural” (SILVEIRA, 2003, p. 38). Foi por meio da internet que Linus Torvalds, o criador do kernel<sup>4</sup> Linux, divulgou seu software e conseguiu um grande número de colaboradores que fizeram do GNU/Linux um dos sistemas operacionais mais utilizados no mundo.

### **1.1.3 A cultura hacker e a cultura tecnomeritocrática**

Na sua análise acerca da história da internet, Manuel Castells (2003) afirma que quatro camadas culturais estiveram envolvidas no desenvolvimento da internet: a cultura tecnomeritocrática, a cultura hacker, a cultura comunitária virtual e a cultura empresarial. Destas, as duas primeiras também são características do Movimento Software Livre.

A cultura tecnomeritocrática, segundo Castells (2003), é caracterizada pela crença no desenvolvimento científico e tecnológico como fator primordial para o progresso da humanidade. Neste sentido, a descoberta tecnológica é considerada como o valor supremo, porém a relevância de cada descoberta não é avaliada pela contribuição teórica ou pelo conhecimento em si, mas sim pela aplicação desse conhecimento para realização de um objetivo. A avaliação dessa relevância é realizada através do exame dos pares entre os membros da comunidade. É este processo de avaliação que também determina quem pertence à comunidade e quem irá coordenar as tarefas e os projetos. “A reputação é central tanto para o ingresso como para a promoção nas fileiras da comunidade” (CASTELLS, 2003, p. 36).

Nas comunidades de desenvolvimento de software livre, a cultura tecnomeritocrática se faz presente na forma como os desenvolvedores são avaliados e recompensados. A reputação de um programador ou de qualquer outra pessoa na comunidade é formada a partir da colaboração que aquele realizou para a mesma. Dentro do mundo hacker, ganhar o respeito de seus pares é considerada como uma das melhores formas de recompensa pelo trabalho prestado.

O outro componente cultural envolvido no surgimento do Movimento Software Livre é a cultura hacker. De acordo com Castells (2003), a cultura hacker é baseada, primeiramente, na cultura tecnomeritocrática, ou seja, todas as características desta

---

4 O kernel é o núcleo do sistema operacional, a camada de software em contato mais direto com o hardware, controlando-o e gerenciando os recursos deste.

se aplicam àquela. O autor também destaca que a busca da excelência tecnológica é o que determina a necessidade comum de compartilhamento e de manutenção do código-fonte aberto, pois a análise e a contribuição de outros hackers é um método bastante eficaz de desenvolvimento.

Castells (2003) defende ainda que a liberdade é um dos principais valores da cultura hacker: “Liberdade para criar, liberdade para apropriar todo conhecimento disponível e liberdade para redistribuir esse conhecimento sob qualquer forma ou por qualquer canal escolhido pelo hacker” (CASTELLS, 2003, p. 42). Foi sobre o princípio da liberdade que Richard Stallman criou a Free Software Foundation. Para ele, mais importante do que a qualidade do software, era o fato de poder utilizar a tecnologia de forma livre e irrestrita.

Por outro lado, o autor aponta que, para a maioria dos hackers, a meta principal é a inovação tecnológica e o prazer de cultivar a criatividade. Contudo, a liberdade continua sendo um componente importante da visão de mundo destes hackers. Castells (2003) defende ainda que a liberdade combina-se com cooperação, pois o hacker publica seu software na internet em busca de reciprocidade. Assim,

Prestígio, reputação e estima social estão ligados à relevância da doação feita à comunidade. Assim, não se trata da retribuição esperada pela generosidade, mas da satisfação imediata que o hacker tem ao exibir sua engenhosidade para todos. Além disso, há a gratificação envolvida no objeto ofertado. Ele não tem apenas valor de troca, tem também valor de uso. O reconhecimento vem não só do ato de doar, como da produção de um objeto de valor (software inovador. (CASTELLS, 2003, p. 43).

O prazer em criar também é outro componente importante dessa cultura. Neste sentido, Castells (2003) afirma que é pelo ímpeto individual de criar que se começa a ser um hacker. Por isso, a existência intelectual destes não depende de instituições, mas sim de uma comunidade autodefinida, construída em torno de redes de computadores. Na verdade, “Há na cultura hacker um sentimento comunitário, baseado na integração ativa a uma comunidade, que se estrutura em torno de costumes e princípios de organização social informal” (CASTELLS, 2003, p. 43).

Nas comunidades de software livre existem o que Castells (2003) denomina de “veteranos tribais”, personalidades com grande prestígio pelas contribuições oferecidas à comunidade. Estes “veteranos tribais” geralmente são mantenedores de

projeto, a exemplo de Linus Torvalds, que mantém o kernel Linux porque criou sua origem.

Convém destacar que o mantenedor tem a função de receber e avaliar as contribuições de outros usuários e as incorporar ao código do software. Segundo o autor citado anteriormente, em alguns casos, a função de liderança do projeto é coletiva, com roteamento de mantenedores. Além disso, existem também os co-mantenedores, os quais ajudam a manter sub-componentes de um projeto.

Ainda segundo Castells (2003), a estrutura modular do GNU/Linux permite a este se ramificar em uma grande diversidade de projetos sem que se perca a compatibilidade entre os softwares. O autor complementa afirmando que “Co-fomentadores empreendem novos projetos por iniciativa própria, ao passo que colaboradores comuns participam da comunidade ajudando na testagem e depuração de novos programas” (CASTELLS, 2003, p. 43).

Castells (2003) adverte, no entanto, que é importante para a comunidade evitar as “bifurcações”, mais conhecidas como “fork”, as quais resultam em divisão de energia em um número excessivo de linhas de trabalho. Desta forma, os forks são aceitáveis apenas quando são esgotadas todas as outras formas de se resolver um conflito.

Ainda a respeito da liderança nas comunidades e os conflitos que surgem nestas, o autor afirma que:

[...] a comunidade aceita a hierarquia da excelência e da superioridade somente na medida em que essa autoridade é exercida para o bem-estar da comunidade como um todo, o que significa que, muitas vezes, novas tribos surgem e se enfrentam. Mas as cisões fundamentais não são pessoais ou ideológicas: são tecnológicas. (CASTELLS, 2003, p. 44).

Outro ponto importante a ser destacado é que, apesar de haver momentos de encontro físico, a maior parte da comunicação entre os hackers acontece no meio eletrônico. Em geral, eles se conhecem apenas pelos nomes que utilizam na internet. Assim, “Embora o grau mais elevado de reconhecimento costume ser associado à identificação por nomes reais, via de regra a informalidade e a virtualidade são características essenciais da cultura hacker” (CASTELLS, 2003, p. 44). Estas duas características fazem com que a cultura hacker se diferencie da cultura acadêmica e de outras culturas meritocráticas. Desta forma:

A cultura hacker é, em essência, uma cultura de convergência entre seres humanos e suas máquinas num processo de interação liberta. É uma cultura de criatividade intelectual fundada na liberdade, na cooperação, na reciprocidade e na informalidade. (CASTELLS, 2003, p. 45).

Finalmente, Castells (2003) afirma que há também comunidades hackers fundadas sob o princípios políticos, como a já citada Free Software Foundation e como a Eletronic Frontier Foundation (EFF), criada no ano de 1990 por John Perry Barlow e Mitch Kapor com o objetivo de proteger a internet do controle de governos.

Assim, vemos que a cultura hacker é o principal componente cultural envolvido no surgimento do Movimento Software Livre e que sua característica primordial é a liberdade para criar e compartilhar conhecimento pelos mais diversos meios e maneiras possíveis. Além disso, o contexto tecnológico das décadas de 1970/80, nas quais ocorreram grandes avanços e os computadores assumiram cada vez mais importância na vida das sociedades, foi outro fator para a emergência de um movimento em defesa da liberdade do software.

## **1.2 Software livre: a ideologia por detrás do código**

Muito além de ser uma alternativa tecnológica aos chamados softwares proprietários<sup>5</sup>, o software livre traz em si toda uma ideologia de liberdade, a qual, como vimos, tem sua origem na cultura hacker. Nesse sentido, é da indignação ativa de um hacker que surge os primeiros esforços em defesa da liberdade do uso da tecnologia. Tais esforços irão culminar no desenvolvimento de diversos softwares e de um sistema operacional completo, bem como na criação de licenças que definem as liberdades que um software deve conter para ser considerado livre.

### **1.2.1 Do GNU ao GNU/Linux**

Como descreve Richard Stallman em seu ensaio intitulado “GNU Project”, o costume de compartilhar o código fonte dos softwares é tão antigo quanto a própria informática e tão comum como o hábito de compartilhar receitas culinárias, porém,

---

<sup>5</sup> Define-se software proprietário como aquele cujo acesso ao código fonte é negado, impedindo dessa forma o estudo de como o software funciona e a realização de modificações na sua estrutura. Alguns softwares proprietários são distribuídos gratuitamente, enquanto outros são vendidos e restringem até mesmo a realização de cópias e o uso para algumas finalidades.

ainda não se usava o termo “*free software*” (software livre) para classificar esses softwares que eram livremente compartilhados.

Stallman (2002) ressalta também que, quando começou a trabalhar no Laboratório de Inteligência Artificial (AI LAB) do MIT, em 1971, passou a fazer parte de uma comunidade de compartilhamento de software que já existia há bastante tempo. No início da década de 80, porém, os computadores da série PDP-10, os quais eram utilizados no Laboratório de Inteligência Artificial do MIT, deixaram de ser produzidos, com isso praticamente todos os programas que os hackers do laboratório haviam desenvolvido em quinze anos se tornaram obsoletos. Além disso, os computadores modernos daquela época tinham seu próprio sistema operacional, os quais eram proprietários. Para obter uma cópia executável desses softwares, era necessário assinar um acordo de confidencialidade (“*nondisclosure agreement*”). Segundo Stallman, “A cooperação em comunidade ficou esquecida. A regra imposta pelos donos dos softwares proprietários era: 'Se você compartilhar com seu vizinho, você é um pirata. Se você deseja qualquer alteração [no software], peça-nos para fazê-las.’” (STALLMAN, 2002, p. 18, tradução nossa).

Foi então que Stallman se questionou a respeito de quais softwares ele deveria desenvolver para tornar a comunidade de compartilhamento possível novamente. “A resposta foi clara: a primeira necessidade era um sistema operacional, o qual é o software crucial para começar a utilizar um computador” (STALLMAN, 2002, p. 19, tradução nossa). Stallman adverte que sabia também que um sistema operacional não consistia apenas de um kernel, eram necessários alguns programas, como compiladores, editores de texto, interpretadores de comando e outros mais. Assim, o nome escolhido por Stallman para esse sistema operacional foi GNU, um acrônimo recursivo para “*GNU's Not Unix*” (GNU não é Unix).

O primeiro software desenvolvido pelo projeto GNU foi o editor de texto Emacs, que, no início de 1985, já estava suficientemente bom para ser usado. Após o Emacs, vieram o compilador para a linguagem de programação C, a biblioteca GNU C Library e outros softwares que continuam sendo desenvolvidos até hoje. Também no ano de 1985, Richard Stallman fundou a Free Software Foundation, uma fundação com o objetivo de buscar recursos para o desenvolvimento de softwares livres.

No início do Projeto GNU, segundo Stallman (2002), esperava-se que fosse

desenvolvido um sistema operacional completo, para, em seguida, ser disponibilizado como um só conjunto. No entanto, cada software que era escrito e disponibilizado ia se tornando popular e pessoas começavam a estendê-lo e portá-lo para outros sistemas. Os desenvolvedores do GNU, por sua vez, investiam seu tempo em adicionar novos recursos e em gerenciar as contribuições de outros usuários, ao invés de desenvolver novos softwares. Este processo fez com que os softwares se tornassem mais poderosos e o GNU ganhou mais colaboradores e recursos financeiros. Porém, Stallman (2002) avalia que, provavelmente, isto também contribuiu para atrasar em vários anos a finalização de um sistema operacional com o mínimo de funcionalidade.

Em 1990, já havia sido desenvolvido quase todo o sistema GNU e faltava apenas o kernel. Assim, foi iniciado o desenvolvimento de um kernel intitulado GNU Hurd. Entretanto, no ano de 1991, Linus Torvalds desenvolveu um kernel compatível com Unix e o chamou de Linux. Um ano depois, o Linux foi combinado com os softwares do Projeto GNU, resultando num completo sistema operacional livre, o GNU/Linux, nome que expressa a combinação do kernel Linux com o sistema GNU.

### **1.2.2 As licenças livres**

Com o início do desenvolvimento dos softwares do projeto GNU, Stallman evidenciou a necessidade de um dispositivo legal que garantisse que estes softwares continuariam livres para todos os usuários. Neste sentido, convém atentar para o fato de que:

Se um software é livre quando sai das mãos de seu autor, não significa necessariamente que este permanecerá livre para todos que tenham uma cópia. Por exemplo, um software de domínio público (ou seja, que não têm copyright) é software livre, porém qualquer pessoa pode modificá-lo e fazer uma versão proprietária. (STALLMAN, 2002, p. 22, tradução nossa).

Com isso, o método que Stallman encontrou para prevenir que os softwares do Projeto GNU fossem convertidos em software proprietário foi denominado de *copyleft*. “Copyleft usa a lei de copyright, mas a inverte para servir ao oposto de seu objetivo usual: ao invés de ser um meio de privatizar o software, torna-se um meio de manter o software livre” (STALLMAN, 2002, p. 22, tradução nossa). O *copyleft* garante aos usuários as quatro liberdades características do software livre, porém determina que as versões modificadas do software devam ser livres também. A GNU

General Public License (ou GNU GPL) é a licença utilizada para implementar o *copyleft* nos softwares do Projeto GNU. A primeira versão da GNU GPL foi publicada no ano de 1989, desde então, a licença passou por duas atualizações.

### **1.2.3 As quatro liberdades**

Para um software ser considerado livre, e compatível com a GNU GPL, ele deve proporcionar quatro liberdades: (1) a liberdade de usar o software para qualquer finalidade; (2) a liberdade de modificar o software para que ele se adapte às diversas necessidades (um requisito para essa liberdade é que se tenha acesso ao código-fonte); (3) a liberdade de redistribuir cópias do software, seja gratuitamente ou vendidas, e (4) a liberdade de distribuir versões modificadas do software, para que a comunidade possa se beneficiar das melhorias.

Comumente o termo software livre é interpretado apenas como software gratuito, porém, como adverte Stallman (2002), software livre é uma questão de liberdade não de preço. O autor afirma ainda que não existe contradição na venda de cópias dos softwares livres, a venda de CD's, inclusive, é uma importante fonte de recursos para o desenvolvimento destes softwares.

### **1.2.4 As motivações do software livre**

Como se percebe, a motivação de Richard Stallman em desenvolver e promover o software livre é bastante ideológica. Sua luta é pela liberdade no uso dos recursos computacionais num mundo onde os computadores assumem uma importância estratégica. Nesse sentido, o acesso ao código fonte dos softwares, ou seja, a possibilidade de estabelecermos o domínio sobre a tecnologia que utilizamos, é considerada como um fator crucial para uma sociedade livre.

Porém, com o alto nível técnico que alguns softwares livres foram adquirindo ao longo do tempo e, conseqüentemente, o uso comercial destes, muitos usuários e programadores começaram a propagar mais as vantagens técnicas do que a ideologia existente por trás do software livre. Neste contexto, em 1998, foi fundada a Open Source Initiative, uma entidade com o objetivo de propagar as vantagens técnicas e econômicas dos softwares de código fonte aberto. Percebe-se, portanto, que a Iniciativa Open Source substitui o termo “free software” por “open source”

(código aberto), retirando o foco do caráter de liberdade para a conveniência da disponibilidade do código fonte e de alta qualidade.

Stallman (2002) afirma que muito esforço tem sido feito pelas comunidades de software livre para se conquistar mais usuários, porém não se vê o mesmo esforço para divulgar a ideologia que está por trás do software livre. O autor considera que muitos usuários têm utilizado softwares livres apenas por motivos técnicos e que nisso há um lado positivo – mais consumidores para o mercado de softwares livres, mais interesse em desenvolvê-los e mais pressão para que as empresas comercializem softwares livres em vez de proprietários. O autor adverte, no entanto, que os usuários devem estar conscientes da necessidade de defender sua liberdade frente aos desafios que o software livre precisa enfrentar<sup>6</sup>.

Nesse sentido, Stallman (2002) critica a Open Source Initiative em função do seu discurso estar mais comprometido com a qualidade técnica do software do que com os ideais de liberdade e comunidade, o que torna ainda mais difícil a difusão destes ideais.

### 1.3 Ideologia e código fonte

Na verdade, muito mais do que uma alternativa tecnológica ao software proprietário, o software livre traz em si novos paradigmas na forma como trata a propriedade intelectual, o acesso ao conhecimento e a dinâmica de produção e uso da tecnologia.

Neste contexto, Silveira (2003) define o Movimento Software Livre como um movimento pelo compartilhamento do conhecimento tecnológico. Conhecimento este que se tornou essencial na sociedade atual, visto que a tecnologia assume um papel muito importante desde a mediação das relações pessoais até a economia dos países. O autor afirma que:

Fazer programas de computador será cada vez mais vital para um país. [...] Capacitar a inteligência coletiva de cada país para dominar os códigos-fonte, principalmente dos sistemas operacionais será cada vez mais decisivo para o desenvolvimento de diversas soluções na área das tecnologias da informação e da comunicação. (SILVEIRA,

---

6 Tais desafios incluem a questão da patente de software, que restringe o uso de determinados algoritmos e recursos computacionais por mais de 20 anos; a questão das especificações de hardware que, se não divulgados pelas empresas, comprometem o suporte de tais componentes em sistemas livres; e as facilidades que determinados softwares proprietários oferecem e que podem seduzir programadores e usuários.

2003, p. 6).

Por outro lado, o software, assim como qualquer outro tipo de informação, não é um bem tangível, ou seja, este pode ser compartilhado sem representar nenhuma perda para aquele que o compartilhou. Silveira (2003) afirma que a ciência só pôde se desenvolver em virtude da transmissão e do compartilhamento de conhecimento ao longo de toda a história. Assim:

Na era informacional, quanto mais se compartilha o conhecimento, mais ele cresce. Os softwares são os principais intermediadores da inteligência humana na era da informação. Garantir seu compartilhamento é essencial para a construção de uma sociedade livre, democrática e socialmente justa. (SILVEIRA, 2003, p. 7).

Portanto, possibilitar o acesso ao conhecimento tecnológico é possibilitar que as pessoas e nações tenham autonomia tecnológica, não se submetam a monopólios e que a inteligência coletiva local possa se desenvolver. O conhecimento livre e aberto é essencial para reduzir as desigualdades entre países e entre os extratos da sociedade.

Prado *et al.* (2005), por sua vez, afirma que o movimento software livre é considerado como um defensor da liberdade na Era da Informação, sendo constantemente apropriado como bandeira e arma estratégica de uma luta contra-hegemônica.

Dessa forma, costuma-se afirmar e defender arduamente que os sistemas operacionais livres são superiores tecnicamente, mais seguros e proporcionam uma imensa economia, por não cobrarem licenças de uso. Portanto, são mais eficientes e econômicos. (PRADO *et al.*, 2005, p. 36).

Segundo Prado *et al.* (2005) apesar desta abordagem ser interessante por tocar em questões relevantes como a desigualdade do desenvolvimento tecnológico mundial, ela ignora características muito importantes do fenômeno do software livre: “sua dinâmica de produção, suas regras de circulação de produção e a mudança de comportamento diante dos meios, operadas por sua lógica de utilização” (PRADO *et al.*, 2005, p. 37).

De acordo com o autor, a primeira perspectiva vê o software livre como um produto que, em sua materialidade, possui uma natureza distinta em relação à do software proprietário. Assim, ao observarmos o software livre pelo âmbito do seu processo de produção e não como um produto estático, perceberemos diferenças radicais entre este e os softwares proprietários. Uma vez que seu processo de produção é colaborativo, público, múltiplo, metaestável e contínuo.

Sendo assim, os softwares livres e os proprietários diferem não só quanto à natureza de sua materialidade, mas principalmente, quanto às relações sociais em que estão inseridos e produzem. O software livre não é melhor que o software proprietário: ele é de outra ordem! (PRADO *et al.*, 2005, p. 37)

Prado *et al.* (2005) afirma ainda que o software livre hoje possui uma dinâmica de produção e um circuito de circulação próprios, os quais, com o passar do tempo extrapolaram a esfera dos especialistas, tais como universidades, centros de pesquisa e grandes empresas, e passou a atingir também o grande público, inserindo-o nesse processo colaborativo de produção pública. O autor defende que “a grande inovação desse processo está na estrutura de divisão do trabalho em uma rede aberta” (PRADO *et al.*, 2005, p. 38). O que protege essa rede e garante sua expansão são as licenças públicas. Estas licenças impedem que o código fonte dos softwares sejam fechados e exigem que os softwares derivados de outros softwares livres devam ser publicados em licenças idênticas.

Com ênfase, Prado *et al.* (2005) destaca dois aspectos da dimensão dinâmica do software livre: “(1) desenvolvimento colaborativo e relação de utilização ativa que pressupõe aprendizado; (2) ressignificação e refundação das relações de trabalho sob outros mecanismos de motivação que apontam para uma outra ecologia do virtual” (PRADO *et al.*, 2005, p. 39).

O autor afirma que a utilização do software livre é ativa, pois exige uma relação de aprendizado do usuário e rompe com o conceito de produto acabado. Desde o lançamento do software, seus erros e deficiências são divulgados, para que, conscientes destes e com o código fonte disponível, outros desenvolvedores possam colaborar no processo de desenvolvimento.

Esse processo de colaboração está aberto à participação de qualquer usuário, independentemente de sua habilidade com programação. Portanto, usuários que não possuem os conhecimentos necessários para propor modificações no código fonte, podem colaborar reportando erros, produzindo ou traduzindo documentação e sugerindo melhorias no software. Assim:

O interessante é que, mesmo para a comunicação de um erro simples, o usuário necessita aprender um procedimento específico. Somando-se a isso a comunicação que ele estabelece com o desenvolvedor e a possibilidade de estudo do código-fonte, as portas para que ele se integre no processo de desenvolvimento estão abertas. (Prado *et al.*, 2005, p. 40).

Prado *et al.* (2005) afirma que nesse processo de colaboração, “[...]”

estabelece-se uma relação horizontal entre produtores e usuários que é completamente diferente da relação existente entre produtor/consumidor ou entre provedor/cliente, produzindo relações sociais de naturezas qualitativamente diferentes” (Prado *et al.*, 2005, p. 40). A relação produtor/consumidor é constituída através de trocas comerciais, mediadas pelo dinheiro. Já na relação entre produtor e usuário de software livre, as relações são constituídas mediante trocas diretas, cujo laço é a comunicação entre pessoas nos canais públicos das comunidades. As trocas são tão intensas que os papéis chegam a se confundir<sup>7</sup>.

A comunicação estabelecida nesse processo de desenvolvimento e utilização dos softwares, segundo Prado *et al.* (2005), criam vínculos pessoais entre indivíduos de todo o planeta, estabelecendo, dessa forma, espaços públicos de comunicação e colaboração tecnológica entre indivíduos das mais diversas culturas e origens. Essa diversidade de perspectivas e contextos culturais reflete no processo de produção, gerando softwares para os mais diversos fins, adaptados para os diversos tipos de hardware e traduzidos para uma grande quantidade de línguas e dialetos.

Além disso, Prado *et al.* (2005) afirma que, enquanto nas relações capitalistas o foco do trabalho gira em torno do capital e visa a produção de valor de troca, na rede de produção de software livre o trabalho é agenciado por projetos coletivos, com o objetivo de produzir valor de uso. Assim, “Ao passo que em um modelo o trabalho é motivado pela competição, no outro é motivado pela colaboração e generosidade” (Prado *et al.*, 2005, p. 43).

Desta forma, enquanto o modelo de produção de software proprietário enfatiza o produto, o modelo de software livre enfatiza o processo. Quando um software livre está sendo desenvolvido, são lançadas versões beta ainda instáveis e incompletas para que usuários e outros desenvolvedores possam contribuir com o processo de produção, e até sugerir novos rumos para este. A partir das colaborações, o software vai sofrendo alterações e ganhando estabilidade. Prado *et al.* (2005) afirma que o software livre permanece constantemente num estado de

---

7 Apesar de o software livre poder ser comercializado, bem como existir a oferta comercial de alguns serviços, encontra-se sempre aberta a oportunidade de um contato direto entre usuário e desenvolvedor por meio de fóruns virtuais, listas de discussão, salas de chat e eventos de software livre, por exemplo. Ao contrário do suporte comercial oferecido por uma empresa de software, tal contato não é mediado pelo dinheiro. O usuário de software livre ao entrar em contato com a comunidade do software pode contribuir com o relato de erros, sugestões e críticas ao programa. Como retribuição, o usuário pode ter suas dúvidas solucionadas e a melhoria da qualidade do software. Com isso, ele deixa de ser apenas um usuário e passa a ser também um colaborador do software.

metaestabilidade, pois está sempre sujeito a possibilidades de novos incrementos.

Por outro lado, os softwares proprietários, quando lançados, são apresentados como um produto acabado e estável. O potencial de evolução desses softwares fica restrito às possibilidades imaginadas por um grupo limitado de pessoas. Leis de patente e de proteção à propriedade intelectual impõem barreiras legais à participação de um maior número de pessoas nesse processo.

A abertura do primeiro modelo proporciona a multiplicidade de projetos que exploram de forma pública diferentes nuances e buscam diferentes objetivos. Por ser um processo aberto, a sociedade pode acompanhar os rumos do desenvolvimento técnico e as escolhas feitas nesse processo. (Prado *et al.*, 2005, p. 45)

Prado *et al.* (2005) afirma que, quando o desenvolvimento tecnocientífico está atrelado à lógica do capital, o campo do virtual, isto é, a esfera de possibilidades de futuro, acaba sendo colonizado. “Em contraposição, o processo de desenvolvimento do software livre proporciona uma ecologia do virtual, sendo este um campo aberto ao debate e à exploração pública” (Prado *et al.*, 2005, p. 45).

Dessa forma, evidencia-se que as quatro liberdades garantidas pelo software livre trazem conseqüências que vão muito além das vantagens técnicas destes em relação a alguns softwares proprietários. O software livre possui um processo de produção completamente distinto e uma outra abordagem em relação à circulação do conhecimento na sociedade. Essas especificidades fazem com que o debate gerado pelo Movimento Software Livre tenha um caráter ideológico, pois afeta os mais diversos setores da sociedade, não apenas a esfera tecnológica.

## **1.4 O método de desenvolvimento do software livre**

Como vimos no tópico anterior, o software livre difere do software proprietário não apenas enquanto produto, mas também em relação às características envolvidas em seu processo de produção. Neste sentido, a percepção do potencial do exame do código entre os membros da comunidade e da colaboração dos usuários no processo de desenvolvimento torna-se de grande importância para o sucesso dos projetos de software livre. Além disso, a construção de estratégias de financiamento tem sido de grande valia para o aumento da qualidade e difusão dos softwares.

### 1.4.1 Os modelos Catedral e Bazar: inovações trazidas por Linus Torvalds

Em sua publicação “A catedral e o bazar”, de 1998, Eric Raymond compara dois modelos de desenvolvimento de software e aponta o modelo bazar, em que as atualizações são lançadas o mais cedo possível, como uma das inovações trazidas por Linus Torvalds quando este iniciou o desenvolvimento do Linux.

Nesta abordagem, o modelo que o autor denominou de “catedral” se preocupa em corrigir os erros dos softwares antes de o liberar para o “mercado” e é o método utilizado pelos softwares proprietários e pelos primeiros desenvolvedores do GNU. Já no modelo que Raymond (1998) denominou de “bazar”, os softwares são liberados o quanto antes e a verificação de erros fica por conta da comunidade de colaboradores e usuários do software. Raymond (1998) considera esta uma das grandes inovações trazidas por Linus Torvalds quando este disponibilizou o Linux. Neste sentido, o autor é categórico ao declarar que “De fato, eu penso que a engenhosidade do Linus e a maior parte do que desenvolveu não foram a construção do kernel do Linux em si, mas sim a sua invenção do modelo de desenvolvimento do Linux” (RAYMOND, 1998. p. 4).

Raymond (1998) também afirma que o Linux é subversivo, pois é um sistema operacional que surgiu do trabalho de milhares de colaboradores espalhados pelo mundo e conectados pela internet, os quais se dedicavam ao Linux apenas em seu tempo livre. O autor aponta também dois fatores que colaboram para a alta qualidade técnica da maioria dos softwares originados nas comunidades de Linux: o primeiro é que os programadores escolhem o que vão desenvolver pela necessidade e vontade pessoais. Outro fator apontado por Eric Raymond (1998) é que “é quase sempre mais fácil partir de uma boa solução parcial do que do nada” (RAYMOND, 1998, p. 2), assim os bons resultados em software livre também podem ser justificados pela grande quantidade de código já escrito e que pode ser reutilizado em outros projetos.

Linus Torvalds, por exemplo, não tentou realmente escrever o Linux do nada. Ao contrário, ele começou reusando código e idéias do Minix, um pequeno sistema operacional Unix-like para máquinas 386. [...] A tradição do mundo Unix de compartilhar o código fonte foi sempre amigável para a reutilização de código. [...] O mundo Linux tem levado esta tradição quase a seu limite tecnológico; tem terabytes de códigos abertos amplamente disponíveis. (RAYMOND, 1998, p. 2).

Raymond (1998) defende também que o usuário deve ser tratado como um personagem importante do processo de desenvolvimento do software. Além de fornecer ao desenvolvedor um retorno em relação à forma como o software satisfaz suas necessidades, os usuários podem tornar-se co-desenvolvedores, visto que, muitos usuários de GNU/Linux são hackers também, e o código-fonte aberto possibilita que qualquer pessoa com conhecimento técnico em programação possa localizar os erros. Como resultado desse processo, o autor aponta uma redução do tempo de depuração e defende que o desenvolvimento seja mais dirigido ao usuário, procurando suprir as necessidades destes e tratando-os como parte importante do processo de produção do software.

Essa abordagem é confirmada por Hexsel (2002), o qual afirma que num projeto no modelo “bazar”, a evolução da funcionalidade do software é orientada pelos usuários deste, principalmente os que também são desenvolvedores. Hexsel (2002) defende que a alta qualidade dos softwares desenvolvidos sob o modelo “bazar” deve-se à quantidade de usuários e desenvolvedores que se envolvem no processo. Por isso, “Quando um programa não atrai uma comunidade suficientemente grande de usuários e programadores dispostos a suportá-lo, geralmente seu desenvolvimento pára e o projeto tende a estagnar” (HEXSEL, 2002, p. 11).

Outra inovação na forma de desenvolvimento adotada por Linus Torvalds é a liberação de novas versões do software tão cedo quanto possível, como afirma Eric Raymond:

Liberações novas e freqüentes são uma parte crítica do modelo de desenvolvimento do Linux. A maioria dos desenvolvedores (incluindo eu) costumava acreditar que esta era uma má política para projetos maiores que os triviais, porque versões novas são quase por definição cheias de erros e você não quer acabar com a paciência dos seus usuários. (RAYMOND, 1998, p. 4).

Raymond (1998) complementa afirmando que liberações novas e freqüentes refletem em um número maior de pessoas para depurar o código, além de manter os usuários e colaboradores constantemente estimulados e recompensados ao ver a melhoria do software. Apesar de Linus Torvalds não ser o pioneiro nessa prática de liberação, foi a pessoa que a levou ao extremo, chegando a lançar uma nova versão do kernel mais de uma vez por dia. Essa prática de liberação pode gerar prejuízos em termos de estabilidade do software, no entanto ajuda a aumentar o número de pessoas-hora dedicadas à depuração e ao desenvolvimento, por conseqüência

“dada uma base grande o suficiente de beta-testers e co-desenvolvedores, praticamente todo problema será caracterizado rapidamente e a solução será óbvia para alguém” (RAYMOND, 1998, p. 5). Ou de forma mais simples, “dados olhos suficientes, todos os erros são triviais” (RAYMOND, 1998, p. 5). Esta é a diferença que Raymond (1998) considera fundamental entre os estilos catedral e bazar. Num sentido oposto, no estilo catedral “Os erros e problemas de desenvolvimento são difíceis, insidiosos, um fenômeno profundo. Leva meses de exame minucioso por poucas pessoas dedicadas a desenvolver confiança de que você se livrou de todos eles.” (RAYMOND, 1998, p. 5).

Já no método bazar, “você assume que erros são geralmente um fenômeno trivial – ou, pelo menos, eles se tornam triviais muito rapidamente quando expostos para centenas de ávidos co-desenvolvedores triturando cada nova liberação” (RAYMOND, 1998, p. 6).

Raymond (1998) define também algumas condições essenciais para o sucesso de um projeto no estilo bazar. Segundo o autor, é muito difícil iniciar um projeto já no estilo bazar, antes é necessário que já exista algum código escrito e, mesmo que o software apresente erros grosseiros e ainda não funcione bem, que seja capaz de convencer possíveis colaboradores de que o projeto possa evoluir para algo realmente bom. Além disso, Raymond (1998) defende que o coordenador do projeto tenha uma capacidade de reconhecer boas idéias de projetos de outras pessoas e também tenha boa habilidade de comunicação e relacionamento.

Além disso, o autor afirma que Linus Torvalds soube utilizar, de forma bastante eficaz, as possibilidades da internet, naquele período que foi justamente o nascimento da World Wide Web. Raymond (1998) defende também que foi importante o desenvolvimento de um estilo de liderança e de algumas formalidades cooperativas, o que possibilitou aos líderes de projetos no estilo bazar atraírem co-desenvolvedores e fazer com que estes se sintam recompensados em cooperar com o projeto. Desta forma:

A 'função empreendedora' que os hackers do Linux estão maximizando não é economia clássica, mas é a intangível satisfação do seu próprio ego e reputação entre outros hackers. [...] Nenhum desenvolvedor de código fechado pode competir com o conjunto de talento que a comunidade do Linux pode dar para resolver um problema. (RAYMOND, 1998, p. 14 e 15)

Assim, vemos que o método de desenvolvimento de software livre denominado por Eric Raymond de bazar mostra-se bastante eficiente. Quando tal

método consegue agregar um bom número de colaboradores e usuários, pode atingir resultados técnicos superiores aos softwares desenvolvidos sob o método catedral, como vem provando diversos softwares livres.

#### **1.4.2 O ciclo de vida de um projeto de software livre**

O processo de desenvolvimento de um software livre é iniciado quando o autor escreve uma versão inicial e publica o seu código-fonte, que, por vezes, é incompleta e com recursos ainda restritos. Se o programa for considerado interessante e útil por outros desenvolvedores e usuários, estes o utilizam, descobrem e reportam erros e propõem melhorias. Cabe então ao desenvolvedor acatar ou não essas sugestões, incorporar as correções e publicar a nova versão do software.

Hexsel (2002, p.10) afirma que “a versão melhorada atrai mais usuários, que descobrem outros erros e introduzem novas melhorias, o que leva a uma nova versão”. Conforme apontado anteriormente, se o programa é considerado útil e consegue atrair um bom número de usuários e desenvolvedores, esse ciclo irá se repetir em poucos meses até chegar numa versão estável. Segundo o autor:

Nessas condições a comunidade de suporte ao programa atinge massa crítica, e isso garante a continuidade de seu desenvolvimento e suporte. [...] Quando um programa não atrai uma comunidade suficientemente grande de usuários e programadores dispostos a suportá-lo, geralmente seu desenvolvimento pára e o projeto tende a estagnar. (HEXSEL, 2002, p. 10-11).

Outro motivo freqüente para a estagnação de um software é a perda de interesse por parte do autor e dos principais desenvolvedores. Hexsel (2002) adverte, no entanto, que isso não é necessariamente ruim, pois pode ocorrer de ter surgido um software de qualidade melhor ou mais promissor, o qual passará a contar com a colaboração desta comunidade.

#### **1.4.3 Estratégias de financiamento de software livre**

Apesar de grande parte dos softwares livres surgirem a partir da dedicação no tempo livre de milhares de desenvolvedores espalhados pelo mundo, a adoção de estratégias de financiamento tem se mostrado de grande importância para o crescimento técnico destes softwares.

Como vimos no tópico anterior, o ciclo de desenvolvimento de um software livre começa no momento em que um programador libera a versão alfa do programa na rede. No entanto, é bom observar que o processo de desenvolvimento e distribuição dos softwares envolve algumas despesas, as quais precisam ser custeadas para que o software possa conquistar um bom número de usuários e colaboradores e, assim, evoluir tecnicamente. Alguns softwares, por exemplo, precisam ser testados e sofrer adaptações para se tornar compatível com os diversos modelos de hardware existentes no mercado. A distribuição e divulgação dos softwares também envolve custos, como a manutenção de servidores, prensagem de CD's e participação em eventos.

Além disso, como afirmado anteriormente, quanto mais pessoas/horas dedicadas à depuração e desenvolvimento do código mais rápido o software poderá evoluir. Neste sentido, ter desenvolvedores que possam se dedicar em tempo integral ao projeto pode fazer bastante diferença.

Com isso, várias comunidades de software livre passaram a adotar estratégias de financiamento para seus projetos. As principais estratégias podem ser classificadas nas seguintes categorias:

- Doações de usuários – procura-se convencer os usuários a colaborarem por meio da doação de dinheiro e, em casos restritos, também de componentes de hardware;
- Patrocínio de empresas e instituições governamentais: podem ser dividido em três tipos: Doação em dinheiro; doação de hardware e serviços de hospedagem na rede; contratação de desenvolvedores que podem se dedicar ao software em tempo integral ou parcial;
- Venda de produtos e serviços – algumas comunidades oferecem serviços como suporte comercial ao software e hospedagem web, e vendem produtos como livros, CD's de instalação, camisetas, entre outros artigos;
- Publicidade nos sites – a venda de espaço de publicidade nos sites da comunidade pode também ajudar na obtenção de recursos para o projeto.

Além disso, várias comunidades passaram a criar fundações, as quais cumprem a função de obter recursos e representar a comunidade frente aos parceiros e patrocinadores. Em geral, os softwares livres mais populares atualmente utilizam alguma estratégia de financiamento, a exemplo do Apache, OpenOffice, Firefox, GNOME, KDE e também do Blender, objeto desta pesquisa.

No terceiro capítulo analisaremos de forma detalhada as estratégias adotadas por cada um desses softwares.

## 1.5 Penetração e diversidade do software livre

Como dito anteriormente, o Projeto GNU marca o início do Movimento Software Livre, com o desenvolvimento dos primeiros aplicativos, a criação da licença GNU GPL e da Free Software Foundation. Um marco da história do software livre é o lançamento do kernel desenvolvido por Linus Torvalds, denominado Linux. O kernel, como já foi afirmado, é o núcleo do sistema operacional e, em conjunto com os aplicativos que já vinham sendo desenvolvidos pelo projeto GNU, formou o sistema operacional GNU/Linux. Nesse sentido, é com o lançamento do Linux que o software livre começou a se popularizar entre os programadores e estudantes de computação de todo o mundo. Desde então, comunidades de desenvolvimento se formam a todo momento e inúmeros softwares são desenvolvidos e constantemente aprimorados.

Após o GNU/Linux, um dos primeiros softwares a se tornar bastante popular e alcançar uma consolidação frente a outros softwares proprietários foi o Apache, um servidor de páginas para a Web. Já em abril de 1996, um ano após ter sido lançada a sua primeira versão, o Apache já era o software servidor mais utilizado no mundo<sup>8</sup>.

Convém ressaltar que, nos primeiros anos, o GNU/Linux e demais softwares livres eram utilizados majoritariamente por nerds, hackers e especialistas em informática, os quais comumente eram aqueles que contribuíam com o processo de desenvolvimento dos programas. Com isso, os softwares foram desenvolvidos de acordo com os interesses e necessidades desses grupos, demorando mais tempo para se consolidar em outras áreas da computação, tais como desktop doméstico, produção gráfica, animação e edição de áudio e vídeo. Entretanto, atualmente o software livre já se afirma como alternativa tecnológica de alta qualidade para as mais diversas áreas, a exemplo da área de animação em que softwares como o Blender já estão sendo largamente utilizados, inclusive para produzir conteúdo de qualidade profissional.

Neste contexto, vários governos, a exemplo do Brasil, Índia, França, México e

---

<sup>8</sup> Dados do site <http://www.netcraft.com>. No mês de janeiro de 2008, 50,61% dos domínios de internet estavam hospedados em servidores rodando o Apache. O software servidor da Microsoft estava na segunda posição com 35,81%.

China, como cita Manuel Castells (2003), estão adotando o Linux e promovendo o seu uso. O Brasil é considerado como um dos países em que a adoção de software livre está mais avançada. Adalto Guessser afirma que “No Brasil, a disseminação de software livre foi impulsionada pelo governo do Rio Grande do Sul, a partir da realização do I Fórum Internacional de Software Livre” (GUESSER, 2004, p. 11), no ano de 2000. Desde então, vários programas de inclusão digital e também órgãos públicos estão utilizando software livre.

Entre os projetos de inclusão digital, destacam-se o projeto “Pontos de Cultura”, do Ministério da Cultura, o “Casa Brasil”, ligado ao Instituto de Tecnologia da Informação (ITI) e o “Acessa São Paulo”, uma rede de telecentros do governo do estado de São Paulo. Já entre os órgãos públicos, destacam-se o Banco do Brasil, o qual encerrou o ano de 2007 com 50 mil máquinas rodando GNU/Linux<sup>9</sup>, a Caixa Econômica Federal, a Dataprev e o INSS.

Além disso, o governo brasileiro tem incentivado a adoção de software livre através do programa “PC Conectado”, cujo objetivo é facilitar a aquisição de computadores pela população. Para isso, o governo reduziu os impostos sobre computadores vendidos com software livre.

As empresas privadas também têm adotado largamente os softwares livres. Guessser (2004) cita o caso de grandes empresas estrangeiras, como Lufthansa, DowJones, Amazon.com e WallMart como expoentes desta iniciativa, além da agência americana NASA e do sistema do Pentágono. Quanto à área de animação, Rowe (2007) afirma que praticamente todos os grandes estúdios de animação e efeitos visuais utilizam GNU/Linux como sistema operacional. Segundo o autor, a DreamWorks talvez seja um dos maiores casos, com mil computadores desktops e três mil servidores utilizando o GNU/Linux. Os softwares gráficos utilizados nestes estúdios, porém, são, em grande parte, proprietários. Já em relação ao Brasil, Guessser (2004) destaca as empresas Lojas Colombo e as Casas Bahia, como exemplo de empresas que utilizam software livre.

Desta forma, vemos que o software livre mostra-se ao mesmo tempo como uma ideologia e como um modelo inovador de desenvolvimento de software. Tal ideologia está baseada na defesa da liberdade do conhecimento, em especial do

---

9 Fonte: Site BR-Linux (<http://br-linux.org/2008/banco-do-brasil-encerra-2007-com-50-mil-estacoes-com-linux/>)

conhecimento tecnológico, e da colaboração entre hackers e usuários de computador de todo o mundo. Ao tornar o conhecimento livre, altera-se também diversas relações: do relacionamento entre usuário e programador à desigualdade social e tecnológica entre os países.

Enquanto modelo de desenvolvimento de software, este também tem se mostrado bastante eficiente, vide a qualidade e reputação que vários softwares livres têm alcançado. Com isso, abordaremos, no próximo capítulo, as questões referentes à produção de animação em computador para, em seguida, analisarmos o caso do Blender, software livre de grande destaque na área de animação tridimensional.

## **2. Ambiente digital de produção de animação**

A arte da animação tem sua origem em dispositivos desenvolvidos ainda no século XVII e, ao longo do tempo, passou por diversas transformações tanto em nível tecnológico, como enquanto linguagem artística. Nas últimas décadas, com o desenvolvimento de recursos avançados de computação gráfica, a animação passou a fazer uso de ferramentas digitais no seu processo de produção. Porém, como veremos nos tópicos seguintes, o uso de novas tecnologias não implicam no desuso das técnicas anteriores, mas sim no aumento das possibilidades expressivas.

Iniciaremos, pois, com um breve resumo da história da animação até a chegada das técnicas de animação digital e, em seguida, entraremos nos detalhes da forma de funcionamento desta.

### **2.1 Um breve resgate histórico da animação**

A palavra animação, segundo Lucena (2005), é derivada do verbo latino *animare*, que tem o sentido de “dar vida a algo” e apenas no século XX é que passou a ser utilizada para descrever imagens em movimento. De acordo com o autor, a essência da animação está no movimento, e o fascínio que a animação provoca em artistas e expectadores pode ser explicado pela grande atração visual provocada pelo movimento.

Neste contexto, o movimento (ou a ilusão de), é conseguida através da rápida sucessão de imagens, seja através de dispositivos óptico-mecânicos, como o zootroscópio, ou pelo uso de técnicas que envolvem o registro das imagens pela fotografia ou ainda pela geração de imagens por computador. A animação, porém, não depende exclusivamente da produção de movimento, Rogério Kondo (1997) cita como exemplos as cenas de metamorfose, em que uma imagem gradativamente transforma-se em outra ou em que ocorre mudanças de cores e/ou de intensidade de luz.

Lucena (2005) considera que a origem da animação encontra-se nos dispositivos óptico-mecânicos, sendo o primeiro deles – a lanterna mágica – inventada no século XVII por Athanasius Kircher. A lanterna mágica consistia de uma caixa com uma fonte de luz e um espelho em seu interior que possibilitava a

projeção de slides. Com o uso de um disco giratório, era possível projetar uma série de imagens que tentavam contar uma história ao público. Ao longo do tempo, a lanterna mágica foi sofrendo modificações até permitir uma exibição verdadeiramente animada, o que só ocorreu em 1736. No final do século XVIII, o espetáculo *Fantasmagorie*, produzido por Etienne Gaspard Robert com o uso da lanterna mágica, foi sucesso de público na Europa e nos Estados Unidos. Segundo Lucena (2005), este fato já revelava o fascínio que, quando bem trabalhada artisticamente, a animação consegue exercer sobre o público.

Fig. 2.1 – Modelo de lanterna mágica fabricado em 1900 na França.



Fonte: [http://www.luikerwaal.com/newframe\\_uk.htm?/inh\\_lantaarns\\_uk.htm](http://www.luikerwaal.com/newframe_uk.htm?/inh_lantaarns_uk.htm)

No século seguinte, foram inventados mais alguns dispositivos óptico-mecânicos, tais como o taumatoscópio<sup>10</sup>, o estroboscópio<sup>11</sup> e o zootoscópio<sup>12</sup>. De acordo com Lucena (2005), estes brinquedos ópticos se basearam no princípio da persistência retiniana publicada por Peter Mark Roget em 1824, o qual afirmava que “o olho humano retém uma imagem por uma fração de segundo, enquanto outra imagem está sendo percebida” (LUCENA, 2005, p. 34). Assim, o olho humano tem a tendência de combinar em um único movimento uma série de imagens exibidas em seqüência.

10 Brinquedo constituído de um disco com uma imagem no verso e uma na frente, ao girar o disco, alternando-se as duas faces, as imagens assumem uma única aparência, resultado da mistura óptica.

11 O estroboscópio é o aperfeiçoamento de um outro brinquedo chamado fenaquitoscópio. Consiste de um disco com desenhos pintados em torno do eixo e frestas entre os desenhos. Ao se girar o disco em frente a um espelho e observar este por meio das frestas, tem-se a ilusão de movimento.

12 O zootoscópio consiste de um tambor com frestas laterais, dentro do qual é colocado uma tira com desenhos. Ao se girar o tambor e observar a tira de papel pelas frestas, percebe-se o movimento nos desenhos.

O brinquedo óptico mais popular e o mais utilizado até hoje é, porém, o flipbook, um livrinho com desenhos em seqüência distribuídos em cada uma das páginas. Seu funcionamento é simples, quando as páginas são viradas rapidamente, cria-se a ilusão do movimento. O flipbook ainda é bastante utilizado em produções feitas a partir de desenhos, pois permite visualizar de forma rápida e prática a animação resultante destes desenhos.

Fig. 2.2 – As duas faces de um taumatoscópio, um estroboscópio e um zootrocópio.



Fontes: <http://courses.ncssm.edu/gallery/collections/toys/html/exhibit06.htm> / <http://en.wikipedia.org/wiki/Phenakistoscope/> <http://en.wikipedia.org/wiki/Zoetrope>

Além do flipbook, outro destaque entre os brinquedos ópticos é o praxinoscópio, um aperfeiçoamento do zootrocópio que contava com um jogo de espelhos e lentes capazes de projetar a animação numa tela. O invento de Emile Reynaud foi um grande sucesso de público. Lucena (2005) afirma que foram feitas cerca de 13 mil apresentações do, assim denominado, Teatro Óptico de Reynaud, o qual contava com trilha sonora e com desenhos coloridos.

A grande contribuição tecnológica para o desenvolvimento da animação é, porém, a fotografia. Com o desenvolvimento da película e dos equipamentos necessários para filmar e projetar, a produção de animação encontrou o meio técnico ideal para sua expressão visual e para alcançar maiores públicos.

As primeiras produções desta fase foram realizadas na década de 1890 e avançaram pelas duas primeiras décadas do século XX. Os primeiros filmes utilizavam uma técnica denominada de substituição por parada de ação, a qual Lucena (2005) considera o primeiro passo para a verdadeira técnica de animação, que consiste em fotografar frame-a-frame<sup>13</sup> a animação. A técnica de substituição por

<sup>13</sup> Frame corresponde a cada um dos quadros (imagens) que compõem um filme ou vídeo.

parada de ação era um truque que consistia em filmar uma cena continuamente e, em seguida, parar a ação, substituindo algum elemento da cena para criar a impressão de que o elemento desapareceu, se transformou em outro ou se movimentou sozinho. Era bastante comum o uso de desenhos em conjunto com esta técnica, como nos filmes *Humorous Phases of Funny Faces* e *The Enchanted Drawin*, ambos produzidos por James S. Blackton, nos quais os desenhos parecem ganhar vida e interagir com o desenhista. Lucena (2005) afirma que inicialmente os truques intrigavam bastante a audiência, no entanto, com o tempo, o segredo por trás dos truques dos filmes foram revelados e o público começou a perder o interesse por este tipo de filme.

Após a fase dos *trickfilms* (filmes de efeitos), os animadores voltaram-se para a produção de desenhos animados. Como mencionado anteriormente, os filmes produzidos com essa técnica inicialmente misturavam a performance ao vivo do desenhista com os desenhos animados, um tipo de apresentação que ficou conhecida como *lightning sketches*. Lucena (2005) considera *Fantasmagorie*<sup>14</sup>, de Emile Cohl, lançado no ano de 1908, como o primeiro desenho animado “legítimo”, fotografado *frame a frame* e com características estilísticas bem definidas. Cohl utilizou papel e tinta nanquim para produzir os desenhos desta animação e lançou mão também da caixa de luz, que permitia sobrepor os papéis para poder retraçar com perfeição as partes que não se alteravam.

A caixa de luz amenizava um pouco o cansativo e repetitivo trabalho de desenhar cada um dos quadros que iriam compôr o filme. Além disso, Lucena (2005) afirma que Cohl descobriu que poderia fotografar cada desenho duas vezes, sem ter prejuízos na continuidade do movimento. Essa descoberta reduziu pela metade a quantidade de desenhos necessários para cada segundo de filme e contribuiu para que os animadores pudessem dedicar mais tempo explorando a expressividade e o refinamento estético do movimento.

Nos anos seguintes, segundo Lucena (2005), a animação iniciou um processo de evolução da expressão artística, principalmente pelos trabalhos de Emile Cohl e de Winsor McCay<sup>15</sup>. As possibilidades plásticas da animação e a experimentação

---

14 Apesar de ter o mesmo nome do espetáculo realizado por Etienne Gaspard Robert com o uso da lanterna mágica, o trabalho de Cohl é distinto daquele.

15 Winsor McCay, animador americano, segundo Lucena (2005), teve grandes méritos como o de trazer para a animação um estilo gráfico bastante sofisticado e a aplicação de características de personalidade aos personagens da animação. Entre os trabalhos de McCay, encontram-se *Little Nemo*, lançado em 1911, e *Gertie the Dinosaur*, de 1941.

com conceitos de aceleração/desaceleração e comprimir/esticar são algumas das inovações artísticas trazidas por esses artistas.

Esse processo de evolução artística ganhou continuidade com o desenvolvimento de alguns dispositivos técnicos que permitiram agilizar o trabalhoso processo de produção de animação, sendo o principal destes o desenho em folhas de acetato<sup>16</sup>. Essas inovações no campo tecnológico possibilitaram o desenvolvimento da animação em todo o seu potencial visual – os cenários puderam ser mais bem elaborados, já que não precisariam ser redesenhados milhares de vezes – e, além disso, permitiu também a “industrialização” da animação com o surgimento dos grandes estúdios, já que a produção de animação se tornou mais rápida e barata. Por esses motivos, Lucena (2005) considera o acetato a maior contribuição técnica para a animação tradicional até o advento da computação gráfica.

As possibilidades abertas com o uso do acetato fizeram surgir as séries de personagens, tais como o gato Felix e, posteriormente, Alice Comedies e Mickey Mouse. Segundo Lucena (2005), as séries de personagens atendiam à necessidade de produção em massa. Como cada personagem possuía um universo ficcional específico, cenários podiam ser reaproveitados, bem como poses, movimentos e expressões também eram reutilizadas em todas as edições, economizando, assim, trabalho e tempo. As séries são frutos dos grandes estúdios de animação, dos quais o maior destaque foi a Disney.

Walt Disney, de acordo com Lucena (2005), cumpre um importantíssimo papel na história da animação como o responsável pelo estabelecimento dos conceitos fundamentais da arte da animação. Segundo o autor, em pouco mais de uma década, de 1928 a 1940, Disney trouxe as melhorias necessárias para que a animação passasse a desfrutar do mesmo reconhecimento artístico que o cinema de ação ao vivo. Para conseguir fazer com que a imagem pudesse comunicar de forma convincente, a ponto de se atingir a ilusão da vida, Disney e seus artistas sistematizaram os princípios fundamentais da animação:

Chegaram a doze princípios: comprimir e esticar, antecipação, encenação, animação direta e posição-chave, continuidade e

---

<sup>16</sup> As folhas de celulóide transparente, mais conhecidas no Brasil como acetato, foram importantes por possibilitar a reutilização dos cenários e demais partes do desenho que não se alteravam ao longo da animação, pois cada elemento da cena era desenhado em uma folha separada, evitando, dessa forma, o trabalho de redesenhar os elementos que não se alteravam de um frame para outro.

sobreposição da ação, aceleração e desaceleração, movimento em arco, ação secundária, temporização, exageração, desenho volumétrico, apelo. [...] Com isso, a animação passava a dispor de um conjunto de regras básicas, uma linguagem com sua própria sintaxe, que, adequadamente compreendida e empregada, possibilitava a obtenção de movimentos realisticamente convincentes, uma animação de boa qualidade (LUCENA, 2005, p. 115-116).

Além disso, Disney também estava preocupado com questões de linguagem e roteirização das histórias a serem animadas. Lucena (2005) afirma que Disney considerava a animação como uma arte voltada, prioritariamente, para o entretenimento, assim, procurou conquistar o público através de histórias de grande apelo popular, tais como Branca de Neve e os sete anões, A bela adormecida, e de personagens como Mickey Mouse (Fig. 2.3) e Pato Donald.

No mesmo sentido, Moreno (1978) afirma que:

[Disney] foi pioneiro nas seguintes experiências: o “gag”<sup>17</sup> visual exato, coordenação com música, a introdução do colorido baseado nos processos iniciais do technicolor, introdução dos valores tonais nos desenhos, o desenho de longa-metragem e o som estereofônico. (MORENO, 1978, p. 53)

Fig. 2.3 – Mickey, personagem dos estúdios Disney.



Fonte: <http://www.eb1-lapaducos.rcts.pt/>

Neste momento histórico, surgem também outros estúdios de animação com propostas estéticas alternativas à Disney, como a UPA, a Warner Brothers, a MGM e o estúdio dos irmãos Fleischer. Os artistas desses estúdios partiam dos princípios artísticos estabelecidos por Disney, porém os exploravam de maneira diferente.

Ou seja, os mesmos princípios que serviam para obter movimentos e ações dramaticamente convincentes, se explorados em seus extremos (como muitas vezes se percebia nos próprios filmes de Disney), com distorções e exagerações nos limites, poderiam conseguir efeitos cômicos estridentes em animações mais descompromissadas

---

17 O termo gag pode ser entendido como piada.

tematicamente. (LUCENA, 2005, p. 124).

Assim, surgem personagens cômicos que marcaram a história da animação, tais como Pernalonga, Patolino, Tom e Jerry, Coiote, Papaléguas e o Diabo da Tasmânia. Como podemos observar pelas Fig. 2.4 e Fig. 2.5, tais personagens possuem traços mais exagerados que fogem à simetria de personagens como Mickey Mouse (Fig. 2.3) , de Walt Disney.

Fig. 2.4 – Patolino, personagem da Warner Brothers



Fonte: [http://www.universohq.com/quadrinhos/2004/idiotas\\_nas\\_hqs2.cfm](http://www.universohq.com/quadrinhos/2004/idiotas_nas_hqs2.cfm)

Fig. 2.5 – O Diabo da Tasmânia e Pernalonga, personagens da Warner Brothers



Fonte: [http://www.universohq.com/quadrinhos/2004/idiotas\\_nas\\_hqs2.cfm](http://www.universohq.com/quadrinhos/2004/idiotas_nas_hqs2.cfm)

A fase seguinte da história da animação será marcada pela introdução da computação gráfica no seu processo de produção. Os primeiros esforços para a geração de imagens animadas por computador vão ser feitos a partir da década de 1970, porém, segundo Lucena (2005), as primeiras produções com verdadeira qualidade artística só vão surgir na década de 80. É nessa década que amadurecem os softwares e hardwares necessários para a produção digital de animação e estes se tornam acessíveis aos artistas, os quais irão transpor para a computação 3D os princípios artísticos da animação tradicional.

Vemos, portanto, que são muitas as inovações tecnológicas que interferiram

na forma de produção da animação. Essas mudanças na esfera tecnológica têm sua importância no momento em que provocam também melhorias à expressividade artística da animação.

## 2.2 Caracterização e conceituação da animação

A animação se diferencia do cinema de ação ao vivo pela forma de produção das imagens. Segundo Moreno (1978), o cinema de ação ao vivo se caracteriza pela tomada ininterrupta de imagens, seja de uma paisagem real ou de atores que representam uma cena. O cinema de animação, ao contrário, é concebido numa dimensão de irrealidade e de descontinuidade, ou seja, busca-se produzir aquilo que não é possível de ser executado no mundo real. Contudo, a intenção é de transmitir ao espectador a ilusão da vida, fazendo-o acreditar nas imagens que passam pelos seus olhos.

A animação também difere do cinema de ação ao vivo pela forma de captação das imagens. Ao invés da filmagem direta, numa animação as imagens são captadas através de fotografia quadro-a-quadro ou geradas por meio de algum dispositivo técnico, tais como computadores.

Moreno (1978) também define alguns métodos de produção de animação, especificaremos agora os principais destes:

- Animação de bonecos ou marionetes: os bonecos ou marionetes são fotografados quadro-a-quadro dentro de um cenário, o qual geralmente é uma maquete. A manipulação dos elementos em cena precisa ser feita com bastante cuidado para conseguir se obter uma animação de boa qualidade.

- Animação de pessoas (pixillation) – consiste em fotografar quadro-a-quadro atores nos devidos movimentos que se queira dar. O filme “Os Vizinhos” de Norman McLaren é um dos exemplos mais representativos das possibilidades artísticas desta técnica.

- Animação de objetos – Basicamente a mesma técnica, porém aplicada a objetos. De acordo com Moreno (1978), o filme “Renascimento”, de Walerian Borowczyk, é um exemplo de bom uso desta técnica. Este possui cenas de livros sendo rasgados aos poucos, objetos mudando de lugar e até explosões.

- Animação com carvão ou com massa de modelar – nesta técnica, cria-se

um desenho ou uma forma e fotografa-se cada etapa do desenho ou movimento feito com as formas. Esta mesma técnica já foi também utilizada para produzir animações com areia. Lucena (2005) denomina essa técnica como pintura-no-tempo, desde que se utilize desenhos ou pinturas.

- Animação de recortes – essa técnica utiliza personagens ou bonecos feitos a partir de recortes e montados de forma a permitir a articulação dos corpos dos mesmos. Os personagens ou objetos são então fotografados minuciosamente quadro-a-quadro em cada uma das etapas do movimento.

- Animação direto na película – com este método, os desenhos são realizados diretamente na película, sem necessidade do uso de uma câmera. Para isso, retira-se, anteriormente, a emulsão da película. Segundo Lucena (2005), “As imagens e o movimento assim obtidos são dotados de uma fluência e de uma continuidade impossíveis de conseguir por outros métodos” (p. 93), porém a pequena área de uma película de 35 mm não permite a produção de desenhos mais complexos. Norman McLaren foi um dos animadores com as mais significativas experiências com o uso dessa técnica.

- Desenho animado – um dos métodos de animação mais largamente utilizados é o desenho animado, inicialmente os desenhos eram feitos em papel e fotografados em película cinematográfica. Com o tempo, surgiram recursos que facilitaram o trabalho de animação, como o acetado, o qual já foi descrito anteriormente, e também recursos digitais.

- Desenho animado com filme de ação ao vivo – consiste em adicionar um ator real sobre uma animação ou, ao contrário, um personagem animado sobre um filme de ação ao vivo.

Esses métodos de animação não devem ser compreendidos como categorias isoladas, mas sim como recursos que podem ser combinados entre si. Além desses métodos definidos por Antônio Moreno, existem os métodos de animação digitais, os quais serão abordados no tópico seguinte.

## **2.3 Animação digital**

A utilização de computadores na produção de animação é a etapa mais recente da história desta arte. Através dos computadores, tornou-se possível

produzir imagens em três dimensões que simulem a realidade. No entanto, conforme veremos, foram necessários muitos anos de pesquisas científicas para a tecnologia tornar-se apta para a realização de tal missão.

### 2.3.1 A evolução da computação gráfica

A animação por computador só foi possível com o desenvolvimento de tecnologias que permitiram o trabalho com gráficos de forma interativa. Tais tecnologias começaram a ganhar dimensão e complexidade na década de 1970. Como vimos no primeiro capítulo, é nesta década que surge o microprocessador, iniciando um ciclo de evolução rápida e constante da capacidade de processamento dos computadores, e também o computador pessoal, o qual tornou a tecnologia mais acessível ao público geral.

Lucena (2005) afirma que a interatividade entre usuário e computador foi a característica responsável pelo sucesso da informática. Além disso, a interatividade deveria se dar da forma mais intuitiva possível e, para isso, a melhor escolha era o uso de imagens. Os avanços em termos de capacidade de processamento e memória ocorridos na década de 1970, tornou possível a interatividade de forma gráfica. Um passo crucial para isso foi o surgimento do *frame buffer*, uma memória especializada no armazenamento temporário dos dados visuais. Assim:

Com o *frame buffer*, havia possibilidade de armazenamento e manipulação de imagens em alta velocidade, trabalhando com vasta gama de cores, opções de iluminação e textura. A ilustração e a pintura digital podiam agora acontecer; já os gráficos aramados 3D teriam a chance de ser recobertos por superfícies sólidas realistas. Sem esse dispositivo, os terminais de vídeo do tipo varredura não se teriam disseminado – por conseqüência, impedindo a popularização da informática. (LUCENA, 2005, p. 276).

Além disso, o *frame buffer* também fez avançar as interfaces gráficas, impulsionando o uso de dispositivos como o *mouse*<sup>18</sup> e a mesa digitalizadora (*tablet*<sup>19</sup>), importantes para a popularização da informática e para o trabalho com animação digital.

Posteriormente, começaram a ser desenvolvidos hardwares voltados

---

18 Segundo Lucena (2005), o *mouse* foi inventado no começo dos anos 1960 por Douglas C. Engelbart, porém inicialmente este invento foi tratado com desdém e passou a ser valorizado somente na década de 1970.

19 O *tablet* é um dispositivo constituído por uma caneta e uma superfície plana com o intuito de permitir ao usuário desenhar diretamente no computador da maneira mais próxima possível ao desenho tradicional.

especialmente para o trabalho com computação gráfica, setor no qual a empresa Silicon Graphics Inc., fundada em 1982, obteve grande destaque. Além disso, foram desenvolvidas bibliotecas gráficas<sup>20</sup> e utilizada a tecnologia de *clusters*<sup>21</sup>. Todos estes recursos foram avanços significativos para a geração de imagens tridimensionais em computador.

Ao longo dos anos, novos aperfeiçoamentos no hardware, em paralelo com inovações na forma de trabalho dos softwares, continuaram a ocorrer, trazendo a possibilidade de expressão artística por meio da animação digital. Nos tópicos seguintes, trataremos do processo de animação digital e, em paralelo, forneceremos mais informações acerca da evolução histórica da computação gráfica.

### 2.3.2 O processo de animação digital

A animação digital pode ser realizada por meio de três métodos principais, os quais podem ou não envolver também o uso de técnicas de animação tradicionais, que foram descritas no início deste capítulo. Esses três métodos são: animação *frame-a-frame* (ou quadro-a-quadro), animação por *keyframe* e animação tridimensional. Os dois primeiros métodos são classificados como “animação auxiliada por computador”. Já o terceiro é denominado como “animação modelada por computador”, de acordo com Camargo (1995).

O método de animação *frame-a-frame*, também conhecido por *stopmotion*, compreende animações cujas imagens são capturadas por uma câmera ou *scanner* ou produzidas digitalmente, desde que quadro-a-quadro. Esse é o método mais simples de animação digital, visto que, além de ser utilizado como plataforma de desenho ou como meio de captura de imagens, o computador vai apenas seqüenciar os quadros, inserir trilha sonora e gerar um arquivo de vídeo com a animação.

Um outro método é a animação por *keyframe*. Lucena (2005) afirma que, na animação tradicional, é comum o trabalho de desenhar as cenas ser dividido entre o animador-chefe, o qual desenhava as posições principais (*key position*), e o

---

20 A biblioteca gráfica, entre outras funções, faz a intermediação entre o software e a placa de vídeo, facilitando, dessa forma, o trabalho de desenvolvimento de softwares com recursos gráficos tridimensionais. A OpenGL, inicialmente desenvolvida pela Silicon Graphics Inc. é uma biblioteca gráfica licenciada sob a GNU GPL e largamente utilizada por diversos softwares livres.

21 Um cluster consiste de vários computadores ligados em rede trabalhando como uma única máquina, com isso é possível obter um aumento significativo na capacidade de processamento.

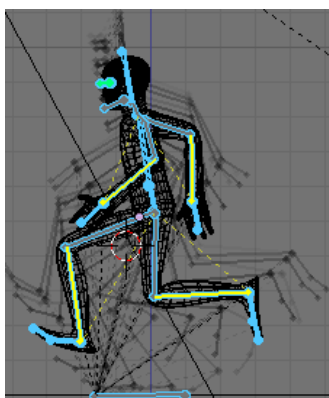
animador-assistente, que fazia os desenhos intermediários (*inbetween frames*) necessários ao complemento do movimento entre as posições principais de uma seqüência de ação. Essa divisão de trabalho permite um maior controle sobre a qualidade da forma e do movimento, já que o animador-chefe, por não precisar gastar seu tempo e esforço com o trabalho repetitivo de desenhar todos os quadros intermediários do filme. De qualquer forma, a carga de trabalho no método de animação tradicional ainda é muito grande.

O computador entra em cena justamente para assumir o trabalho de geração dos desenhos intermediários, os quais representam a maior parte do trabalho em uma animação. Portanto:

Definidos os *keyframes* (posições chaves) pelo animador e os demais parâmetros associados ao objeto da animação e a seqüência a ser produzida, o computador, através de um algoritmo de interpolação, calcula as posições do objeto no espaço procedendo às mudanças de forma que atendam aos parâmetros de um *keyframe* a outro. (LUCENA, 2005, p. 314-315).

De acordo com Lucena (2005), os *keyframes* podem ser combinados com o uso de esqueletos. Os esqueletos são linhas conectadas entre si, tais como os ossos de um organismo, que controlam o posicionamento e a ação local ou global na imagem que foi associada a eles. Desta forma, é possível coordenar movimentos mais complexos de forma mais eficiente, sem a necessidade de o animador fazer muitas alterações nos desenhos nem aumentar a quantidade de *keyframes*. Na Fig. 2.6, podemos visualizar um esqueleto associado à um personagem no software Blender.

Fig. 2.6 – Personagem sendo animado através de controle por esqueleto e *keyframes*



Fonte: [http://wiki.blender.org/index.php/Manual/Armature\\_Objects](http://wiki.blender.org/index.php/Manual/Armature_Objects)

A técnica de animação por *keyframes* é bastante utilizada na produção de animações bidimensionais (2D), porém os softwares de animação 3D também fazem uso do algoritmo de interpolação de quadros.

Além do método de animação *frame-a-frame* e da animação por *keyframe*, temos a animação tridimensional (3D). Este último método tem como intuito a geração em computador de imagens realistas, ou seja, trata-se de recriar a realidade por meio da computação gráfica, porém não necessariamente mantendo todos os aspectos do mundo real idênticos. No próximo tópico, abordaremos em detalhes o ambiente de animação tridimensional.

### **2.3.3 O ambiente de animação tridimensional**

Por ter como intuito recriar a realidade através da geração de imagens tridimensionais, uma animação 3D envolve uma série de etapas, as quais exigiram, e ainda hoje exigem de cientistas e programadores soluções de alta complexidade tecnológica. Basicamente, “Os softwares de animação 3D trabalham baseados em representações geométricas vetoriais, ou seja, algoritmos matemáticos que descrevem sólidos no espaço e que podem ser convertidos em uma visualização animada” (WERNECK, 2005, p. 79). Ainda de acordo com Werneck (2005), os elementos básicos de uma animação são os sólidos geométricos, a luz e a câmera.

Porém, para se obter imagens com alto nível de realismo, foi necessário avançar além destes elementos básicos, daí surgiram soluções como o mapeamento de textura, diversos métodos de iluminação e modelagem, entre outros. Iremos nos sub-tópicos seguintes detalhar os principais recursos que compõem um ambiente de animação tridimensional.

#### **2.3.3.1 Modelagem**

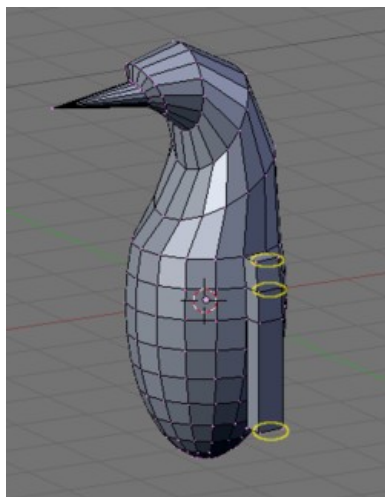
Segundo Lucena (2005), a modelagem é a primeira etapa para a simulação da imagem computadorizada. É nesse estágio que o artista, tal qual um escultor, irá dar forma aos personagens, objetos e cenários que farão parte da animação. Inicialmente, o maior desafio dos ambientes de animação era permitir a manipulação de dados geométricos de forma interativa, proporcionando ao artista facilidade e resposta em tempo real. Com os avanços tecnológicos da década de 1970, porém,

esse desafio foi vencido.

É também nessa década que, segundo Lucena (2005), são desenvolvidos ou aperfeiçoados os principais métodos de modelagem 3D que são utilizados hoje, tais como representação por primitivas, geometria sólida construtiva, modelagem de forma livre, modelagem por procedimento, técnicas de extrusão, revolução, seção transversa serial, entre outras. Descreveremos a seguir os principais métodos definidos pelo autor.

A modelagem por primitivas geométricas disponibiliza ao usuário um conjunto de objetos simples tanto em duas como em três dimensões. As opções variam de acordo com os softwares, porém os objetos mais comuns são cubos, esferas, cones, cilindros, círculos e quadrados. Partindo dessas formas, os artistas podem combiná-las, movimentá-las pelo espaço tridimensional, realizar operações como cortar, esticar, duplicar, espelhar e torcer até atingir a forma desejada. Na Fig. 2.7, temos, como exemplo, uma esfera cuja parte superior foi modificada para dar forma à cabeça de um pingüim.

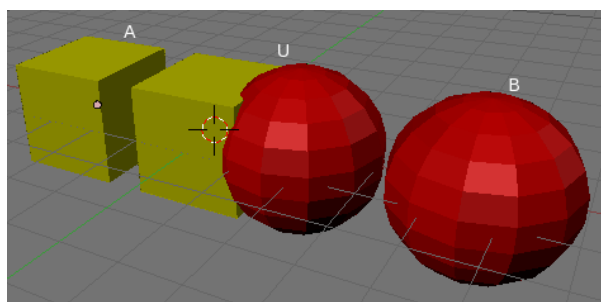
Fig. 2.7 – Pingüim sendo modelado a partir de uma esfera



Fonte: [http://en.wikibooks.org/wiki/Blender\\_3D:\\_Noob\\_to\\_Pro/Penguins\\_from\\_spheres](http://en.wikibooks.org/wiki/Blender_3D:_Noob_to_Pro/Penguins_from_spheres)

A modelagem sólida, também chamada em inglês de *costrutive solid geometry* ou CSG, utiliza como base as primitivas geométricas, porém com o intuito de permitir que as primitivas interajam por meio de operações booleanas. Em outras palavras, na modelagem sólida, pode ser aplicada uma operação de união, intersecção ou diferença entre duas ou mais formas. Na Fig. 2.8, por exemplo, os objetos A e B foram unidos num único objeto U.

Fig. 2.8 – Operação booleana de união entre um cubo e uma esfera

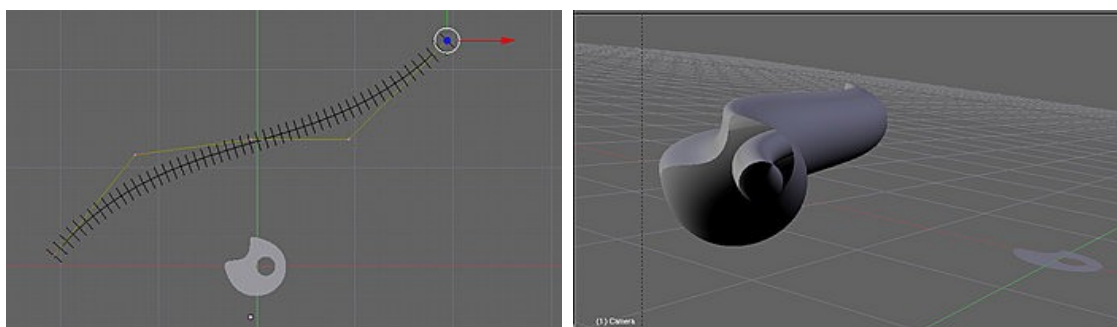


Fonte: <http://wiki.blender.org/index.php/Manual/Booleans>

A modelagem de forma simples, segundo Lucena (2005), é um método de modelagem bastante demorado e geralmente é aplicado apenas em ajustes finos na estrutura de um modelo. Esse método permite fazer uma alteração localizada na forma através dos pontos de controles dos fragmentos que compõem a estrutura de uma forma.

Já a modelagem por derivação é constituída por algumas técnicas, as quais envolvem procedimentos simples que, no entanto, produzem efeitos de grande relevância. Tais procedimentos utilizam uma ou mais formas bidimensionais, as quais, ao serem deslocadas, geram um modelo com três dimensões. As técnicas mais populares de modelagem por derivação são a extrusão, revolução e seção transversa serial.

Fig. 2.9 Extrusão de uma forma ao longo de um eixo

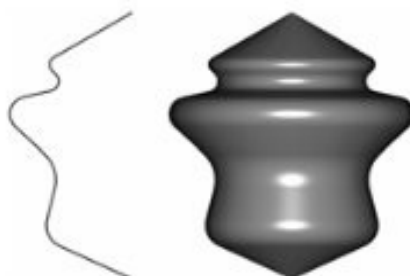


Fonte: <http://blenderartists.org/forum/showthread.php?t=72006>

Com a extrusão, uma cópia da forma bidimensional é deslocada ao longo de um eixo e, em seguida, o software conecta as duas faces, obtendo, dessa forma, um modelo tridimensional. Na Fig. 2.9, podemos ver no lado esquerdo uma forma

bidimensional e o caminho por onde esta foi deslocada para dar origem ao objeto 3D que vemos no lado direito da imagem.

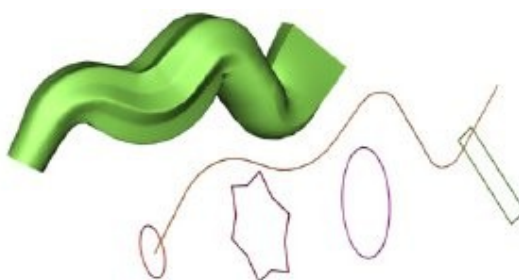
Fig. 2.10 – Objeto produzido pela técnica de revolução



Fonte: MELO & SILVEIRA, 2007, p. 03.

Já a revolução realiza um processo equivalente ao que o torno mecânico e a roda de oleiro fazem no mundo real. “A partir de uma forma bidimensional, que pode ser aberta ou fechada, o programa efetua um giro completo de em torno do seu próprio eixo vertical” (LUCENA, 2005, p. 292). Na Fig. 2.10, temos um objeto tridimensional produzido a partir do giro de uma linha em torno de seu próprio eixo.

Fig. 2.11 – Modelo produzido por seção transversal serial



Fonte: [http://www.ceart.udesc.br/revista\\_dapesquisa/volume2/numero2/design/Walter.pdf](http://www.ceart.udesc.br/revista_dapesquisa/volume2/numero2/design/Walter.pdf)

Por fim, a seção transversal serial, uma variação do conceito de extrusão, permite a modelagem de formas naturais através da geração de fatias (seções) dispostas em torno de um caminho em linha reta ou curva. Esta técnica é bastante útil para a modelagem de formas menos regulares, tais como corpos de animais e montanhas. Na Fig. 2.11 podemos visualizar um modelo produzido através da modelagem por seção transversal serial.

Além da modelagem por derivação, temos também a modelagem por procedimento.

Esse método se baseia na noção de que a informação pode ser gerada por um processo dinâmico e não-linear sempre que se queira, em oposição à mera recuperação passiva de dados. [...] Resumindo, a descrição de um modelo por procedimento não trata de o esculpir geometricamente, mas antes, definir seu comportamento. Alterando simplesmente as regras desse comportamento, ativa-se uma cadeia complexa de transformações generalizadas, praticamente impossível de criar por outras técnicas de modelagem digital. (LUCENA, 2005, p. 293-294).

Uma grande utilidade para o emprego dessa técnica está na produção de fenômenos naturais, tais como chuva, fumaça, tornados, poeira e vapor, e outros fenômenos como explosões. É nesse método de modelagem que se baseia o sistema de geração de partículas dos softwares atuais.

Como podemos ver, a modelagem é a base da animação 3D e uma das etapas de maior complexidade para o artista no processo de produção de uma animação. Os recursos de modelagem e a eficiência na interação com o usuário são pré-requisitos básicos para qualquer software de animação tridimensional.

### **2.3.3.2 Iluminação**

Numa animação tridimensional, a iluminação assume um papel extremamente importante. A aplicação de técnicas adequadas de iluminação é imprescindível para garantir uma aparência realista às imagens. Os softwares atuais permitem a inserção de diversas fontes de luz que visam simular todas as fontes de luz que temos no mundo real, como as lâmpadas e a iluminação solar. Além disso, os programas utilizam alguns métodos de iluminação, os quais determinam a maneira como o cálculo de iluminação é realizado.

Segundo Lucena (2005), desde a década de 1980, o método mais avançado de iluminação é o *Ray Tracing*, cuja maior inovação é considerar, no cálculo da iluminação, a influência de todos os elementos em cena. “Os métodos vistos anteriormente (Lambert, Gourard e Phong) se valiam do cálculo local da luz, procedendo à iluminação dos objetos como se eles estivessem sozinhos no espaço” (LUCENA, 2005, p. 362)

A idéia base do *ray tracing* é seguir o caminho dos raios de luz desde a fonte até os olhos do espectador registrando as características dos objetos atingidos no percurso. Assim, o *ray tracing* considera informações como textura, reflexão, refração, transparência, sombra e cor dos objetos no cálculo da luminosidade.

Lucena (2005) afirma que, para diminuir o volume de cálculo exigido pelo *ray tracing*, a análise da difusão da luz entre os objetos foi suprimido do método. Contudo, a capacidade de processamento exigida pelo *ray tracing* é, ainda hoje, muito grande, de modo que este método foi pouco aplicado na década 1980. Em 1984, pesquisadores desenvolveram o método *radiosity* (radiosidade) com o objetivo de suprir o recurso que havia sido retirado do *ray tracing*. “A radiosidade modela com perfeição sutilezas da iluminação, como passagens suaves nas áreas de sombra, delicadas fusões de cores, áreas adjacentes à fonte de luz e, claro, a iluminação indireta entre os elementos de cena” (LUCENA, 2005, p. 365).

O segredo por trás do método de radiosidade é aplicar à superfície dos objetos em cena uma malha de polígonos (também chamado *patches*) e calcular a distância e a angulosidade entre os polígonos. Quando dois polígonos estão localizados paralelamente, há uma maior transferência de energia luminosa entre eles. Já quando estes estão localizados perpendicularmente, menos energia é transferida. A luz que não for completamente absorvida pela superfície de um objeto é refletida, se necessário, sucessivas vezes até ser completamente dissipada no espaço.

A quantidade final de luz de um polígono específico é dada pela porcentagem de iluminação que o atinge vinda de todos os outros polígonos – no caso de ser também fonte de luz, a quantidade de luz que ele naturalmente emite é adicionada. (LUCENA, 2005, p. 365).

Uma limitação do método de radiosidade, porém, é o fato de não tratar a transparência dos objetos nem a reflexão especular. Por isso, posteriormente, pesquisadores trabalharam para conseguir unir a radiosidade com o método de *ray tracing*.

Lucena (2005) afirma que o sucesso de produções cinematográficas com alto nível de realismo nos gráficos produzidos em computador se deve em grande parte à alta qualidade dos algoritmos de modelagem luminosa.

### **2.3.3.3 Textura**

A textura, assim como a iluminação, é um recurso de grande importância para se atingir o realismo gráfico. De acordo com Lucena (2005), inicialmente, a texturização era realizada através da associação de uma imagem do tipo bitmap à superfície de um objeto ou personagem. Além disso, agregava-se também a

informação dos valores de iluminação de cada ponto da imagem. Apesar de já representar um avanço na busca pelo realismo, os objetos ainda aparentavam possuírem uma superfície invariavelmente plana, a qual comprometia a percepção de suas características materiais.

Posteriormente foram desenvolvidos importantes algoritmos capazes de simular a rugosidade das texturas (*bump mapping*), o efeito de substâncias nebulosas, névoa por dispersão de partículas (*particle scattering*) e a simulação do ambiente circundante (*environment mapping*).

James Blinn, através do uso das descobertas da física experimental a respeito de como as superfícies refletem a luz, desenvolveu o algoritmo do *bump mapping*, o qual cria a ilusão de rugosidade na superfície dos objetos sem a necessidade de alterar a geometria do objeto. O *bump mapping* apenas altera a direção com que os raios de luz são refletidos.

Já para o algoritmo de mapeamento do ambiente circundante (*environment mapping*), a solução foi imaginar que o objeto estivesse envolvido por uma grande esfera. Dentro dessa esfera estaria uma imagem do ambiente que circundava o objeto. Bastava então projetar esta imagem esférica sobre o objeto para criar a simulação do ambiente circundante.

Por fim, para simular o efeito de névoa e de outras substâncias nebulosas, James Blinn distribuiu, aleatoriamente, por todo o ambiente da cena, uma infinidade de partículas (esferas minúsculas). Essas partículas tinham a propriedade de absorver e refletir luz em qualquer direção. “Quanto mais densa a nuvem de partículas, menores as chances de a luz atingir o observador” (LUCENA, 2005, p. 302).

Lucena (2005) afirma que, com o desenvolvimento desses algoritmos de mapeamento de textura, estava finalizado o ciclo das tecnologias fundamentais para geração de imagens estáticas utilizadas pelos pacotes gráficos 3D comercializados durante as décadas de 1980 e 1990.

#### **2.3.3.4 Animação**

Apesar dos grandes avanços registrados no quesito geração de imagem digital, Lucena (2005) avalia que, até o final da década de 1970, a geração de movimento – essência da animação – ainda era insuficiente para a animação de

formas mais complexas.

No início dos anos 1970, foram desenvolvidas as técnicas de animação digital por *keyframe* e a de controle por esqueleto, já citadas no início deste capítulo, e que, segundo Lucena (2005), ganharam bastante repercussão na área de animação computadorizada e são, ainda hoje, um dos recursos mais importantes para a geração de movimentos. O próximo passo é dado por Mark Levoy, pesquisador da Cornell University que, no ano de 1977, conseguiu unir a tecnologia de animação por *keyframe* e a de pintura de imagens num ambiente gráfico capaz de animar múltiplos planos de uma cena separadamente.

O sistema desenvolvido por Levoy permitia que o animador visualizasse em tempo real seu trabalho tanto em formato vetorial quanto em varredura, com a imagem já pintada. No entanto, segundo Lucena (2005), o maior mérito desse sistema é mesmo a capacidade de animar elementos em múltiplos planos. A utilidade desse recurso é permitir que os elementos de uma cena pudessem ser divididos em planos. Assim, cada um desses planos poderia ter seus parâmetros alterados de forma independente.

A posição do observador imaginário podia ser alterada em qualquer direção e cada plano podia ser movido, rotacionado e graduado, de maneira a estabelecer a correta relação de distância entre as imagens da composição, garantindo a precisão dos movimentos e a ilusão tridimensional. [...] Mesmo trabalhando basicamente com imagens em duas dimensões, o sistema podia combinar animação plana dentro de cenários tridimensionais num arranjo espacial por retroprojeção – cuja precisão perspectiva era assegurada pelos ajustes que o esquema de múltiplos planos oferecia. A sofisticação do conceito de planos múltiplos constitui a base de respeitados sistemas profissionais da atualidade. (LUCENA, 2005, p. 317-318).

Já na década de 1980, são desenvolvidas duas técnicas de grande importância para a animação 3D: a técnica de modelagem e animação por procedimento (*procedural techniques*) e a cinemática inversa.

A animação por procedimento, de acordo com Lucena (2005), funciona a partir de algoritmos, os quais são baseados em leis físicas e regras biológicas, permitindo assim simular propriedades complexas de fenômenos naturais, tais como gravidade, atrito, aceleração/desaceleração, compressão/estiramento, movimento secundário, entre outros. Além disso, a modelagem e animação por procedimento é bastante utilizada para a geração de cabelos, pêlos e roupas dos personagens.

O autor também afirma que esta técnica pode ser considerada como a transposição para a animação digital dos princípios fundamentais da animação que

havam sido definidos há algumas décadas por Disney. A animação e a modelagem por procedimento exigem, em seu desenvolvimento, algoritmos bastante complexos, no entanto oferecem ao animador mais agilidade e resultados bastante difíceis de se conseguir sem o uso desta técnica.

Já a cinemática inversa (*inverse kinematics*) pode ser considerada como um aperfeiçoamento da técnica de animação por *keyframe* e de controle por esqueleto. “Com a cinemática inversa, em vez de especificar explicitamente as posições do modelo, o animador determina o movimento com base em instruções implícitas ao sistema” (LUCENA, 2005. p. 376). A animação por *keyframe*/controle de esqueleto exige que, por exemplo, para se movimentar o braço de um personagem em direção a um objeto, seja determinada a posição de cada uma das articulações (ombro, cotovelo, pulso e dedo). Já com a cinemática inversa, é necessário apenas deslocar o dedo do personagem até o objeto, pois o computador já possui as instruções de hierarquia da estrutura que compõe o esqueleto e da restrição de movimento de cada parte deste. “De posse dessas regras, o computador calcula automaticamente como as conexões devem movimentar-se, de modo que a estrutura apresente uma solução final (uma posição aceitável) que faça sentido” (LUCENA, 2005, p. 377). De qualquer forma, a intervenção do animador é ainda necessária para corrigir determinadas posições que se apresentam como anti-naturais.

Um outro método importante é a animação com curvas. De acordo com Patmore (2003), tal método é bastante útil para animação de objetos inanimados, a exemplo de uma bola saltando. Com o uso deste método, o animador simplesmente define um caminho pelo qual o objeto irá se deslocar. Além disso, é possível também fazer com que os objetos respeitem as leis da física, obtendo, desta forma, movimentos mais realistas.

Assim, vemos que a etapa de animação envolve um conjunto de complexos algoritmos, os quais demandaram bastante tempo para serem desenvolvidos. A aplicação de cada um destes métodos de animação irá variar conforme o tipo de movimento que se queira produzir.

### **2.3.3.5 Renderização**

A renderização é a última fase do processo de animação. De acordo com Patmore (2003), apesar de ser uma fase essencialmente técnica, ultimamente tem

sido considerada com uma das mais importantes no processo de animação. A renderização exige bastante poder de processamento dos computadores, pois milhares de operações matemáticas são necessárias para gerar cada quadro da animação. Nesta fase são calculadas as posições dos personagens e objetos, luzes e câmera ao longo do tempo, bem como os cálculos de difusão e refração da luz, os quais são de extrema importância para o resultado gráfico do trabalho. Por tais cálculos exigirem bastante dos computadores e, por conseqüência, demandarem muito tempo, é comum a utilização de *render farms*, um conjunto de computadores ligados em rede que dividem entre si o trabalho de renderização.

Apesar de todos os principais sistemas de animação da atualidade já possuírem renderizadores embutidos, é cada vez mais comum o processo de renderização ser realizado em conjunto com um software específico para essa função.

Em linhas gerais, são estes os principais recursos necessários a um ambiente digital de produção de animação 3D. Descreveremos em seguida os principais softwares comerciais desenvolvidos para o trabalho com animação tridimensional.

#### **2.3.4 Os principais sistemas comerciais de animação 3D**

De acordo com Lucena (2005), é em meados da década de 1980 que surgem as primeiras *software houses*<sup>22</sup> voltadas para o desenvolvimento de programas de modelagem e animação 3D e é nessa época também que ocorre uma expansão da animação digital pelo mundo. Porém, os softwares para animação 3D com maior espectro de recursos eram suportados apenas por grandes computadores e, como estes tinham um custo bastante elevado, eram acessíveis apenas aos estúdios de animação. Dessa forma, os animadores independentes e/ou amadores, inicialmente, tinham como opção utilizar softwares bem menos sofisticados e que rodavam nos microcomputadores da época.

No entanto, como dito no primeiro capítulo, o poder de processamento dos computadores passou a aumentar constantemente e a tecnologia foi tornando-se cada vez mais acessível. De forma que hoje é bastante sutil a diferença entre os recursos de hardware que um grande estúdio de animação pode contar e aqueles

---

<sup>22</sup> *Software houses* é um termo comumente utilizado para denominar grandes empresas de produção de softwares.

que um animador independente tem acesso. No entanto, o custo dos softwares, como veremos a seguir, impedem essa igualdade de condições.

Basicamente, os softwares de animação tridimensional são utilizados em três áreas principais: produções cinematográficas, publicidade e mercado doméstico. Na área cinematográfica, são exigidos dos softwares, prioritariamente, recursos de modelagem e de animação de personagens. Já no mercado publicitário, o foco está na criação de efeitos visuais e de animações que, geralmente, não são tão complexas quanto às do mercado cinematográfico. A publicidade exige softwares com soluções mais rápidas, em decorrência do ritmo de produção deste mercado. Por fim, no mercado doméstico, a característica mais exigida dos softwares é a facilidade no aprendizado, o que implica em interfaces simples e intuitivas.

De acordo com Patmore (2003) os principais softwares comerciais para produção de animações 3D, de nível profissional, são o 3ds Max e o Maya, produzidos pela Autodesk<sup>23</sup>, o Softimage XSI<sup>24</sup>, da Softimage, o LightWave<sup>25</sup>, da NewTek e o Cinema 4D<sup>26</sup>, da Maxon. Todos estes softwares trazem recursos de modelagem, animação e renderização de gráficos tridimensionais, porém cada um deles prioriza certos recursos em detrimento de outros.

O Maya, de acordo com o site da empresa Autodesk, é voltado para criação de animações 3D para cinema e vídeo, contando com recursos avançados para animação e modelagem de personagens e cenários. Já o 3ds Max tem o foco na criação de efeitos visuais, na visualização de imagens estáticas (principalmente para as áreas de arquitetura e design) e no desenvolvimento de jogos eletrônicos em 3D, os quais também necessitam de recursos de animação.

Por sua vez, o Softimage XSI se aproxima mais da proposta do Maya, oferecendo recursos avançados para produção de animação 3D para cinema, televisão e jogos eletrônicos. Já a proposta do LightWave, também de acordo com o site oficial, é trazer uma interface intuitiva para produção de efeitos visuais e de animações. O Cinema 4D também traz uma proposta parecida. Segundo o site oficial, o software é um sistema de animação 3D de nível profissional de fácil utilização. Para isso, o software é estruturado de forma modular: o núcleo do programa traz apenas o essencial e o usuário adiciona módulos conforme suas

---

23 <http://www.autodesk.com/>

24 <http://www.softimage.com/>

25 <http://www.newtek.com/>

26 <http://www.maxon.net/>

necessidades.

Se, como abordado no início deste tópico o acesso ao hardware já se encontra relativamente democratizado, o preço das licenças de softwares pode ser considerado como uma barreira que impede o acesso dos animadores independentes aos recursos de animação digital. O custo da licença dos principais softwares de nível profissional variam de 895 (LightWave) até 4995 dólares (Maya e Softimage XSI), sendo necessária a compra de uma nova licença a cada nova versão do programa. Muitos estúdios de animação, a exemplo da DreamWorks, possuem equipes dedicadas exclusivamente ao desenvolvimento dos softwares necessários ao processo de produção dos filmes de animação<sup>27</sup>.

Nesse contexto, o software livre assume um papel de extrema importância por possibilitar aos animadores independentes acesso a ferramentas de computação gráfica sem custos com licenças. Além disso, pelo fato de o código-fonte estar disponível para modificações, o usuário tem a possibilidade de acrescentar novas funcionalidades, caso tenha o conhecimento necessário para executar tal tarefa, ou de sugerir modificações ao mantenedor do software. O compartilhamento de soluções entre os usuários pode representar uma boa estratégia de crescimento dos produtores independentes em relação aos grandes estúdios de animação, da mesma forma que o software livre, através do trabalho colaborativo de milhares de pessoas, já impôs sua qualidade frente às empresas de software proprietário.

Vemos, portanto, que a trajetória da animação é marcada por inovações tecnológicas que interferem na linguagem e na expressividade artística da animação. O surgimento de um novo dispositivo tecnológico, porém, não implica a superação dos anteriores, mas sim o estabelecimento de novas possibilidades expressivas e de maior praticidade no processo de produção. Nesse sentido, as principais possibilidades abertas pela animação digital são a produção de imagens realistas e a eliminação do trabalho repetitivo de redesenhar milhares de quadros. Por fim, o software livre pode representar um avanço em direção à democratização das ferramentas de produção de animação digital.

---

27 ROWE, 2007.

### 3. Blender: descrição de recursos e modelo de desenvolvimento

O Blender pode ser considerado um caso especial no atual cenário de software livre. Voltado para modelagem e animação tridimensional e para criação de *game engines*<sup>28</sup>, o Blender tem se destacado entre os softwares da área de computação gráfica e multimídia por sua alta qualidade técnica, sendo atualmente o maior destaque entre os softwares livres desta área. Neste capítulo, trataremos dos recursos trazidos pelo Blender e analisaremos as estratégias de financiamento adotadas pela sua comunidade, de forma a entender como estas influenciaram no desenvolvimento do Blender.

#### 3.1 Do surgimento ao código livre

A origem do Blender encontra-se num estúdio de animação holandês chamado NeoGeo, o qual foi co-fundado por Ton Roosendaal em 1988. A NeoGeo utilizava um ambiente de animação desenvolvido dentro do próprio estúdio e chegou a conquistar alguns prêmios por suas produções. O desenvolvimento de software e a direção de arte da NeoGeo era coordenado por Ton Roosendaal. No ano de 1995, a equipe da NeoGeo começou a reescrever todo o código do software de animação que eles utilizavam. O código reescrito desse software deu origem ao Blender.

No ano de 1998, surgiu a *NaN (Not a Number)*, uma subsidiária da NeoGeo, fundada no intuito de criar mercado e desenvolver o Blender. A *NaN* tinha ainda como princípio distribuir o Blender de forma gratuita e, dessa forma, garantir ao público de computação em geral acesso a uma ferramenta de modelagem e animação 3D de nível profissional. Para isso, a empresa baseou seu modelo de negócios na venda de produtos e serviços em torno do Blender.

Inicialmente, os resultados dos planos da *NaN* foram satisfatórios. Investidores garantiram recursos para que a empresa pudesse contar com 50 funcionários e, no final do ano 2000, o Blender já estava em sua versão 2.0 e possuía mais de 250.000 usuários registrados. Porém no ano de 2001, as ambições

---

<sup>28</sup> *Game engine*, ou motor de jogo, é um software com o objetivo de simplificar o desenvolvimento de jogos e de outros tipos de simulação em tempo real, fornecendo algoritmos capazes de renderizar gráficos, simular leis físicas, prover animação, entre outros recursos.

da *NaN* não mais encontraram respaldo no mercado. A empresa tentou ainda comercializar o Blender Publisher, uma versão do Blender focada na criação de conteúdo interativo em 3D para internet, porém as vendas fracassaram e a *NaN* acabou indo à falência, interrompendo assim o desenvolvimento do Blender.

Vendo que não era possível retomar as atividades da empresa com um número suficientemente grande de funcionários para continuar a desenvolver o Blender, Ton Roosendaal desloca seus esforços para a criação da *Blender Foundation*. O objetivo da fundação era – e continua sendo até hoje – desenvolver e promover o Blender, entretanto não apenas como um software gratuito, mas sim como um software livre desenvolvido em comunidade. Dessa forma, foi iniciada uma campanha para arrecadar doações em dinheiro para que a *Blender Foundation* pudesse adquirir os direitos sobre o código fonte do Blender. Em apenas sete semanas, a campanha conseguiu adquirir os cem mil euros necessários para tornar o Blender um software livre e, em outubro de 2002, este foi liberado sob a licença GNU GPL. Desde então, o Blender vem sendo desenvolvido por uma comunidade de colaboradores de diversos locais do mundo.

## **3.2 Os recursos do Blender**

Desde alguns anos, o Blender provê toda a gama de recursos necessários para a produção de uma animação 3D de qualidade profissional. A animação “*Elephants dream*”, lançada em maio de 2006, foi produzida pelos esforços da *Blender Foundation* com o objetivo de demonstrar o potencial do software e também para auxiliar o desenvolvimento deste. Neste tópico, trataremos dos principais recursos oferecidos pelo Blender e demonstraremos a aplicação de alguns destes no vídeo “*Elephants dream*”.

### **3.2.1 Ferramentas de modelagem**

Como vimos no capítulo anterior, o processo de produção de uma animação 3D é iniciada pela modelagem, sendo assim, de acordo com Brito (2007), o Blender é um modelador poderoso capaz de criar modelos complexos de objetos, cenários, personagens, curvas e textos 3D. O autor afirma também que o sistema de

modelagem do Blender é quase que totalmente baseado em subdivisão, a qual consiste em utilizar uma forma simples criada pelo software e subdividir esta em formas mais complexas.

O Blender disponibiliza recursos de modelagem baseados em todos os métodos citados no capítulo anterior. Utilizando o método de modelagem por primitivas geométricas, o Blender possibilita adicionar as seguintes formas geométricas, denominadas pelo software como objetos *Mesh*: plano, grade (um plano com várias subdivisões a critério do usuário), cubo, polígonos com ilimitado números de lados, esferas (com faces quadradas ou triangulares), cilindro, tubo e cone. Na criação destes três últimos objetos, devemos informar o número de vértices desejados. Dessa forma, é possível criar outras formas geométricas, por exemplo, determinando três vértices para um cilindro, teremos um prisma, já determinando quatro vértices para um cone, cria-se uma pirâmide.

Esses objetos podem ser alterados através de uma diversa gama de recursos oferecidos pelo Blender, conforme descrito por Brito (2007). Os recursos incluem ferramentas de extrusão, revolução, espelhamento, seccionamento, subdivisão, suavizamento de bordas e curvas, modificação por operações booleanas (união, intersecção e diferença), entre outros. Além disso, estão disponíveis também outras ferramentas de modelagem como a *Lattice*<sup>29</sup>, *DupliVerts*, *DupliFaces* e *DupliGroup* – todas as três são ferramentas de duplicação e clonagem – e o *Sculpt Mode* – um modo de modelagem com o qual é possível esculpir o objeto com o uso do mouse.

Fig. 3.1 – Exemplo de curva Bezier

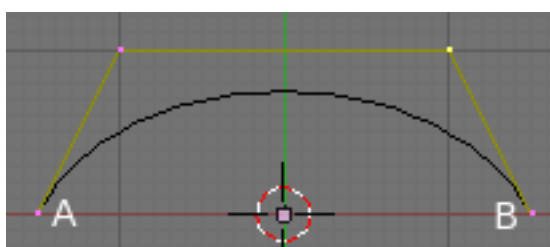


Fonte: <http://wiki.blender.org/index.php/Manual/PartII/Curves>

<sup>29</sup> A *Lattice* consiste em criar uma malha em torno do objeto e fazer com que este seja deformado de acordo com as transformações efetuadas na malha. De acordo com Brito (2007), esta ferramenta é bastante utilizada em softwares de modelagem 3D e muito útil para animação facial e modelagem em geral.

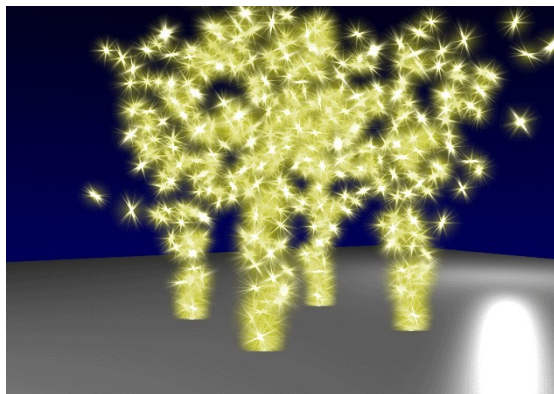
Em relação à modelagem de curvas, o Blender oferece as do tipo Bezier e as NURBS. A diferença entre uma curva Bezier e uma do tipo NURBS é a forma como esta pode ser manipulada pelo usuário. Enquanto na Bezier (Fig. 3.1) a curva passa sobre todos os pontos de controle de curvatura, na NURBS, que utiliza o algoritmo B-spline, a linha da curva não passa diretamente no ponto de controle (Brito, 2007; Lucena, 2005). Como pode ser visualizado na Fig. 3.2, os pontos de controle de uma curva NURBS (em amarelo e lilás) não se localizam sobre a linha da curva (em preto).

Fig. 3.2 – Exemplo de curva do tipo NURBS



Fonte: <http://wiki.blender.org/index.php/Manual/PartII/Curves>

Fig. 3.3 – Sistema de partículas e material Halo



Fonte: <http://wiki.blender.org/index.php/Manual/Particles>

Quanto à modelagem por procedimento, através do sistema de partículas do Blender é possível simular efeitos naturais conforme descrito no Capítulo 2. Nesse sentido, temos a opção do software criar as partículas ou de utilizarmos objetos como partículas. Dentro da primeira opção, podemos escolher partículas dinâmicas, as quais são utilizadas no momento em que precisamos criar efeitos de fumaça, chuva, poeira, vapor, entre outros, ou, partículas estáticas, com as quais é possível simular pelos e vegetação. Por sua vez, a opção de utilizar um objeto como partícula, permite que se crie partículas a partir de qualquer objeto já modelado. Na

Fig. 3.3, podemos visualizar a utilização do sistema de partículas para gerar uma chuva de estrelas.

Complementando a parte de modelagem, o Blender possibilita a aplicação de materiais e texturas aos objetos. De acordo com Brito (2007), os materiais têm como função prover maior realismo aos objetos modelados, informando ao software como a superfície destes deve se comportar em relação à luz. Entre os recursos de materiais do Blender encontram-se opções avançadas como a simulação de superfícies especulares e efeitos semelhantes a um Halo<sup>30</sup>. Na Fig. 3.3, o material Halo foi utilizado para simular a luz de estrelas.

Já em relação às texturas, é possível utilizar tanto as baseadas em imagem, como as procedurais, as quais são baseadas em cálculos matemáticos, e aplicá-las, não apenas a objetos, mas também a luzes e ao ambiente. Brito (2007) afirma que existem vários tipos de textura disponíveis no Blender, sendo a maioria do tipo procedural. Qualquer imagem nos formatos jpeg, png e tga podem ser adicionados como textura no Blender. Além disso, é possível mapear as texturas utilizando desde métodos mais simples até o mapeamento UV. O mapeamento é necessário para definir como a textura ficará distribuída ao longo da superfície do objeto e o método UV é o que provê um maior controle sobre esse processo, pois possibilita definir exatamente qual parte da textura será aplicada a cada face do modelo.

Sendo assim, avaliamos que o Blender oferece um diversificado arranjo de ferramentas de modelagem<sup>31</sup>, as quais, conforme afirmou Brito (2007), possibilitam a criação de modelos de alta complexidade. Num tópico seguinte, teremos a oportunidade de ver alguns modelos criados com o Blender para a animação “*Elephants dream*”.

### 3.2.2 Iluminação e renderização

A iluminação de uma cena num software de animação 3D é realizada através da adição das fontes de luz, as quais são identificadas no Blender pelo termo “*Lamp*” (lâmpada). Sendo assim, estão disponíveis no Blender cinco tipos de lâmpadas: *Sun*, simula a luz do sol; *Hemi*, simula a iluminação de um dia nublado, bastante difusa e uniforme; *Lamp*, irradia energia luminosa para todos os lados ou

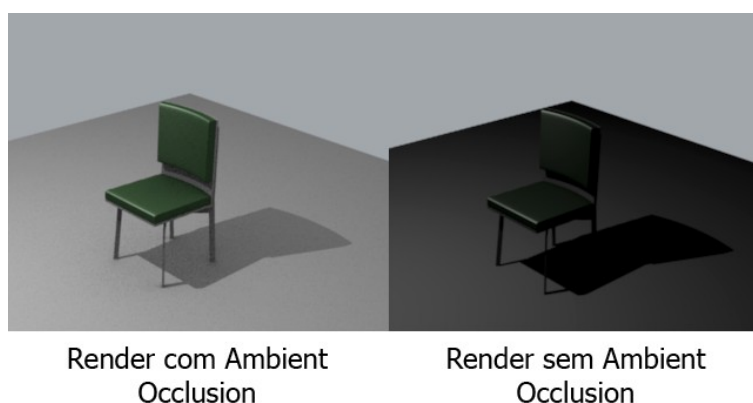
<sup>30</sup> “O efeito de halo consiste na dispersão e na reflexão da luz em cristais” (BRITO, 2007, p. 219)

<sup>31</sup> Além dos recursos citados, o Blender permite ainda importar um objeto que já tenha sido modelado em outros softwares, como o 3ds Max, o LightWave, entre outros.

apenas para a uma área esférica; *Area*, lâmpada com formato quadrado ou retangular; e, por fim, *Spot*, a qual tem o formato de um cone de luz.

Todas estas lâmpadas possuem parâmetros de intensidade, alcance, cor, entre outros, os quais permitem obter a iluminação desejada de acordo com cada cena. Além disso, o Blender conta com o método de iluminação *ray tracing*, com cálculos de radiosidade e com uma ferramenta chamada *Ambient Occlusion*. Esta ferramenta é um dos métodos de iluminação global<sup>32</sup> e faz com que o fundo da imagem emita raios luminosos, o que, conforme afirma Brito (2007) e é demonstrado na Fig. 3.4, torna a iluminação do ambiente mais uniforme e realista.

Fig. 3.4 – Demonstração da ferramenta Ambient Occlusion



Fonte: BRITO, 2007, p. 283.

Em relação à renderização, o Blender possui um renderizador nativo e pode utilizar também um renderizador externo, como o YafRay, o qual também é software livre e totalmente compatível com o Blender. Brito (2007) afirma que o YafRay é capaz de criar renderizações fotorrealísticas e adiciona vários recursos de iluminação às cenas, como diversos métodos de iluminação global, caustics<sup>33</sup> e HDRI<sup>34</sup>. Além disso, o Blender é compatível com outros renderizadores, tais como o PovRay, o qual também é um software livre, *Mental Ray*, *Renderman*, *Indigo* e *Virtualight*.

32 A iluminação global é um método de iluminação que procura equilibrar a luminosidade, reduzindo o contraste entre os ambientes iluminados e os sombreados.

33 O *Caustics*, de acordo com Brito (2007), simula o efeito que acontece quando a luz passa por um objeto transparente e sua energia concentra-se num ponto, um bom exemplo é o reflexo que aparece nas lupas.

34 HDRI ou *High Dinamic Range Image*, “[...] é um tipo especial de imagem que armazena a informação luminosa no momento em que foi criada” (BRITO, 2007, p. 296). Utilizando o YafRay em conjunto com o Blender, é possível iluminar um objeto com a informação de luminosidade disponível numa imagem HDRI.

Desta forma, vemos que em relação à iluminação e renderização, o Blender também oferece as ferramentas necessárias para o trabalho com animação 3D de nível profissional. O poder de tais ferramentas pode ser melhor visualizado no tópico acerca da animação “*Elephants dream*”.

### 3.2.3 Animação

Como vimos no capítulo anterior, a geração de movimento é uma das etapas mais complexas do processo de produção de uma animação. Nesse contexto, o Blender oferece todos os métodos de animação citados anteriormente: animação por *keyframe*, controle por esqueleto, animação com curvas, procedural (geração de partículas) e cinemática inversa.

Em relação à animação por *keyframes*, destacamos a presença do *Ipo Curve Editor*, que pode ser traduzido como editor de curvas de interpolação. Tal editor permite o controle da animação através de gráficos, os quais podem controlar diversos parâmetros, como posição horizontal, vertical, rotação, tamanho, etc. As curvas de interpolação podem ser aplicadas na animação não apenas de objetos, mas também de materiais, texturas, lâmpadas, câmera, entre outras ferramentas. Além disso, o Blender disponibiliza a ferramenta *DupliFrames*, a qual, conforme descreve Brito (2007), possibilita criar cópias dos objetos em animação, e os filtros de *Constraints*, que controlam ou limitam as transformações dos objetos ao longo da animação.

Outro recurso bastante importante na fase de animação é a simulação de física real. Nesse contexto, o Blender possui as seguintes ferramentas: *Fluidos*, *Force Fields*, *Deflection*, *Rigid Body* e *Soft Body*.

Os Fluidos, como o próprio nome já esclarece, têm a utilidade de simular o comportamento de fluidos. Já o *Force Fields* pode gerar um campo de forças que irá agir numa determinada região, influenciando o movimento dos objetos que se encontrarem à frente. Com esta ferramenta é possível simular o efeito do vento, por exemplo, como pode ser visualizado na Fig. 3.5. A ferramenta *Deflection*, por sua vez, determina como será o resultado de uma colisão de um objeto *Mesh* com partículas ou com um *Soft Body*. Isso é realizado através do estabelecimento de parâmetros que determinam quanto do impacto será absorvido pelo objeto *Mesh* e quanto será refletido.

Fig. 3.5 – Simulação de vento produzida com o uso da ferramenta *Force Fields*



Fonte: <http://www.blender.org/development/release-logs/blender-237a/forces-and-deflection/>

Por fim, temos as duas últimas ferramentas: *Rigid Body* e *Soft Body*. A primeira simula as propriedades de corpos rígidos e pode ser utilizada, por exemplo, para gerar a colisão entre dois corpos. O *Soft Body* por sua vez, permite simular propriedades elásticas, o que facilita bastante o trabalho de animação das roupas dos personagens e de outros objetos macios ou elásticos. As propriedades mais importantes que podem ser aplicadas aos modelos são fricção (determina a facilidade com que o objeto será deformado), velocidade da deformação, gravidade, massa (quanto maior a massa, mais lento será o movimento), elasticidade e colisões (determina a maneira como os objetos colidem consigo próprios e com obstáculos).

Desta forma, encerramos aqui a descrição dos principais recursos voltados para a produção de animação tridimensional do Blender. Como podemos observar, todos os recursos essenciais a um ambiente de animação tridimensional citados no capítulo anterior estão presentes no Blender. É importante observar ainda que o software é compatível com um grande número de sistemas operacionais, tais como o GNU/Linux, Mac OS X, Microsoft Windows, FreeBSD, Solaris e outros.

### **3.3 Análise dos recursos presentes no vídeo “*Elephants Dream*”**

A animação “*Elephants Dream*”, lançada no ano de 2006, foi um projeto coordenado pela *Blender Foundation* com o intuito de demonstrar o potencial do Blender e também de auxiliar no desenvolvimento do software. Com o projeto, foram

desenvolvidos novos recursos para o Blender e diversos outros puderam ser melhorados.

O cenário da animação é o interior de uma máquina retratado de maneira bastante surreal. Apenas dois personagens humanos compõem a história: Emo e Proog. O personagem Proog vai apresentar o funcionamento da máquina para o jovem Emo, o qual não aceita a visão de Proog e os dois acabam entrando em conflito. Com isso, descreveremos abaixo alguns dos recursos mais complexos que foram utilizados no *“Elephants Dream”*.

Fig. 3.6 Personagem visto através do reflexo na água



A imagem acima é um quadro da primeira cena do vídeo, na qual podemos constatar a utilização do recurso de Fluidos para simular o movimento da água e também a simulação de uma superfície especular, refletindo a face do personagem Proog. A reflexão especular, por sua vez, só é possível pela utilização do método de radiosidade em Proog, pois a imagem que vemos refletida na água tem sua origem numa fonte de luz que é refletida no personagem. Estes recursos conseguiram proporcionar grande realismo à cena, visto que a reflexão varia de acordo com a posição da face do personagem e o espelho d'água se movimenta adequadamente. Além disso, percebemos o uso de materiais e textura no piso e uma boa aplicação da iluminação, com sombras dotadas de grande realismo.

Já na cena representada pela Fig. 3.7, podemos perceber o uso do sistema de geração de partículas, as quais surgem no momento da colisão dos cabos com a parede e podem ser vistas na parte inferior esquerda da imagem. O gerador de

partículas é utilizado também para criação dos cabelos dos personagens e para simular uma explosão em uma outra cena do vídeo. No lado esquerdo da imagem, percebemos o uso de recursos atmosféricos para a simulação de névoa. Além disso, podemos notar também a simulação de profundidade de campo pela câmera, já que parte da imagem se encontra em foco e outra parte não. O uso de tal recurso proporciona um grande realismo, já que traz para a animação um recurso estético próprio do cinema ao vivo.

Fig. 3.7 Uso do sistema de partículas e efeitos atmosféricos



Um outro recurso presente nesta cena é o *“motion blur”*, o qual gera um borrão de movimento a partir do deslocamento dos cabos. O deslocamento dos cabos na cena é dotado de um preciso controle de aceleração e desaceleração, demonstrando, dessa forma, a qualidade do *“Ipo Curve Editor”*, ferramenta do Blender na qual é realizada o controle de animação.

Já na Fig. 3.8, temos uma boa amostra da capacidade dos recursos de iluminação do Blender. A técnica de iluminação global foi utilizada para tornar visível o fundo do cenário e, no centro da sala, temos uma lâmpada do tipo *“Lamp”*, dentro de uma luminária apontada para o chão. A luz dessa luminária é refletida de forma bastante realista pelo tapete vermelho. Por fim, é possível perceber mais uma vez uma boa aplicação de materiais e texturas no chão e nos objetos presentes na parede de trás da sala.

Fig. 3.8 Utilização da iluminação global e do recurso Lamp

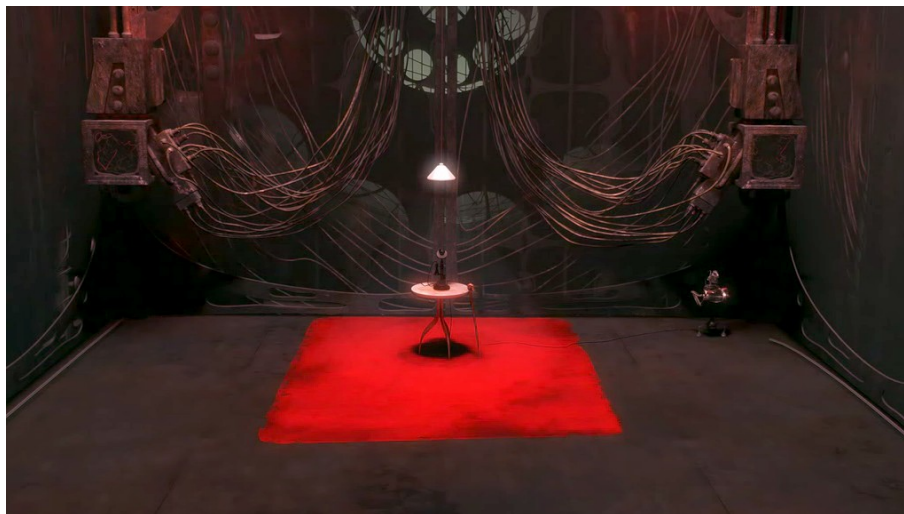


Fig. 3.9 Uso do sistema de simulação de física real e da cinemática inversa



Acima, temos parte de um quadro da cena em que o personagem Proog salta pelas teclas de uma máquina de escrever gigante. Nesta cena podemos ver em ação o sistema de simulação de física real através da aplicação da ferramenta de *Soft Body* à roupa de Proog. O casaco segue os movimentos do corpo do personagem, porém também está sujeito à força da inércia e à gravidade. Em algumas outras cenas, percebe-se o uso da ferramenta de *Rigid Body* para simular colisões, como no momento em que o personagem Emo choca-se com Proog. Além disso, podemos notar o uso do recurso de criação de esqueletos e ossos e da cinemática inversa na animação do personagem.

Desta forma, vemos que os recursos do Blender já podem ser aplicados na

produção de animações de qualidade profissional, sendo o “*Elephants Dream*” uma grande amostra da qualidade técnica do software. Nesta animação podemos visualizar a aplicação de complexos recursos computacionais, a exemplo da simulação de física real, da geração de partículas e de diversos métodos de iluminação.

### 3.4 A avaliação dos usuários do Blender

Com o objetivo de conhecer a opinião dos usuários do Blender a respeito da qualidade técnica do software, foi realizada uma pesquisa de campo através de um formulário disponibilizado na internet. O formulário ficou disponível à participação dos usuários no período de 02 a 18 de março, tendo sido divulgado em dois sites nacionais e um internacional pertencentes à comunidade do Blender, além do Br-Linux<sup>35</sup>, um dos mais visitados sites de software livre do Brasil, e em diversos blogs. A pesquisa visou atingir a maior diversidade possível de usuários do Blender, porém sem o intuito de ser uma amostra estatisticamente fiel da diversidade destes. Pudemos contar, pois, com a participação de 43 usuários do Blender, sendo 27 residentes no Brasil e os outros 16 de diversos locais do mundo.

Sendo assim, em relação à qualidade técnica, 61,9% dos usuários participantes da pesquisa responderam que consideram o Blender um software muito bom tecnicamente. Já outros 31% avaliam o software como bom e apenas 7,1% avaliaram como razoável. Nenhum dos usuários apontou o Blender como um software ruim tecnicamente.

Convém destacar que o público que respondeu ao formulário, em sua maioria, tem bastante experiência com o Blender, sendo que 48,8% utiliza-o há mais de três anos e 32,6% há mais de um ano. Os usuários que utilizam o software há menos de um ano contabilizam apenas 18,6% do total de entrevistados.

Em relação aos aspectos técnicos do Blender, a interface gráfica é o que causa maior controvérsia entre os usuários participantes da pesquisa: 37,2% destes consideram-na como um aspecto positivo do software. Por outro lado, 20,9% avaliam-na como um aspecto negativo, em geral apontando que esta é pouco intuitiva e de difícil utilização. Em relação a esse aspecto do Blender, Brito (2007) afirma que esta é um pouco diferente da dos outros softwares de modelagem 3D,

<sup>35</sup> <http://www.br-linux.org>

principalmente pelo fato de muitas operações serem realizadas por meio de atalhos de teclado. Porém, segundo o autor, apesar de a interface inicialmente parecer estranha, esta é altamente otimizada e muito eficaz. De fato, são os usuários iniciantes os que mais reprovam a interface. Entre os usuários que responderam ao formulário e têm menos de um ano de experiência com o software, 50% apontaram a interface entre os aspectos negativos e apenas 25% a avaliaram positivamente. Já entre os usuários com mais de um ano de experiência, a interface gráfica foi avaliada positivamente por 48,6% e negativamente por 17,1%. Estes dados, em conjunto com a abordagem de Brito (2007), demonstram que a interface do Blender, apesar de não parecer intuitiva ao usuário iniciante, garante eficiência aos usuários que já compreenderam a lógica de seu funcionamento.

Os outros aspectos qualificados como mais positivos no software são os recursos de modelagem (25,6% qualificaram positivamente), animação (14%), simulação de leis físicas (11,6%) e o fato de o software não exigir muitos recursos do computador (9,3%).

Já entre os aspectos considerados negativos está o renderizador interno do Blender, o qual foi qualificado negativamente por 18,6% dos usuários participantes da pesquisa. Essa limitação, porém, pode ser suprida pela utilização de um renderizador externo, como o YafRay<sup>36</sup>, o qual também é um software livre, totalmente compatível com o Blender, e cujos recursos são avaliados positivamente por Brito (2007). A utilização de outros softwares para realização de tarefas específicas é bastante comum nas comunidades de software livre e visa a economia de esforços de programação. Se já existe um software capaz de realizar uma determinada tarefa com a qualidade desejada, é mais produtivo a utilização deste que o desprendimento de esforços no desenvolvimento de tal recurso em um outro programa. No vídeo *“Elephants Dream”*, por exemplo, além do Blender foram utilizados mais alguns softwares livres, como o Gimp<sup>37</sup>, o Inkscape<sup>38</sup>, o Cinepaint<sup>39</sup> e a linguagem de programação Python<sup>40</sup>.

Entre os outros aspectos considerados negativos estão os recursos de

---

36 <http://www.yafaray.org>

37 GIMP, ou *GNU Image Manipulation Program*, é um editor de imagens bitmap. <http://www.gimp.org>

38 Inkscape é um software de edição de gráficos vetoriais. <http://www.inkscape.org>

39 Cinepaint é um fork do GIMP voltado para a pintura e retoque de frames de um vídeo. <http://www.cinpaint.org/>

40 A linguagem de programação Python, entre outras finalidades, pode ser utilizada no desenvolvimento de scripts para o Blender. <http://www.python.org/>

modelagem por NURBS, apontado como negativo por 7% dos usuários, e algumas outras ferramentas de modelagem, com a qualificação também de 7% dos usuários.

Desta forma, concluímos que a grande maioria dos usuários do Blender participantes da pesquisa o avaliam positivamente em relação à qualidade técnica e que, apesar de a interface não se apresentar intuitiva ao usuário iniciante, esta satisfaz a maioria dos usuários, principalmente os mais experientes. Como em todo software, existem recursos que ainda precisam serem aprimorados. Nesse sentido, conforme afirmamos anteriormente, a opinião dos usuários é de grande importância para o crescimento de um projeto de software livre.

### 3.5 A comunidade do Blender

Como foi abordado no primeiro capítulo, o modelo de desenvolvimento denominado por Eric Raymond (1998) de “bazar” mostrou-se bastante vantajoso para o crescimento dos projetos de software livre. Contudo, o próprio Raymond (1998) alertou que são necessários alguns esforços por parte dos desenvolvedores dos softwares para fazer o projeto crescer em número de colaboradores e usuários e, conseqüentemente, em qualidade técnica.

Nesse contexto, a comunidade do Blender mostra-se bastante organizada. Sob a liderança de Tom Roosendaal, cerca de 45 pessoas contribuem atualmente com a programação, correção de erros ou manutenção de módulos do software, além da manutenção do site oficial<sup>41</sup>. O número de desenvolvedores ativos do Blender atualmente é bem maior do que na época em que este ainda era de código fechado, cujo número máximo de desenvolvedores não passou de dezesseis<sup>42</sup>.

Este dado é mais uma comprovação do potencial do modelo “bazar” e dos softwares de código aberto frente ao modelo de produção de software proprietário. Ainda em relação ao interesse de programadores em colaborar com o desenvolvimento do Blender, é bom retomar o que Raymond (1998) defende ser uma das condições necessárias para o bom desempenho de um projeto no modelo “bazar”:

---

41 BLENDER FOUNDATION. **List of Contributors**. Disponível em: [http://wiki.blender.org/index.php/List\\_of\\_Contributors](http://wiki.blender.org/index.php/List_of_Contributors) . Acesso em 21/03/2008.

42 BLENDER FOUNDATION. **Manual.pt/PartI/Introduction**. Disponível em: <http://wiki.blender.org/index.php/Manual.pt/PartI/Introduction> . Acesso em 21/03/2008.

Quando você começa a construção de uma comunidade, o que você precisa ter capacidade de apresentar é uma promessa plausível. Seu programa não precisa funcionar particularmente bem. Ele pode ser grosseiro, cheio de erros, incompleto, e pobremente documentado. O que não pode deixar de fazer é convencer co-desenvolvedores em potencial de que ele pode evoluir para algo realmente elegante em um futuro próximo. (RAYMOND, 1998, p. 11-12)

Desta maneira, podemos considerar o fato do Blender já contar com uma base de código no momento em que foi disponibilizado sob a licença GNU GPL, como um possível fator para ter atraído o interesse de um bom número de desenvolvedores.

Além do aumento no número de desenvolvedores, desde que o Blender foi disponibilizado como software livre, no final do ano de 2002, ocorreu também um grande acréscimo de recursos ao programa. Apenas no primeiro ano foram lançadas cinco novas versões do Blender, sendo que a maior modificação no software foi a nova interface gráfica.

Já no ano de 2004, os recursos implementados incluem, entre outros, a técnica de iluminação global denominada “*Ambient Occlusion*”, novas texturas procedimentais, interação e duplicação de partículas, mapeamento UV, integração com o renderizador YafRay e também foram realizadas melhorias no renderizador interno. Em meados de 2004, o número de linhas de código já ultrapassava 330 mil, o que representa um crescimento de cerca de 50% em relação à última versão lançada com o código fonte fechado, a qual possuía 221.739 linhas<sup>43</sup>.

No ano seguinte, foram acrescentados os recursos de simulação de leis físicas (*soft body*, *rigid body*, *deflection*, campo de forças e fluídos), simulação de pêlos com partículas, o sistema de controle por esqueletos foi reprogramado, entre outros recursos<sup>44</sup>. A última versão lançada no ano de 2005 (numerada como 2.40) contou com a colaboração de 47 desenvolvedores.

A grande quantidade de recursos e correções acrescentadas ao Blender em seus primeiros anos como software livre mostram que rapidamente se formou uma comunidade ativa de desenvolvedores em torno do software. Tal comunidade conseguiu implementar inovações ao programa num ritmo avançado, provando, mais uma vez, a viabilidade do método de desenvolvimento de software livre.

---

43 BLENDER FOUNDATION. **Code History**. Disponível em: <http://www.blender.org/development/coding-guides/code-history/>. Acesso em 21/03/2008.

44 BLENDER FOUNDATION. **Manual.pt/PartI/Introduction**. Disponível em: <http://wiki.blender.org/index.php/Manual.pt/PartI/Introduction>. Acesso em 21/03/2008.

### 3.5.1 As estratégias de financiamento adotadas pela comunidade do Blender

A comunidade do Blender utiliza algumas estratégias de obtenção de recursos, fator este que, como analisaremos, pode ter influenciado o ritmo de inovação do Blender. Na verdade, antes mesmo de o Blender ter seu código fonte aberto, foi fundada a *Blender Foundation*, entidade que cumpriu a função de receber as doações em dinheiro para a compra da propriedade intelectual sobre o código do software.

Neste sentido, o site oficial afirma que a *Blender Foundation* é uma entidade sem fins lucrativos com os seguintes objetivos:

- Manter e melhorar o Blender, sempre com o princípio de disponibilizá-lo sob a licença GNU GPL;
- Obter fundos e desenvolver fontes de recursos para a realização dos outros objetivos da fundação e para cobrir as despesas desta;
- Estabelecer serviços para desenvolvedores e usuários ativos do Blender;
- Oferecer acesso à tecnologia 3D para toda a comunidade usuária da Web, tendo o Blender como núcleo desta tecnologia.

As formas de obtenção de recursos adotados pela *Blender Foundation* compreendem doações, manutenção de uma loja eletrônica, realização de cursos e dos “*open projects*”. Em relação à loja eletrônica, os principais produtos oferecidos são livros e DVD's com material didático do Blender e também os DVD's dos projetos de “*open projects*”.

Por sua vez, os “*open projects*” envolvem a realização de projetos, como vídeos e jogos eletrônicos, com o objetivo de, ao mesmo tempo, promover o Blender enquanto ferramenta técnica de qualidade profissional e auxiliar o desenvolvimento do Blender e de outros softwares livres. Os recursos que possibilitam a execução dos projetos têm origem em fundos de cultura, patrocínio de empresas e da venda de DVD's. Ao fim do projeto, todos os resultados são disponibilizados na rede sob uma licença que permita a livre utilização, modificação e redistribuição destes por qualquer pessoa.

Além disso, a *Blender Foundation* possui uma sede, localizada na Holanda, a qual abriga a equipe dos “*open projects*” e onde também são realizados cursos e workshops de Blender. Os recursos adquiridos com os projetos, cursos, loja

eletrônica e doações possibilitam à fundação pagar para que Ton Roosendaal e uma pequena equipe de desenvolvedores se dediquem em tempo integral ao Blender, além de cobrir despesas administrativas e possibilitar a organização de eventos como a “*Blender Conference*” e a divulgação do software em outros eventos.

### 3.5.2 Contribuições dos “*open projects*”

O primeiro “open project” executado pela Blender Foundation foi a animação “*Elephants Dream*”<sup>45</sup>, lançada em maio de 2006. A equipe central do projeto foi composta por oito pessoas, porém também contou com a contribuição de mais alguns artistas, da equipe de desenvolvedores do Blender e da comunidade em geral. O projeto “*Elephants Dream*” possibilitou a melhoria e implementação de muitos recursos ao Blender. Entre estes, podemos citar melhorias no sistema de renderização, na simulação de fluídos, no editor de vídeo, nas opções de materiais e nas ferramentas de animação de personagens. Além disso, foram adicionados os recursos de *motion blur*<sup>46</sup>, transmissividade<sup>47</sup>, utilização de imagens HDRI e agrupamento de objetos<sup>48</sup>.

Atualmente existem dois outros projetos sendo executados pela Blender Foundation, os quais seguem as mesmas diretrizes do “*Elephants Dream*”. Um projeto de animação intitulado “*Peach*”, e que já está em fase de finalização, e um de jogo eletrônico intitulado “*Apricot*”. Quanto às contribuições destes projetos ao Blender, podemos citar as melhorias na simulação de tecidos, no sistema de partículas, principalmente para geração de pêlos, e novas opções de *soft body*, além das melhorias nos recursos voltados para a criação de jogos. Alguns desses recursos já estão disponíveis na versão 2.46RC1 do Blender, lançada em 10 de março de 2008.

A realização dos “*open projects*”, portanto, tem apresentado contribuições significativas para o desenvolvimento de recursos do Blender, constituindo-se como

---

45 No site <http://www.elephantsdream.org> podem ser encontradas mais informações sobre o projeto. Neste site também estão disponíveis para download os arquivos de produção, bem como a animação finalizada.

46 O *motion blur* é utilizado para simular o borrão de movimento na tela quando algo se movimenta de forma rápida. Esse recurso, como citado anteriormente, foi utilizado em diversas cenas da animação.

47 O recurso de transmissividade é utilizado em conjunto com o método de *ray tracing* para calcular a penetração de um raio de luz através de um objeto.

48 Este recurso provê mais facilidade na animação de objetos que se comportam de forma semelhante.

uma estratégia de sucesso para a sustentabilidade do software. Além do ganho de qualidade técnica, os “open projects” também têm o mérito de divulgar o potencial do programa, o que pode trazer mais benefícios a médio e longo prazo.

### **3.5.2 As estratégias adotadas pela comunidade do Blender em relação às de outros softwares**

O bom resultado das estratégias de financiamento do Blender nos leva a um questionamento: a comunidade do Blender tem repetido uma fórmula de sucesso de outros softwares livres ou tem inovado em suas estratégias?

Para obter respostas à esta pergunta, realizamos uma pesquisa de quais estratégias de sustentabilidade são utilizadas por alguns dos softwares livres mais populares atualmente. Para isso consultamos os sites oficiais dos seguintes softwares: Apache, GNOME, GNU Project, KDE, Linux, Mozilla Firefox e OpenOffice. Temos abaixo os resultados encontrados:

#### **Apache**

O Apache, como já foi citado, é o software servidor web mais utilizado no mundo. Este possui uma fundação com o intuito de angariar recursos para suportar o desenvolvimento e também para manter a infra-estrutura de hardware, comunicação e divulgação necessária para a distribuição do software. Conta com patrocínio de grandes empresas como Google, Yahoo e HP. Além disso, existe divulgação de livros a respeito do Apache no site e a fundação recebe comissão das lojas pelas vendas destes.

#### **GNOME**

O GNOME é um conjunto de softwares<sup>49</sup> que juntos oferecem uma área de trabalho completa em software livre. A comunidade conta com a GNOME Foundation, a qual recebe patrocínio de grandes empresas como Intel, Nokia, IBM e HP, entre outras. Além disso, também incentiva doações de usuários e realiza a comercialização de alguns poucos artigos.

#### **GNU Project**

O GNU, como já foi abordado, é um conjunto de aplicativos, que em conjunto com o kernel, compõe o sistema operacional GNU/Linux. O Projeto GNU conta com

---

<sup>49</sup> Esse conjunto de softwares incluem, por exemplo, editor de texto, aplicações para configuração do sistema, aplicativos multimídia, jogos e gerenciador de arquivos.

a Free Software Foundation (FSF) para obtenção de recursos para promover o desenvolvimento e divulgação dos softwares. Desta forma, grandes empresas como Google, IBM e Sun Microsystems oferecem patrocínio à FSF. Além disso, a fundação recebe doações de usuários e possui uma loja virtual onde são vendidos CD's, manuais, camisetas e outros itens relacionados aos softwares.

### **KDE**

O KDE também é um ambiente de trabalho, o qual, juntamente com o GNOME, são os dois mais utilizados nos sistemas GNU/Linux. O KDE recebe patrocínio de várias empresas, as quais cedem recursos de hardware ou pagam para que alguns desenvolvedores possam se dedicar integral ou parcialmente ao software. Também incentiva doações e comercializa alguns poucos artigos, porém não possui uma fundação.

### **Linux**

O kernel Linux também conta com o apoio de uma fundação, a Linux Foundation, e com o patrocínio de grandes empresas como HP, IBM e Sun Microsystems. O site oficial não menciona doações nem venda de produtos.

### **Mozilla Firefox**

O Firefox é, há mais de quatro anos, o segundo navegador de internet mais utilizado no mundo<sup>50</sup>. Conta com uma fundação que capta e disponibiliza recursos para o desenvolvimento do Firefox e de outros softwares como o cliente de e-mails Thunderbird. Possui, além disso, uma loja virtual que comercializa produtos como CD's e materiais de ajuda dos softwares, camisetas e outros artigos. O site oficial também incentiva empresas e pessoas físicas a realizarem doações.

### **OpenOffice**

O OpenOffice é há bastante tempo a principal suíte de escritório de código aberto. Este software tem origem no código fonte do StarOffice, o qual foi liberado pela empresa Sun Microsystems no ano 2000. Ainda hoje esta empresa é a principal financiadora do OpenOffice. Não possui uma fundação e, apesar de aceitar doações em dinheiro, prefere que as pessoas se voluntariem ao projeto.

Com isso, podemos perceber que o pedido de doações e a criação de fundações são estratégias de financiamento bastante utilizadas, sendo que apenas o KDE e o OpenOffice, entre os softwares pesquisados, não possuem uma fundação.

---

50 Dados da página: [http://en.wikipedia.org/wiki/Usage\\_share\\_of\\_web\\_browsers](http://en.wikipedia.org/wiki/Usage_share_of_web_browsers).

Também é possível perceber que grandes empresas da área de tecnologia têm se interessado em patrocinar o desenvolvimento de software livre, visto que estas podem se beneficiar diretamente com a melhoria na qualidade técnica destes. Já a comercialização de produtos e serviços mostra-se como a estratégia menos utilizada pelas comunidades pesquisadas.

Contraopondo estes resultados com as estratégias utilizadas pela comunidade do Blender, percebemos que este difere pela realização dos “*open projects*” e por não contar com o patrocínio de empresas. É importante observarmos, no entanto, que o Blender possui uma nível de utilização bastante restrito se comparado aos outros softwares pesquisados, os quais são praticamente essenciais em um sistema operacional moderno. Dessa forma, o Blender ainda não atrai o interesse de um grande leque de empresas como o Apache ou o Linux, por exemplo, atraem. Além disso, a área de modelagem e animação tridimensional ainda é dominada por empresas de software proprietários, não tendo o Blender ainda alcançado uma boa colocação neste mercado.

Em relação aos “*open projects*”, podemos afirmar que estes se adequam perfeitamente à área de atuação do Blender, visto que este é um software que tem seu valor pela utilização na produção de conteúdo. A idéia dos “*open projects*”, constitui-se, portanto, numa estratégia bastante inteligente da Blender Foundation, não sendo possível aplicá-la a softwares como o GNOME, o Mozilla Firefox ou o Apache, por exemplo.

Dessa forma, concluímos que a comunidade do Blender soube utilizar de forma adequada as especificidades do software na elaboração das estratégias de financiamento. Consideramos, portanto, os “*open projects*” como uma estratégia inovadora e que pode servir de modelo para outros softwares livres de áreas afins à do Blender.

### **3.5.3 A participação dos usuários do Blender na evolução do software**

Conforme afirmou Hexsel (2002), a qualidade dos softwares desenvolvidos sob o modelo bazar deve-se à quantidade de usuários e desenvolvedores que se envolvem no processo. Neste sentido, além de um bom número de desenvolvedores e de estratégias de financiamento, outro fator que pode influenciar bastante o ritmo de desenvolvimento do software é a participação dos usuários neste processo.

No caso do Blender, os usuários podem colaborar de diversas maneiras: reportando erros, ajudando na documentação, contribuindo na programação de scripts ou do código do software, e com a compra de artigos na loja virtual e realização de doações.

Na pesquisa de campo realizada com os usuários do Blender, procuramos saber se os usuários fazem alguma colaboração com a comunidade do software. Neste contexto, 59,5% dos usuários participantes da pesquisa afirmaram que contribuem de alguma das formas supra-citadas, enquanto que 40,5% afirmaram utilizar o Blender sem realizar nenhuma contribuição à comunidade. Este resultado mostra que há uma participação ativa dos usuários no processo de desenvolvimento do software, o que constitui-se como mais um fator para explicar o ganho de qualidade do Blender desde o momento em que o código fonte deste foi aberto.

Assim, concluímos que o Blender já pode ser considerado como uma alternativa de alta qualidade técnica na produção de animações tridimensionais, visto que este possui todos os principais recursos necessários a um ambiente de animação digital e que a grande maioria dos usuários entrevistados o avaliam positivamente, conforme demonstrado na nossa pesquisa.

Além disso, o Blender pode ser visto também como mais uma demonstração do potencial do modelo de desenvolvimento “bazar” e do software livre para a produção de softwares de alta qualidade técnica. Após o Blender ser disponibilizado como software livre, formou-se uma grande comunidade de programadores que conseguiram dar continuidade ao desenvolvimento do programa de maneira bastante satisfatória. Além da participação de muitos desenvolvedores, outros dois fatores podem ter contribuído para que o Blender alcançasse um alto nível técnico: a grande colaboração dos usuários com a comunidade e a adoção de estratégias de financiamento, as quais têm garantido recursos para promover a utilização e o desenvolvimento do Blender.

## Considerações Finais

Este trabalho teve como objetivo principal compreender os fatores que contribuíram para o sucesso do Blender, enquanto software livre desenvolvido de forma colaborativa. Não pretendemos com essa pesquisa esgotar todos os aspectos deste tema, visto que, para uma melhor elucidação desta questão, faz-se necessário um trabalho de pesquisa mais aprofundado. Entretanto, acreditamos ter contribuído para o entendimento das questões envolvidas no processo de desenvolvimento do Blender.

Desta forma, vemos que o Movimento Software Livre surge com o intuito de defender a liberdade no uso dos computadores e recuperar o antigo costume de hackers compartilharem códigos de software entre si. Richard Stallman foi quem estabeleceu os primeiros esforços em prol do software livre, iniciando o desenvolvimento dos softwares necessários para um sistema operacional livre, entre outros esforços para garantir à qualquer pessoa a liberdade de acessar o código fonte dos softwares e de modificá-lo. Podemos considerar a motivação do Stallman bastante ideológica, visto que o objetivo primordial deste não era obter softwares de boa qualidade.

Posteriormente, o software livre mostrou-se também como uma maneira de se produzir softwares bastante eficiente. O desenvolvimento realizado de forma aberta e colaborativa, onde atualizações são lançadas frequentemente e o código fonte é depurado por um grande número de colaboradores, conseguiu produzir bons resultados. Estes podem ser percebidos pela larga utilização de softwares livres nas mais diversas áreas da informática e em diversos locais do mundo, conseqüência, portanto, do alto nível de inovação tecnológica que as comunidades de software livre têm alcançado.

O Blender, por sua vez, marca o estabelecimento do software livre na área de computação gráfica, a qual, como vimos, envolve recursos computacionais bastante complexos e hoje está estritamente relacionada à produção audiovisual. Com isso, o Blender pode ser considerado uma opção de qualidade profissional para a produção de animações 3D, visto os trabalhos que têm sido realizado com essa ferramenta e a boa avaliação que os usuários deste software fizeram em nossa pesquisa de campo.

Em relação ao desenvolvimento do Blender, podemos considerar que alguns fatores podem ter influenciado de forma importante a maneira como este evoluiu tecnicamente desde que foi disponibilizado como software livre. Em primeiro lugar, após ser liberado sob a licença GNU GPL, a quantidade de programadores dedicados ao Blender atingiu números bastantes superiores aos da época em que este tinha o código fonte fechado. Além disso, os resultados da nossa pesquisa de campo sugerem que uma boa parcela dos usuários contribuem de alguma maneira para a evolução do software. Por fim, pudemos perceber que a comunidade do Blender possui algumas estratégias de financiamento cujos resultados podem ser considerados bastante satisfatórios, visto que tais estratégias têm garantido recursos para a promoção e o desenvolvimento do software.

No entanto, para podermos conhecer exatamente o nível de influência de cada um destes fatores, faz-se necessária uma pesquisa mais aprofundada, estabelecendo relações com a maneira como outros softwares livres evoluíram.

## Bibliografia Utilizada

BRITO, Allan. *Blender 3D: Guia do Usuário*. São Paulo: Novatec, 2007.

CAMARGO, J. T. F. *Tutorial: Fundamentos da animação modelada por computador*. 1995. Disponível em: [http://www.dca.fee.unicamp.br/projects/prosim/publiPS/T\\_SIB95.zip](http://www.dca.fee.unicamp.br/projects/prosim/publiPS/T_SIB95.zip) Acesso em: 07/03/2008.

CASTELLS, Manuel. *A galáxia da internet: reflexões sobre a internet, os negócios e a sociedade*. Rio de Janeiro: Jorge Zahar Ed., 2003. Págs. 13 a 55.

GUESSER, Adalto H. . *O movimento internacional pela adoção do software livre e as motivações de disputas acerca de controvérsias tecnocientíficas: um estudo teórico*. Sociedade e cultura, Goiânia, v. 7, n. 2, 2004. Disponível em: <http://br.geocities.com/aguesser/publications/soccultura.pdf> . Acesso em 27/02/08.

HEXSEL, Roberto A. *Propostas de ações de governo para incentivar o uso de software livre*. Relatório Técnico do Departamento de Informática da UFPR, 004/2002, Curitiba: UFPR, 2002. Disponível em: [http://www.inf.ufpr.br/info/techrep/RT\\_DINF004\\_2002.pdf](http://www.inf.ufpr.br/info/techrep/RT_DINF004_2002.pdf). Acesso em 27/03/08.

Imasters. *BB termina 2007 com 50 mil estações com Linux*. 2008. Disponível em: [http://imasters.uol.com.br/noticia/7922/pesquisas/bb\\_termina\\_2007\\_com\\_50\\_mil\\_estacoes\\_com\\_linux/](http://imasters.uol.com.br/noticia/7922/pesquisas/bb_termina_2007_com_50_mil_estacoes_com_linux/) . Acesso em 15/02/2008.

LUCENA, Alberto. *Arte da animação: técnica e estética através da história*. São Paulo: Editora Senac, 2005.

MELO, Andrei K.; SILVEIRA, Walter D.. *Técnicas de animação em ambientes 3D*. 2007. Disponível em: [http://www.ceart.udesc.br/revista\\_dapesquisa/volume1/numero2/design/animacao\\_ambientes\\_3D.pdf](http://www.ceart.udesc.br/revista_dapesquisa/volume1/numero2/design/animacao_ambientes_3D.pdf). Acesso em: 21/03/08.

MORENO, Antônio. *A experiência brasileira no cinema de animação*. Rio de Janeiro: Artenova, 1978. Págs. 01-26.

PATMORE, Chris. *The Complete Animation Course*. New York: Barron's, 2003.

PRADO, C.; CAMINATI, F. & NOVAES, T. *Sinapse XXI: Novos Paradigmas em Comunicação*. In: *Mídias Digitais: Convergência Tecnológica e Inclusão Social*. São Paulo: Paulinas, 2005.

RANGEL, Ricardo. *Passado e futuro da era da informação*. Rio de Janeiro: Editora Nova Fronteira, 1999.

RAYMOND, Eric. *A catedral e o bazar*. Tradução de Erik Kohler. 1998. Disponível em: <http://www.geocities.com/CollegePark/Union/3590/pt-cathedral-bazaar.html> . Acesso em 20/06/07.

ROWE, Robin. *DreamWorks Animation "Shrek the Third": Linux Feeds an Ogre*. 2007. Disponível em: <http://www.linuxjournal.com/article/9653> . Acesso em 16/02/2008.

SILVEIRA, Sérgio Amadeu. *Inclusão digital, software livre e globalização contra-hegemônica*. In: SILVEIRA, S. A. & CASSINO, J. (orgs.) *Software Livre e Inclusão Digital*. São Paulo: Conrad, 2003.

STALLMAN, Richard. *Free Software, Free Society: selected essays of Richard M. Stallman*. Boston: GNU Press, 2002.

WERNECK, Daniel. *Estratégias Digitais para o Cinema de Animação Independente*. - 2005. Dissertação de Mestrado. Universidade Federal de Minas Gerais. Escola de Belas Artes.

#### **Sites Consultados:**

<http://www.blender.org>

<http://peach.blender.org>

<http://apricot.blender.org>

<http://www.elephantsdream.org/>

<http://www.linux-foundation.org/>

<http://apache.org/>

<http://www.kde.org/>

<http://www.gnome.org/>

<http://www.fsf.org/>

<http://www.gnu.org/>

<http://www.openoffice.org/>

<http://www.mozilla.org/foundation/>

## Apêndices

### Apêndice A – Formulário utilizado na pesquisa de campo<sup>51</sup>

#### Perfil

Nome:

E-mail:

Idade:

País onde reside:

Profissão:

Descreva brevemente qual tem sido a sua relação com computação gráfica e animação:

#### Relação com o Blender

**1. Defina, do mais importante para o menos importante, os motivos que o levaram a utilizar o Blender?**

- O fato de ser software livre
- Ausência de custos com licenças (software gratuito)
- Qualidade técnica avançada

Outros:

**2. Há quanto tempo você utiliza o Blender?**

- Menos de seis meses
- Entre seis meses e um ano
- Mais de um ano
- Mais de três anos

**3. Qual/quais a(s) forma(s) como você colabora com o Blender?**

- Relato de bugs (bug reports)
- Compra de artigos (livros, dvd's, etc) na loja virtual do Blender
- Contribuição na documentação do Blender

---

<sup>51</sup> Também foi disponibilizado na internet uma versão em inglês deste formulário.

- ( ) Desenvolvimento do código-fonte e de scripts
- ( ) Ainda não colaboro com o Blender, apenas utilizo-o

**4. Como você avalia o nível técnico do Blender?**

- ( ) Muito Bom
- ( ) Bom
- ( ) Razoável
- ( ) Ruim

**5. Em relação à qualidade técnica do Blender, quais aspectos você considera mais positivos?**

**6. E quais os aspectos tecnicamente mais negativos do Blender?**

**7. Você utiliza outros softwares proprietários similares ao Blender?**

- ( ) Não
- ( ) Sim . Quais:

**8. Qual o motivo da utilização desses softwares e não do Blender?**