MARCELO SANTOS DAIBERT

MONITORAMENTO DE RISCOS EM PROJETOS DE SOFTWARE: UMA ABORDAGEM BASEADA EM DINÂMICA DE SISTEMAS E TÉCNICAS DE INTELIGÊNCIA COMPUTACIONAL

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae.*

VIÇOSA
MINAS GERAIS – BRASIL
2010

MARCELO SANTOS DAIBERT

MONITORAMENTO DE RISCOS EM PROJETOS DE SOFTWARE: UMA ABORDAGEM BASEADA EM DINÂMICA DE SISTEMAS E TÉCNICAS DE INTELIGÊNCIA COMPUTACIONAL

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae.*

APROVADA: 18 de março de 2010.

_____	_____
Prof. Alcione de Paiva Oliveira (Co-Orientador)	Prof. Mauro Nacif Rocha (Co-Orientador)
_____	_____
Prof. Marco Antônio Pereira Araújo	Prof. Luiz Aurélio Raggi

_____

Prof. José Luis Braga
(Orientador)

Dedico essa dissertação aos meus pais
Helene e Ricardo

Ao meu filho
Yago

À minha amada noiva
Jenifer

Aos colegas, amigos, professores e alunos que
participaram desta caminhada

AGRADECIMENTOS

Inicialmente agradeço a Deus por ter me colocado neste caminho e por ter me dado forças para realização deste trabalho. Me conduziu pelos caminhos que Ele considerava os melhores e que me proporcionaram um maior aprendizado. Me fez passar por situações que além de me prepararem para a realização deste trabalho, também me prepararam para a grande jornada da vida.

Agradeço aos meus pais, Ricardo e Helene, que plantaram uma semente do conhecimento quando eu mal sabia andar. Eles me fizeram perceber a importância dos estudos e é por esta razão que estou executando este trabalho. Sempre buscaram me proporcionar o que houve de melhor e não mediram sacrifícios para que isso ocorresse; sem vocês, nada disso seria possível. Agradeço também a todos os familiares e amigos da família por todo o incentivo e orgulho.

Ao meu filho Yago, que mesmo de longe me estimulou a lutar cada vez mais, para que eu pudesse vencer as batalhas diárias dos estudos e principalmente do trabalho. Me deu forças em momentos de desânimo e abatimento.

A minha amada noiva Jenifer, pelo carinho, dedicação, ajuda e compreensão. Por todos os momentos compartilhados e por todo amor dedicado a mim. Todo cuidado e atenção que me faz viver em constante estado de “apaixonamento”.

Ao meu orientador, o Professor José Luis Braga, por acreditar em mim. Me conduziu ao melhor caminho deste trabalho e com conselhos partilhou sua experiência de vida e me fez aprender muito mais do que “simples” teorias e temas “exotéricos” sobre *Engenharia de Software* (como ele mesmo definia). Me fez aprender um pouco mais da vida e de como ter serenidade diante dos problemas. Sou outro ser humano depois que lhe conheci. Obrigado por tudo e desculpe todo e qualquer problema que eu causei.

Ao Professor Antônio de Pádua Braga e sua equipe de alunos por ter me recebido com toda atenção no departamento de engenharia eletrônica da UFMG (Universidade Federal de Minas Gerais) em seu laboratório de inteligência computacional. Nesta ocasião o Professor Braga e seus alunos assistiram a um

seminário sobre este trabalho e puderam colaborar com o projeto. Obrigado por todas as valiosas opiniões e dicas sobre o andamento do projeto.

Agradeço aos meus co-orientadores, os Professores Alcione de Paiva Oliveira e Mauro Nacif Rocha, pelo incentivo e pelo exemplo de como ser bons professores.

Ao meu orientador de graduação, Professor Marco Antônio Pereira Araújo, que me fez compreender e mais do que isso, entender, que a vida é feita de desafios, e estes foram feitos para serem vencidos. Conseguiu deixar muito claro para mim que somente com muito trabalho e dedicação conseguimos chegar aos nossos objetivos. Você é um exemplo de profissional para mim. Sem seus conselhos não estaria aqui e não seria o que sou hoje. Obrigado por todos os emails, telefonemas, conversas e principalmente obrigado por discordar de mim.

A todos os professores do corpo docente do Departamento de Informática da Universidade Federal de Viçosa, pela formação, conhecimentos recebidos, pelo incentivo e pela dedicação à atividade docente. Em especial aos professores que tive contato direto durante boa parte do período de realização deste trabalho: Jugurta, Carlos e Vladimir. Agradeço especialmente pelos desafios que me foram apresentados nas suas disciplinas. Estes desafios me motivam até hoje.

A todos os meus ex-professores de graduação. Todos foram exemplos e inspiradores para meus estudos.

Ao Altino Alves de Souza Filho, secretário de pós-graduação do departamento de informática, pelas conversas e por sempre me auxiliar nas burocracias que envolviam os procedimentos durante todo o curso de mestrado. A todos os demais funcionários do Departamento de Informática, pela dedicação e amor no trabalho.

Aos amigos André Castro, Alexandre Fraga e Juliana Campos pelo companheirismo e amizade durante o período de realização deste trabalho. A todos os demais amigos e colegas, do mestrado e graduação, que me incentivaram, ajudaram e contribuíram direta ou indiretamente para a realização deste trabalho.

Aos colegas e amigos da Faculdade Governador Ozanam Coelho (FAGOC) que durante a realização deste trabalho sempre me motivaram. Em especial ao amigo Clayton Fraga Filho, que além de ter sido o coordenador do Curso de Computação da

FAGOC, foi um grande amigo para todas as horas. Obrigado pela grande parceria nos anos que tivemos a oportunidade de estarmos juntos.

A todos os meus alunos. Vocês são fundamentais para que eu continue na luta diária. Obrigado por todo carinho e especialmente o respeito demonstrado. Contem comigo hoje e sempre.

Aos colegas e amigos da Optical Soluções em Informática LTDA (OPTICALHOST) pela atenção e principalmente pela compreensão durante estes anos. Em especial aos amigos Alexandre Aleixo e Gustavo Hipólito que contribuíram muito para que eu entrasse no mestrado e pudesse concluir este trabalho.

BIOGRAFIA

MARCELO SANTOS DAIBERT, filho de Ricardo Daibert Pinto e Helene Maria Santos de Mendonça Daibert, brasileiro nascido em 28 de setembro de 1983 na cidade de Juiz de Fora, no estado de Minas Gerais. Pai de Yago Souza Daibert.

Concluiu o ensino médio no Colégio Cristo Redentor (Academia de Comércio) em 2002 na cidade de Juiz de Fora, Minas Gerais. Neste mesmo ano e na mesma cidade ingressou no curso de graduação em Sistemas de Informação pela Faculdade Metodista Granbery (FMG), concluído no ano de 2005.

Em 2006 iniciou o curso de pós-graduação *lato sensu* em Ciência da Computação na Universidade Federal de Viçosa pelo Departamento de Informática, sendo este concluído em 2008.

Em 2007, foi aprovado na seleção do programa de pós-graduação *stricto sensu* do Departamento de Informática, onde em março de 2007, iniciou o mestrado em Ciência da Computação na Universidade Federal de Viçosa, defendendo sua dissertação em março de 2010.

Desde quando obteve o grau de Bacharel em Sistemas de Informação vêm atuando como Gerente Técnico da empresa Optical Soluções em Informática LTDA (OPTICALHOST). Em 2007 foi contratado pela Faculdade Governador Ozanam Coelho (FAGOC), em Ubá – Minas Gerais, para compor do quadro de docentes da instituição. Em Dezembro de 2009 foi convidado a atuar como coordenador do curso de computação da FAGOC, função que exerce atualmente e em paralelo com os serviços prestados à OPTICALHOST.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTAS DE CÓDIGOS FONTE

RESUMO

DAIBERT, Marcelo Santos, M. Sc., Universidade Federal de Viçosa, março de 2010. **Monitoramento de riscos em projetos de software: uma abordagem baseada em dinâmica de sistemas e técnicas de inteligência computacional**. Orientador: José Luis Braga. Co-Orientadores: Alcione de Paiva Oliveira e Mauro Nacif da Rocha.

Com o passar dos anos foi possível observar uma profunda evolução na utilização da informática e computação. Em consequência disso, a utilização de aplicativos foi se tornando cada vez mais comum e necessário. Os processos de desenvolvimento de software foram amadurecendo em consequência das necessidades do mercado e, com isso, a qualidade de software passou a ser uma busca constante em todo o processo de desenvolvimento de software. A utilização da dinâmica de sistemas no contexto da engenharia de software busca contribuir para a simulação de ambientes de desenvolvimento de software, servindo como um modelo de apoio à decisão para os gestores, possibilitando assim uma melhoria de qualidade em todo o processo de desenvolvimento de software. Neste sentido, este trabalho propõe uma abordagem de monitoramento de riscos em projetos de software utilizando simulações de dinâmica de sistemas e algumas técnicas de inteligência computacional (redes neurais artificiais e regras de produção). Além do *framework* de monitoramento de riscos, ou seja, os conceitos e as bases do funcionamento do sistema, este trabalho especificou e implementou uma ferramenta de apoio a decisão que monitora os riscos inerentes ao projeto de software que foram configurados pelos gestores. Esta ferramenta é alimentada pelas simulações dos modelos de dinâmica de sistemas e alerta os gestores para a eminência de materialização de algum risco. Desta forma, os gestores podem tomar medidas para prevenir, transferir, mitigar ou aceitar a materialização dos riscos negativos (que podem causar prejuízos ao projeto), ou então provocar, compartilhar, melhorar ou aceitar os riscos positivos (que podem causar benefícios no projeto).

ABSTRACT

DAIBERT, Marcelo Santos, M. Sc., Universidade Federal de Viçosa, March, 2010. **Monitoring risk in software projects: an approach based on system dynamics and computational intelligence techniques**. Adviser: José Luis Braga. Co-Advisers: Alcione de Paiva Oliveira and Mauro Nacif da Rocha.

Over the years it was possible to observe a profound evolution in the use of informatics and computing. As a result, the use of software was becoming increasingly common and is now a necessity. During this phenomenon, the software processes development were maturing as a result of market needs today and with this, Software Quality has become a constant search throughout the software development process. In this context, System Dynamics can contribute to the simulation of software development environments, serving as a model of decision support for managers. In this sense, this work proposes an approach to monitoring risks in software projects using simulations of dynamic systems and some computational intelligence techniques (neural networks and rule based system). Beyond the monitoring risk's framework, this work specified and implemented a decision support tool that monitors the risks inherent in software project that were set by software managers. This tool is powered by the simulations of system dynamic's model and alert the software managers to the brink of occurs some risk. Thus, managers can take measures to prevent, transfer, mitigate or accept the materialization of negative risks (which may cause damage to the project), or cause, share, improve or take positive risks (which may cause benefits to the project).

1 INTRODUÇÃO

O cenário de desenvolvimento de software com mais de dez anos era limitado quando comparado com os tempos atuais. Antigamente, existiam várias limitações, como poder de processamento das máquinas, linguagens de programação, a falta de um ambiente integrado e visual para o desenvolvimento, entre outros. O escopo em que se enquadrava o software era muito bem definido. Sabia-se onde o software iria iniciar a sua atuação e onde iria terminar. De fato, o problema que o software iria resolver era específico e linear na maioria dos casos. Com o escopo bem definido era possível então construir um sistema de forma rápida e eficiente, guardadas as devidas proporções (PFLEEGER, 2004).

Com a evolução tecnológica encadeada com a modificação do pensamento humano, foi possível observar que nos últimos anos (especialmente nos últimos dez anos), os sistemas de informação foram cada vez mais inseridos no cotidiano das pessoas com o objetivo de substituir o que pudera ser automatizado. Com isso, a complexidade destes sistemas foram aumentando a níveis nunca planejados. Da mesma forma, a dificuldade de desenvolvimento também foi incrementada. O foco então muda da dificuldade tecnológica antes identificada, para a dificuldade de compreensão dos requisitos a serem implementados em um escopo ora não muito bem definido, ora bem definido, mas muito grande e de difícil compreensão. É fato que hoje são disponibilizadas máquinas com alto poder de processamento, linguagens maduras com um intuitivo ambiente visual integrado para o desenvolvimento, entre outros. Ou seja, a tecnologia existe, mas hoje, a dificuldade está na identificação e definição do problema e seu contexto (minimundo).

A utilização de sistemas computacionais deixou de ser um luxo para ser uma necessidade mercadológica, com profundo impacto estratégico. A computação tornou possível o acesso a vários dados e informações nas mais diversas áreas, onde somente com o esforço do homem seria impossível atingir estes resultados. O software tornou-se então um elemento estratégico para a diferenciação de produtos e serviços (PRESSMAN, 2006), além de necessário para realização de várias atividades nas mais diversas áreas de conhecimento. Cada uma dessas áreas apresenta diferentes tipos de perfis de problemas, sendo que cada perfil apresenta características específicas inerentes ao seu ambiente.

Neste contexto, a Engenharia de Software se apresenta como a alternativa que busca tratar as questões relativas ao desenvolvimento de sistemas, levando em consideração todos os aspectos relacionados ou correlacionados da área, como a análise e levantamento de requisitos, gerenciamento destes requisitos, projeto e gerência do desenvolvimento, execução e codificação, testes, fornecimento, manutenção, entre outros.

Os esforços que buscaram disciplinar o processo de software como um todo, surgiram em meados dos anos 90, com algumas propostas de modelos de processo de software. A primeira proposta que teve destaque era baseada em um modelo sequencial linear, que apresentava uma abordagem linear para o desenvolvimento de software com os seguintes processos: análise e definição dos seus requisitos, projeto de sistema e de software, implementação, integração e testes e operação e manutenção. Um exemplo conhecido deste modelo é o modelo em cascata ou *waterfall model* (PFLEEGER, 2004).

Evoluções foram ocorrendo e os processos de software também caminhavam nestas evoluções, principalmente pela mudança das necessidades de mercado (SOMMERVILLE, 2003). Com isso, surgiram vários outros processos e modelos de desenvolvimento de software (DEMARCO; PLAUGER, 1979), como, por exemplo, a abordagem de desenvolvimento evolucionário, incremental e por fim, o modelo em espiral (BOEHM, 1988). É importante destacar as abordagens ágeis, que são baseados em ciclos curtos de desenvolvimento (AGILE MANIFESTO, 2008) (AGILE ALLIANCE, 2008) e são mais adequados para problemas cujo contexto muda constantemente. A comunicação tem uma grande importância e é mais informal. Há um maior envolvimento do cliente na verificação de requisitos, definição de prioridades, planejamento e validação. SCRUM (SCRUM, 2008) e XP (*Extreme Programming*) (XP, 2008) são alguns dos exemplos para métodos ágeis, mesmo que a atuação hoje do XP esteja bem limitada, mas sua importância na evolução dos métodos ágeis deve ser destacada.

O desenvolvimento de software está longe de ser uma tarefa trivial. São várias as influências que esta atividade sofre, e um modelo de desenvolvimento de software perfeito ainda não é uma realidade. Por isso, o ideal é estabelecer um processo de desenvolvimento de software para cada empresa, de forma personalizada, analisando o ambiente particular de cada uma, seus funcionários, o tipo de software construído,

entre outros. Assim, contribuindo para um melhor desenvolvimento, e consequentemente em um produto final de melhor qualidade.

A dinâmica de sistemas, base fundamental para este trabalho, foi criada por Jay Forrester em 1958, mas teve seu primeiro livro publicado sobre o assunto em 1961 (FORRESTER, 1961). É um método de análise de ambiente onde são observadas as diversas variáveis que o afetam, de forma a permitir um melhor entendimento do ambiente como um todo, proporcionando uma nova ótica para tomada de decisão. É uma ferramenta para análise da interdependência das variáveis de um ambiente e como estas variáveis se relacionam, de forma a analisar o impacto destas variáveis sobre elas mesmas ou sobre o ambiente.

1.1 O Problema, Sua Importância e Justificativa

Devido à dificuldade apresentada no contexto atual de desenvolvimento de software, muito se discute sobre a baixa qualidade e produtividade no desenvolvimento de sistemas, que causa insatisfação aos usuários e prejuízos financeiros. Inclusive este foi o principal motivador para a criação de uma disciplina como a Engenharia de Software para atuar nesta área (PFLEEGER, 2004).

A Engenharia de Software estuda e estabelece, principalmente, os processos de desenvolvimento de software e o gerenciamento deles, com o objetivo de melhoria de qualidade do produto final. Um dos esforços significativos neste sentido corresponde à definição de métodos voltados à disciplinar o processo de desenvolvimento de software através de práticas sistemáticas de forma a proporcionar um controle para a tarefa de desenvolvimento. Contudo, os processos de software buscam disciplinar as atividades pertinentes ao desenvolvimento e não os aspectos dinâmicos que esta tarefa agrega (PFLEEGER, 2004). Ou seja, definem os processos, os passos para o desenvolvimento, mas não provêm métodos para identificação e resolução de problemas que possam ocorrer durante esta fase.

Variáveis como tamanho de equipe, rotatividade de pessoal (*turnover* de pessoal), competência dos profissionais, volatilidade dos requisitos, existência de requisitos não especificados (BOEHM; TURNER, 2003), problemas pessoais, carga

horária, entre outros, sequer são analisados pelos métodos e processos da Engenharia de Software existentes. Todavia estas variáveis influenciam diretamente no resultado final da tarefa de desenvolvimento de software. Não gerenciar estas variáveis no processo de desenvolvimento contribui para o fracasso do projeto, uma vez que estas variáveis se apresentam como riscos inerentes à atividade (BOEHM, 1991) (CHARETTE, 1990) (FAIRLEY, 1994) (HIGUERAG, 1994).

Modelos de maturidade de software, como o CMMI (*Capability Maturity Model Integration*) (SEI, 2008) e MPS.BR (Melhoria de Processo do Software Brasileiro) (SOFTEX, 2008) descrevem a necessidade de se gerenciar riscos em projetos de software, desde sua identificação, até análise e ações a serem tomadas. E é neste sentido que este trabalho se insere, propondo uma abordagem de se utilizar as técnicas de dinâmica de sistemas, juntamente com técnicas de inteligência computacional para monitoramento de riscos.

1.2 Hipótese

A utilização de uma ferramenta de suporte à decisão, com base em modelos e simulações de dinâmica de sistemas, possibilita o monitoramento de riscos inerentes a projetos de desenvolvimento de software.

1.3 Objetivos

1.3.1 Objetivo Geral

Definir um *framework* conceitual para monitoramento de riscos no processo de desenvolvimento de software com base em técnicas de modelagem e simulação de dinâmica de sistemas.

1.3.2 Objetivos Específicos

- Apresentar um levantamento bibliográfico sobre Riscos em Projetos de Software, Dinâmica de Sistemas, Qualidade de Software e Técnicas de Inteligência Computacional;
- desenvolver um levantamento sobre trabalhos que utilizam as técnicas da dinâmica de sistemas na área de Engenharia de Software;
- especificar um sistema de informação executiva (EIS – *Executive Information System*) de forma a monitorar os riscos envolvidos nas atividades de projetos de software. Este sistema deve permitir ao gerente (gestor) de software configurar / personalizar o risco de acordo com o modelo de simulação de dinâmica de sistemas;
- implementar um protótipo do sistema de informação executiva proposto;
- avaliar a aplicabilidade da dinâmica de sistemas apoiando tarefas de nível gerencial em projetos de software com dois estudos de caso.

1.4 Método de Desenvolvimento do Trabalho

Este trabalho teve seu início em 2006, quando o assunto Engenharia de Software e Dinâmica de Sistemas foi tema de uma monografia de pós-graduação (*lato sensu*) apresentada pelo mesmo discente autor desta dissertação à UFV (Universidade Federal de Viçosa). Na oportunidade, o trabalho intitulado "*Dinâmica de Sistemas: Uma Abordagem Voltada à Engenharia de Software*" (DAIBERT, 2008) conferiu o título de especialista em Ciência da Computação ao autor e serviu como base bibliográfica e motivação para esta dissertação.

Almejando alcançar os objetivos propostos, o desenvolvimento deste trabalho passou pelos seguintes passos:

1. **Pesquisa bibliográfica e levantamento bibliográfico:** Nesta fase, iniciada em 2006 na monografia de especialização, foram obtidos os conhecimentos e fundamentos necessários sobre como aplicar as técnicas de dinâmica de sistemas. Foi alvo investigatório a utilização da dinâmica de sistemas em tarefas da Engenharia de Software. Ainda nesta fase foi feito um estudo sobre Riscos, especificamente riscos em projetos de software. As estratégias de monitoramento de riscos foram estudadas, apesar da pouca bibliografia sobre o tema de monitoramento de riscos em projetos de software com a dinâmica de sistemas. Conceitos sobre qualidade de software, padrões e modelos de maturidade foram avaliados para suporte conceitual ao trabalho. Ao final, algumas técnicas de inteligência computacional foram avaliadas para suportar processos de identificação de riscos na abordagem proposta neste trabalho.

2. **Conhecer trabalhos correlatos:** Inicialmente foi realizada a leitura da dissertação de Ambrósio (2008), intitulada "*Modelagem da Fase de Requisitos em Processos de Desenvolvimento de Software: Uma Abordagem Utilizando Dinâmica de Sistemas*". Este trabalho está inserido no contexto do grupo de pesquisa de Engenharia de Software / Qualidade de Software do Prof. José Luis Braga, orientador do trabalho de Ambrósio (2008). O prof. José Luiz Braga atua no DPI (Departamento de Informática) da UFV (Universidade Federal de Viçosa). Nele, Ambrósio apresenta como sugestão de trabalho futuro a construção de um *framework* de monitoramento de riscos utilizando redes neurais artificiais. Neste sentido, este trabalho busca estabelecer uma continuidade ao trabalho de Ambrósio. Outros trabalhos foram analisados e os principais estão descritos e apresentados no Capítulo 3 desta dissertação.

3. **Definir um modelo de rede neural artificial para suportar a identificação de riscos com base aos resultados das simulações de modelos de dinâmica de sistemas:** para esta atividade foram desenvolvidas as seguintes subatividades:

3.1. Identificar qual topologia de rede neural mais adequada para a tarefa de identificação e classificação de riscos de acordo com o modelo de dinâmica de sistemas;

3.2. Identificar o conjunto de resultados que seriam os neurônios de entrada da rede neural artificial;

3.3. Identificar o conjunto de resultados que seriam os neurônios de saída da rede neural artificial;

3.4. Estabelecer a estratégia de identificação dos riscos na rede projetada.

Com base nos resultados desta fase, é possível identificar que é viável utilizar uma rede neural artificial para classificar e monitorar riscos utilizando os resultados de modelos simulados pelas técnicas de dinâmica de sistemas. Entretanto, mesmo a abordagem se apresentando como satisfatória, foi desenvolvida outra abordagem baseada em regras de produção e com resultado igual e menor esforço. Este resultado foi alcançado graças à observação dos resultados desta fase e do modelo matemático apresentado pela rede neural artificial. A esta nova abordagem foi dado o nome de Linguagem de Inferência Booleana de Riscos ou simplesmente LIBR. Esta alteração impactou os demais passos de desenvolvimento deste trabalho, entretanto os resultados com a utilização de redes neurais artificiais serão mantidos com o objetivo de nortear novas pesquisas, inclusive tratadas em trabalhos futuros nesta dissertação.

4. Definir o framework de monitoramento de riscos: para esta atividade foram desenvolvidas as seguintes subatividades:

4.1. Definir uma estratégia de configuração da simulação dos modelos de dinâmica de sistemas pelo gestor do sistema de apoio a decisão, com base nos dados do seu próprio projeto;

4.2. Permitir o mapeamento das estruturas da dinâmica de sistema em indícios de riscos;

4.3. Permitir a configuração do risco de forma personalizada com base nos indícios mapeados pelo gestor do sistema de apoio a decisão;

4.4. Estabelecer uma estratégia de identificação dos riscos no modelo simulado. Nesta etapa foram utilizados os resultados da fase 3 (a anterior a esta), descrita nesta seção.

5. **Especificar um Sistema de Apoio a Decisão (SAD):** nesta fase foi especificada uma ferramenta que implementasse a proposta deste trabalho em um sistema SAD;

6. **Construir um protótipo da ferramenta especificada:** para esta fase, foi construído um protótipo funcional da ferramenta especificada na fase 5;

7. **Verificar a aplicabilidade da ferramenta:** esta fase constitui a definição dos estudos de caso, onde é possível observar a utilização do SAD para monitoramento de riscos em projetos de software. Ainda é apresentado a utilização e a aplicabilidade da ferramenta para monitoramento de riscos em outras atividades, não somente em projetos de software, apresentando a ferramenta como um SAD genérico para monitoramento de riscos utilizando dinâmica de sistemas.

1.5 Organização do Trabalho

Como já apresentado, este trabalho pode ser considerado como extensão ao trabalho de Ambrósio (2008), onde foi definido, apresentado e validado um modelo de dinâmica de sistemas para a fase de levantamento de requisitos no processo de desenvolvimento de software. O capítulo 2 e 3 são os responsáveis por apresentar, respectivamente, o referencial teórico (revisão bibliográfica) e os trabalhos relacionados. No referencial teórico são apresentados os conceitos de riscos em projetos de software, qualidade de software, com foco especial nas normas, padrões e modelos de maturidade de software que descrevem atividades de gerência de riscos, dinâmica de sistemas e técnicas de inteligência artificial, com foco na área de redes neurais artificiais.

O capítulo 4 apresenta a descrição da técnica de monitoramento de riscos utilizando as simulações de modelos da dinâmica de sistemas. Nesse capítulo é

discutida a utilização das redes neurais artificiais como abordagem de identificação de riscos e a posterior solução utilizando a lógica booleana. Nele é descrito a técnica de configuração da simulação dos modelos, o mapeamento da dinâmica de sistemas com os indícios dos problemas, a configuração dos riscos (onde é definido o comportamento dinâmico e seus valores e a linguagem booleana de inferência de riscos (LBIR)) e como se dá a identificação dos riscos.

O capítulo 5 descreve o Sistema de Apoio a Decisão especificado, apresentando suas principais funcionalidades e sua arquitetura (a especificação completa da ferramenta é disponível nos arquivos anexos deste trabalho). O capítulo 6, apresenta os estudos de caso para este trabalho, onde são definidos ambientes de utilização da ferramenta de apoio a decisão. Nesse contexto é objetivo desse capítulo é apresentar a ferramenta no ambiente de monitoramento de riscos em projetos de software e em qualquer outro ambiente.

No capítulo 7, são apresentadas as considerações finais sobre este trabalho, bem como a discussão dos resultados. Ainda neste capítulo são apresentados os trabalhos futuros. Ao final, no Apêndice A, é apresentada ainda a especificação do software resultante da proposta deste trabalho.

2 REVISÃO BIBLIOGRÁFICA

2.1 Riscos em Projetos de Software

A palavra risco tem sua origem do italiano antigo (*risicare*), e quer dizer ousar (BERNSTEIN, 1997) e para o sentido aproximado de incerteza ou problema ela é derivada do latim *risicu* e *riscu*. Muito se discute sobre risco, de forma generalizada, como sendo a expectativa de ocorrência de um problema, quando na verdade o risco deve ser interpretado como um conjunto de incertezas quando se ousa fazer algo. O risco não deve ser encarado como um elemento ruim no processo, já que o mesmo pode ser considerado fundamental para o crescimento (SCOY, 1992). É ao arriscar que existem as maiores chances de ganhos, mas em contrapartida, é arriscando que há maior possibilidade das perdas.

Risco é considerado o grau de perigo combinado com a chance de o perigo ocasionar um incidente, ou seja, é a probabilidade de ocorrer um problema. Já problema é a materialização do risco (LEVESON, 1997). O risco é um elemento inerente à sociedade, e qualquer indivíduo, querendo ou não, está de alguma forma convivendo com eles.

Em cada área específica de conhecimento, há uma definição também específica para a utilização do termo risco. Em computação, por exemplo, o termo é utilizado em diversas subáreas, como por exemplo, em redes de computadores, engenharia de software, projeto de software, linguagens de programação, entre outros. Em administração, economia, estatística, matemática, especialmente a matemática financeira, além de ser muito empregado, o risco é tratado como elemento chave para tomada de decisão.

Nos principais livros que abordam riscos, os exemplos históricos de sua utilização sempre estão ligados ao tratamento de incertezas. Na totalidade dos exemplos, os riscos sempre eram tratados juntamente com a matemática e estatística. Com a existência de números, os matemáticos tinham a habilidade de calcular médias, desvios padrões e estabelecer medidas de amostragem. A maioria das decisões, inclusive de hoje, são feitas através das medidas analisadas de amostragens, sendo considerada essencial para se poder lidar com incerteza, já que a mesma mapeia de forma ordenada o existente. Para Bernstein (1997), em 1660, o inglês John Graunt

utilizou os desprezados registros de óbitos das pessoas de Londres entre 1604 e 1661, e com métodos de amostragem e probabilidade permitiram estabelecer parâmetros de quantas pessoas existem, de cada sexo, estado, idade, religião e profissão. Com isso ele pode apresentar em seus trabalhos as principais doenças que levavam à morte, bem como definir estatísticas de longevidade de adultos, crianças e idosos.

Graunt foi precursor da teoria da amostragem, e a forma com que analisou os dados estabeleceu os fundamentos da estatística. Hoje, as conclusões tiradas a partir de amostras de um conjunto de pessoas é conhecida como inferência estatística, que é utilizada, por exemplo, em suporte à decisão em empresas seguradoras. As seguradoras são um excelente exemplo de empresas que vivem do risco, pois assumem os riscos das pessoas mediante um pagamento estipulado em contrato.

O risco busca atribuir um grau às incertezas e por isso o gerenciamento de riscos é tão importante, já que consiste em identificar as possíveis incertezas no ambiente e tentar controlá-las. Este conhecimento não pode ser desprezado, especialmente no mundo globalizado e exigente que a sociedade está inserida. A falta de informações necessárias para a tomada de decisão caracteriza a incerteza e a atividade de minimizar esta incerteza contribui para o processo de tomada de decisão do gestor. A Figura 2.1 apresenta o espectro de gerenciamento de riscos em um projeto e é possível observar que sua atuação está justamente no tratamento das incertezas do projeto. A escala define três estados da informação para tomada de decisão: sem informação (*unknowns unknowns*), informação parcial (*knowns unknowns*) e informação completa (*knowns*). Ao ascender a escala, a incerteza sai de total incerteza até total certeza, passando por incerteza geral e incerteza específica. Durante incerteza geral e específica é quando o gerenciamento de riscos tem um grau maior de atuação. O gerenciamento de riscos é considerado um ambiente incerto, complexo e dinâmico para auxílio à tomada de decisões (SALLES JÚNIOR; SOLER; VALLE; RABECHINI JÚNIOR, 2009).

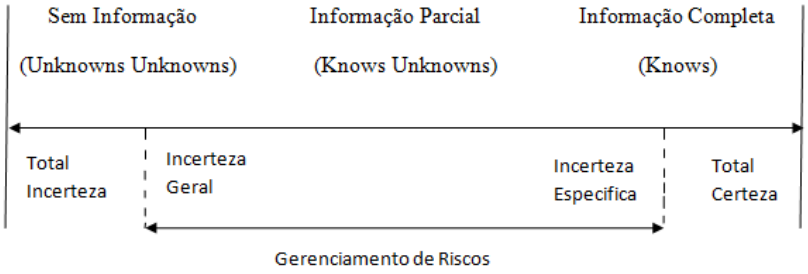


Figura 2.1. Grau de Incertezas no Gerenciamento de Riscos em Projetos.
Fonte: (SALLES JÚNIOR; SOLER; VALLE; RABECHINI JÚNIOR, 2009).

Entre as principais técnicas de identificação de riscos, merece destaque à observação do ambiente, a experiência do gestor, registros históricos, *brainstorming* (tempestade de ideias), a técnica Delphi e análise de SWOT (*strenghts, weaknesses, opportunities e threats*) ou FOFA (forças, oportunidades, fraquezas e ameaças). Assim como o gerenciamento de riscos, a atividade de identificar riscos é igualmente complexa, pois depende de um ambiente dinâmico e vários fatores. Exatamente por isso, que o monitoramento de riscos é uma atividade importante para o projeto, uma vez que assim, é possível identificar a possibilidade de ocorrência de um problema, permitindo um tempo maior de reação ao risco e análise de impacto (COELHO, 2009).

Em projetos de software, Barry Boehm (1991), Bob Charrete (1990), Tom DeMarco e Timothy Lister (2003) foram os grandes divulgadores da necessidade de se aplicar métodos de gerência de riscos no processo de desenvolvimento de software. Várias foram as propostas para este tema e vários foram os relatórios de sucesso. No entanto, nenhum método até hoje é aplicado ativamente e sistematicamente pelas organizações, salvo alguns exemplos especialmente em grandes organizações.

A gerência de risco associada com engenharia de software tem como uma de suas finalidades a de aumentar a qualidade do produto e do processo de desenvolvimento de software. Observa-se que os projetos de desenvolvimento de software, em geral, apresentam atrasos de cronogramas, custos realizados além do planejado e funcionalidades aquém das expectativas. Esses problemas, embora considerados inerentes ao desenvolvimento de software por muitos autores, podem ser minimizados e controlados pelo contínuo gerenciamento de risco de projetos (FAIRLEY, 1994).

DeMarco e Lister (2003) no livro "*Waltzing with bears: Managing risk on software projetcs*" tratam o tema de gerenciamento de riscos de forma específica para projetos de software. O livro define as principais técnicas de gestão de riscos, genéricas para qualquer área, e faz uma análise nos principais riscos envolvidos na tarefa de desenvolvimento de software definidos pelos autores: estimativas de cronograma, requisito mal especificado, rotatividade de pessoas (*turnover*), falta de acordo entre as partes interessadas e colaboradores com baixo desempenho. Para os autores, a maioria dos problemas inerentes a projetos de software são de fato previsíveis, de forma que gerenciar e monitorar os riscos irá aumentar as chances de sucesso do projeto (aumentar o risco de sucesso), consequentemente aumentar a qualidade do produto final.

O tempo de reação aos riscos é um fator preponderante para economia, de acordo com Charrete (1990) e sua escala definida em seu trabalho. Salles Júnior (SALLES JÚNIOR; SOLER; VALLE; RABECHINI JÚNIOR, 2009) compartilha desta afirmação e define um plano cartesiano comparando o tempo de reação ao risco com o custo para o projeto, conforme pode ser visto na Figura 2.2.

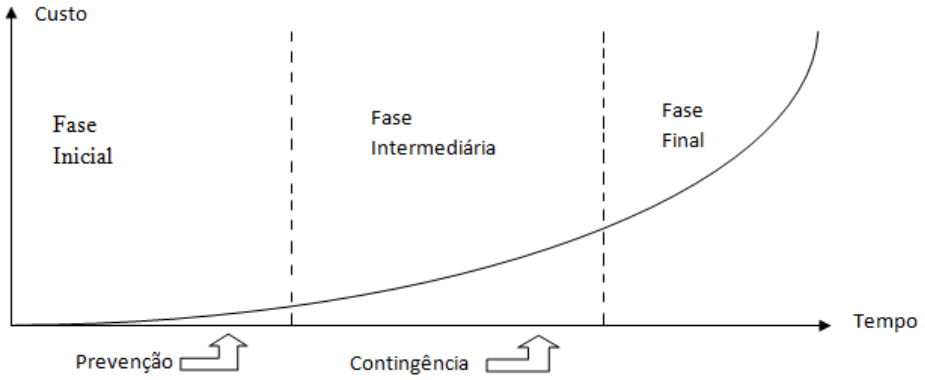


Figura 2.2. Momento da Reação aos Riscos.
Fonte: Adaptada de (SALLES JÚNIOR; SOLER; VALLE; RABECHINI JÚNIOR, 2009).

A reação imediata, feita no momento da identificação e da análise dos riscos, é chamada de reação de prevenção ou de contenção, e acontece antes da decisão final sobre o projeto, alterando as principais variáveis de impacto no projeto, como escopo, qualidade, tempo ou condições financeiras. O quanto antes ocorrer o momento de decisão, menores serão os custos caso o problema venha a impactar o projeto.

Entretanto, esta análise (presente na maioria dos livros sobre o tema), avalia somente os riscos negativos, onde há iminência de um problema ocorrer. Os riscos positivos não são avaliados. A identificação e monitoramento destes riscos positivos contribuem para que o gestor possa levar o projeto em sua direção, permitindo que o risco impacte o projeto. A Tabela 2.1, apresenta as principais reações de acordo com o tipo de risco identificado, sendo positivo ou negativo.

Riscos Negativos	Riscos Positivos
Prevenir	Provocar
Transferir	Compartilhar
Mitigar	Melhorar
Aceitar	Aceitar

Tabela 2.1. Reações a Riscos.
Fonte: Adaptada de (CHARRETE, 1990).

2.2 Qualidade de Software

> "*Qualidade é relativa. O que é qualidade para uma pessoa pode ser falta de qualidade para outra.*" (WEINBERG, 1994).

Qualidade de Software é uma área de conhecimento da disciplina de Engenharia de Software que trata definir parâmetros para buscar garantir qualidade na fase de desenvolvimento de software e qualidade no produto final do desenvolvimento. Entretanto, qualidade é um conceito amplo e de várias definições distintas e de vários pontos de vista. Associa-se qualidade muito ao software entregue, sem considerar o processo de desenvolvimento. Contudo, a qualidade do software final depende da qualidade existente durante todas as fases de desenvolvimento.

Do ponto de vista do desenvolvedor, a qualidade está na concepção de que o software vá atender as necessidades do cliente, e do ponto de vista do cliente, está na utilidade do software e ao cumprimento dos requisitos estabelecidos. Segundo a Borland (2009), qualidade é "*a convergência entre requisitos completos, o código*

correto e o mínimo de defeitos – todos alinhados para atingir os objetivos do negócio". De acordo com Pressman (2006) sempre é necessário definir explicitamente o termo qualidade de software, quando o mesmo é dito. Para se buscar qualidade é necessário criar um conjunto de atividades que irão ajudar a garantir que cada produto de trabalho da engenharia de software exiba alta qualidade, realizando atividades de segurança da qualidade em cada projeto, com o uso de métricas para desenvolver estratégias para a melhoria de um processo de software e, como consequência, a qualidade no produto final.

Na contramão da definição, Lewis (2004), apresenta os principais indicadores de falta de qualidade: (a) o software que foi entregue frequentemente apresenta falhas, (b) inaceitáveis consequências de falhas de sistema, desde financeiras até cenários reais de aplicação, (c) sistemas não estão frequentemente disponíveis para o uso pretendido, (d) sistemas são frequentemente muito caros e (e) custo de detectar e remover os defeitos são excessivos.

McCall e Cavano (1978) sugerem como métricas para qualidade de software os seguintes fatores da qualidade, que avaliam o software em três distintas dimensões: transição do produto, avaliando a portabilidade, reutilização e interoperabilidade, a revisão do produto, que avalia a manutenção, flexibilidade e testabilidade, e a operação do produto, que avalia a correção, confiabilidade, usabilidade, integridade e eficiência do software. A Figura 2.3 apresenta as dimensões propostas por McCall e Cavano (1978).

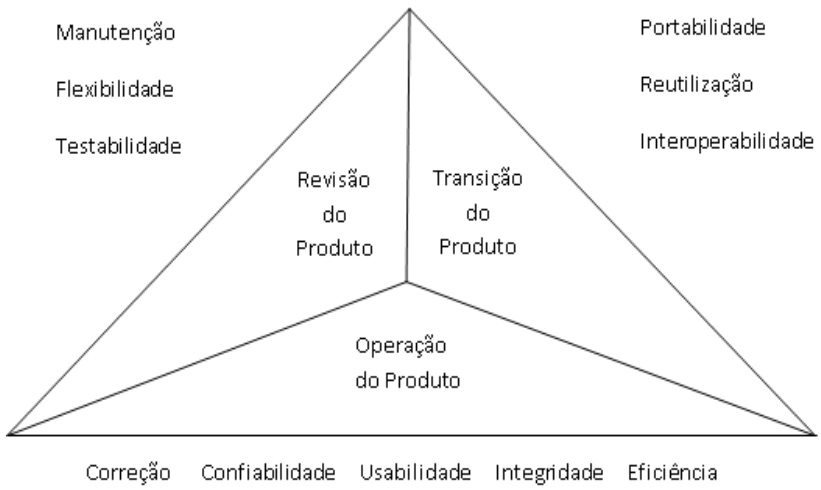


Figura 2.3. Fatores de Qualidade de McCall e Cavano.
Fonte: Adaptada de (MCCALL; CAVANO, 1978).

Entretanto, mesmo com estas definições, qualidade é um conceito relativo, não exato. Ao apoiar a liberdade de expressão no século 16, Voltaire disse que "*as leis devem ser claras, uniformes e precisas. Interpretar as leis é quase sempre corrompê-las*". Na época ele defendia a criação de uma lei única para toda população. E com o objetivo de tornar [ou pelo menos tentar tornar] claro, uniforme e preciso o conceito de qualidade, foram criadas diversas normas e padrões que descrevem detalhadamente o procedimento para se buscar qualidade no processo de software. Entre os principais organismos normativos existentes, destacam-se o ISO (*International Organization for Standardition*), IEEE (*Intitute of Electrical and Electronic Engineers*) e a ABNT (Associação Brasileira de Normas Técnicas). A Tabela 2.2 apresenta as principais normas e suas descrições que tratam qualidade de software.

Norma	Descrição
ISO 12207	Processo de ciclo de vida de software
ISO/IEC 12119:1994	Pacotes de software – requisitos de qualidade e testes
ISO/IEC 14598-1:1999	Avaliação de qualidade de produtos de software
ISO/IEC 9126-1:2001	Modelos de qualidade – características
ISO/IEC 25000:2005	Modelo de qualidade de software, nova versão das séries 14598 e 9126
ISO/IEC 20926:2003	Medida de software por pontos de função
ISO/IEC 90000-3:2004	Diretivas para aplicação da ISO 9001 ao software
ISO/IEC 9001:2000	Requisitos para sistemas de gerenciamento de qualidade (aplicável a qualquer empresa, de software ou não)
NBR 13596	Listam um conjunto de características que devem ser verificadas em um software para que ele seja considerado um software de qualidade

Tabela 2.2. Principais Normas de Qualidade de Software.
Fonte: Elaborada pelo Autor.

Neste sentido, foi criado, sob o patrocínio da IEEE, o documento SWEBOK (2004) – g*uide to the software engineering body of knowledge* – com a finalidade de servir como referência sobre questões de engenharia de software. A qualidade de software foi abordada neste documento em um tópico exclusivo, onde são definidos os fundamentos de qualidade de software, os processos de gerenciamento de qualidade de software e apresenta ainda algumas considerações práticas sobre o tema.

A busca por qualidade deve impactar todos os processos de desenvolvimento de software, inclusive as pessoas que são as grandes responsáveis pela execução das tarefas. Não irá adiantar ter processos bem definidos, padrões estabelecidos se as pessoas que conduzam os processos não estiverem engajadas na busca por qualidade (KOSCIANSKI, 2007). Pessoas mal preparadas e desorganizadas, ao conduzirem algum processo, especialmente de desenvolvimento de software, minam a qualidade do produto final, além de serem menos eficientes. Com essa premissa, sistemas como o Kaizen (KAIZEN, 2009) defendem que a qualidade é resultado do método e filosofia de trabalho das pessoas. A principal técnica do Kaizen é o 5s (*seiri* – organização, utilização e liberação da área, *seiton* – ordem e arrumação, *seiso* - limpeza, *seiketsu* – padronização, asseio e saúde e *shitsuke* – disciplina e autodisciplina) que prega organização pessoal, especialmente no ambiente de trabalho, buscando maior produtividade pela redução da perda de tempo com tarefas desnecessárias. Redução de despesas e melhor aproveitamento de materiais, melhoria da qualidade de produtos e serviços, menos acidentes de trabalho e maior satisfação das pessoas com o seu próprio trabalho (5S, 2009).

Ainda com o objetivo de definir qualidade no conceito de engenharia de software, foram criados modelos de maturidade de software, como por exemplo, o CMMI (*Capability Maturity Model for Software*) (CMMI, 2009) e o MPS.BR (Melhoria de Processo de Software Brasileiro) (SOFTEX, 2006). O MPS.BR é um modelo para melhoria de processo do software brasileiro e está em desenvolvimento desde 2003. Ele é dividido em sete níveis de maturidade: A (em otimização), B (gerenciado quantitativamente), C (definido), D (largamente definido), E (parcialmente definido), F (gerenciado) e G (parcialmente gerenciado). A escala de maturidade se inicia no nível G e evolui até o nível A. Esses níveis de maturidade definem o grau de melhoria para um predeterminado conjunto de processos no qual todos os resultados esperados são atendidos. A meta principal do MPS.BR é definir e aprimorar um modelo de melhoria e avaliação de processo de software. É definido também um processo e um método de avaliação, o qual dá sustentação e garante que o MPS.BR está sendo empregado de forma coerente com as suas definições. Já o CMMI, é uma proposta do SEI (*Software Engineering Institute*) (SEI, 2007), com objetivo semelhante ao MPS.BR, mas este é dividido em 5 níveis: do 1 ao 5. Sendo o primeiro o nível inicial, o segundo repetitivo, o terceiro definido, o quarto gerenciado quantitativamente e por fim, o quinto, otimizado.

Especificamente no nível C do MPS.BR, dentro dos atributos de processos (APs) são definidas a ADR (análise de decisão e resultado), sendo estas de 1 ao 7, e a GRI (gerência de risco), estas de 1 ao 9. Este nível tem especial importância no contexto deste trabalho, pois trata sobre o gerenciamento de riscos. O GRI-1 define o escopo de gerência de riscos, o GRI-2 trata as origens e categorias de riscos, além dos parâmetros usados para analisar os riscos, categorizá-los e controlar o esforço da gerência do risco, o GRI-3 apresenta as estratégias apropriadas para a gerência de riscos, o GRI-4 define como os riscos do projeto são identificados e documentados, definindo possíveis consequências para o projeto e partes interessadas. O GRI-5 apresenta a definição de como priorizar, estimar e classificar os riscos em categorias, já o GRI-6 define como desenvolver planos de mitigação para os riscos, o GRI-7 apresenta as técnicas de análise dos riscos e a prioridade de aplicação dos recursos para o monitoramento desses riscos. O GRI-8 define as medições dos riscos para avaliar o progresso de seu tratamento e, por fim, o GRI-9 apresenta a seção onde ações apropriadas são executadas para corrigir ou evitar o impacto do risco, baseadas na sua prioridade, probabilidade, consequência ou outros parâmetros definidos pelo gestor.

2.3 Dinâmica de Sistemas

A dinâmica de sistemas baseia-se em teorias e modelos matemáticos para simulação de sistemas complexos, não lineares e que contenham estruturas de repetição, os chamados *feedback loops* (STERMAN, 2000). Em geral, a dinâmica de sistemas, não está interessada em valores precisos. O foco principal está nas tendências do sistema. O objetivo é saber se o sistema está estável ou instável, se tende a crescer, oscilar, declinar ou manter o equilíbrio. A base das teorias da dinâmica de sistemas é o comportamento dinâmico que é consequência de sua estrutura causal (FORRESTER, 1961).

Para Sterman (2000), a dinâmica de sistemas é um método para aumentar o aprendizado em sistemas complexos. Assim como empresas de aviação usam simuladores de voo para auxiliar pilotos, a dinâmica de sistemas é em parte, um método para construção e desenvolvimento destes simuladores de voo. O conceito central deste

método é a *two-way causation* ou *feedback loop*. O pressuposto é que decisões são derivadas de informações sobre o sistema. Essas decisões resultam em ações que têm como objetivo mudar o sistema. Quando uma nova informação chega sobre as condições do sistema pode-se, então, verificar se o próprio sistema mudou ou não. Essa nova informação gera outras decisões/ações que podem produzir mais mudanças no sistema. Isso é uma sequência circular de causas e efeitos, o que em dinâmica de sistemas chama-se de *feedback loop* ou *two-way causation*. Os modelos da dinâmica de sistemas são formados por várias destas estruturas inter-relacionados.

Quando foi criada, a utilização da dinâmica de sistemas foi limitada devido à falta de um ambiente de simulação. Praticamente não existiam computadores e a aplicação das teorias era realizada de forma manual, o que limitava o escopo de análise do ambiente. Com o avanço tecnológico e principalmente com o avanço da computação que ocorreu após a década de 80 foi possível utilizar a dinâmica de sistemas de forma mais efetiva, podendo tratar inclusive um ambiente de simulação complexo. O desenvolvimento de ferramentas para dinâmica de sistemas também foi outro facilitador da utilização do método. A ferramenta precursora foi a Dynamo (*Dynamic Models*) (STERMAN, 2000). Era muito rústica, mas já apresentava muitas melhorias quando comparado com o método utilizado anteriormente. Novas ferramentas foram surgindo e hoje elas se estabelecem de forma segura e madura para o desenvolvimento de modelos dinâmicos. Entre as principais ferramentas disponíveis que atuam direta ou indiretamente no auxilio ao desenvolvimento de um modelo de dinâmica de sistemas são apresentadas: iThink[1], Stella[2], PowerSim[3], Vensim[4], Extend[5], Vissim[6], Simile[7], System Dynamics[8], Sphinx SD Tools[9], MatLab[10].

Todo o embasamento da dinâmica de sistemas está no pensamento sistêmico (*System Thinking*) explorado por Peter Senge (1990) em seu livro "A Quinta Disciplina". O pensamento sistêmico compreende um corpo de métodos, ferramentas e

[1] iThink – Disponível em http://www.iseesystems.com
[2] Stella – Disponível em http://www.iseesystems.com
[3] PowerSim – Disponível em http://www.powersim.com
[4] Vensim – Disponível em http://www.vensim.com
[5] Extend – Disponível em http://www.imaginethatinc.com
[6] Vissim – Disponível em http://www.vissim.com
[7] Simile – Disponível em http://www.simulistics.com
[8] System Dynamics – Disponível em http://sourceforge.net/projects/system-dynamics
[9] Sphinx SD Tools – Disponível em http://sourceforge.net/projects/sphinxes
[10] MatLab – Disponível em http://www.mathworks.com

princípios orientados para as inter-relações sistêmicas e o pensamento de processo (SENGE, 1990).

Esta abordagem de pensamento sistêmico se contrapõe à ideia de pensamento analítico, exemplificado como sendo uma abordagem anterior à abordagem sistêmica. Todos os fenômenos, nesse universo, poderiam ser entendidos isolando e analisando as partes que os compõem. No entanto, esta abordagem torna-se subjetiva, já que houve um crescimento extraordinário da interdependência, devido aos sistemas complexos construídos pelo homem, fazendo necessária uma abordagem sistêmica, que analisa a interdependência de n fatores em um determinado ambiente que esteja sendo modelado.

A chave do pensamento sistêmico está na interdependência, ou seja, na forma como os elementos de um sistema estão ligados uns aos outros, o que, para Forrester (1961), é mais importante do que a análise pontual destes elementos de forma independente, cartesiana (pensamento analítico). Essas ligações são descritas na forma de diagramas de influência (ou diagrama de causalidades), exemplificado na Figura 2.4, que têm como objetivo a análise da interação entre os elementos do problema (variáveis, dados constantes e outros elementos), suas estruturas de *feedback* e a visualização de efeitos defasados no tempo (*delays*) de alterações nas condições do problema.

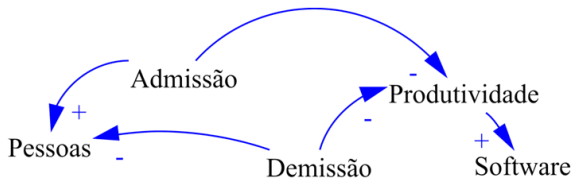


Figura 2.4. Diagrama de Influências.
Fonte: Adaptada de (MADACHY, 2007).

As setas indicam a direção da causalidade, enquanto os sinais mostram se o efeito está no mesmo sentido (sinal positivo) ou sentido inverso (sinal negativo). Sinal positivo indica relação direta entre as variáveis (aumento/diminuição em uma variável ocasiona aumento/diminuição na outra variável), e sinal negativo indica relação inversa (aumento/diminuição em uma variável indica diminuição/aumento na outra variável). Uma seta cruzada por dois traços paralelos indica que a relação de causa e efeito entre

as variáveis não é imediata: há uma defasagem de tempo (*delay*) entre o estímulo e a resposta (STERMAN, 2000).

A Figura 2.4 apresenta uma situação onde são definidas algumas variáveis do contexto de desenvolvimento de software. A admissão em projetos de software causa um efeito positivo à variável pessoas. Entretanto, causa efeito inverso na produtividade. Por exemplo, caso haja admissões, pessoas é impactado positivamente e produtividade é impactada de forma inversa, ou seja, decresce. Segundo o modelo de Madachy (2007), isso ocorre, pois é necessário um período de treinamento para que o novo membro da equipe admitido tenha rendimento satisfatório. É a chamada curva de aprendizagem. Demissão possui efeito inverso nas variáveis pessoas e produtividade. Por fim, produtividade possui efeito positivo em software, que busca representar o processo de desenvolvimento de software do projeto como um todo.

Nos diagramas de influência, é possível definir dois tipos de *feedback*. Um deles positivo ou *feedback* de reforço, e outro negativo ou *feedback* de equilíbrio. Um *feedback* positivo é o que gera um círculo vicioso ou virtuoso. Em um círculo vicioso uma queda em uma variável, numa corrente causal, resulta em mais piora da mesma variável. No círculo virtuoso uma melhora numa variável do sistema faz com que a variável melhore ainda mais. O *feedback* positivo é a força geradora de crescimento ou declínio em um sistema; ele tende a amplificar qualquer distúrbio em um sistema. Já o *feedback* negativo ou de equilíbrio, tenta manter o sistema em equilíbrio, ele reage a qualquer distúrbio e tende a manter o sistema em equilíbrio.

Um exemplo de *feedback* positivo, que pode ser representado pelo diagrama de influências é a representação sistêmica da Lei de Brooks (BROOKS, 1995). A lei afirma que adicionar mais pessoas a um projeto que está atrasado irá atrasá-lo ainda mais. A modelagem sistêmica é apresentada na Figura 2.5. A variável software planejado tem um efeito positivo ao cronograma, que por sua vez tem um efeito positivo em pessoal (pessoas), este impacta positivamente em treinamento e comunicações. Um novo membro na equipe necessita de treinamento e aumenta o *overhead* de comunicação. Estas duas variáveis, impactam inversamente a produtividade. Esta apresenta um efeito positivo em software desenvolvimento. Ou seja, se produtividade cai, software desenvolvido cai. Se produtividade aumenta, software desenvolvido aumenta. E este último trás um impacto inverso ao cronograma.

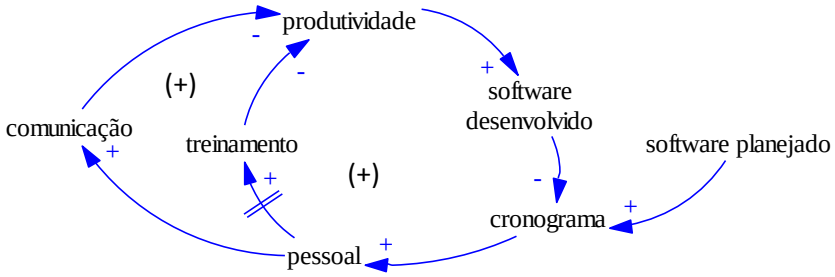


Figura 2.5. Lei de Brooks Representada pelo Diagrama de Influências.
Fonte: Adaptada de (MADACHY, 2007).

O pensamento sistêmico é o primeiro passo para a formalização de problemas de dinâmica de sistemas que, por meio do uso de computadores, tornam possível a representação explícita, formalizada e matematizada dos modelos mentais, permitindo comunicar, de forma clara, suas pressuposições aos demais agentes interessados, os quais poderão elaborar, de modo construtivo, críticas e melhorias referentes ao modelo. Ele permite uma nova ótica do ambiente, respeitando as variáveis que o torna complexo e dinâmico.

Os diagramas de influência são boas ferramentas para representar a interdependência e processos de *feedback*. Entretanto, não capturam a estrutura de estoques e fluxos de um sistema, que são os principais elementos da dinâmica de sistemas. Os estoques retratam as variáveis representativas de recursos que são acumuladas no sistema, já os fluxos são as funções de decisão ou políticas de um sistema, descritas na forma de equações algébricas simples, responsáveis pela acumulação ou consumo dos estoques. Estas equações envolvem variáveis auxiliares e, ou, o valor atual dos estoques (STERMAN, 2000). A Figura 2.6 apresenta a representação gráfica dos estoques e fluxos. São apresentados também os demais itens que podem compor um diagrama de estoques e fluxos. É importante observar que este é um exemplo.

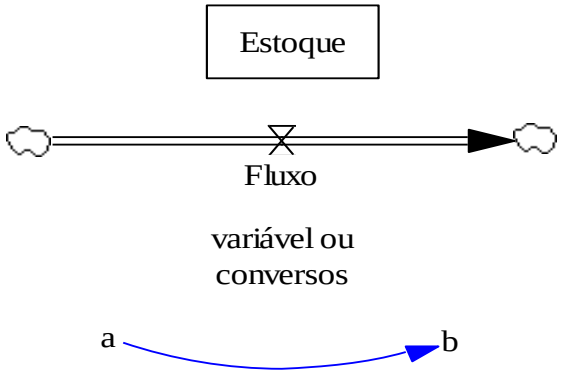


Figura 2.6. Taxonomia para o Diagrama de Estoques e Fluxos.
Fonte: Elaborada pelo Autor.

Outras notações / taxonomias podem ser encontradas, principalmente se for comparada uma ferramenta com outra. Mas a base conceitual é a mesma. Na Figura 2.6 é possível identificar que um estoque é representado por um retângulo. Um fluxo por uma espécie de torneira limitando o tráfego de informações. As variáveis podem ser representadas por círculos ou simplesmente letras no diagrama, como apresentada na Figura 2.6 e a seta estabelece uma ligação entre os elementos do diagrama, definindo uma relação causal no diagrama.

A visualização do comportamento dinâmico, no diagrama de estoques e fluxos, pode ser traduzida em funções que posteriormente são apresentadas em gráficos. Estas são definidas por variáveis de influência e pelos *feedbacks* apresentados pelo sistema modelado. A dinâmica de sistemas aponta que três modos fundamentais de estruturas de *feedback* são a base para se entender muitos dos comportamentos dinâmicos que são observados. Há basicamente o gráfico de crescimento, gerado por *feedback* positivo, de decrescimento, gerado pelo feedback negativo, quando não há busca do sistema pelo equilíbrio, de equilíbrio, quando ocorre o equilíbrio no sistema e por fim, o comportamento do gráfico de oscilação. Neste último, também é causado pelo *feedback* negativo, mas quando há *delays* nas ações (atrasos). Desta forma, as ações corretivas demoram algum tempo para terem efeito. Essa dinâmica repetida inúmeras vezes, acaba gerando o comportamento oscilatório.

2.4 Técnicas de Inteligência Computacional

Na década de 30, muitos filósofos, matemáticos, estatísticos e probabilistas já conduziam seus trabalhos na possibilidade da simulação do pensamento e inteligência humana. Entretanto, a inteligência artificial nasceu oficialmente em 1956 com a conferência de verão de *Dartmouth College*, nos Estados Unidos (MCCORDUCK, 1979). De acordo com Barr e Feigenbaum (1981), a inteligência artificial "*é a parte da ciência da computação que compreende o projeto de sistemas computacionais que exibam características associadas, quando presentes no comportamento humano, à inteligência*". Já para Charniak e McDermott (1985) "*é o estudo das faculdades mentais através do uso de modelos computacionais*".

Existem basicamente duas linhas de pesquisa para a construção de sistemas inteligentes: a linha conexionista e a linha simbólica. A conexionista trabalha com a proposta de simular a inteligência humana através da simulação de componentes do cérebro, isto é, os neurônios e suas ligações sinápticas. Esta linha trabalha basicamente com a subárea de Redes Neurais Artificiais. Já a linha simbólica estabelece a manipulação simbólica de um grande número de fatos especializados sobre um domínio restrito como o paradigma corrente, servindo como base para definição de sistemas especialistas (do inglês *expert systems*) (HAYKIN, 2008).

As redes neurais artificiais são sistemas de processamento paralelo que buscam reproduzir as funções das redes neurais biológicas, implementando seu comportamento funcional e sua dinâmica. Nas redes neurais artificiais, o procedimento usual na solução de problemas passa por uma fase de aprendizagem, no qual um conjunto de exemplos é apresentado à rede, que extrai informações para a resolução do problema. Esta capacidade de aprender por meio dos exemplos e de generalizar as informações para a resolução dos problemas é sua principal característica.

O primeiro modelo matemático que se propunha simular um neurônio biológico foi apresentado em 1943 por Warren McCulloch e Walter Pitts em "*A logical calculus of the ideas immanent in nervous activity*" (MCCULLOCH; PITTS, 1943) e é apresentado na Figura 2.7. Ele foi chamado de modelo MCP (McCulloch e Pitts). O modelo apresenta *n* terminais de entrada (representando os dendritos do modelo biológico) (x_1, x_2, ..., x_i). Estes representam as ativações dos neurônios anteriores, caso

haja. A há apenas uma saída, representando o axônio (y_j). Para estabelecer o comportamento das sinapses, os terminais de entrada possuem pesos acoplados, estes chamados de pesos sinápticos (w_{1j}, w_{2j}, ..., w_{ij}). Estes pesos são estabelecidos para determinar o grau em que o neurônio deve considerar sinais de disparo para estabelecer uma determinada conexão (BRAGA; CARVALHO; LUDERMIR, 2007). No neurônio biológico ocorre um disparo quando a soma dos impulsos que ele recebe excede o seu limiar de excitação, o chamado *threshold*. No artificial, o mesmo é representado por um mecanismo de somatório ponderado. Ele faz a soma dos valores $x_i w_i$ recebidos pelo neurônio e decide se deve ou não disparar (saída 1, no caso de disparar, e 0 caso não disparar). No modelo MCP, o somatório ponderado serve de entrada para uma função que determina se o neurônio irá ou não ser ativado. Esta função é chamada de função de ativação.

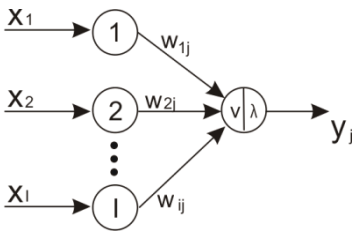


Figura 2.7. Modelo MCP de Neurônio Artificial.
Fonte: Adaptada de (MCCULLOCH; PITTS, 1943).

O primeiro modelo de rede neural artificial de fato, ou seja, vários neurônios interligados formando uma rede, foi proposto por Rosenblatt (1958) e foi chamado de Perceptron. Ele descreveu uma topologia de rede baseada na utilização de neurônios MCP, mas ligados, formando uma rede. Sua principal contribuição foi na definição das estruturas de ligação entre os neurônios e no algoritmo de treinamento que permitia a rede executar determinados tipos de funções. Em 1969, Minsky e Papert (1969) apresentaram algumas limitações da rede Perceptron, especialmente em resolver problemas linearmente separáveis, ou seja, problemas cuja solução somente pode ser obtida dividindo-se o espaço de entrada em duas regiões por meio de uma superfície linear. Por este motivo, o modelo de rede Perceptron foi conhecido como Perceptron simples (BRAGA; CARVALHO; LUDERMIR, 2007).

Após este período, e até a década de 80, a abordagem conexionista ficou adormecida, especialmente por conta da repercussão do trabalho de Minsky e Papert. A partir da década de 80, o tema despertou interesse para a comunidade internacional, especialmente por conta dos avanços da tecnologia, em especial com o avanço da microeletrônica, que permitiu o desenvolvimento físico dos neurônios e as simulações das redes. Houve avanços das linguagens de programação, ambientes computacionais e de *hardware*. Até antes disso, não haviam sistemas capazes de realizar as simulações das redes neurais. Não havia poder de processamento para suportar tais operações, ainda mais em modelos baseados em processamento paralelo.

Avanços existiram e novas visões foram propostas. Com alguns trabalhos, foi possível identificar que a visão de Minsky e Papert era pessimista quanto à utilização da rede Perceptron. As redes neurais de múltiplas camadas compostas por neurônios com funções de ativação *sigmoidais* nas camadas intermediárias, em 1988 (CYBENKO, 1989), já obtinha êxito em resolver problemas não lineares. Estas redes foram chamadas de Perceptron de Múltiplas Camadas ou MLP (*Multilayer Perceptron*). Muitos outros avanços ocorreram, com novas propostas e novas soluções, especialmente nas estratégias de treinamento e aprendizado das redes.

Há basicamente dois tipos de aprendizado: o supervisionado, onde se destacam as abordagens baseadas em correção de erros e por reforço. E o aprendizado não supervisionado, onde são definidos, entre outros, os aprendizados hebbiano e por competição. No aprendizado supervisionado a rede recebe exemplos de pares de entradas e saídas, onde para cada entrada a rede produz uma resposta na saída. A resposta é comparada com o sinal de saída desejado e assim, a rede gera um sinal de erro que corresponde à diferença do sinal. E assim sucessivamente, ajustando os pesos sinápticos, até que a rede tenha absorvido o aprendizado. Neste paradigma, dados válidos são necessários para servir de exemplo para a rede. Os principais algoritmos são o Regra Delta (WIDROW; HOFF, 1960) e sua generalização para redes de múltiplas camadas e o algoritmo chamado retro propagação (*back-propagation*) (RUMELHART; HINTON; WILLIAMS, 1986).

Já no aprendizado não supervisionado não há saída desejada. A rede é treinada através de excitações ou padrões de entrada, para então, arbitrariamente, organizar os padrões em categorias. Esse aprendizado é destinado a problemas que

visam a descoberta de características com base em dados estatísticos (nas entradas). Os algoritmos mais conhecidos são os mapas de Kohonem (1982) e os modelos de ART (CARPENTER; GROSSBERG, 1988 apud BRAGA; CARVALHO; LUDERMIR, 2007).

As redes neurais artificiais se aplicam basicamente em problemas em que existem dados para servir de entrada e que possibilite o aprendizado. Com este aprendizado a rede usará sua capacidade de generalização para gerar algum resultado. A Tabela 2.3, apresenta as principais aplicações das redes neurais artificiais e as estratégias de aprendizado a elas associadas (BRAGA; CARVALHO; LUDERMIR, 2007).

Tarefas	Algumas Aplicações
Classificação	Reconhecimento de caracteres, reconhecimento de imagens, diagnóstico, análise de risco de crédito, detecção de fraudes, detecção de falhas em sistemas industriais.
Categorização	Agrupamento de sequencias de DNA, mineração de dados, análise de expressão gênica, agrupamento de clientes.
Previsão	Previsão de tempo, previsão financeira, modelagem de sistemas dinâmicos e previsão de sequencias de DNA.

Tabela 2.3. Principais Aplicações para Redes Neurais Artificiais.
Fonte: (BRAGA; CARVALHO; LUDERMIR, 2007).

A linha simbólica também tem seu destaque, especialmente com as propostas de modelos para representação de incertezas, com o modelo nebuloso, probabilista, possibilita e da evidência. Modelos de computação evolutiva, como programação evolutiva e algoritmos genéticos são representações desta linha, além dos sistemas especialistas.

Há também modelos para representação de conhecimento, como as técnicas baseadas em regras (RBS – *Rule Based Systems*) como, por exemplo, as regras de produção. Estas são representações procedimentais, baseadas na lógica de primeira ordem, que definem algum tipo de conhecimento. Desta forma, o conhecimento é representado como uma coleção de regras do tipo SE condição ENTÃO ação. Este esquema é considerado um dos melhores e mais simples meios disponíveis para representar a experiência de especialistas na resolução de problemas (RUSSEL;

NORVIG, 2003). Bratko (1990) apresenta a modularidade, facilidade de edição e transparência do sistema como as principais vantagens para o uso desta técnica. Modularidade, pois cada regra pode ser considerada um elemento de conhecimento independente das demais; Facilidade de edição, já que a natureza modular das regras é fácil de acrescentar, editar e excluir regras; e, por fim, transparência do sistema, uma vez que a técnica garante maior legibilidade da base de conhecimentos definida.

3 TRABALHOS RELACIONADOS

A dinâmica de sistemas, como já apresentado, foi criada por Jay Forrester em 1958, entretanto o livro que divulgou esta técnica de modelagem de sistemas foi lançado três anos depois, em 1961 (FORRESTER, 1961). O método, ainda chamado de dinâmica industrial, era retratado especificamente para o ambiente industrial. Posteriormente, Forrester publicou outro livro apresentando a técnica sendo aplicada em um ambiente urbano (FORRESTER, 1969), para então publicar um novo livro apresentando a dinâmica de sistemas como uma técnica de modelagem de sistemas abrangente (FORRESTER, 1971).

Após a publicação das teorias de Forrester, vários autores publicaram trabalhos fazendo uso da técnica em suas áreas de atuação, especialmente na economia e administração. Em 1964, em sua tese de doutorado, o professor Edwards B. Robert, propôs, pela primeira vez, a utilização das técnicas de modelagem de sistemas com dinâmica de sistema no gerenciamento de projetos (ROBERTS, 1964). Com o passar dos anos, e em especial com a evolução do desenvolvimento de software na época, este trabalho de Robert cativou autores que atuam na área de desenvolvimento de software, devido a semelhanças das áreas de gerenciamento de projeto e gerenciamento de projetos de software. Foi quando surgiram os primeiros trabalhos aplicando as técnicas de dinâmica de sistemas e temas correlatos à engenharia de software.

O primeiro trabalho que se destacou nesta área foi o de Tarek Abdel-Hamid e Stuart E. Madnick, apresentado no artigo intitulado "*Lessons Learned from Modeling the Dynamics of Software Development*" (ABDEL-HAMID; MADNICK, 1989) e relatado no livro "*Software Project Dynamics: An Integrated Approach*" (ABDEL-HAMID; MADNICK, 1991). Tanto no artigo, quanto no livro, os autores apresentam a utilização da dinâmica de sistemas no suporte ao entendimento das variáveis e suas relações para o gerenciamento de projetos de software, de forma a permitir uma melhor compreensão do ambiente.

A Figura 3.1 apresenta o modelo, em diagrama de influências, proposto por Abdel-Hamid e Madnick em seu livro (ABDEL-HAMID; MADNICK, 1991). Para os autores, os processos de gerenciamento de recursos humanos dispõem variáveis que representam o nível de experiência da equipe, treinamento de membros novatos, tempo

para assimilação deste treinamento, perdas em comunicação interna dada a quantidade de membros da equipe, entre outros. Estas variáveis se relacionam entre si, influenciando uma a outra. E com isso, influenciam diretamente a atividade de gerenciamento de projetos no seu resultado final.

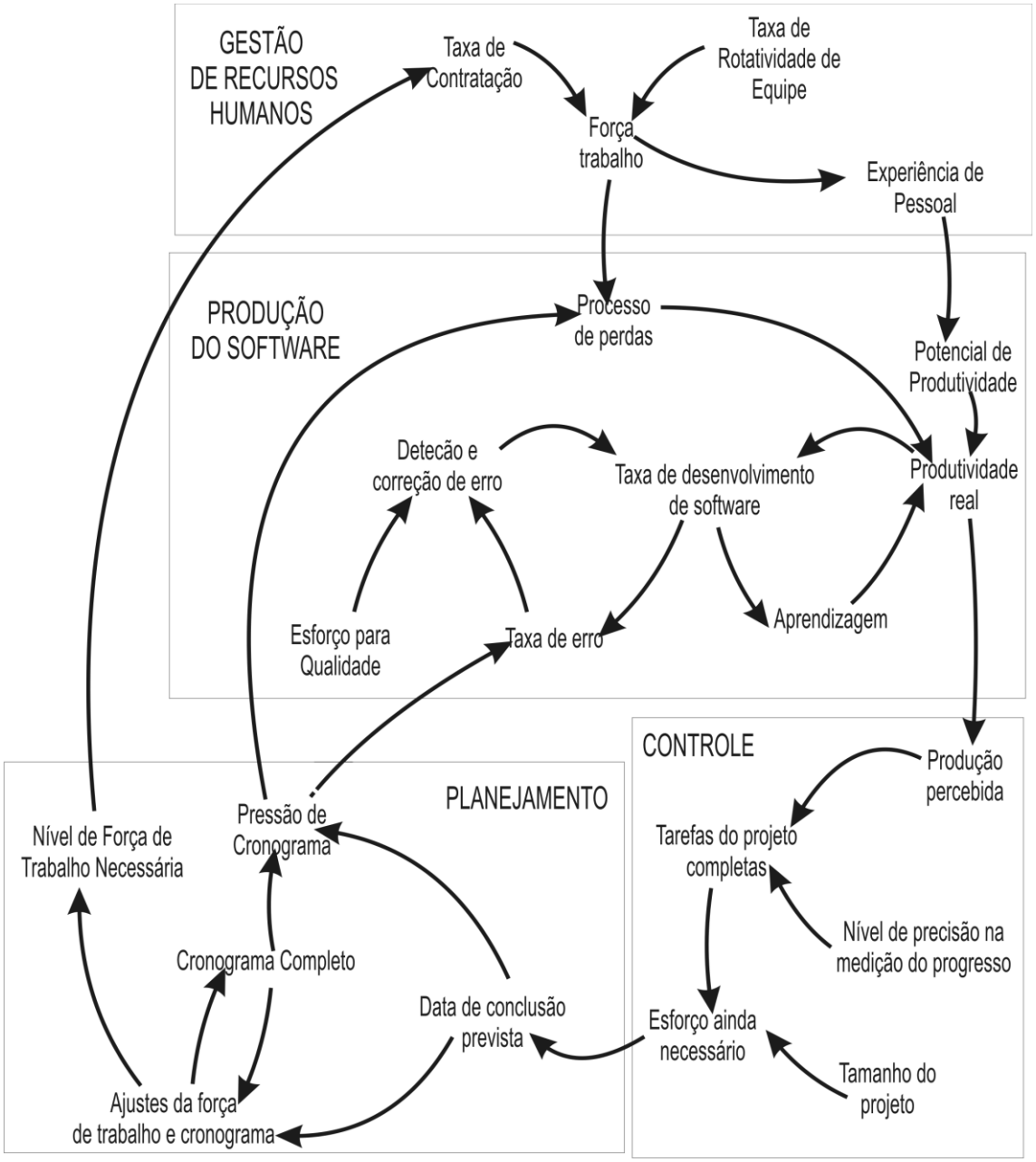


Figura 3.1. Diagrama de Influências do Modelo Dinâmico para Gerenciamento de Projetos de Software.
Fonte: Adaptada de (ABDEL-HAMID; MADNICK, 1991).

O modelo proposto define as principais variáveis de quatro grandes áreas do gerenciamento de projetos de software: planejamento (*planning*), controle (*control*), produção de software (*software production*) e gerenciamento de recursos humanos (*human resource management*). Cada uma destas áreas com suas respectivas variáveis interagindo com as demais.

Com os experimentos realizados, este modelo conseguiu apresentar novos conceitos e situações inerentes à atividade de gerenciamento de projetos de software, que se quer eram avaliadas pelos gestores. Foi possível verificar que cada estimativa para um projeto de software origina um projeto diferente, ou seja, os prazos disponíveis influenciam diretamente nas decisões e no comportamento dos envolvidos no projeto, diminuindo ou aumentando a produtividade da equipe. Quanto maior for o índice de segurança para a estimativa do prazo (ou seja, prazo maior), maior é o custo do projeto, pois os desenvolvedores se sentem mais tranquilos quando o prazo é maior, o que causa um prejuízo para sua produtividade.

Em 1996, Abdel-Hamid publicou um artigo apresentando sua visão sobre melhoria de produtividade no desenvolvimento de software, onde ele afirma que a tarefa é complexa e melhorar a produtividade requer conhecer muito bem o ambiente e não somente as técnicas e linguagens para o desenvolvimento (ABDEL-HAMID, 1996). Ele descreve a utilização da dinâmica de sistemas para apoiar esta atividade em busca da melhoria de produtividade.

Abdel-Hamid continuou trabalhando no tema, onde orientou alguns trabalhos, entre eles: (LIN; ABDEL-HAMID; SHERIF, 1997) e (SENGUPTA, ABDEL-HAMID; BOSLEY, 1999). No trabalho com Lin e Sherif foi definido um modelo de simulação para processos de engenharia de software, que recebeu o nome de SEPS (*Software-Engineering Process Simulation Model*). Já com Sengupta e Bosley, foi realizado um estudo experimental para investigar os atrasos em projetos de software relacionados com a equipe de desenvolvimento. Ambos apresentando a dinâmica de sistemas no contexto de desenvolvimento dos trabalhos.

James Collofello, professor da Universidade do Estado do Arizona[11], responsável pelas cadeiras de introdução à engenharia de software e projeto de software, processos e gerenciamento de qualidade, também é investigador da área sendo

[11] Universidade do Estado do Arizona - http://www.asu.edu/

responsável por um grupo de estudo na sua universidade. Os seus principais trabalhos sobre o tema são relacionados: (COLLOFELLO; RUS, 1998a), (COLLOFELLO, 1998), (COLLOFELLO; RUS, 1998b), (COLLOFELLO; RUS; CHAUHAN; HOUSTON; SYCAMORE; SMITH-DANIELS, 1998), (COLLOFELLO; ROEHLING, 1999), (COLLOFELLO; SYCAMORE, 1999), (RUS; COLLOFELLO, 1999), (RUS; COLLOFELLO; LAKEY, 1999), (RUS; COLLOFELLO, 2001), (HOUSTON; COLLOFELLO, 2001) e (HOUSTON; COLLOFELLO; MACKULACK, 2001).

Em seus trabalhos, Collofello investiga a utilização da dinâmica de sistemas em projetos de software como base de um sistema de suporte a decisão. Para ele, compreender as variáveis e as influencias delas em projetos de software, contribui para a definição de um ambiente adequado para tomada de decisão. Ao longo de seus textos o tema risco ou gerenciamento de risco é sempre lembrado, sendo tema inclusive de um de seus trabalhos intitulado "*Simulating Risk Factors for Software Development Risk Managemet*" (HOUSTON; COLLOFELLO; MACKULACK, 2001).

Em período semelhante, Ray Madachy cria um grupo de estudos aplicado a engenharia de software com dinâmica de sistemas. O foco de seus trabalhos é na utilização da dinâmica de sistemas para apoiar a modelagem de processos de software e estimativas de software. Seus principais trabalhos são os listados: (MADACHY, 1995), (MADACHY, 1996a), (MADACHY, 1996b), (MADACHY, 1996c), (MADACHY, 1996d), (MADACHY, 1998), (MADACHY; TARBET, 1998), (MADACHY, 1999), (KELLNER, MADACHY, RAFFO, 1999), (MADACHY; TARBET, 2000).

Como resultado de seu trabalho, Madachy publicou um livro chamado "*Software Process Dynamics*", dedicado a apresentar as técnicas de modelagem de dinâmica de sistemas para processos de software. Em seu livro, Madachy apresenta alguns modelos de dinâmica de sistemas e suas simulações específicas para apoio à atividade de processo de desenvolvimento de software nas suas principais fases. A Figura 3.2 exibe um modelo de estoques e fluxos do processo de desenvolvimento de software em uma visão de alto nível. O autor define este modelo de alto nível para apresentar submodelos deste, mais específicos e com mais detalhamentos. Desta forma ele faz uso de uma abordagem *top-down*, indo do geral para o específico para apresentar seu trabalho.

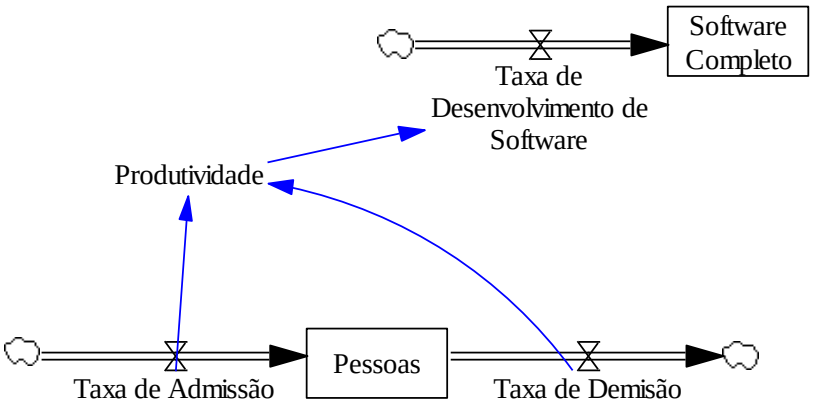


Figura 3.2. Diagrama de Estoques e Fluxos do Processo de Software – Visão Alto Nível.
Fonte: Adaptada de (MADACHY, 2007).

Ainda na década de 90, no Brasil, o professor Guilherme Horta Travassos e a professora Cláudia Maria Lima Werner, juntamente com o grupo de engenharia experimental na COPPE-UFRJ (Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia da Universidade Federal do Rio de Janeiro), coordenado pelo professor Guilherme, publicaram trabalhos envolvendo o tema dinâmica de sistemas e gerência de projetos de software. O principal foco do trabalho destes pesquisadores é na construção de modelos para simular situações de risco nos projetos de desenvolvimento de software, de forma a fornecer conhecimento ao gestor, permitindo-o analisar e avaliar os riscos, além de verificar os impactos das intervenções planejadas.

Entre os trabalhos mais relevantes sobre o tema correlato do grupo, destacam-se os trabalhos referentes à tese de doutorado de Márcio de Oliveira Barros, orientado do professor Guilherme e da professora Claudia: (BARROS; WERNER; TRAVASSOS, 1999), (BARROS; WERNER; TRAVASSOS, 2001a), (BARROS; WERNER; TRAVASSOS, 2001b), (BARROS; WERNER; TRAVASSOS, 2001c) e (BARROS; WERNER; TRAVASSOS, 2003).

Ainda no grupo da COPPE-UFRJ, vários são os trabalhos que se utilizam das técnicas de dinâmica de sistemas para representação de conhecimento, como por exemplo, o trabalho de Araújo (2005) que contextualizou o ambiente de evolução de software, conforme pode ser observado pela Figura 3.3, utilizando o diagrama de influências. No modelo são retratadas as seguintes variáveis: requisitos: requisitos especificados do software / artefatos gerados; tamanho: a quantidade de artefatos produzidos em cada fase do processo de desenvolvimento de software proposto;

esforço: a quantidade de intervenções nos artefatos (número de inclusões, alterações e exclusões em cada artefato); eficiência: identificados pela quantidade de pessoas e recursos alocados, tempo gasto e produtividade média da equipe, de acordo com a versão de cada artefato; periodicidade: o intervalo de tempo decorrido entre cada versão de cada artefato produzido; complexidade: os elementos que permitem medir a complexidade estrutural do artefato; manutenabilidade: o tempo gasto na identificação e remoção de defeitos; modularidade: o acoplamento e coesão identificados nos artefatos; confiabilidade: a quantidade de defeitos identificados pelo artefato em cada versão do mesmo, além de disponibilidade do sistema.

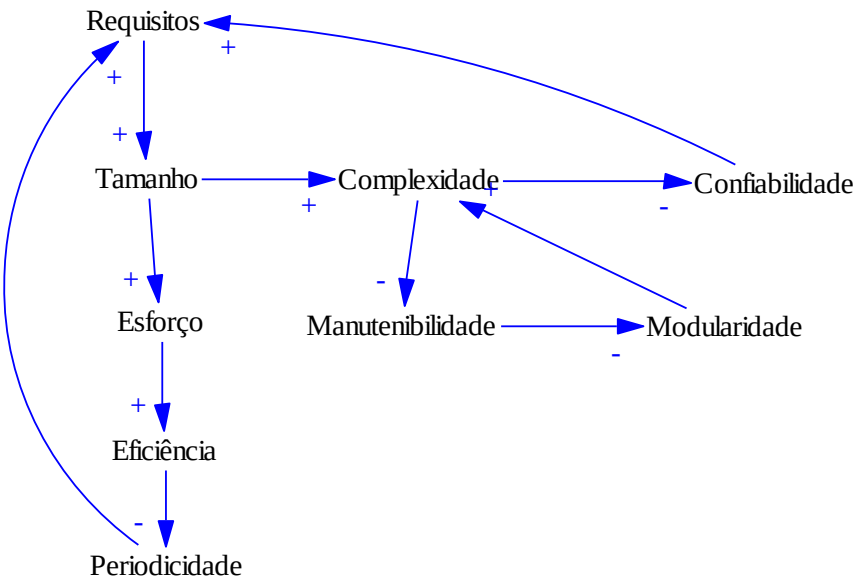


Figura 3.3. Diagrama de Influência para o Ambiente de Evolução de Software.
Fonte: (ARAÚJO; TRAVASSOS, 2005).

Em 2000, Rodrigues defendeu sua tese de doutorado onde ele propõe um método de integração para gerenciamento de projetos com dinâmica de sistemas, onde a avaliação de riscos do projeto é suportado (RODRIGUES, 2000). Em 2001, como resultado de sua tese, Rodrigues publicou um artigo propondo um *framework* para gerenciamento e modelagem de riscos em projetos utilizando a dinâmica de sistemas (RODRIGUES, 2001).

Na UFV[12] (Universidade Federal de Viçosa), no grupo de pesquisa dedicado à engenharia de software e coordenado pelo professor José Luis Braga, no Departamento de Informática, Bernardo Ambrósio em sua dissertação de mestrado, apresenta um modelo de dinâmica de sistemas para a fase de requisitos em processos de desenvolvimento de software (AMBRÓSIO, 2008). Ele aborda a modelagem das variáveis e dos relacionamentos envolvidos na fase de requisitos, com base no método proposto por Boehm e Turner para análise de risco em projetos de software: tamanho da equipe, rotatividade de pessoal (*turnover*), competência dos profissionais, volatilidade dos requisitos e existência de requisitos emergentes não previstos inicialmente (BOEHM; TURNER, 2003).

Em seu trabalho, Ambrósio apresenta e discute o modelo de dinâmica de sistemas construído para a fase de levantamento de requisitos. Para facilitar o entendimento, ele apresenta o modelo a partir de três visões diferentes, que ele definiu de Nível 0, 1 e 2. O modelo nível 0 é uma visão que contém todas as variáveis e relacionamentos do modelo. O nível 1 é uma visão que contém as principais variáveis e os respectivos relacionamentos existentes no modelo. Essa é a principal visão do modelo e é utilizada para descrever toda a sua estrutura e as interações dinâmicas entre as variáveis. O modelo nível 2 é uma visão mais simplificada do modelo, constituída por apenas um pequeno subconjunto das variáveis contidas no Modelo Nível 1. O modelo nível 2 é apresentado na Figura 3.4 onde é possível observar as principais variáveis, estoques e relacionamentos do modelo.

Nas pesquisas realizadas foi possível identificar trabalhos associando risco em projetos de software com dinâmica de sistemas. Nestes trabalhos, os autores constroem modelos de dinâmica de sistemas, específicos para cada área, de forma a identificar os riscos inerentes ao ambiente simulado. Sempre com um ambiente planejado para a identificação dos riscos. Não foram encontrados trabalhos onde a tarefa de identificação / monitoramento de riscos é independente do modelo simulado e que permita a configuração do risco de acordo com a vontade e experiência do gestor. Apesar de ser uma técnica muito utilizada em áreas industriais, econômicas e na administração, não foi possível identificar, na literatura, relatos de sucesso da aplicação

[12] Universidade Federal de Viçosa – http://www.ufv.br

das técnicas da dinâmica de sistemas na área comercial da engenharia de software, fora da área acadêmica.

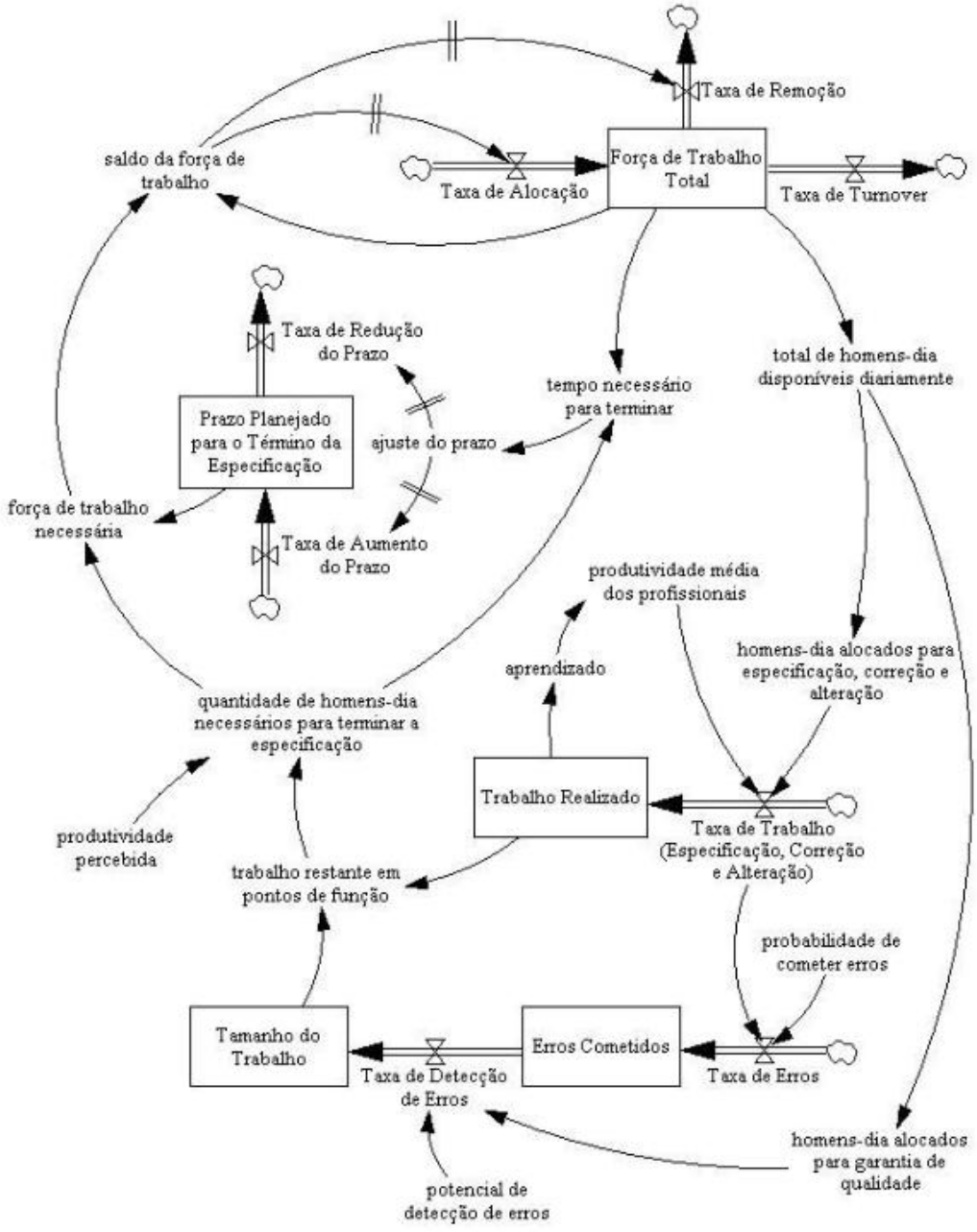


Figura 3.4. Modelo de Dinâmica de Sistema para a Fase de Requisitos de Processos de Software – Modelo Nível 2.
Fonte: (AMBRÓSIO, 2008).

A Figura 3.5 apresenta uma linha do tempo destacando os principais trabalhos publicados relatados neste capítulo com seus respectivos autores (os principais

dos trabalho), desde a criação da dinâmica de sistemas em 1961 até a finalização deste trabalho, em 2010. A figura busca representar o período da primeira até a última publicação do autor sobre o tema, de acordo com o apresentado neste capítulo.

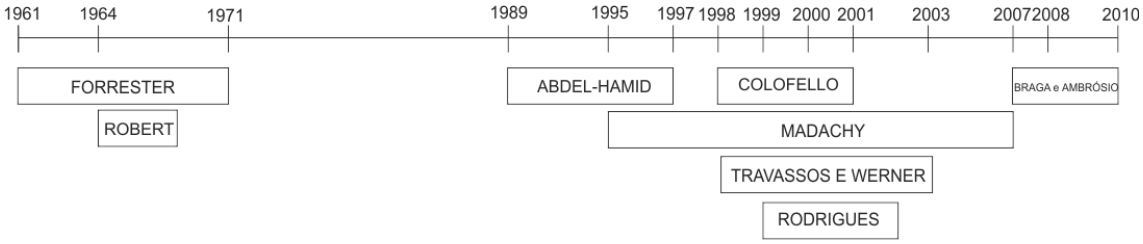


Figura 3.5. Linha do Tempo com Autores Citados em Trabalhos Relacionados.
Fonte: Elaborada pelo Autor.

Diferentemente dos trabalhos apresentados neste capítulo, este trabalho está interessado em apresentar uma técnica de monitoramento de riscos utilizando os resultados de modelos de simulações de dinâmica de sistemas. Não é objetivo apresentar um novo modelo de dinâmica de sistemas para projetos de software.

A técnica descrita neste trabalho pode ser utilizada para qualquer modelo de dinâmica de sistemas e não somente dos modelos específicos para simulação de ambientes de projetos de software, podendo, inclusive, contribuir para outras áreas de domínio não conexas a este trabalho.

4 MONITORAMENTO DE RISCOS UTILIZANDO AS SIMULAÇÕES DE DINÂMICA DE SISTEMAS E TÉCNICAS DE INTELIGÊNCIA COMPUTACIONAL

4.1 Introdução

A capacidade de enxergar e compreender o sistema como um todo é fundamental nas decisões. Em projetos de software, os gestores, quando buscam visualizar de forma integrada todos os fatores de influência do seu processo de desenvolvimento, eles terão maior capacidade de tomar decisões promissoras. No geral, quando os gestores não possuem esta visão do sistema como um todo, eles tomam decisões reativas, considerando apenas o problema presente, sem relacioná-lo com o seu ambiente, suas variáveis e demais fatores correlacionados (SENGE, 1990).

Desenvolvendo modelos voltados à engenharia de software, a dinâmica de sistemas busca contribuir para a predição e identificação de problemas que poderão impactar um projeto de software (identificação dos indicativos que podem se tornar materialização de riscos). Isso ocorre durante as simulações dos modelos, onde é possível estabelecer períodos de tempo de execução, como, por exemplo, simular o modelo para os próximos seis meses, um ano, dois anos, ou mais. Esses indicativos servem como base para a proposta deste trabalho.

A Figura 4.1 (a) apresenta um cenário em que os gerentes conduzem o projeto fazendo uso de alguma estratégia de gestão de riscos. Na mesma Figura, mas no contexto b, é o que se espera na utilização integrada desta abordagem de monitoramento de riscos utilizando dinâmica de sistemas e técnicas de inteligência computacional, onde o risco é identificado de forma prematura, dando mais tempo ao gestor ou à equipe de agir ao eminente problema. Em ambos os contextos da Figura 4.1 é definida uma linha do tempo (*time line*) onde são observadas duas entradas: o risco identificado pela equipe e o problema. A diferença entre estas duas situações é o tempo Δt para a tomada de decisão.

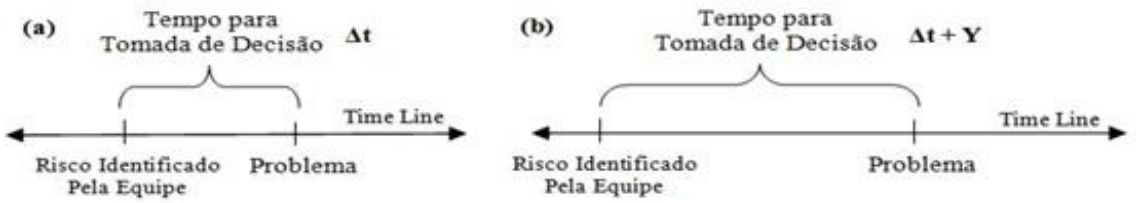


Figura 4.1. Abordagem Comparativa no Tempo para Identificação de Riscos.
Fonte: Elaborada pelo Autor.

A busca por esta identificação do risco de forma prematura é o grande motivador desta proposta (no contexto b da Figura identificado por Δt + Y). Fazendo uma analogia com carros em uma autoestrada, é possível imaginar dois carros andando um atrás do outro. Quando o primeiro freia, o segundo deve frear também para evitar uma batida. Quanto maior for a distância do primeiro para o segundo carro, mais tempo de reação o motorista do segundo carro terá.

Assim que é definido o modelo de dinâmica de sistemas e suas variáveis configuradas de acordo com o ambiente analisado, o mesmo é executado, gerando resultados desta simulação. Estes resultados representam o comportamento do sistema de acordo com o período estabelecido e são definidos por funções matemáticas, que para melhor visualização são apresentados em forma de gráfico. Este comportamento, basicamente, pode representar uma tendência oscilatória, de crescimento, de decrescimento ou de estabilidade do sistema. É com estas tendências que a proposta deste trabalho se baseia, utilizando técnicas de inteligência computacional para a configuração e identificação dos riscos. O objetivo é estabelecer um framework para monitoramento de risco utilizando a experiência do gestor para as configurações, como em um sistema de apoio a decisões.

Esta proposta é representada por duas abordagens: uma utilizando redes neurais artificiais e outra utilizando sistemas baseados em regras / regras de produção. A primeira abordagem foi desenvolvida e aplicada, entretanto durante a avaliação dos resultados foi possível identificar que era possível aplicar outra técnica, baseada na álgebra de *boole*, que apresentava o mesmo resultado e com menor esforço (a comparação foi feita analisando os passos seguidos em ambas abordagens). Foi quando a segunda abordagem foi desenvolvida com base no modelo para representação de conhecimento de regras de produção.

Como a abordagem utilizando redes neurais artificiais se mostrou satisfatória, a mesma é apresentada no contexto deste trabalho para direcionar novas pesquisas sobre a área, inclusive sugeridas em trabalhos futuros. A segunda abordagem também é apresentada, e é a utilizada para implementar o protótipo da ferramenta SAD especificada.

Na seção 4.2 é apresentada a arquitetura da proposta, sendo exibida a arquitetura da primeira e da segunda abordagem. Já no seção 4.3 são apresentadas provas de conceito da arquitetura. Um exemplo pequeno de utilização para as duas abordagens e por fim, na seção 4.4, são apresentadas as conclusões desta proposta e considerações finais sobre sua utilização.

4.2 Arquitetura

A Figura 4.2 exibe o esquema da arquitetura da proposta, que é definido em quatro etapas. A primeira representa a definição, configuração e simulação do modelo de dinâmica de sistemas (DS). Esta etapa é independente da proposta, e por isso está sendo representada visualmente como um componente na figura. A etapa dois é onde ocorre o mapeamento dos resultados da simulação dos modelos de dinâmica de sistemas com os indícios de riscos. A terceira etapa é onde é feita a definição e configuração dos riscos, com base nos indícios mapeados na segunda etapa. E por fim, na quarta etapa são identificados os riscos de acordo com as definições apresentadas nas etapas anteriores. As etapas dois, três e quatro são o escopo desta proposta e definem os limites do *framework* para monitoramento de risco proposto no trabalho para implementação de um sistema de apoio a decisões (SAD). Na Figura 4.2 ainda são definidas setas entre os passos dois e quatro, que representam a possibilidade de voltar para alguma etapa caso seja necessário para fazer algum refinamento, adaptação, modificação ou exclusão de alguma configuração.

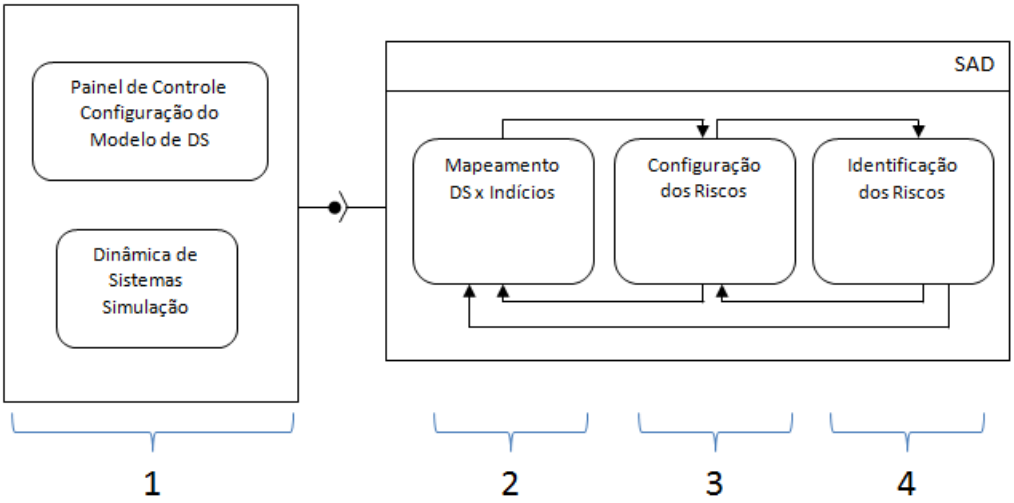


Figura 4.2. Esquema da Arquitetura da Proposta para Monitoramento de Riscos Utilizando Dinâmica de Sistemas e Técnicas de Inteligência Computacional.
Fonte: Elaborada pelo Autor.

Nas próximas seções deste capítulo são apresentadas as quatro etapas que definem esta arquitetura: configuração da simulação, mapeamento dinâmica de sistemas x indícios, configuração / definição dos riscos e por último a identificação dos riscos. Conforme fora apresentado, estão sendo exibidas duas abordagens para o monitoramento de riscos. A primeira utilizando redes neurais artificiais e a segunda utilizando regras de produção. Em ambos os casos, as etapas são as mesmas, entretanto há especificidades para cada um dos casos em algumas delas. A etapa um e dois são exatamente iguais para as duas abordagens, já nas etapas três e quatro há diferenças e por esta razão há uma divisão para cada uma das abordagens no texto.

4.2.1 Etapa 1 - Configuração da Simulação

Com o ambiente de simulação preparado e carregado, o gestor deve configurar as variáveis do modelo para ajustar de acordo com o ambiente que ele queira simular. O modelo deve ser escolhido de acordo com as necessidades de monitoramento, ou seja, o modelo deve ser compatível e tratar as variáveis e seus relacionamentos com base nas necessidades de monitoramento do gestor. De acordo com Ambrosio (2008), "*nesta etapa os gerentes devem ajustar o modelo de acordo com*

o cenário que será simulado e as características da organização, da equipe e do projeto que definem o contexto da simulação".

Esta tarefa consiste em configurar as variáveis do modelo que será simulado. Cada ferramenta de simulação possui sua forma de configuração. Na ferramenta Vensim, por exemplo, há basicamente dois tipos de variáveis: as de configuração, responsáveis pela portabilidade dos modelos e que permitem ajustá-lo de acordo com as características da organização, da equipe e do projeto; e as variáveis gráficas, que representam variáveis complexas e difíceis de serem definidas por meios de equações ou operações matemáticas. Neste último caso, é configurada a curva do gráfico na variável. Nem todas as ferramentas possuem as variáveis gráfico para utilização. E claro que a configuração das variáveis depende do tipo de variáveis que o modelo escolhido utilizou.

Como já apresentado, não é objetivo deste trabalho definir um modelo de dinâmica de sistemas para projetos de desenvolvimento de software. No capítulo 3 deste trabalho são apresentados diversos autores que trabalharam no desenvolvimento destes modelos e podem ser utilizados nesta etapa de acordo com a necessidade de cada projeto.

Ambrosio (2008) define um modelo para a etapa de levantamento de requisitos em projeto de software e, em seu trabalho, ele utiliza a ferramenta Vensim para a modelagem e simulação de seu modelo (trabalho apresentado no capítulo 3 em trabalhos relacionados). Essa ferramenta possui uma particularidade que é a criação de um painel de controle capaz de representar todas as variáveis do modelo em um único lugar, como um painel, sem que sejam exibidos os estoques e fluxos. Somente as variáveis configuráveis do modelo são exibidas, facilitando assim a tarefa de configuração do modelo pelo gestor, que muita das vezes não compreende o diagrama inteiro de estoques e fluxos da dinâmica de sistemas. Das ferramentas analisadas, somente a Stella/iThink e a Vensim possuem esta característica.

Algumas versões dessas ferramentas possuem também a possibilidade de se integrar em outros sistemas utilizando uma API (*Application Programming Interface*). Assim é possível desenvolver um módulo configurador das variáveis do modelo em um sistema à parte. Todavia, este recurso somente é disponibilizado nas versões pagas de algumas ferramentas, com é o caso da Vensim. Para este caso em particular, é

disponibilizado um pacote JAR (extensão com classes Java / componente) para integrar todos os recursos de simulação e configuração do modelo dentro da ferramenta em qualquer aplicação JAVA[13].

4.2.2 Etapa 2 - Mapeamento Dinâmica de Sistemas x Indícios

Após o modelo ser simulado na Etapa 1, estes resultados são usados como entradas para a Etapa 2. Nesta etapa são mapeados os recursos do modelo simulado de dinâmica de sistemas para os indícios de riscos da proposta deste trabalho. Os indícios são indicações, meios de prova da ocorrência de algum risco no sistema simulado.

O objetivo é portar o modelo da dinâmica de sistemas para a proposta deste trabalho para monitoramento de riscos. O mapeamento é feito com os recursos dos modelos de dinâmica de sistemas: os estoques, variáveis e fluxos, e deve ser feito, seguindo as regras apresentadas na Tabela 4.1.

Dinâmica de Sistemas	Mapeamento de Indícios
Estoque, Variável, Fluxo e Taxa	Podem ser mapeados em um indício associado a um comportamento dinâmico.

Tabela 4.1. Mapeamento Dinâmica de Sistemas x Indícios.
Fonte: Elaborada pelo Autor.

É possível observar que não há obrigatoriedade no mapeamento. Em todos os casos, sempre é utilizada a palavra *pode* ao estabelecer o mapeamento (representa uma associação do tipo 0 para 1, sem obrigatoriedade). O recurso somente é mapeado caso haja a necessidade de sua utilização para posterior configuração de algum risco com o indício na Etapa 3. Em todos os três casos, cada um dos indícios mapeados deve estar associado a um comportamento dinâmico. Este comportamento dinâmico é definido de acordo com a simulação executada e pode ser: crescimento, decrescimento, oscilação e estável.

[13] Java – Linguagem de Programação - http://java.sun.com

Esses indícios, definidos nessa etapa, servem como entrada para a configuração e definição dos riscos na Etapa 3. O exemplo apresentado na seção 4.3, serve como prova de conceito desta proposta e é exibido para facilitar o entendimento dos conceitos aqui definidos.

4.2.3 Etapa 3 - Configuração / Definição dos Riscos

Até a Etapa 3, todos os procedimentos eram exatamente iguais para as duas abordagens: utilizando redes neurais artificiais ou regras de produção. A partir desta, há procedimentos específicos para cada uma das abordagens apresentadas. Nesta etapa é onde são feitas as definições e configurações dos riscos de acordo com os indícios. Os indícios, conforme já apresentado, são baseados nos mapeamentos realizados na Etapa 2 e no comportamento dinâmico simulado na Etapa 1.

Nesta etapa o gestor tem a possibilidade de criar e configurar os riscos que serão monitorados no projeto, de acordo com a sua experiência. Cada risco criado é associado a um ou mais indícios de risco com um determinado comportamento dinâmico. Esse comportamento dinâmico é configurado pelo gestor de acordo com a necessidade. Já o indício, deve ser somente os que foram mapeados na Etapa 2.

Após o risco estar definido e configurado, de acordo com os indícios e comportamento dinâmico, estes indícios (riscos configurados) são entradas para a Etapa 4, onde os riscos são identificados. A seguir, é apresentado o processo para definição e configuração dos riscos em cada uma das abordagens.

Utilizando Redes Neurais Artificiais:

O primeiro passo é associar os indícios para a criação de cada risco. Neste caso é necessário estabelecer qual indício e qual comportamento será parte integrante da configuração do risco. É possível associar 1 ou mais indícios para se configurar um determinado risco. Para isso é utilizada a álgebra de *boole* para estabelecer as associações, sendo permitida a utilização dos operadores E (AND), OU (OR) e o de negação (NOT) de acordo com esta proposta.

Supondo a existência de três indícios: Taxa de Produção de Software, Produtividade Individual e Software Completo, poder-se-ia criar um risco chamado *fadiga em membro da equipe* e configurá-lo da seguinte maneira: produtividade individual decresce E taxa de produção de software decresce. Desta forma, o monitor de riscos iria alertar o gestor quando o modelo simulado apresentasse esta situação representada pela associação destes dois indícios e comportamentos dinâmicos.

Essa configuração serve como entrada para a rede neural que é discutida em detalhes na Etapa 4 – Identificação dos Riscos. O objetivo da rede neural artificial neste caso é atuar na identificação do risco de acordo com seus indícios.

Utilizando Regras de Produção:

Para a definição e configuração utilizando regras de produção, foi criada, para este trabalho, uma micro linguagem lógica com base nos conceitos de regras de produção para definir a configuração dos riscos. Esta linguagem foi chamada de Linguagem Booleana de Inferência de Riscos (LIBR). Esta linguagem faz a configuração dos riscos de maneira semelhante à abordagem utilizando redes neurais artificiais, entretanto, os riscos são definidos utilizando a LIBR como estratégia de representação do conhecimento do gestor. Os operadores lógicos presentes na definição da LIBR são: E (AND), OU (OR) e negação (NOT).

A Listagem 4.1 apresenta o padrão que deve ser usado para a definição dos riscos. A cláusula base é a SE ENTÃO. Se indício(s), então o risco ocorre. Para cada risco definido pode haver um ou mais indícios associados com base na LIBR.

Listagem 4.1. Sintaxe para Inferência da LIBR.
Fonte: Elaborada pelo Autor.

```
1.  SE
2.    INDICIO(s)
3.  ENTÃO
4.    RISCO
```

A LIBR trata ainda a definição de valores e limites para os indícios, diferentemente da abordagem utilizando a rede neural artificial. Neste caso, para cada

indício é possível definir um valor ou um limite de valor, além do comportamento dinâmico, respeitando o que é definido na Tabela 4.2.

Descrição	Símbolo
Maior	>
Maior ou igual	>=
Igual	=
Menor ou igual	<=
Menor	<

Tabela 4.2. Limite de Valores para os Indícios em Regras de Produção - LIBR.
Fonte: Elaborada pelo Autor.

4.2.4 Etapa 4 - Identificação dos Riscos

Nesta etapa os riscos são identificados de acordo com as definições feitas nas Etapas anteriores. Para cada uma das abordagens há sua especificação.

Utilizando Redes Neurais Artificiais:

Cada risco e seu conjunto de indícios devem ser mapeados em variáveis (X1, X2, ..., Xi para o conjunto de indícios e Y1, Y2, ..., Yi para o risco – por exemplo). Estas variáveis formam o conjunto de entrada (indícios) e de saída (riscos) da rede neural artificial. Ou seja, cada conjunto de indícios, representados pelas variáveis Xi, são as entradas da rede. De acordo com os resultados da simulação dos modelos da dinâmica de sistemas, estas entradas irão se ativar ou não, fazendo com que a rede decida qual risco irá ocorrer, ativando assim o(s) risco(s) (definidos com variáveis Yi) que foram identificados.

A rede neural artificial modelada é uma rede do tipo *Perceptron* de uma única camada, já que o problema é do tipo linearmente separável. Depois disso é necessário codificar as informações de entradas e saídas esperadas em base binária, ou seja, 0 ou 1. Para representar as variáveis mapeadas em binário, basta saber quantas são as variáveis (base 10) e verificar quantos dígitos binários serão necessários para a sua

representação (base 2) (isso pode ser obtido com a fórmula: x <= 2^n, onde n é o número *bits* necessário para a representação das variáveis e o x é o número de variáveis mapeadas). Caso haja dois riscos e três conjuntos de indícios mapeados, serão necessários 2 dígitos para representar os indícios (00, 01 e 10) e apenas 1 dígito para representar os dois riscos (0 e 1). A representação visual da rede neural artificial modelada é apresentada na Figura 4.3.

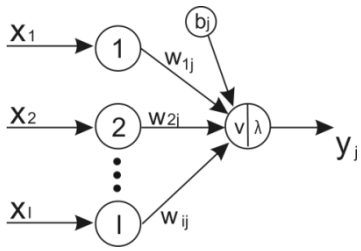


Figura 4.3. Rede Perceptron para Identificação de Riscos.
Fonte: Elaborada pelo Autor.

A rede pode ser formada por vários neurônios semelhantes ao apresentado na Figura 4.3. Um único neurônio pode processar *N* indícios de *M* Riscos. Entretanto, há a possibilidade de estabelecer um neurônio por risco. Possibilitando inclusive que a saída de um neurônio (a existência de um determinado risco ou não) seja a entrada de outro neurônio. Assim a relação de existência deste risco é o indício de ocorrência do risco antes processado.

O modelo artificial modelado também inclui uma polarização ou bias de entrada, representado na Figura 4.3 pela letra b. Esta variável é incluída no somatório da função de ativação com o objetivo de aumentar a capacidade de aproximação da rede. O valor do bias é igualmente ajustado como os pesos sinápticos. O bias permite que um neurônio apresente um resultado não nulo ainda que todas as suas entradas sejam nulas. Um exemplo de sua representatividade é o problema do ou exclusivo da lógica, que leva de duas falsidades a uma verdade.

Após definir as variáveis binárias que representam as entradas e saídas do modelo neural artificial, a rede deve passar pela fase de treinamento. Este treinamento é supervisionado, uma vez que existem os dados de entrada e saídas desejadas disponíveis. A função de ativação da rede é apresentada em (1), já a de transferência é

definida em (2). Esta função de transferência considera somente a saída para um único bit, que pode representar até dois riscos. Caso seja necessário mais bits, basta criar novos intervalos de valores para definir mais saídas. Em (3) é apresentada a fórmula de cálculo da taxa de erro, onde *d* é o sinal de saída desejado (esperado) e *y* é o sinal de saída calculado pela rede. Por fim, em (4) é definida a função de correção para os pesos sinápticos durante o treinamento da rede, onde *w(it)* é o valor do peso corrigido, *w(i)* é o valor do peso na iteração anterior do treinamento, μ é a taxa de aprendizado (para este cenário é definida arbitrariamente sempre com o valor 1), *E* é a taxa de erro calculada e por fim *x(i)* é sinal de entrada da interação anterior.

$$v = \sum_{i=1}^{n} w(i)x(i). \quad (1)$$

$$\gamma(v) = \begin{cases} 1, \forall\, v > 0 \\ 0, \forall\, v \leq 0 \end{cases}. \quad (2)$$

$$E = d - y. \quad (3)$$

$$w(it) = w(i) + \mu E x(i). \quad (4)$$

Durante todo o treinamento, o bias possuí valor arbitrado em 1 e para o primeiro passo do treinamento todos os pesos sinápticos são 0. Após treinada, a rede está apta a receber os estímulos (os indícios) e com eles identificar a existência do risco, afim de monitorá-lo.

Utilizando Regras de Produção:

A identificação dos riscos utilizando regras de produção é mais simples. O processo consiste na inferência lógica da expressão definida de cada risco pela LIBR. Com base nos resultados da simulação, cada uma das condições serão aferidas e o resultado da inferência irá indicar a existência ou não do risco. Caso o resultado a inferência seja VERDADEIRO, haverá risco. Caso FALSO, não haverá o risco no ambiente simulado.

4.3 Prova de Conceito

O objetivo desta seção é apresentar uma prova de conceito sob a utilização da técnica proposta neste trabalho de monitoramento de riscos em projetos de software utilizando os resultados da simulação de dinâmica de sistemas e técnicas de inteligência computação. As duas abordagens propostas serão tratadas nas seções a seguir.

O ambiente modelado pela dinâmica de sistemas é único para as duas abordagens. O modelo é pequeno e simples, justamente para servir de prova de conceito para a proposta. A Figura 4.4 apresenta em (a) o diagrama de influência do sistema modelado neste cenário. A variável Admissão influencia de forma positiva a variável Pessoas, ou seja, ao aumentar a Admissão, Pessoas também aumenta. Admissão também influi em Produtividade, mas desta vez de forma inversa. Ao aumentar a Admissão, a produtividade tende a cair, o que é similar ao que acontece em um ambiente real, já que com novos membros na equipe, os membros mais experientes devem treinar estes novos. A comunicação também irá aumentar, diminuindo a produtividade. Este é um modelo simplificado da Lei de Brooks. Outra variável é a Demissão, semelhante à Admissão, mas com efeito de remoção de membros da equipe. Produtividade influi na variável Software, aqui representando a construção da aplicação final. Quando Produtividade aumenta, há uma tendência de aumentar também as saídas em Software.

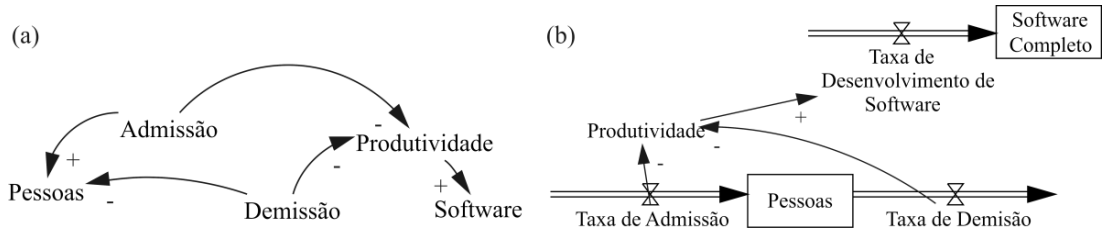


Figura 4.4. (a) Diagrama de Influências e (b) Diagrama de Estoques e Fluxos da Modelagem de Dinâmica de Sistemas da Prova de Conceito.
Fonte: Elaborada pelo Autor.

Ainda na Figura 4.4, mas em (b), é apresentado o diagrama de estoques e fluxos (modelado neste exemplo com a ferramenta Vensim), a ser utilizado nas simulações. Este diagrama é o modelo de dinâmica de sistemas que é simulado e onde é

possível extrair informações sobre as tendências do sistema, em especial ao analisar os estoques e seus fluxos. É possível, por exemplo, saber se o estoque Pessoas tende a aumentar, diminuir ou até mesmo se manter em equilíbrio.

4.3.1 Redes Neurais Artificiais

Etapa 1 – Configuração da Simulação

A configuração das variáveis é feita de acordo com um cenário fictício aqui apresentado para esta situação. O tempo de simulação é de 21 dias (três semanas), e os valores das variáveis configuradas para a simulação do ambiente estão resumidas na Tabela 4.3. Neste período, houve duas admissões e duas demissões. O resultado da simulação deste ambiente é apresentado na Figura 4.5, onde é possível observar que o estoque Pessoas se manteve estável, em equilíbrio (manteve os 10 membros na equipe), mas a função da variável produtividade está em declínio, apresentando um comportamento dinâmico de decrescimento.

Recurso	Valor
Estoque Pessoas	10
Taxa de Demissões	2
Taxa de Admissões	2
Tempo Simulado	21 dias

Tabela 4.3. Valores das Variáveis para a Simulação do Cenário Fictício.
Fonte: Elaborada pelo Autor.

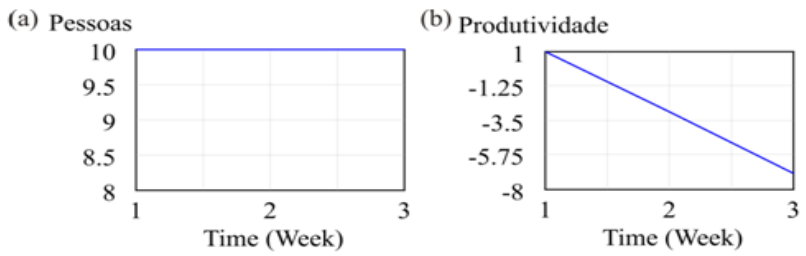


Figura 4.5. Resultado da Simulação do Modelo de Dinâmica de Sistemas. (a) Estoque Pessoas e (b) Variável Produtividade.
Fonte: Elaborada pelo Autor.

Etapa 2 – Mapeamento Dinâmica de Sistemas x Indícios

Recebendo o resultado da simulação feita na Etapa 1, esta etapa é responsável pelo mapeamento do modelo simulado para indícios. Utilizando a técnica descrita anteriormente, ocorre o mapeamento descrito na Tabela 4.4.

Recurso (DS)	Indício	Comportamento
Estoque Pessoas	Pessoas	Estável (10)
Taxa de Demissões	Demissão	Crescimento (2)
Taxa de Admissões	Admissão	Crescimento (2)
Produtividade	Produtividade	Decrescimento

Tabela 4.4. Valores das Variáveis para a Simulação do Cenário Fictício.
Fonte: Elaborada pelo Autor.

O estoque pessoas é mapeado para o indício pessoa e este apresenta o comportamento estável, de acordo com a simulação realizada. A taxa de demissões e Admissões foram mapeadas para os indícios Demissão e Admissão, respectivamente. Ambos com comportamento dinâmico de crescimento (houveram duas admissões e duas demissões). Já produtividade foi mapeada para o indício produtividade e seu comportamento é decrescimento.

Etapa 3 – Configuração / Definição dos Riscos

Nesta etapa são definidos os riscos que serão monitorados. Esta tarefa é executada pelo gestor e é feita com base na sua experiência de gerenciamento de projetos. Nesse cenário fictício, o objetivo é monitorar dois tipos de riscos (que o gestor fictício nesse contexto queira monitorar de acordo com o modelo): rotatividade de pessoal (turnover) e falta de produtividade. Na Tabela 4.5 são apresentados os riscos e seus respectivos conjuntos de indícios: X1, para quando o estoque Pessoas tende a aumentar e a Produtividade tende a diminuir, X2 quando Pessoas tende a se manter estável e Produtividade tente a diminuir, havendo neste caso admissões e demissões no ambiente e X3, para o caso do estoque Pessoas e a variável Produtividade tenderem a diminuir. O Risco 1 (rotatividade de pessoal) é mapeado para Y1 e o Risco 2 (Falta de

Produtividade) é mapeado para a variável Y2. Os indícios X1 e X3 são para o risco Y2 e o X2 para o risco Y1.

Indícios	Risco 1 – Rotatividade de Pessoal Y1	Risco 2 – Falta de Produtividade Y2
X1 – Pessoas tende a aumentar e produtividade tende a diminuir		X
X2 – Pessoas tende a se manter estável e produtividade tende a diminuir, havendo admissões e demissões	X	
X3 – Pessoas tente a diminuir e produtividade tende a diminuir		X

Tabela 4.5. Riscos e Seus Respectivos Indícios.
Fonte: Elaborada pelo Autor.

Com base nos dados da Tabela 4.5 (pode ser considerado o conjunto de exemplos) é feito o treinamento da rede para que ela extraia as características necessárias para representar uma futura informação recebida. Ou seja, a rede aprende por meio de exemplos e generaliza a informação aprendida nas respostas futuras, podendo identificar os riscos na Etapa 4.

Etapa 4 – Identificação dos Riscos

Nesta Etapa ocorre o treinamento da rede para que a rede possa inferir os resultados da simulação da dinâmica de sistemas. Para isso, é necessário representar os indícios e riscos, definidos pela Tabela 4.5, em números binários e estes estão representados na Tabela 4.6 (isso pode ser obtido com a fórmula $x <= 2^n$ – são 3 indícios, e para isso necessita de dois bits (00, 01 e 10 – "sobra" o 11) e são 2 riscos. Neste caso 1 bit é suficiente (0 e 1)). Para o treinamento, o *bias* possui valor arbitrado em 1 e para o primeiro passo do treinamento todos os pesos sinápticos são 0. O treinamento da rede foi feito em seis passos, de A a F, como apresentado pela Tabela 4.7.

Indício - Risco	Representação Binária
Indício X1	00
Indício X2	01
Indício X3	10
Risco Y0	0
Risco Y1	1

Tabela 4.6. Representação Binária de Indícios e Riscos.
Fonte: Elaborada pelo Autor.

Passos do Treinamento	*v*	*γ*	*E*	*b(bias)*	*w(1)*	*w(2)*
A	0	0	1	1	1	0
B	1	1	-1	0	1	-1
C	0	0	1	1	1	-1
D	2	1	0	1	1	-1
E	0	0	0	1	1	-1
F	1	1	0	1	1	-1

Tabela 4.7. Passos para o Treinamento da Rede Neural Artificial.
Fonte: Elaborada pelo Autor.

Para iniciar o treinamento (passo A), apresenta-se o sinal do indício X3 (10) para a rede (é arbitrada a primeira entrada, como em um sorteio), conforme pode ser visualizado na Figura 4.6(a). Calcula-se a função de ativação em (5) e a transferência em (6) com base nestes valores de entrada e o valor arbitrado do *bias*.

$$v = 0x1 + 0x1 + 0x1 = 0. \quad (5)$$

$$\gamma(v) = 0. \quad (6)$$

O valor da função de ativação (v) é zero. Em consequência disso, a função de transferência também é zero, conforme é possível visualizar na Tabela 4.7, na linha representativa do passo A do treinamento. A saída não está correta (função de transferência), já que pelos dados da Tabela 4.5, o sinal de entrada X3 deveria ter uma saída S2. A rede, neste primeiro passo, estabeleceu uma saída Y1. Calcula-se então a

taxa de erro da rede (7) e as correções para os pesos sinápticos do bias, w(1) e w(2), em (8)(9)(10), respectivamente.

$$E = 1 - 0 = 1. \quad (7)$$
$$b(bias) = 0 + 1 \times 1 \times 1 = 1. \quad (8)$$
$$w1 = 0 + 1 \times 1 \times 1 = 1. \quad (9)$$
$$w2 = 0 + 1 \times 1 \times 0 = 0. \quad (10)$$

Com os novos pesos, é apresentado um novo sinal de entrada, X2 (01), conforme visualizado na Figura 4.6(b). O mesmo processo de cálculo é feito e o resultado é apresentado na linha do passo B na Tabela 4.7. Novamente a saída não está correta, já que se esperava para a entrada X2, a saída Y1. No passo C, o sinal de entrada é o X1 (00), como apresentado pela Figura 4.6(c). Ainda a saída não está correta. Era esperada para a entrada X1, a saída Y2. O passo D inicia-se com o sinal de entrada X3 (10) apresentado na Figura 4.6(d). Desta vez a saída está correta, já que realmente era esperada uma saída Y2 para a entrada X3. No passo E – Figura 4.6(e) – e F – Figura 4.6(f) – o resultado esperado como saída da rede também está correto. No E, o sinal de entrada foi o X2 (01) e a saída foi definida com o sinal Y1. No passo F, o sinal foi o X1 (00) e sua saída foi a Y1. Estas últimas todas corretas.

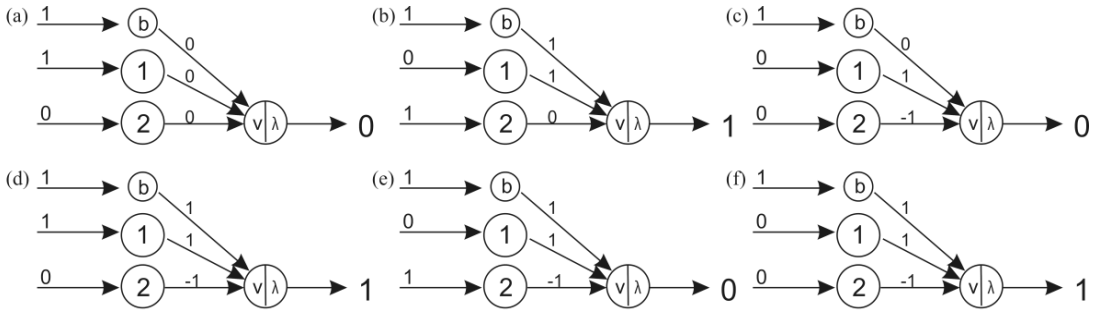


Figura 4.6. Processo de Treinamento da Rede Neural Artificial para o Exemplo da Prova de Conceito.
Fonte: Elaborada pelo Autor.

Desde o passo D, a rede não está mais errando. Ao se repetir o conjunto de treinamento novamente, encontra-se que o erro médio é zero. Isso significa que esta rede aprendeu a classificar estes indícios de problemas nos devidos riscos. Dessa forma, a utilização desta rede irá contribuir para a identificação do risco de acordo com a

classificação especificada. Cada conjunto de indícios e riscos merece um treinamento específico. Esse treinamento é específico para o treinamento desta rede para este problema apresentado. Em outros ambientes, a rede deve ser treinada com base nos dados do problema apresentado. É necessário apresentar uma entrada e fazer a aferição com o resultado desejado. Caso não seja correto o resultado, deve-se inferir novos testes até chegar em um conjunto de pesos sinápticos compatível para que a rede possua o menor índice de erro possível.

Com a rede treinada (pesos sinápticos b = 1, w1 = 1, e w2 = -1), ela está apta a exercer a função de classificação dos riscos em um ambiente real. No exemplo apresentado para esta prova de conceito, inclusive, foi obtida uma taxa de erro média igual a zero. Esse é o ideal, mas nem sempre é o obtido devido à complexidade da rede e qualidade dos dados de treinamento. O resultado da simulação do modelo de dinâmica de sistemas serve como entrada para a rede neural artificial. Como visualizado na Etapa 1, neste exemplo, a tendência do modelo é que o número de pessoas se mantenha o mesmo (ou seja, equilíbrio, estável – se mantém em 10 pessoas), e com dois funcionários sendo demitidos e outros dois sendo admitidos no período simulado. A produtividade mostrou uma tendência de decrescimento. Isso ocorre, pois os novos que entram na equipe devem ser treinados, ocupando tempo dos mais experientes. Em consequência, a produtividade diminui.

Esse ambiente simulado ativa as entradas definidas pelo indício X2 (o estoque Pessoas tende a se manter estável e Produtividade tende a diminuir, havendo admissões e demissões). Esse indício identificado, ativa a rede neural com os seguintes sinais de entrada (X2 – mapeamento binário em 01): bias = 1 (definido pelo treinamento), Entrada 1 = 0 e Entrada 2 = 1. Esses sinais ao serem processados, definem a função de transferência 0, que resulta na saída Y1, ou seja, indicativo de rotatividade de pessoal (*turnover*), conforme pode ser visualizado na Figura 4.7.

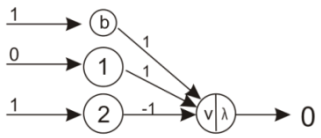


Figura 4.7. Rede Neural Artificial sendo Excitada pela Saída do Resultado da Simulação do Modelo de Dinâmica de Sistemas.
Fonte: Elaborada pelo Autor.

Desta forma a rede, ao receber a excitação dos resultados das simulações dos modelos de dinâmica de sistemas, identifica o risco de acordo com o mapeamento estabelecido pelo gestor. Neste exemplo, de prova de conceito, o ambiente simulado é limitado e com poucas variáveis. Entretanto, por se tratar de um modelo matemático baseado em lógica, ao serem adicionadas opções (leia-se neurônios para este modelo) o comportamento será o mesmo que o apresentado.

4.3.2 Regras de Produção

Etapa 1 – Configuração da Simulação

A configuração das variáveis é feita de acordo com um cenário fictício aqui apresentado para esta situação e é igual à prova de conceito definida para a abordagem utilizando Redes Neurais Artificiais no item 4.3.1 deste trabalho. Isso ocorre pois a Etapa 1 e a Etapa 2 são iguais para as duas abordagens apresentadas.

Etapa 2 – Mapeamento Dinâmica de Sistemas x Indícios

Igualmente como fora definido na Etapa 1, esta Etapa é igual à Etapa 2 da abordagem utilizando Redes Neurais Artificiais.

Etapa 3 – Configuração / Definição dos Riscos

Nesta Etapa, o gestor tem a possibilidade de definir e configurar os riscos que ele queira monitorar. Buscando uma isonomia com a prova de conceito utilizando redes neurais artificiais, os riscos apresentados são os mesmos: rotatividade de pessoal e falta de produtividade. Nesse sentido foram definidas as regras de produção utilizando a LIBR na Listagem 4.2 e 4.3.

Listagem 4.2. Regra de Produção para o Risco Turn-Over.
Fonte: Elaborada pelo Autor.

```
1.  SE
2.      Pessoas [ESTAVEL] AND
3.      Admissão >= 2 AND
4.      Demissão >= 2 AND
5.      Produtividade[DECRESCE]
6.  ENTÃO
7.      RISCO[TURNOVER]
```

Listagem 4.3. Regra de Produção para o Risco Falta de Produtividade.
Fonte: Elaborada pelo Autor.

```
1.  SE
2.      (Pessoas[CRESCIMENTO] AND
3.      Produtividade[DECRESCIMENTO]) OU
4.      (Pessoas[DECRESCIMENTO] AND
5.      Produtividade[DECRESCIMENTO])
6.  ENTÃO
7.      RISCO[FALTA DE PRODUTIVIDADE]
```

Nas Listagens apresentadas, é possível verificar a capacidade da LIBR para transformar o conhecimento do gestor em regras de produção para estabelecer os riscos. A utilização da linguagem utilizando regras de produção facilita o entendimento e as possibilidades. A possibilidade de atribuir limites de valores permite aumentar a destreza para identificação dos riscos de forma a aumentar o poder de personalização para definição e configuração de cada risco.

Etapa 4 – Identificação dos Riscos

Esta quarta Etapa consiste na inferência das regras de produção definidas na Etapa anterior com base nos resultados da simulação dos modelos de dinâmica de sistemas. Os resultados da simulação apresentam Pessoas estável e Produtividade decrescendo, com as taxas de admissão e demissão com o valor dois. Fazendo a inferência lógica é definido o resultado apresentado em (11) e (12) para os riscos rotatividade de pessoal e falta de produtividade, respectivamente.

$$(V \wedge V \wedge V \wedge V) \rightarrow V \quad (11)$$

$$((F \wedge V) \vee (F \wedge V)) \rightarrow F \quad (12)$$

Com os resultado, é possível identificar que em (11) é apresentado um resultado VERDADEIRO, ou seja, o risco turn-over é identificado. Isso, pois a sentença Pessoas [ESTÁVEL] é VERDADEIRA, ADMISSÃO E DEMISSÃO são maiores ou iguais que dois e por fim, PRODUTIVIDADE [DECRESCE] também é verdadeiro. Todas as sentenças verdadeiras sendo unidas pelo operador lógico E (AND), resulta em VERDADEIRO, identificando assim o risco TURNOVER. Já em (12) o resultado é FALSO, não identificando, portanto, o risco FALTA DE PRODUTIVIDADE.

4.4 Conclusões

Apesar de simples, o exemplo apresentado como prova de conceito pode avaliar a aplicabilidade da técnica descrita neste trabalho para monitoramento de riscos em projetos de software, utilizando os resultados da dinâmica de sistemas e algumas técnicas de inteligência computacional. Mesmo o trabalho tendo como foco a sua utilização em projetos de software, a técnica se mostra abrangente o suficiente para ser aplicada em qualquer área que a dinâmica de sistemas atua.

A primeira abordagem, utilizando redes neurais artificiais colaborou para a compreensão do problema, especialmente ao identificar que a tarefa de identificar os riscos diretamente pelas simulações de dinâmica de sistemas seria uma tarefa linearmente separável. Inicialmente, conjecturava-se que seria um problema de difícil aprendizado (não linear) e por isso a utilização de redes neurais foi o foco inicial do trabalho. Com os avanços e pesquisas, foi possível identificar que, apesar de totalmente funcional, a abordagem com rede neural poderia ser facilmente substituída por outra abordagem, com o mesmo resultado, mas com menor esforço. Foi neste momento que a utilização da lógica de *boole* foi avaliada até chegar na formalização das regras de produção e a LIBR.

5 SISTEMA DE APOIO A DECISÃO: FRAMEWORK PARA MONITORAMENTO DE RISCOS

Conforme já pode ser verificado, a proposta deste trabalho parte do principio de se utilizar a experiência do gestor no processo de definição e configuração dos riscos com base nos indícios. O objetivo desta abordagem é permitir que o conhecimento do gestor possa estar integrado no processo de monitoramento dos riscos e foi exatamente por isso que a proposta faz uso de técnicas de representação de conhecimento. Por conta disso, a definição de um sistema de apoio à decisão, mais especificamente um sistema de informação executivo, tem por objetivo permitir a implantação da proposta deste trabalho em uma aplicação que irá atuar como uma ferramenta de suporte gerencial.

Os sistemas de apoio à decisão podem ser baseados em dados, cujo foco principal é na obtenção, exibição de dados, extração de padrões a partir dos dados com base em técnicas de prospecção de dados (*datamining*), exibição de dados em gráficos, planilhas e tabelas, entre outros. Ou podem também ser baseados em modelos, em que modelos específicos de decisão, de pesquisa operacional ou ciências gerenciais, simulação, estatísticos, são utilizados como elemento central do sistema, permitindo a obtenção de informações a partir da execução desses modelos, empregando os dados disponíveis. Há ainda sistemas de apoio à decisão híbridos que fazem uso destas duas abordagens (TURBAN e ARONSON, 1998, apud BRAGA et al., 2004).

Neste capítulo são apresentadas as funções do SAD aqui definido e sua arquitetura. A especificação de software completa desta aplicação é apresentada em arquivos anexos ao trabalho, e consiste na modelagem do SAD utilizando as técnicas de modelagem orientada a objeto. São apresentadas neste capítulo as funções, as classes e o funcionamento interno das principais funcionalidades da aplicação. O nome do SAD é SoftEnRisk – Software Engineering Risk Analysis and Management Application.

5.1 Funções do Sistema de Apoio a Decisão

A Figura 5.1 apresenta o diagrama de casos de uso da aplicação onde são definidas as principais funcionalidade macro da aplicação. Os casos de uso definidos

são: gerir projeto, importar simulação, mapear automaticamente, importar simulação manualmente, gerir mapeamento, gerir indícios, gerir riscos e identificar riscos.

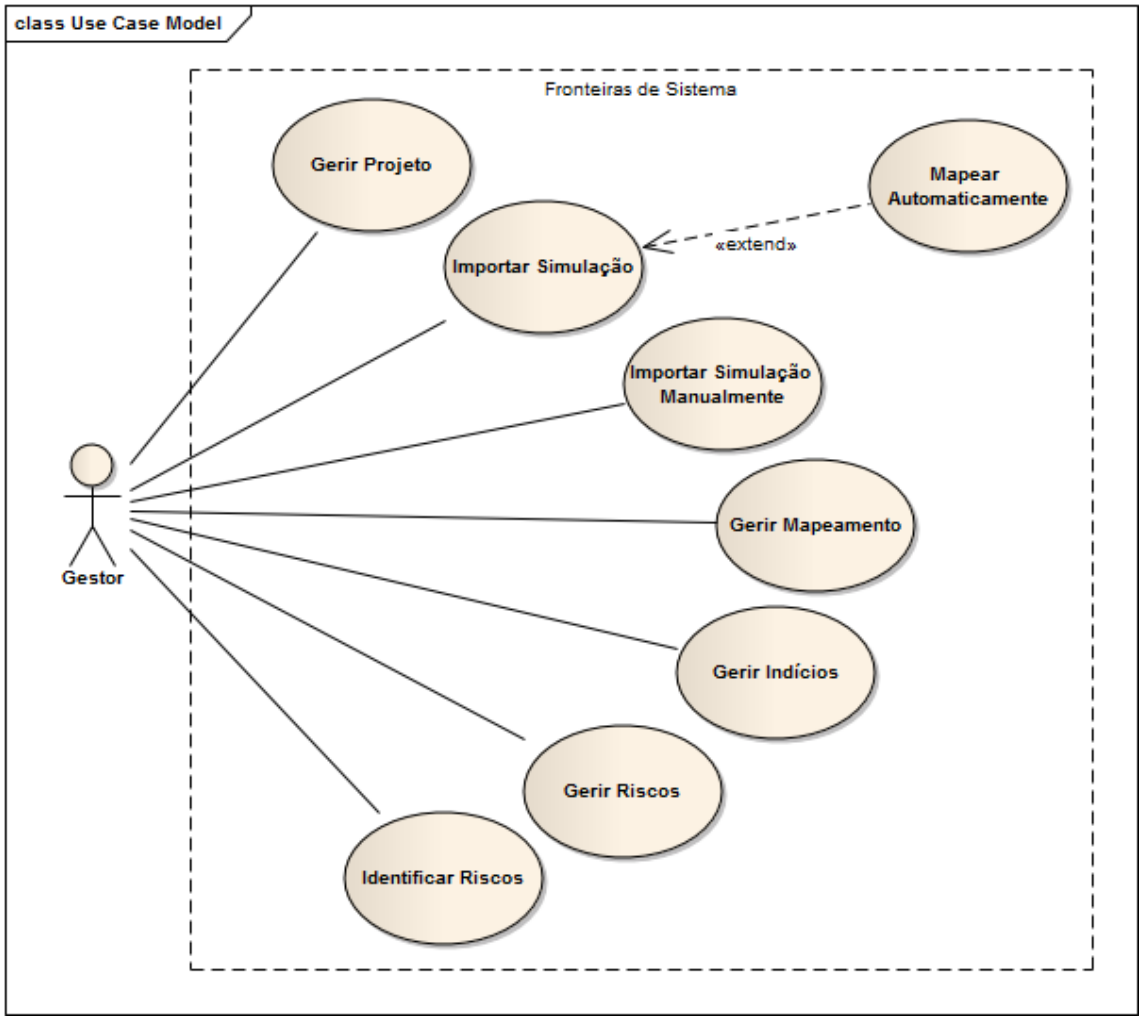


Figura 5.1. Diagrama de Casos de Uso do SAD.
Fonte: Elaborada pelo Autor.

O caso de uso gerir projeto se inicia quando o gestor que criar, editar, excluir ou alterar algum projeto que a ferramenta SAD irá atuar. Essa função existe para dar a possibilidade do gestor manipular vários projetos ao mesmo tempo na ferramenta. Cada projeto possui suas simulações, riscos e indícios. Ao iniciar a aplicação, o gestor escolhe o projeto para abrir e tem acesso aos dados específicos de cada projeto. Importar Simulação trata a funcionalidade de importar a simulação dos modelos da dinâmica de sistemas. Com bases nesta importação que o SAD irá identificar os riscos inerentes ao projeto. Há uma extensão desta função que permite ao gestor ao importar a simulação, fazer o mapeamento dos recursos da dinâmica de sistemas com os indícios de forma automática. Essa função, por ser automática, faz um mapeamento de todos os

recursos existentes no modelo de dinâmica de sistemas (todos os fluxos, variáveis e estoques) para indícios de riscos. Ela adiciona automaticamente os indícios mapeados nas respectivas bases.

Ainda sobre o caso de uso Importar Simulação, é importante destacar que cada ferramenta de dinâmica de sistemas possui uma saída diferente. Algumas possuem saída da simulação em XML, outras em arquivo texto, outras sequer possuem esta funcionalidade. No atual estágio de desenvolvimento do SAD, as ferramentas que são compatíveis com a funcionalidade de importação automática são a Vensim e a Sphinx. Vislumbrando esta limitação da aplicação, o caso de uso Importar Simulação Manualmente existe para permitir ao gestor importar manualmente os recursos e os resultados da dinâmica de sistemas gerados por qualquer ferramenta, mesmo as não suportadas pela função de importação automática. Na função manual, o gestor deve cadastrar recurso por recursos (fluxos, estoques e variáveis) e definir de acordo com o seu desejo o comportamento de cada um dos recursos estabelecidos pelos resultados da simulação do modelo (crescimento, decrescimento, oscilação ou estável).

O caso de uso Gerir Mapeamento trata a funcionalidade de incluir, excluir e atualizar os mapeamentos dos recursos da dinâmica de sistemas com os indícios. É possível nessa funcionalidade definir manualmente o comportamento dinâmico simulado pelo modelo de dinâmica de sistemas (somente caso seja necessária alguma alteração do resultado da simulação). O caso de uso Gerir Risco é o responsável por definir e configurar os riscos de acordo com os indícios. É nessa função que é permitido ao gestor definir as regras de produção de cada um dos riscos. Esse caso de uso trata as questões de inclusão, exclusão e atualização dos riscos e suas regras de produção. Por fim, o caso de uso Identificar Risco, é responsável por analisar o resultado da simulação e com base nisso realizar as inferências das regras de produção e determinar qual risco irá ser ativado. Essa função exibe os riscos identificados na tela principal da ferramenta de forma a facilitar a visualização dos riscos.

5.2 Arquitetura do SAD

O SoftEnRisk foi projetado observando as principais técnicas de desenvolvimento com padrões de projeto. Ele é parte de um projeto do grupo da UFV

que atua no estudo da dinâmica de sistemas aplicada a engenharia de software. O objetivo neste grupo é desenvolver um portal onde os dados de empresas de desenvolvimento de software são manipulados para identificar riscos nestes projetos em tempo real. Com o objetivo de permitir o aproveitamento do código fonte desenvolvimento no contexto de trabalho, alguns padrões foram utilizados. O primeiro padrão utilizado foi o desenvolvimento em camadas, estabelecido pelo padrão de arquitetura MVC. As três camadas são observadas do padrão MVC, com a adição de uma camada de acesso aos dados persistidos. Com isso o sistema possui quatro camadas: visão, controle, modelo e persistência. A camada de persistência é a implantação do padrão de projeto DAO (*data access object*) que encapsula todo o acesso aos dados e é responsável pelo mapeamento objeto relacional, já que um sistema gerenciador de banco de dados relacional é utilizado. Os padrões de projeto *singleton* e *abstract factory* também são utilizados.

A Figura 5.2 exibe o diagrama de classes, suprimindo os atributos e métodos, com a visão de modelo do SAD. As classes são dispostas para contemplar a implementação da proposta deste trabalho. São classes de modelo do SAD: projeto, modelosimulacao, fluxo, estoque, variável, indício, comportamento e risco. Um projeto pode possuir zero ou mais modelos de simulação (modelosimulacao). Cada modelo de simulação é de um único projeto e pode possuir zero ou mais variáveis, estoques e/ou fluxos. Estes por sua vez, vão instanciar de um único modelo de simulação. As classes fluxo, estoque e variável representam os recursos de dinâmica de sistemas importados ao SAD. Estes são mapeados em indícios, mas não de forma obrigatória, por isso a associação do tipo zero ou um. E cada indício deve ter a associação com seu respectivo recurso. Cada indício possui um comportamento associado, da classe comportamento (uma classe de enumeração: crescimento, decrescimento, oscilação e estável). Um indício pode ter um comportamento e um comportamento pode estar associado a zero ou mais indícios. Um indício pode servir de definição para zero ou mais riscos e um risco pode estar associado a zero ou mais indícios também. Esta associação é de suma importância, pois ela é a responsável por definir as regras de produção dos riscos. A classe risco ainda possui uma associação com a classe modelosimulacao para gerar dados estatísticos de quantos riscos possui em um projeto de forma mais fácil.

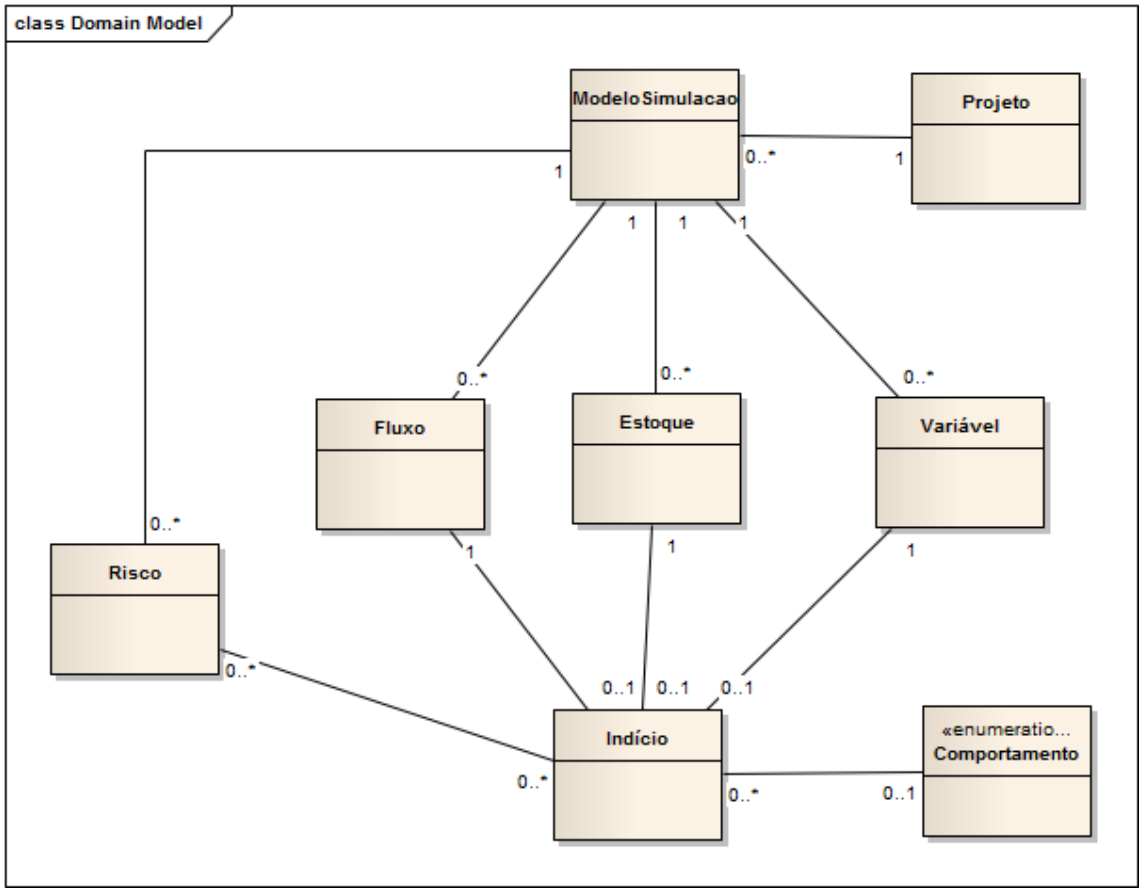


Figura 5.2. Diagrama de Classe - Modelo.
Fonte: Elaborada pelo Autor.

Já a Figura 5.3, apresenta o diagrama de classes, representando a arquitetura MVC e definindo as classes controladoras da aplicação. As classes de modelo e de visão são apresentadas como pacotes, bem como as classes de persistência. A estratégia utilizada para as classes de controle é definir um controlador por caso de uso. A classe cntrProjeto controla as interações entre a interface e o modelo relacionadas às funções de projeto. A cntrImportarSimulacao as funções relacionadas à importação automática e manual, inclusive de auto mapeamento. A classe cntrMapeamento é a controladora da funcionalidade de gerir mapeamento. Já a cntrIndicio controla as interação referentes ao caso de uso de gerir indícios. A funcionalidade de gestão de riscos é responsabilidade da controladora cntrRisco e por fim, a cntrIdentificarRisco atua no controle das funções de Identificar riscos.

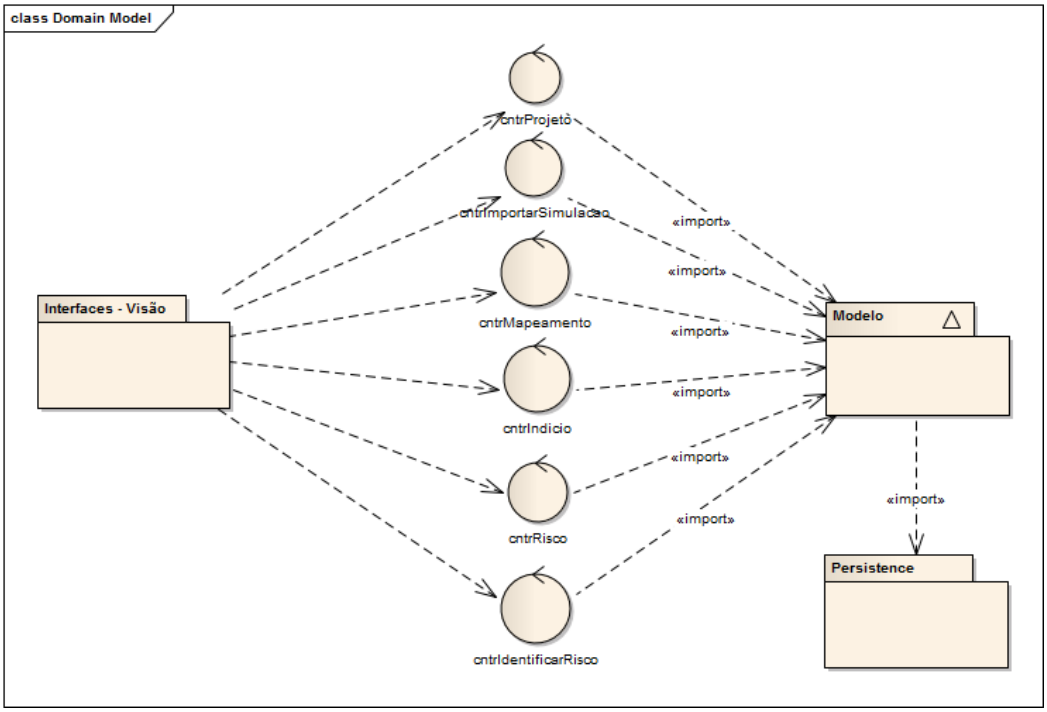


Figura 5.3. Diagrama de Classe – MVC - Controladoras.
Fonte: Elaborada pelo Autor.

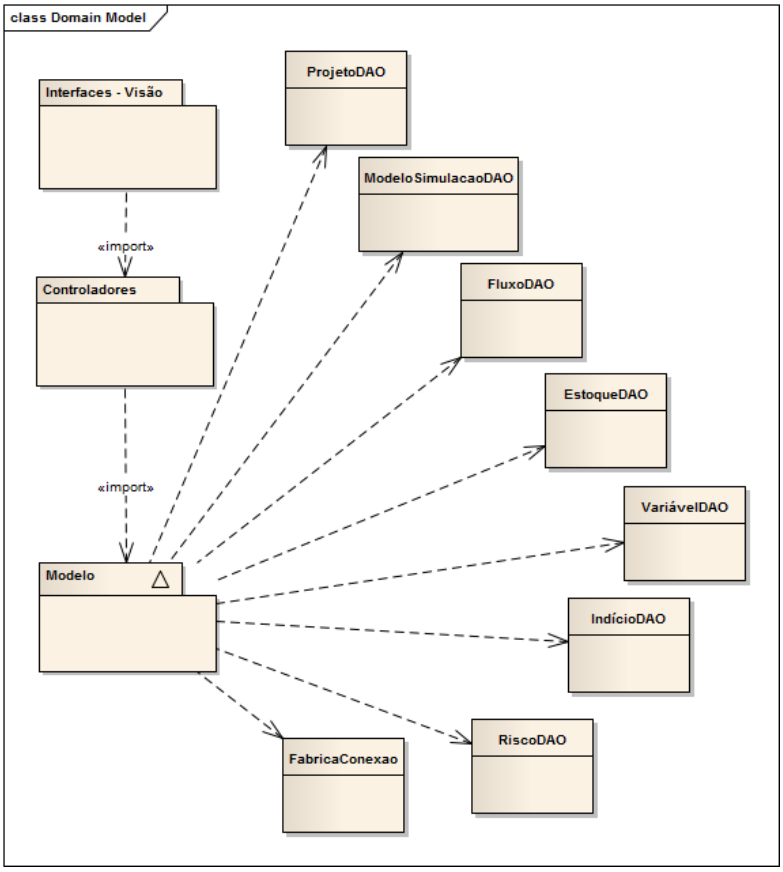


Figura 5.4. Diagrama de Classe – MVC - Persistência.
Fonte: Elaborada pelo Autor.

Já a Figura 5.4, apresenta o mesmo modelo de diagrama, em camadas, mas com especificação das classes de persistência. Estas representam o padrão de projeto DAO (*data access object*) e é responsável pelo mapeamento objeto relacional das classes do modelo, a manipulação dos dados diretamente no banco e conexão com o banco. Este último, responsabilidade da classe FabricaConexao (esta classe representa o padrão de projeto *factory* e *singleton*).

5.3 Projeto de Interfaces

A Figura 5.5 apresenta o diagrama de navegação entre as interfaces propostas do SAD. A interface Principal é o acesso a todas as funções do software e por ela que o usuário inicia o acesso ao software. E é somente por esta interface que ele pode fechar a aplicação. Há a interface de projeto, onde é possível criar, editar e excluir projetos, além de permitir o carregamento de algum projeto já cadastrado. As interfaces Importar Automático e Importar Manual são responsáveis pelas importações das simulações dos modelos de dinâmica de sistemas. Já Mapeamento é responsável por estabelecer ou alterar o mapeamento do projeto vigente de acordo com os recursos da dinâmica de sistemas importados. A interface Indícios permite a visualização dos indícios mapeados de forma isolada. Nela é possível também alterar o comportamento do indício importado da simulação. Já a interface de Riscos permite ao gestor a criação, alteração e exclusão de riscos. No caso de alterar e solicitar a adição de um novo risco é aberta uma nova interface, chamada Novo Risco – Configuração onde é possível definir as regras de produção do risco, bem como alterar as suas propriedades, como o nome dele. E por fim, existe a interface Sobre, onde são apresentadas algumas informações sobre o software. A Figura 5.6 exibe o protótipo da interface Principal da ferramenta, que além de permitir o acesso às demais opções, também permite a visualização dos riscos identificados pela abordagem deste trabalho.

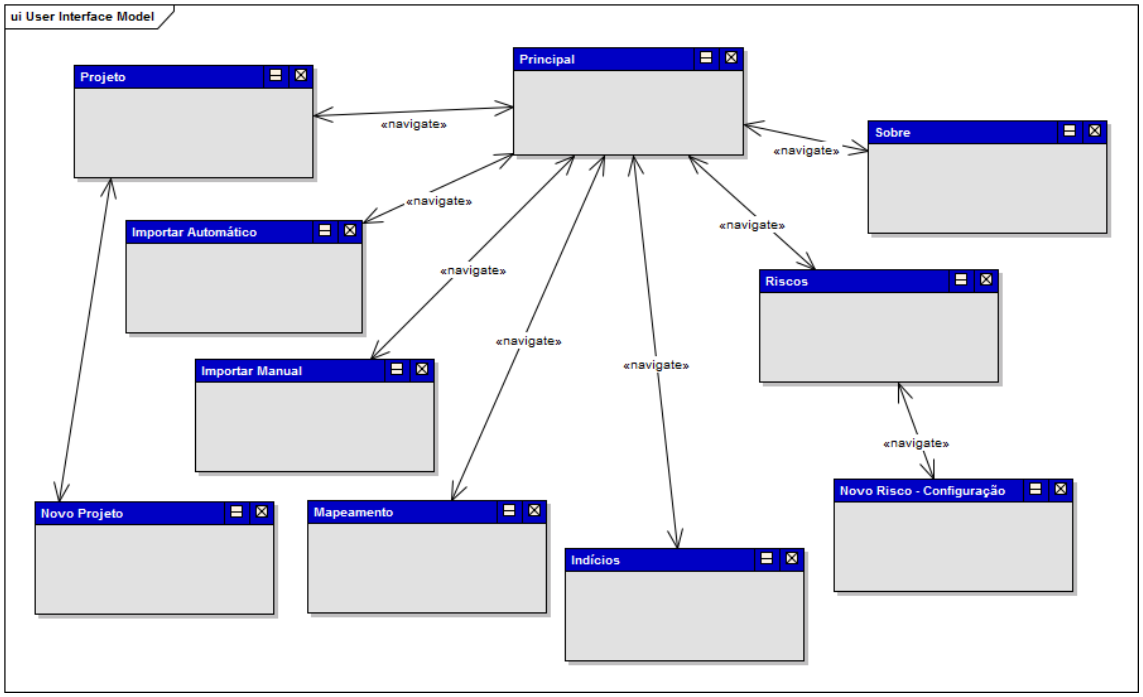


Figura 5.5. Diagrama de Navegabilidade das Interfaces.
Fonte: Elaborada pelo Autor.

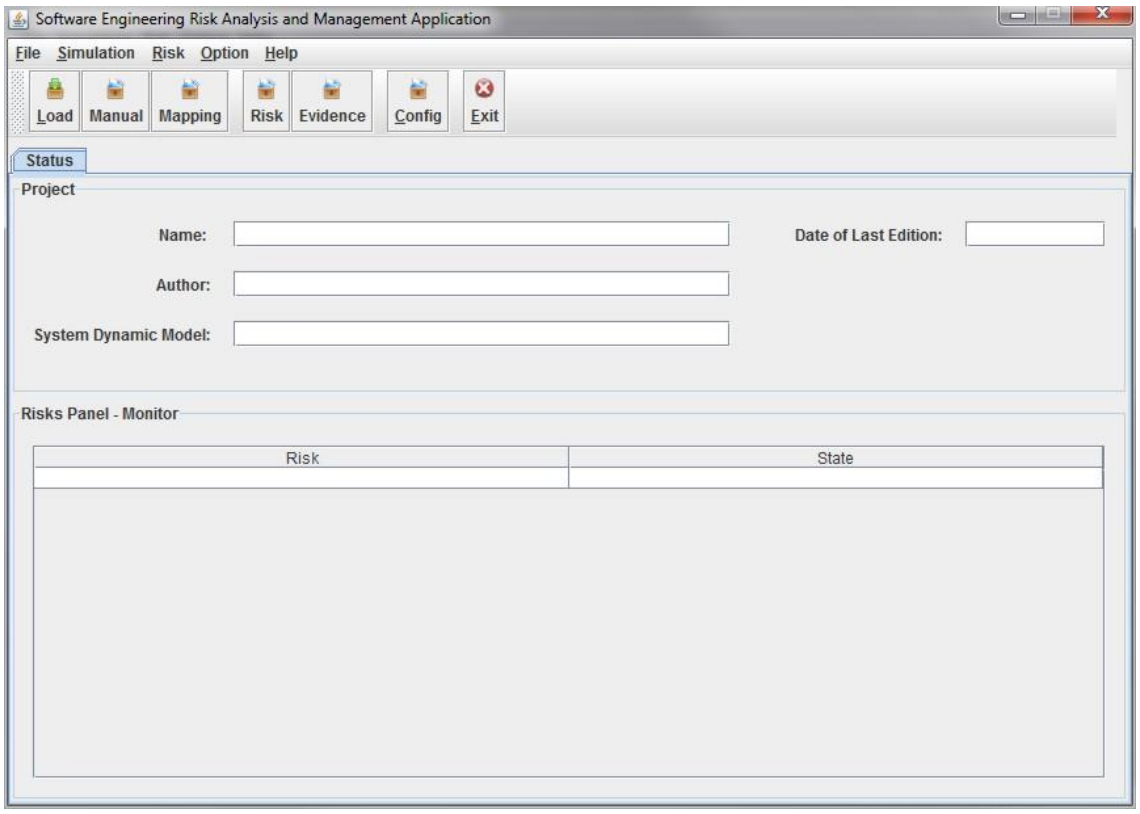


Figura 5.6. Protótipo da Interface Principal.
Fonte: Elaborada pelo Autor.

5.4 Desenvolvimento

Desde a fase de especificação do software, foram utilizadas ferramentas para gestão de manutenção e configuração do software, como por exemplo a utilização de um servidor de versionamento. No contexto deste desenvolvimento foi utilizada a ferramenta SVN (*subversion*). Foi mantido o *baseline* tanto da documentação, quanto do código fonte. A linguagem escolhida para o desenvolvimento foi JAVA, uma vez que já estava definida em um dos requisitos não funcionais esta necessidade. A utilização da linguagem JAVA era necessária pois os códigos deste projeto irão ser reaproveitados no contexto do grupo de estudo da UFV, como já apresentado. Os projetos já desenvolvidos no grupo de estudo estão utilizando JAVA.

O banco de dados utilizado pelo SoftEnRisk é o MySQL e para esta escolha não há justificativas. Entretanto, com a arquitetura que foi desenvolvido o software a troca do banco de dados é simples. Basta alterar a classe de conexão para acessar outro banco e outra base. Nenhum acoplamento foi feito no banco como utilização de gatilhos (*triggers*) e procedimento armazenados (*stored procedures*). Foi utilizando SQL ANSI (padrão internacional para consultas – SQL – *Structured Query Language*) para as consultas.

Para demonstrar o funcionamento do software e seu projeto interno, é apresentado na Figura 5.7 o diagrama de sequencia da funcionalidade de identificar risco.

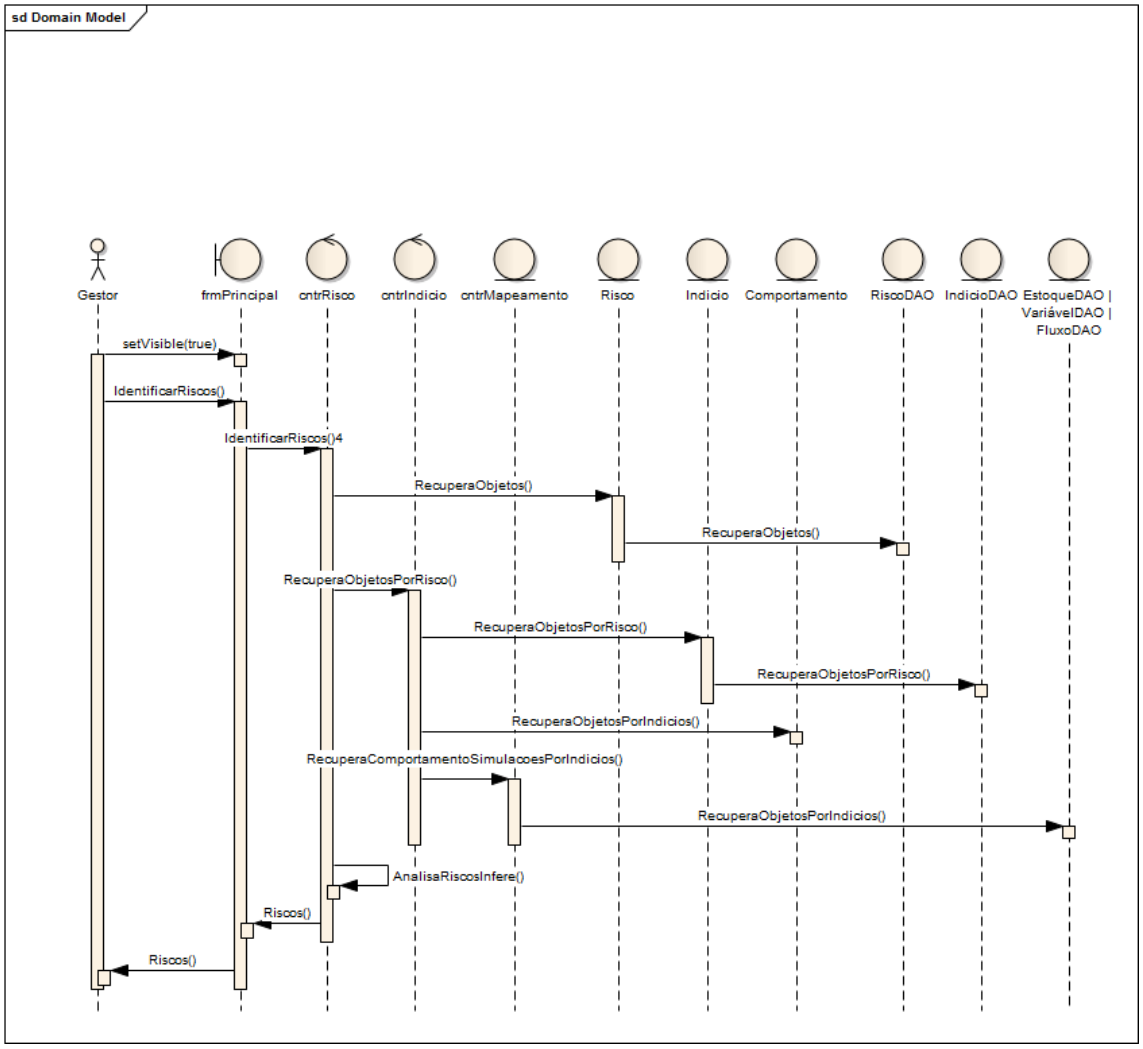


Figura 5.7. Diagrama de Sequencia da Função de Identificar Riscos.
Fonte: Elaborada pelo Autor.

A ação se inicia quando o gestor solicita a abertura da interface principal. Imediatamente após esta ação, é iniciada a tarefa de identificar riscos com a chamada ao método IdentificarRiscos do frmPrincipal. Este método faz um acesso à classe controladora de Riscos cntrRiscos solicitando todos os objetos risco do projeto (RecuperaObjetos()). A controladora, por sua vez, acessa a camada de modelo e faz a mesma solicitação e esta passa à camada de persistência. As mensagens de retorno do diagrama de sequencia foram omitidas para minimizar a complexidade do diagrama. Assim que todos os objetos da classe Risco chegam na controladora de riscos, camada que havia feito a solicitação, ela solicita à controladora de indícios (cntrIndicios) todos os objetos associados aos riscos objetivos. Para isso, a classe faz acesso ao método ReperaObjetosPorIndicios da classe controladora de indícios. Esta passa a instrução à

classe de modelo, e esta para a classe de persistência que faz acesso à base de dados e devolve os registros em um vetor de instancias de objetos. Quando este vetor chega de retorno à classe cntrIndicios, esta o varre e para cada instância solicita o comportamento dinâmico a ela associado. Isso acontece na chamada RecuperaObjetosPorIndicios() para a classe Comportamento.

Após os comportamentos estarem carregados para cada indícios, a classe cntrIndicios necessita do comportamento das simulações para cada recurso da dinâmica de sistemas que foram antes importados. O método que realiza esta operação é o método RecuperaObjetosPorIndícios() que deve ser chamado das classes Estoque, Fluxo ou Variável, todas DAO da camada de persistência. De posse destas informações, a classe controladora dos indícios, a cntrIndicios, monta um vetor de objetos com todas essas informações e passa para a controladora dos riscos. Esta invoca o método AnalisaRiscosInfere que faz a aferição lógica de acordo com as regras de inferência de cada risco configurado para identificar algum risco. Para cada risco identificado, o mesmo é marcado no objeto e é gerado um DTO (*data transfer object*) com todos os riscos (identificados ou não) do projeto. Este DTO é enviado à interface que apresenta os riscos ao usuário.

6 ESTUDOS DE CASO

6.1 Introdução

O estudo de caso está divido em duas partes. A primeira parte faz uma análise de utilização da abordagem proposta por este trabalho em um projeto de software, utilizando parcialmente o modelo de dinâmica de sistemas proposto por Ambrósio (2008). Este modelo faz alusão à etapa de levantamento de requisitos em um projeto de software. Já a segunda parte é analisada a proposta em um modelo da área da economia, mais especificamente utilizando um modelo que analisa o ambiente da previdência social – aposentadoria no Brasil.

O objetivo ao exemplificar a utilização da proposta deste trabalho em duas abordagens totalmente diferentes é mostrar sua capilaridade e aplicabilidade não somente na área de software, mas também em qualquer outra área, demonstrando assim a generalidade da proposta.

O desenvolvimento do estudo de caso irá seguir o arcabouço da proposta deste trabalho. Ou seja, a arquitetura irá ser utilizada para apresentar cada uma das etapas da proposta com base nos modelos simulados. Todas as 4 etapas serão racionalizadas nas próximas seções: Etapa 1 – Configuração da Simulação, Etapa 2 – Mapeamento Dinâmica de Sistemas x Indícios, Etapa 3 – Configuração / Definição dos Riscos e por fim, a Etapa 4 – Identificação dos Riscos.

6.2 Projeto de Software

Ambrósio (2008), em seu trabalho, propôs um modelo de dinâmica de sistemas para a etapa de levantamento de requisitos em um projeto de software. Nele são avaliadas diversas variáveis que constituem o ambiente de levantamento de requisitos e fatores que avaliam o desenvolvimento da atividade. Para este estudo de caso foi extraído uma parte deste modelo que apresenta a definição de busca de qualidade na especificação dos requisitos com base na avaliação da quantidade de requisitos especificados com erro. A Figura 6.1 apresenta esta parte do modelo mencionada.

O modelo apresenta algumas variáveis que são variáveis complexas, ou seja, sua definição faz parte de outro modelo de dinâmica de sistemas. Mas para o entendimento deste estudo de caso, a representatividade do nome já é suficiente. O modelo define uma variável taxa de especificação, que representa os requisitos ainda não especificados, os homens-dia alocados para especificarem novos requisitos e a quantidade de requisitos especificados por homem-dia. Esta taxa de especificação impacta tanto a taxa de erros quanto a taxa de especificação correta. Ou seja, sempre que for especificado um novo requisito ou este está correto, ou está com erro.

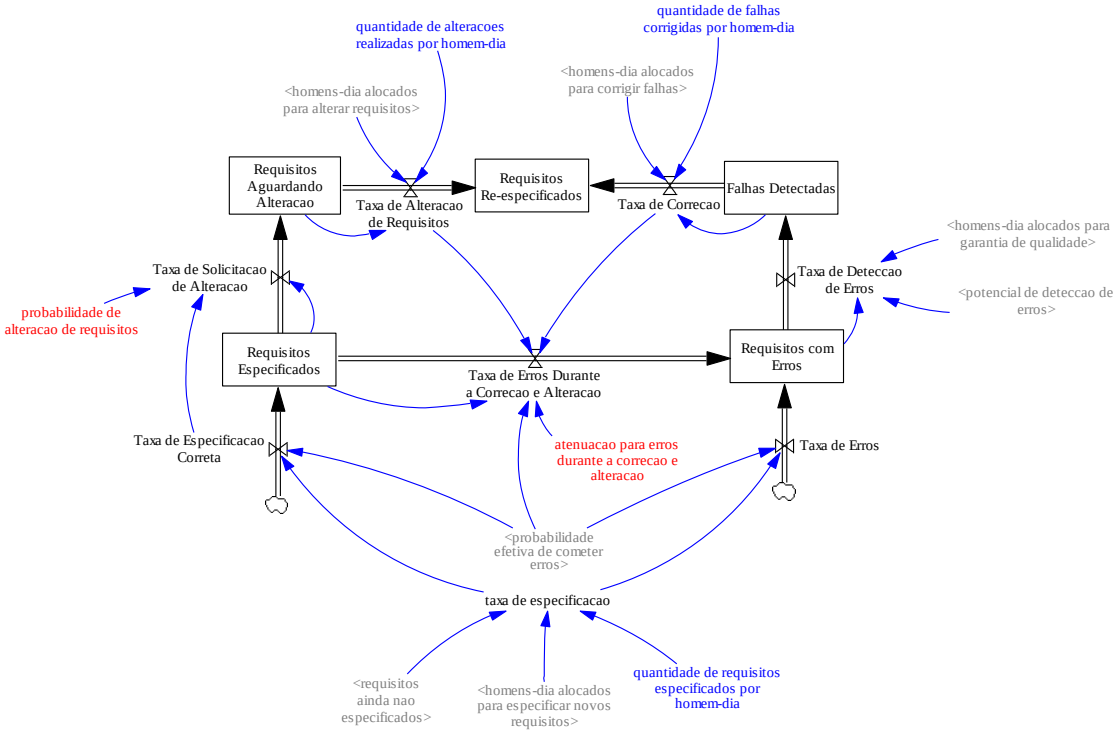


Figura 6.1. Modelo de Dinâmica de Sistemas – Requisitos de Software.
Fonte: Adaptada de (ABRÓSIO, 2008).

A taxa de erros, que é um fluxo que alimenta o estoque requisitos com erros ainda é influenciada pela variável probabilidade efetiva de cometer erros. O estoque requisitos com erros é ainda alimentado pela taxa de erros durante a correção e alteração dos requisitos, uma vez que em alguma alteração ou até mesmo correção, pode causar / gerar requisitos [reespecificados] com erros. O estoque requisitos com erros é drenado pela taxa de detecção de erros, que por sua vez alimenta um estoque falhas detectadas. Esta taxa de detecção de erros é mantida pelas variáveis homens-dia alocados para garantia de qualidade e potencial de detecção de erros da equipe.

O estoque falhas detectadas possui o fluxo taxa de correção como saída, alimentando o estoque requisitos reespecificados. O fluxo taxa de correção é definido pelas variáveis homens-dia alocados para corrigir falhas e quantidade de falhas corrigidas por homem-dia. O estoque requisitos reespecificados é ainda alimentado pelo estoque requisitos alterados, através da taxa de alteração de requisitos, este definido pelas variáveis homens-dia alocados para alterar requisitos e quantidade de alterações realizadas por homem-dia. O fluxo taxa de alteração de requisitos é a saída do estoque requisitos aguardando alteração que é alimentado pela taxa de solicitação de alteração, este saída do estoque requisitos especificados já citado. A taxa de solicitação de alteração é influenciada pela variável probabilidade de alteração de requisitos no software.

Outras variáveis são utilizadas para simular o modelo, especialmente as variáveis de configuração do ambiente simulado e entre elas destacam-se (para a compreensão da simulação neste estudo de caso): quantidade de requisitos especificados e entregues em uma liberação do software, produtividade médias dos integrantes da equipe, tamanho inicial da equipe responsável pela especificação, probabilidade de cometer erros durante a especificação dos requisitos, probabilidade de alteração de requisitos, aumento no esforço disponibilizado pelos membros da equipe quando há pressões de prazo e risco de atraso, entre outros. Estas variáveis fazem parte do modelo completo de Ambrósio (2008).

Com o modelo apresentado, nas próximas seções são apresentadas as etapas da proposta deste trabalho e o desenvolvimento desta tarefa.

Etapa 1 – Configuração da Simulação

A configuração da simulação irá seguir, de acordo com a proposta de Ambrósio (2008), as bases literárias para as principais variáveis. Ou seja, as configurações iniciais do modelo foram extraídas com base no levantamento e exploração bibliográfica sobre o tema. Na Tabela 6.1 é possível identificar o valor padrão para estas variáveis. Estas serão ajustadas, posteriormente, para simular um ambiente adverso onde um risco irá ser observado e monitorado pelo software SoftEnRisk.

As variáveis 1 a 8 possuem o seu valor definido em teorias definidas na literatura, e as variáveis 9, 10 e 11 possuem valores relativos a um ambiente fictício simulado.

Variáveis da Simulação	Valor Padrão
1 – Quantidade de requisitos especificados e entregues em uma liberação	120 requisitos estimados em 120 pontos de função
2 – Produtividade média dos integrantes da equipe	2 pontos de função / homem-dia
3 – Tamanho inicial da equipe responsável pela especificação	2
4 – Probabilidade de cometer erros durante a especificação dos requisitos	12% ou 0.12
5 – Probabilidade de alteração de requisitos	3% ou 0.03
6 – Porcentagem dos requisitos de uma liberação que não são previstos antes de iniciar a especificação	0% - todos os requisitos são previsto
7 – Porcentagem do esforço disponibilizado pela equipe que é alocado para as atividades de garantia de qualidade	20% ou 0.20
8 – Aumento no esforço disponibilizado pelos membros da equipe quando há pressões de prazo e risco de atraso	50% - até 50% a mais de esforço ou 0.50
9 – Dias simulados	15
10 – Quantidade de erros descobertos por dia	1
11 – Percentual de membros novatos na equipe	50% ou 0.5

Tabela 6.1. Valores Padrão das Variáveis do Modelo para Simulação.
Fonte: Adaptada de (AMBRÓSIO, 2008).

A Figura 6.2 apresenta o resultado da simulação, onde é possível observar que 15 dias foram simulados, 64.97 foram os requisitos especificados, destes 6.735 com erros. 0.1393 foram os requisitos aguardando alteração, com 0.5173 falhas detectadas, tendo 10.50 requisitos reespecificados.

Outra simulação é feita, agora ajustando os valores das variáveis dois e sete. A variável dois mede a produtividade média dos integrantes da equipe, que agora passa a ter valor 1 ponto de função / homem-dia. E a variável sete apresenta a percentagem do esforço disponibilizado pela equipe que é alocado para as atividades de garantia de qualidade que teve seu valor alterado para 0.05, ou seja 5%. Esta simulação busca simular um ambiente onde a produtividade da equipe caiu e a quantidade / esforço da equipe alocada para buscar garantir qualidade cai. Ambiente muito comum de acontecer em períodos de pressão para entrega em equipes de projeto de software (AMBRÓSIO, 2008). O resultado desta simulação pode ser observado na Figura 6.3.

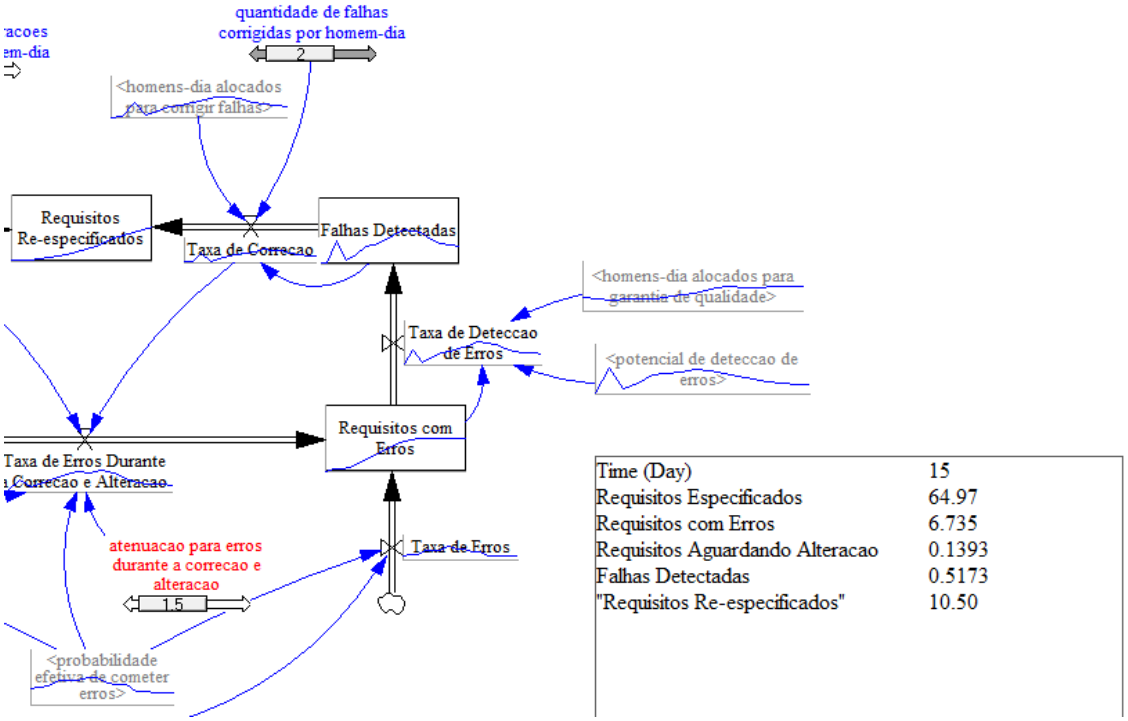


Figura 6.2. Resultado da Simulação – Valores Padrão.
Fonte: Elaborada pelo Autor.

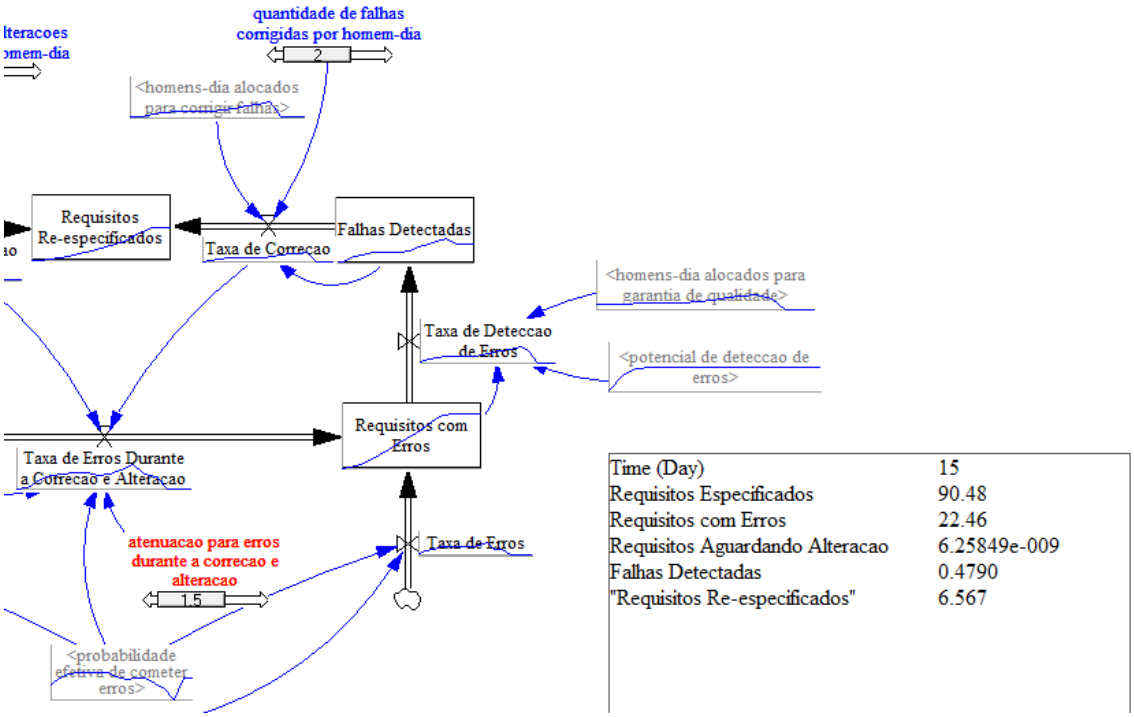


Figura 6.3. Resultado da Simulação – Variáveis Ajustadas.
Fonte: Elaborada pelo Autor.

O resultado apresentado na Figura 6.3 é representativo com base nas alterações feitas na simulação. O tempo simulado continua os 15 dias, entretanto a

quantidade de requisitos especificados aumentou para 90.48. Isso por que o tempo gasto com a busca pela garantia de qualidade passou de 20% para 5%, sobrando tempo para que o trabalho de especificação de requisito aumentasse. Entretanto, com pouco esforço para a garantia de qualidade, os requisitos com erros aumentaram para 22.46. A quantidade de falhas detectadas e de requisitos aguardando alteração diminuíram, bem como requisitos reespecificados.

Etapa 2 – Mapeamento Dinâmica de Sistemas x Indícios

Esta etapa já é realizada dentro da ferramenta e por este motivo, esta etapa e as próximas etapas as configurações são apresentadas com interfaces do SoftEnRisk em funcionamento. A Figura 6.4 exibe a interface principal da ferramenta com o projeto Estudo de Caso configurado e pronto para as configurações.

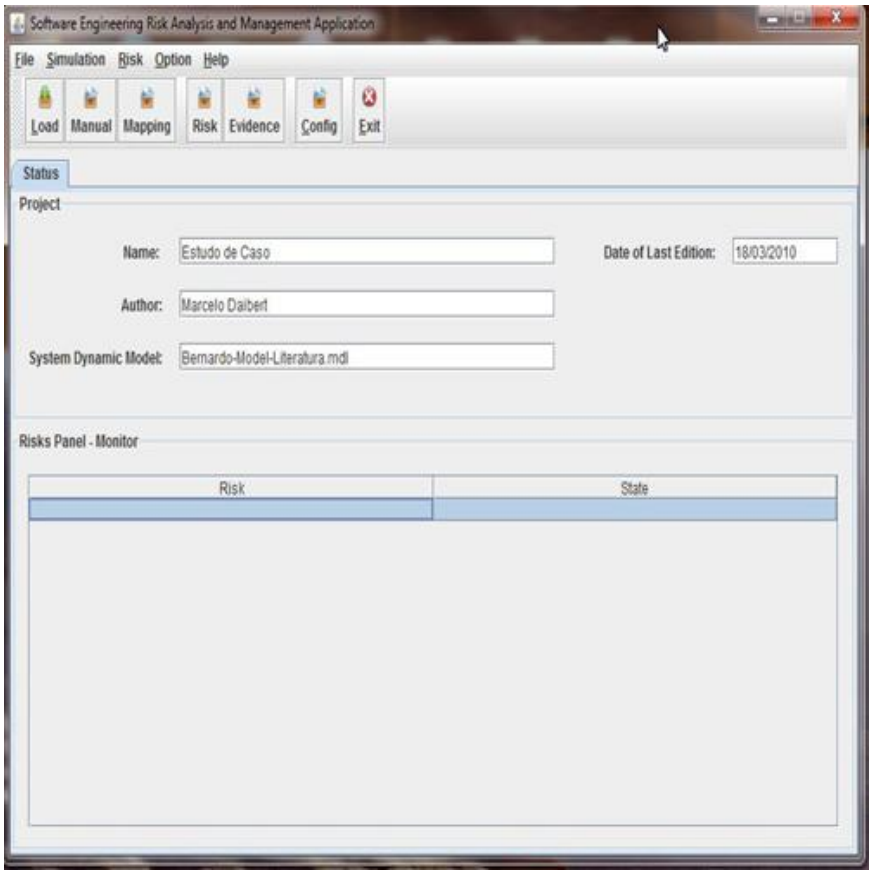


Figura 6.4. Interface do SoftEnRisk Configurada com o Projeto Estudo de Caso.
Fonte: Elaborada pelo Autor.

A Figura 6.5 apresenta a interface de importação automática dos resultados da simulação feitas na etapa 1. Neste caso a simulação está sendo importada da

ferramenta Vensim e o mapeamento automático estará ativo para que esta funcionalidade seja chamada pelo sistema. Como definido na ferramenta e na sua especificação no Anexo A deste trabalho, o mapeamento automático define que todo elemento importado do modelo é mapeado em indício dentro do framework de monitoramento de risco definido neste trabalho.

Após a importação dos resultados da simulação, o mapeamento automático é exibido na interface da ferramenta, onde é possível personalizar conforme a necessidade do gestor para a configuração do risco na próxima etapa, como é possível verificar na Figura 6.6.

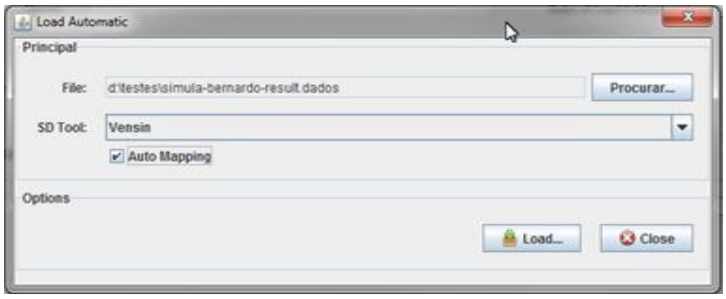


Figura 6.5. Importação do Resultado da Simulação.
Fonte: Elaborada pelo Autor.

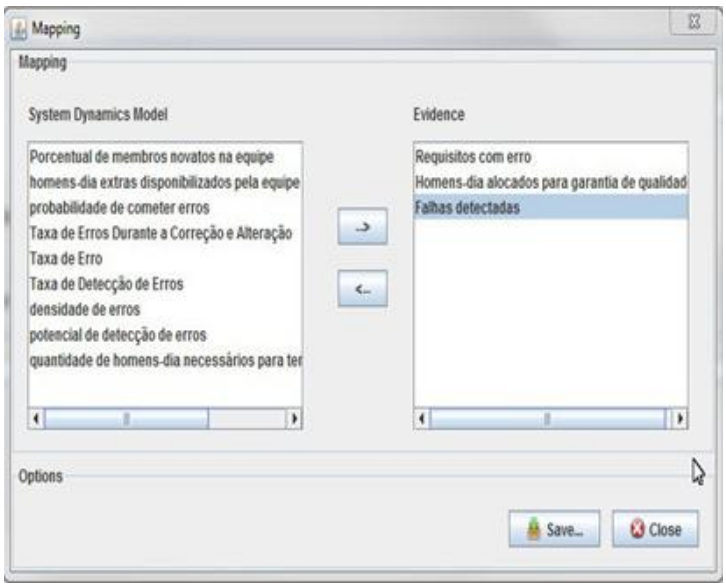


Figura 6.6. Mapeamento Dinâmica de Sistemas x Indícios.
Fonte: Elaborada pelo Autor.

Etapa 3 – Configuração / Definição dos Riscos

Com o mapeamento feito, a etapa de configuração / definição dos riscos é permitida. Nesta etapa o gestor irá configurar o risco com base na linguagem de inferência booleana de riscos, ou simplesmente a LIBR, definida neste trabalho. Esta linguagem é baseada, como já apresentado, nas teorias de regra de produção e na teoria de Boole – lógica booleana.

Esta configuração é feita na interface de definição de novo risco na ferramenta SoftEnRisk, como visualizado na Figura 6.7. Na oportunidade deste estudo de caso o risco Falta de Qualidade é definido com a expressão apresentada na Listagem 6.1.

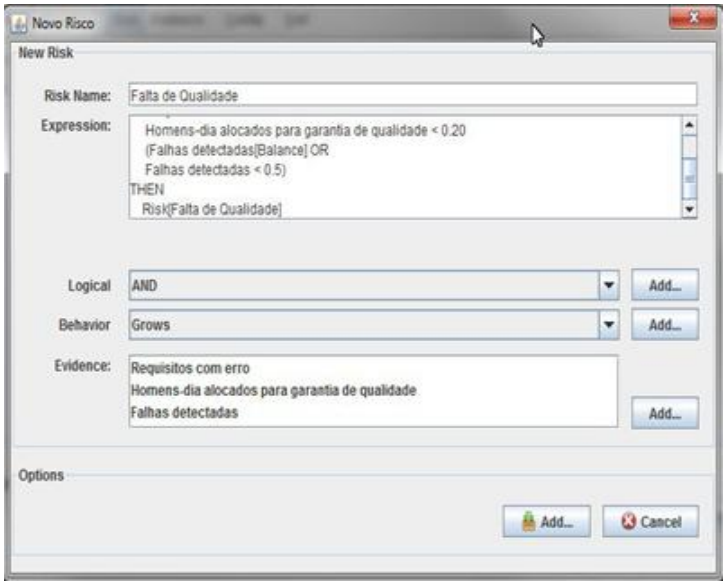


Figura 6.7. Definição de um Novo Risco.
Fonte: Elaborada pelo Autor.

Listagem 6.1. Expressão para o Risco Falta de Qualidade da Equipe Simulada.
Fonte: Elaborada pelo Autor.

```
1.  IF
2.        Requisitos com erro[Grows] AND
3.        Requisitos com erro > 15 AND
4.        Homens-dia alocados para garantia de qualidade < 0.20 AND
5.        (Falhas detectadas[Balance] OR
6.         Falhas detectadas < 0.5)
7.  THEN
8.        Risk[Falta de Qualidade]
9.
```

O risco novo configurado é Falta de Qualidade, e tem sua definição quando os requisitos com erro está em crescente E os requisitos com erro estão acima de 15 E

Homens-dia alocados para garantia de qualidade é menor que 0.20 (20%) E falhas detectadas está estável ou menor que 0.5, ou seja, 50%. Quando este ambiente ocorrer e for identificado pela simulação, há probabilidade de ocorrer o problema de falta de qualidade no produto especificado.

Etapa 4 – Identificação dos Riscos

Na etapa de identificação de riscos, cada expressão e inferência lógica da definição do risco são comparadas com o comportamento e valores dos indícios da dinâmica de sistemas anteriormente importados. Este resultado é exibido na interface principal da ferramenta, onde são listados todos os riscos configurados e é exibida a possibilidade em percentual de ocorrência do risco. Esta calculada com base na quantidade de inferências verdadeiras na expressão quando comparadas com os resultados da simulação.

Com o risco Falta de Qualidade configurado no ambiente já importado, o resultado é apresentado na Figura 6.8.

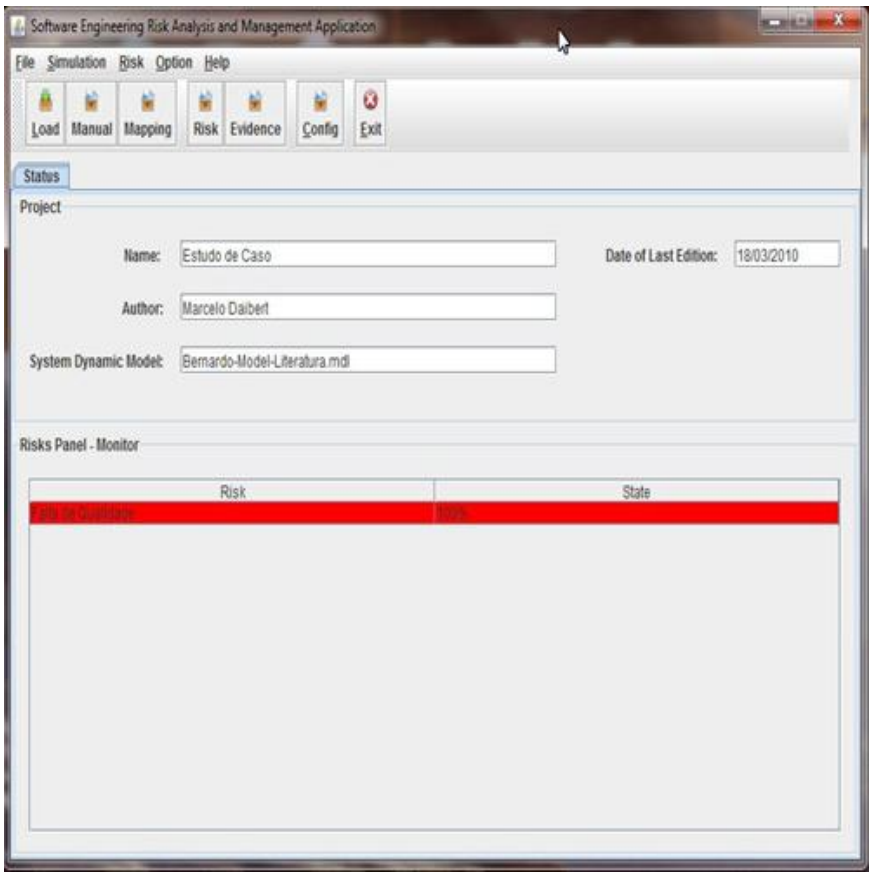


Figura 6.8. Resultado do Monitoramento de Riscos no Projeto Estudo de Caso.
Fonte: Elaborada pelo Autor.

Na Figura 6.8 é possível observar que o risco Falta de Qualidade foi classificado como crítico (cor vermelha) com possibilidade de ocorrência de 100%. Isso por que, com base nos resultados da simulação o estoque requisitos com erro está em crescimento, existem 22.46 requisitos com erro (ou seja, mais de 15), homens-dia alocados para garantia de qualidade está configurada em 0.05 (menor que 0.20) e falhas detectadas esta em 0.47 – abaixo dos 0.5 definidos como máximo.

O gestor ao verificar no software o alerta, o mesmo poderá tomar medidas reativas com um prazo maior do que o reativo ao problema. Desta forma, contribui para a qualidade final do produto desenvolvido. O software configurado com diversos riscos contribui ainda para a visualização indireta dos riscos em volume, o que é muito difícil de se conseguir quando o monitoramento é feito somente pelo gestor (a olho nú), mesmo que este tenha experiência.

6.3 Outra Área – Economia

Como já apresentado, a proposta deste trabalho pode ser estendido a qualquer área de conhecimento que faça uso de modelos de dinâmica de sistemas. Como também já apresentado, a dinâmica de sistemas possui uma grande abrangência, podendo ser utilizada em qualquer área e com isso, a proposta deste trabalho pode ser utilizada de igual forma. Na Figura 6.9 é apresentado um modelo de dinâmica de sistemas que trata uma questão econômica no Brasil, relacionando o tema aposentadoria (SGRILLO, 2010).

O modelo apresenta o sistema de aposentadoria. Há basicamente três estoques de fato: população ativa (pop ativa), pop aposentada (pop aposentada) e saldo (representando o saldo da previdência). A taxa de ingresso representa parte da população que entra no sistema previdenciário do brasil. Este alimenta o estoque população ativa. O tempo de trabalho influência o fluxo aposentadoria, que por sua vez alimenta o estoque de população aposentada. A saída do estoque de população aposentada é o fluxo morte, representando a morte do aposentado. O estoque saldo é alimentado pelo fluxo de receitas que é representado pelo percentual de desconto do salário do trabalhador com o seu salário médio. As saídas do estoque saldo são

referentes ao fluxo despesa representados pelo valor da pensão pago à população aposentada.

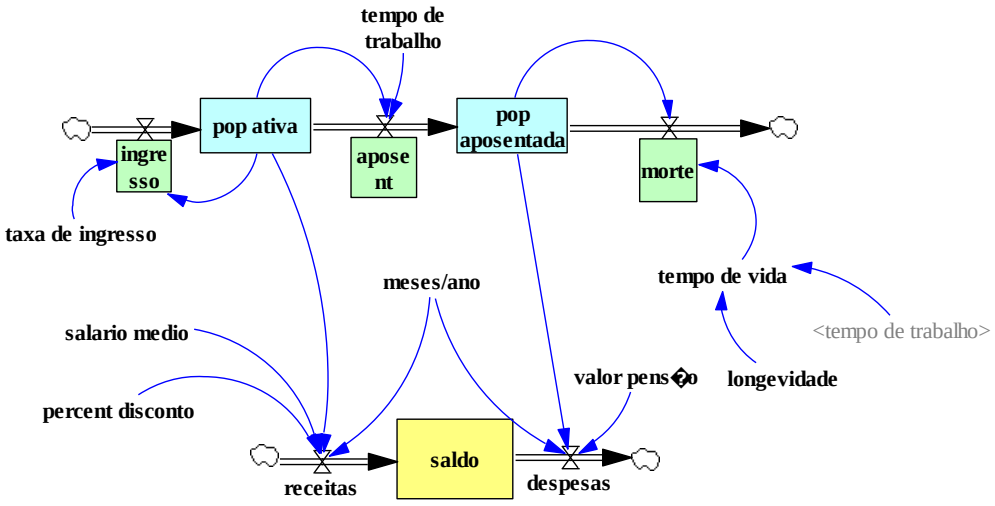


Figura 6.9. Modelo Dinâmica de Sistemas – Sistema de Aposentadoria.
Fonte: Elaborada pelo Autor.

O modelo propõe em suas variáveis padrão uma simulação entre os anos 1950 à 2020, com uma longevidade de 65 anos. Segundo a ONU (Organização das Nações Unidas) (ONU, 2008), a longevidade média da população em países desenvolvidos e em desenvolvimento até 2015 irá ultrapassar os 80 anos. Ajustando a simulação, agora com 80 anos, o resultado é apresentado em formato de gráficos no modelo pela Figura 6.10.

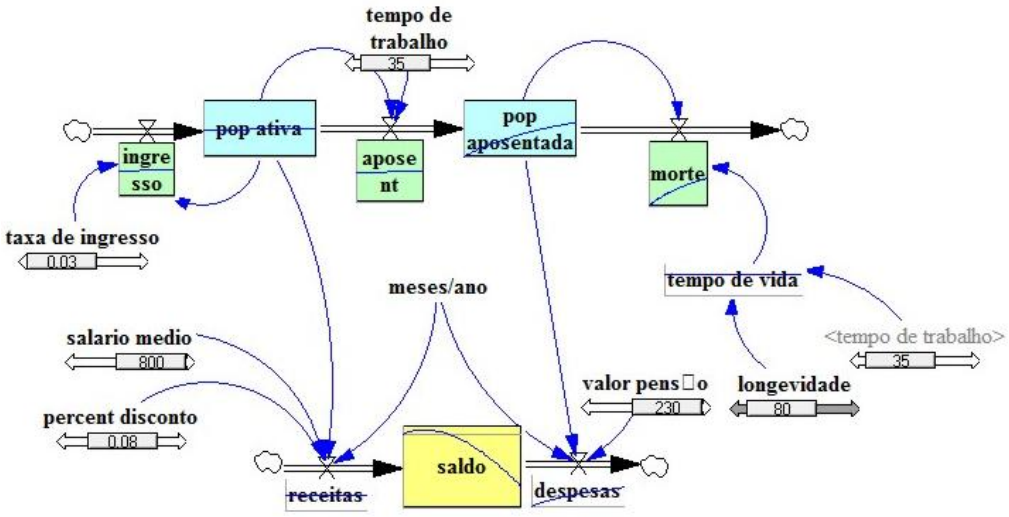


Figura 6.10. Modelo Dinâmica de Sistemas – Simulação com 80 anos.
Fonte: Elaborada pelo Autor.

Ao ser desenvolvido uma análise de riscos, é claro o risco de falta de dinheiro para pagar os aposentados. Uma expressão muito simples é se o saldo estiver em decréscimo, haverá risco de falta de dinheiro. O que de fato pode ser observado no estoque saldo, configurando assim o risco monitorado e mostrando a aplicabilidade da proposta deste trabalho em qualquer modelo simulado de dinâmica de sistemas

7 CONSIDERAÇÕES FINAIS

O monitoramento de riscos em projetos de software é uma atividade dentro do processo de desenvolvimento muito pouco utilizado no ambiente empresarial. Poucas organizações a fazem, e dessas, são basicamente as maiores que possuem alguma formalização quanto ao tratamento dos riscos e incertezas. Por conta disso, a maioria dos gestores se veem com pouco tempo para a tomada de decisão face um risco ou já sua materialização - problema. Assim a decisão é muito mais reativa do que preventiva.

A proposta deste trabalho busca contribuir para esse ambiente de tratamento de incertezas e riscos, proporcionando aos gestores um tempo maior para tomada de decisão, bem como proporciona um melhor entendimento da tarefa de monitoramento de riscos como um todo. Aplicando a técnica de dinâmica de sistemas, é possível através de modelos simular o ambiente, estabelecendo parâmetros para predição de riscos, muito antes deles acontecerem. Com isso, utilizando algumas técnicas de inteligência computacional, é possível monitorar os riscos, fornecendo uma ferramenta de apoio ao gestor para suas decisões.

Com a utilização desta proposta, além da possibilidade de se mapear os riscos negativos, é possível também mapear os riscos positivos, de forma a permitir o gestor a provocar ou aceitar estes riscos. Esta proposta é genérica e permite sua utilização em qualquer ambiente ou modelo simulado utilizando a dinâmica de sistemas. Com isso, além de contribuir para o monitoramento de riscos em projetos de software, este trabalho contribui também para o monitoramento de riscos de toda e qualquer área onde a dinâmica de sistemas seja utilizada.

Desta forma é possível afirmar que a hipótese apresentada neste trabalho é verdadeira. Ou seja, a utilização de uma ferramenta de suporte à decisão com base em modelos e simulações de dinâmica de sistemas possibilita o monitoramento de riscos inerentes a projetos de desenvolvimento de software. Tanto a proposta deste trabalho, quanto as pesquisas na literatura foram capazes de comprovar esta hipótese. Com isso cumpre-se também o objetivo geral deste trabalho, de definir um *framework* conceitual para monitoramento de riscos no processo de desenvolvimento de software com base nas técnicas de modelagem e simulação de dinâmica de sistemas. Este *framework*,

precisou no seu suporte conceitual da utilização de algumas técnicas de inteligência computacional, como as regras de produção (e na primeira abordagem de redes neurais artificiais). A utilização de um método para representação do conhecimento do gestor foi de fundamental importância para a definição do *framework*. Somente assim, foi possível permitir que o gestor pudesse configurar o risco e seus indícios da forma que ele acredita ser melhor de acordo com sua experiência e de acordo com o projeto que ele está coordenando.

Cada projeto possui uma característica diferente, com diferentes membros de equipe. Desta forma, dar a opção ao gestor definir e configurar o risco contribuir para aumentar o nível de acerto da proposta na sua tarefa de identificar e monitorar os riscos. Os objetivos específicos também foram alcançados, sendo que o levantamento bibliográfico e os trabalhos correlatos foram apresentados, respectivamente, no capítulo dois e três deste trabalho. Desta forma foi possível desenvolver um estudo sobre trabalhos que aplicam as técnicas de dinâmica de sistemas na área de engenharia de software. O quarto capítulo apresentou toda a base teórica da proposta de monitoramento de riscos, bem como apresentou uma prova de conceito buscando maximizar o entendimento dos conceitos, tanto na primeira abordagem, utilizando redes neurais artificiais, quanto da segunda, utilizando regras de produção.

Foi possível especificar um SAD (sistema de apoio a decisão) definindo um sistema de informação executiva, ou do inglês *information executive system*, de forma a implementar os conceitos da proposta deste trabalho, apresentando sua aplicabilidade real. Um protótipo funcional deste SAD foi implementado e sua arquitetura foi apresentada, constando seu diagrama de classe e suas funções.

Assim este trabalho contribui para avaliar a aplicabilidade da dinâmica de sistemas apoiando tarefas de nível gerencial em projetos de software. Com isso, este trabalho se constitui como mais um com esta tarefa e justificar a utilização da dinâmica de sistemas para auxiliar gestores em tomada de decisões.

7.1 Trabalhos Futuros

Este trabalho propõe uma técnica para monitoramento de riscos utilizando os resultados das simulações de dinâmica de sistemas e não foi objetivo definir nenhum

modelo. Uma possível extensão para este trabalho seria propor um conjunto de riscos e suas definições com os mapeamentos de acordo com modelos de dinâmica de sistemas dos principais autores sobre a área. Dessa forma seria possível definir heurísticas para as incidências dos riscos, melhorando a sua percepção e identificação.

Esta proposta poderia ser submetida a um trabalho de experimentação onde a técnica aqui definida seria utilizada por gestores de projeto de software e seriam analisados os resultados do monitor de riscos com o que ocorreu na realidade, identificando assim o índice de acerto da proposta, juntamente com o modelo de dinâmica de sistemas. Assim, seria possível validar de forma experimental a proposta, avaliando os resultados.

Seguindo a linha experimental, avaliar os dados mantidos pela utilização da proposta pode favorecer a identificação de padrões sobre os riscos e seus mapeamentos, que não foram identificados pelo gestor. A aplicação de técnicas de descoberta de conhecimento em bases de dados e mineração de dados pode favorecer a descoberta de novos padrões para riscos e indícios já existentes ou de novos.

Outra sugestão é em utilizar uma rede neural artificial que tenha a capacidade de, após a observação do sistema funcionando, encontrar automaticamente os padrões dos riscos e indícios. Desta forma, a rede neural iria propor ao gestor novos riscos e seus indícios ao projeto, permitindo ao gestor aceitar ou não a sugestão da rede neural artificial. Esta sugestão vai de encontro com a Etapa 3 da proposta aqui descrita, denominada Configuração / Definição dos Riscos e seria capaz de estabelecer novas regras de produção com base nos resultados de identificação de padrões de um modelo de rede neural artificial.

O SoftEnRisk, na versão atual durante a finalização deste trabalho, não possui integração direta com a simulação da dinâmica de sistema. O sistema é limitado, pois apenas importa as simulações e não há como interagir com elas. A sugestão feita é de desenvolver um ambiente no SoftEnRisk onde o modelo possa ser simulado dentro da ferramenta e seja possível, diretamente pela ferramenta, alterar parâmetros da simulação (*feedback*) e imediatamente após (em tempo real) identificar os riscos configurados.

REFERÊNCIAS BIBLIOGRÁFICAS

5S. 5S Organização Pessoal. Disponível em < http://www.5s.com.br/>. Acesso em: 02 Jan. 2010.

ABDEL-HAMID, T. K.; MADNICK, S. E. Lessons Learned from Modeling the Dynamics of Software Development. Communication of the ACM, v. 32, n. 12, p. 1426-1438, 1989.

ABDEL-HAMID, T. K. Investigating the cost/schedule trade-off in software development. IEEE Software, v. 7, n. 1, p. 97-105, 1990.

ABDEL-HAMID, T. K.; MADNICK, S. E. Software Project Dynamics: an Integrated Approach. Englewood Cliffs: Prentice Hall, 1991. 264 p.

ABDEL-HAMID, T. K. Adapting, correcting, and perfecting software estimates: a maintenance metaphor. IEEE Computer, v. 26, n. 3, p. 20-29, 1993.

ABDEL-HAMID, T. K. The slippery path to productivity improvement. IEEE Software, v. 13, n. 4, p. 43-52, 1996.

ABDEL-HAMID, T. K.; SENGUPTA, K.; SWETT, C. The Impact of Goals on Software Project Management: An Experimental Investigation. MIS Quarterly, v. 23, n. 4, p. 531-555, 1999.

AGILE ALLIANCE. Disponível em <http://www.agilealliance.org/>. Acesso em: 15 Abr. 2007.

AGILE MANIFESTO. Disponível em <http://agilemanifesto.org/>. Acesso em: 15 Abr. 2007.

AMBRÓSIO, B. G. Modelagem da Fase de Requisitos em Processos de Desenvolvimento de Software: Uma Abordagem Utilizando Dinâmica de Sistemas. Viçosa: Universidade Federal de Viçosa, 2008. Dissertação.

ARAÚJO, M. A. P; TRAVASSOS, G. H. Systems Dynamics, Semi-quantitative Analysis and Systematic Reviews in the context of Software Evolution Experimental Studies. 2005.

BARR, A; FEIGENBAUM, E. A. The Handbook of Artificial Intelligence, volume I-II. William Kaufmann Inc., Los Altos, California, 1981.

BARROS, M. O.; WERNER, C. M. L.; TRAVASSOS, G. H. Risk Analysis: a Key Success Factor for Complex System Development. In: INTERNATIONAL CONFERENCE ON SOFTWARE AND SYSTEMS ENGINEERING AND THEIR APPLICATIONS, 12., 1999, Paris. Proceedings… Paris: CNAM, 1999.

BARROS, M. O.; WERNER, C. M. L.; TRAVASSOS, G. H. Explaining the Behavior of System Dynamics Model. In: INTERNATIONAL CONFERENCE OF THE SYSTEM DYNAMICS SOCIETY, 19., 2001a, Atlanta. Abstract… Albany: System Dynamics Society, 2001. p. 34.

BARROS, M. O.; WERNER, C. M. L.; TRAVASSOS, G. H. From Metamodels to Models: Organizing and Reusing Domain Knowledge in System Dynamics Model Development. In: INTERNATIONAL CONFERENCE OF THE SYSTEM DYNAMICS SOCIETY, 19., 2001b, Atlanta. Abstract… Albany: System Dynamics Society, 2001. p. 34.

BARROS, M. O.; WERNER, C. M. L.; TRAVASSOS, G. H. Scenario Oriented Project Management Knowledge Reuse within a Risk Analysis Process. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING AND KNOWLEDGE ENGINEERING, 13., 2001c, Buenos Aires. Proceedings… Buenos Aires, 2001. p. 37-44.

BARROS, M. O.; WERNER, C. M. L.; TRAVASSOS, G. H. System Dynamics Extension Modules for Software Process Modeling. In: PROSIM - INTERNATIONAL WORKSHOP ON SOFTWARE PROCESS SIMULATION AND MODELING, 4., 2003, Oregon.

BRAGA, A. P., Carvalho, A. P. L. F., Ludemir, T. B.; Redes Neurais Artificiais: Teoria e Aplicações. Editora LTC, 2007.

BRAGA, J. L; SILVA, C. A. B; WIAZOWSKI, B. A; AVELLAR, S. O. C. Modelagem Com Dinâmica De Sistemas. Métodos quantitativos em economia, Viçosa, UFV, n.pag. 33, ISBN: 8572691901, 2004.

BRATKO, I., MOZEITIE, I., LAVRAE, N., 1990. Kardio: a study in deep and qualitative knowledge for expert systems, MIT Press, Cambridge, MA, USA.

BERNSTEIN, P.L. 1997. Desafio aos deuses: a fascinante história do risco. São Paulo, Editora Campus, 389 p.

BOEHM, B; TURNER, R. Balancing Agility and Discipline: A Guide for the Perplexed. Addison Wesley. 2003.

BOEHM, B. A Spiral Model of Software Development and Enhancement. IEEE Computer 21. 1988.

BOEHM, B. W. Software Risk Management: principles and practices, IEEE Software, Volume 8. No1. p 32-40, 1991.

BORLAND. Site Oficial. Disponível em <http://www.borland.com/br/>. Acesso em 07 Jan. 2010.

BROOKS, F. P. The Mythical Man-Month: Essays on Software Engineering. 20th Anniversary Edition. Boston: Addison-Wesley, 1995. 322 p.

CHARETTE, R. Application strategies for risk analysis. New York: MultiScience Press. p 17-21, 1990.

CHARNIAK, E; MCDERMOTT, D. Introduction to Artificial Intelligence. Addison-Wesley Publishing Company, Reading, MA, 1985.

CMMI. Software Engineering Institute. Disponível em < http://www.sei.cmu.edu/cmmi/>. Acesso em: 02 Jan. 2010.

COELHO, J. G. Gestão de Segurança da Informação: um método de identificação e análise de riscos de segurança. Ubá – FAGOC – Faculdade Governador Ozanam Coelho. Monografia de Conclusão de Curso, 2009.

COLLOFELLO, J; RUS, I; CHAUHAN, A; HOUSTON, D; SYCAMORE, D; SMITH-DANIELS, D. A System Dynamics Process Simulator for Staffing Policies Decision Support, Proceedings of the Hawaii International Conference on System Sciences (HICSS), January 1998, pp. 103-111.

COLLOFELLO, J. "Integrating Process Improvements into an Undergraduate Software Engineering Course, Proceedings of the Frontiers in Education Conference 1998, pp. 1298-1301.

COLLOFELLO, J; RUS, J. "Software Process Simulation for Reliability Assessment", Proceedings of the Software Process and Simulation Modeling Workshop, 1998a.

COLLOFELLO, J; SYCAMORE, D. "Using System Dynamics Modeling to Manage Projects", Proceedings of COMPSAC 99, pp. 376-381.

COLLOFELLO, J; ROEHLING, S. "System Dynamics Modeling Applied to Software Outsourcing Decision Support", Proceedings of the Software Process and Simulation Modeling Workshop, 1999.

CYBENKO, G. Approximations by superpositions of sigmoidal functions. Math. Control, Signals, Systems, 2:303—314, 1989.

DAIBERT, M. S. Dinâmica de Sistemas: Uma Abordagem Voltada à Engenharia de Software. Viçosa: UFV, 2008. 70p. (Monografia – Pós-Graduação Lato Sensu em Ciência da Computação)

DEMARCO, T.; LISTER, T. Waltzing with Bears: Managing Risk on Software Projects. New York: Dorset House Publishing Co Inc, 2003. 144 p.

FAIRLEY, R. Risk Management For Software's Projects. IEEE Software. p 54-66, 1994.

FORRESTER, J. W. Industrial dynamics. Portland: Productivity Press, 1961.

FORRESTER, J. W. Urban dynamics. MIT Press, 1969.

FORRESTER, J. W. World dynamics. Wright-Allen Press, 1971.

HAYKIN, S. Redes Neurais: princípios e prática. Porto Alegre, Bookman, 2001.

HIGUERAG, P.R. An Introduction to Team Risk Management, Technical Report. Software Engineering Institute, Carnegie Mellon University. USA, 1994.

HOUSTON, D; COLLOFELLO, J; MACKULACK, G. "Simulating Risk Factors for Software Development Risk Management", Journal of Systems and Software, Vol. 59, Issue 3, Dec. 2001, pp. 247-257.

HOUSTON, D; COLLOFELLO, J. "Finding the Influential Factors in Software Process Simulation Models", Journal of Systems and Software, Vol. 59, Issue 3, Dec. 2001, pp. 259-270.

KAIZEN. Kaizen Institute. Disponível em < http://pt.kaizen.com/>. Acesso em: 02 de Jan. 2010.

KELLNER, M, MADACHY, R; RAFFO, D. Software Process Simulation Modeling: Why? What? How?, Journal of Systems and Software, Spring 1999.

Kohonen, T. Self-organized formation of topologically correct feature maps. Biological Cybernetics, 43:59-69, 1982.

KOSCIANSKI, A; SOARES, M. S. Qualidade de Software. 2º Ed. São Paulo: Novatec Editora. 2007.

LEVERSON, N. Safeware: System Safety an Computers. Addison-Weslwy, 1995.

LEWIS, W. E. Software Testing and Continuous Quality Improvemen, 2005.

LIN, C. Y.; ABDEL-HAMID, T.; SHERIF, J. S. Software-Engineering Process Simulation Model (SEPS). Journal of Systems and Software, v. 38, n. 3, p. 263-277, 1997.

MCCALL, J; CAVANO, J. P; A Framework for the Measurement to Software Quality. Em Proc. of the ACM Software Quality Assurance Workshop, Nov., 1978.

MCCORDUCK, P. Machines Who Think. Freeman, San Francisco, 1979.

MCCULLOCH, W. S.; PITTS, W. H. A logical calculus of the ideas immanent in nervous activity. Um cálculo lógico das idéias imanente na atividade nervosa. Bulletin of Mathematical Biophysics , 5:115-133. Bulletin of Mathematical Biophysics, 5:115-133, 1943

MADACHY, R. Software Process Dynamics, Wiley-IEEE Press, Washington D.C., December 2007.

MADACHY, R; TARBET, D. Case Studies in Software Process Modeling with System Dynamics, Software Process Improvement and Practice, Spring 2000.

MADACHY, R. Cost/Schedule/Process Modeling via System Dynamics, Proceedings of the Fourteenth International Forum on COCOMO and Software Cost Modeling, USC, Los Angeles, CA, October 1999.

MADACHY, R; TARBET, D. Case Studies in Software Process Modeling with System Dynamics, Proceedings of the ProSim '99 Workshop, Silver Falls, OR, June, 1999.

MADACHY, R. Tutorial: System Dynamics Modeling Applied to Software Development Processes, Proceedings of ProSim '98 Workshop, Silver Falls, OR, June, 1998.

MADACHY, R. Tutorial: Process Modeling with System Dynamics, Conference on Software Process Improvement, UC Irvine, January, 1997, and Proceedings of the Eighth Software Engineering Process Group Conference, Atlantic City, NJ, May 1996d.

MADACHY, R. Modeling Software Processes with System Dynamics: Current Developments, Proceedings of the 1996 International System Dynamics Conference, Cambridge, MA, July, 1996c.

MADACHY, R. System Dynamics Modeling of an Inspection-Based Process, Proceedings of the Eighteenth International Conference on Software Engineering, IEEE Computer Society Press, Berlin, Germany, March 1996b.

MADACHY, R. System Dynamics Modeling of an Inspection-Based Process, Proceedings of the Litton Software Technology Management Conference, Santa Barbara, CA, April 1996a.

MADACHY, R. System Dynamics and COCOMO: Complementary Modeling Paradigms, Proceedings of the Tenth International Forum on COCOMO and Software Cost Modeling, SEI, Pittsburgh, PA, October 1995.

MINSKY, M. L; PAPERT, S. A. Perceptrons. Cambridge, MA:MIT Press, 1969.

ONU. Organização das Nações Unidas. Disponível em: <http://www.onu-brasil.org.br/>. Acesso em: 15 Maio 2008.

PFLEEGER, S. L. Engenharia de Software: Teoria e Prática. 2 ed. São Paulo: Prentice Hall, 2004.

PRESMAN, R. S. Software Engineering. 6th Edition. McGraw-Hill. 2006.

ROBERTS, E. B. The Dynamics of Research and Development. Thesis (Ph.D. in Economics) - Department of Economics, Massachusetts Institute of Technology, Cambridge, MA, 1964.

RODRIGUES, A. G. The application of system dynamics to project management: an integrated methodology (SDPIM). PhD Dissertation Thesis. Departament of Management Science, University of Strathclyde, 2000.

RODRIGUES, A. G. Managing and Modelling Project Risk Dynamics: A System Dynamics-based Framework. In: EUROPEAN PROJECT MANAGEMENT CONFERENCE, PMI, 4., 2001, London.

ROSENBLATT, F. The Perceptron: A probabilistic model for information storage and organization in the brain. Psychol, 1958.

RUMELHART, D. E; HINTON, G. E; WILLIAMS, R. J. Learning representations by back-propagating errors. Nature, 323:533, 1986.

RUS, I; COLLOFELLO, J. "Integrating Process Simulation and Reliability Models", CrossTalk: The Journal of Defense Software Engineering, Vol. 14, No.1, January 2001, pp. 15-18.

RUS, I; COLLOFELLO, J; LAKEY, P. "Software Process Simulation for Reliability Planning, Management and Control", Journal of Systems and Software, Vol. 46, No.2, April 1999, pp. 173-182.

RUS, I; COLLOFELLO, J. "A Decision Support System for Software Reliability Engineering Strategy Selection", Proceedings of COMPSAC 99, pp. 213-217.

RUSSEL, S. Inteligência Artificial. Editora Campus, 2003.

SALLES JÚNIOR, SOLER, VALLE, RABECHINI JÚNIOR. Gerenciamento de riscos em projetos. Editora FGV, 2009.

SCRUM. Scrum Alliance. Disponível em <http://www.scrumalliance.org/>. Acesso em: 02 Jan. 2010.

SEI. SOFTWARE ENGINEERING INSTITUTE. Disponível em <http://www.sei.cmu.edu/>. Acesso em: 15 Abr. 2007.

SENGE, P.M. The fifth discipline: the art & practice of the learning organization. New York: Currency Doubleday, 1990.

SENGUPTA, K.; ABDEL-HAMID, T. K.; BOSLEY, M. Coping with staffing delays in software project management: an experimental investigation. IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans, v. 29, n. 1, p. 77-91, 1999.

SGRILLO, R. Dinâmica de Sistemas. Disponível em < http://www.sgrillo.net/sysdyn/>. Acesso em : 15 Abr. 2007.

SOFTEX. MPS.BR – Melhoria de Processo do Software Brasileiro: Guia Geral. Softex. 2006.

SOMMERVILLE, I. Engenharia de Software. 6º Ed.São Paulo: São Paulo, 2003.

STERMAN, J. D. Business dynamics - systems thinking and modeling for a complex world. Boston: Irwin McGraw-Hill, 2000.

SWEBOK. Guide to the Software Engineering Body of Knowledge, 2004.

TURBAN, E., ARONSON, J.E. Decision-support systems and intelligent systems: management support systems. Uppersaddle River, NJ: Prentice-Hall, 1998.

WEINBERG, G. Página Pessoal. Disponível em <http://www.geraldmweinberg.com>. Acesso em: 30 Jan. 2010.

WINDROEW, B; HOFF, M. E. Adaptive switching circuits. Institute of Radio Engineers, Western Electronic Show and Convention, 1960

XP. Extreme Programming: A Gentle Introduction. Acesso em: 02 de Jan. 2010.

BIBLIOGRAFIA

ANDRADE, A. L. Pensamento sistêmico: um roteiro básico para perceber as estruturas da realidade organizacional. [24 dez. 2001]. (http://read.adm.ufrgs.br/read05/artigo/andrade.htm).

AVELLAR, S., SILVA, C.A.B. Gestão estratégica de cadeias produtivas: uma aplicação da simulação dinâmica ao setor de laticínios. In: MOSTRA DE TRABALHOS FINANCIADOS PELA FAPEMIG, 2, 2003, Belo Horizonte. Anais... Belo Horizonte: FAPEMIG, 2003.

BATALHA, M.O. Sistemas agroindustriais: definições e correntes metodológicas. In: BATALHA, M.O. (Coord.). Gestão agroindustrial. São Paulo: GEPAI, 1997. p. 23-48.

CLOUTIER, L.M., DOEHRING, T., SCHROEDER, R.C. The language of system dynamics: creating a clear vision in a complex world. Champaign: Ag. Education & Consulting, 1999. 83 p.

CLOUTIER, L.M., SONKA, S.T. A system dynamics model of information feedback and activity coordination in an agricultural value chain. In: INTERNATIONAL CONFERENCE OF THE SYSTEMS DYNAMICS SOCIETY, 6, 1998, Quebec. Proceedings… Quebec City, Canada, 1998.

COVER, J. Introduction to system dynamics. Herndon: Powersim Pres, 1996. 44 p.

COZZARIN, B.P. Essays on organizational form and function in agricultural production alliances. Urbana: University of Illinois, 1998. Thesis (Ph.D.) – University of Illinois, 1998.

DAVIDSEN, P.I., ASHEIM, L.J. A system dynamics approach to the structure and economy of fur farming and trading. System Dynamics Review, New York, v. 9, n. 3, p. 265-285, 1993.

DAVIDSEN, P.I. A tool for system dynamics modeling. In: ____. Powersim2.5 user's guide. Herndon: Powersim Press, 1996.

FORD, A. Modeling the environment: an introduction to systems dynamics modeling of environmental systems. Washington, D.C.: Island Press, 1999. 401 p.

FORD, D., STERMAN, J. Dynamic modeling of product development processes. System Dynamics Review, v. 14, n. 1, p. 31-68, 1998.

FORRESTER, J.W. Learning through system dynamics as a preparation for the 21st century. 1994a. (Road Map D-4434-1). (http://sysdyn.clexchange.org/road-maps/rm-toc.html).

FORRESTER, J.W. System dynamics, system thinking and soft OR. 1994b. (Road Map D-4405-1) (http://sysdyn.clexchange.org/road-maps/rm-toc.html).

FORRESTER, J.W. World dynamics. Cambridge, Massachussets: The Wright-Allen Press Inc., 1971.

FORRESTER, J.W. Urban dynamics. Cambridge, Massachussets: The MIT Press, 1969.

FORRESTER, J.W. Market growth as influenced by capital investment. Industrial Management Review, v. 9, n. 2, p. 83-105, 1968.

FORRESTER, J.W. Industrial dynamics. Portland: Productivity Press, 1961.

GENTA, P.J., SEVILLE, D. Economic lift-off, boom and bust: a system dynamics view of the Bangkok, Thailand economy 1980-1992. In: SYSTEM DYNAMICS CONFERENCE, 1995, Japan. Proceedings… Japan, 1995. (http://www.magellangroup.com/pubs.htm).

HAMER, G.L., HANSEM, J.W., PHILLIPS, J.G., MJELDE, J.W., HILL, H., LOVE, A., POTGIETER, A. Advances in application of climate prediction in agriculture. [25 out. 2001]. (http://www.elsevier.com/locate/agsy).

HANNON, B., RUTH, M. Dynamic modeling. New York: Springer Verlag, 1994.

HOLSAPPLE, C.W., WHINSTON, A.B. Decision-support systems: a knowledgebased approach. St. Paul, MN: West Pub. Co., 1996.

ITHINK ANALYST. An introduction to systems thinking. HPS – High Performance Systems, Inc., 1997.

LOURENZANI, W., SILVA, C.A.B. Analyzing the economic sustainability of small scale agroindustrial enterprises: a system dynamics approach. Agribusiness: An International Journal, 2004.

LOURENZANI, W.L. Sustentabilidade de empreendimentos agroindustriais de pequeno porte: uma aplicação da metodologia de dinâmica de sistemas. Viçosa: UFV, 2001. 125 p. Dissertação (Mestrado em Ciência e Tecnologia de Alimentos) - Universidade Federal de Viçosa, 2001.

MAIER, F. New product diffusion models in innovation management: a system dynamics perspective. System Dynamics Review, v. 14, n. 4, p. 285-308, 1998.

MARTIN, L.A. An introduction to feedback 1997. (Road Map D-4691). (http://sysdyn.clexchange.org/road-maps/rm-toc.html).

MCCALL, J; RICHARDS, P; WALTERS, G. Factor in software quality. Technical Report (RADC) – TR-77-369,v.13,Rome Air Development Center, United States Air Force, Hanscom AFB,MA,1977.

MORECROFT, J., VAN ACKERE, A. Systems thinking and the art of modelling. Executive Briefing, London Business School, 1998.

MORECROFT, J. A behavioral model of diversification and performance in a mature industry. In: 1996 INTERNATIONAL SYSTEM DYNAMICS CONFERENCE, 1996, Cambridge. Proceedings… Cambridge, 1996.

MORECROFT, J.D.W., LANE, D.C., VIITA, P.S. Modeling growth strategy in abiotechnology startup firm. In: RICHARDSON, G.P. (Eds.). Modelling for management simulation in support of system thinking. Cambridge: Darmouth, 1996. v.1, p. 279-302.

MORECROFT, J. Executive knowledge, models and learning. In: MORECROFT, J.,

STERMAN, J. (Eds.). Modeling for learning organizations. Portland: Productivity Press, 1994.

MORECROFT, J.D.W., LANE, D.C. Modeling growth strategy in a biotechnology startup firm. System Dynamics Review, v. 7, n. 2, p. 93-116, 1991

MORECROFT, J.D.W. Strategy support models. In: Modeling for Management I: Strategic Management Journal, Darthmouth, Aldershot, v. 5, p. 215-229, 1984.

PIDD, M. Tools for thinking. London: West Sussex, 1996.

PORTER, M.E. Estratégia competitiva: técnicas para análise de indústrias e da concorrência. Rio de Janeiro: Campus, 1986. 362 p.

POWERSIM Co. Introduction to systems dynamics. Reston: PowerSim Press, 1996a.

POWERSIM Co. Learning dynamic modeling. Reston: PowerSim Press, 1996b.

POWERSIM 2.5. User's guide. Herndon: Powersim Press, 1996c.

REAGAN-CIRINCIONE, P., SCHUMAN, S., RICHARDSON, G. Decision modeling: tools for strategic thinking. Interfaces, v. 21, p. 52-65, 1991.

RICHARDSON, G.P. System dynamics: simulation for policy analysis from a feedback perspective. In: FISHWICK, P.A., LUKER, P.A. (Eds.). Qualitative Simulation modeling and analysis. New York: Springer-Verlag, 1991. p.144-169.

RISCH, J.D., BERMÚDEZ, L.T., STERMAN, J.D. Designing corporate strategy with system dynamics: a case study in the pulp and paper industry. System Dynamics Review, v. 11, n. 4, p. 249-274, 1995.

RUTH, M., CLOUTIER, L.M., GARCIA, P. A nonlinear model of information and coordination in hog production: testing the Coasian-Fowlerian dynamic hypotheses. In: ANNUAL MEETING OF THE AMERICAN AGRICULTURAL ECONOMICS ASSOCIATION, 1998, Utah. Proceedings… Utah, 1998.

SCHROEDER, R.C. Incentive distortions in commodity markets: lessons from the soybean industry. 1998. (http://www.centrec.com/Articles/IncentiveDistortionsCommMrkts.pdf).

SENGE, P.M. The fifth discipline: the art & practice of the learning organization. New York: Currency Doubleday, 1990.

SENGE, P.M. et al. A quinta disciplina: caderno de campo. Rio de Janeiro: Qualitymark, 1999. 543 p.

SHRECKENGOST, R.C. Dynamic simulation models: how valid are they? 1985. (Road Map D-4436). (http://sysdyn.clexchange.org/road-maps/rm-toc.html).

SICE, P., MOSEKILDE, E., MOSCARDINI, A., LAWLER, K., FRENCH, I. Using system dynamics to analyse interactions in duopoly competition. [10 abr. 2001]. (http://www.elsevier.com).

SONKA, S., CLOUTIER, M. System dynamics to evaluate information coordination in agricultural supply chains. Revista Brasileira de Agroinformática, v. 1, n. 1, p. 1-16, 1998.

STERMAN, J.D. Business dynamics - systems thinking and modeling for a complex world. Boston: Irwin McGraw-Hill, 2000. 982 p.

STERMAN, J.D. Systems dynamics modeling for project management. Cambridge, MA: MIT Press, 1992.

TOWILL, D.R. Industrial dynamics modelling of supply chains. Logistics Information Management, v. 9, n. 4, p. 43-56, 1996.

TOWILL, D.R., NAIM, M.M., WIKNER, J. Industrial dynamics simulation models in the design of supply chains. International Journal of Physical Distribution & Logistic Management, v. 22, n. 5, p. 3-11, 1992.

TURBAN, E., ARONSON, J.E. Decision-support systems and intelligent systems: management support systems. Uppersaddle River, NJ: Prentice-Hall, 1998.

WIAZOWSKI, B.A. Dinâmica de sistemas: uma aplicação à análise da coordenação vertical no agronégocio da carne bovina. Viçosa: UFV, 2001. 125 p. Dissertação (Mestrado em Economia Rural) - Universidade Federal de Viçosa, 2001.

WIAZOWSKI, B.A., LOURENZANI, W.L., SILVA, C.A.B. O uso de sistemas dinâmicos como ferramenta de aprendizagem. Economia Rural, Viçosa, v. 3, n. 10, p. 29-33, 1999.

WIAZOWSKI, B., SILVA, C.A.B. Coordenação de cadeias produtivas: uma aplicação de sistemas dinâmicos ao agronegócio da carne bovina. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE INFORMÁTICA APLICADA À AGROPECUÁRIA E AGROINDÚSTRIA, 2, 1999, Campinas. Anais. Campinas, 1999.

APÊNDICE A – ESPECIFICAÇÃO DO SOFTENRISK

UNIVERSIDADE FEDERAL DE VIÇOSA
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
DEPARTAMENTO DE INFORMÁTICA

Documentação de Sistema
SoftEnRisk

Autor:
MARCELO SANTOS DAIBERT

VIÇOSA
MINAS GERAIS – BRASIL
2010

Revisões Anteriores

Revisor	Descrição	Data
Marcelo Daibert	Primeira Elaboração	20/02/2009
Marcelo Daibert	Definição dos Requisitos Funcionais	22/02/2009
Marcelo Daibert	Definição dos Requisitos não funcionais	29/02/2009
Marcelo Daibert	Matriz de Rastreabilidade	15/03/2009
Marcelo Daibert	Casos de uso e matriz de dependência	16/03/2009
Marcelo Daibert	Diagrama de classes – MVC – Persistência	16/03/2009
Marcelo Daibert	Especificação de Casos de Uso	20/03/2009
Marcelo Daibert	Alteração nos requisitos funcionais	25/03/2009
Marcelo Daibert	Alteração nos requisitos funcionais	26/03/2009
Marcelo Daibert	Alteração no Diagrama de casos de uso	29/03/2009
Marcelo Daibert	Protótipos	09/04/2009
Marcelo Daibert	Alteração na Especificação de casos de uso	10/04/2009
Marcelo Daibert	Protótipos	10/05/2009
Marcelo Daibert	Diagrama de Navegabilidade de Interfaces	11/05/2009
Marcelo Daibert	Alteração na especificação de casos de uso	11/05/2009
Marcelo Daibert	Alteração nas Matrizes	12/05/2009
Marcelo Daibert	Especificação de casos de uso – Adição do diagrama de sequencia no UC8	10/11/2009
Marcelo Daibert	Integração da Especificação ao Corpo da Dissertação	11/09/2010

SUMÁRIO

1. INTRODUÇÃO

1.1 Propósito do Documento

O propósito deste documento é descrever e especificar a ferramenta SoftEnRisk – Software Engineering Risk Analysis and Management Application. Esta ferramenta é um SAD (sistema de apoio a decisão) para a utilização durante o desenvolvimento de *software*.

1.2 Escopo

O objetivo do SoftEnRisk é fornecer ao gestor de projetos de software informações sobre a atividade dos riscos mantidos e configurados no ambiente da ferramenta. A ferramenta irá atuar como manipuladora dos dados resultantes das simulações dos modelos de dinâmica de sistemas, e com base neles determinar o grau (em uma escala de 0 a 100) de materialização de algum risco configurado. Para tanto, a ferramenta deverá permitir o controle e consulta dos seguintes itens:

- gestão e Manutenção de Projetos de Software: elemento que permite ao gestor, no mesmo ambiente, manipular dados e riscos de diversos projetos de software;
- importação das Simulações dos Modelos da Dinâmica de Sistemas: permite a importação manual e automática dos modelos da dinâmica de sistemas. No caso automático, a ferramenta estará limitada à importação das ferramentas nela configurada (Vensim e Sphinx);
- mapeamento dos Indícios dos Modelos para os Riscos: permite a gestão e manutenção dos registros de mapeamento. Caso o gestor faça a utilização da funcionalidade de importação automática, o mapeamento também será automático. O sistema irá sugerir os indícios com face aos elementos do modelo importado juntamente com a sua simulação;
- gestão e Manutenção dos Indícios: permite a manipulação dos indícios de riscos pelo sistema;

- gestão e Manutenção dos Riscos: permite a manipulação e configuração dos riscos pelo gestor no sistema com base aos indícios identificados na simulação dos modelos da dinâmica de sistema;
- identificação dos Riscos: calcular a escala de materialização do risco.

1.3 Registro de Projeto

Este projeto e esta documentação foram desenvolvidos no contexto do projeto de mestrado "Monitoramento de Riscos em Projetos de Software: Uma Abordagem Utilizando Dinâmica de Sistemas" do programa de pós graduação do departamento de informática da UFV – Universidade Federal de Viçosa.

2. DESCRIÇÃO GERAL

2.1 Visão Geral do Produto

O monitoramento de riscos em projetos de software consiste em uma atividade de gerência responsável por identificar com antecedência problemas que, na maioria das vezes, causam prejuízos aos projetos de software. Esta atividade atua na prevenção da ocorrência de problemas. Neste sentido, o SoftEnRisk irá colaborar nesta atividade de forma integrada com as teorias da dinâmica de sistemas com base na abordagem proposta pelo trabalho cuja esta especificação faz parte.

2.2 Perspectivas do Produto

- o SoftEnRisk deve ser desenvolvido em um único módulo, mas que posteriormente possa ser acoplado em um outro sistema;
- o SoftEnRisk deve ser constituído como um componente, com interfaces de acesso às suas rotinas;
- deve ser desenvolvido no paradigma orientado a objeto e em camadas (arquitetura arcabouço MVC) .

2.3 Funções do Produto

- Gestão de Projetos;
- Importar Simulação;
- Mapear Simulação;
- Gerir Mapeamento;
- Gerir Indícios;
- Gerir Riscos;
- Identificar Riscos.

2.4 Componentes de Evolução do Produto

O desenvolvimento deste sistema será realizado levando em consideração a possível necessidade de integração com outros sistemas. Há a busca por minimizar o acoplamento da camada de persistência do sistema, podendo então alterar este mecanismo facilmente.

Desenvolvimento baseado em camadas. Neste nível macro, sendo as seguintes definidas: Camada de Visão, Camada de Controle, Camada de Modelo, Camada de Persistência e Base de Dados.

2.5 Limitações do Produto

O SoftEnRisk é um sistema independente. Entretanto ele depende dos resultados das simulações dos modelos de dinâmica de sistemas para agir. Estes resultados são gerados por ferramentas específicas para fim ou pela resolução matemática dos modelos de dinâmica de sistemas. Não há integração com as simulações dos modelos, nem é possível interagir diretamente pela ferramenta com os modelos de dinâmica de sistemas.

2.6 Usuários do Sistema

O usuário do SoftEnRisk é o gestor de *software*. Personagem responsável por manter e gerir a equipe de desenvolvimento em um projeto de *software*.

3. REQUISITOS ESPECÍFICOS

3.1 Requisitos Funcionais

Requisito Funcional 1: O sistema deve permitir ao gestor a inclusão, alteração, consulta e exclusão dos dados do projeto no qual o irá atuar.

Requisitos de Dados: nome do projeto, autor, modelo simulado da dinâmica de sistemas e data da última edição – identificação dos riscos.

RF1: Gerir Projetos	Estado: Implementado
Prioridade: Alta	Estabilidade: Alta
Descrição: O sistema deve permitir a manutenção dos dados de projeto. Para iniciar a utilização do sistema, o gestor precisa cadastrar o projeto para que os próximos dados possam ser inseridos no contexto do projeto cadastrado.	

Requisito Funcional 2: O sistema deve permitir ao gestor importar de forma automática os resultados da simulação dos modelos de dinâmica de sistemas.

Requisitos de Dados: variáveis, fluxos e estoques do modelo, bem como seus valores durante toda a simulação.

Limitações: o sistema somente pode importar as simulações das ferramentas vensim e sphinx.

RF2: Importar Simulação Automaticamente	Estado: Implementado
Prioridade: Média	Estabilidade: Baixa - Volátil
Descrição: O sistema deve permitir a importação de forma automática dos resultados da simulação dos modelos de dinâmica de sistemas.	

Requisito Funcional 3: O sistema deve permitir ao gestor importar de forma manual os resultados da simulação dos modelos de dinâmica de sistemas.

Requisitos de Dados: variáveis, fluxos e estoques do modelo, bem como seus valores durante toda a simulação.

RF3: Importar Simulação Manualmente	Estado: Implementado
Prioridade: Alta	Estabilidade: Alta
Descrição: O sistema deve permitir a importação de forma manual dos resultados da simulação dos modelos de dinâmica de sistemas.	

Requisito Funcional 4: O sistema deve permitir ao gestor a inclusão, alteração e consulta dos mapeamentos entre os elementos dos modelos de dinâmica de sistemas e indícios.

Requisitos de Dados: variáveis, fluxos e estoques do modelo, e respectivo indício representativo no sistema.

RF4: Gerir Mapeamento	Estado: Implementado
Prioridade: Alta	Estabilidade: Média
Descrição: O sistema deve permitir a inclusão, alteração e dos mapeamentos entre os elementos dos modelos de dinâmica de sistemas e indícios. Este requisito permite estabelecer uma relação entre o elemento do modelo de dinâmica de sistema com a evidência que será tratada pelo sistema para identificação da materialização do risco.	

Requisito Funcional 5: O sistema deve sugerir com base na importação dos resultados da simulação do modelo de dinâmica de sistemas um mapeamento de seus elementos para indícios no sistema.

Requisitos de Dados: variáveis, fluxos e estoques do modelo, e respectivo indício representativo no sistema.

RF5: Gerar Mapeamento Automático	Estado: Implementado
Prioridade: Baixa	Estabilidade: Alta
Descrição: O sistema deve sugerir com base na importação dos resultados da simulação do modelo de dinâmica de sistemas um mapeamento de seus elementos para indícios no sistema. A regra utilizada para este requisito é que todo elemento do modelo de dinâmica de sistema irá, obrigatoriamente, se tornar um indício com o mesmo nome do elemento. O gestor poderá então remover o indício caso seja necessário posteriormente pelo requisito número 4.	

Requisito Funcional 6: O sistema deve permitir ao gestor a alteração e consulta de indícios do projeto.

Requisitos de Dados: nome do indício, comportamento dinâmico – calculado de acordo com os dados da simulação do modelo (enumeração - crescimento, decrescimento, oscilação ou estável).

RF6: Gerir Indícios	Estado: Implementado
Prioridade: Alta	Estabilidade: Alta
Descrição: O sistema deve permitir ao gestor a alteração e consulta de indícios do projeto.	

Requisito Funcional 7: O sistema deve permitir ao gestor a inclusão, alteração, consulta e exclusão de riscos no projeto.

Requisitos de Dados: nome do risco e a expressão LIBR de ocorrência do risco.

RF7: Gerir Riscos	Estado: Implementado
Prioridade: Alta	Estabilidade: Alta
Descrição: O sistema deve permitir ao gestor a inclusão, alteração, consulta e exclusão de riscos no projeto. Neste momento o gestor faz a configuração de ocorrência do risco baseado na LIBR.	

Requisito Funcional 8: O sistema deve identificar riscos.

Requisitos de Dados: nome do risco, data e hora da identificação.

RF8: Identificar Riscos	Estado: Implementado
Prioridade: Alta	Estabilidade: Alta
Descrição: O sistema deve identificar riscos com base nas regras da LIBR definidas no requisito 7.	

3.2 Requisitos Não Funcionais

Requisito Não Funcional 1: O sistema deve ser desenvolvido utilizando a linguagem de programação JAVA.

RNF1: Linguagem	Estado: Implementado
Prioridade: Alta	Estabilidade: Alta
Descrição: O sistema deve ser desenvolvido utilizando a linguagem de programação JAVA.	

Requisito Não Funcional 2: O sistema deve ser desenvolvido como componente, visando o reaproveitamento do mesmo em outro sistema.

RNF2: Componente	Estado: Implementado
Prioridade: Alta	Estabilidade: Alta
Descrição: O sistema deve ser desenvolvido como componente, visando o reaproveitamento do mesmo em outro sistema.	

Requisito Não Funcional 3: O sistema deve ser desenvolvido respeitando a arquitetura arcabouço MVC – Model View Controller.

RNF3: Arquitetura	Estado: Implementado
Prioridade: Alta	Estabilidade: Alta
Descrição: O sistema deve ser desenvolvido respeitando a arquitetura arcabouço MVC – Model View Controller.	

4. MATRIZ DE RASTREABILIDADE

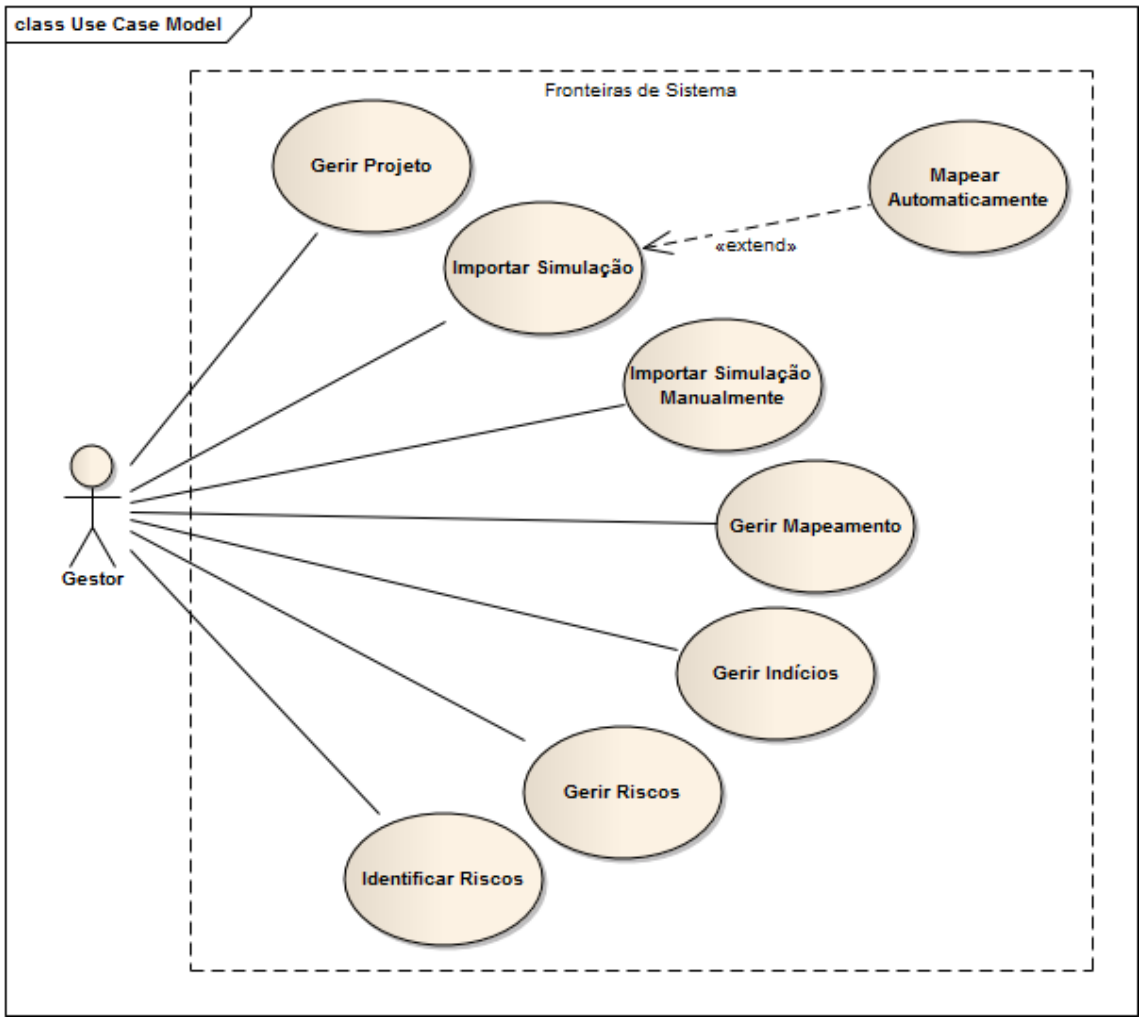
4.1 Requisitos Funcionais X Requisitos Funcionais

	RF1	RF2	RF3	RF4	RF5	RF6	RF7	RF8
RF1								
RF2	X							
RF3		X						
RF4		X	X					
RF5		X	X					
RF6		X		X	X			
RF7		X		X	X	X		
RF8		X					X	

4.2 Requisitos Funcionais X Requisitos Não Funcionais

	RF1	RF2	RF3	RF4	RF5	RF6	RF7	RF8
RNF1	X	X	X	X	X	X	X	X
RNF2	X	X	X	X	X	X	X	X
RNF3	X	X	X	X	X	X	X	X

5. DIAGRAMA DE CASOS DE USO



UC1: Gerir Projeto
Ator: Gestor
Descrição: O caso de uso é utilizado para descrever a primeira ação do gestor ao incluir os dados do projeto no qual irá trabalhar. Este caso de uso abre todas as demais opções no sistema, de forma que o ambiente é configurado para a utilização com base no projeto escolhido / criado.

UC2: Importar Simulação
Ator: Gestor
Descrição: O caso de uso é utilizado pelo gestor para importar de forma automaticamente os resultados da simulação dos modelos de dinâmica de sistema

analisados pelo sistema.

UC3: Importar Simulação Manualmente
Ator: Gestor
Descrição: O caso de uso é utilizado pelo gestor para importar de forma manual os resultados da simulação dos modelos de dinâmica de sistema analisados pelo sistema.

UC4: Gerir Mapeamento
Ator: Gestor
Descrição: O caso de uso é utilizado pelo gestor para manipular os dados do mapeamento entre os elementos da dinâmica de sistemas e os indícios dos riscos a serem estabelecidos.

UC5: Mapear Automaticamente
Ator: Gestor
Descrição: O caso de uso é iniciado de forma automática quando o gestor escolhe por mapear automaticamente os indícios com base na importação automática dos modelos de dinâmica de sistema, definido pelo caso de uso 2.

UC6: Gerir Indícios
Ator: Gestor
Descrição: O caso de uso é utilizado pelo gestor para poder manter os dados dos indícios definidos no sistema pelo mapeamento. Esta opção é possível verificar o comportamento do indício, calculado com base no resultado da simulação do modelo. É possível alterar o comportamento, caso o gestor queira (por motivos de simulação).

UC7: Gerir Riscos
Ator: Gestor
Descrição: O caso de uso é utilizado pelo gestor para manter as informações sobre os riscos. Esta opção permite a utilização da linguagem LIBR, definida pelo trabalho associado a esta especificação, para a definição dos riscos com base a uma lógica booleana juntamente com os indícios de riscos.

UC8: Identificar Riscos
Ator: Gestor
Descrição: Este caso de uso é inicializado pelo gestor quando o mesmo faz acesso à interface do SoftEnRisk e solicita a identificação de riscos. Esta identificação é feita com base na expressão LIBR definida pelo caso de uso 7. E contemplando todos os planos lógicos, define-se a iminência de risco. Esta opção gera um percentual de iminência com base ao total de planos lógicos existentes / o número que são verdadeiros. No caso de todos verdadeiras, há a 100% de possibilidade de ocorrência.

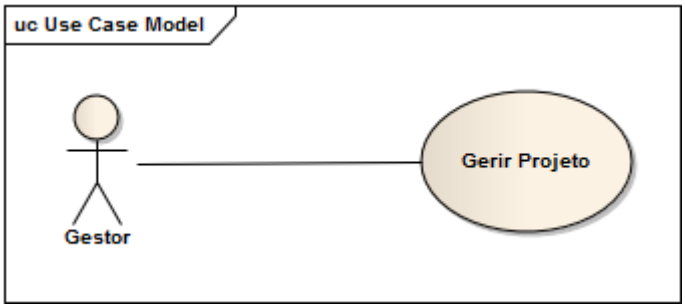
6. MATRIZ DE DEPENDÊNCIA

6.1 Casos de Uso x Requisitos Funcionais

	RF1	RF2	RF3	RF4	RF5	RF6	RF7	RF8
UC1	X							
UC2		X						
UC3			X					
UC4				X				
UC5					X			
UC6						X		
UC7							X	
UC8								X

7. ESPECIFICAÇÃO DE CASOS DE USO

7.1 UC1 – Caso de Uso: Gerir Projeto



Ator Principal:
Gestor.

Sumário:
O caso de uso é utilizado para descrever a primeira ação do gestor ao incluir os dados do projeto no qual irá trabalhar. Este caso de uso abre todas as demais opções no sistema, de forma que o ambiente é configurado para a utilização com base no projeto escolhido / criado.

Pré-Condições:
Não Aplicável.

Fluxo Principal:
1. O sistema exibe a tela de projetos;
2. O sistema executa o subfluxo Pesquisa Projeto.

Fluxos Alternativos:
Não Aplicável.

Subfluxo: Operação Incluir:
1. O gestor aciona a opção de incluir novo projeto com todos os campos habilitados;
2. O sistema solicita a entrada dos seguintes dados: nome do projeto, autor e modelo simulado da dinâmica de sistemas;
3. O gestor informa os dados solicitados;
4. O gestor confirma os dados digitados;
5. O sistema efetua a validação dos dados conforme a RN1;
6. O sistema cadastra o projeto.

Subfluxo: Operação Alterar / Visualizar / Excluir:
1. O sistema efetua a leitura do registro a partir do projeto escolhido previamente na interface de pesquisa;
2. O sistema exibe os dados do projeto escolhido (nome do projeto, autor, modelo simulado da dinâmica de sistemas e data a última edição – atualização);
3. Se desejar, o gestor, tem a opção de alterar os dados do projeto ou excluí-lo;

4. O sistema altera ou excluí os dados cadastrais do projeto selecionado e fecha a interface – No caso de exclusão, todos os demais dados de projeto são excluído (exclusão cascata).

Subfluxo: Escolher Projeto;
1. O sistema efetua a leitura do registro a partir do projeto escolhido previamente na interface de pesquisa;
2. O sistema inicializa a interface principal da aplicação.

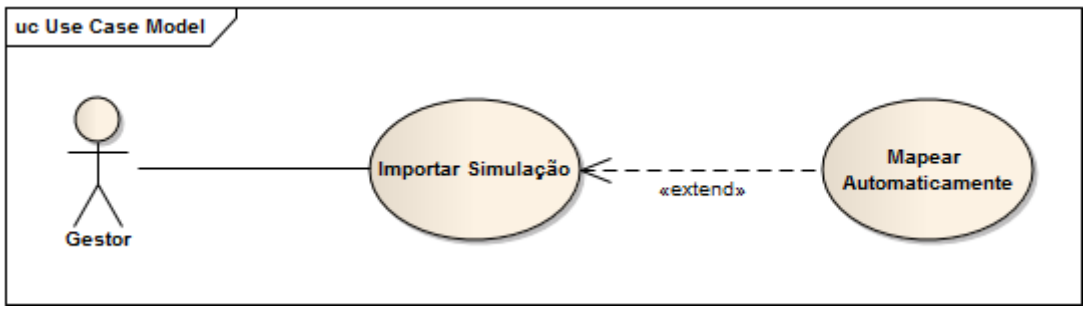
Subfluxo: Pesquisa Projeto:
1. O sistema exibe os projetos já inseridos (nome do projeto – listagem somente com este dado);
2. O sistema apresenta as opções na interface para incluir, alterar/visualizar/excluir projeto e escolher projeto;
3. Com base na opção escolhida é acionado o subfluxo referente.

Fluxos de Exceção:
1. O nome do projeto, autor e/ou modelo simulado da dinâmica de sistemas não foram preenchidos – o sistema exibe uma mensagem e retorna ao formulário de novo projeto;
2. Alteração de registro alterado. Sistema exibe uma mensagem informando que a operação não pode ser realizada indicando o motivo do cancelamento da operação;
3. Alteração de registro excluído. Sistema exibe uma mensagem informando que a operação não pode ser realizada indicando o motivo do cancelamento da operação;
4. Exclusão de registro alterado. Sistema exibe uma mensagem informando que a operação não pode ser realizada indicando o motivo do cancelamento da operação.

Pós Condições:
São habilitas as seguintes funcionalidades: importar simulação (automaticamente e manualmente), e parcialmente: gerir de mapeamento (manualmente e automaticamente), gerir indícios, riscos e identificar riscos.

Regras de Negócio:
RN1 – Os campos obrigatórios são: nome do projeto, autor e modelo simulado da dinâmica de sistemas.

7.2 UC2 – Caso de Uso: Importar Simulação



Ator Principal:
Gestor.

Sumário:
O caso de uso é utilizado pelo gestor para importar de forma automaticamente os resultados da simulação dos modelos de dinâmica de sistema analisados pelo sistema.

Pré-Condições:
Haver projeto cadastrado e selecionado.

Fluxo Principal:
1. O gestor solicita ao sistema a exibição da interface de importar simulação – automaticamente.
2. O sistema solicita os dados de importação automática: local do arquivo com os resultados da simulação e a ferramenta que foi feita a simulação.
3. O sistema faz a importação dos dados e exibe uma mensagem identificando a finalização da tarefa.

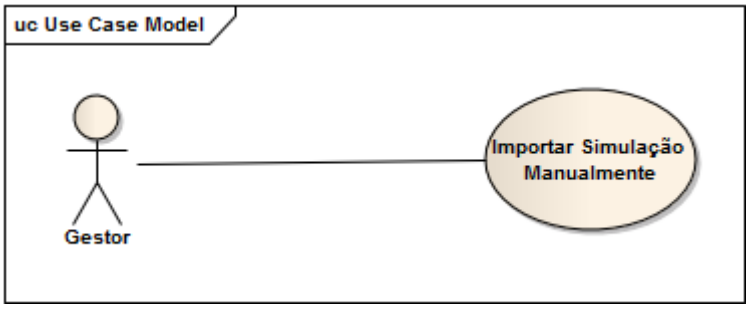
Fluxos Alternativos:
1. Caso o usuário escolha na interface para fazer o mapeamento automático, é executado o caso de uso Mapear Automaticamente.

Fluxos de Exceção:
1. Arquivo mal formatado: o sistema exibe uma mensagem informando que a operação não foi realizada por se tratar de um arquivo com formato não conhecido.

Pós Condições:
Habilita as funcionalidades de gerir mapeamento e gerir indícios.

Regras de Negócio:
Não Aplicável

7.3 UC3 – Caso de Uso: Importar Simulação Manualmente



Ator Principal:
Gestor.

Sumário:

O caso de uso é utilizado pelo gestor para importar de forma manual os resultados da simulação dos modelos de dinâmica de sistema analisados pelo sistema.

Pré-Condições:
Haver projeto cadastrado e selecionado.

Fluxo Principal:
1. O gestor solicita ao sistema a exibição da interface de importar simulação – manualmente.
2. O sistema solicita os dados de importação manual: variáveis, fluxos e estoques – cada elemento da dinâmica de sistemas com o seu devido comportamento dinâmico - crescimento, decrescimento, oscilação ou estável.
3. O gestor confirma os dados apresentados;
4. O sistema efetua as validações conforme a regra de negócio RN1;
5. O sistema faz a leitura e a gravação dos dados e exibe uma mensagem informando que a tarefa foi finalizada.

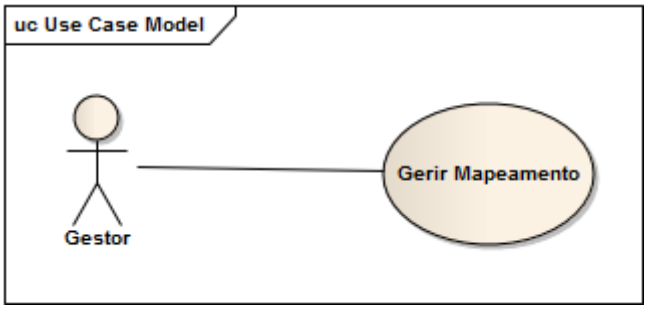
Fluxos Alternativos:
Não Aplicável.

Fluxos de Exceção:
1. Os dados de pelo menos uma variável / fluxo ou estoque e seu comportamento dinâmico não foi preenchido. Sistema exibe uma mensagem e retorna ao formulário de importar simulação manualmente.

Pós Condições:
Habilita as funcionalidades de gerir mapeamento e gerir indícios.

Regras de Negócio:
RN1 – Os campos obrigatórios são: dados de pelo menos uma variável / fluxo ou estoque e o seu comportamento dinâmico.

7.4 UC4 – Caso de Uso: Gerir Mapeamento



Ator Principal:
Gestor.

Sumário:

O caso de uso é utilizado pelo gestor para manipular os dados do mapeamento entre os elementos da dinâmica de sistemas e os indícios dos riscos a serem estabelecidos.

Pré-Condições:
Haver projeto cadastrado e selecionado.
Haver Simulação importada / definida.

Fluxo Principal:
1. O gestor solicita ao sistema a visualização da interface de mapeamento;
2. O sistema exibe os mapeamentos já inseridos (exibe em uma listagem do lado esquerdo o nome de todos os elementos importados da simulação dos modelos de dinâmica de sistemas e do lado direito quais deles foram mapeados para indícios – Como regra de base o indício possui o mesmo nome do modelo, para não se perder a referência);
3. O sistema apresenta opções de selecionar os elementos e os indícios, permitindo inserir ou remover do mapeamento;
4. O usuário faz a seleção e escolhe sua opção;
5. O sistema persiste os dados alterados.

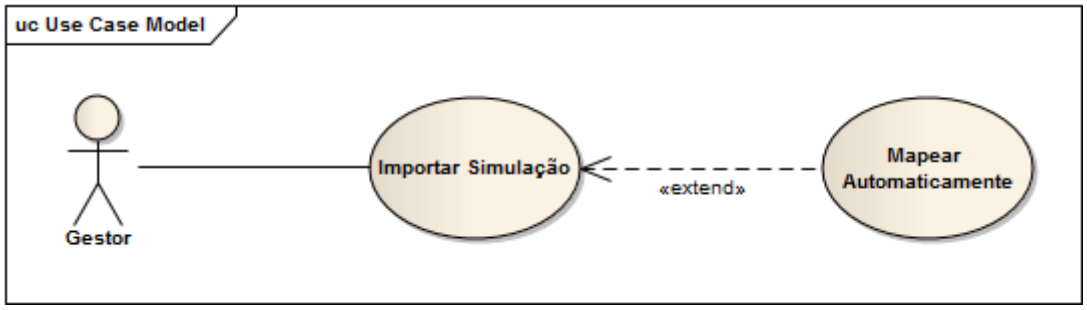
Fluxos Alternativos:
Não Aplicável.

Fluxos de Exceção:
1. Alteração de registro alterado. Sistema exibe uma mensagem informando que a operação não pode ser realizada indicando o motivo do cancelamento da operação;
2. Alteração de registro excluído. Sistema exibe uma mensagem informando que a operação não pode ser realizada indicando o motivo do cancelamento da operação.

Pós Condições:
Habilita as funcionalidades de gestão de riscos.

Regras de Negócio:
Não Aplicável.

7.5 UC5 – Caso de Uso: Mapear Automaticamente



Ator Principal:
Gestor.

Sumário:
O caso de uso é iniciado de forma automática quando o gestor escolhe por mapear automaticamente os indícios com base na importação automática dos modelos de dinâmica de sistema, definido pelo caso de uso 2.

Pré-Condições:
Haver projeto cadastrado e selecionado;
Haver Simulação importada / definida.

Fluxo Principal:
1. O gestor solicita ao sistema, durante o caso de uso importar simulação automaticamente para que haja o mapeamento automático, iniciando este caso de uso;
2. O sistema faz o mapeamento de forma automática, conforme a RN1.
3. O sistema apresenta a interface de mapeamento com a sugestão sendo exibida;
4. O usuário pode fazer alterações conforme sua necessidade, invocando o caso de uso 4.
5. O sistema persiste os dados alterados.

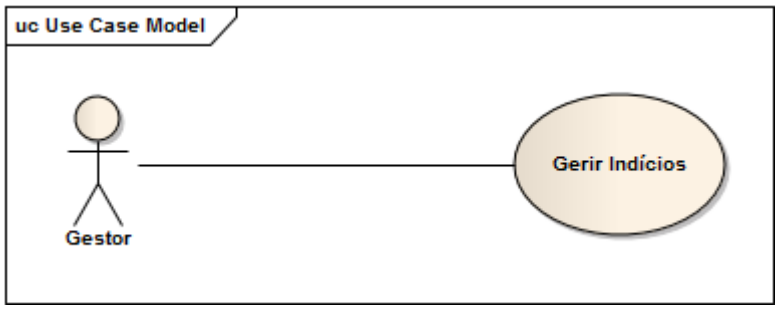
Fluxos Alternativos:
Não Aplicável.

Fluxos de Exceção:
1. Alteração de registro alterado. Sistema exibe uma mensagem informando que a operação não pode ser realizada indicando o motivo do cancelamento da operação;
2. Alteração de registro excluído. Sistema exibe uma mensagem informando que a operação não pode ser realizada indicando o motivo do cancelamento da operação.

Pós Condições:
Habilita as funcionalidades de gestão de riscos.

Regras de Negócio:
RN1 - É adicionando como indício todo e qualquer elemento importado do modelo de dinâmica de sistema.

7.6 UC6 – Caso de Uso: Gerir Indícios



Ator Principal:
Gestor.

Sumário:
O caso de uso é utilizado pelo gestor para poder manter os dados dos indícios definidos no sistema pelo mapeamento. Esta opção é possível verificar o comportamento do indício, calculado com base no resultado da simulação do modelo. É possível alterar o comportamento, caso o gestor queira (por motivos de simulação).

Pré-Condições:
Haver projeto cadastrado e selecionado.
Haver Simulação importada / definida.
Haver Mapeamento definido.

Fluxo Principal:
1. O gestor solicita a exibição da interface de gestão de indícios;
2. O sistema carrega todos os indícios mantidos pelo sistema e exibe em uma listagem (nome do indício, referente a qual mapeamento e qual comportamento).

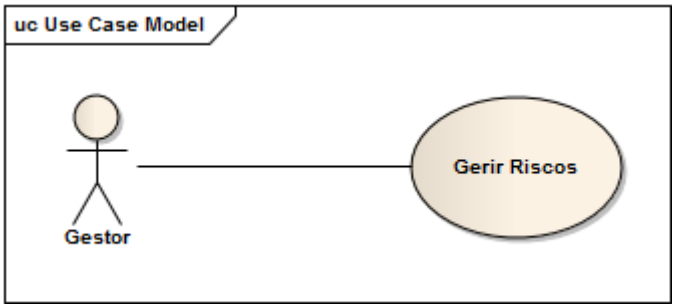
Fluxos Alternativos:
1. O gestor, caso queira, pode alterar o comportamento diretamente na listagem.
2. O sistema efetua as alterações executadas.

Fluxos de Exceção:
1. Alteração de registro alterado. Sistema exibe uma mensagem informando que a operação não pode ser realizada indicando o motivo do cancelamento da operação;
2. Alteração de registro excluído. Sistema exibe uma mensagem informando que a operação não pode ser realizada indicando o motivo do cancelamento da operação.

Pós Condições:
Não Aplicável.

Regras de Negócio:
Não Aplicável.

7.7 UC7 – Caso de Uso: Gerir Riscos



Ator Principal:
Gestor.

Sumário:
O caso de uso é utilizado pelo gestor para manter as informações sobre os riscos. Esta opção permite a utilização da linguagem LIBR, definida pelo trabalho associado a esta especificação, para a definição dos riscos com base a uma lógica booleana juntamente com os indícios de riscos.

Pré-Condições:
Haver projeto cadastrado e selecionado.
Haver Simulação importada / definida.
Haver Mapeamento definido.
Haver Indícios definidos.

Fluxo Principal:
1. O sistema exibe a tela de gestão de riscos;
2. O sistema executa o subfluxo Pesquisa Projeto.

Fluxos Alternativos:
Não Aplicável.

Subfluxo: Operação Incluir:
1. O gestor aciona a opção de incluir novo risco com todos os campos habilitados;
2. O sistema solicita a entrada dos seguintes dados: nome do risco e a expressão LIBR referente ao risco com base nos indícios;
3. O gestor informa os dados solicitados;
4. O gestor confirma os dados digitados;
5. O sistema efetua a validação dos dados conforme a RN1;
6. O sistema cadastra o projeto.

Subfluxo: Operação Alterar / Visualizar / Excluir:
1. O sistema efetua a leitura do registro a partir do projeto escolhido previamente na interface de pesquisa;
2. O sistema exibe os dados do risco escolhido (nome do risco e a expressão LIBR);
3. Se desejar, o gestor, tem a opção de alterar os dados do projeto ou excluí-lo;
4. O sistema altera ou excluí os dados cadastrais do projeto selecionado e fecha a interface.

Subfluxo: Pesquisa Projeto:
1. O sistema exibe os riscos já inseridos (nome do risco e evidência – expressão do risco);
2. O sistema apresenta as opções na interface para incluir, alterar/visualizar/excluir projeto e escolher projeto;
3. Com base na opção escolhida é acionado o subfluxo referente.

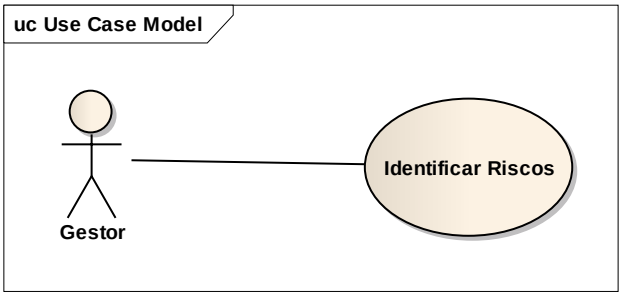
Fluxos de Exceção:
1. O nome do risco e expressão não foram definidos – o sistema exibe uma mensagem e retorna ao formulário de novo projeto;
2. Alteração de registro alterado. Sistema exibe uma mensagem informando que a operação não pode ser realizada indicando o motivo do cancelamento da operação;

3. Alteração de registro excluído. Sistema exibe uma mensagem informando que a operação não pode ser realizada indicando o motivo do cancelamento da operação;
4. Exclusão de registro alterado. Sistema exibe uma mensagem informando que a operação não pode ser realizada indicando o motivo do cancelamento da operação.

Pós Condições:
Habilita a funcionalidade de identificar riscos.

Regras de Negócio:
RN1 – Os campos obrigatórios são: nome do risco e a expressão LIBR.

7.8 UC8 – Caso de Uso: Identificar Riscos



Ator Principal:
Gestor.

Sumário:
Este caso de uso é inicializado pelo gestor quando o mesmo faz acesso à interface do SoftEnRisk e solicita a identificação de riscos. Esta identificação é feita com base na expressão LIBR definida pelo caso de uso 7. E contemplando todos os planos lógicos, define-se a iminência de risco. Esta opção gera um percentual de iminência com base ao total de planos lógicos existentes / o número que são verdadeiros. No caso de todos verdadeiras, há a 100% de possibilidade de ocorrência.

Pré-Condições:
Haver projeto cadastrado e selecionado.
Haver Simulação importada / definida.
Haver Mapeamento definido.
Haver Indícios definidos.
Haver Riscos definidos

Fluxo Principal:
1. O gestor solicita ao sistema a identificação de riscos;
2. O sistema exibe os riscos identificados com base na LIBR dos riscos configurados no sistema;
3. Com base nos valores lógicos identificados, o sistema irá calcular um índice de possibilidade de ocorrência do risco. Este risco é calculado com base ao total de inferências lógicas existentes na LIBR. As verdadeiras entram na estatística existente, gerando um percentual.

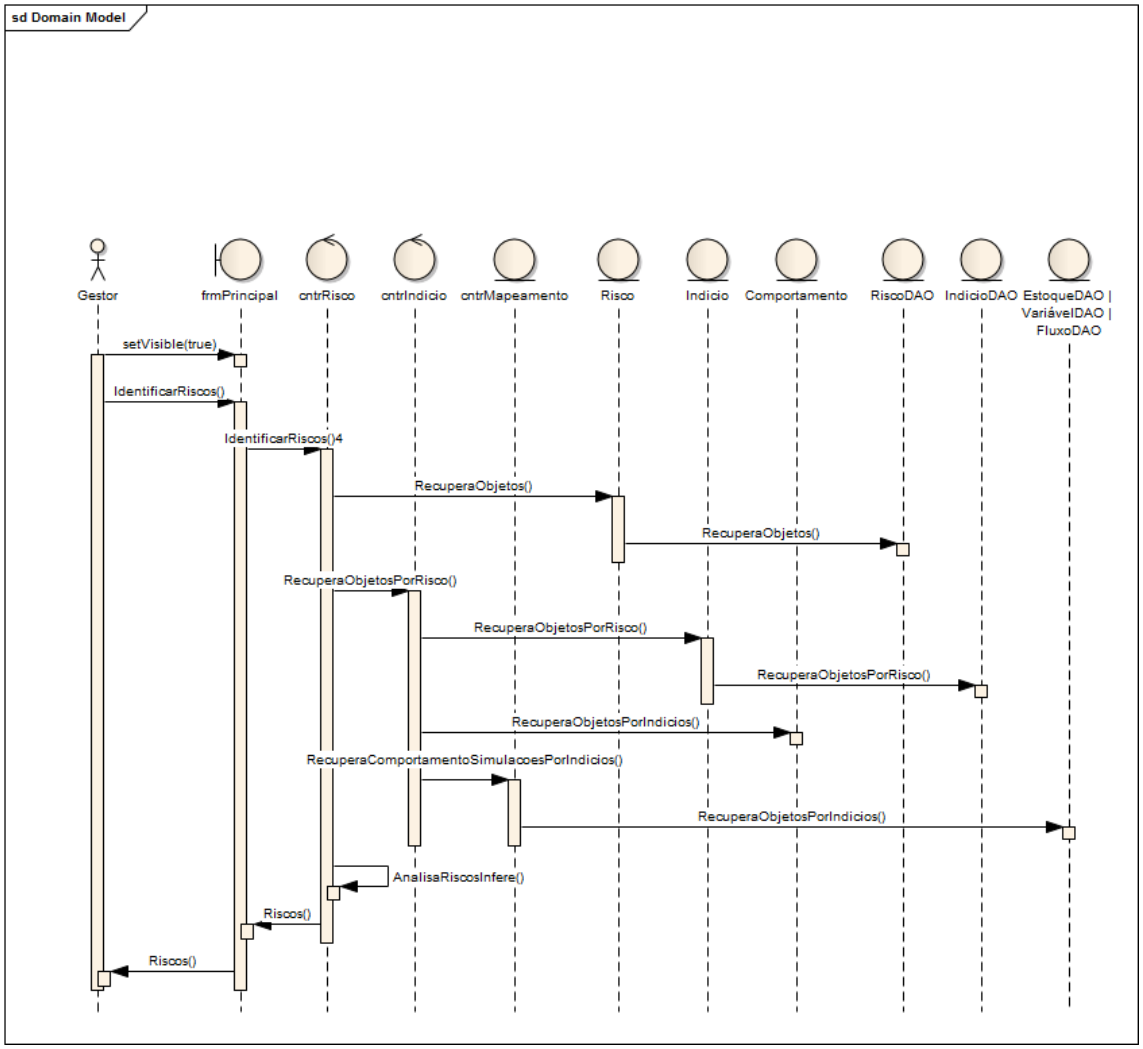
Fluxos Alternativos:
Não Aplicável;

Fluxos de Exceção:
1. Quando não há riscos definidos – o sistema exibe uma mensagem informando que não há riscos definidos para a identificação;
2. Quando há erros na LIBR de algum risco definido – o sistema exibe uma mensagem informando que o risco específico possui um erro na LIBR.

Pós Condições:
Não Aplicável.

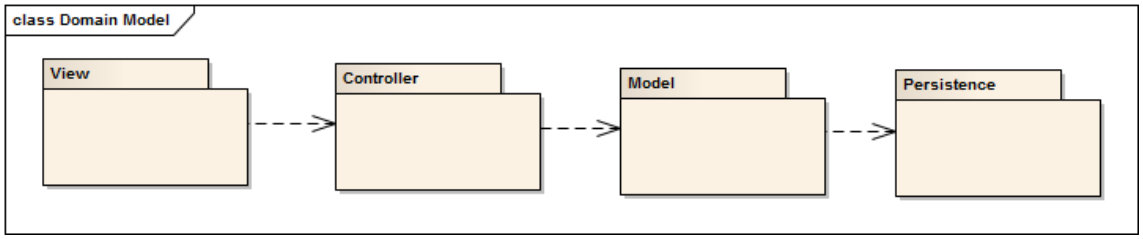
Regras de Negócio:
Não Aplicável.

Diagrama de Sequência:

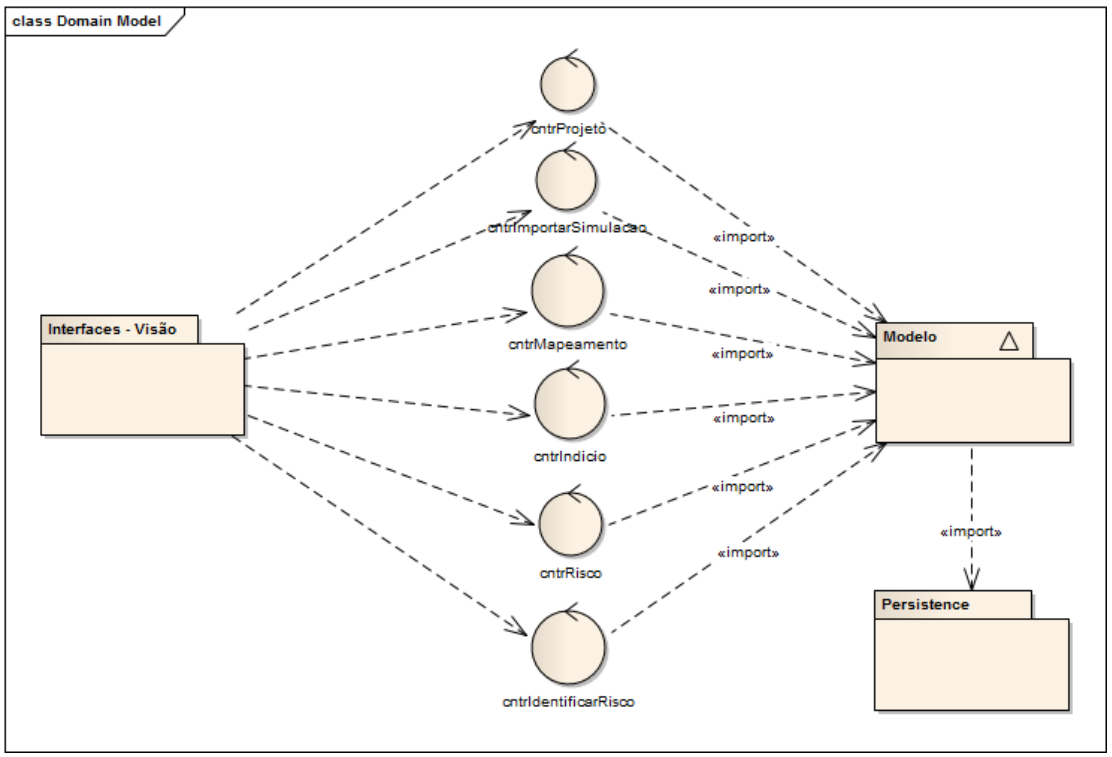


8. MODELAGEM CONCEITUAL

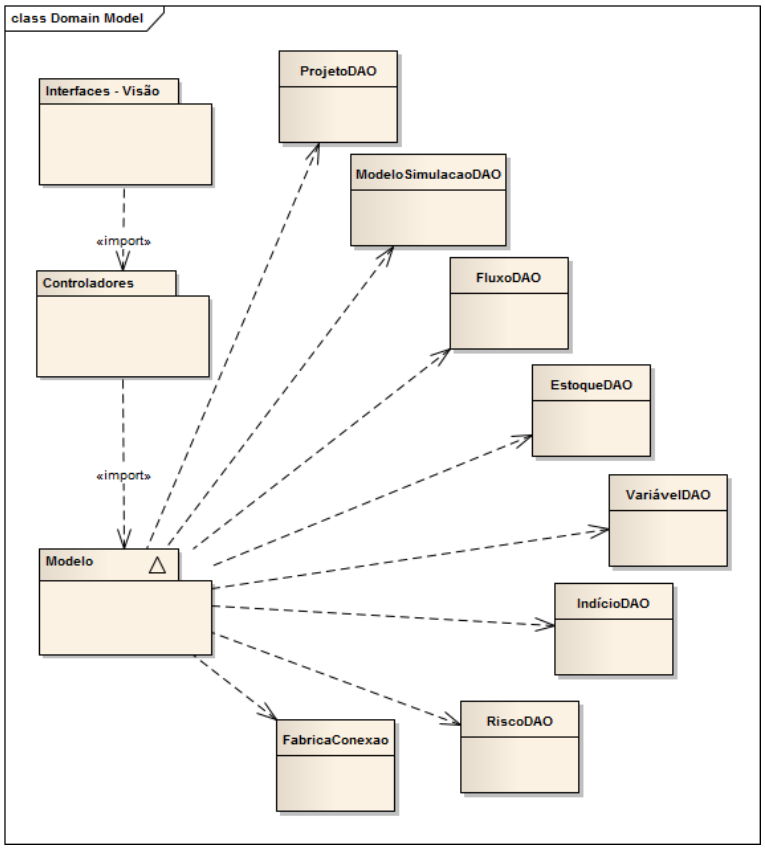
8.1 Modelo MVC (*Model View Controller*) com Persistence



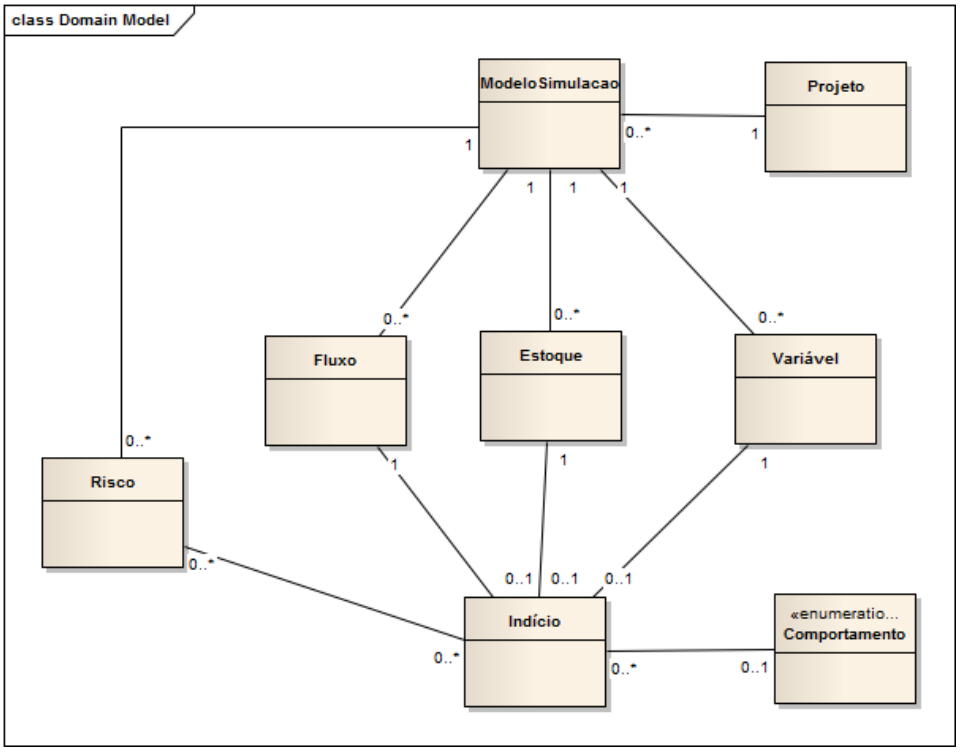
8.2 Diagrama de Classes (Controladoras)



8.3 Diagrama de Classes (Persistência – Padrão *DAO – Data Acess Object*)

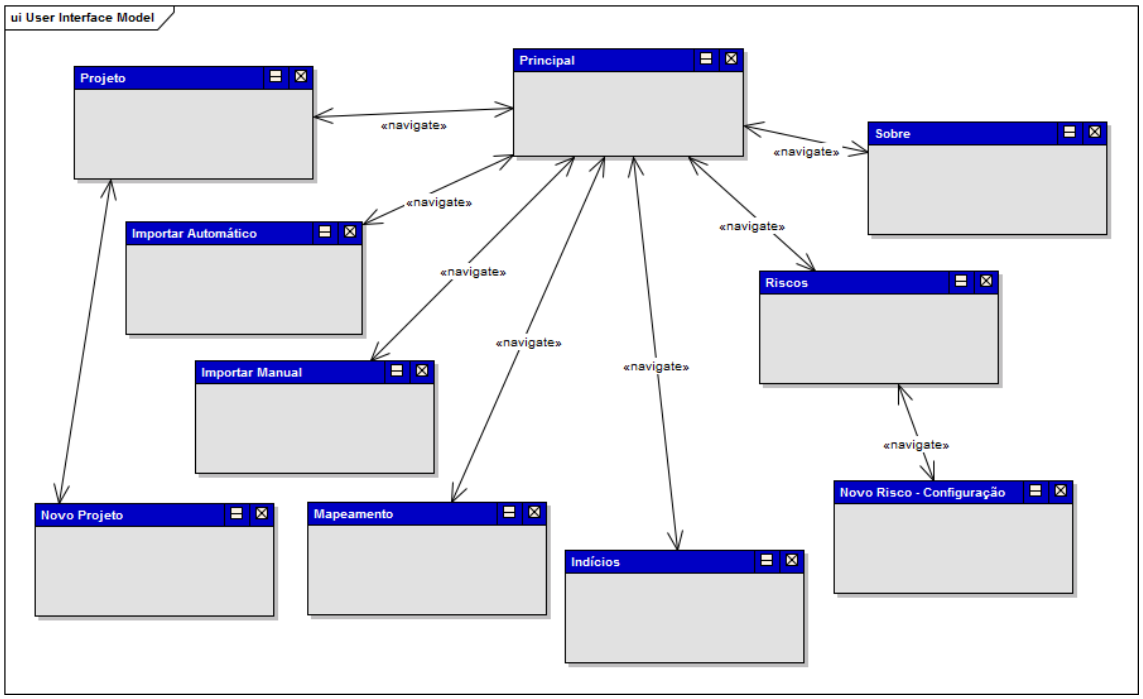


8.4 Diagrama de Classes (Model)

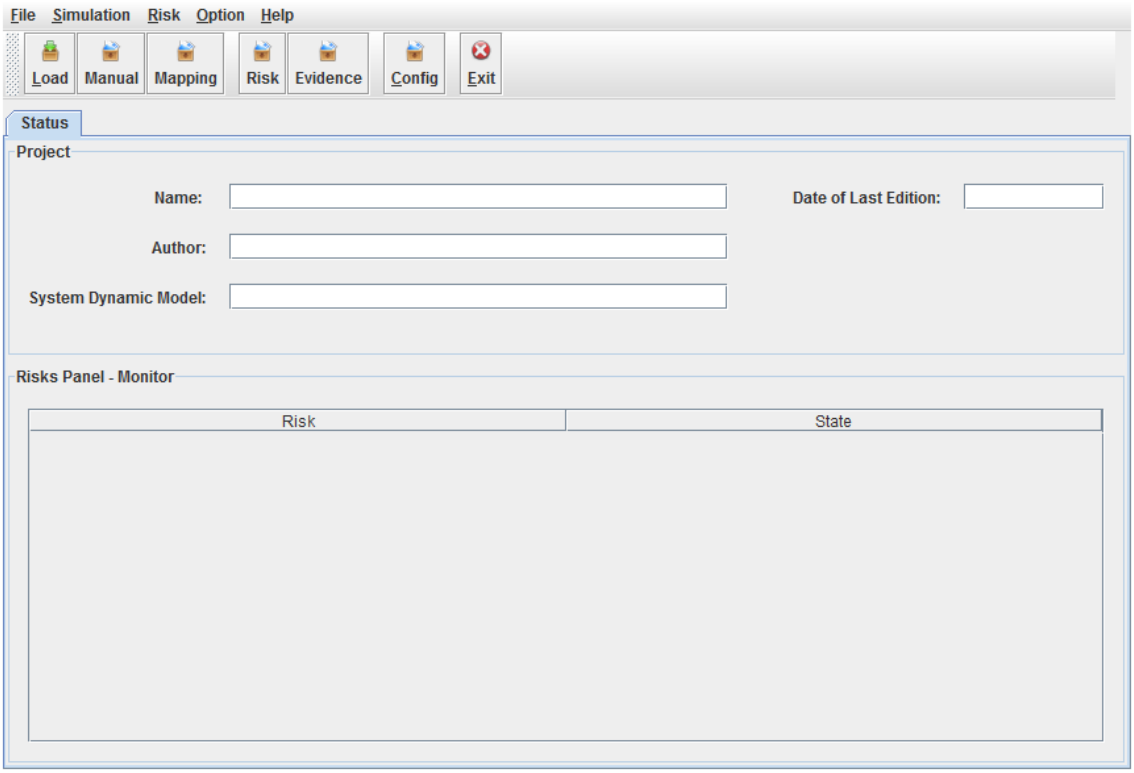


9. PROTÓTIPOS DE INTERFACE

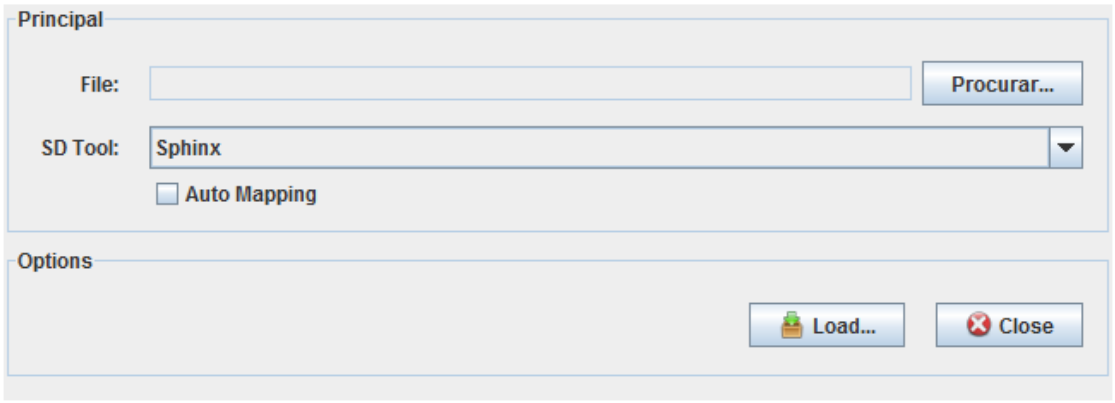
9.1 Navegação Entre Interfaces



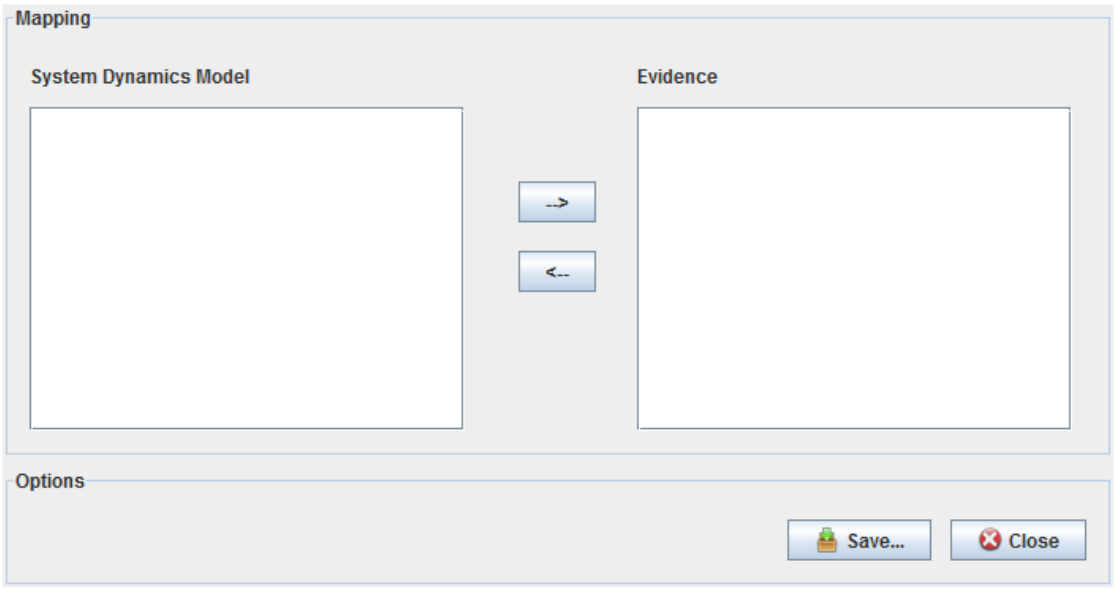
9.2 Protótipos – Interface Principal



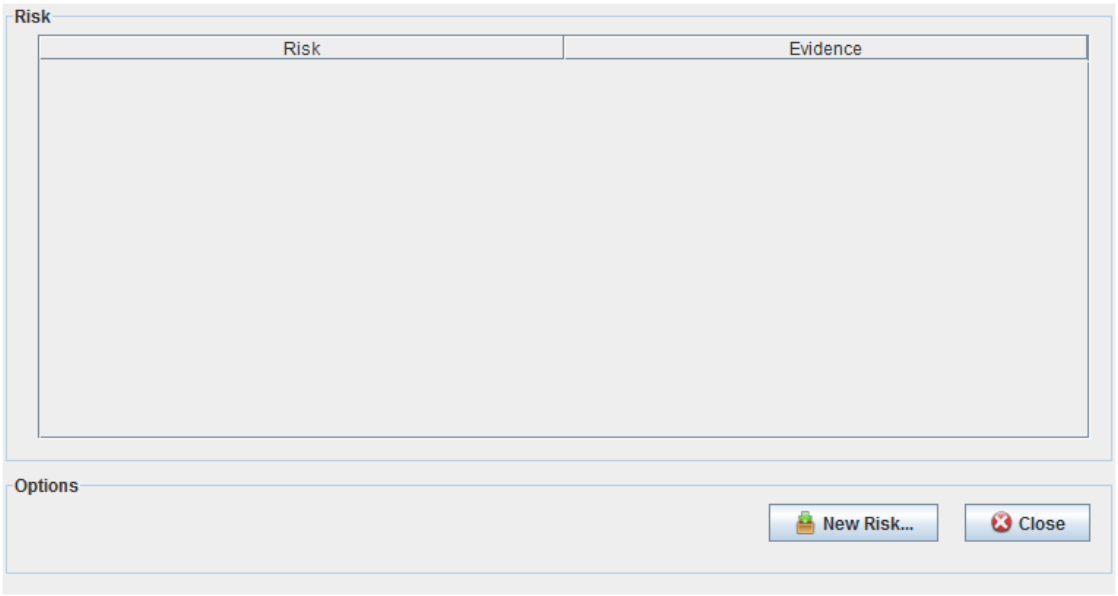
9.3 Protótipos – Importar Automático



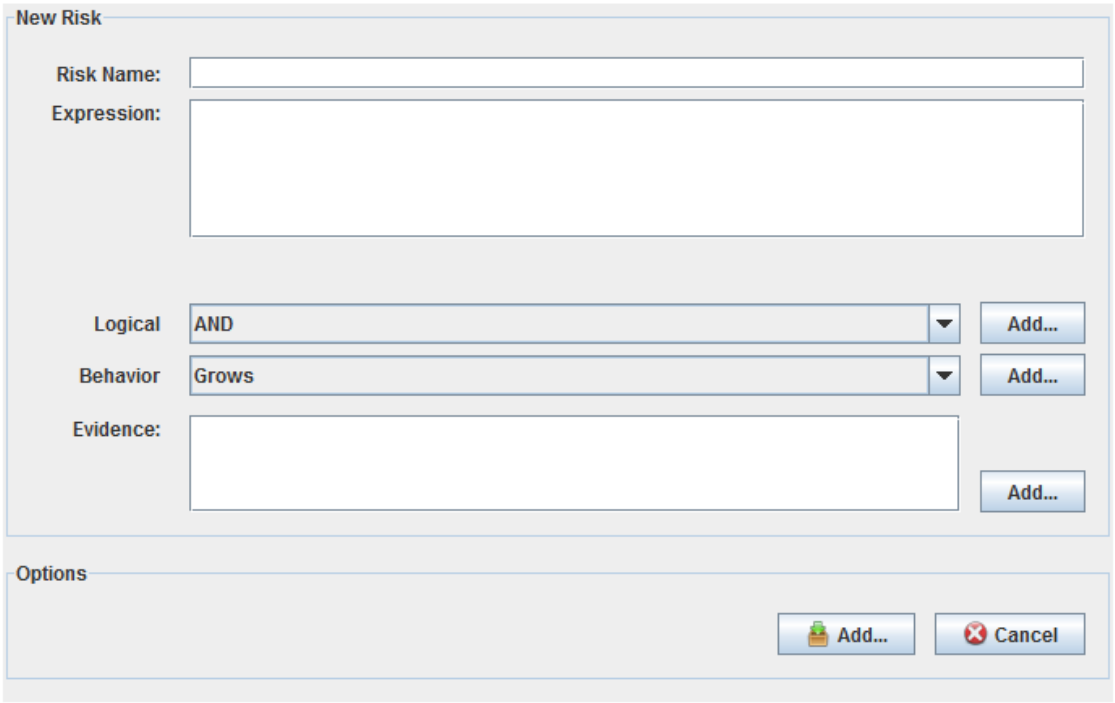
9.4 Protótipos – Interface Mapeamento



9.5 Protótipos – Interface Risco Pesquisa



9.6 Protótipos – Interface Risco Novo / Editar



9.7 Protótipos – Interface Splash – Logo