

**UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE GEOCIÊNCIAS
PROGRAMA DE PÓS-GRADUAÇÃO EM ANÁLISE E
MODELAGEM DE SISTEMAS AMBIENTAIS**

TerraME GIMS – Uma interface gráfica para descrição de
modelos ambientais para a plataforma *TerraME*

Tiago França Melo de Lima

Seminário de Dissertação do Curso de Pós-Graduação em Análise e Modelagem de
Sistemas Ambientais.

Sérgio Donizete Faria
(orientador)
Tiago Garcia de Senna Carneiro
(co-orientador)

UFMG

2009

RESUMO

A intensificação das mudanças ambientais causadas pelas atividades antrópicas tem levado ao desenvolvimento e utilização de diversas plataformas de *software* para modelagem e simulação dos processos e fenômenos antrópicos e biofísicos e suas interações. O *TerraME* é um ambiente para modelagem e simulação de processos dinâmicos espaciais. Através de uma linguagem de programação de alto nível, também denominada *TerraME*, oferece uma base conceitual e serviços que permitem a seus usuários representar estruturas de dados e regras que irão definir o comportamento do modelo. Os principais usuários do ambiente *TerraME* são pesquisadores e profissionais com formações diversas tais como biólogos, engenheiros, ecólogos, economistas, sociólogos que muitas vezes não possuem uma conhecimentos básicos algoritmos e técnicas de programação. A utilização direta de uma linguagem de programação portanto é um fator limitante para o amplo potencial de utilização do ambiente *TerraME*, sendo portanto necessário um nível maior de abstração, permitindo que os usuários se concentrem na resolução dos problemas pertencentes ao domínio de aplicação dos modelos, e ignorem os problemas envolvidos na sua representação computacional. Este trabalho propõe o desenvolvimento do *TerraME Graphical Interface for Modelling and Simulation – TerraME GIMS*, uma interface gráfica com o usuário que irá fornecer os recursos necessários à modelagem e simulação de sistemas ambientais utilizando o ambiente *TerraME* por meio de metáforas visuais que representem graficamente o modelo.

SUMÁRIO

RESUMO	2
SUMÁRIO	3
LISTA DE FIGURAS	4
LISTA DE SIGLAS E ABREVIATURAS.....	5
1 INTRODUÇÃO	6
1.1 CONTEXTUALIZAÇÃO DO TRABALHO	6
1.2 OBJETIVOS	7
1.2.1 <i>Objetivos gerais</i>	7
1.2.2 <i>Objetivos específicos</i>	8
1.3 JUSTIFICATIVAS	8
1.4 ESTRUTURA DO DOCUMENTO.....	9
2 REVISÃO DA LITERATURA	10
2.1 CONCEITOS BÁSICOS SOBRE MODELAGEM E SIMULAÇÃO	10
2.2 O AMBIENTE DE MODELAGEM <i>TERRAME</i>	11
2.2.1 <i>Arquitetura do TerraME</i>	12
2.2.2 <i>A linguagem de modelagem TerraME</i>	13
2.3 AMBIENTES DE DESENVOLVIMENTO INTEGRADO	13
2.4 A PLATAFORMA <i>ECLIPSE</i>	14
2.4.1 <i>Eclipse Modeling Framework</i>	16
2.4.2 <i>Graphical Editing Framework</i>	17
3 DESENVOLVIMENTO DO <i>TERRAME GIMS</i>.....	18
3.1 METODOLOGIA DE DESENVOLVIMENTO	18
3.2 ESPECIFICAÇÃO DE REQUISITOS	18
3.2.1 <i>Escopo do sistema</i>	18
3.2.2 <i>Requisitos Funcionais</i>	18
3.2.3 <i>Requisitos não-funcionais</i>	21
3.3 ARQUITETURA DO <i>TERRAME GIMS</i>	22
4 AVALIAÇÃO DO <i>TERRAME GIMS</i>	24
4.1 IDENTIFICAÇÃO E ANÁLISE DO ATUAL PERFIL DA COMUNIDADE DE USUÁRIOS DO <i>TERRAME</i>	24
4.2 AVALIAÇÃO DO <i>TERRAME GIMS</i>	24
5 EXEMPLOS DE APLICAÇÕES	25
6 RESULTADOS.....	26
6.1 RESULTADOS PARCIAIS	26
6.2 RESULTADOS ESPERADOS.....	28
6.3 PLANO DE EXECUÇÃO	28
6.4 CONSIDERAÇÕES PARCIAIS.....	30
7 BIBLIOGRAFIA.....	31

LISTA DE FIGURAS

Figura 1 – Processo cíclico de desenvolvimento de modelos.....	11
Figura 2 – Arquitetura do <i>TerraME</i>	12
Figura 3 – Visão geral do <i>Eclipse</i> SDK.	15
Figura 4 – Arquitetura em camadas do <i>Eclipse</i>	15
Figura 5 – Visão geral do GEF sobre o padrão MVC.....	17
Figura 6 – Modelagem gráfica de autômatos.....	19
Figura 7 – Requisitos de interface do <i>TerraME GIMS</i>	21
Figura 8 – Arquitetura em camadas do <i>TerraME GIMS</i>	23
Figura 9 – Arquitetura em camadas do <i>TerraME GIMS</i>	27
Figura 10 – Visualização hierárquica de um modelo <i>TerraME GIMS</i>	28

LISTA DE SIGLAS E ABREVIATURAS

API – *Application Programming Interface*

EMF – *Eclipse Modeling Framework*

GEF – *Graphical Editing Framework*

GUI – *Graphical User Interface*

IDE – *Integrated Development Environment*

JAR – *Java Archive*

JDT – *Java Development Tools*

JVM – *Java Virtual Machine*

MDA – *Model Driven Architecture*

MVC – *Model View Controller*

PDE – *Plug-in Development Environment*

SDK – *Software Development Kit*

SIG – *Sistema de Informações Geográficas*

SWT – *Standard Widget Toolkit*

TerraME – *TerraLib Modeling Environment*

XMI – *XML Metadata Interchange*

XML – *Extensible Markup Language*

UML – *Unified Modeling Language*

URL – *Uniform Resource Locator*

WYSIWYG – *What You See Is What You Get*

1 INTRODUÇÃO

Neste capítulo são apresentados o contexto no qual se insere este trabalho, os objetivos do trabalho e motivações que levaram ao seu desenvolvimento, e como este documento está organizado.

1.1 Contextualização do trabalho

A intensificação das mudanças ambientais causadas por processos antrópicos exige respostas cada vez mais rápidas por parte de pesquisadores, empresários e governantes, e a necessidade de uma maior compreensão acerca do funcionamento dos sistemas terrestres viabilizando intervenções de menor impacto ao meio ambiente. A interação dos sistemas sócio-econômicos (de origem antrópica: sistema de uso do solo, sistema de transporte urbano, sistema financeiro) com os sistemas biofísicos (de origem natural: sistema climático, ecossistemas aquáticos e terrestres) é um processo extremamente complexo, e seu entendimento exige um trabalho multidisciplinar, envolvendo especialistas e profissionais das mais diversas áreas de conhecimento.

Diversas plataformas de software, como por exemplo NetLogo¹, Swarm², Repast³, Vensim⁴, Dinâmica⁵, tem sido utilizadas para modelar e simular processos antrópicos e biofísicos e suas interações. Muitos são os aspectos que diferenciam estas plataformas, tais como a base teórica a partir da qual foram desenvolvidas e as funcionalidades que oferecem aos usuários (e conseqüentemente limitações).

O *TerraME – TerraLib Modelling Environment* (Carneiro, 2003), é um ambiente de modelagem e simulação de processos ambientais com representação explícita no espaço. Componente da família de soluções *TerraLib* (Câmara et al., 2000), apresenta uma linguagem de programação de alto nível, também chamada *TerraME* que permite ao modelador (usuário) representar as estruturas e dados e as regras que regerão o comportamento dos modelos de forma mais clara e eficiente quando comparada à linguagens de programação de uso geral, como C++ (Stroustrup, 1994) ou Java⁶ (Gosling et al., 1996). Porém, a assimilação dos conceitos e das construções existentes na linguagem *TerraME* ainda é uma tarefa que apresenta acentuada curva de aprendizado para profissionais e pesquisadores não familiarizados com algoritmos e técnicas de programação, conforme constatado em avaliações realizadas por participantes de cursos e apresentações sobre o ambiente *TerraME* (Câmara et al., 2007; Câmara et al., 2008). Entretanto, são

¹ <http://ccl.northwestern.edu/netlogo/>

² <http://www.swarm.org/>

³ <http://repast.sourceforge.net/>

⁴ <http://www.vensim.com/>

⁵ <http://www.csr.ufmg.br/dinamica/>

⁶ <http://java.sun.com>

esses os profissionais mais interessados no desenvolvimento dos modelos ambientais e que detêm o conhecimento a respeito do domínio de aplicação dos modelos. Dentre estes profissionais podemos citar geólogos, biólogos, ecólogos, geógrafos e sociólogos, por exemplo. Desta maneira, o uso direto e obrigatório de uma linguagem de programação para representação de modelos dinâmicos constitui-se como a principal barreira para difusão e ampliação da sua comunidade de usuários.

Diversos trabalhos já utilizaram o ambiente *TerraME* como plataforma de modelagem e simulação (Andrade et al., 2009; Almeida et al., 2008; Carneiro et al., 2008; Moreira et al., 2008; Pimenta et al., 2008). Acredita-se no entanto que o atual número de usuários é pequeno tendo em vista os recursos e funcionalidades oferecidos pelo ambiente e o seu potencial de aplicação. A crescente demanda por aplicações e ferramentas para o estudo das ciências terrestres em virtude principalmente dos atuais processos de mudanças climáticas também contribuem para o grande potencial de crescimento de utilização do ambiente.

Para permitir que os usuários do *TerraME* se concentrem na resolução de problemas pertencentes ao domínio de aplicação dos modelos, e ignorem a maioria dos problemas envolvidos na sua representação computacional, um novo e mais alto nível de abstração é necessário. A representação dos modelos através de componentes gráficos como diagramas, ao invés de algoritmos, deve tornar mais intuitivo e eficiente os processos de modelagem e simulação de sistemas ambientais, aumentando a produtividade dos atuais usuários do ambiente *TerraME* e diminuindo sobremaneira a curva de aprendizado percebida por seus novos usuários.

1.2 Objetivos

1.2.1 Objetivos gerais

Para construir modelos utilizando o *TerraME* os usuários podem utilizar qualquer editor de textos, sendo necessários para a execução do modelo a instalação do interpretador *TerraME* e a configuração do editor. As limitações apresentadas na utilização de editores não específicos para a construção de modelos para o ambiente *TerraME* e a dificuldade de muitos usuários em descrever estes modelos na forma de algoritmos dificultam sua utilização. Neste sentido, a disponibilização de um ambiente de desenvolvimento integrado específico para a plataforma *TerraME*, que ofereça funcionalidades para a criação e visualização dos modelos, faz-se necessária. Um ambiente desta natureza tornará mais ágil o processo de desenvolvimento e análise de modelos, aumentará a troca de informações e colaboração entre os usuários (uma vez que estes utilizarão o mesmo ambiente de desenvolvimento), além de tornar mais intuitivo o processo de construção de modelos para sistemas ambientais, fornecendo então subsídios que permitirão o aumento da comunidade de usuários e a

adoção e consolidação do ambiente *TerraME* como plataforma para modelagem e simulação de sistemas terrestres.

1.2.2 Objetivos específicos

Este trabalho propõe o desenvolvimento do *TerraME GIMS – TerraME Graphical Interface for Modelling and Simulation*, uma interface gráfica para a construção de modelos dinâmicos espaço-temporais e simulação baseados no ambiente *TerraME*. Esta ferramenta disponibilizará funcionalidades e recursos que viabilizarão a construção de modelos sem exigir que o modelador possua um conhecimento profundo sobre programação, permitindo que estes sejam descritos de forma gráfica, através de diagramas e da utilização de *widgets*⁷, facilitando a interação modelo/usuário. A especificação dos objetos a constituir um modelo *TerraME (Model)* – ambiente (*Environment*), espaço (*CellularSpace*), comportamento (*Automaton*) e tempo (*Timer*), serão feitas por meio de uma interface gráfica. Somente a especificação das regras que irão reger o comportamento do modelo exigirá do modelador algum conhecimento sobre algoritmos.

Desta forma, é esperado que o *TerraME GIMS* torne mais eficiente e intuitivo o processo de desenvolvimento de modelos ambientais, uma vez que abstrairá muitos dos conceitos de programação através de elementos visuais (ícones, figuras, diagramas etc.) a partir das quais o código correspondente do modelo será gerado automaticamente; contribuindo por fim para o aumento da comunidade de usuários do ambiente *TerraME*.

Um levantamento e estudo com o objetivo de traçar um perfil da atual comunidade de usuários do *TerraME* também será realizado. Este permitirá identificar as principais dificuldades enfrentadas pelos atuais usuários, as principais áreas de pesquisa em que o ambiente vem sendo utilizado, a abrangência de sua utilização em termos de áreas de pesquisa e de usuários/instituições, além de tornar possível acompanhar a evolução da comunidade de usuários.

1.3 Justificativas

Apesar da linguagem de modelagem *TerraME* facilitar a representação de modelos dinâmicos espaciais integrados a sistemas de informações geográficas (SIG's), ela apresenta uma acentuada curva de aprendizado devido à complexidade inerente aos conceitos que ela implementa e aos fenômenos ambientais (geográficos, biofísicos, sociais) aos quais se aplica. Profissionais sem conhecimentos básicos sobre algoritmos e técnicas de programação apresentam muitas dificuldades na sua utilização, conforme foi constatado em avaliações realizadas por participantes de cursos e

⁷ *Widget* é um termo sem tradução que pode ser entendido como componente de interface gráfica que permite a interação entre o *software* e o usuário.

apresentações sobre o ambiente *TerraME* (Câmara et al., 2007; Câmara et al., 2008). Este é exatamente o caso da maioria dos profissionais que se ocupam do estudo das interações entre os sistemas socioeconômicos e biofísicos: geógrafos, ecólogos, biólogos, antropólogos, sociólogos, economistas etc. Assim uma ferramenta que abstraia detalhes computacionais de implementação do processo de desenvolvimento de modelos ambientais deve encontrar respaldo na comunidade de modeladores, e permitir que estes se concentrem nos aspectos relacionados ao domínio do problema.

1.4 Estrutura do documento

Neste capítulo (Introdução), é apresentada uma breve contextualização do trabalho, os objetivos e justificativas, e a organização do documento, aqui apresentada. No Capítulo 2 é apresentada uma breve revisão da literatura; no Capítulo 3 são abordados aspectos relacionados ao desenvolvimento do *software TerraME GIMS*; o Capítulo 4 introduz como será realizada a avaliação do *TerraME GIMS*; no Capítulo 5 são apresentados alguns exemplos de aplicação a serem implementados; e no Capítulo 6 são apresentados os resultados parciais e esperados, um plano de execução das atividades inerentes ao desenvolvimento da dissertação e as considerações parciais. A bibliografia é listada no Capítulo 7.

2 REVISÃO DA LITERATURA

Este capítulo traz uma breve revisão da literatura apresentando alguns conceitos e ferramentas necessários ao entendimento e desenvolvimento deste trabalho, como: modelagem e simulação, o ambiente *TerraME*, ambiente de desenvolvimento integrado e a plataforma Eclipse.

2.1 Conceitos básicos sobre modelagem e simulação

O processo de modelagem consiste em produzir representações (modelos) da estrutura e/ou funcionamento de um sistema específico com o objetivo de melhor compreender a realidade observada. Esta compreensão permite, por meio de análises, a elaboração de estimativas sobre o comportamento futuro desse sistema. A simulação de um sistema é a execução de um modelo matemático-computacional que o represente (Maria, 1997). É utilizada quando a solução de problemas possui um custo muito elevado ou até mesmo impossível de ser obtida por meio de experimentos. A simulação permite que problemas muito complexos sejam tratados de forma analítica pelo modelador.

Muitos problemas de natureza complexa exigem um tratamento interdisciplinar, envolvendo pessoas das mais variadas áreas do conhecimento. Por exemplo, a criação de um modelo para o processo de desmatamento da Floresta Amazônica deveria envolver biólogos, ecólogos, sociólogos, economistas, geógrafos, políticos, tecnólogos etc.

A representação de um modelo deve ser de comum entendimento para todas as pessoas envolvidas no processo de modelagem, sendo específica o suficiente para facilitar a representação do conhecimento que cada um tem a respeito do sistema, e geral o suficiente para não restringir a representação do modelo a um domínio de aplicação específico. Desta forma, a escolha da linguagem a ser utilizada na representação de um modelo é fator crucial para o sucesso de um projeto de modelagem que envolva a colaboração de profissionais de diferentes formações. Ademais, o desenvolvimento de modelos computacionais exige que a linguagem escolhida seja livre de ambigüidades para permitir a representação e execução dos modelos por um computador.

A modelagem e simulação de sistemas tem sido utilizadas nos mais diversos campos científicos na busca de um melhor entendimento do objeto de estudo e maior compreensão da realidade observada. A Ciência dos Sistemas Terrestres, devido à sua natureza inerente complexa, exige a utilização de técnicas e ferramentas de modelagem e simulação para representar e estudar o funcionamento dos processos antrópicos e biofísicos e suas interações.

De acordo com Carneiro (2003) o processo de modelagem de fenômenos espaciais dinâmicos ocorre de forma cíclica, conforme ilustrado na Figura 1, compreendendo as seguintes

fases: (a) desenvolvimento da base de dados; (b) desenvolvimento do modelo; (c) calibração, verificação e validação do modelo; (d) execução e visualização do modelo, e análise de relatórios; (e) projeção de cenários. As atividades de desenvolvimento de modelos para sistemas terrestres e as atividades de desenvolvimento de software baseado no modelo espiral (Boehm, 1988) são essencialmente similares, e envolvem as seguintes “macro-atividades”: concepção, *design*, implementação e testes.

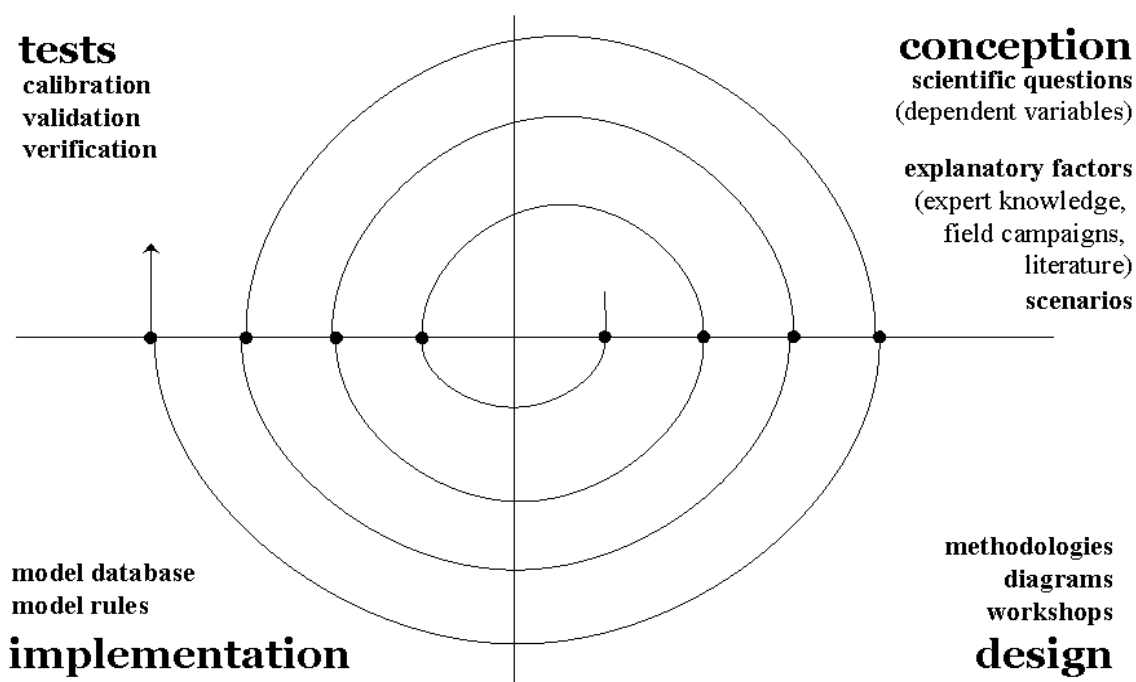


Figura 1 – Processo cíclico de desenvolvimento de modelos.

Fonte: Carneiro, 2003

Diversas plataformas de software, como por exemplo *Swarm* (Minar et al., 1996), *Stella* (Anderson et al., 1983), *RePast* (North et al., 1995), vem sendo utilizadas no desenvolvimento de modelos e simulações de processos dos sistemas terrestres. Porém muitas destas não foram projetadas para este propósito específico, apresentando algumas limitações em sua utilização para esta finalidade (Carneiro, 2003). O *TerraME*, apresentado na Seção 2.2, foi especificamente projetado para permitir a representação e simulação de processos espaciais dinâmicos, fornecendo uma série de serviços e estruturas de dados para tal.

2.2 O ambiente de modelagem *TerraME*

O *TerraME* (*TerraLib Modelling Environment*) (Carneiro, 2003) é um componente da família de soluções *TerraLib* (Câmara et al., 2000) para a implementação e simulação de modelos ambientais que envolvam a representação explícita do espaço. Desenvolvido a partir de um trabalho de doutorado no Instituto Nacional de Pesquisas Espaciais (INPE) concluído em 2003, foi disponibilizado para *download* e utilização a partir de um *website*. Em contínuo desenvolvimento

desde então, é atualmente mantido pelo *TerraLAB* (Laboratório para Modelagem e Simulação de Sistemas Terrestres), uma parceria entre o INPE e a Universidade Federal de Ouro Preto, que possui como missão o projeto e desenvolvimento de um ambiente livre de modelagem e simulação, a Plataforma *TerraME* (Carneiro et al., 2009).

O *TerraME* provê mecanismos que permitem a fácil representação e a eficiente simulação de modelos espaciais dinâmicos integrados a um Sistema de Informações Geográficas (SIG). Os componentes de sua arquitetura de *software* oferecem serviços específicos a usuários de diferentes níveis. Usuários experientes podem implementar modelos utilizando diretamente o código fonte da arquitetura através da linguagem de programação C++, enquanto aqueles que possuem apenas o conhecimento básico sobre algoritmos e modelagem computacional podem utilizar a linguagem de programação de alto nível *TerraME Modeling Language* – uma extensão da linguagem de programação *LUA* (Ierusalimsky et al., 1996), que permite a fácil escrita, leitura e alteração dos modelos (Carneiro, 2003).

2.2.1 Arquitetura do *TerraME*

O *TerraME* foi construído baseado na arquitetura em camadas, onde as camadas inferiores fornecem funcionalidades sobre as quais as camadas superiores são implementadas, conforme mostrado na Figura 2.

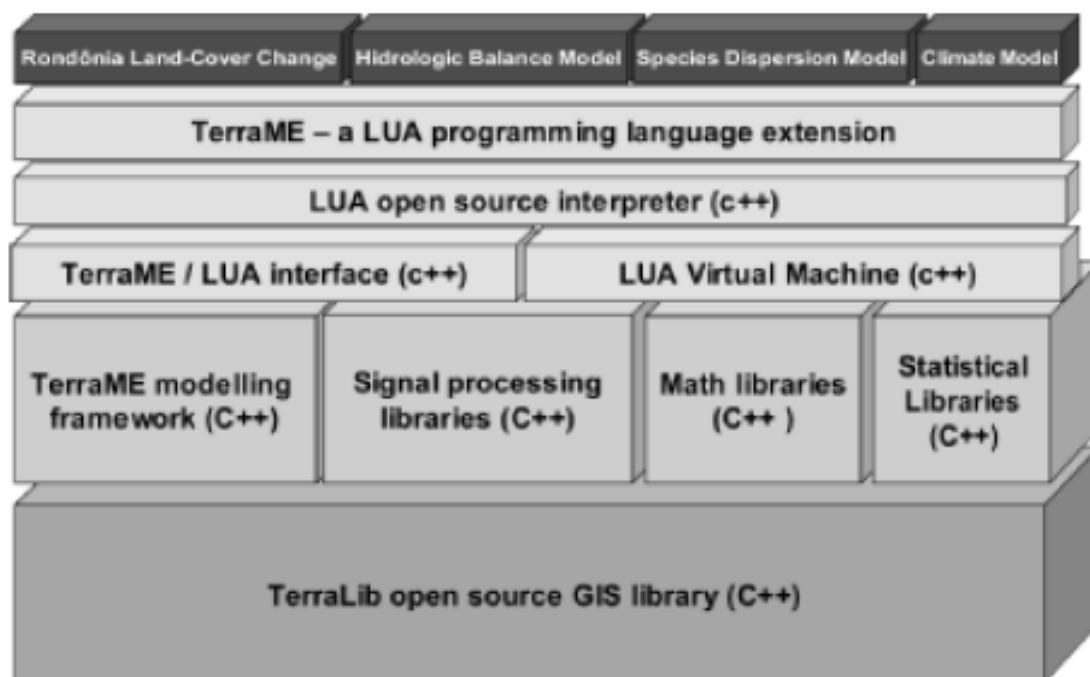


Figura 2 – Arquitetura do *TerraME*.

Fonte: Carneiro, 2003

Conforme definido por Carneiro (2003), na camada inferior, a biblioteca *TerraLib* (Câmara et al., 2000) fornece serviços para gerenciamento e análise de dados espaço-temporais. Na segunda camada, o *TerraME Modeling Framework* fornece serviços para a simulação, calibração e validação de modelos que podem ser utilizados através da linguagem C++. Nesta camada, o modelador tem acesso a um poderoso conjunto de serviços para modelagem e simulação, como por exemplo, a não restrição quanto ao tipo de representação espacial utilizada para modelar objetos espaciais – as células podem ser *pixels* em uma imagem, triângulos em um modelo digital de elevação, ou pontos, linhas e polígonos em um mapa vetorial. Contudo, sua API (*Application Programming Interface*) apresenta uma sintaxe complexa, e por isso, tendenciosa a erros de programação. A camada seguinte é aquela formada pelo interpretador e pelo ambiente de execução da linguagem de modelagem *TerraME* (Seção 2.2.2), que estende a linguagem de programação *LUA* pela inclusão de novos tipos de dados, especialmente projetados para a modelagem espacial dinâmica, e de serviços para simulação e avaliação de modelos. Sobre as demais, está a camada de aplicação, composta por modelos ambientais desenvolvidos pelos usuários da arquitetura.

2.2.2 A linguagem de modelagem *TerraME*

A linguagem de programação *TerraME Modeling Language* permite a representação de modelos espaciais dinâmicos a partir dos seguintes tipos de dados:

- *Environment*: utilizado para representar o conceito de ambiente (escala) e permitir o desenvolvimento de modelos que considerem múltiplas escalas;
- *CellularSpace*, *Cell*, *Neighborhood*: utilizados para representar o espaço, suas propriedades e relações topológicas;
- *Agent*, *Automaton*, *State*, *Jump*, *Flow* e *Trajectory*: utilizados para representar o comportamento de sistemas – processos ou atores que dinamicamente alteram as propriedades do espaço e interação entre si;
- *Timer*, *Event* e *Message*: utilizados para representar o tempo e definir o momento e a ordem na qual os modelos comportamentais serão executados.

2.3 Ambientes de desenvolvimento integrado

O desenvolvimento de um sistema de computação ou de um modelo computacional para representar fenômenos naturais ou antrópicos são atividades essencialmente similares. Tanto o sistema de computação quanto o modelo precisam ter seus requisitos identificados, suas arquiteturas e estruturas especificadas e seus comportamentos implementados na forma de regras expressas em alguma linguagem de programação. É comum estes dois tipos de projetos envolverem uma equipe

de desenvolvimento multidisciplinar e numerosa, algumas vezes dispersa geograficamente. Desta maneira, os atuais ambientes de desenvolvimento integrado (ou IDE, do inglês *Integrated Development Environment*) amplamente utilizados na indústria de desenvolvimento de *software* poderiam ser também utilizados para o desenvolvimento de modelos ambientais.

Um IDE é um ambiente cujas características e funcionalidades buscam agilizar o processo de desenvolvimento de *software*. Geralmente utilizam técnicas de RAD⁸ (*Rapid Application Development*) para permitir maior produtividade e eficiência no desenvolvimento. Frequentemente permitem a representação de projetos de *software* através de diagramas, fornecendo funcionalidades que automatizam o processo de implementação do *software*, gerando automaticamente o código-fonte da aplicação a partir dos diagramas. Também podem fornecer serviços que organizam e facilitam a colaboração da equipe de desenvolvimento.

O desenvolvimento de um ambiente desta natureza é uma tarefa extremamente dispendiosa. Uma alternativa ao desenvolvimento de um IDE totalmente novo para o ambiente *TerraME* é a reutilização de um IDE pré-existente, como por exemplo a plataforma *Eclipse*⁹.

2.4 A plataforma *Eclipse*

O *Eclipse* é um IDE de domínio público desenvolvido como um conjunto de *frameworks* de *software*. Estes *frameworks* podem ser reutilizados para o desenvolvimento de novas aplicações que demandem o desenvolvimento de um IDE que torne eficiente a colaboração entre membros da equipe de desenvolvimento e que permita a especificação do sistema por meio de diagramas e a geração automática de código (Rivieres e Wiegand, 2004; Rivieres e Beaton, 2006).

O grande diferencial da plataforma *Eclipse* é a sua capacidade de integração obtida por meio de uma arquitetura baseada em *plug-ins*. Desenvolver uma aplicação sobre a plataforma *Eclipse* permite que ela seja integrada a outras aplicações também implementadas sobre a plataforma. A plataforma *Eclipse* é construída num mecanismo de descobrir, integrar, e executar módulos chamados *plug-ins* (Rivieres e Beaton, 2006).

Na Figura 3 são mostrados os principais componentes do *Eclipse Software Development Kit* (*Eclipse SDK*) e sua arquitetura baseada em *plug-ins*. O *Eclipse SDK* inclui a plataforma *Eclipse* além de duas ferramentas úteis ao desenvolvimento de *plug-ins*: a *Java Development Tools* (JDT), que implementa um ambiente de desenvolvimento *Java*; e a *Plug-in Development Environment*

⁸ *Rapid Application Development* (RAD), ou Desenvolvimento Rápido de Aplicação, é um modelo de processo de desenvolvimento de *software* iterativo e incremental que se baseia em ciclos de desenvolvimento extremamente curtos. (http://pt.wikipedia.org/wiki/Rapid_Application_Development)

⁹ <http://www.eclipse.org>

(PDE) que adiciona ferramentas específicas para o desenvolvimento de *plug-ins* e extensões. Novas aplicações são desenvolvidas estendendo-se o sistema através de *plug-ins* (Eclipse).

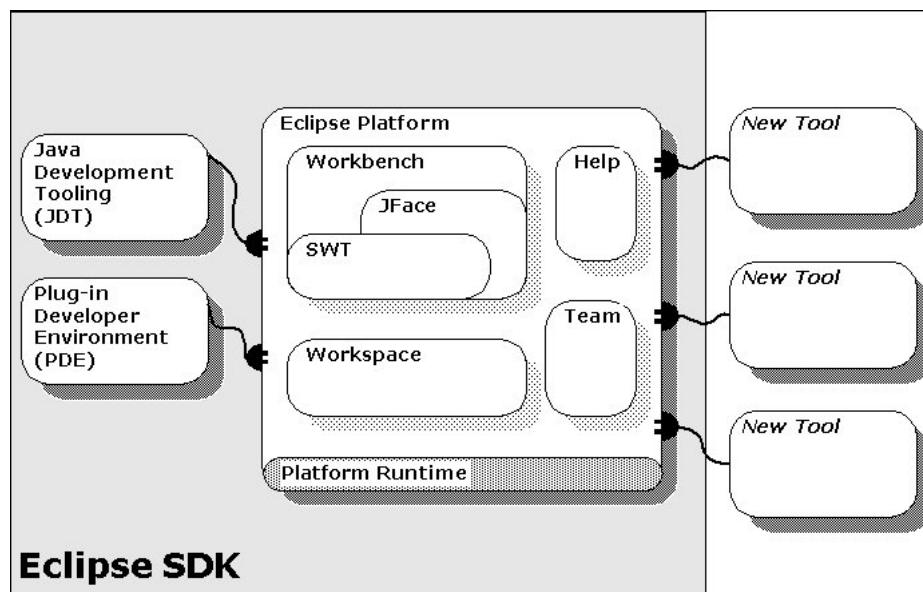


Figura 3 – Visão geral do *Eclipse* SDK.

Fonte: Rivieres e Beaton, 2006

Na Figura 4 é mostrada a arquitetura baseada em camadas da plataforma *Eclipse*. Na primeira camada temos a *Java Virtual Machine* (JVM), ou Máquina Virtual Java, sobre a qual é executada a plataforma *Eclipse*. A camada seguinte (*Platform*) constitui a base da plataforma e suas funcionalidades. Acima desta temos a JDT, oferecendo suporte ao desenvolvimento *Java*, e por último a PDE, constituindo o ambiente de desenvolvimento de *plug-ins*, que permite estender a plataforma acrescentando a ela funcionalidades e ferramentas ou mesmo criar aplicações completas.

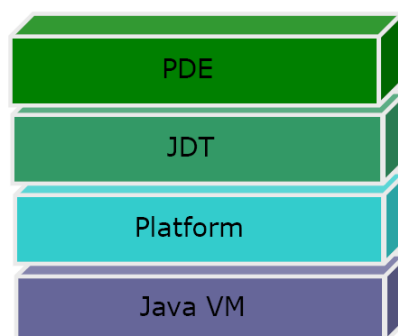


Figura 4 – Arquitetura em camadas do *Eclipse*.

Fonte: TecComm

Um *plug-in* é a menor unidade funcional da plataforma *Eclipse* passível de ser desenvolvida e distribuída separadamente. Exceto por um pequeno *kernel* conhecido como *Platform Runtime*, toda a funcionalidade da plataforma *Eclipse* é disponibilizada por meio de *plug-ins* (Rivieres e

Beaton, 2006). Os *plug-ins* são codificados na linguagem de programação *Java* e são distribuídos na forma de bibliotecas de classes e objetos encapsulados em um *Java Archive* (JAR)

O desenvolvimento do *TerraME GIMS* utilizará ainda dois *frameworks* da plataforma Eclipse: o *Eclipse Modeling Framework* e o *Graphical Editing Framework*; apresentados a seguir nas seções 2.4.1 e 2.4.2:

2.4.1 Eclipse Modeling Framework

O *Eclipse Modeling Framework* (EMF) é um *framework* para modelagem e um instrumento de geração de código (*code generation facility*) que permite construir ferramentas e outras aplicações baseadas em um modelo de dados estruturado (Eclipse EMF, 2005). O EMF faz parte da arquitetura MDA (*Model Driven Architecture*), cuja base é o desenvolvimento de aplicações com foco no modelo (More et al., 2003). O *framework* fornece, a partir de especificações de modelos escritas em XMI (*XML Metadata Interchange*), ferramentas e *runtime* para a geração de um conjunto de classes *Java* correspondente ao modelo, classes adaptadoras que permitem sua visualização e edição, além de um editor básico. Novos modelos podem ser especificados através de interfaces *Java*, documentos XML (*Extensible Markup Language*) ou diagramas UML (*Unified Modeling Language*) e então importadas para o *framework*. Além disso, o EMF fornece serviços para interoperabilidade entre as aplicações cujo desenvolvimento foi nele baseada (Eclipse EMF).

Três elementos fundamentais compõem o *framework*:

- EMF – é o núcleo do *framework*, que inclui a base para o meta-modelo *Ecore*, que permite descrever modelos, e o suporte de sua execução (com notificação de mudanças, suporte a persistência com serialização XMI, API para manipulação de objetos EMF);
- EMF.Edit – inclui classes genéricas reutilizáveis que permitem a construção de editores para os modelos EMF;
- EMF.Codegen – é o instrumento de geração de código EMF; gera código que permite construir editores para modelos EMF e inclui GUI a partir da qual opções podem ser especificadas e geradores podem ser invocados.

O EMF, em sua integração modelo/geração de código, suporta operações de criação, restauração, atualização e remoção, além de suportar restrições de cardinalidade (*cardinality constraints*), relacionamentos complexos e estruturas de herança, definições de contenção (*containment definition*), e um conjunto de descrições de atributos, integrando o modelo ao código gerado (Powell, 2004).

2.4.2 Graphical Editing Framework

O *Graphical Editing Framework* (GEF) é um *framework* que permite a criação de ambientes de edição gráfica a partir de modelos. Baseado na biblioteca SWT (*Standard Widget Toolkit*), consiste de dois *plug-ins*:

- *org.eclipse.draw2d* – fornece um conjunto de ferramentas de *layout* e renderização para visualização de gráficos;
- *org.eclipse.gef* – fornece ferramentas como seleção, criação e conexão; dois tipos de *viewer's* (*graphical viewer* e *tree viewer*); um *framework* de controle (*Controller*) que permite mapear modelos em visões; suporte a comandos desfazer/refazer (*undo/redo*)

O GEF utiliza a arquitetura MVC (*Model-View-Controller*), que permite aplicar mudanças ao modelo a partir do *view*. O GEF é completamente neutro de aplicações e, através dele, é possível criar qualquer editor WYSIWYG (*What You See Is What You Get*). Na Figura 5 podemos observar o modelo de funcionamento do *framework*.

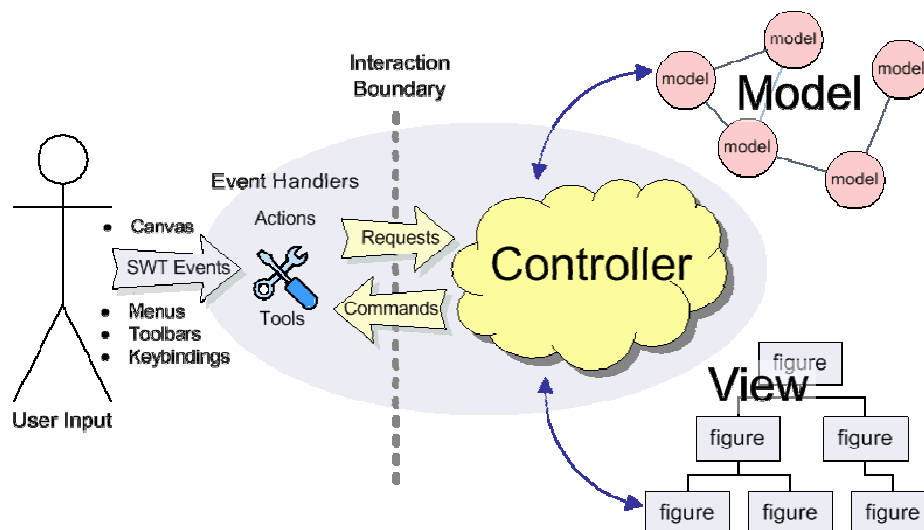


Figura 5 – Visão geral do GEF sobre o padrão MVC.

Fonte: Eclipse

As regras do MVC são aplicadas ao GEF da seguinte forma:

- **model**: o modelo deve possuir algum tipo de mecanismo de notificação, de forma este possa ser modificado pelas ações do usuário;
- **view**: objetos visíveis ao usuário, sendo que tanto figuras quanto elementos de árvore podem ser elementos *view*;
- **controller**: chamado de *EditPart*, faz a ligação entre o *model* e o *view*.

3 DESENVOLVIMENTO DO *TerraME GIMS*

Neste capítulo é apresentada a metodologia a ser utilizada no desenvolvimento do *TerraME GIMS*, os requisitos e arquitetura do *software* e as etapas de desenvolvimento.

3.1 Metodologia de desenvolvimento

O desenvolvimento do *TerraME GIMS* acontecerá de forma incremental e iterativa, em espiral, com prototipação de *releases*¹⁰ e versões, sendo obtida a cada ciclo uma nova versão documentada do *TerraME GIMS*. Serão realizados 3 ciclos de desenvolvimento, sendo cada um constituído de quatro etapas: iniciação, elaboração, construção e transição. Na iniciação serão realizadas atividades de levantamento dos requisitos; na elaboração será feita a modelagem destes requisitos; na fase de construção será realizada a implementação; e na fase de transição, serão realizados os testes e validação dos requisitos, e atividades de preparação para o próximo ciclo. O primeiro ciclo de desenvolvimento já foi realizado e um primeiro protótipo foi implementado (Lima et al, 2008).

3.2 Especificação de Requisitos

3.2.1 Escopo do sistema

Este projeto tem como escopo a construção do *TerraME Graphical Interface for Modeling and Simulation (TerraME GIMS)* – um IDE que permitirá aos usuários construir modelos dinâmicos espaço-temporais baseados na linguagem de modelagem *TerraME* a partir de interações com componentes da interface gráfica do aplicativo, sendo o código *TerraME* correspondente ao modelo gerado automaticamente.

3.2.2 Requisitos Funcionais

O *TerraME GIMS* apresenta os seguintes requisitos funcionais:

- **RF01 – Criação de modelos dinâmicos espaço-temporais:** o *TerraME GIMS* permitirá criar novos modelos correspondentes ao tipo *Model* do *TerraME* através de um menu.
- **RF02 – Criação de escalas:** o *TerraME GIMS* permitirá criar novas “escalas” correspondentes ao tipo *Environment* do *TerraME* e inseri-las em um modelo.
- **RF03 – Criação de escalas aninhadas:** o *TerraME GIMS* permitirá criar novas escalas correspondentes ao tipo *Environment* do *TerraME* de forma aninhada, de modo que uma escala possa conter outras escalas.

¹⁰ Um *release* de software se refere a uma distribuição inicial ou a uma versão nova e atualizada de um produto de software [Wikipedia]

- **RF04 – Criação de modelos de comportamento:** o *TerraME GIMS* permitirá criar novos modelos que representarão o comportamento do modelo, correspondentes aos tipos *Agent* e *Automaton*, através de um menu.
- **RF05 – Criação de modelos de espaço:** o *TerraME GIMS* permitirá criar novos modelos que representarão o espaço a ser modelado, correspondentes ao tipo *CellularSpace*, através de um menu.
- **RF06 – Criação de modelos de tempo:** o *TerraME GIMS* permitirá criar novos modelos para representar o tempo para simulação do modelo, correspondente ao tipo *Timer*, através de um menu.
- **RF07 – Visualização hierárquica dos objetos do modelo:** o *TerraME GIMS* apresentará o modelo ao usuário de forma hierárquica, através de uma estrutura de árvore, semelhante ao *Windows Explorer* do sistema operacional *Windows*.
- **RF08 – Navegação e seleção de objetos através de estrutura hierárquica de visualização:** o *TerraME GIMS* permitirá ao usuário navegar pelo modelo e selecionar objetos através da estrutura de árvore de visualização, correspondente ao RF07.
- **RF09 – Construção de modelos de comportamento através de diagramas de estados:** o *TerraME GIMS* permitirá ao usuário construir modelos de comportamento através de autômatos (*Agent* ou *Automaton*) por meio de diagramas, utilizando o mouse, clicando sobre objetos gráficos correspondentes aos estados do autômato (*State*) e regras de transição (*Jump*), inserindo, selecionando, editando e excluindo objetos do modelo, conforme ilustrado na Figura 6.

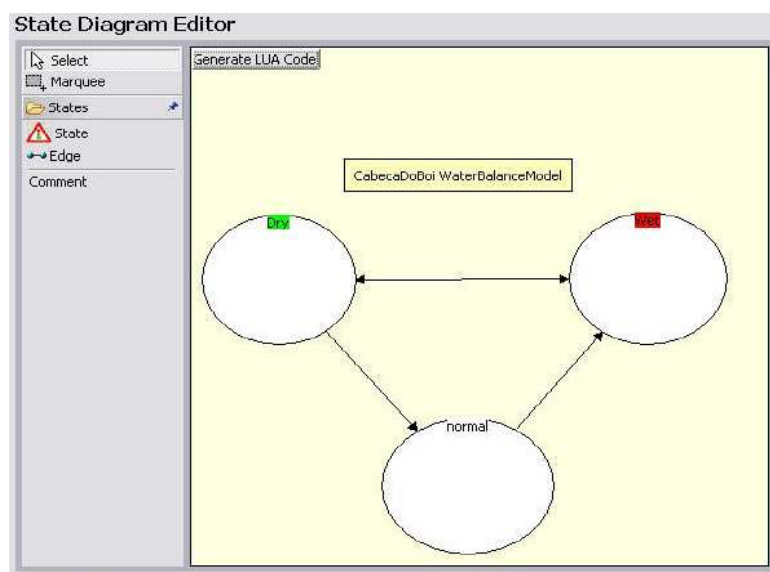


Figura 6 – Modelagem gráfica de autômatos

- **RF10 – Criação de trajetórias espaciais:** o *TerraME GIMS* permitirá ao usuário inserir trajetórias espaciais, correspondentes ao tipo *Trajectory* do *TerraME*, através de um menu.
- **RF11 – Associação de um espaço celular a uma base de dados:** o *TerraME GIMS* permitirá ao usuário, através de componentes de interface gráfica, associar um espaço celular a uma base de dados, conforme os seguintes parâmetros definidos pelo *TerraME*: *dbtype, host, database, user, password, layer, theme, select, where*.
- **RF12 – Associação de vizinhança a um espaço celular a partir de uma base de dados:** o *TerraME GIMS* permitirá ao usuário carregar a partir de uma base de dados as relações de vizinhança correspondentes ao tipo *Neighborhood* do *TerraME* para um espaço celular (*CellularSpace*).
- **RF13 – Inserir eventos em um modelo de tempo:** o *TerraME GIMS* permitirá ao usuário inserir eventos, correspondentes ao tipo *Event* do *TerraME*, em um modelo de tempo.
- **RF14 – Inserir mensagem correspondente a um evento em um modelo de tempo:** o *TerraME GIMS* permitirá ao usuário inserir uma mensagem, correspondentes ao tipo *Message* do *TerraME*, associada a um evento (*Event*) em um modelo temporal.
- **RF15 – Fornecimento de propriedades dos objetos por meio de componentes de interface gráfica:** o *TerraME GIMS* permitirá ao usuário fornecer as propriedades específicas dos objetos dos modelos, conforme definido pela linguagem *TerraME*, através de componentes de interface gráfica com o usuário. São exemplos destas propriedades: *latency* e *past* do objeto *Cell*, *time* e *period* do objeto *Event* e identificadores (*id's*) de objetos.
- **RF16 – Criação de funções:** o *TerraME GIMS* permitirá ao usuário especificar funções de forma textual, através da linguagem *TerraME*, de modo que estas façam parte do modelo e do código *TerraME* final a ser gerado pelo *TerraME GIMS*.
- **RF17 – Geração de código *TerraME*:** o *TerraME GIMS* irá, a partir dos modelos criados por meio da interação do usuário com a interface gráfica do aplicativo, gerar o código *TerraME* correspondente.
- **RF18 – Criação e edição de código *TerraME*:** o *TerraME GIMS* permitirá ao usuário construir modelos diretamente a partir de código *TerraME*.

3.2.3 Requisitos não-funcionais

Os requisitos não-funcionais, relacionados à implementação, interface, usabilidade e empacotamento, são apresentados a seguir.

Requisitos de implementação:

- **RN01 – Arquitetura baseada na plataforma *Eclipse*:** o *TerraME GIMS* deverá basear sua arquitetura e ser implementada a partir da plataforma *Eclipse*. Desta forma, será utilizada a linguagem *Java* e o próprio *Eclipse* como ferramenta de desenvolvimento.

Requisitos de interface:

- **RN02 – Interface gráfica para modelagem:** o *TerraME GIMS* deverá apresentar interface gráfica de aparência semelhante à interface mostrada na Figura 7.

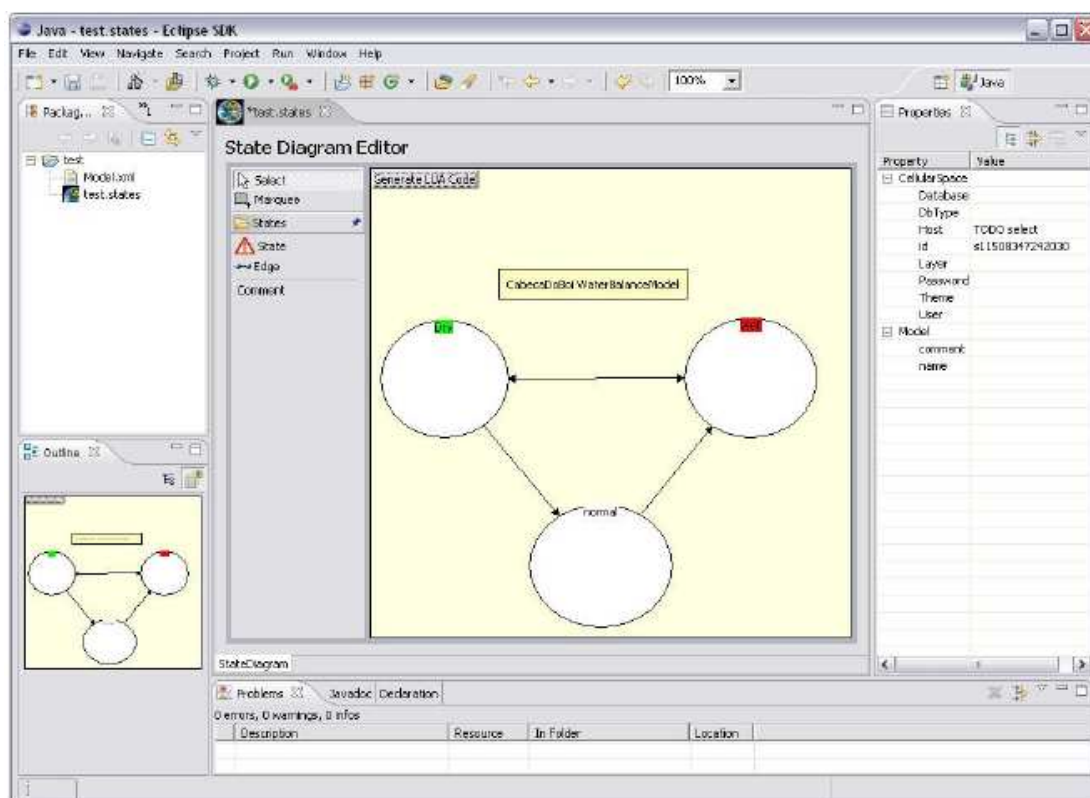


Figura 7 – Requisitos de interface do *TerraME GIMS*

- **RN03. Adequação às *guidelines* da interface gráfica do *Eclipse*:** a interface gráfica do *TerraME GIMS* deverá estar em conformidade com as *guidelines* de interface gráfica com o usuário do *Eclipse* (Edgar et al., 2004).

Requisitos de usabilidade:

- **RN04 – Manual de instalação:** o *TerraME GIMS* disponibilizará a seus usuários um manual para instalação e configuração do aplicativo.

- **RN05 – Manual de uso do aplicativo:** o *TerraME GIMS* disponibilizará a seus usuários um sistema de ajuda na forma de um tutorial para orientar a utilização do aplicativo.

Requisitos de empacotamento:

- **RN06 – Distribuição do *TerraME GIMS*:** o aplicativo *TerraME GIMS*, juntamente com sua documentação e código-fonte, deverá ser disponibilizado em URL a ser definida. Por se tratar de uma extensão da plataforma *Eclipse*, deverá disponibilizar todos os recursos necessários para execução do aplicativo (plataforma *Eclipse* e *plug-ins*) ou indicar as URL's para os mesmos.

3.3 Arquitetura do *TerraME GIMS*

Duas alternativas de desenvolvimento foram analisadas para a implementação do *TerraME GIMS*. A primeira é o desenvolvimento completo do aplicativo, sua interface gráfica e integração com o *TerraME*. Esta foi desconsiderada devido à elevada complexidade do sistema, alto custo e necessidade de maior prazo e de uma equipe de desenvolvimento.

A alternativa viável a ser utilizada é a implementação a partir de *frameworks* ou bibliotecas já existentes. A plataforma *Eclipse*, por ser amplamente extensível, permitir a integração com diversas ferramentas e oferecer um vasto conjunto de *frameworks* de *software* de domínio público, foi escolhida para servir de base para o desenvolvimento.

Para acelerar o processo de desenvolvimento do *TerraME GIMS*, este será desenvolvido na forma de *plug-ins* para a plataforma *Eclipse*. Os *frameworks* EMF e GEF dessa plataforma serão estendidos para permitir a geração de código *TerraME* a partir de diagramas.

Desta forma, o *TerraME GIMS*, em conformidade com as plataformas sobre as quais está sendo desenvolvido (*TerraME* e *Eclipse*), terá uma arquitetura em camadas. Na arquitetura em camadas, as camadas inferiores oferecem serviços a serem utilizados pelas camadas superiores para implementação de seus serviços. O *TerraME GIMS* irá compor uma nova camada, entre o *TerraME* e o usuário final, sendo a plataforma *Eclipse* uma camada intermediária entre o *TerraME* e o *TerraME GIMS*. Desta forma, não há restrição para a criação de modelos diretamente sobre a linguagem *TerraME* quando utilizando o *TerraME GIMS*.

A arquitetura é apresentada de forma detalhada na Figura 8, incluindo as camadas da arquitetura *TerraME* e as camadas da plataforma *Eclipse*. Acima da camada correspondente à plataforma *Eclipse* estão os *frameworks* EMF e GMF, utilizados para implementar funcionalidades relacionadas à criação e edição de modelos e de edição de componentes gráficos, respectivamente.

A última camada é a camada de aplicação, onde se encontram os modelos desenvolvidos pelos usuários finais do *TerraME GIMS*.

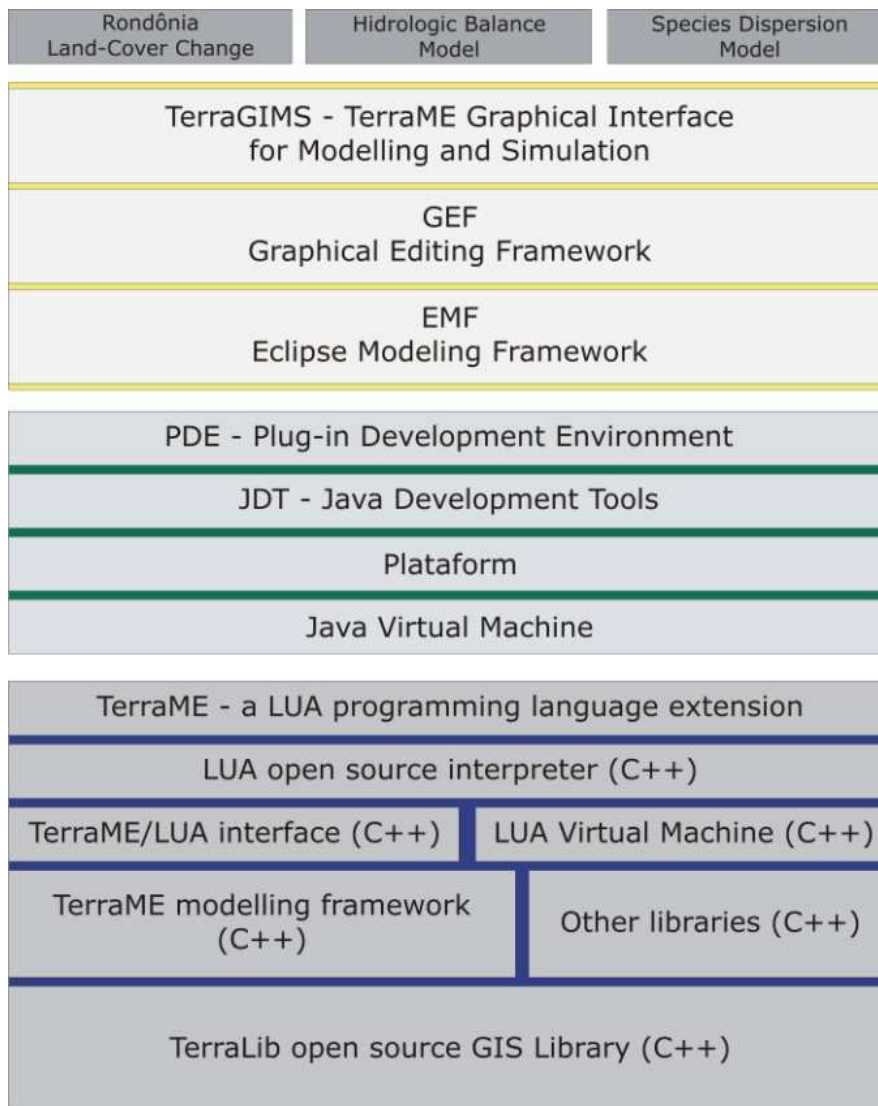


Figura 8 – Arquitetura em camadas do *TerraME GIMS*

4 AVALIAÇÃO DO *TerraME GIMS*

O *TerraME GIMS* será avaliado com o objetivo de analisar se os resultados esperados a partir da implementação do mesmo foram obtidos. Desta forma, inicialmente será realizado um levantamento com o objetivo de identificar o atual perfil da comunidade de usuários do *TerraME*. Em seguida será realizada a avaliação do *TerraME GIMS* junto aos usuários do *TerraME*. Dentre os critérios a serem analisados estão: a facilidade de uso do *TerraME GIMS*, a produtividade no desenvolvimento de modelos para sistemas terrestres utilizando o *TerraME*, a curva de aprendizado para novos usuários do ambiente *TerraME* utilizando o *TerraME GIMS*, a evolução do perfil da comunidade de usuários do *TerraME* após a disponibilização do *TerraME GIMS*.

4.1 Identificação e análise do atual perfil da comunidade de usuários do *TerraME*

Com o objetivo de fazer a avaliação do *TerraME GIMS* e acompanhar a evolução da comunidade de usuários do *TerraME* antes e após sua disponibilização será realizado um levantamento de dados que permita identificar o atual perfil da comunidade de usuários do *TerraME*. A partir deste perfil espera-se ter uma estimativa do tamanho atual da comunidade de usuários, a diversidade da comunidade quanto à formação e área de atuação dos usuários, extensão da comunidade quanto às instituições (e localização destas) que utilizam o *TerraME*, abrangência quanto às áreas de pesquisa das aplicações em que o ambiente está sendo utilizado.

Também serão levantados e analisados aspectos quanto ao perfil dos usuários do ambiente *TerraME*, com o objetivo de identificar o nível de domínio de tecnologias, as principais dificuldades encontradas ao utilizar o ambiente, domínio de algoritmos e técnicas de programação, o ambiente de desenvolvimento utilizado para construção de modelos, e demais informações que possam contribuir para o desenvolvimento e avaliação do *TerraME GIMS*.

Para coletar estas informações será elaborado um questionário a ser disponibilizado e preenchido através de um *web site*, em URL a ser definida.

4.2 Avaliação do *TerraME GIMS*

Uma vez implementado o *TerraME GIMS* o mesmo será avaliado junto aos atuais usuários do *TerraME*. Para a avaliação serão elaborados um conjunto de experimentos com os usuários na forma de tarefas a serem executadas pelos mesmos.

5 EXEMPLOS DE APLICAÇÕES

Com o objetivo de validar o *TerraME GIMS* e apresentar algumas das possíveis aplicações do ambiente *TerraME* como ambiente de modelagem serão desenvolvidos e implementados alguns exemplos de modelos. Porém, é importante ressaltar que o objetivo de desenvolver tais modelos não é apresentar estudos de casos que descrevam com precisão os fenômenos em questão (que exigiriam um estudo cuidadoso de todos os processos e parâmetros envolvidos, bem como etapas de calibração e validação, indo além do escopo deste trabalho), e sim modelos representativos que permitam demonstrar algumas das diversas aplicações possíveis para o ambiente *TerraME*.

O primeiro estudo de caso é um modelo simplificado de balanço hídrico, que tem como propósito simular o fenômeno de drenagem de águas pluviais (Carneiro, 2003; Carneiro and Câmara, 2006). Este modelo permitirá validar e exemplificar a utilização de conexão com bases de dados (*CellularSpace*), a utilização de modelos temporais (*Timer*) e a utilização de autômatos celulares (*Automaton*).

A implementação de outros modelos também está sendo avaliada, como por exemplo para simular os processos de propagação de fogo (Almeida et al. 2008; Almeida et al. 2008b), de crescimento urbano.

6 RESULTADOS

Neste capítulo são apresentados os resultados parciais obtidos, os resultados esperados ao final do projeto e o cronograma de execução das atividades, bem como algumas considerações parciais a respeito do desenvolvimento do trabalho.

6.1 Resultados Parciais

O trabalho possui como resultados parciais a publicação de artigos e a implementação de uma primeira versão na forma de protótipo do *TerraME GIMS*.

Os artigos publicados foram os seguintes:

- LIMA, T. F. M.; CARNEIRO, T. G. S.; FARIA, S. D. Desenvolvimento de uma Plataforma Gráfica para a Descrição de Modelos de Sistemas Ambientais. In: Simpósio Brasileiro de GeoInformática, 2008, Rio de Janeiro. Proceedings of Simpósio Brasileiro de GeoInformática, 2008. v. 1. p. 10-20.

Este artigo descreve a implementação do primeiro *release* do *TerraME GIMS* e os resultados parciais obtidos.

- LIMA, T. F. ; FARIA, S. D. ; SOARES FILHO, B. S. ; CARNEIRO, T. G. S. . Modelagem de sistemas baseada em agentes: alguns conceitos e ferramentas. In: XIV Simpósio Brasileiro de Sensoriamento Remoto, 2009, Natal. Anais do Simpósio Brasileiro de Sensoriamento Remoto, Natal, 2009. Anais do XIV Simpósio Brasileiro de Sensoriamento Remoto. São José dos Campos: INPE, 2009. p. 5279-5286.

Este artigo apresenta uma breve introdução à modelagem de sistemas baseada em agentes.

- CARNEIRO, T. G. S; LIMA, T. F. M.; FARIA, S. D. TerraLAB – Using Free Software for Earth System Research and Free Software Development. In: Workshop de Software Livre, Porto Alegre (RS). 2009.

Este artigo, aprovado e a ser publicado e apresentado no X WSL – X Workshop de Software Livre em junho de 2009, descreve a experiência de utilização de *software* livre no desenvolvimento de *software* livre e pesquisa no campo da Ciência dos Sistemas Terrestres.

A primeira versão (protótipo) do *TerraME GIMS* permite aos usuários criar modelos de forma gráfica através de uma estrutura de árvore e interagir com o modelo por meio de componentes gráficos, inserindo e alterando as propriedades dos elementos do modelo, sendo o código correspondente gerado de forma automática (Lima et al., 2008). Os seguintes requisitos

funcionais, descritos na Seção 3.2, encontram-se implementados nesta primeira versão: RF01, RF02, RF03, RF04, RF05, RF06, RF07, RF08, RF10, RF11, RF12, RF13, RF14, RF15, RF16, RF17, R18.

A Figura 9 apresenta o ambiente de desenvolvimento dos usuários do *TerraME GIMS*. Através do *Model Package Explorer* os usuários podem acessar os diversos arquivos que fazem parte do projeto. O *Model Structure Editor* permite navegar pelo modelo estruturado hierarquicamente na forma de uma árvore. Uma vez selecionado um elemento do modelo, este poderá ser editado a partir do *Model Graphical Editor* e do *Model Property Editor*. No *Model Graphical Editor*, em fase de implementação, serão apresentadas ao usuário representações gráficas correspondentes aos diversos elementos do modelo.

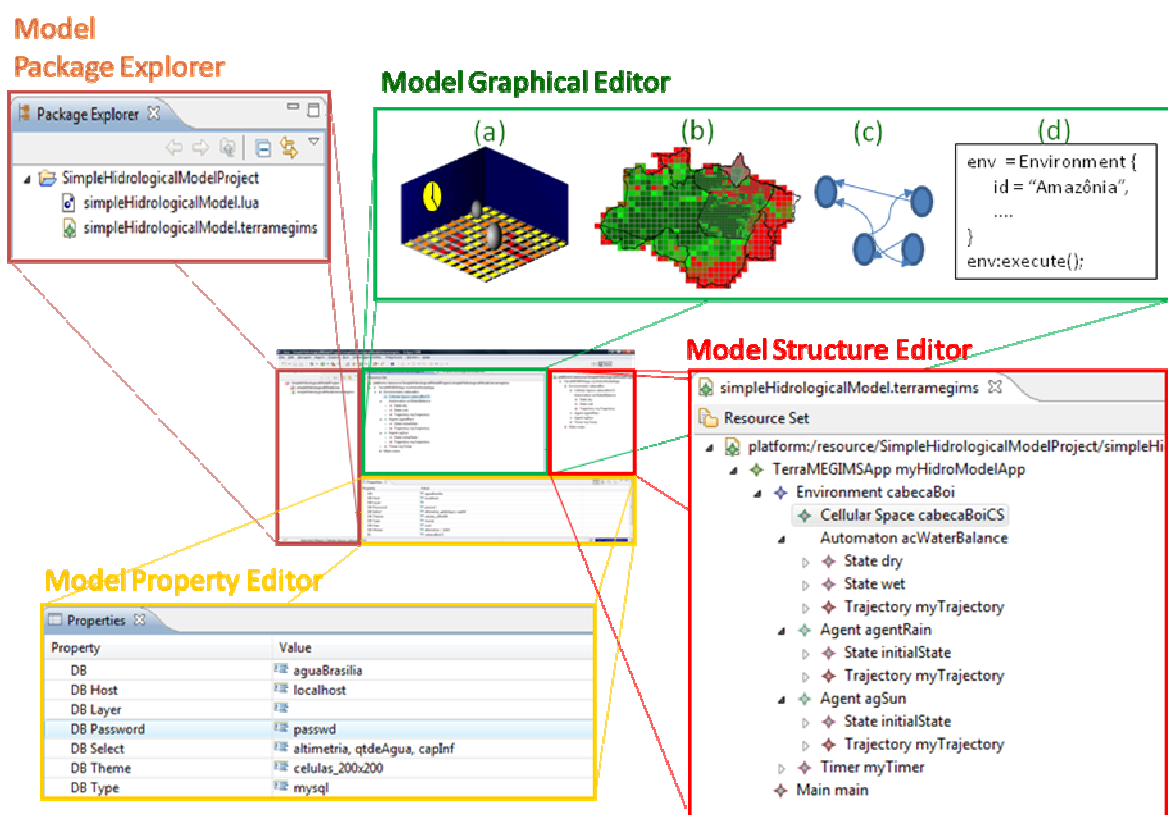


Figura 9 – Arquitetura em camadas do *TerraME GIMS*

Na Figura 10 podemos visualizar trecho de um modelo simples de balanço hídrico, representado em uma estrutura hierárquica na forma de uma árvore (*Model Structure Editor*) e a edição das propriedades dos elementos do modelo por meio de campos de texto e menus (*Model Property Editor*), bem como o respectivo código *TerraME* gerado automaticamente (*Model Graphical Editor*).

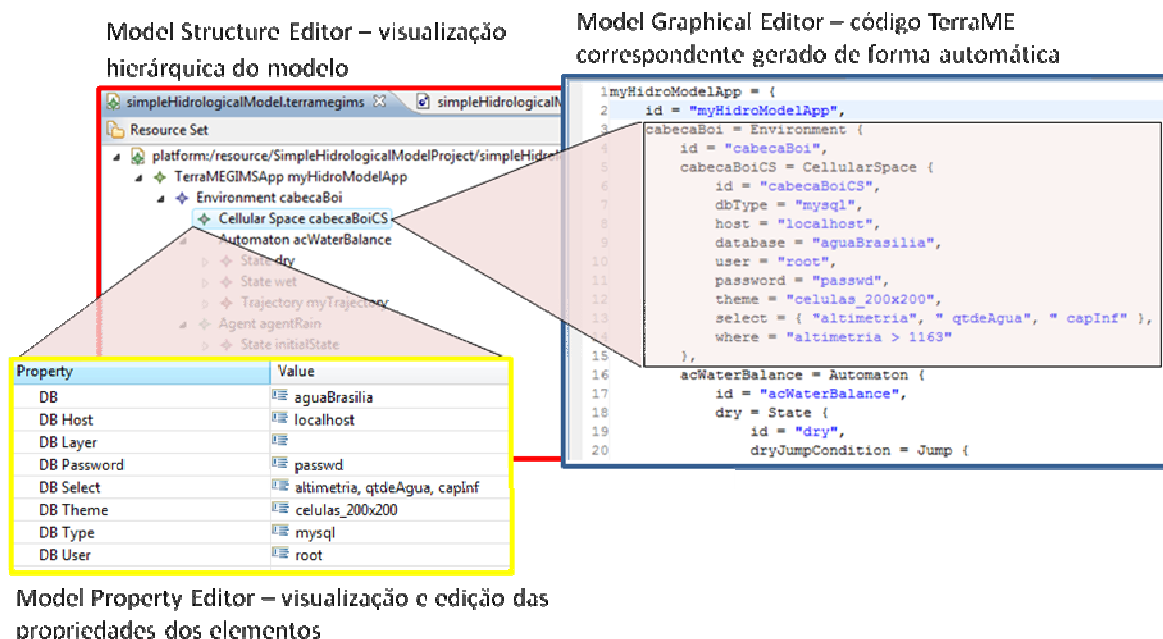


Figura 10 – Visualização hierárquica de um modelo *TerraME GIMS*

6.2 Resultados Esperados

Ao final do desenvolvimento do trabalho os seguintes resultados são esperados:

- implementação do *TerraME GIMS*, uma interface gráfica que irá permitir aos usuários construir modelos de forma gráfica para o ambiente *TerraME* ;
- avaliação do *TerraME GIMS*;
- identificação do perfil da atual comunidade de usuários do *TerraME*;
- base de dados que permitirá acompanhar a evolução do perfil da comunidade de usuários do *TerraME* e futuramente do *TerraME GIMS*;
- identificação do perfil dos usuários atuais do *TerraME*;
- base de dados que permitirá acompanhar a evolução do perfil dos usuários do *TerraME* e futuramente do *TerraME GIMS*;
- desenvolvimento de tutoriais e exemplos de demonstração de aplicações.

6.3 Plano de execução

As seguintes atividades serão realizadas para se atingir os objetivos do projeto e os resultados esperados.

Atividade	Ano 2009 – Semestre 2												2010
	Jul	Ago	Set	Out	Nov	Dez	Jan/10						
1. Perfil de usuários e da comunidade													
1.1. Elaboração do questionário de pesquisa													
1.2. Disponibilização do questionário													
1.3. Análise dos resultados													
2. Implementação do TerraME GIMS													
2.1. Implementação do RC 2 (RF09)													
2.2. Implementação do TerraME GIMS 1.0													
2.3. Disponibilização do TerraME GIMS 1.0													
3. Estudos de Caso													
3.1. Seleção dos processos a serem estudados e revisão bibliográfica													
3.2. Implementação dos estudos de caso utilizando o TerraME GIMS RC 2													
3.3. Implementação dos estudos de caso utilizando o TerraME GIMS 1.0													
4. Avaliação do TerraME GIMS													
4.1. Revisão bibliográfica e definição da metodologia de avaliação													
4.2. Avaliação do TerraME GIMS RC 2													
4.3. Análise dos resultados da avaliação do TerraME GIMS RC 2													
4.4. Avaliação do TerraME GIMS 1.0													
4.5. Análise dos resultados da avaliação do TerraME GIMS 1.0													
5. Dissertação													
5.1. Entrega da dissertação para revisão													
5.2. Revisão da dissertação													
5.3. Entrega da dissertação													
5.4. Defesa													
5.5. Revisão da dissertação final													

6.4 Considerações Parciais

Os resultados parciais obtidos demonstram a importância da implementação do *TerraME GIMS* tornado mais produtivo o processo de modelagem e diminuindo a curva de aprendizado para novos usuários do ambiente *TerraME*, contribuindo por fim para o aumento de comunidade de usuários e da diversidade de aplicações a utilizar o ambiente. Além disto, a construção de uma base de dados com informações que permitem identificar o perfil dos usuários e da comunidade de usuários do *TerraME* permitirá acompanhar a evolução destes, orientando a realização de novos esforços no desenvolvimento do ambiente.

7 BIBLIOGRAFIA

- ALMEIDA, R. M., MACAU, E. E. N., FRANCA, H., RAMOS, F. M., CARNEIRO, T. G. S. Simulando Padrões de Incêndio no Parque Nacional das Emas, Goiás, Brasil In: Simpósio Brasileiro de GeoInformática, 2008, Rio de Janeiro, RJ. Simpósio Brasileiro de GeoInformática. 1: 1, 2008. v.1. p.10 – 20. 2008.
- ALMEIDA, R. M.; MACAU, E. E. N.; FRANÇA, H.; RAMOS, F. M. Modelo de propagação do fogo em incêndios florestais e a teoria de percolação. In: XXXI Congresso Nacional de Matemática Aplicada e Computacional, 2008, Belém - Pará. Anais... São Carlos - SP: Sociedade Brasileira de Matemática Aplicada e Computacional - SBMAC, 2008. v. 1. On-line. Disponível em: <http://www.sbmec.org.br/publi_cnmac/trabalhos2008.php>. Acessado em: junho de 2009. 2008b.
- ANDERSEN, D.; ROBERTS, N.; DEAL, R.; GARET, MICHAEL; SHAFFER, W. D. Introduction to Computer Simulation: The System Dynamics Approach, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1983
- ANDRADE, P.R.; MONTEIRO, A. M. V.; CÂMARA, G.; SANDRI, S. Games on Cellular Spaces: How Mobility Affects Equilibrium. Journal of Artificial Societies and Social Simulation (JASS). 12(1) p. 5. 2009.
- BOEHM, B. W. A Spiral Model of Software Development and Enhancement. Computer, May 1988, pp. 61-72. 1988.
- CÂMARA, G.; SOUZA, R. C. M.; PEDROSA, B. M.; VINHAS, L.; MONTEIRO, A. M. V.; PAIVA, J. A.; CARVALHO, M. T. M.; GATTASS, M.. TerraLib: Technology in support of GIS Innovation. In: GeoInfo2000, 2000, São Paulo - SP. II Workshop Brasileiro de Geoinformática, 2000.
- CÂMARA, G.; CARNEIRO, AGUIAR, A. P. D.; CARNEIRO, T. G. S.; NETO, P. R.; FEITOSA, F. Course on Environmental Modeling – LUCC. 2008. Disponível em <http://www.dpi.inpe.br/cursos/environmental_modelling/index.html>
- CÂMARA, G.; CARNEIRO, T. G. S.; BEYILACQUA, L. I Curso de Verão Geoma 2007 – Modelagem Dinâmica com TerraME. 2007. Disponível em: <http://www.dpi.inpe.br/geoma/curso_verao/index.php>. Acesso em: agosto de 2008.
- CARNEIRO, T. G. S. Nested-CA: a foundation for multiscale modeling of land use and land change. 2003. 29 p. (INPE -5522-TDI/519) Tese (Doutorado em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2003.

CARNEIRO, T. G. S.; CÂMARA, G. A gentle Introduction to TerraME. INPE Report, 2006. Disponível em: <<http://www.dpi.inpe.br/TerraME/introd.pdf>>. Acessado em: junho de 2009.

CARNEIRO, T. G. S., MARETTO, E. V., CÂMARA, G. Irregular Cellular Spaces: Supporting Realistic Spatial Dynamic Modeling over Geographical Databases In: Simpósio Brasileiro de GeoInformática, 2008, Rio de Janeiro, RJ. Simpósio Brasileiro de GeoInformática. 1: 1, 2008. v.1. p.1 – 1

CARNEIRO, T. G. S; LIMA, T. F. M.; FARIA, S. D. TerraLAB – Using Free Software for Earth System Research and Free Software Development. In: Workshop de Software Livre, Porto Alegre (RS). 2009.

ECLIPSE. Documentação do Eclipse. Disponível em: <<http://help.eclipse.org/help32/index.jsp>>. Acessado em: junho de 2009.

ECLIPSE EMF, Eclipse Modeling Framework Project. Disponível em: <<http://www.eclipse.org/modeling/emf/>>. Acessado em junho de 2009.

ECLIPSE EMF 2005, The Eclipse Modeling Framework (EMF) Overview. 2005. Eclipse Documentation. Disponível em <<http://help.eclipse.org/ganymede/index.jsp?topic=/org.eclipse.emf.doc/references/overview/EMF.html>> . Acessado em junho de 2009.

EDGAR, N.; HAALAND, K.; LI, J.; PETER, K. Eclipse User Interface Guidelines. 2004. Disponível em: <<http://www.eclipse.org/articles/Article-UI-Guidelines/Index.html>> Acessado em: junho de 2009.

GOSLING, J.; JOY, B.; STEELE, G. L. The Java TM Language Specification. Addison-Wesley, 1996.

IERUSALIMSKY, R.; FIGUEIREDO, L. H.; CELES, W. Lua - an extensible extension language, Software: Practice & Experience 26 #6 (1996) 635-652.

LIMA, T. F. M; CARNEIRO, T. G. S.; FARIA, S. D. Desenvolvimento de uma Plataforma Gráfica para a Descrição de Modelos de Sistemas Ambientais. In: Simpósio Brasileiro de GeoInformática, 2008, Rio de Janeiro. Proceedings of Simpósio Brasileiro de GeoInformática, 2008. v. 1. p. 10-20.

LIMA, T. F. ; FARIA, S. D. ; SOARES FILHO, B. S. ; CARNEIRO, T. G. S. . Modelagem de sistemas baseada em agentes: alguns conceitos e ferramentas. In: XIV Simpósio Brasileiro de Sensoriamento Remoto, 2009, Natal. Anais do Simpósio Brasileiro de Sensoriamento Remoto, Natal, 2009. Anais do XIV Simpósio Brasileiro de Sensoriamento Remoto. São José dos Campos: INPE, 2009. p. 5279-5286.

MARIA, A. Introduction to Modelling and Simulation. In: Proceedings of the 1997 Winter Simulation Conference. 1997.

MINAR, N.; BURKHART, R.; LANGTON, C.; ASKENAZI, M. 1996. The Swarm Simulation System: A toolkit for building multi-agent simulations. Working Paper 96-06-042, Santa Fe Institute, Santa Fe.

MORE, B; DEAN, D.; GERBER, A.; WAGENKNECHT, G.; VANDERHEYDEN, P. Eclipse Development using the Graphical Editing Framework and the Eclipse Modeling Framework. IBM Redbooks. 2003.

MOREIRA, E.; AGUIAR, A. P.; COSTA, S. S.; Câmara, G. Spatial relations across scales in land change models. In: X Brazilian Symposium on Geoinformatics, GeoInfo 2008, Rio de Janeiro, 2008.

NORTH, M. J.; HOWE, T.R.; COLLIER, N.T.; VOS, J.R. Vos, "Repast Symphony Runtime System," in C.M. Macal, M.J. North, and D. Sallach (eds.), Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms, ANL/DIS-06-1, co-sponsored by Argonne National Laboratory and The University of Chicago, Oct. 13-1. Disponível em: < http://agent2007.anl.gov/2005procpdf/Agent_2005_North_Runtime.pdf>.

PIMENTA, P.; COELHO, A.; COSTA, S. S.; MOREIRA, E ; AGUIAR, A. P.; CÂMARA, G.; ARAÚJO, R.; RIBEIRO, A . Land change modeling and institutional factors: heterogeneous rules of territory use in the Brazilian Amazonia. X Brazilian Symposium on Geoinformatics, GeoInfo 2008, Rio de Janeiro, 2008.

POWELL, A. Model with the Eclipse Modeling Framework. 2004. Disponível em: <<http://www-128.ibm.com/developerworks/library/os-ecemf1/>>. Acessado em: junho de 2009.

RIVIERES, J.; WIEGAND, J. Eclipse: A platform for integrating development tools. IBM Systems Journal, 43(2), 2004.

RIVIERES, J.; BEATON, W. Eclipse Technical Platform Overview. 2006. Disponível em <<http://www.eclipse.org/articles/Whitepaper-Platform-3.1/eclipse-platform-whitepaper.html>> Acessado em: junho de 2009.

STROUSTRUP, B. The Design and Evolution of C++. Addison-Wesley. 1994.

TecComm. Apresentação: ‘Visão Geral da Plataforma Eclipse’. Disponível em: <http://server.teccomm.les.inf.puc-rio.br/SeminarioAcompanhamento/03_VisaoGeralPlataformaEclipse.pdf>. Acessado em: junho de 2009.