

UERJ

Dissertação de Mestrado em Engenharia de Computação

**SIMULAÇÃO DE ACIDENTES RADIOLÓGICOS
ATRAVÉS DE
SOFTWARE BASEADO EM AGENTES**

Autor: Tadeu Augusto de Almeida Silva

Orientador: Oscar Luiz Monteiro de Farias

Programa de Pós-Graduação Em Engenharia de Computação
Área de Concentração: Geomática

Setembro – 2007



Faculdade de Engenharia

SIMULAÇÃO DE ACIDENTES RADIOLÓGICOS ATRAVÉS DE SOFTWARE BASEADO EM AGENTES

Tadeu Augusto de Almeida Silva

Dissertação submetida ao corpo docente da Faculdade de Engenharia da Universidade do Estado do Rio de Janeiro – UERJ, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Computação –Área de Concentração Geomática.

Orientador: Oscar Luiz Monteiro de Farias

Programa de Pós-Graduação Em Engenharia de Computação
Área de Concentração: Geomática

Rio de Janeiro

Setembro – 2007

SILVA, TADEU AUGUSTO DE ALMEIDA SILVA

Simulação de Acidentes Radiológicos através de Software baseado em Agentes, [Rio de Janeiro] 2007.

vii, 66 p. 29,7 cm (FEN/UERJ, M.Sc., Engenharia de Computação – Área de Concentração Geomática, 2007).

Dissertação - Universidade do Estado do Rio Janeiro – UERJ

1. Agentes. 2.Acidente Radiológico. 3. Sistemas de Informação Geográfica. 4. Simulação.

I. FEN/UERJ. II. Título (série).

FOLHA DE JULGAMENTO

Título: Simulação de Acidentes Radiológicos através de Software baseado em Agentes.

Candidato: Tadeu Augusto de Almeida Silva

Programa de Pós-Graduação em Engenharia de Computação
Área de Concentração: Geomática

Data de Defesa: 14 de setembro de 2007

Aprovada por:

Orientador: Oscar Luiz Monteiro de Farias, D.Sc.,UERJ

Neide dos Santos, D.Sc., UERJ

Evaldo Simões da Fonseca, D.Sc.,Laboratório de Nêutrons, LNMRI, IRD

DEDICATÓRIA

Aos meus pais por tudo que me proporcionaram em felicidade, educação e oportunidades.

AGRADECIMENTOS

Gostaria de fazer aqui um agradecimento a todos aqueles que de alguma forma tiveram uma contribuição nesta dissertação.

Ao meu orientador Professor Oscar, pela clara orientação e dedicação.

À minha querida família: Luciana, Isabel e Leonardo, cujo amor me alimenta.

Ao meu irmão Francisco César que me auxiliou em questões sobre proteção radiológica.

Ao meu chefe no IRD Marcos César que confiou em mim.

Ao Instituto de Radioproteção e Dosimetria – IRD que me ofereceu uma rica fonte de material para pesquisa e estudo.

À Carlos Bernardo González Pecotche (Raumsol) por seus ensinamentos e a sua escola filosófica, que têm me estimulado sempre a buscar o saber.

Resumo da Dissertação apresentada a FEN / UERJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M. Sc.).

SIMULAÇÃO DE ACIDENTES RADIOLÓGICOS ATRAVÉS DE SOFTWARE BASEADO EM AGENTES

Tadeu Augusto de Almeida Silva

Setembro/2007

Orientador: Oscar Luiz Monteiro de Farias, DSc., UERJ

Programa de Pós-Graduação em Engenharia de Computação - Área de Concentração em Geomática – Mestrado.

Acidentes radiológicos podem provocar grave exposição à radiação em trabalhadores e no público, em geral. Estes acidentes, geralmente, ocorrem como consequência do mau uso da radiação ou de substâncias radioativas em atividades humanas nas áreas militar, industrial, médica, agrícola e pesquisa, sujeitas à exposição de radiação acima dos limites naturais. Os efeitos dos desastres radiológicos precisam ser imediatamente investigados e a determinação das doses estimada, a fim de oferecer tratamento médico adequado para as pessoas expostas, estudar o seu possível impacto na sociedade, bem como aprender lições para o futuro, a fim de prevenir outros acidentes. Nesta dissertação apresentamos o uso de *software* multiagentes inseridos numa representação geográfica, como uma opção viável e econômica para simular acidentes radiológicos e realizar o cálculo da dose. Construir um ambiente, onde se possa simular a dispersão, não natural, da radiação, em uma área delimitada, pode fornecer parâmetros para análise, estudo e gerenciamento destes sistemas dinâmicos.

Palavras-Chave: Agentes, Acidente Radiológico, Sistemas de Informação Geográfica, Simulação.

Abstract of Dissertation presented do FEN / UERJ as a partial fulfillment of requirements for the degree of Master of Science (M.Sc.).

SIMULATING RADIOLOGICAL ACCIDENTS THROUGH SOFTWARE AGENTS

Tadeu Augusto de Almeida Silva

September/2007

Advisor: Oscar Luiz Monteiro de Farias

Program of Computing Engineering – Geomatic.

Radiological accidents can result in significant radiation exposures of workers and of the public. These accidents usually occur as a consequence of human activities in military, industry, medicine, agriculture and research, involving the wrong use of radiation and radioactive substances that cause radiation exposure in addition to the natural exposure. The effects of radiological disasters must be promptly investigated and the assessment of doses estimated, in order to provide adequate medical treatment to the persons affected, study their impact in society, as well learning lessons for the future and prevent further accidents. In this dissertation we introduce the use of software multi-agents systems immersed in a geographical representation of the world, as a viable and economic option to simulate radiological accidents and assess doses. Build an environment, where the researcher can simulate a non-natural radioactive dispersion, in a delimited area, can furnish parameters for analyzing, studying and managing these dynamic systems.

Keywords: Agents, Radiological Accident, Geographical Information Systems, Simulation.

Sumário

.....	i
Faculdade de Engenharia.....	i
SILVA, TADEU AUGUSTO DE ALMEIDA SILVA.....	i
ORIENTADOR: OSCAR LUIZ MONTEIRO DE FARIAS, D.Sc.,UERJ.....	I
NEIDE DOS SANTOS, D.Sc., UERJ.....	I
EVALDO SIMÕES DA FONSECA, D.Sc.,LABORATÓRIO DE NÊUTRONS, LNMRI, IRD.....	I
RESUMO DA DISSERTAÇÃO APRESENTADA A FEN / UERJ COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS (M. Sc.).....	II
SIMULAÇÃO DE ACIDENTES RADIOLÓGICOS ATRAVÉS DE SOFTWARE BASEADO EM AGENTES.....	II
TADEU AUGUSTO DE ALMEIDA SILVA.....	II
SETEMBRO/2007.....	II
TADEU AUGUSTO DE ALMEIDA SILVA.....	III
SEPTEMBER/2007.....	III
ÍNDICE DAS FIGURAS.....	XII
ÍNDICE DAS TABELAS.....	XIII
LISTA DE NOMENCLATURAS.....	XIII
Todo esta estrutura é necessária em decorrência da magnitude e dos riscos envolvidos em possíveis acidentes radiológicos e nucleares. Em nosso país, vivemos uma situação de grande impacto, quando do acidente radiológico de Goiânia, em setembro de 1987. Neste evento, uma fonte de Césio-137, utilizada em equipamentos de radioterapia, foi dispersada acidentalmente junto à população, o que provocou , além da contaminação de áreas urbanas, a contaminação em doses elevadas de 249 pessoas, a contaminação interna e externa de outras 129 pessoas, alterações hematológicas de 20 pessoas e a morte de 4 pessoas.	1
As medidas defensivas e ações de emergência envolveram 575 profissionais durante 6 meses; a retirada de 200 habitantes de 41 casas, trabalho de descontaminação de 85 casas, demolição de 7 casas e a monitoração de 3,5 toneladas de rejeitos. Além dos efeitos psicológicos negativos, que traumatizaram a população por vários anos, foram impetrados vários processos judiciais custosos para a União. [1].....	2
1.1 ACIDENTES EM INSTALAÇÕES NUCLEARES E RADIATIVAS.....	7
1.2 Medição da Radioatividade.....	10
2 SISTEMAS BASEADOS EM AGENTES AUTÔNOMOS.....	14
3.1 UTILIZAÇÃO DE SIG NO MAPEAMENTO DE ÁREAS GEOGRÁFICAS.....	27
.....	30
3.2 Princípios de Modelagem em Simulação.....	31
3.3 USO DE FERRAMENTAS DE SIMULAÇÃO BASEADA EM AGENTES AUTÔNOMOS.....	35
4 ESTUDO DE CASO: ACIDENTE RADIOLÓGICO DE COCHABAMBA, BOLÍVIA.....	36
Assento do Ônibus.....	42
5 CONCLUSÕES	43
5.1 TRABALHOS FUTUROS.....	44

BIBLIOGRAFIA.....	46
ANEXO I – CÓDIGO FONTE EM JAVA COM AGENTES UTILIZADOS NA SIMULAÇÃO.....	48
//MODELO DA SIMULAÇÃO DE ACIDENTE RADIOLÓGICO NO ÔNIBUS //	48
IMPORT JAVA.AWT.COLOR;.....	48
IMPORT JAVA.AWT.DIMENSION;.....	48
IMPORT JAVA.AWT.EVENT.ACTIONEVENT;.....	48
IMPORT JAVA.AWT.EVENT.ACTIONLISTENER;.....	48
IMPORT JAVA.UTIL.ARRAYLIST;.....	48
IMPORT JAVA.UTIL.ITERATOR;.....	48
IMPORT UCHICAGO.SRC.SIM.ANALYSIS.OPENSEQUENCEGRAPH;.....	48
IMPORT UCHICAGO.SRC.SIM.ANALYSIS.SEQUENCE;.....	48
IMPORT UCHICAGO.SRC.SIM.ENGINE.AUTOSTEPABLE;.....	48
IMPORT UCHICAGO.SRC.SIM.ENGINE.SIMPLEMODEL;.....	48
IMPORT UCHICAGO.SRC.SIM.GUI.DEFAULTGRAPHLAYOUT;.....	48
IMPORT UCHICAGO.SRC.SIM.GUI.DISPLAYSURFACE;.....	48
IMPORT UCHICAGO.SRC.SIM.GUI.NETWORK2DDISPLAY;.....	48
IMPORT UCHICAGO.SRC.SIM.UTIL.RANDOM;.....	48
IMPORT UCHICAGO.SRC.SIM.UTIL.REPASTEXCEPTION;.....	48
IMPORT UCHICAGO.SRC.SIM.UTIL.SIMUTILITIES;.....	48
PUBLIC CLASS MODELOSIMULACAO EXTENDS SIMPLEMODEL {.....	48
PROTECTED LOCAL LOCAL;.....	48
PRIVATE INT NUMPASSAGEIROS = 1;.....	48
PRIVATE DISPLAYSURFACE LOCALDISPLAYSURFACE;.....	48
PRIVATE OPENSEQUENCEGRAPH LOCALERRORGRAPH;.....	48
PRIVATE OPENSEQUENCEGRAPH INDIVIDUALGRAPH;.....	48
PROTECTED INT LOCALWIDTH = 1000, LOCALHEIGHT = 1000;.....	48
PUBLIC MODELOSIMULACAO() {.....	48
// DEFINIR GERAÇÃO RANDÔMICA DE AGENTES.....	48
RANDOM.CREATEUNIFORM();.....	48
IF (SUPER.AGENTLIST == NULL).....	48
SUPER.AGENTLIST = NEW ARRAYLIST();.....	48
}.....	48
.....	48
PUBLIC STRING[] GETINITPARAM() {.....	48
RETURN NEW STRING[] {	48

"NumPASSAGEIROS",	48
"POSICÃOX",	48
"POSICÃOY"	48
};.....	48
}.....	48
PUBLIC VOID BEGIN() {.....	48
TRY {.....	48
SUPER.BEGIN();.....	48
// ZERAR NÚMERO DE AGENTES.....	49
PASSAGEIRO.RESETÍNDICES();.....	49
IF (SUPER.AGENTLIST == NULL).....	49
SUPER.AGENTLIST = NEW ARRAYLIST();.....	49
SUPER.AGENTLIST.CLEAR();.....	49
LOCAL = NEW LOCAL(LOCALWIDTH, LOCALHEIGHT);.....	49
.....	49
// UTILIZAR O NÚMERO DEFINIDO DE AGENTES	49
LOCAL.HIREPASSAGEIRO(NumPASSAGEIROS);.....	49
LOCAL.HIREORDENADOR();.....	49
LOCAL.HIREORIENTADOR();.....	49
// LISTAR OS AGENTES	49
SUPER.AGENTLIST.ADDALL(LOCAL.GETPASSAGEIRO());.....	49
SUPER.AGENTLIST.ADDALL(LOCAL.GETORDENADOR());.....	49
SUPER.AGENTLIST.ADD(LOCAL.GETORIENTADOR());.....	49
THIS.BUILDDISPLAY();.....	49
THIS.BUILDGRAPHS();.....	49
} CATCH (REPASTEXCEPTION EX) {.....	49
SIMUTILITIES.SHOWERROR("ERRO AO LER O MODELO", EX);.....	49
SUPER.STOP();.....	49
}.....	49
}.....	49
/**	49
* EXIBIR O DISPLAY DO AMBIENTE	49
*/.....	49
PROTECTED VOID BUILDDISPLAY() {.....	49
LOCALDISPLAYSURFACE =	49

new DISPLAYSURFACE(.....	49
new DIMENSION(LOCALWIDTH, LOCALHEIGHT),	49
THIS,	49
"SIMULAÇÃO ACIDENTE RADIOLÓGICO");.....	49
// CRIAR GRÁFICO.....	49
DEFAULTGRAPHLAYOUT LAYOUT = new DEFAULTGRAPHLAYOUT(LOCALWIDTH,.....	49
LOCALHEIGHT);.....	49
LAYOUT.GETNodelist().ADDAll(LOCAL.GETPASSAGEIRO());.....	49
LAYOUT.GETNodelist().ADD(new OBJETIVO(LOCAL));.....	49
NETWORK2DDisplay LOCALNetDisplay = new NETWORK2DDisplay(LAYOUT);.....	49
LOCALDISPLAYSURFACE.ADDDISPLAYABLEPROBEABLE(LOCALNetDisplay,.....	49
"ESTIMATIVAS");.....	49
.....	49
LOCALDISPLAYSURFACE.SETBACKGROUND(COLOR.WHITE);.....	49
LOCALDISPLAYSURFACE.DISPLAY();.....	49
THIS.REGISTERDISPLAYSURFACE("SIMULAÇÃO", LOCALDISPLAYSURFACE);.....	49
}.....	49
.....	50
/**	50
* CRIAR GRÁFICO MOVIMENTO	50
*/	50
PROTECTED VOID BUILDGRAPHS() {.....	50
LOCALERRORGRAPH = new OPENSEQUENCEGRAPH("MONITORA MOVIMENTO", THIS);.....	50
LOCALERRORGRAPH.ADDSEQUENCE("", new SEQUENCE() {.....	50
PUBLIC DOUBLE GETSVVALUE() {.....	50
DOUBLE TOTALERR = 0;.....	50
FOR (Iterator ITER = LOCAL.GETPASSAGEIRO().ITERATOR(); ITER.....	50
.HASNEXT();) {.....	50
TOTALERR += ((PASSAGEIRO) ITER.NEXT()).GETERROR();.....	50
}.....	50
RETURN TOTALERR / LOCAL.GETPASSAGEIRO().SIZE();.....	50
.....	50
}.....	50
});.....	50
LOCALERRORGRAPH.SETYRANGE(-.1, 1.1);.....	50

LOCALERRORGRAPH.SETXRANGE(0, 5);.....	50
LOCALERRORGRAPH.DISPLAY();.....	50
/*** CRIAR GRÁFICO DE ESTIMATIVA DE MOVIMENTO ***/.....	50
INDIVIDUALGRAPH = NEW OPENSEQUENCEGRAPH("ESTIMATIVA DE MOVIMENTO", THIS);.....	50
.....	50
FOR (ITERATOR ITER = LOCAL.GETPASSAGEIRO().ITERATOR(); ITER.HASNEXT();) {.....	50
FINAL PASSAGEIRO PASS = (PASSAGEIRO) ITER.NEXT();	50
INDIVIDUALGRAPH.ADDSEQUENCE(.....	50
PASS.GETNODELABEL() ,	50
NEW SEQUENCE() {.....	50
PUBLIC DOUBLE GETSVVALUE() {.....	50
RETURN PASS.GETERROR();.....	50
}	50
},.....	50
PASS.GETCOLOR());.....	50
.....	50
}	50
.....	50
INDIVIDUALGRAPH.SETYRANGE(-.1, 1.1);.....	50
INDIVIDUALGRAPH.SETXRANGE(0, 5);.....	50
INDIVIDUALGRAPH.DISPLAY();.....	50
}	50
.....	50
/**	50
* ZERAR O MODELO PARA NOVA EXECUÇÃO.....	50
*/.....	50
PUBLIC VOID SETUP() {.....	51
SUPER.SETUP();.....	51
.....	51
IF (LOCALERRORGRAPH != NULL).....	51
LOCALERRORGRAPH.DISPOSE();.....	51
IF (INDIVIDUALGRAPH != NULL).....	51
INDIVIDUALGRAPH.DISPOSE();.....	51
IF (LOCALDISPLAYSURFACE != NULL).....	51
LOCALDISPLAYSURFACE.DISPOSE();.....	51

LOCALDISPLAYSURFACE = NULL;.....	51
.....	51
GETMODELMANIPULATOR().ADDBUTTON("DISPERSÃO DOS AGENTES", NEW ACTIONLISTENER() {	51
.....	51
PUBLIC VOID ACTIONPERFORMED(ACTIONEVENT E) {.....	51
ITERATOR ITER = ((ARRAYLIST) LOCAL.GETPASSAGEIRO().CLONE()).ITERATOR();.....	51
FOR (; ITER.HASNEXT();) {.....	51
PASSAGEIRO PASS = (PASSAGEIRO) ITER.NEXT();.....	51
.....	51
PASS.SETX(RANDOM.UNIFORM.NEXTINTFROMTO(0, LOCAL.GETWIDTH()));.....	51
PASS.SETY(RANDOM.UNIFORM.NEXTINTFROMTO(0, LOCAL.GETHEIGHT()));.....	51
}.....	51
.....	51
LOCALDISPLAYSURFACE.UPDATEDISPLAYDIRECT();.....	51
}.....	51
});.....	51
}.....	51
.....	51
PROTECTED VOID PRESTEP() {.....	51
TRY {.....	51
FOR (ITERATOR ITER = SUPER.AGENTLIST.ITERATOR(); ITER.HASNEXT();) {.....	51
((AUTOSTEPABLE) ITER.NEXT()).PRESTEP();.....	51
}.....	51
} CATCH (EXCEPTION EX) {.....	51
SIMUTILITIES.SHOWERROR("ERRO NA PREDEFINIÇÃO DA SIMULAÇÃO", EX);.....	51
SUPER.STOP();.....	51
}.....	51
}.....	51
.....	51
PROTECTED VOID STEP() {.....	51
TRY {.....	51
FOR (ITERATOR ITER = SUPER.AGENTLIST.ITERATOR(); ITER.HASNEXT();) {.....	51
OBJECT O = ITER.NEXT();.....	51
((AUTOSTEPABLE) O).STEP();.....	51
}.....	51

} CATCH (EXCEPTION EX) {.....	52
SIMUTILITIES.SHOWERROR("ERRO NA PREDEFINIÇÃO DA SIMULAÇÃO", EX);.....	52
SUPER.STOP();.....	52
}.....	52
}.....	52
PROTECTED VOID POSTSTEP() {.....	52
TRY {.....	52
FOR (ITERATOR ITER = SUPER.AGENTLIST.ITERATOR(); ITER.HASNEXT();) {.....	52
((AUTOSTEPABLE) ITER.NEXT()).POSTSTEP();.....	52
}.....	52
} CATCH (EXCEPTION EX) {.....	52
SIMUTILITIES.SHOWERROR("ERRO NA PREDEFINIÇÃO DA SIMULAÇÃO", EX);.....	52
SUPER.STOP();.....	52
}.....	52
LOCALDISPLAYSURFACE.UPDATEDISPLAY();.....	52
LOCALERRORGRAPH.STEP();.....	52
INDIVIDUALGRAPH.STEP();.....	52
}.....	52
.....	52
PUBLIC STRING GETNAME() {.....	52
RETURN "SIMULAÇÃO ACIDENTE";.....	52
}.....	52
/**.....	52
* @RETURN NUMERO DE PASSAGEIROS.....	52
*/.....	52
PUBLIC INT GETNUMPASSAGEIROS() {.....	52
RETURN NUMPASSAGEIROS;.....	52
}.....	52
PUBLIC VOID SETNUMPASSAGEIROS(INT NUMPASSAGEIROS) {.....	52
THIS.NUMPASSAGEIROS = NUMPASSAGEIROS;.....	52
}.....	52
PUBLIC INT GETLOCALHEIGHT() {.....	52
RETURN LOCALHEIGHT;.....	52
}.....	52
PUBLIC VOID SETLOCALHEIGHT(INT LOCALHEIGHT) {.....	52

THIS.LOCALHEIGHT = LOCALHEIGHT;.....	52
}.....	52
PUBLIC INT GETLOCALWIDTH() {.....	52
RETURN LOCALWIDTH;.....	52
}.....	52
PUBLIC VOID SETLOCALWIDTH(INT LOCALWIDTH) {.....	52
THIS.LOCALWIDTH = LOCALWIDTH;.....	52
}.....	53
.....	53
PUBLIC STATIC VOID MAIN(STRING[] ARGS) {.....	53
UCHICAGO.SRC.SIM.ENGINE.SIMINIT INIT = NEW UCHICAGO.SRC.SIM.ENGINE.SIMINIT();.....	53
MODELOSIMULACAO MODEL = NEW MODELOSIMULACAO();.....	53
IF (ARGS.LENGTH > 0).....	53
INIT.LOADMODEL(MODEL, ARGS[0], FALSE);.....	53
ELSE.....	53
INIT.LOADMODEL(MODEL, NULL, FALSE);.....	53
}.....	53
}.....	53

Índice das Figuras

FIGURA 1 – ACIDENTE RADIOLÓGICO.....	9
FIGURA 2 – EFEITOS DA RADIAÇÃO.....	10
FIGURA 3 – TEMPO, DISTÂNCIA E BLINDAGEM: PRINCÍPIOS DA RADIOPROTEÇÃO.....	13
AUTOR: TADEU AUGUSTO DE ALMEIDA SILVA.....	I
DEDICATÓRIA.....	I
SIMULATING RADIOLOGICAL ACCIDENTS THROUGH SOFTWARE AGENTS.....	III

Índice das Tabelas

Lista de Nomenclaturas

BDE	- Banco de Dados Espaciais
CNEN	- Comissão Nacional de Energia Nuclear
DEM	- Digital Elevation Model
IAEA	- International Atomic Energy Agency
IBTEN	- Instituto Boliviano de Tecnologia e Energia Nuclear
ONU	- Organização da Nações Unidas
RAD	- Rapid Application Development
REPAST	- Recursive Porous Agent Simulation Toolkit
SIG	- Sistema de Informação Geográfica
SMA	- Sistema MultiAgente
UNSCEAR	- United Nations Scientific Committee on the Effects of Atomic Radiation

Introdução

O aumento do uso de equipamentos, que utilizam fontes radioativas, em diversos setores da sociedade (indústrias, hospitais, clínicas médicas e odontológicas, geração de energia, conservação de alimentos, esterilização de produtos, aviação, extração petrolífera) gerou a necessidade de um maior controle nacional e internacional da exposição radioativa.

O Comitê das Nações Unidas para os Efeitos da Radiação Atômica (UNSCEAR), é o organismo científico da ONU encarregado de gerar relatórios dos níveis de radiação e dos efeitos causados pela exposição à radiação ionizante. A Agência Internacional de Energia Atômica (IAEA) é o órgão da ONU encarregado de definir e difundir os padrões de proteção e segurança para a radiação ionizante e garantir o uso pacífico das atividades nucleares. Muitos países possuem órgãos encarregados de fiscalizar o uso da radiação, prevenir acidentes radiológicos e seu impacto na sociedade. No Brasil, estas atribuições estão a cargo da Comissão Nacional de Energia Nuclear (CNEN) .

Todo esta estrutura é necessária em decorrência da magnitude e dos riscos envolvidos em possíveis acidentes radiológicos e nucleares. Em nosso país, vivemos uma situação de grande impacto, quando do acidente radiológico de Goiânia, em setembro de 1987. Neste evento, uma fonte de Césio-137, utilizada em equipamentos de radioterapia, foi dispersada acidentalmente junto à população, o que provocou , além da contaminação de áreas urbanas, a contaminação em doses elevadas de 249 pessoas, a contaminação interna e externa de outras 129 pessoas, alterações hematológicas de 20 pessoas e a morte de 4 pessoas.

As medidas defensivas e ações de emergência envolveram 575 profissionais durante 6 meses; a retirada de 200 habitantes de 41 casas, trabalho de descontaminação de 85 casas, demolição de 7 casas e a monitoração de 3,5 toneladas de rejeitos. Além dos efeitos psicológicos negativos, que traumatizaram a população por vários anos, foram impetrados vários processos judiciais custosos para a União. [1]

Isto nos faz concluir que o planejamento de repostas imediatas e de ações de emergências é fator crítico para estabelecer políticas de segurança. Modelos de simulação, utilizando a tecnologia de agentes, são ferramentas que podem ser empregadas com sucesso nestes casos. Elas podem representar o “mundo real” e a interação de pessoas, materiais e fontes radioativas. A construção destes modelos de simulação permitiriam o estudo prévio e mais detalhado dos ambientes envolvidos e das possíveis repercussões junto ao público em geral.

Em geral, para se estimar as doses, a que as pessoas expostas foram submetidas empregam-se as seguintes alternativas: i) realiza-se a reprodução do acidente com a utilização de modelos (*fantomas*) que representam o corpo humano; ii) efetua-se a análise de sangue das pessoas contaminadas, quando estas podem ser localizadas; iii) medições dos dosímetros que estavam sendo utilizados por pessoal técnico; iv) cálculo matemático. Pode-se observar a limitação de alguns destes procedimentos, pois ao reproduzir o acidente necessita-se o uso de fontes radioativas reais; para a análise hematológica é necessário a presença das pessoas, possivelmente, contaminadas; além de todo impacto social e psicológico que tais reproduções podem vir a provocar. A simulação através de *software* permite diminuir estas dificuldades, bem como cria um ambiente controlado para a realização dos experimentos.

O objetivo desta dissertação é mostrar como criar um ambiente computacional, onde um *software* baseado em agentes simula um acidente radiológico. Os agentes terão propriedades, tais como: mobilidade, reatividade e objetividade. Pessoas e fontes

radioativas serão representadas através de agentes, desta forma, através da simulação baseada em agentes, pretende-se estudar, analisar e gerenciar estes complexos sistemas que são os acidentes radiológicos.

Para alcançar o objetivo exposto, esta dissertação foi dividida em 5 capítulos: no capítulo 1, apresenta-se os conceitos de radioatividade e acidente radiológicos, e o modelo matemático que embasará a simulação. No capítulo 2, são enumerados os conhecimentos sobre o universo dos sistemas baseados em agentes, bem como suas características e propriedades. No capítulo 3, os fundamentos da construção de uma arquitetura de sistema baseado em agentes, com finalidade de simular acidentes radiológicos. São especificados os sistemas de informação geográfica, os princípios que regem a modelagem da simulação, bem como a ferramenta utilizada para a construção do modelo. No capítulo 4, estuda-se um acidente radiológico específico ocorrido em Cochabamba e bem documentado pela Agência Internacional de Energia Atômica (IAEA) [17]. Neste capítulo apresenta-se o simulador construído através do *software* de agentes *Repast*, bem como os resultados alcançados. No último capítulo (5), elencam-se as conclusões frutos desta dissertação.

1 Radioatividade e Acidentes Radiológicos

“A radioatividade natural foi descoberta por Becquerel, em 1896, através do estudo da fluorescência do mineral Urânio. Ele observou que estes minerais eram capazes de escurecer placas fotográficas, que ainda não haviam sido submetidas à luz. Este fato levou Becquerel a sugerir que o mineral Urânio emitia um tipo de radiação capaz de penetrar através de camadas de papel e ainda sensibilizar placas fotográficas virgens. Depois, o casal Curie através de novas experiências, em 1898, denominou esta forma de energia de radioatividade”. [8]

As radiações foram, a partir destes experimentos, classificadas em alfa, beta e gama. A radiação alfa possui baixo poder de penetração e alto poder de ionização (capacidade de formar um par de íons: negativo, elétron livre, e positivo, átomo sem um de seus elétrons). A radiação beta possui maior poder de penetração do que a alfa e menor poder de ionização. A radiação gama, que é uma onda eletromagnética, possui muito poder de penetração e pouco poder de ionização.

Podemos dizer que as radiações são produzidas por processos de ajustes que ocorrem no núcleo do átomo ou suas camadas eletrônicas, ou pela interação de outras radiações ou partículas com o núcleo ou com a camada eletrônica [8].

“A radiação ionizante representa ondas eletromagnéticas e partículas que podem ionizar, isto é, remover um elétron de um átomo ou molécula do meio pelo qual eles se propagam. A radiação ionizante pode ser emitida no processo de decaimento natural de alguns núcleos instáveis ou pela excitação de átomos e seus núcleos em reatores nucleares, ciclotrons, máquinas de raios-X ou outro instrumento”. [9]

“A radiação alfa é emitida quando núcleos instáveis e pesados, por exemplo, urânio, rádio, radônio ou plutônio, sofrem decaimento. A partícula alfa é um núcleo de hélio formado por dois prótons e dois nêutrons. É cerca de 8.000 vezes mais pesada que um elétron e tem o quádruplo de sua carga elétrica. Uma vez emitidas, elas se deslocam relativamente devagar (aproximadamente 1/20 da velocidade da luz) em virtude de sua carga elétrica e grande massa. Partículas alfas são facilmente ionizadas, e param rapidamente quando penetram em um material. A distância em que elas se

deslocam no ar é de poucos centímetros. Em materiais sólidos a distância é somente de poucos centésimos de milímetros. Se a radiação alfa atinge o corpo humano desprotegido, ela não consegue ultrapassar a camada de células mortas da pele, não causando, assim, danos. Uma substância que emite radiação alfa somente causará dano ao corpo humano se for inalada ou ingerida. Nestes casos, a radiação pode atingir células vivas e causar muito prejuízo.” [10]

“A radiação beta é emitida durante o decaimento radioativo de núcleos com excesso de nêutrons em relação a prótons. Exemplos de beta-emissores puros são os radionuclídeos Estrôncio-90 com meia vida de 27,7 anos e Trítio com 12,3 anos de meia vida. A partícula beta, que é um elétron, é emitida quando um nêutron em um núcleo é transformado em próton. A partícula beta é muito leve. Sua massa é cerca de $1/2.000$ da do próton. As partículas betas se ionizam com menos facilidade que as partículas pesadas e, portanto, elas possuem muito mais alcance. O caminho que percorrem é deformado. Seu alcance no ar pode ser de vários metros. Em tecido ou água, seu alcance é de vários milímetros. Óculos ou roupa grossa costumam ser suficientes para deter a radiação beta. A pele desprotegida exposta a uma intensa radiação beta pode queimar, como resultado de uma dose elevada sobre a pele. O grande risco associado aos emissores de radiação beta, ocorre na ingestão de alimentos ou na inalação de substâncias que contenham esta radiação.” [13]

Os raios gama são um conjunto de fótons eletromagnéticos. Os fótons gama são os fótons mais energizados no espectro eletromagnético. Eles são emitidos a partir do núcleo de átomos instáveis. Eles não possuem massa e nenhuma carga elétrica – eles são energia eletromagnética pura. A radiação gama, é um tipo de radiação ionizante, com elevada quantidade de energia. Os fótons gama têm 10.000 vezes mais energia do que os fótons na escala visível do espectro eletromagnético. Apesar dos raios gama e raios-X apresentarem o mesmo risco, eles diferem em sua origem. Os raios gama se originam no núcleo, enquanto os raios-X se originam nos campos elétricos que envolvem o núcleo.

A emissão de radiação gama ocorre quando o núcleo de um átomo radioativo possui muita energia e ao buscar uma estabilização emite esse excesso de energia em forma de onda eletromagnética.

Por causa da sua alta energia, os fótons gama se deslocam na velocidade da luz e percorrem centenas de milhares de metros no ar antes de perder sua energia. Eles podem atravessar muitos tipos de materiais, incluindo o tecido humano. Materiais muito densos, como o chumbo, são comumente usados como blindagem para deter os fótons gama.

Os raios gama existem enquanto tiverem energia. Uma vez que sua energia seja dissipada, seja no ar ou em um material sólido, eles cessam de existir. O mesmo acontece com os raios- X.

Por causa do poder de penetração dos raios gama e sua capacidade de se deslocar a grandes distâncias ele é considerado como o principal fator de risco para a população em caso de acidente radiológico.

“O processo de ionização em matéria viva necessariamente altera átomos e moléculas, ao menos temporariamente, e pode provocar danos às células. Se o dano celular ocorre e não é adequadamente reparado, isto pode impedir que a célula sobreviva ou impedir que desempenhe suas funções normais. Uma outra situação, é que isto pode resultar, também, em uma célula viável, mas modificada.” [9]

“A grandeza usada para expressar a exposição de um material à radiação, por exemplo, o corpo humano, é a dose absorvida, cuja unidade é o gray (Gy). No entanto, os efeitos biológicos por unidade da dose absorvida varia com o tipo de radiação e a parte do corpo que foi exposta. Para levar em conta estas variações, uma grandeza definida como dose efetiva, é usada e sua unidade é sievert (Sv).”

“Os efeitos da radiação são causados pelo danos inflingidos às células pelas interações radioativas. O dano pode resultar em morte celular ou modificações que podem afetar o funcionamento normal de órgãos e tecidos. A maior parte dos órgãos e tecidos não são afetados pela perda de células. Entretanto, se o número de perdas for muito grande, se poderá observar um prejuízo ao órgão ou ao tecido e, conseqüentemente, ao indivíduo. Este efeito só ocorre se a dose de radiação for suficientemente grande para matar um grande número de células. Este tipo de prejuízo

ocorre em todos os indivíduos que recebem uma dose de gravidade tal que exceda o limite tolerável e seu efeito é denominado “determinístico” .”

Se a célula não morre, mas é apenas modificada pela radiação, o dano na célula é geralmente reparado. Se a correção não for perfeita, a modificação será transmitida para as células filhas e pode gerar câncer no tecido ou no órgão que foi exposto. Se as células afetadas são as que transmitem informações genéticas para os descendentes, distúrbios hereditários podem surgir. Tais efeitos nos indivíduos ou em seus descendentes são chamados de “estocásticos”, que significam de natureza aleatória.

Em resumo, efeitos determinísticos (agudo) ocorrerão somente se a dose de radiação for substancial, tal como ocorre em acidentes. Efeitos estocásticos (câncer e efeitos hereditários) podem ocorrer em uma única célula. Como a dose sobre um tecido cresce a partir de um nível mais baixo, cada vez mais células são danificadas e a probabilidade de efeitos estocásticos acaba aumentando.”[9]

Para a avaliação da dose recebida por indivíduos é preciso levar em consideração a taxa de liberação da fonte radioativa, o seu ponto de localização, a existência de objetos entre a fonte e os indivíduos (blindagem) e o tempo de exposição do indivíduo.

1.1 Acidentes em Instalações Nucleares e Radiativas

Acidentes radiológicos são acidentes que ocorrem em instalações nucleares e radiativas e são caracterizados pela existência de campos intensos, intencionais ou não, de radiação, não controlada, liberada no ambiente por uma quantidade de material radioativo e que envolva exposição ou contaminação de seres humanos ou do ambiente, sendo capaz de causar sérios prejuízos ou morte.

Existem duas formas nas quais os seres humanos podem ser expostos a radiação – e elas podem ocorrer simultaneamente. A fonte de radiação pode estar totalmente fora do corpo – e é então chamada de irradiação externa. Aqui a radiação atinge o indivíduo e é absorvida, dependendo de suas características físicas. A energia será absorvida, mas a pessoa não se torna radioativa.

Na outra forma, a radiação pode se depositar na pele. Este material radioativo pode entrar no corpo através do ar, ser inalado ou ingerido, produzindo o que é chamado de irradiação interna ou contaminação radioativa.

A liberação acidental de radionuclídeos no ambiente pode causar a contaminação de áreas e pessoas. Então, é necessário fazer uso de ferramentas que permitam um diagnóstico dos efeitos da exposição da população, avaliar as consequências e sugerir medidas de proteção. [14]

A população, em geral, está exposta à radiação natural. A radiação de origem cósmica é aquela produto de substâncias que existem dentro da Terra e dentro do corpo humano (família Urânio 238 [U-238], Tório 232 [Th-232] e Potássio 40 [K-40]). Uma contribuição significativa à exposição natural dos seres humanos é proporcionada pelo gás radônio, que emana do solo e pode estar concentrado nas habitações.[9]

No entanto, em virtude de acidentes podemos ficar expostos à radiação não natural que se origina de exposição por tratamento médico, partículas liberadas (*fallout*) por causa de testes nucleares, geração de energia e acidentes radioativos propriamente ditos (*Figura 1*). A fim de prevenir estes acidentes são sistematizados procedimentos de segurança que configuram a radioproteção. Esta se constitui na disciplina que trata dos efeitos das radiações ionizantes sobre seres vivos e da proteção do pessoal contra os efeitos nocivos destas radiações. É necessária, então, a realização de ações urgentes para diminuir os possíveis efeitos da radiação. Em primeiro lugar, precisamos identificar a quantidade de pessoas contaminadas, grau ou dose de radiação recebida e um mapa mais elaborado (informações georeferenciadas) das áreas atingidas [15]. Se soubermos com antecedência a magnitude do acidente radiológico, poderemos evitar alarmar a população desnecessariamente, o que seria uma resposta inadequada ao evento. Isto poderia causar um dano social muito maior do que o acidente propriamente dito.



Figura 1. Acidente radiológico

Existe uma grande quantidade de equipamentos (industriais e médicos) que utilizam fontes radioativas seladas sujeitas a algum tipo de erro na utilização e transporte, o que pode provocar contaminação. Estes acidentes radioativos podem causar impacto na população, no ambiente e nas atividades sociais e econômicas.

Um acidente pode comprometer os serviços de transporte, gerar zonas de exclusão nas áreas ou locais contaminados, levando ao deslocamento de pessoas; comprometer o abastecimento de água; e superlotar os serviços hospitalares, causando sérios problemas sociais [15]. Basicamente acidentes podem ser reduzidos através de adoção de medidas de radioproteção (por exemplo, o operador usar um medidor de radiação para confirmar que a fonte está no interior da blindagem). [22]

A exposição a elevados níveis de radiação pode causar dano às células vivas, matando algumas ou modificando outras. Tais efeitos (*Figura 2*) mostram que a radiação tem sido associada à leucemia, a diversas formas de câncer, dermatite, catarata, esterilidade e aberração cromossômica. [13]

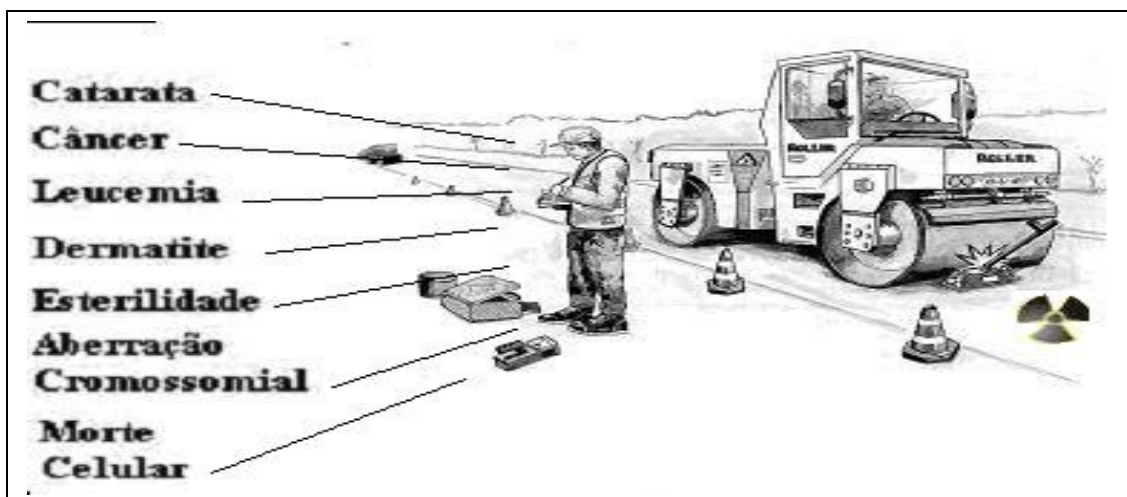


Figura 2. Efeitos da radiação

1.2 Medição da Radioatividade

Para a determinação da dose recebida por indivíduos é necessário levar em consideração a intensidade da fonte radioativa, sua exata distância dos indivíduos expostos, a existência de blindagem entre a fonte radioativa e as pessoas expostas e o tempo de exposição destas pessoas.

Existem quatro grandezas utilizadas para determinação da radiação: Atividade, Taxa de Exposição, Dose Absorvida e Dose Equivalente.

i) Atividade de uma fonte radioativa

A atividade de uma fonte radioativa é caracterizada pelo número de desintegrações nucleares ou transformações que ocorrem em um determinado intervalo de tempo. A atividade é proporcional ao número de átomos excitados que existem em um elemento radioativo e pode ser expresso pela fórmula:

$$A=A_0e^{-\lambda \cdot t} \quad (1)$$

Onde : A_0 , é a atividade no tempo $t=0$, e λ é a constante de desintegração, que significa a taxa na qual a desintegração ocorre.

A unidade do Sistema Internacional para a atividade é becquerel (Bq), que indica o decaimento ou desintegração do material radioativo ocorrido em 1 segundo. O becquerel é uma unidade pequena. Em situações práticas, a radioatividade é quantificada em kilobecquerels (kBq) ou megabecquerels (MBq). O curie (Ci) é, também, comumente usada como unidade para atividade em determinados tipos de fontes. O curie é a quantidade de material radioativo no qual $3,7 \times 10^{10}$ átomos são desintegrados por segundo. Isto é, aproximadamente, a quantidade de radioatividade emitida por 1 grama de **Ra**²⁵⁶.

ii) Exposição

A Exposição é a medida da força do campo radioativo em um determinado ponto do ar. É a medida de ionização das moléculas em uma massa de ar. É comumente definida como a quantidade de carga (a soma de todos os íons de mesmo sinal) produzida em uma unidade de massa de ar, quando a interação dos fótons for completamente absorvida nesta massa. Deste modo, a unidade de medida de exposição é o Coulomb/kg. A exposição radioativa é associada com o efeito da radiação produzida nos seres vivos. A unidade mais comumente usada para a taxa de exposição é o Roentgen (R).

A taxa de exposição é definida como a variação da exposição no tempo, usualmente medida em Roentgens por hora (R/h). A taxa de exposição pode ser associada a atividade gama de uma fonte radioativa através da seguinte fórmula:

$$X = \Gamma . A / d^2 \quad (2)$$

Onde:

X = taxa de exposição, em R/h (Roentgen / hora)

A = atividade da fonte, em Ci (Curie)

d = distância entre a fonte e o ponto de medida, em metros (m)

Γ = uma constante característica de cada fonte radioativa, também, conhecida como fator gama, em (R.m²) / (h.Ci)

iii) Dose Absorvida

A grandeza básica usada para expressar a exposição sofrida por um material como, por exemplo, o corpo humano, é a dose absorvida. Enquanto a exposição é definida para o ar, a dose absorvida é a quantidade de energia que a radiação ionizante incorpora a uma determinada massa de matéria. A unidade Internacional para a dose absorvida é o gray (Gy), que é definida como a dose de um joule por quilograma. [16]

A grandeza de dose absorvida é expressa através de uma fórmula que depende da atividade da fonte radioativa, da distância da fonte e do tempo de exposição a esta fonte:

$$D = X.t \quad (3)$$

Onde,

D = dose absorvida, em Gray (Gy)

X = taxa de exposição, em R/h

t = tempo, em horas (h)

iv) Dose Equivalente

Quando se considera a interação da radiação com tecidos vivos é importante, também, levar em conta o tipo de radiação. Muito embora, os efeitos da radiação sejam dependentes da dose absorvida, alguns tipos de radiação produzem efeitos diferenciados relativamente a outros, que representam a mesma quantidade de energia dissipada. Por exemplo, para idênticas doses absorvidas, as partículas alfa podem ser 20 vezes mais prejudiciais do que as partículas beta. Afim de, se levar em conta estas variações, quando descrevemos os riscos a saúde humana provocado pela exposição à radiação, uma grandeza denominada dose equivalente (DE) é usada. Ela é a dose absorvida multiplicada por um fator de ajuste ou qualidade, que indica o dano biológico potencial que um determinado tipo de radiação provoca.

O fator de qualidade (FQ) é usado em radioproteção para valorar a dose absorvida com relação ao seu efeito biológico presumido. Radiação como FQ elevado causará grande dano ao tecido vivo. O rem é um termo usado para descrever uma unidade especial de dose equivalente. Rem é a abreviação de “roentgen equivalent in

man.”. A unidade do sistema Internacional (SI) é o sievert (Sv); um rem é equivalente a 0,01 Sv. Doses de radiação recebidas por trabalhadores são registradas em rem, no entanto, sievert tem sido usado como uma transição para o Sistema Internacional de unidades. [16]

$$DE = D.FQ \quad (4)$$

Onde,

DE = dose equivalente, em Sievert (Sv)

D = dose absorvida, em Gray (Gy)

FQ = fator de qualidade da radiação, para radiação gama igual a 1

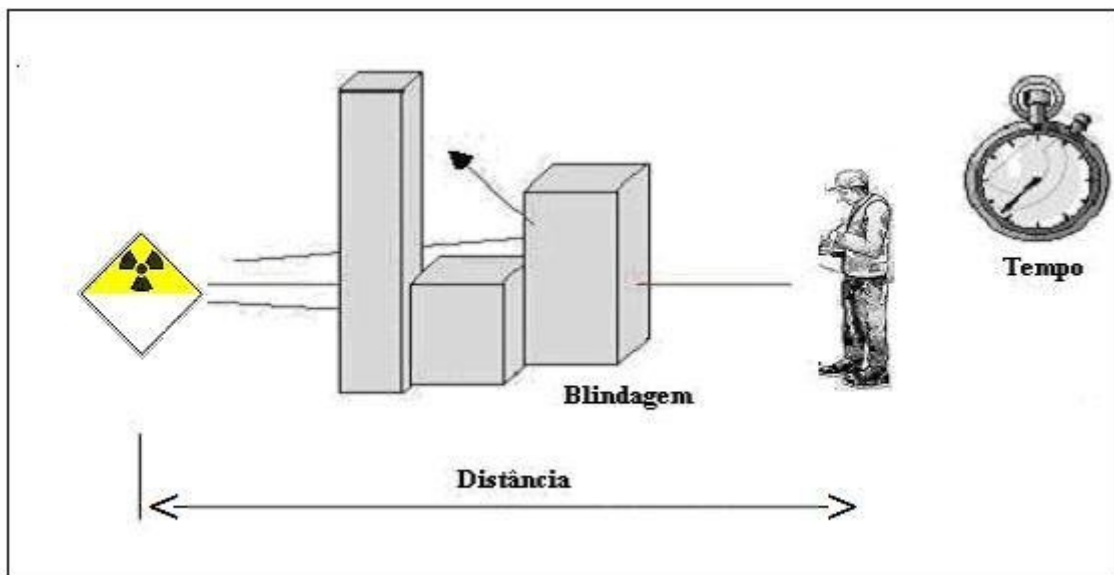


Figura 3. Tempo, distância e blindagem: princípios da radioproteção

Princípios de radioproteção nos levam a identificar três fatores que influenciam na dose recebida por uma pessoa: a distância, o tempo e blindagem. A distância indica que quanto mais afastado estivermos de uma fonte radioativa, menos radiação receberemos. O tempo indica que quanto mais tempo ficarmos expostos a uma fonte radioativa mais radiação estaremos recebendo. É essencial exercermos o mais rapidamente possível qualquer ação junto à fonte, a fim de que o tempo de exposição à mesma não seja elevado. A blindagem indica que, dependendo do material utilizado, tais como chumbo, ferro, concreto, é possível diminuir a exposição à radioatividade do corpo humano (*Figura 3*).

2 Sistemas Baseados em Agentes Autônomos

Por milhares de anos as pessoas têm criado modelos para ajudá-las a compreender o mundo. Estes modelos não só ajudam a entender o fenômeno em estudo, mas, também, ajudam a transmitir suas idéias para outras pessoas. Através da simulação computacional de modelos baseados em agentes, as pessoas podem perceber, com detalhes, o efeito da contaminação radioativa em caso de acidentes, o que torna mais fácil entender o risco envolvido e os danos causados em tais situações.

A palavra “agente” se refere a todas as entidades que possuem a habilidade, capacidade e permissão para agir como representante de outro ente. Falando em termos humanos, agentes podem ser pessoas com mais conhecimento ou mais recursos específicos em determinada área, de modo a ajudar outras pessoas em suas tarefas. Por exemplo, uma secretária pode decidir situações no lugar do seu chefe, substituindo-o em determinadas tarefas, tais como tomar notas de apontamentos de visitas e encontros, agendar horários, etc. [2]

Pode-se dizer, também, que um agente é uma entidade computacional com um comportamento autônomo que lhe permite escolher suas próprias ações [19]. A decisão de qual ação levar a cabo é determinada pelo agente, tendo em consideração as mudanças acontecidas no ambiente em que atua e o desejo de alcançar seus objetivos. A idéia principal em um sistema MultiAgente (SMA) é que um comportamento global inteligente pode ser alcançado a partir do comportamento individual dos agentes.

Segundo Wooldridge e Jennings [20], agentes são sistemas que apresentam um comportamento determinado por um processo de raciocínio baseado na representação de suas atitudes, tais como crenças, comprometerimentos e desejos. Eles acreditam que um sistema pode ser visto como um agente se ele possuir as seguintes propriedades: autonomia, habilidade social, reatividade, pró-atividade.

Outra definição importante nos faz Franklin e Graesser [20] ao definir um agente autônomo como um sistema situado em um meio ambiente, do qual ele faz parte e percebe, agindo sobre o mesmo continuamente, num período de tempo, buscando suas ações em sua própria agenda, podendo suas ações afetar suas percepções futuras.

Também para Wooldridge e Jennings [20], um agente humano possui olhos, ouvidos, e outros órgãos de sensoriamiento, e mãos, pernas, boca, e outras partes do corpo como atuadores. Um agente robótico utiliza câmeras e sensores (ultra-som, infravermelho) como sensores e vários motores como atuadores. Já um agente de *software* possui *strings de bits* codificados como suas percepções e ações.

A partir destas definições pode-se dizer que é possível extrair algumas características comuns aos agentes: devem estar inseridos em um ambiente, devem ter capacidade de sensoriar o seu entorno e atuar sobre ele, e devem agir segundo os objetivos para os quais foram implementados. Portanto, devem possuir uma série de atributos ou propriedades que os definem como agentes (não necessariamente o agente deve apresentar todas estas propriedades) [21].

Pode-se destacar, assim, como características principais de um agente:

- Autonomia: os agentes podem operar sem a intervenção de humanos ou de outros agentes;
- Sociabilidade: os agentes são capazes de interagir com outros agentes (humanos ou não) por meio de uma linguagem de comunicação entre agentes;
- Reatividade: os agentes são capazes de perceber estímulos do seu entorno e reagir a ditos estímulos;
- Pró-atividade, iniciativa: os agentes não são apenas entidades que reagem a um estímulo. Tem caráter empreendedor e podem atuar guiados por seus objetivos;
- Continuidade temporal: os agentes estão continuamente ativos (podem ser processos/*threads* executando em *background* ou *foreground*). A maioria dos programas existentes, ao contrário dos agentes, executa uma ou mais tarefas e termina a sua execução;
- Orientação por objetivos: um agente é capaz de resolver tarefas complexas, pois decompõe-nas em sub-tarefas menos complexas, decidindo a ordem em que tentará resolvê-las e como irá subdividi-las para atingir seus objetivos;
- Mobilidade: capacidade de um agente de locomover-se através de uma rede de computadores;

- Veracidade: assunção de que um agente não comunica informação falsa de propósito;
- Benevolência: assunção de que um agente está disposto a ajudar outros agentes se isto não entra em conflito com seus próprios objetivos;
- Racionalidade: assunção de que um agente atua de forma racional, tentando cumprir seus objetivos, se são viáveis;
- Adaptabilidade: o agente deverá adaptar-se aos hábitos, métodos de trabalho e preferências do utilizador. Um agente com capacidade de adaptação é capaz de alterar o seu comportamento com base na experiência (aprendizagem).

Desta forma pode-se simular o comportamento de pessoas em um determinado ambiente através da tecnologia computacional de agentes, bem como elementos que interagem com estas pessoas usando esta mesma tecnologia, observando seu comportamento e as conseqüências desta interação.

Muitas definições de agentes autônomos coexistem, porque os autores adotam diferentes perspectivas para este mesmo assunto.[3],[4],[2]. Nesta dissertação adotamos a seguinte definição da referência [5]: *“Um sistema situado dentro de um dado ambiente, que interage com este ambiente através de mecanismos de percepção, e atua neste ambiente e/ou em outros agentes, à medida que o tempo passa, em perseguição de suas próprias prioridades, planos ou crenças. Eventualmente, os mecanismos de percepção/ação do agente se desenvolvem com o tempo.”*

Podem-se distinguir duas noções extremas de agentes [3]:

- Noção Fraca: é uma noção considerada relativamente insatisfatória, a qual consiste basicamente em definir um agente como sendo uma entidade capaz de intercambiar mensagens por meio do uso de uma linguagem de comunicação de agentes (habilidade social). Segundo essa noção, além da habilidade social, o agente mostra as seguintes propriedades: autonomia, reatividade e pró-atividade.
- Noção Forte: este enfoque apresenta uma definição mais abrangente e satisfatória, na qual um agente, além de apresentar as propriedades de

autonomia, habilidade social, reatividade e pró-atividade, se define como uma entidade implementada utilizando-se propriedades aplicáveis à pessoas, como por exemplo, conhecimento, crenças e intenções (estados mentais). Vários outros atributos são geralmente discutidos no contexto de agente: mobilidade, veracidade, benevolência e racionalidade.

A classificação dos agentes pode ser efetuada de diversas formas, sendo que a mais usual é aquela realizada de acordo com a linha de pesquisa e desenvolvimento, na qual é priorizado a função ou objetivo principal do agente. Dessa forma, podemos realizar a seguinte classificação:

- Agentes de interface: também chamados *assistentes pessoais* ou *agentes de usuário*, possuem objetivo de simplificar as tarefas rotineiras realizadas por um usuário. Esse tipo de agente observa e monitora as ações tomadas pelo usuário na interface, aprende novos “atalhos” e sugere melhores formas de desenvolver a tarefa. O agente de interface age como um assistente pessoal autônomo, o qual coopera com o usuário na realização de determinadas tarefas. Ele aprende e cria o perfil do usuário basicamente de quatro formas:
 1. Observando e imitando o usuário;
 2. Mediante recebimento de um *feedback* positivo ou negativo por parte do usuário;
 3. Recebendo instruções explícitas do usuário;
 4. Mediante ajuda de outros agentes.
- Agentes colaborativos: enfatizam a autonomia e a cooperação (com outros agentes) de maneira a executar tarefas para seus proprietários. Eles podem ter algumas características de aprendizado, mas este aspecto não é usualmente o foco central das suas operações. De forma a ter um grupo coordenado de agentes colaborativos, eles devem ser capazes de negociar entre si, a fim de encontrar soluções mutuamente aceitáveis em determinados assuntos.

- Agentes móveis: representam um novo paradigma na computação distribuída. O conceito de mobilidade implica na capacidade do agente de se deslocar entre diversas máquinas, evitando dessa maneira uma sobrecarga de comunicação ou permitindo-lhes utilizar recursos não existentes em sua máquina de origem. Um dos principais problemas relacionados a este tipo de agente é o relativo à segurança – um agente dessa categoria seria, por exemplo, um excelente disseminador de vírus dentro da rede na qual estivesse inserido.
- Agentes de recuperação de informação: também conhecidos como agentes de Internet. Têm o seu desenvolvimento favorecido pela grande quantidade de informação disponível na Internet e a dificuldade gerada para encontrar e indexar essa informação de forma a fornecê-la ao usuário. Um exemplo desse tipo de agentes são os *SoftBots* os quais permitem o cumprimento de objetivos de alto nível como, por exemplo, buscar os artigos referentes ao ano de 2001 de Pattie Mae; o *SoftBot* teria que realizar a busca dos arquivos, solicitá-los por correio eletrônico, etc.

Quanto ao método utilizado para dotar o agente de inteligência (ou comportamento inteligente), podemos classificar os agentes em três grupos distintos de arquiteturas, dependendo do tipo de processamento utilizado [3]:

- Agentes Deliberativos (também denominados Simbólicos ou Cognitivos): contém um modelo do mundo, possivelmente incluindo eles mesmos. Tal modelo é de certa forma pré-concebido, mas seu estado é alterado pelo agente em resposta às novas informações sobre o ambiente, percebidas pelos sensores dos agentes. Com base na interpretação deste modelo, o agente estima quais ações serão necessárias para atingir o objetivo proposto, executando, então, as ações que levarão a sua realização.
- Agentes Reativos (ou Reflexivos): estes não modelam o mundo para determinar suas ações. São geralmente agentes simples que possuem um mapeamento de situações e respostas associadas. Dessa forma, quando um estado ambiental é alterado, o agente executa a ação correspondente para satisfazer o novo estado. Este processo é conhecido por estímulo-resposta.

- Agentes Híbridos: mostram-se como uma alternativa para dotar o agente com capacidades reativas apropriadas, visando solucionar a incapacidade de ação adequada por parte de um agente puramente deliberativo no momento em que o mesmo deve tomar uma decisão rápida e espontânea ao enfrentar uma situação imprevista. Em contra-partida, servem também para proporcionar a um agente puramente reativo, capacidade de raciocínio e planejamento quando o mesmo se deparar com uma situação na qual o ambiente diverge bastante dos seus objetivos iniciais.

Não necessariamente um agente deve apresentar todas as características mencionadas anteriormente, posto que seria extremamente difícil implementar um agente que incorporasse todas aquelas características. Dado isso, o conjunto de características componentes de um agente vai depender do tipo de aplicação a que ele se propõe.

Os agentes podem melhorar sua coordenação e coerência gerenciando o quê, como e quando se comunicam entre si. Por meio da comunicação os agentes podem ter uma visão menos local do problema e, desta forma, adquirir o conhecimento e a sincronia (em relação aos demais agentes) necessária à resolução do problema. Deve-se levar em conta, que se a comunicação entre agentes for excessiva, estará sendo criada uma sociedade de agentes *burocráticos* onde a carga de comunicação poderá ser maior que o trabalho efetivamente realizado.

As formas mais usuais de comunicação entre agentes são [3]:

- Sem comunicação: os agentes podem interagir sem existência de comunicação entre eles, podendo o agente inferir as ação de outros agentes. Esta situação pode ocorrer em virtude de falhas de *hardware*, impossibilidade de comunicação ou pelo desejo de uma maior autonomia por parte dos agentes.
- Comunicação primitiva: comunicação restrita a um número de sinais com interpretação fixa. Tem sido aplicada em SMA para resolução de conflitos entre dois agentes mediante um mediador, mas a coordenação que pode ser obtida é bastante limitada;

- Quadro-negro: consta de três componentes principais: o quadro-negro, um conjunto de Fontes de Conhecimento (*KSs*, *Knowledge Sources*) e um mecanismo de controle. O quadro-negro é uma base de dados global contendo dados e hipóteses (soluções parciais potenciais). É frequentemente utilizado o conceito de agenda para gerenciar a concorrência no acesso ao quadro-negro. Esta agenda é gerenciada por um monitor que ativa as fontes de conhecimento adequadas, de acordo com o conteúdo do quadro-negro. Este tipo de comunicação provém da metáfora de um grupo de pessoas que resolve um problema conjuntamente, utilizando “tempestade de idéias“. Ou seja: uma pessoa escreve no quadro uma idéia para a solução do problema em foco, outra dá uma nova idéia ou sugere uma modificação à(s) idéia(s) já existente(s) e escreve no quadro, e assim por diante, até chegar a uma solução que agrada a todos os membros do grupo.
- Passagem de mensagens: permite que um agente envie uma mensagem a um ou mais agentes cujos nomes deve conhecer. A diferença principal em relação ao quadro-negro, é que os agentes devem conhecer o seu entorno para saber exatamente a qual agente deve enviar a mensagem. Estas mensagens podem ser tanto de solicitação de serviços e informações, quanto de respostas a tais solicitações.

A comunicação de alto nível tem sido uma meta perseguida para se estabelecer as interações entre os agentes no nível de conhecimento, ao invés de no nível simbólico. Desta forma, tenta-se fazer com que os agentes compreendam as intenções, desejos e objetivos de outros agentes, e que os mesmos sejam capazes também de transmitir estes desejos, objetivos e intenções por meio do uso de teorias provenientes da área da linguagem natural, especialmente na análise e geração de discurso.

Outra modalidade de comunicação, também, muito pesquisada tem sido a comunicação entre um agente artificial e um agente humano. Para isto duas abordagens básicas são utilizadas: i) encapsular o agente humano modelando suas interações em uma linguagem de comunicação de agentes; ii) aproveitar a tecnologia multi-agente para simplificar as interfaces homem-máquina.

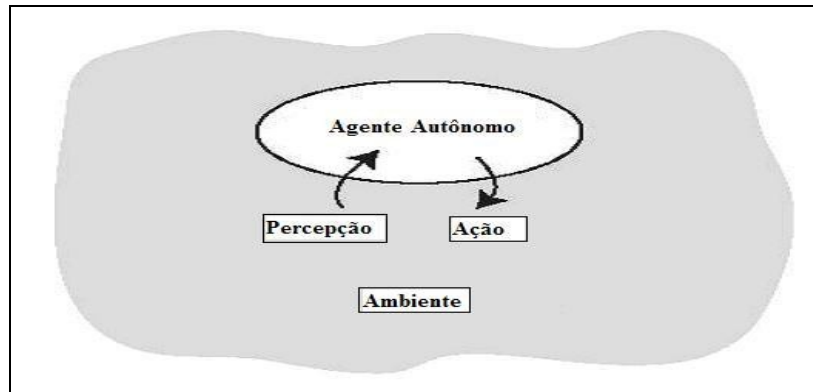


Figura 4. Agente autônomo

Sistemas MultiAgentes (SMA) são compostos de muitos agentes, que, além das características anteriormente mencionadas, podem interagir entre si através de mecanismos de comunicação. Na maioria dos casos estes agentes podem ser completamente diferentes entre si e ter um comportamento competitivo ou colaborativo, de acordo com a situação definida [2].

Os agentes são capazes de ações independentes e autônomas (*Figura 4*), a fim de cumprir com as tarefas que lhes foram delegadas. Eles são, também, capazes de interagir (cooperação, negociação, etc.) com outros agentes. Muitas vezes precisamos que sistemas de computação sejam capazes de decidir por eles mesmos o que precisam fazer, a fim de atender suas funcionalidades projetadas. Estes sistemas são chamados de agentes. Um agente inteligente é capaz de ações flexíveis e autônomas (*Figura 5*). Onde a flexibilidade significa: reatividade (reagir a estímulos/situações do meio ambiente), pró-atividade (ter iniciativa) e sociabilidade (capacidade de interagir com outros agentes).

Agentes possuem um modelo de decisão que utiliza o raciocínio do tipo *means-end*. É um processo de decidir como alcançar um fim (uma intenção que se têm) usando meios disponíveis (ações que se deseja executar). É conhecido na comunidade de inteligência artificial como planejamento. Planejamento é essencialmente programação automática. As entradas do sistema são:

- A meta, intenção ou tarefa. É alguma coisa que o agente deseja alcançar.
- O estado atual do ambiente.
- As ações disponíveis para o agente.

Como *output* um algoritmo de planejamento gera um plano. Espera-se que, ao ser executado o plano, a partir do estado em que o mundo é descrito, o objetivo, intenção ou tarefa seja alcançado.[2]

A arquitetura do tipo *subsumption* possui duas características essenciais: A primeira é que o processo decisório de uma agente é realizado através de um conjunto de comportamentos voltados à execução de tarefas; cada comportamento pode ser pensado como uma função de ação individual, que continuamente recebe entradas sensoriais (*perceptual input*) e os mapeia em ações a serem realizadas. Cada um desses módulos de comportamento está associado ao cumprimento/execução de uma tarefa. Os módulos para a execução das tarefas não incluem qualquer representação simbólica complexa e não incorporam qualquer forma de raciocínio simbólico. A segunda característica é que neste tipo de arquitetura vários comportamentos podem se disparados simultaneamente. Um mecanismo deve existir para selecionar um dentre os vários comportamentos possíveis a partir de uma dada situação.[2]

Podemos modelar o comportamento de pessoas em alguns determinados ambientes, como, também, considerar outros elementos que interagem com estas pessoas, observando o seu comportamento e as consequências dessas interações, através da tecnologia computacional de agentes.

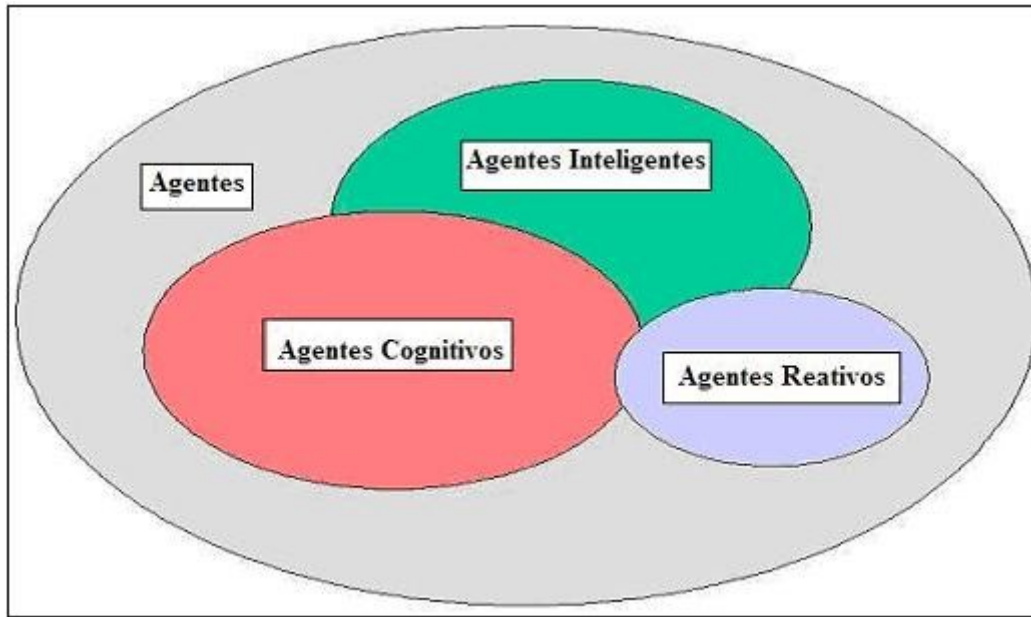


Figura 5. Tipos de agentes

3 Arquitetura de um sistema baseado em agentes simulando um acidente radiológico

A implementação de modelos de sistemas complexos e dinâmicos torna-se uma tarefa de alta complexidade quando não se dispõe de ferramentas que permitam um processo de construção, análise e descrição de modelos sem a necessidade de avançados conhecimentos de matemática e programação. Através de ferramentas de simulação, pode-se construir e explorar modelos que nos auxiliam no entendimento mais profundo do comportamento de um processo específico.

Não é desejável que, através de simulação pretendida, identifique-se apenas a fonte radioativa e áreas contaminadas, mas o foco é a quantidade de pessoas irradiadas que receberam significativa dose e a dose efetiva recebida. Os casos mais gerais, em que

tanto a fonte como os indivíduos expostos encontram-se em movimento relativo entre si, deve ser capaz de ser simulado (Figura 6).

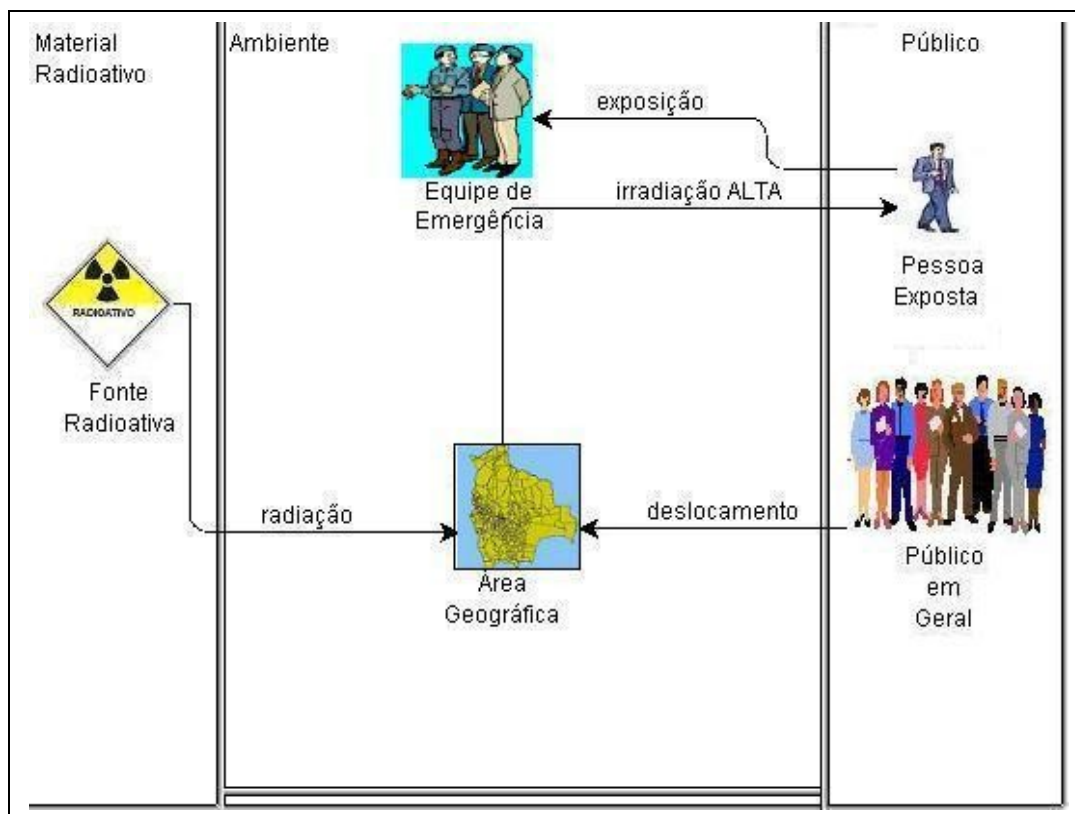


Figura 6. Arquitetura do sistema de simulação.

Através de uma simulação se poderia formar um sistema de resposta capaz de minimizar as conseqüências de um acidente radiológico, bem como aumentar a capacidade de gerência do risco envolvido.

Desta forma existe a necessidade de *software* ou sistemas que estimem, em tempo hábil, as possíveis doses recebidas e quantidade de pessoas contaminadas, a fim de permitir uma tomada de decisão quanto aos procedimentos mais urgentes de descontaminação e de proteção da população.

O objetivo desta dissertação é, através de um estudo de caso, explorar as potencialidades de sistemas baseados em agentes na simulação de acidentes radiológicos, coletando e agregando uma série de informações úteis relacionadas a um dado acidente com material radioativo. Estas informações permitem estimar: i) a

quantidade de pessoas que foram expostas; ii) a dose efetiva que elas receberam; iii) a localização e a extensão das áreas contaminadas.

Nem sempre é possível determinar quais são as pessoas que foram expostas, tampouco o seu número exato. Este seria o caso, por exemplo, de um hipotético ataque terrorista com uma fonte radioativa em um sistema público de transporte (trem ou metrô). Não obstante, é muito importante para o sistema público de saúde, ter uma estimativa de pessoas envolvidas e a dose efetiva que receberam, com o objetivo de calcular os recursos médicos que serão necessários para encarar o problema, para prevenir a população dos riscos da exposição e, também, para gerenciar o evento como um todo: procedimentos de descontaminação, acondicionamento de rejeitos, infraestrutura para suporte de equipes de emergência, etc.

Sistemas baseados em agentes podem ser úteis na simulação dos riscos associados com o transporte de fontes radioativas em áreas urbanas. Neste caso, pode-se escolher a melhor rota, e, em caso de acidente radiológico, identificar qual seria a rota que contaminaria o menor número de pessoas e qual a que causaria menos impacto ao meio ambiente e à sociedade.

O modelo proposto nessa dissertação baseia-se na fórmula de dispersão radioativa no ar (equação 5), levando em conta o radionuclídeo e sua atividade, a distância das pessoas expostas ao elemento radioativo, o tempo de exposição, como também, possíveis blindagens. Considera, também, movimentos eventuais das pessoas e da fonte radioativa. Ambos são representados por agentes reativos. Os efeitos de blindagem podem, da mesma forma, serem incorporados ao modelo. Basta, na equação 4, incluir-se um fator $0 \leq k \leq 1$, que dependerá do tipo e da espessura do material utilizado como blindagem.

$$X = k \cdot \Gamma \cdot A / d^2, 0 \leq k \leq 1 \quad (5)$$

O agente que simula a fonte radioativa tem as seguintes variáveis de estado: i) O identificador de fonte; ii) sua posição, dada por um conjunto de coordenadas (x, y, z); iii) sua atividade A ; iv) O fator Γ para a fonte específica; v) O fato de qualidade (FQ) usado para quantificar a dose absorvida com relação aos efeitos biológicos presumidos. Um agente típico representativo de uma pessoa possui o seguinte conjunto de variáveis de estado: i) A identificação de pessoa; ii) Sua posição, dada por um conjunto de coordenadas (x, y, z); iii) tempo de exposição; iv) Efeitos de blindagem; v) A dose absorvida; vi) A dose efetiva.

O modelo precisa, também, de uma representação do espaço geográfico, o ambiente onde os agentes captam estímulos como *input* e produzem ações como *output*. Normalmente, Sistemas de Informação Geográfica (SIG) usam estruturas do tipo *raster* ou vetorial para representar o espaço bidimensional. Em alguns casos, uma terceira grandeza é utilizada para se representar o espaço tri-dimensional, através de modelos de elevação digital (*digital elevation model - dem*) do terreno. Dada uma representação espacial de um Sistema de Informação Geográfica - SIG (um *shape file*, por exemplo), acrescenta-se a ela os agentes, com a finalidade de simular a dinâmica de um acidente radiológico. A representação espacial SIG é o ambiente ou o local, onde os agentes do modelo irão atuar.

Para cada fenômeno específico, interessam somente as informações particulares do ambiente. Então, considerando o espaço geográfico onde um fenômeno se desenvolve, precisamos filtrar os aspectos do ambiente que interessam, isto é, as propriedades que irão compor este ambiente como visto pelos agentes, e por todos os processos a serem simulados. Esta não é uma tarefa difícil, porque os dados são organizados em um *software* SIG em diferentes camadas (*layers*), tais como: serviços públicos, rios e lagos, mapas do solo, lotes, etc. É preciso selecionar somente camadas que interessam. Objetos que não são de interesse devem ser descartados, porque um ambiente cheio de informações supérfluas complicará a implementação e reduzirá a performance da simulação. Desta forma, dependendo do problema, pode-se simplificar consideravelmente a simulação como se verá no caso de estudo analisado nesta dissertação. [6], [7]

No estudo de caso analisado, o acidente radiológico em Cochabamba [17], um *shapefile* fornece a representação geográfica do ambiente, contendo uma fonte radioativa e a movimentação de indivíduos (passageiros) utilizando o conceito de agentes. A cada simulação do movimento do agente (passageiro) se determina a sua posição espacial em relação à fonte radioativa, desta forma obtém-se as variáveis para o cálculo das doses a que estes passageiros foram expostos.

3.1 Utilização de SIG no Mapeamento de Áreas Geográficas

A Sociedade da Informação para oferecer um benefício ao exercício de nossas atividades exige uma estrutura tecnológica que disponibilize em tempo hábil e de forma estruturada os dados que queremos utilizar. Em Geomática este paradigma se configura na dualidade Sistema de Informação Geográfica (SIG) + Banco de Dados Espaciais (BDE) (*Figura 7*).

O uso de métodos de levantamento e *softwares*, conhecidos como sistemas de informação geográfica (SIG), têm sido adotados em diferentes áreas de trabalho. Toda atividade que necessita de informações georeferenciadas (possuem localização no espaço geográfico), pode usar um SIG. O potencial do SIG e do sensoriamento remoto facilita o planejamento e a implementação do controle e gerenciamento de extensas áreas geográficas.

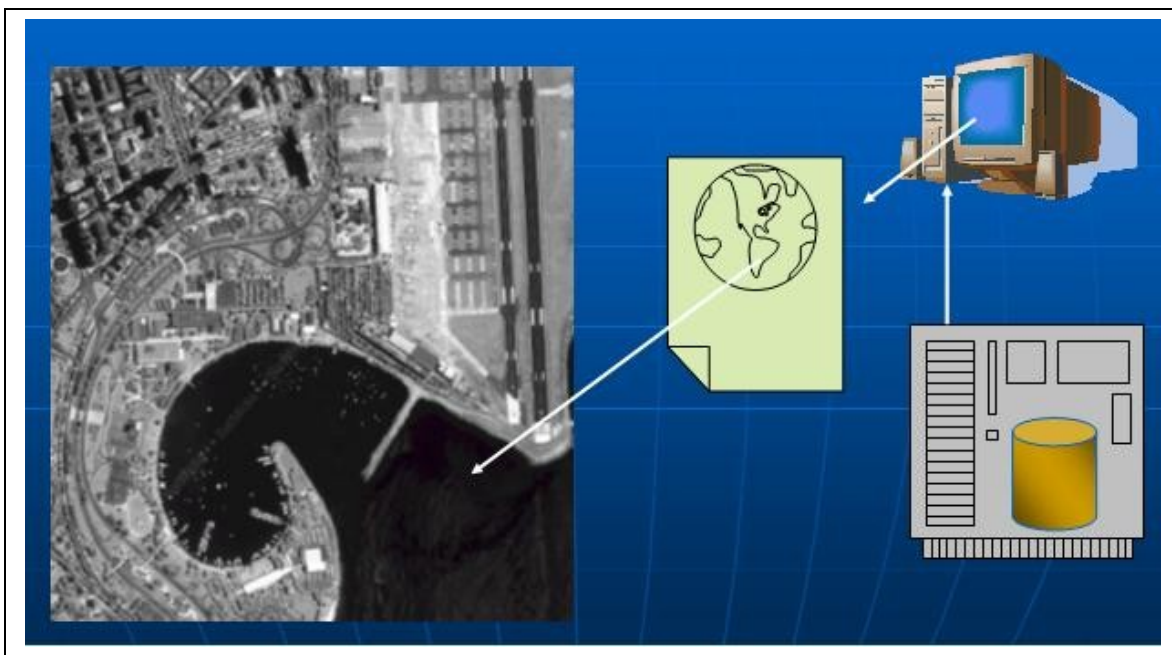


Figura 7. Integração de SIG com banco de dados espaciais.

A coleta de dados georeferenciados permite a análise espacial e temporal e é capaz de fornecer parâmetros para implementação de um programa de gerenciamento de áreas.

No contexto de áreas contaminadas pela radiação, um mapa georeferenciado pode revelar as áreas atingidas, com medidas do nível de radiação, acidentes geográficos existentes, estradas, cidades e, dependendo do nível de detalhamento, ruas e edifícios existentes. Estes dados permitem um estudo e uma análise mais ampla de um acidente radiológico ocorrido e seu impacto dentro de uma área urbana.



Figura 8. Exemplo de utilização de dados espaciais: raster e vetorial.

Em SIG utiliza-se a combinação de dados tabulados com dados espaciais. No processo de tomada de decisão os dados georeferenciados (espaciais) permitem visualizar as relações espaciais entre as diferentes variáveis consideradas. As estruturas do tipo *raster* (imagens de satélite) e vetoriais (ponto, linha e polígono), são tipos de dados usados para representar digitalmente objetos espaciais (*Figura 8*). As estruturas vetoriais identificam objetos espaciais através de pontos, linhas ou polígonos (espaço bi-dimensional). Este tipo de identificação é combinada com a elaboração de um polígono “de fundo”, que complementa a cobertura de todo o plano de informação. Já a estrutura *raster* (matricial) utiliza unidades discretas (células) com as quais representa os objetos espaciais sob forma de conjuntos (agregados de células). Através desta combinação podemos visualizar camadas (*layers*) de dados espaciais e sua vinculação com dados tabulados (rodovias, prédios, população, plantações, ferrovias, portos, aeroportos, consumidores, construções, redes de água, etc.). Desta forma podemos representar o mundo real através de vários aspectos geográficos, que são identificados através de camadas temáticas (*Figura 9*). [23], [24]

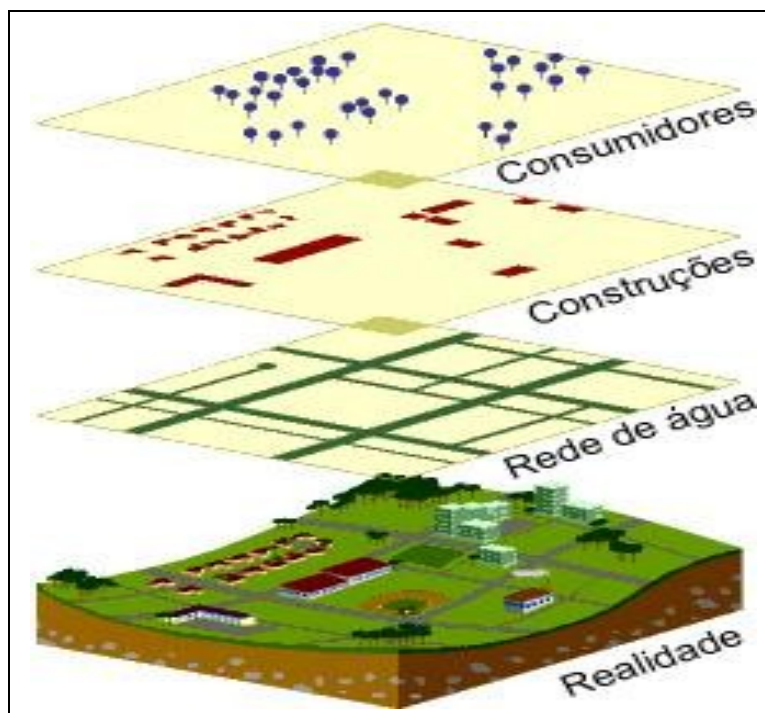


Figura 9. Exemplo de camadas temáticas.

Para o eficiente uso desta tecnologia SIG é preciso definir uma metodologia para representar a área geográfica em estudo e eleger as principais variáveis que se quer analisar. Esta metodologia consiste nas seguintes etapas:

- Eleger a área de estudo (definição de limites);
- Definir as fontes de informação;
- Coletar os dados;
- Analisar os dados coletados;
- Disponibilizar os dados georeferenciados;
- Gerenciar o uso da informação.

3.2 Princípios de Modelagem em Simulação

A simulação é o processo de construção de um modelo de um sistema real e a condução de experiências com esse modelo, com o objetivo de compreender o comportamento do sistema e/ou avaliar estratégias para a sua operação. A simulação pode ser usada para descrever e analisar o comportamento de um sistema, responder questões e contribuir para se projetar o sistema no mundo real. Tanto sistemas reais como conceituais podem ser modelados através de uma simulação.

Existem muitos conceitos que apóiam a simulação. Dentre eles destacam-se: sistema, modelo, variáveis de estado, entidades e seus atributos, atividades, fluxo, retardo e evento-discreto. [18]

Um modelo é a representação de um sistema real, podendo, ser suficientemente complexo para responder questões levantadas e, desta forma, buscar a representação da realidade.

As variáveis de estado são uma coleção de todas as informações necessárias para definir o que está acontecendo no sistema em determinado instante do tempo. A determinação das variáveis de estado se dá em função do propósito da investigação, porque o que vale para alguns casos pode não interessar para outras situações, mesmo que o sistema físico seja o mesmo.

Uma entidade representa um objeto que requer uma definição explícita. Uma entidade pode ser dinâmica, quer dizer, ela se move através do sistema, ou ela pode ser estática e servir a outras entidades. Uma entidade possui atributos que pertencem somente a ela. Da mesma forma que as variáveis de estado, os atributos definidos para uma linha de investigação podem não interessar para outros casos. [18]

Uma atividade desenvolve-se durante um período de tempo cuja duração é conhecida com antecedência ao início da atividade. Daí que a sua finalização pode ser programada quando de seu início.

Pode-se definir os seguintes pontos como vantagens da simulação [18]:

Escolha Correta: a simulação permite testar diversos aspectos de uma mudança proposta sem consumir recursos. Isto é crítico porque uma vez tomada uma decisão, recursos e pessoal começam a ser utilizados, o que gera custos.

Comprimir ou expandir o tempo: a simulação permite comprimir ou expandir o tempo de execução de um processo ou fenômeno, de tal forma a permitir uma análise completa do mesmo. Pode-se examinar um problema em questão de minutos ou dispendar horas para examinar todos os eventos.

Entender o porquê: freqüentemente precisa-se conhecer porque determinado fenômeno ocorre em um sistema real. Com a simulação, determina-se as respostas para os “porquês” através da reconstrução do cenário e realizando-se uma análise microscópica do sistema, pode-se determinar porque o fenômeno ocorre. Em um sistema real não se consegue fazer isto, porque não se pode vê-lo ou controlá-lo completamente.

Diagnose do problema: os fenômenos dinâmicos possuem um grau elevado de complexidade, entender as interações entre as diversas variáveis se constitui em uma vantagem oferecida pela simulação. O diagnóstico dos problemas permite perceber seus efeitos em todo o sistema.

Preparar para as mudanças: sabe-se que o futuro trará mudanças. Responder os “porquês” e “como” com antecedência é altamente útil para reprojeter os sistemas/processos existentes e prever o comportamento frente a diferentes cenários.

Treinar equipes: modelos de simulação permitem excelente treinamento quando projetados para tal propósito. Usado desta maneira, as equipes fornecem soluções e avaliam os seus erros frente a cada situação proposta.

Pode-se elencar como desvantagens da simulação os seguintes pontos [18]:

O Modelo construído requer treinamento especial: aprender com a experiência e com o tempo é uma arte. Se dois modelos do mesmo sistema forem construídos por dois diferentes e competentes técnicos, eles podem ser semelhantes, mas é improvável que sejam iguais.

A Simulação pode ser de difícil interpretação: quando a maioria das saídas da simulação forem variáveis aleatórias, podem ser de difícil análise.

A Modelagem da simulação e análise pode consumir tempo e ser cara: economizar em recursos para modelar e analisar pode resultar em um modelo insuficiente para a tarefa.

Para a construção de um modelo de simulação é necessário um estudo preliminar, que pode ser definido através da metodologia mostrada na *Figura 10*, que possui as seguintes etapas [18]:

1. **Formulação do Problema:** define o problema a ser estudado, incluindo uma declaração por escrito de qual o objetivo da solução do problema.
2. **Definição de objetivos e planejamento do projeto:** permite levantar as perguntas que deverão ser respondidas pela simulação e os cenários que serão investigados. O planejamento do projeto irá indicar o tempo necessário, recursos de *hardware*, *software* e *peopleware* envolvidos, controles e *outputs* a serem gerados.
3. **Modelo Conceitual:** é feita uma abstração do sistema do mundo real que está sendo investigado em um modelo matemático e de lógica relacional.
4. **Coleta de Dados:** dados necessários para a simulação são identificados, especificados e armazenados para dar suporte ao modelo.
5. **Construção do Modelo:** constrói-se o modelo conceitual usando uma ferramenta de simulação.
6. **É Verdadeiro?:** verifica se a execução do modelo esta sendo feito corretamente. É um processo contínuo.

7. **É Válido?:** verifica se o modelo conceitual é uma representação razoavelmente “exata” do sistema do mundo real.
8. **Simulação dos Cenários:** diferentes cenários e suas parametrizações são definidas nesta etapa.
9. **Execução e Análise dos Cenários:** esta etapa é para estimar medidas de performance para os cenários que estão sendo simulados.
10. **Executar Novamente?:** baseado na análise realizada determina se novas execuções são necessárias para simulação dos cenários.
11. **Documentação e relatório:** fornece informações sobre o resultado das execuções realizadas.
12. **Implementação:** com base na simulação realizada determinar ações a serem tomadas para a solução do problema que se investigou.

Nesta dissertação adotou-se esta metodologia para a simulação de acidentes radiológicos baseados em agentes.

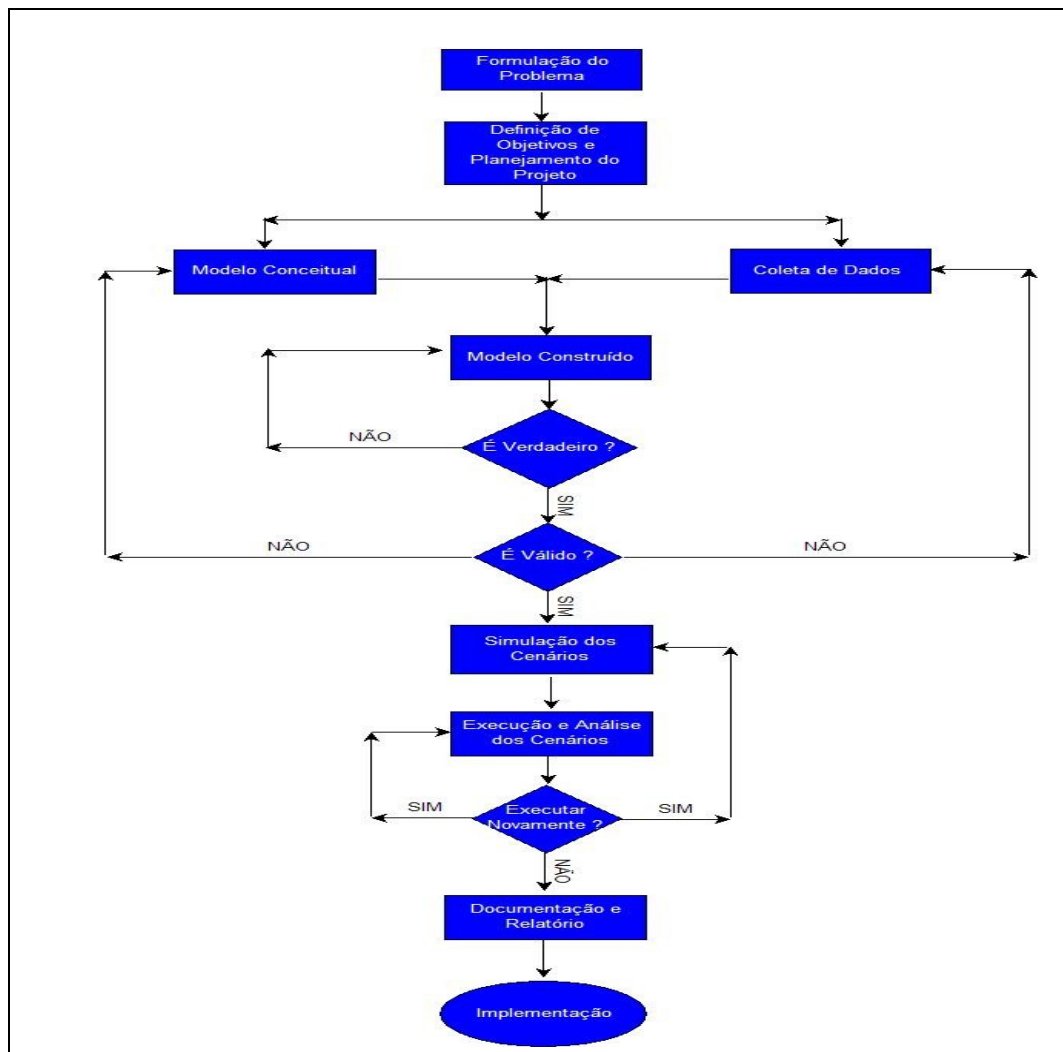


Figura 10. Metodologia para estudo de uma simulação. [18]

3.3 Uso de Ferramentas de Simulação baseada em Agentes Autônomos

Através de uma ferramenta de simulação baseada em agentes é possível definir agentes que irão atuar em um determinado ambiente, através da capacidade de percepção com que tenham sido construídos e instruí-los a adotar determinadas ações.

Para construir simulações multi-agentes existem diversas ferramentas tais como: *Repast*, *Cougaar*, *Zeus*, *Jack*, *Jade*, *Swarm*, *StarLogo* e *NetLogo*.

A escolha de uma ferramenta que possua recursos de visualização dos resultados da simulação é um critério básico, pois os agentes utilizados podem ser melhor monitorados.

Utilizamos nesta dissertação a ferramenta *Repast* (Figura 11), pelas seguintes razões:

Possui uma integração dos agentes com informações georeferenciadas (SIG);

Seu código fonte é em Java, o que torna a ferramenta extremamente portátil;

Ser uma ferramenta do tipo RAD (*rapid application development*);

Permite o monitoramento do agente e do modelo construído;

Possui recursos gráficos para *output* dos resultados;

Possui uma interface gráfica amigável para construção dos agentes;

É uma ferramenta *open source* que está em constante desenvolvimento;

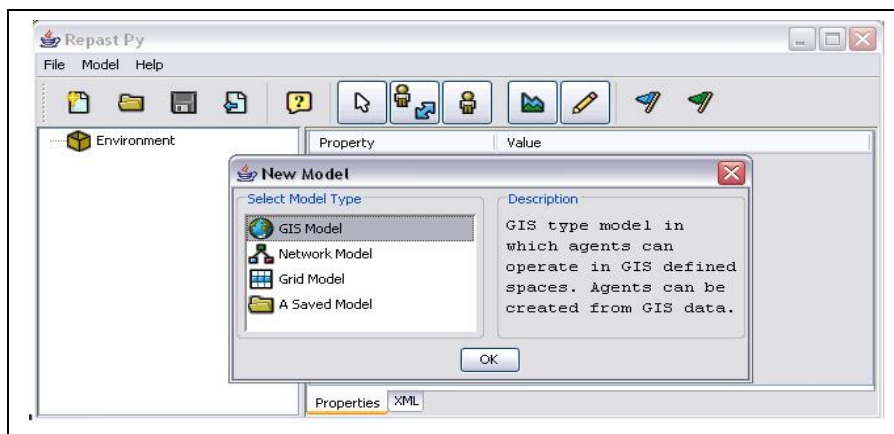


Figura 11. Ferramenta de simulação baseada em agentes e SIG – Repast.

4 Estudo de Caso: acidente radiológico de Cochabamba, Bolívia

Tomamos, nesta dissertação, como estudo de caso o acidente radiológico ocorrido em Cochabamba, Bolívia in 2002. Este acidente foi minuciosamente documentado e fisicamente reconstruído pela *International Agency of Energy Atomic* (IAEA) [17], bem como, a estimativa das doses envolvidas. Desta forma, tem-se um valioso conjunto de dados que pôde ser comparado com os resultados da simulação baseada em agentes e, também, pôde validar o modelo proposto.

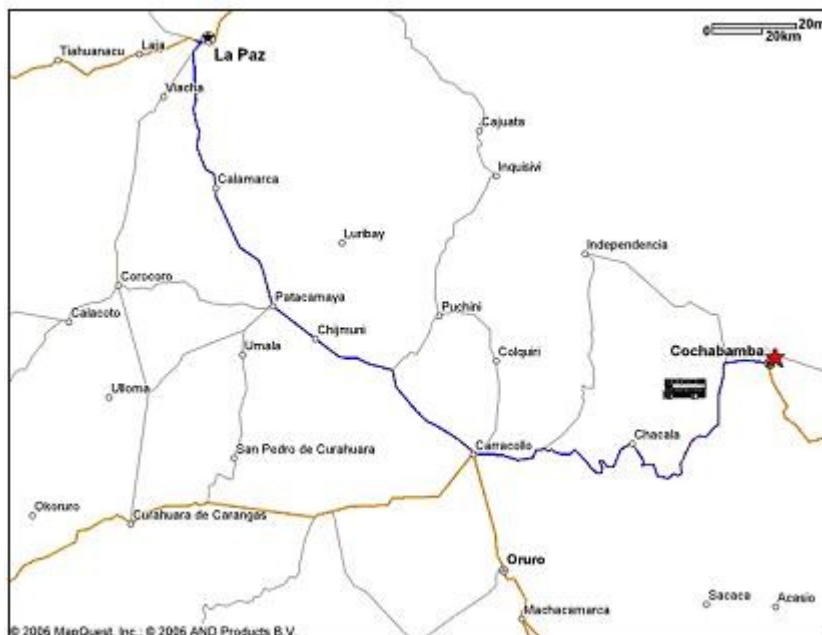


Figura 12. Trajeto do ônibus – Cochabamba à La Paz.

Em abril de 2002 um acidente envolvendo uma fonte de Ir^{192} (meia-vida de 74,5 dias) de um equipamento de gamagrafia industrial ocorreu em Cochabamba, Bolívia, situada a 400 km da capital La Paz. A fonte, transportada em um *container*, ficou exposta sem que os técnicos se advertissem a tempo. O *container* foi transportado dentro do compartimento de bagagem de um ônibus de carreira, após a sua utilização pelos operadores do equipamento em um serviço de inspeção. Este ônibus estava com sua lotação máxima (55 passageiros) e realizou um trajeto, com 8 horas de duração, de Cochabamba até a cidade de La Paz (Figura 12 e 13).

Ao chegar no destino o *container* foi recolhido pelos funcionários da empresa e levado à sua sede num táxi. Somente no final, ao realizar procedimentos de rotina de radioproteção, é que verificaram que a fonte não havia sido corretamente selada e que teria ficado exposta durante todo o trajeto [17].

No nosso estudo de caso, em virtude das características da fonte radioativa, o efeito da radiação somente foi relevante para as pessoas situadas a poucos metros da fonte, ou seja, para as pessoas situadas dentro do ônibus. Na análise das fórmulas de taxa de exposição (equação 2), de dose absorvida (equação 3) e de dose equivalente (equação 4), verifica-se que todas essas grandezas – dada uma fonte específica – são dependentes somente da posição relativa das pessoas expostas e do tempo desta

35–38	0,52	0,29	0,17
39–42	0,17	0,14	0,11
43–46	0,09	0,08	0,07
47–50	0,05	0,05	0,04
51–55	0,03	0,03	0,03

Em nossa simulação baseada em agentes (Figuras 15 e 16), empregamos essencialmente a mesma fórmula (equação 2) para estimação das doses. Na análise comparativa entre nossa simulação através de agentes, usando o *Repast*, e as doses estimadas pelo IBTEN encontramos uma diferença média de 4 % (Tabela 2). Esta diferença ocorreu, tendo em vista, que em nossa simulação não se levou em conta os efeitos de blindagem e que, também se considerou a altura do dorso do passageiro para fins de cálculo da distância da fonte radioativa. Esta reduzida diferença nos permitiu verificar que, em ambas as estimativas, as doses máximas se concentraram nos passageiros que estavam sentados nos assentos 27-30 (doses de 2,77 Gy e 3,14 Gy) e 31-34 (doses de 2,29 Gy e 2,0792 Gy) e que as doses mínimas encontradas, também nas duas estimativas, ocorreram nos assentos 1-4 (doses de 0,02 Gy e 0,0196 Gy) (Figura 17 e 18). Estes dados evidenciam que a nossa estimativa feita através da simulação usando agentes (*Repast*) conseguiu detectar as posições de máxima e mínima contaminação ocorrida dentro do ônibus de passageiros, bem como o valor das doses absorvidas.

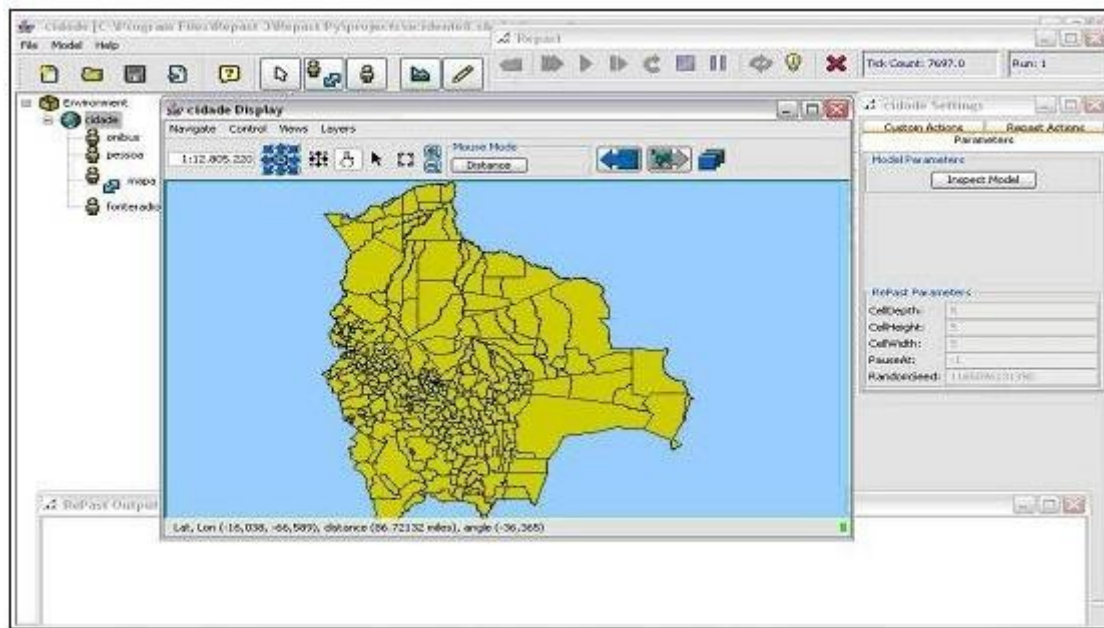


Figura 15. Simulação usando Repast – imagem vetorial da Cochabamba-La Paz.

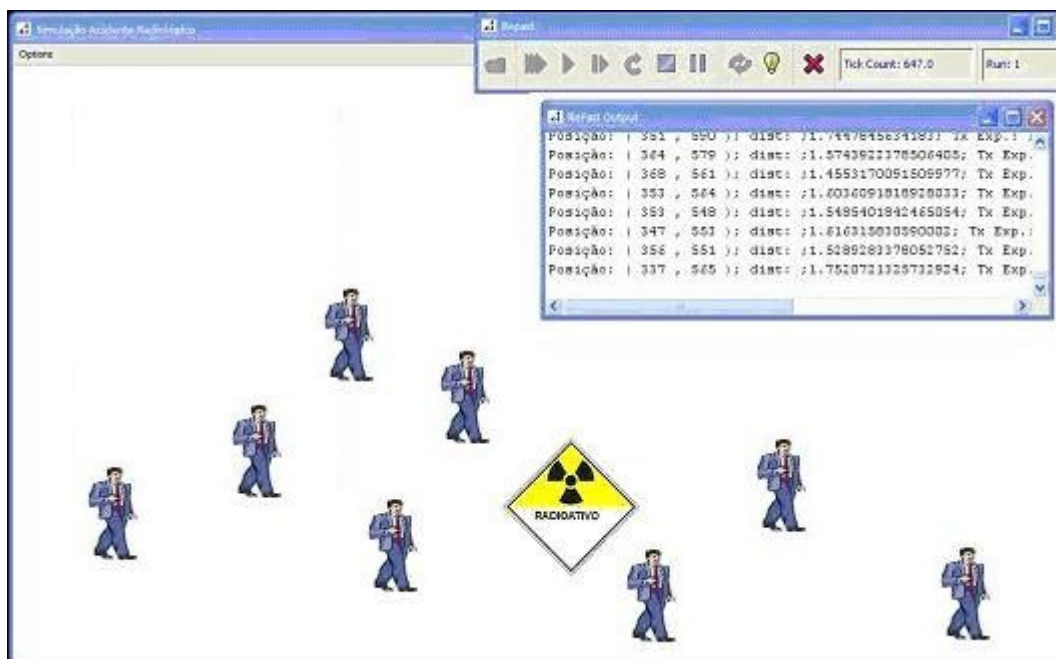


Figura 16. Simulação usando Repast – passageiros e fonte radioativa.

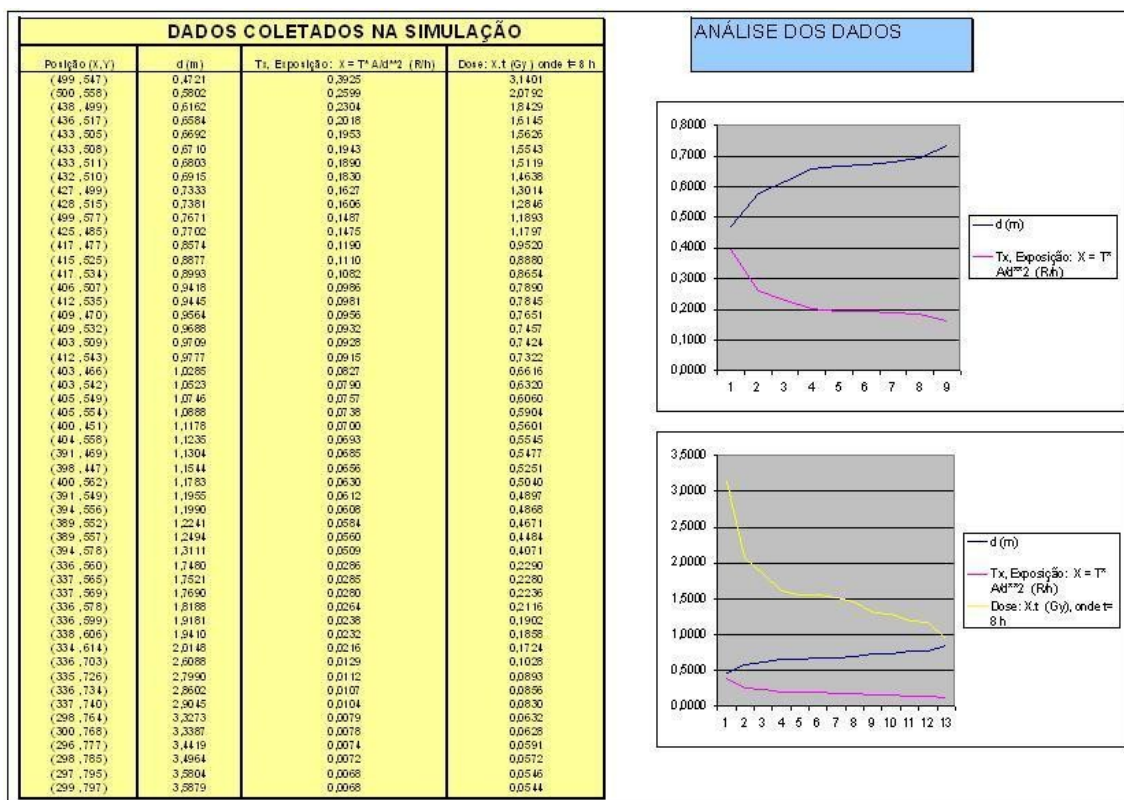


Figura 17. Simulação usando Repast – análise dos dados coletados

Tabela 2. Estimativas de doses (Gy) para os passageiros realizadas pela simulação utilizando agentes

Assento do Ônibus	Distância (m)	Distância (m)	Dose (Gy)	Dose (Gy)
	(IBTEN)	(REPAST)	(IBTEN)	(REPAST)
1-4	5,915	5,9709	0,02	0,0196
5-6	4,830	4,8021	0,03	0,0303
7-10	4,183	3,9509	0,04	0,0448
11-14	3,162	3,3273	0,07	0,0632
15-18	2,236	2,0148	0,14	0,1724
19-22	1,357	1,3111	0,38	0,4071
23-26	0,653	0,6584	1,64	1,6145
27-30	0,503	0,4721	2,77	3,1401
31-34	0,553	0,5802	2,29	2,0792
35-38	1,160	1,1544	0,52	0,5251
39-42	2,029	2,0148	0,17	0,1724
43-46	2,789	2,7990	0,09	0,0893
47-50	3,741	3,7485	0,05	0,0498
51-55	4,830	4,8021	0,03	0,0303

IBTEN – Instituto Boliviano de Tecnologia e Energia Nuclear

REPAST – Ferramenta de simulação baseada em agentes

Outra conclusão que se pode tirar, é que sendo a dose recebida anualmente de radiação natural em torno de 1,12 mSv ($\approx 1,12$ mGy) e que, neste caso, de uma fonte de Ir^{192} , foram necessários apenas 4 segundos de exposição para ultrapassar o limite desta dose anual . Como os passageiros ficaram expostos a 8 horas de radiação emitida pela fonte do Ir^{192} , eles receberam uma dose de no mínimo 17,5 vezes maior que o limite anual . É preciso destacar, também, que não se levou, na simulação realizada, em consideração efeitos de blindagem, o que produziu uma elevação nas doses estimadas. [12]

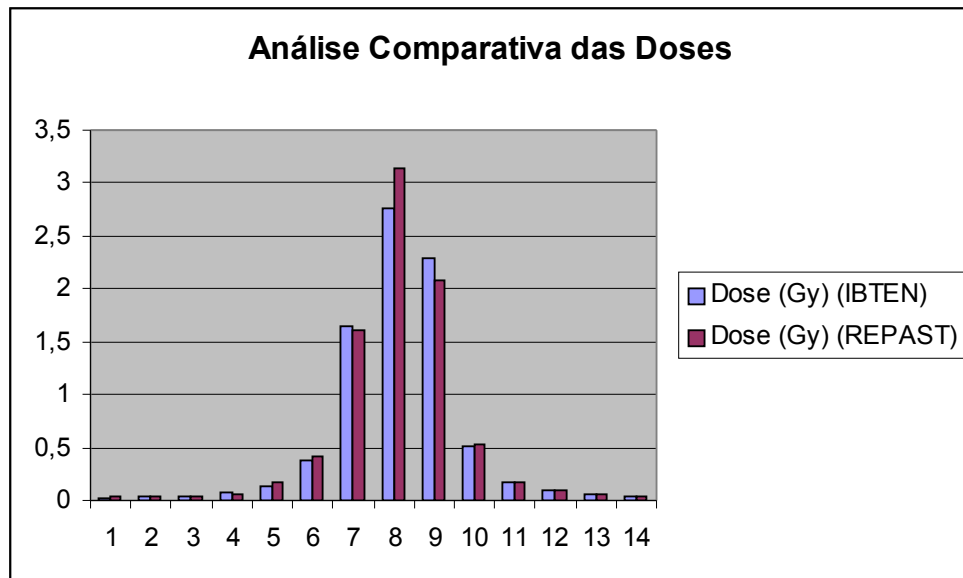


Figura 18. Simulação usando Repast – análise comparativa das doses (Gy)

5 Conclusões

As simulações através de modelos computacionais permitem, com maior facilidade, a predição do comportamento de sistemas reais sob condições as mais variadas, o que fornece uma compreensão mais aprofundada da estrutura destes sistemas. Pôde-se através deste estudo, perceber diversas vantagens de se trabalhar com modelos e simulações, dentre elas, não trabalhar diretamente com o mundo real, o que poderia promover custos elevados, procedimentos onerosos e procedimentos muito difíceis de serem realizados (no caso em estudo, envolvimento de pessoas e fontes radioativas reais). Outra vantagem, observada é que as simulações podem ser repetidas *ad nauseam* e são não destrutivas, não interferindo no mundo real.

Este exemplo simples nos deu uma visão das amplas possibilidades da simulação de acidentes radiológicos usando Sistemas Baseados em Agentes. Pode-se, por exemplo, simular as conseqüências para população, de uma fonte radioativa que possa ser deixada no metrô, estimar as doses absorvidas, número de pessoas envolvidas, conseqüências físicas, distribuição espacial das pessoas contaminadas, as possibilidades de contaminação, etc. Com isto é possível se ter uma primeira idéia da magnitude de um acidente radiológico, seu impacto no sistema de saúde, estatísticas de número de pessoas contaminadas, o tipo de dano que elas sofreriam, etc.

Sistemas Baseados em Agentes possibilitam a construção de cenários de acidentes radiológicos que permitem avaliar as conseqüências da contaminação acidental. Com esta ferramenta de simulação pode-se, também, garantir maior precisão na implementação de medidas de segurança no uso da radioatividade, e, conseqüentemente, aumentar a proteção da população em geral.

5.1 Trabalhos Futuros

Como trabalhos futuros destacam-se os seguintes pontos:

- Ampliar o alcance da simulação para áreas urbanas;
- Criar um simulador específico para acidentes radiológicos;
- Permitir uma simulação que envolva cálculo de dose absorvida internamente.
- Ampliar o alcance da simulação para contaminação do meio ambiente, ampliando as camadas temáticas de SIG;
- Ampliar a simulação para acidentes em instalações nucleares;
- Criar parceria com a IAEA para desenvolvimento de SIG e simulações baseadas em agentes.

Bibliografia

- [1] International Atomic Energy Agency, IAEA, The Radiological Accident in Goiânia, Viena, Austria, 1988.
- [2] Wooldridge, M. J. An Introduction to MultiAgent Systems. John Wiley & Sons, Ltd, April, 2002.
- [3]Wooldridge M. J., and N. R. Jennings, “Agents theories, architectures, and languages: a survey.” In Proceedings of ECAI’94 Workshop on Agent Theories, Architectures and Languages, 1995, pp.1-22.
- [4]J. Ingham, “What is an agent?” Technical Report #6/99, Centre for Software Maintenance, University of Durhan, Durhan, London.
- [5]S. Franklin, and A. Graesser, “Is it an agent, or just a program?: a taxonomy for autonomous agents.” In Inteligents Agents III: Agent Theories, Architectures, and Languages, edited by J. P. Muller, M. J. Wooldridge, and N. R. Jennings, Berlin: Springer, 1997, pp.21-35.
- [6]O. L. M. Farias and N. Santos. Agent-Based Geographical Information System. In: International Conference on Intelligent Agents, Web Technologies and Internet Commerce - IAWTIC'2005, 2005, Vienna. Proceedings of the International Conference on Intelligent Agents, Web Technologies and Internet Commerce - IAWTIC'2005. Los Alamitos, California : Edited by M. Mohammadian, 2005. v. 1. p. 998-1005.
- [7]O. L. M. Farias, and L. T. Leite. Simulation and Control of Maritime Area through Multi-Agent Systems and Geographical Information Systems . In: International Conference on Intelligent Agents, Web Technologies & Internet Commerce - IAWTIC 2005, 2005, Vienna. Proceedings of the International Conference on Intelligent Agents, Web Technologies and Internet Commerce - IAWTIC'2005. Los Alamitos, California : Edited by M. Mohammadian, 2005. v. 1. p. 734-738.
- [8]Tauhata, Luiz, Salati, Ivan, Di Prinzo Rato, Di Prinzo Antonieta, Radioproteção e Dosimetria: Fundamentos, Instituto de Radioproteção e Dosimetria – IRD/ CNEN, 2002.
- [9]United Nations, “Report of the United Nations Scientific Committee on the Effects of Atomic Radiation to the General Assembly”, 2000.
- [10]http://www.kose.ee/nucbasic/nucpedia/uk/alpha_radiat.htm, acesso em 27 de Julho de 2007.
- [11]Silva, Tadeu Augusto de Almeida ; Farias, O. L. M. ; Santos, Neide dos . “Simulating Radiological Accidents through Software Agents.” In: International Conference on Intelligent Agents, Web Technologies & Internet Commerce - IAWTIC 2006, 2006, Sydney, Australia.

[12]Silva, Tadeu Augusto de Almeida ; Farias, O. L. M. ; “Employing Software Multi-Agents for Simulating Radiological Accidents”, In: 9th International Conference on Enterprise Information Systems - ICEIS 2007, 2007, Funchal, Madeira - Portugal.

[13]http://www.rdc.gov.lv/nucpedia/uk/beta_radiat.htm, acesso em 27 de Julho de 2007.

[14]International Atomic Energy Agency, IAEA, Radiation Safety, Viena, Austria, 1996.

[15] A.M. Agapov, R.V. Arutyunyan, A.Yu.Kudrin, A.M. Eliseev “Experience of past radiation accidents and problems of response to possible dispersion of radioactive material in urban conditions”, , International Conference on the safety and security of radioactive sources towards a global system for the continuous control of sources throughout their life cycle, 27 June-July 2005.

[16]http://www.ndt-ed.org/EducationResources/CommunityCollege/RadiationSafety/quant_units/units.htm, acesso em 27 de Julho de 2007.

[17]International Atomic Energy Agency, IAEA, The Radiological Accident in Cochabamba, Viena, Austria, 2004.

[18]Banks Jerry , “Handbook of Simulation, Principles, Methodology, Advances, Applications and Practice”, Emp Books, 1998.

[19]Alvares, L.O.; Sichman, J. Introdução aos Sistemas Multiagentes. In: Jornada de Atualização em Informática, 16.; CONGRESSO DA SBC, 17., 1997, Brasília. Anais... Brasília: SBC, 1997. p.1-38.

[20]Silva, Cláudio Antônio, Modelagem Comportamental para Agentes Autônomos em Ambientes Reais, Tese de Mestrado em Engenharia de Computação, UERJ, 2003.

[21]Wooldrige, Michael, Agent Theories, Architectures and Language: A Survey, 1999.

[22]F.C. A. da Silva, J.G. Hunt, A. T. Ramalho, VR Crispim, “Dose Reconstruction of a Brazilian Industrial Gamma Radiography Partial-Body Overexposure Case” , Journal of Radiological Protection , 2003.

[23]Burrough, P.A .; McDonnel, R. A “Principles of Geographical Information Systems”, Oxford University Press, 1998.

[24]Maguire, D.; Goodchild M. F. ; Rhind D. W. “Geographical Information Systems: Principles and Applications”, vol. 2:Applications. Longman Scientific & Technical, 1991.

Anexo I – Código Fonte em Java com Agentes Utilizados na Simulação

//Modelo da Simulação de Acidente Radiológico no Ônibus //

```
import java.awt.Color;
import java.awt.Dimension;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import java.util.Iterator;
import uchicago.src.sim.analysis.OpenSequenceGraph;
import uchicago.src.sim.analysis.Sequence;
import uchicago.src.sim.engine.AutoStepable;
import uchicago.src.sim.engine.SimpleModel;
import uchicago.src.sim.gui.DefaultGraphLayout;
import uchicago.src.sim.gui.DisplaySurface;
import uchicago.src.sim.gui.Network2DDisplay;
import uchicago.src.sim.util.Random;
import uchicago.src.sim.util.RepastException;
import uchicago.src.sim.util.SimUtilities;

public class ModeloSimulacao extends SimpleModel {
    protected Local local;
    private int NumPassageiros = 1;
    private DisplaySurface localDisplaySurface;
    private OpenSequenceGraph localErrorGraph;
    private OpenSequenceGraph individualGraph;
    protected int localWidth = 1000, localHeight = 1000;

    public ModeloSimulacao() {
        // Definir geração randômica de agentes
        Random.createUniform();
        if (super.agentList == null)
            super.agentList = new ArrayList();
    }

    public String[] getInitParam() {
        return new String[] {
            "NumPassageiros",
            "posiçãoX",
            "posiçãoy"
        };
    }

    public void begin() {
        try {
            super.begin();
        }
    }
}
```



```

        // Zerar número de agentes
        Passageiro.resetIndices();

        if (super.agentList == null)
            super.agentList = new ArrayList();
        super.agentList.clear();
        local = new Local(localWidth, localHeight);

        // Utilizar o número definido de agentes
        local.hirepassageiro(NumPassageiros);
        local.hireOrdenador();
        local.hireOrientador();

        // Listar os agentes
        super.agentList.addAll(local.getpassageiro());
        super.agentList.addAll(local.getOrdenador());
        super.agentList.add(local.getOrientador());

        this.buildDisplay();
        this.buildGraphs();

    } catch (RepastException ex) {
        SimUtilities.showError("Erro ao ler o modelo", ex);
        super.stop();
    }
}

/**
 * Exibir o display do ambiente
 */
protected void buildDisplay() {
    localDisplaySurface =
        new DisplaySurface(
            new Dimension(localWidth, localHeight),
            this,
            "Simulação Acidente Radiológico");

    // Criar gráfico
    DefaultGraphLayout layout = new DefaultGraphLayout(localWidth,
        localHeight);
    layout.getNodeList().addAll(local.getpassageiro());
    layout.getNodeList().add(new Objetivo(local));
    Network2DDisplay localNetDisplay = new Network2DDisplay(layout);
    localDisplaySurface.addDisplayableProbeable(localNetDisplay,
        "Estimativas");

    localDisplaySurface.setBackground(Color.WHITE);
    localDisplaySurface.display();
    this.registerDisplaySurface("Simulação", localDisplaySurface);
}

```

```

/**
 * Criar gráfico Movimento
 */
protected void buildGraphs() {
    localErrorGraph = new OpenSequenceGraph("Monitora Movimento",
this);

    localErrorGraph.addSequence("", new Sequence() {
        public double getSValue() {
            double totalErr = 0;
            for (Iterator iter = local.getpassageiro().iterator(); iter
                .hasNext();) {
                totalErr += ((Passageiro) iter.next()).getError();
            }

            return totalErr / local.getpassageiro().size();
        }
    });

    localErrorGraph.setYRange(-.1, 1.1);
    localErrorGraph.setXRange(0, 5);
    localErrorGraph.display();

    /**** Criar gráfico de estimativa de movimento ***/
    individualGraph = new OpenSequenceGraph("Estimativa de
Movimento", this);

    for (Iterator iter = local.getpassageiro().iterator(); iter.hasNext(); ) {
        final Passageiro pass = (Passageiro) iter.next();
        individualGraph.addSequence(
            pass.getNodeLabel() ,
            new Sequence() {
                public double getSValue() {
                    return pass.getError();
                }
            },
            pass.getColor());
    }

    individualGraph.setYRange(-.1, 1.1);
    individualGraph.setXRange(0, 5);
    individualGraph.display();
}

/**
 * Zerar o modelo para nova execução
 */

```

```

public void setup() {
    super.setup();

    if (localErrorGraph != null)
        localErrorGraph.dispose();
    if (individualGraph != null)
        individualGraph.dispose();
    if (localDisplaySurface != null)
        localDisplaySurface.dispose();

    localDisplaySurface = null;

    getModelManipulator().addButton("Dispersão dos Agentes", new
    ActionListener() {
        public void actionPerformed(ActionEvent e) {
            Iterator iter = ((ArrayList)
            local.getpassageiro().clone()).iterator();
            for (; iter.hasNext(); ) {
                Passageiro pass = (Passageiro) iter.next();

                pass.setX(Random.uniform.nextIntFromTo(0,
            local.getWidth()));
                pass.setY(Random.uniform.nextIntFromTo(0,
            local.getHeight()));
            }

            localDisplaySurface.updateDisplayDirect();
        }
    });
}

protected void preStep() {
    try {
        for (Iterator iter = super.agentList.iterator(); iter.hasNext(); ) {
            ((AutoStepable) iter.next()).preStep();
        }
    } catch (Exception ex) {
        SimUtilities.showError("Erro na predefinição da simulação", ex);
        super.stop();
    }
}

protected void step() {
    try {
        for (Iterator iter = super.agentList.iterator(); iter.hasNext(); ) {
            Object o = iter.next();
            ((AutoStepable) o).step();
        }
    }
}

```

```

        } catch (Exception ex) {
            SimUtilities.showError("Erro na predefinição da simulação", ex);
            super.stop();
        }
    }

    protected void postStep() {
        try {
            for (Iterator iter = super.agentList.iterator(); iter.hasNext();) {
                ((AutoStepable) iter.next()).postStep();
            }
        } catch (Exception ex) {
            SimUtilities.showError("Erro na predefinição da simulação", ex);
            super.stop();
        }
        localDisplaySurface.updateDisplay();
        localErrorGraph.step();
        individualGraph.step();
    }

    public String getName() {
        return "Simulação Acidente";
    }

    /**
     * @return numero de passageiros.
     */
    public int getNumPassageiros() {
        return NumPassageiros;
    }

    public void setNumPassageiros(int NumPassageiros) {
        this.NumPassageiros = NumPassageiros;
    }

    public int getlocalHeight() {
        return localHeight;
    }

    public void setlocalHeight(int localHeight) {
        this.localHeight = localHeight;
    }

    public int getlocalWidth() {
        return localWidth;
    }

    public void setlocalWidth(int localWidth) {
        this.localWidth = localWidth;
    }

```

```

    }

    public static void main(String[] args) {
        uchicago.src.sim.engine.SimInit init = new
uchicago.src.sim.engine.SimInit();
        ModeloSimulacao model = new ModeloSimulacao();
        if (args.length > 0)
            init.loadModel(model, args[0], false);
        else
            init.loadModel(model, null, false);
    }
}

```

//Definição do Agente Passageiro de ônibus

```

import java.awt.Color;
import java.awt.Image;
import java.awt.geom.Rectangle2D;
import javax.swing.ImageIcon;
import org.joone.engine.DirectSynapse;
import org.joone.engine.FullSynapse;
import org.joone.engine.LinearLayer;
import org.joone.engine.Monitor;
import org.joone.engine.Pattern;
import org.joone.engine.SigmoidLayer;
import org.joone.io.MemoryInputSynapse;
import uchicago.src.sim.adaptation.neural.NeuralException;
import uchicago.src.sim.adaptation.neural.NeuralUtils;
import uchicago.src.sim.adaptation.neural.RepastNeuralWrapper;
import uchicago.src.sim.engine.AutoStepable;
import uchicago.src.sim.gui.OvalNetworkItem;
import uchicago.src.sim.gui.SimGraphics;
import uchicago.src.sim.network.DefaultDrawableNode;
import uchicago.src.sim.util.RepastException;
import uchicago.src.sim.util.SimUtilities;

public class Passageiro extends DefaultDrawableNode implements AutoStepable {
    public static final int DO_NOTHING = 0;

    public static final int DO_SOMETHING = 1;

    public static final int Ordenador_A = 0;

    public static final int Ordenador_B = 1;

    private static Image passageiroPicture;

    private static final Color[] colors = new Color[] {

```

```

        Color.PINK,
        Color.BLUE,
        Color.GRAY,
        Color.GREEN,
        Color.MAGENTA,
        Color.YELLOW,
        Color.WHITE,
        Color.CYAN
    };
    private static int colorIndex = 0;

    private static int baseIdNumber;

    protected RepastNeuralWrapper net;

    protected MemoryInputSynapse desiredNetworkOutput;
    protected MemoryInputSynapse inputForTraining;

    protected DirectSynapse inputForRetrieval;

    private int actionPerformedInStep = DO_NOTHING;

    private double retrievedValue = 0.0;

    private boolean wasScolded = false;

    private double[] OrdenadorCommands =
        new double[] { DO_NOTHING, DO_NOTHING };

    private double[] prevStepCommands =
        new double[] { DO_NOTHING, DO_NOTHING };

    private double angst = 1;

    private double error = 0.0;

    //método de carga da figura e da construção da rede //
    public Passageiro(double x, double y) throws RepastException {
        //super() é uma chamada direta ao construtor da superclasse //
        //No caso abaixo estou chamando o construtor da classe OvalNetworkItem() //
        super(new OvalNetworkItem(x, y));

        loadpassageiroPicture();

        buildNeuralNetwork();

        this.setColor(getNextColor());
        this.setHeight(5);
        this.setWidth(2);
    }

```

```

        //this.setNodeLabel("Passageiro " + ++baseIdNumber);
        this.setNodeLabel("");
    }

    public Passageiro() throws RepastException {
        this(0, 0);
    }

    private static void loadpassageiroPicture() {
        if (passageiroPicture == null) {
            java.net.URL passageiroPicURL =
Passageiro.class.getResource("passageiro.gif");
            passageiroPicture = new
ImageIcon(passageiroPicURL).getImage();
        }
    }

    private static Color getNextColor() {
        if (colorIndex == colors.length)
            colorIndex = 0;

        return colors[colorIndex++];
    }

    public static void resetIndices() {
        baseIdNumber = 0;
    }

    private void buildNeuralNetwork() throws RepastException {
        //Definindo os elementos da rede 2 inputs, 3 camadas e 1 output //
        //Faz a conexões de sinapse com os neurônios da rede //
        this.net = NeuralUtils.buildNetwork(new int[] {2, 3, 1},
                                            new Class[] {

LinearLayer.class,

SigmoidLayer.class,

SigmoidLayer.class

                                            },
                                            new Class[] {

FullSynapse.class,

FullSynapse.class});

        this.net.getNet().getMonitor().setLearningRate(.8);
        this.net.getNet().getMonitor().setMomentum(0.3);

```

```

        this.inputForTraining = new MemoryInputSynapse();
        /* As primeiras 2 colunas contêm os valores de input */
        this.inputForTraining.setAdvancedColumnSelector("1,2");

        this.desiredNetworkOutput = new MemoryInputSynapse();
        this.desiredNetworkOutput.setAdvancedColumnSelector("1");

        this.inputForRetrieval = new DirectSynapse();
        this.inputForRetrieval.setName("RetrievingInput
MemoryInputSynapse");
        this.net.getNet().getTeacher().setDesired(desiredNetworkOutput);
    }

    private synchronized void train() throws NeuralException {
        int actionShouldveBeen;

        if (wasScolded) {
            if (actionPerformedInStep == DO_NOTHING)
                actionShouldveBeen = DO_SOMETHING;
            else
                actionShouldveBeen = DO_NOTHING;
        } else {
            actionShouldveBeen = this.actionPerformedInStep;
        }
        this.net.getNet().getMonitor().setLearningRate(.8);
        this.net.getNet().getMonitor().setMomentum(0.3);

        this.error = Math.abs(actionShouldveBeen - retrievedValue);

        Monitor monitor = net.getNet().getMonitor();

        monitor.setTrainingPatterns(1);
        /* Quantas vezes a rede precisa ser treinada */
        monitor.setTotCicles(1);

        this.desiredNetworkOutput
            .setInputArray(new double[][]
{ { actionShouldveBeen } });
        synchronized(OrdenadorCommands) {
            this.inputForTraining
                .setInputArray(
                    new double[][] { (double[])
this.prevStepCommands.clone() });
        }

        try {
            this.net.train(inputForTraining);
        } catch (NeuralException ex) {
            SimUtilities.showError("Error training neural network for
agent \""

```



```

        + getNodeLabel() + "\"", ex);
    throw ex;
}
}

private synchronized double retrieve() throws NeuralException {
    synchronized(OrdenadorCommands) {
        inputForRetrieval.fwdPut(new Pattern((double[])
(OrdenadorCommands.clone())));
    }
    Pattern retrievedPattern = net.retrieve(inputForRetrieval);

    return retrievedPattern.getValues()[0];
}

public void preStep() throws NeuralException {
    train();

    this.wasScolded = false;
}

public void step() throws NeuralException {
    try {

        synchronized(OrdenadorCommands) {
            prevStepCommands = (double[])
OrdenadorCommands.clone();
        }

        this.retrievedValue = retrieve();

        if (Math.round(retrievedValue) == DO_NOTHING) {
            doNothing();
        } else {
            doSomething();
        }

    } catch (NeuralException ex) {
        SimUtilities.showError(
            "Error computing the next action to perform. \n"
            + "Agent \"" + getNodeLabel() +
            "\".", ex);
        throw ex;
    }
}

public void postStep() {
}

```

```

public void draw(SimGraphics g) {
    g.drawImageScaled(passageiroPicture);

    int width = passageirosPicture.getWidth(null);

    g.setFont(super.getFont());

    Rectangle2D bounds = g.getStringBounds(this.getNodeLabel());

    //Aqui desenho o nome "passageiro" e acompanha imagem do
passageiro //
    g.setDrawingCoordinates((float) (this.getX() + width / 2.0 -
bounds.getWidth() / 2.0),
                                (float) (this.getY() -
bounds.getHeight() - 2),
                                0f);

    g.drawString(getNodeLabel(), Color.BLUE);
}

public void scold(Ordenador Ordenador) {
    this.angst += .1;
    this.wasScolded = true;
}

public void praise(Ordenador Ordenador) {
    this.angst -= .05;
}

public void receiveCommand(int OrdenadorID, int command) {
    synchronized(OrdenadorCommands) {
        OrdenadorCommands[OrdenadorID] = command;
    }
}

protected void doNothing() {
    actionPerformedInStep = DO_NOTHING;
}

protected void doSomething() {
    actionPerformedInStep = DO_SOMETHING;
}

public double getActionPerformed() {
    return actionPerformedInStep;
}

public double[] getCommands() {
    synchronized(OrdenadorCommands) {

```

```

        return OrdenadorCommands;
    }
}

public double getError() {
    return error;
}
}

```

//Agente que orienta e corrige o Agente Passageiro
//Calcula a dose e taxa de exposição

```

import java.util.Iterator;
import uchicago.src.sim.engine.AutoStepable;
import uchicago.src.sim.util.Random;

public class Orientador extends Ordenador implements AutoStepable {
    private static final int CLOSER = 0;
    private static final int FARTHER = 1;
    private int andando = 0;
    public Orientador(Local local) {
        super(local);
    }

    public void preStep() {
    }

    public void step() {

    }

    public void postStep() {
        for (Iterator iter = local.getpassageiro().iterator(); iter.hasNext();) {
            Passageiro pass = (Passageiro) iter.next();

            double actionPerformed = pass.getActionPerformed();
            double[] commands = pass.getCommands();

            int[] simplifiedCommands = new int[] {
                (int) Math.round(commands[0]),
                (int) Math.round(commands[1]) };

            int correctAction = simplifiedCommands[0] ^
simplifiedCommands[1];

            if (correctAction == Math.round(actionPerformed)) {
                pass.praise(this);
                movepass(pass, CLOSER);
            }
        }
    }
}

```

```

    }

    else {
        pass.scold(this);
        movepass(pass, FARTHER);
    }
}

}

private void movepass(Passageiro pass, int direction) {
    int xSide, ySide;
    double dist, D, X;

    if (pass.getX() < (double) local.getWidth() / 2) {
        xSide = 2;
    } else if (pass.getX() == (double) local.getWidth() / 2) {
        xSide = 1;
    } else {
        xSide = 0;
    }

    if (pass.getY() < (double) local.getHeight() / 2) {
        ySide = 0;
    } else if (pass.getY() == (double) local.getHeight() / 2) {
        ySide = 1;
    } else {
        ySide = 2;
    }

    double xMove = Random.uniform.nextDoubleFromTo(0, 20);
    double yMove = Random.uniform.nextDoubleFromTo(0, 20);

    int mod = (direction == CLOSER) ? 1 : -1;

    double newX = pass.getX(), newY = pass.getY();

    if (xSide == 0) {
        if (ySide == 0) {
            newX = (pass.getX() - xMove * mod);
            newY = (pass.getY() + yMove * mod);
        } else {
            newX = (pass.getX() - xMove * mod);
            newY = (pass.getY() - yMove * mod);
        }
    } else if (xSide == 2) {
        if (ySide == 0) {
            newX = (pass.getX() + xMove * mod);
            newY = (pass.getY() + yMove * mod);
        } else {

```

```

        newX = (pass.getX() + xMove * mod);
        newY = (pass.getY() - yMove * mod);
    }
} else {
    if (ySide == 0)
        newY = (pass.getY() + yMove * mod);
    else if (ySide == 2)
        newY = (pass.getY() - yMove * mod);
    else {
        if (direction == CLOSER) {
        } else {
            newX = (pass.getX() - xMove * mod);
            newY = (pass.getY() - yMove * mod);
        }
    }
}

if (newX < -(pass.getWidth() / 2.0))
    newX = -(pass.getWidth() / 2.0);
else if (newX > local.getWidth() + pass.getWidth() / 2.0)
    newX = local.getWidth() + pass.getWidth() / 2.0;

if (newY < -(pass.getHeight() / 2.0))
    newY = -(pass.getHeight() / 2.0);
else if (newY > local.getHeight() - pass.getHeight() / 2.0)
    newY = local.getHeight() - pass.getHeight() / 2.0;

//Posiciona o passageiro
pass.setX(newX);
pass.setY(newY);

//distância da pessoa à fonte radioativa
dist = Math.pow((500 - newX),2) + Math.pow(( 500 - newY),2);
dist = Math.sqrt(dist);
dist = dist /100;

//Parâmetros da Fonte Radioativa de 'Ir 192'
//Atividade: A = 0,18 Ci
//T= 0,486 (R.m2) / (h.Ci)

//Taxa de Exposição  $X = T * A/d^{**2}$ 
X = (0.486 * 0.18) / (Math.pow(dist,2)) ;

//Dose Absorvida  $D = X * t$ , sendo t = 8 h ( tempo da viagem de ônibus )
D = X * 8;

System.out.print("Posição: ( "+Math.round(newX)+" ,
"+Math.round(newY)+" ),"+" dist: ;"+dist+"; Tx Exp.: ;"+X+"; Dose: ;"+ D +";\n");

```

```

    }

}

```

//Ambiente de atuação do Agente Passageiro

```

import java.util.ArrayList;
import uchicago.src.sim.util.Random;
import uchicago.src.sim.util.RepastException;

public class Local {
    private ArrayList passageiro = new ArrayList();
    private ArrayList Ordenador = new ArrayList();
    private Orientador orientador;
    private int width;
    private int height;
    public Local() {
        super();
    }

    public Local(int width, int height) {
        super();

        this.width = width;
        this.height = height;
    }

    public int fireAllpassageiro() {
        int passageiroFired = passageiro.size();

        passageiro.clear();

        return passageiroFired;
    }

    public void hirepassageiro(int numpassageiro) throws RepastException {
        for (int i = 0; i < numpassageiro; i++) {
            double x = Random.uniform.nextDoubleFromTo(0, width);
            double y = Random.uniform.nextDoubleFromTo(0, height);
            passageiro.add(new Passageiro(x, y));
        }
    }

    public void hirepassageiro(Passageiro pass) {
        passageiro.add(pass);
    }

    public void hireOrdenador() throws RepastException {
        if (Ordenador.size() == 2)
            return;
    }
}

```

```

        Ordenador OrdenadorA = new Ordenador(this, (double) width / 2,
(double) height / 2
            + height / 20);
        OrdenadorA.setOrdenadorID(Passageiro.Ordenador_A);

        Ordenador OrdenadorB = new Ordenador(this, (double) width / 2,
(double) height / 2
            - height / 20);
        OrdenadorB.setOrdenadorID(Passageiro.Ordenador_B);

        Ordenador.add(OrdenadorA);
        Ordenador.add(OrdenadorB);
    }

    public void hireOrientador() throws RepastException {
        if (orientador != null)
            return;

        orientador = new Orientador(this);
    }

    public int getHeight() {
        return height;
    }

    public void setHeight(int height) {
        this.height = height;
    }

    public int getWidth() {
        return width;
    }

    public void setWidth(int width) {
        this.width = width;
    }

    public ArrayList getpassageiro() {
        return passageiro;
    }
    public ArrayList getOrdenador() {
        return Ordenador;
    }
    public Orientador getOrientador() {
        return orientador;
    }
}

```

// Classe que define o Objetivo do Agente

```

import java.awt.Image;
import javax.swing.ImageIcon;

```

```

import uchicago.src.sim.gui.SimGraphics;
import uchicago.src.sim.network.DefaultDrawableNode;

public class Objetivo extends DefaultDrawableNode {
    private static Image ObjetivoPicture;
    public Objetivo(Local local) {
        super();
        loadObjetivoPicture();
        this.setX(local.getWidth() / 2.0 - 34);
        this.setY(local.getHeight() / 2.0 - 15);
    }

    private static void loadObjetivoPicture() {
        if (ObjetivoPicture == null) {
            java.net.URL ObjetivoURL =
Objetivo.class.getResource("fonderadioativa.gif");

            ObjetivoPicture = new ImageIcon(ObjetivoURL).getImage();
        }
    }

    public void draw(SimGraphics g) {
        g.drawImage(ObjetivoPicture);
    }
}

```

//Agente que ordena o movimento dos passageiros

```

import java.awt.Color;
import java.util.Iterator;
import uchicago.src.sim.engine.AutoStepable;
import uchicago.src.sim.gui.RectNetworkItem;
import uchicago.src.sim.network.DefaultDrawableNode;
import uchicago.src.sim.util.Random;

public class Ordenador extends DefaultDrawableNode implements AutoStepable {
    protected Local local;
    private int OrdenadorID;
    public Ordenador(Local local, double x, double y) {
        super(new RectNetworkItem(x, y));

        super.setColor(Color.RED);
        this.local = local;
    }

    public Ordenador(Local local) {
        this(local, 0, 0);
    }
}

```



```

    public void preStep() {
        for (Iterator iter = local.getpassageiro().iterator(); iter.hasNext();) {
            Passageiro pass = (Passageiro) iter.next();

            pass.receiveCommand(OrdenadorID, getNextCommand());
        }
    }

    public void step() {
    }
    public void postStep() {
    }
    private int getNextCommand() {
        return getRandomCommand();
    }
    private int getRandomCommand() {
        double randCommand = Random.uniform.nextDoubleFromTo(0, 1);
        if (randCommand >= .5)
            return Passageiro.DO_NOTHING;
        else
            return Passageiro.DO_SOMETHING;
    }
    public int getOrdenadorID() {
        return OrdenadorID;
    }
    public void setOrdenadorID(int OrdenadorID) {
        this.OrdenadorID = OrdenadorID;
    }
}

```

// **Modelo SIG**

```

import java.util.Iterator;
import anl.repast.gis.data.OpenMapData;
import anl.repast.gis.display.OpenMapDisplay;
import uchicago.src.sim.engine.DefaultGroup;
import uchicago.src.sim.engine.SimInit;
import uchicago.src.simbuilder.base.gis.GISSimModelImpl;

```

```

public class gismodel extends GISSimModelImpl
{
    private DefaultGroup boliviamapaGroup;
    private static Class class$gisambiente$boliviamapa;
    public gismodel() {
        params = new String[0];
        name = "GIS Model";
    }

    public DefaultGroup getBoliviamapaGroup() {
        return boliviamapaGroup;
    }
}

```

```

public void initAgents() {
}

public void updateDisplay() {
    this.updateGISDisplay();
}

public void writeAgents() {
}

public void _setup() {
    boliviamapaGroup
        = new DefaultGroup(((class$gisambiente$boliviamapa != null
                               ? class$gisambiente$boliviamapa
                               : (class$gisambiente$boliviamapa
                                  = _class$("boliviamapa"))),
                             "step");
}

static Class _class$(String string) {
    Class var_class;
    try {
        var_class = Class.forName(string);
    } catch (ClassNotFoundException classnotfoundexception) {
        throw new NoClassDefFoundError(classnotfoundexception
                                         .getMessage());
    }
    return var_class;
}

public void _build() {
    OpenMapData openmapdata = OpenMapData.getInstance();
    boliviamapaGroup.addAll
        (openmapdata.createAgents
         ((class$gisambiente$boliviamapa != null
            ? class$gisambiente$boliviamapa
            : (class$gisambiente$boliviamapa
               = _class$("gisambiente.boliviamapa"))),
          "C:/Program Files/Repast 3/Repast
Py/projects/limite_municipal_Bolivia/limites_municipales.shp"));
    Iterator iterator = boliviamapaGroup.iterator();
    while (iterator.hasNext()) {
        boliviamapa var_boliviamapa = (boliviamapa) iterator.next();
        var_boliviamapa.setModel(this);
    }
    initAgents();
}

public void _display() {

```

```

omDisplayUpdater.reset();
esriDisplayUpdater.reset();
updater = omDisplayUpdater;
omDisplay = new OpenMapDisplay("Modelo SIG");
this.registerMediaProducer("Modelo SIG", omDisplay);
omDisplay.setModel(this);
omDisplay.addLayer(boliviamapaGroup, "boliviamapaGroup");
omDisplayUpdater.addLayer("boliviamapaGroup", boliviamapaGroup);
omDisplay.centerMap
    (OpenMapData.getInstance().getCenter
    ("C:/Program Files/Repast 3/Repast
Py/projects/limite_municipal_Bolivia/limites_municipales.shp"));
}

public void _schedule1() {
    Iterator iterator = boliviamapaGroup.iterator();
    while (iterator.hasNext()) {
        boliviamapa var_boliviamapa = (boliviamapa) iterator.next();
        var_boliviamapa.step();
    }
}

public void _schedule() {
    schedule.scheduleActionBeginning((double) 1, this, "_schedule1");
}

public static void main(String[] strings) {
    SimInit siminit = new SimInit();
    gismodel var_gismodel = new gismodel();
    siminit.loadModel(var_gismodel, "", false);
}
}

```

// Mapa da Bolívia

```

import java.awt.Color;
import com.bbn.openmap.omGraphics.OMGraphic;
import uchicago.src.simbuilder.base.gis.AbstractOpenMapGISAgent;

public class boliviamapa extends AbstractOpenMapGISAgent
{
    private gismodel model;
    private String[] propNames = new String[0];
    private String CODSECCION;
    private int COUNT;
    private OMGraphic Geometry;
    private int MUNIDE;

    public boliviamapa(gismodel var_gismodel) {
        this();
        model = var_gismodel;
    }
}

```

```

public boliviamapa() {
}
public String[] getProbedProperties() {
    return propNames;
}
public void step() {
}
public void setCODSECCION(String string) {
    CODSECCION = string;
}
public String getCODSECCION() {
    return CODSECCION;
}
public void setCOUNT(int i) {
    COUNT = i;
}
public int getCOUNT() {
    return COUNT;
}
public void setGeometry(OMGraphic omgraphic) {
    Geometry = omgraphic;
}
public OMGraphic getGeometry() {
    return Geometry;
}
public void setMUNIDE(int i) {
    MUNIDE = i;
}
public int getMUNIDE() {
    return MUNIDE;
}
public void initDrawingMap() {
    defaultColor = new Color(-65536);
    drawingMap.clear();
}
public Color setupDraw() {
    return defaultColor;
}
public void setModel(gismodel var_gismodel) {
    model = var_gismodel;
}
}

```
