UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIENCIAS EXATAS
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Cláudio Augusto Silveira Lélis

UM MODELO DINÂMICO DE REPUTAÇÃO PARA APOIAR A MANUTENÇÃO COLABORATIVA DE SOFTWARE

Juiz de Fora
2017

UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIENCIAS EXATAS
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Cláudio Augusto Silveira Lélis

UM MODELO DINÂMICO DE REPUTAÇÃO PARA APOIAR A MANUTENÇÃO COLABORATIVA DE SOFTWARE

Dissertação apresentada ao Programa de Pós-graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. D.Sc. Marco Antônio Pereira Araújo
Coorientador: Prof. D.Sc. José Maria Nazar David

Juiz de Fora
2017

Ficha catalográfica elaborada através do programa de geração
automática da Biblioteca Universitária da UFJF,
com os dados fornecidos pelo(a) autor(a)

Lelis, Claudio Augusto Silveira.
Um modelo dinâmico de reputação para apoiar a manutenção colaborativa de software / Claudio Augusto Silveira Lelis. -- 2017.
141 f. : il.

Orientador: Marco Antônio Pereira Araújo
Coorientador: José Maria Nazar David
Dissertação (mestrado acadêmico) - Universidade Federal de Juiz de Fora, Instituto de Ciências Exatas. Programa de Pós Graduação em Ciência da Computação, 2017.

1. Reputação. 2. Dinâmica de Sistemas. 3. Colaboração. 4. Visualização de Dados. 5. Manutenção de Software. I. Araújo, Marco Antônio Pereira, orient. II. David, José Maria Nazar, coorient. III. Título.

Cláudio Augusto Silveira Lélis

UM MODELO DINÂMICO DE REPUTAÇÃO PARA APOIAR A MANUTENÇÃO COLABORATIVA DE SOFTWARE

Dissertação apresentada ao Programa de Pós-graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Aprovada em 30 de Agosto de 2017.

BANCA EXAMINADORA

_____

Prof. D.Sc. Marco Antônio Pereira Araújo - Orientador
Universidade Federal de Juiz de Fora

_____

Prof. D.Sc. José Maria Nazar David – Coorientador
Universidade Federal de Juiz de Fora

_____

Prof. D.Sc. Regina Maria Maciel Braga
Universidade Federal de Juiz de Fora

_____

Prof. D.Sc. Arilo Claudio Dias Neto
Universidade Federal do Amazonas

AGRADECIMENTOS

Agradeço inicialmente aos meus pais, Neide e Afonso, que SEMPRE me apoiaram, me ensinaram a ter fé. Fé em Deus, mas principalmente fé em mim mesmo e em minha capacidade. São o meu esteio, minha referência. Amo vocês.

À minha mãe-avó e familiares, sempre orando a Deus que me apoiava com as ferramentas necessárias para que eu pudesse trilhar esse caminho que não é fácil. O meu obrigado!

Aos amigos de mestrado, ao pessoal do TeamNEnC, da banda Olar, Renan e à tantos outros que ofereciam os ouvidos para os desabafos. Sempre em conversas regadas a café e muita *zueira* para descontrair e aliviar o estresse. Meu imenso obrigado.

Ao professor Arilo pelas considerações e contribuições para conclusão deste trabalho.

À professora Regina que vinha acompanhando o desenvolvimento deste trabalho ao longo do mestrado, pelas considerações, contribuições e parceria, meu obrigado.

Aos meus orientadores professor Marco Antônio e professor José Maria, pela orientação e ponderações aplicadas nos momentos certos. Meu obrigado.

“Na ciência não existem fracassos,
existem lições aprendidas.”
Claudio Lélis

“Se você quiser alguém em quem
confiar, confie em si mesmo.
Quem acredita sempre alcança!”
Renato Russo

RESUMO

A importância dos softwares nas organizações é crescente. No entanto, para manter seu valor, o software deve ser alterado e atualizado. A manutenção de software depende da alocação de recursos humanos para o cumprimento das atividades de alteração definidas. Entretanto, em um cenário distribuído no qual a colaboração é fundamental para o bom funcionamento das atividades, torna-se uma tarefa não trivial designar desenvolvedores para as atividades de manutenção. Neste contexto, a reputação se torna um elemento chave, afetando os elementos de colaboração, tais como: a coordenação, a cooperação, e a comunicação. Portanto, o acompanhamento da evolução da reputação é importante para promover a colaboração nas atividades de manutenção. A teoria de Dinâmica de Sistemas pode ser aplicada no acompanhamento da evolução da reputação. Através dos dados obtidos, é possível compreender o passado, estabelecer o que ocorre no presente e projetar o comportamento futuro da reputação. Diante disso, este trabalho apresenta um modelo para cálculo da reputação dos desenvolvedores de software, apoiado por técnicas de Dinâmica de Sistemas, o qual permite simular como a reputação se comporta ao longo do tempo. Este modelo serviu de base para a construção de uma infraestrutura para informações de reputação dinâmica, cujo objetivo é possibilitar o gerenciamento e acompanhamento de informações de reputação dos desenvolvedores geograficamente distribuídos de forma a apoiar a alocação desses desenvolvedores às tarefas de manutenção. Além disso, oferece elementos de visualização e colaboração, em um ambiente integrado às atividades de manutenção de software. Uma prova de conceito e um experimento realizados com dados reais de uma empresa são apresentados com o intuito de identificar a viabilidade e aderência do modelo proposto, bem como dos demais recursos oferecidos pela infraestrutura.

Palavras-chave: Reputação, Dinâmica de Sistemas, Colaboração, Visualização de Dados, Manutenção de Software.

ABSTRACT

The importance of software in organizations is growing. However, to maintain its value, the software must be changed and updated. Software maintenance depends on the allocation of human resources to fulfill defined change activities. However, in a distributed scenario in which collaboration is critical to the well running of activities, designate developers for maintenance activities becomes a non-trivial task. In this context, reputation becomes a key element, affecting elements of collaboration, such as: coordination, cooperation, and communication. Therefore, tracking reputation evolution is important to promote collaboration in maintenance activities. The theory of System Dynamics can be applied in monitoring the evolution of reputation. Through the data obtained, it is possible to understand the past, establish what occurs in the present, and project future reputation behavior. Therefore, this work presents a model for calculating the reputation of software developers, supported by System Dynamics techniques, which allows simulating how reputation behaves over time. This model served as the basis for building an infrastructure for dynamic reputation information, which aims to enable the management and tracking reputation information of geographically distributed developers to support the allocation of these developers to maintenance tasks. In addition, it provides visualization and collaboration elements in an integrated environment for software maintenance activities. A proof of concept and an experiment made with real data of a company are presented with the intention of identifying the feasibility and adherence of the proposed model, as well as of the other resources offered by the infrastructure.

Keywords: Reputation, System Dynamics, Collaboration, Data Visualization, Software Maintenance.

LISTA DE FIGURAS

LISTA DE TABELAS

Sumário

1. INTRODUÇÃO

As organizações fazem grandes investimentos em seus sistemas de software. Constituem importantes ativos de negócio. No entanto devem ser atualizados constantemente para manter seu valor. Diante disso e da necessidade de manter a produtividade e a qualidade, fizeram com que as empresas distribuíssem geograficamente suas atividades. O aprimoramento dos meios de comunicação e a popularização da Internet aproximaram consumidores de fornecedores, facilitaram a distribuição do conhecimento bem como a formação de comunidades virtuais que trabalham, por meio desses recursos, colaborativamente. Neste contexto, cria-se um cenário no qual produtores de software procuram, além de suas fronteiras geográficas, por recursos mais especializados ou com melhores custos. Dessa forma, os recursos especializados passaram a ser alocados ao processo de produção, mesmo estando eles distribuídos geograficamente. Surge, assim, a abordagem de Desenvolvimento Distribuído de Software (DDS) na qual os recursos humanos distribuídos geograficamente (programadores, projetistas, clientes, entre outros) executam colaborativamente suas atividades dentro do processo de desenvolvimento (AUDY; PRIKLADNICKI, 2007).

A Fase de Manutenção de Software é das mais problemáticas do Ciclo de Vida do Software (SOMMERVILLE, 2015). Embora as alterações sejam inevitáveis, os sistemas devem permanecer em produção. A manutenção pode despender mais de 70% de todo esforço de uma organização, e ao serem comparados, os custos de manutenção geralmente excedem os custos do desenvolvimento (PRESSMAN; MAXIM, 2014). Por serem afetados por fatores técnicos e não técnicos (as pessoas envolvidas, por exemplo) esses custos aumentam conforme o software é mantido. Além disso, a manutenção pode corromper a estrutura do software, tornando a manutenção posterior mais difícil (SOMMERVILLE, 2015). Por exemplo, quando uma alteração é realizada em um trecho de código, isso pode conduzir à necessidade de alterar também outros trechos de código, sejam classes (TAVARES et al., 2015) ou métodos (LÉLIS et al., 2016c).

Dentre os fatores não técnicos que afetam o custo da manutenção, um deles é a estabilidade da equipe. Segundo Sommerville (SOMMERVILLE, 2015), os custos são reduzidos se o mesmo grupo estiver reunido há a algum tempo. Por outro lado, equipes

com alta taxa de *turnover* recebem, com frequência, novos membros desconhecidos e assim, a inexperiência e o conhecimento limitado de domínio pode também elevar os custos da manutenção.

Em um cenário distribuído onde a colaboração é fundamental para o bom funcionamento das atividades (AUDY; PRIKLADNICKI, 2007), tais problemas anteriormente citados se somam a diversos aspectos organizacionais, técnicos e interpessoais que influenciam na condução de projetos distribuídos. Para ilustrar essa situação, é possível citar outro problema clássico da manutenção de software, relativo à dificuldade de compreensão do código ou artefato desenvolvido e mantido por outro desenvolvedor (SOMMERVILLE, 2015). Nesse caso, os aspectos organizacionais e técnicos (PRIKLADNICKI et al., 2004), como o uso de processos distintos e de maturidades distintas, bem como o atendimento a padrões, podem tornar a compreensão mais difícil. Cada desenvolvedor possui uma forma de programar diferente, por exemplo, ao atribuir nomes pouco representativos às variáveis. Com relação aos aspectos interpessoais e não técnicos (PRIKLADNICKI et al., 2004), a diversidade cultural e de idiomas, bem como as diferenças de fusos horários e a própria distância, agravam o problema já que a oportunidade de encontros presenciais é escassa, ou ausente. Neste tipo de cenário, os comportamentos dos parceiros bem como suas perspectivas estão mais sujeitas a más interpretações.

A distribuição traz consigo uma série de desafios quase sempre relacionados à falta de meios de comunicação mais ricos e informais, com problemas de coordenação e falta de coesão entre os grupos envolvidos (AUDY; PRIKLADNICKI, 2007). Quando a distribuição de trabalho acontece entre regiões, aumenta a preocupação quanto à coordenação das atividades, mais precisamente quanto a atribuições de tarefas, monitoramento das atividades e gerenciamento de suas dependências (PRIKLADNICKI et al., 2004). Os aspectos não técnicos da manutenção, bem como os aspectos organizacionais e interpessoais relativos ao cenário distribuído, transformam em tarefa não trivial designar desenvolvedores para as atividades de manutenção.

A integração e aquisição de confiança entre as partes distribuídas configura solução possível a ser aplicada, visando minimizar os problemas dos aspectos interpessoais (PRIKLADNICKI et al., 2003) (PRIKLADNICKI et al., 2004). Entende-se aqui como confiança o grau de credibilidade que um indivíduo/entidade atribui a outro quanto à apresentação de um comportamento desejado (JØSANG, 2007). Durante

a interação, as partes envolvidas podem não conhecer uns aos outros, nem tampouco ter informações sobre os seus contextos (LÉLIS et al., 2016b). Quando isso ocorre e os indivíduos a serem selecionados são desconhecidos para o recrutador, é fundamental ter um mecanismo que subsidie o estabelecimento da confiança de que estes são bons desenvolvedores. Em sistemas como *Stack Overflow*[1] e *GitHub*[2], utilizados pelos desenvolvedores nas tarefas de codificação e gerenciamento de versão, respectivamente, a reputação é fornecida como um balizador para estabelecer a confiança inicial entre indivíduos desconhecidos.

A confiança pode ser encarada como a credibilidade que um indivíduo atribui a outro indivíduo, grupo ou organização na realização de determinada ação ou na manifestação de característica específica (AL-ANI; REDMILES, 2009). Essa confiança é influenciada pelas experiências ocorridas diretamente entre os indivíduos, pela recomendação de terceiros e pela reputação que cada um possui perante o grupo. Como exemplo, pode-se citar um cenário no qual um gerente de projetos deposita confiança na capacidade de um desenvolvedor de sua equipe em entregar a refatoração de um módulo, em um sistema, no prazo acordado e a execução dos trabalhos depende dessa entrega. A alocação de alguém que não tenha credibilidade pode impactar na entrega, no atendimento dos prazos e aumentar os custos da manutenção. No entanto, o fato de o gerente ter confiança no desenvolvedor indica a sua credibilidade na realização da ação determinada (TRAINER; REDMILES, 2012). Esse fato reflete em uma boa reputação que o desenvolvedor possui.

Assim, a reputação, através da confiança estabelecida, torna-se um elemento importante para as atividades de colaboração, com influência na comunicação. Trainer e Redmiles (TRAINER; REDMILES, 2012), por exemplo, investigaram a forma pela qual desenvolvedores e gerentes usam múltiplos meios de comunicação, ou seja, se estão disponíveis através de *chat* e são ativos na lista de discussão do projeto ou se dispõe à comunicação apenas via e-mail. Como resultado, esses desenvolvedores e gerentes ativos e disponíveis podem ser bem vistos pelos demais membros e, consequentemente, alcançam boa reputação por serem considerados confiáveis. Nesse

[1] http://pt.stackoverflow.com/

[2] https://github.com/

caso por exemplo, designar um desenvolvedor que não está disposto a colaborar pode levar a equipe a falhar completamente (AUDY; PRIKLADNICKI, 2007).

Reputação pode também ser aplicada como indicador de cooperação. Wang e Redmiles (WANG; REDMILES, 2013) utilizam a teoria de jogos para entender o surgimento de confiança em equipes globalmente distribuídas. Quando os parceiros recusam a sua cooperação, o desenvolvedor é penalizado com a perda de sua reputação. Isso reflete negativamente na execução das tarefas que a equipe deve desempenhar afetando a coesão entre seus membros. De modo semelhante ao exemplo anterior, a alocação de um desenvolvedor propenso a cooperação torna a equipe mais coesa (JARVENPAA et al., 2004), o que configura fator de sucesso nos projetos (PRIKLADNICKI et al., 2004).

Em essência, os dados de reputação podem servir de base aos usuários para decidirem quem escolher para interagir, em quem confiar, e em que grau (HENDRIKX et al., 2015). Para isso, contam com os Sistemas de Reputação, empregados para facilitar a coleta, agregação e distribuição de informações de reputação sobre uma entidade (HENDRIKX et al., 2015). Considerando o contexto de desenvolvimento de software, mais especificamente no gerenciamento da manutenção e evolução do software, sistemas como esses poderiam ser utilizados, também, para apoiar gerentes na formação de grupos ou alocação de desenvolvedores às tarefas de manutenção como, por exemplo, atender uma solicitação de mudança ou efetuar uma refatoração de código.

Para um acompanhamento adequado das atividades a que um sistema de reputação se propõe, uma quantidade significativa de dados deve ser coletada, processada e armazenada ao longo do tempo. Estes dados devem ser apresentados em uma interface que promova interação com o usuário. Um Ambiente Interativo baseado em Múltiplas Visões (AIMV) (CARNEIRO et al., 2012) oferece recursos de interação e mecanismos de visualização para apoiar a análise de dados e também a descoberta e o reconhecimento de informação relevante aos gestores. Um exemplo, seria a composição de diferentes visualizações, na qual as informações se complementam.

Sistemas desse tipo deveriam também permitir o acompanhamento da evolução da reputação para que se tomem decisões no sentido de corrigir problemas ou evitar seu agravamento. Adicionalmente, devido à evolução e manutenção constantes em um software, torna-se fundamental a construção de teorias e modelos que permitam a

compreensão do passado, presente e futuro do ciclo de manutenção (PAREDES et al., 2014).

1.1 PROBLEMA

Diante desse cenário, da motivação e a contextualização de pesquisa apresentados, o problema tratado neste trabalho se refere à alocação de desenvolvedores em tarefas de manutenção colaborativa e evolução de software, no contexto de equipes geograficamente distribuídas.

Um maior conhecimento sobre o processo de aquisição de confiança e estabelecimento da credibilidade torna-se necessário, no sentido de facilitar a assimilação sobre a reputação dos membros de uma equipe sob o ponto de vista do grupo ou do domínio ao qual esses membros estão associados. Nessa perspectiva, a construção de um modelo que permita observar os principais critérios que podem levar ao crescimento ou decrescimento da reputação, simulando seu comportamento, intenciona contribuir para um maior entendimento do seu processo evolutivo.

A análise de dados absolutos coletados de empresas reais, com seus respectivos desenvolvedores e equipes, através de métricas pré-estabelecidas e relacionadas a esses critérios de reputação não representa necessariamente o comportamento futuro da reputação desses desenvolvedores. Na verdade, permite observar o comportamento passado, já ocorrido. Assim, tendências de crescimento ou diminuição dos critérios de reputação podem oferecer informações relevantes sobre o comportamento da reputação de cada desenvolvedor ao designá-los às tarefas de manutenção.

1.2 ENFOQUE DA SOLUÇÃO

A solução proposta neste trabalho possui enfoque no suporte a equipes de desenvolvimento que conduzem as atividades de manutenção colaborativa de forma distribuída. Este trabalho tem como foco a dimensão humana, o aspecto da reputação dos desenvolvedores do time e que, no enfoque considerado, estarão geograficamente distribuídos.

A reputação mencionada será tratada através de uma abordagem multicritério, apoiada por um AIMV (Ambiente Interativo baseado em Múltiplas Visões). A teoria de

Dinâmica de Sistemas é aplicada através dos modelos dinâmicos para a construção da reputação considerando-se dados de entrada a serem analisados. Tal fato possibilita a projeção e acompanhamento da evolução dos valores de reputação.

1.3 OBJETIVO

Neste contexto, é importante investigar o provimento de informações relativas à reputação de desenvolvedores, uma vez que esse é um aspecto importante para melhorar a colaboração no contexto das equipes de desenvolvimento que trabalham distribuídas e de forma colaborativa. Conforme ilustrado pelos exemplos anteriormente discutidos nesta seção, oferecer recursos e mecanismos de visualização para apoiar a análise de tais informações, tem potencial para melhorar a descoberta e o reconhecimento de informação relevante aos gestores, e consequentemente à tomada de decisão para a alocação de desenvolvedores geograficamente distribuídos.

O objetivo principal deste trabalho é tratar a alocação de recursos humanos em tarefas de manutenção colaborativa de software através de uma infraestrutura (solução) de suporte à reputação que permita o gerenciamento e acompanhamento das informações de reputação em um contexto de manutenção colaborativa de software.

Assim, este trabalho apresenta a pesquisa realizada para, inicialmente, avaliar os mecanismos de reputação existentes, em seguida definir os critérios para cálculo de reputação e assim, projetar e desenvolver uma infraestrutura, baseada em um modelo dinâmico, para observação do comportamento da reputação de desenvolvedores de software.

Em virtude da característica dinâmica do trabalho em equipe, influenciando a colaboração entre os envolvidos, essa infraestrutura deve ser apoiada por um sistema de reputação, no sentido de coletar e distribuir essas informações de reputação aos usuários finais interessados, os gestores e demais desenvolvedores. Essa infraestrutura deve integrar também ferramentas capazes de interpretar e simular tendências de determinados critérios de reputação, como expertise, relacionamento ou confiabilidade, que possam afetar outros critérios e determinar sua evolução com o tempo.

A solução proposta está no contexto de um AIMV com ferramentas que apoiam a colaboração entre equipes geograficamente distribuídas nas tarefas de manutenção e evolução de software. Em (SILVA et al., 2012) é dito que AIMVs fornecem meios para

que análises sobre dados sejam feitas a partir da utilização de visualizações, e que tais ambientes devem fornecer meios para que haja a coordenação entre as visualizações. Também é dito que, combinar diferentes visualizações pode permitir a análise de dados em diferentes perspectivas. A interatividade fica por conta da definição de recursos que permitem interagir diretamente nas visualizações fornecidas como, por exemplo, através do envio de mensagens entre membros da equipe.

Diante do objetivo deste trabalho, foi estabelecida uma parceria com uma empresa de desenvolvimento de software, que será chamada neste trabalho de Empresa Parceira. O objetivo é verificar a viabilidade de uso da solução em um contexto real e seu apoio ao gerenciamento de recursos da equipe, tais como alocação a tarefas de manutenção.

1.4 ORGANIZAÇÃO DA DISSERTAÇÃO

Este trabalho é organizado em mais cinco capítulos além desta introdução, que apresentou a motivação e o contexto no qual está inserida esta dissertação. Sua organização textual segue a estrutura abaixo:

- **Capítulo 2 – Referencial Teórico:** são abordados os principais aspectos relativos à Reputação, Sistemas de Reputação, Visualização, Colaboração e Manutenção de Software, Dinâmica de Sistemas, e apresentados estudos relacionados encontrados na literatura técnica especializada.
- **Capítulo 3 – Modelo de cálculo de reputação para desenvolvedores de Software:** é apresentado o modelo de reputação, resultado do levantamento dos critérios para cálculo de reputação, suas fórmulas e as influências entre si.
- **Capítulo 4 – Infraestrutura de reputação para desenvolvedores de Software:** é tratada a questão relativa à arquitetura da proposta, apresentando-se os recursos oferecidos pela infraestrutura, ferramentas integradas e o fluxo das informações entre os módulos que compõem a infraestrutura.
- **Capítulo 5 – Avaliação da infraestrutura de reputação:** uma prova de conceito e um experimento são apresentados e exploram a viabilidade e aderência do modelo proposto, bem como dos demais recursos oferecidos pela infraestrutura.

- **Capítulo 6 – Conclusão e Perspectivas Futuras:** são apresentadas as conclusões e contribuições do trabalho, além de indicada a continuação da pesquisa através de trabalhos futuros.

2. REFERENCIAL TEÓRICO

Para contextualizar o problema, bem como a solução proposta, faz-se necessária uma abordagem teórica sobre os principais assuntos que permeiam este trabalho. Nas seções seguintes serão apresentados os conceitos teóricos na pesquisa sobre manutenção distribuída de software, visualização, colaboração, reputação, sistemas de reputação e Dinâmica de Sistemas. Além de abordar esses temas, este capítulo também destina-se a fornecer uma visão geral das relações existentes entre esses conceitos.

2.1 MANUTENÇÃO DE SOFTWARE

A manutenção, concentrando grande parte dos recursos de desenvolvimento do software, possui diversas abordagens que foram experimentadas com intuito de produzi-lo eficientemente, dentro das expectativas relativas ao prazo, custo e qualidade. A estratégia na qual os recursos humanos envolvidos em sua produção estão geograficamente dispersos, dá-se o nome de Desenvolvimento Distribuído de Software (DDS) (AUDY; PRIKLADNICKI, 2007).

Apesar dos problemas levantados no capítulo anterior quanto à distribuição das atividades, um fator motivador para utilização de recursos remotos é a redução de custos tanto de produção bem como de identificação de programadores especialistas em determinada tecnologia necessária ao público-alvo do software a ser produzido. Uma organização pode terceirizar, por exemplo, a atividade de codificação e testes para uma fábrica de software remota, quando essa oferece menores custos de produção com equipe especializada.

Embora sejam alcançados benefícios com a distribuição das atividades relativos à redução do custo do produto e flexibilidade para seu desenvolvimento, o software deve inevitavelmente mudar após sua implantação para continuar útil (SOMMERVILLE, 2015). Com o passar do tempo, as necessidades e interesses dos clientes mudam, leis podem surgir e os negócios podem tomar rumos diversos levando à obrigatoriedade de novas mudanças. Melhoria de desempenho, adaptação a uma nova plataforma e defeitos encontrados durante a operação do software também levam a mudanças. Tais mudanças que ocorreram em um software ao longo do seu ciclo de

manutenção, bem como os impactos causados, constituem o processo de evolução de um software (LEHMAN, 1996).

Pressman (PRESSMAN; MAXIM, 2014) define a manutenção de software como um processo geral de mudanças de um sistema depois que é entregue, e classifica-o em quatro tipos de manutenção de software. A manutenção para reparo de defeitos, ou também manutenção corretiva, é o primeiro deles, e defeitos podem ser tanto erros na codificação, no projeto, ou ainda na elicitação dos requisitos dos clientes. O segundo tipo é a manutenção para adaptar o software a um ambiente operacional diferente ou simplesmente manutenção adaptativa. É necessária quando o hardware, o sistema operacional ou outro software de apoio são alterados obrigando o sistema a se adequar. A manutenção para adicionar funcionalidade ao sistema, chamada manutenção evolutiva, ocorre quando mudanças organizacionais ou de negócios são detectadas. Por fim, a manutenção preventiva, para tratar modificações feitas no software como forma de prevenção de possíveis problemas no futuro.

A norma (ISO/IEC 14764, 2006) apresenta definição na qual considera manutenção de software as atividades realizadas após o desenvolvimento e entrega de um produto objetivando a correção de falhas detectadas, a melhoria de desempenho e outros atributos, ou adaptação do produto às novas exigências do ambiente, como por exemplo, mudanças na legislação.

O processo de controle de mudança descrito por Sommerville (SOMMERVILLE, 2015) se aproxima da realidade e do enfoque tratado neste trabalho. É iniciado descrevendo a mudança necessária para o sistema através de um formulário de solicitação de mudança (*CR – Change Request*). Nesse formulário registram-se as recomendações sobre a mudança, os custos estimados e as datas de quando foi solicitada, aprovada, implementada e validada. Pode ainda incluir uma seção onde o analista descreve como a mudança será implementada. Uma vez recebida a solicitação de mudança, deve ser registrada no banco de dados de configuração. O passo seguinte é verificar se a mudança solicitada é de fato necessária, e não proveniente de um mal-entendido, ou defeito já conhecido, sendo rejeitada nesses casos. Será rejeitada também ao ser julgada inválida, duplicada, ou já tratada. Os interessados são notificados sobre os motivos da rejeição.

Na gerência de mudanças, ferramentas de código aberto como *Bugzilla*[3] e *Mantis Bug Tracker*[4] são opções para redução dos custos. De forma geral, Sommerville (SOMMERVILLE, 2015) descreve recursos que as ferramentas buscam prover, tais como: (i) um editor de formulários permitindo a criação dos formulários com propostas de mudança; (ii) um sistema de fluxo de trabalho (*workflow*) que permite definir quem deve processar o formulário de mudança e sua ordem de processamento. Também é função desse sistema informar aos interessados, no tempo e hora certa, o progresso da mudança. Outro recurso é (iii) um banco de dados, usado para gerenciar todas as propostas de mudança com um sistema de consulta que permite encontrar uma solicitação de mudança específica. Há ainda (iv) um sistema de relato de mudanças que gera relatórios gerenciais.

Diante disso, o planejamento nas diversas fases de manutenção do software pode gerar informações quantitativas e qualitativas sobre o desempenho da equipe e do projeto em andamento. Tais informações podem gerar indicadores para futuras manutenções de software. Assim, as requisições de mudanças servirão de insumos para a análise histórica de dados realizada por esta pesquisa.

A manutenção, quando realizada por equipes distribuídas, pode ser analisada como atividade colaborativa, na qual diferentes envolvidos podem necessitar de diferentes perspectivas sobre a informação analisada em conjunto com outras pessoas. Os aspectos de visualização, discutidos na próxima seção, podem auxiliar equipes distribuídas na seleção de informação relevante e descoberta de conhecimento que não seriam facilmente conhecidos se examinados diretamente os repositórios que contêm tais informações (STOREY et al., 2008).

2.2 VISUALIZAÇÃO DE INFORMAÇÃO

A visualização de informação pode ser referenciada como parte de um processo de compreensão que objetiva atingir um conhecimento aprofundado a respeito de um tema a partir de um conjunto de dados. Além disso, é fundamental para se criar um modelo

[3] https://www.bugzilla.org

[4] https://www.mantisbt.org

mental consistente sobre as informações a respeito de uma determinada situação ou realidade (SPENCE, 2007).

Segundo Wang-Baldonado et al. (2000) visualizações podem diferir em seus dados ou na representação visual dos dados. Independentemente se os conjuntos de dados diferem, representações visuais podem variar. Por exemplo, uma visualização pode mostrar um gráfico de barras, enquanto outra visão mostra um gráfico de dispersão. Para a análise dos dados, pode-se selecionar uma única metáfora visual, ou uma agregação de metáforas disponibilizadas em um ambiente interativo baseado em múltiplas visões (AIMV). AIMVs são ambientes que fornecem diferentes visualizações para o mesmo conjunto de informações, permitindo também que dados de interação entre usuários sejam registrados (SILVA et al., 2012).

A visualização única de um objeto pode ser entendida como um conjunto de dados, além de uma especificação de como exibir esses dados visualmente. Note-se que o formato de exibição pode ser tanto textual, por exemplo, em forma de tabela, ou gráfica, por exemplo, num gráfico de barras (WANG BALDONADO et al., 2000). Se diz, ainda, que duas ou mais dessas visualizações são diferentes se permitirem ao usuário aprender sobre os aspectos do objeto de interesse, por exemplo, através da apresentação de informações distintas, ou ao enfatizar aspectos diversos da mesma informação. Um sistema de múltiplas visões utiliza duas ou mais dessas visões distintas para apoiar a investigação de um determinado objeto e para minimizar o surgimento de interpretações equivocadas, se comparado com a análise realizada com uma visão única (AINSWORTH, 1999).

A visualização de software remete à análise de informações inerentes ao software. Visualizações podem ser utilizadas para analisar a evolução de um mesmo conjunto de dados ao longo dos anos, como de um software que evolui com o tempo. Um exemplo seria a visualização da contribuição dada por cada desenvolvedor em um projeto de software ao longo dos anos. Técnicas de visualização de software podem expor informações sobre os responsáveis bem como os artefatos produzidos e, quando combinadas com múltiplas visões, podem exibir diferentes aspectos da informação compartilhada. Há ainda a possibilidade de se criar compartilhamentos de informações de forma diferenciada, de tal modo que detalhes da informação sejam ocultados ou exibidos apenas para um grupo determinado de pessoas. Como exemplo pode-se ter

informações a nível gerencial e financeiro que não devem ser visualizadas pelo pessoal que está no nível operacional.

O termo visualização colaborativa surgiu para denominar a fusão das áreas de visualização e colaboração e que, por sua natureza interdisciplinar, abre espaço para novos desafios de pesquisa (ISENBERG et al., 2011).

A definição elaborada por Isenberg et al. (2011) estabelece que visualização colaborativa é o uso compartilhado e interativo de representações visuais de dados por mais de uma pessoa, apoiado por computador, com o objetivo comum de contribuir para as atividades de processamento de informação conjunta. Melhora a visualização tradicional, ao reunir muitos especialistas, de modo que cada um pode contribuir para o objetivo comum de compreensão do objeto, fenômeno, ou dados sob investigação. Além disso, permite que os usuários geograficamente separados acessem um ambiente compartilhado para visualizar e manipular conjuntos de dados para a resolução de problemas sem ter que se deslocarem fisicamente.

O entendimento sobre a divisão de tarefas entre equipes distribuídas foi aperfeiçoado com o surgimento de elementos como cooperação, comunicação e coordenação. Tais elementos são assunto da próxima seção.

2.3 COLABORAÇÃO EM SOFTWARE

A colaboração é fundamental para o bom funcionamento das atividades distribuídas de equipes de manutenção que desejam colaborar, devendo se organizar e focar em elementos como Cooperação, Comunicação e Coordenação (FUKS et al., 2003).

Cooperação é a ação em conjunto entre membros de uma equipe que compartilham o mesmo espaço de trabalho (FUKS et al., 2003), no caso, o mesmo espaço do ambiente de colaboração, objetivando realizarem alguma tarefa em parceria. Um exemplo envolvendo manutenção colaborativa seria a realização de atividades (como reuniões), através da qual há cooperação para se obter entendimento sobre os motivos de um desenvolvedor acumular altos índices de reabertura de defeitos. O trabalho cooperativo nesse caso poderá envolver ainda aspectos da comunicação.

Comunicação diz respeito à efetividade com a qual uma mensagem é entregue e entendida pelo destinatário (FUKS et al., 2003). Uma falha na comunicação pode prejudicar a tomada de uma decisão, fazendo com que o destinatário não compreenda a

intenção da comunicação estabelecida e, assim, tome uma decisão equivocada. Um exemplo seria a realização de atividades nas quais há troca de informações entre membros da equipe e o gerente de projeto, sobre uma tarefa de manutenção em um módulo que apresenta alta taxa de complexidade e qual desenvolvedor deveria tratá-la.

Coordenação é a tarefa de articular um grupo de pessoas trabalhando com um objetivo comum, em um ou vários processos que se conectam de alguma forma. Coordenar, portanto, é articular a pré-realização de uma tarefa, a realização da tarefa e a pós-realização da tarefa (FUKS et al., 2003). Um exemplo seria a realização de atividades de preparação do tema de uma reunião, convite dos participantes, condução da reunião rumo à tomada de decisões e delegação das decisões tomadas aos responsáveis por executá-las.

A Percepção pode ser considerada um elemento de colaboração, dado que diz respeito às informações capazes de fornecer o contexto das interações realizadas entre membros de uma equipe. Um exemplo pode ser o recurso que permite ao usuário obter conhecimento de que uma mensagem enviada por ele, sobre a finalização de uma atividade de manutenção, foi visualizada pelo destinatário. A colaboração pode existir através do envio e recebimento de mensagens entre membros de uma equipe de diferentes formas, como exemplo, através de mensagens de correio eletrônico ou mensagens inseridas diretamente em visualizações, disponíveis no software ou no *framework* de suporte ao desenvolvimento da equipe.

Como já discutido no capítulo anterior, a colaboração nas atividades realizadas por equipes de manutenção são influenciadas pela confiança depositada entre os membros, bem como a reputação adquirida (WANG; REDMILES, 2013) (TRAINER; REDMILES, 2012). Devido à sua importância e ao fato de constituir a base para a solução proposta neste trabalho, aspectos de reputação são apresentados na próxima seção.

2.4 REPUTAÇÃO

A reputação é um importante ativo uma vez que afeta a colaboração em equipes tanto distribuídas como co-localizadas. Segundo Mui et al. (MUI et al., 2002) a reputação pode ser entendida como a percepção que um agente (entidade) cria por meio de ações anteriores sobre suas intenções, regras e crenças. Considerando o contexto de um

desenvolvedor desempenhando tarefas de manutenção, sua reputação seria a percepção criada mediante as tarefas anteriormente tratadas. Os dados passados fornecem a reputação atual do desenvolvedor. Por meio desse conceito, se percebe a possibilidade da reputação mudar com o tempo.

Reputação pode, ainda, ser considerada como uma medida coletiva de confiabilidade com base nas referências ou classificações de membros de uma comunidade. Confiança subjetiva de um indivíduo pode ser derivada de uma combinação de referências recebidas e a experiência pessoal (JØSANG et al., 2007). Adicionalmente, a reputação pode variar conforme o contexto. Como exemplo, um analista de sistemas pode ter ótima reputação como programador dentro de uma equipe de desenvolvimento. Porém, ao mesmo tempo, sua reputação como analista de requisitos pode ser ruim.

Reputação pode relacionar-se a um grupo ou a um indivíduo. A reputação de um grupo pode, por exemplo, ser modelada como a média de todas as reputações individuais dos seus membros, ou através da média de como o grupo é percebido pelas entidades externas (JØSANG et al., 2007). Dessa forma, a reputação do grupo pode indicar a expectativa de reputação de um de seus membros.

A construção da reputação é gerenciada pelos Sistemas de Reputação, tema da próxima seção.

2.5 SISTEMAS DE REPUTAÇÃO

Os sistemas de reputação são empregados para facilitar a coleta, agregação e distribuição de informações de reputação sobre uma entidade (HENDRIKX et al., 2015). Alguns termos têm sido amplamente utilizados pelos pesquisadores para representarem as entidades que estão envolvidas em Sistemas de Reputação. Liu e Munro (2012) esclarecem esses termos:

- *Fonte de informação:* fornece informações para os sistemas de reputação. Tal informação pode ser fornecida por uma pessoa, isto é, um avaliador. Em outros casos, a fonte de informações pode ser um sistema. Por exemplo, alguns sistemas de reputação coletam informações de outros sistemas, em vez de avaliadores.

- *Alvo:* refere-se à entidade avaliada, para a qual avaliadores fornecem informações. Um alvo pode ser um produto, pessoa, uma transação ou mesmo um histórico.
- *Usuário final:* utiliza sistemas de reputação para a busca de informações sobre um alvo. Os usuários finais podem ser, por exemplo, os visitantes de um site, enquanto que os avaliadores podem ou não ser visitantes.
- *Informação de Reputação:* referem-se a informações relacionadas à reputação de um alvo, tais como comentários ou avaliações de um produto. Sistemas de reputação, normalmente, coletam dois tipos principais de informação de reputação: informação explícita e informação implícita.
    - *Informação explícita:* indica a informação que os avaliadores fornecem ativamente, tais como uma classificação ou um comentário de texto.
    - *Informação implícita:* é normalmente derivada a partir das atividades do avaliador. Por exemplo, o número total de visualizações de um vídeo.

Além dos termos, Liu e Munro (2012) definem cinco componentes que todos os sistemas de reputação devem ter, independentemente de suas interfaces, funções ou papéis. São eles:

- *Entrada*: é o processo de coleta de informações de reputação a partir de fontes de informação. Refere-se à coleção de classificações, comentários de texto e outras informações de reputação relevantes. É um dos componentes mais importantes porque os outros quatro componentes dependem da informação coletada a partir da entrada.
- *Processamento*: é o procedimento de computação e agregação da informação de reputação, ou seja, um conjunto de atividades que transformam a informação pura em uma forma mais significativa.
- *Saída*: indica a divulgação das informações de reputação, ou seja, sua função é relatar e divulgar as informações de reputação. Existem dois tipos principais de informação que um sistema de reputação precisa relatar: informação agregada e informação individual. O objetivo de proporcionar a informação agregada de reputação é apresentar a reputação geral do alvo em um formato conciso e comparável. Já a informação individual é a informação que é fornecida por cada avaliador. Isso é necessário aos sistemas para apresentar a informação pura, tal

qual foi coletada. Além disso, sistemas de reputação também precisam fornecer mais informações sobre os avaliadores e o *feedback* das avaliações.

- *Ciclo de feedback*: é a coleta de *feedback* de uma avaliação que pode ser visto como a "revisão da avaliação". O conteúdo da avaliação é um fator importante para o sistema de reputação.
- *Armazenamento*: se refere ao processo de armazenar todas as informações que foram coletadas e processadas ou geradas por outros componentes. Sistemas de reputação podem armazenar informações de modo centralizado ou distribuído. Existem três tipos de informações que precisam ser armazenados: a informação coletada a partir da entrada, as informações recolhidas a partir do ciclo de *feedback* e as informações geradas pelo processamento.

São apresentas, na Figura 2.1, as relações entre os cinco componentes. Através dela observa-se que a informação de reputação flui a partir de fontes de informação para o componente de processamento. Depois de ser agregada, será publicada. Se os usuários finais estiverem interessados, podem ser autorizados a deixar *feedback* (as linhas pontilhadas indicam que o ciclo de *feedback* é um componente opcional). Durante todo o processo, toda a informação tem de ser armazenada no componente de armazenamento.

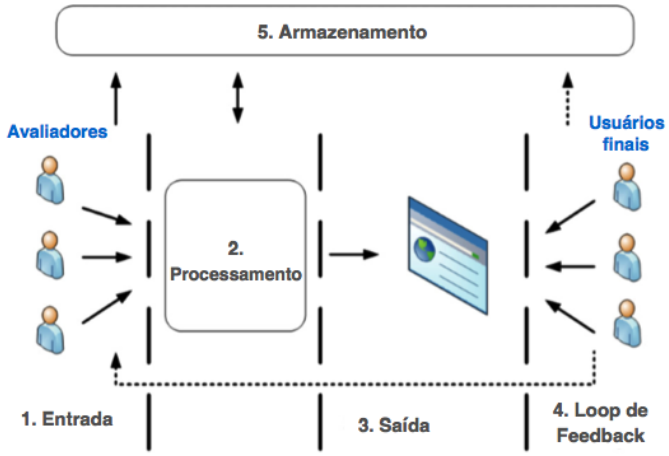

Figura 2.1 - Modelo Estrutural dos Sistemas de Reputação (LIU; MUNRO, 2012)

Segundo Josang et al. (2007), um Sistema de Reputação é qualificado sobre dois aspectos técnicos: a sua arquitetura de rede e o modelo para motor de reputação. A função da arquitetura é determinar como se dará o fluxo de informações no sistema, ou seja, como os dados necessários para a construção da reputação são coletados e como a

reputação gerada é fornecida aos interessados. Já o motor de reputação tem a função de processar os dados relevantes à reputação, agregá-los e disponibilizá-los para consumo.

Há dois tipos principais para arquitetura de rede: a centralizada e a distribuída (JØSANG, 2007). Sistemas de Reputação centralizados possuem um ponto central (servidor de reputação) para o qual as qualificações sobre um indivíduo são enviadas, processadas e disponibilizadas para toda a comunidade. Nessa arquitetura, a reputação de um alvo é construída com base em todas as qualificações existentes sobre ele no contexto considerado. Assim, cada usuário final que quiser obter a reputação de um alvo específico, deverá requisitar esta informação ao servidor de reputação.

Outro aspecto relevante ao se construir uma solução de Sistema de Reputação é a definição de um modelo para o Motor de Reputação. A função de um motor de reputação é processar as qualificações dos indivíduos que estão registradas no Sistema de Reputação, efetuando todo o tratamento necessário, fornecendo uma informação sobre a reputação dos indivíduos (JØSANG et al., 2007).

Sobre o processamento das qualificações existem algumas questões que são comuns de serem tratadas pelos motores de reputação. A primeira é a definição de como será a reputação de um indivíduo novo no grupo, ou seja, que não tenha ainda qualificações registradas. Uma alternativa é atribuir-lhe uma reputação inicial que represente uma média das reputações de todos os indivíduos de uma comunidade (JØSANG et al., 2007) ou que se assemelhe ao seu perfil. À medida que os indivíduos interajam com ele, esta reputação evolui para um valor que representa melhor o seu comportamento. Uma alternativa seria não atribuir informação de reputação ao novo indivíduo e aguardar que tenha uma quantidade pré-determinada de interações para efetuar o cálculo da reputação (JØSANG, 2007).

Outra questão a ser tratada é o envelhecimento das qualificações de um indivíduo. Pode ser conveniente que qualificações mais antigas tenham um peso menor no cálculo da reputação. Assim, evita-se, por exemplo, que indivíduos permaneçam com baixos índices, mesmo após modificarem sua atitude (JØSANG; QUATTROCIOCCHI, 2009).

No entanto, ao passar do tempo, o caráter dinâmico das interações entre avaliadores e alvos, e com a chegada de novas qualificações, podem ser aplicadas técnicas para entender e influenciar a forma com que os elementos de um sistema variam ao longo do tempo. Modelos dinâmicos são resultado da adoção de tais técnicas.

Os aspectos relativos aos modelos e a área de Dinâmica de Sistemas são discutidos na próxima seção.

2.6 DINÂMICA DE SISTEMAS

Dinâmica de Sistemas é uma disciplina de modelagem cujas técnicas foram aplicadas inicialmente para compreender a dinâmica industrial (FORRESTER, 1961) quase exclusivamente em problemas de corporações e gerenciamento. Em seguida, para a dinâmica urbana (FORRESTER, 1970) e também para o sistema socioeconômico mundial (FORRESTER, 1971). No contexto da manutenção e evolução de software, Araújo (2009) aplicou modelos dinâmicos em conjunto às Leis de Evolução de Software (LEHMAN, 1996) para compreender o comportamento do decaimento do software. Nesse sentido, aplicar modelos dinâmicos também seria útil para observar o comportamento da reputação de desenvolvedores e designar tarefas com base em seu comportamento futuro.

Os modelos desenvolvidos com as técnicas da Dinâmica de Sistemas utilizam conceitos de controle com retroalimentação (ciclos de *feedback*) para organizar as informações disponíveis a respeito de um sistema, criando modelos que podem ser simulados em um computador (FORRESTER, 1993). Podem ser representados através de diagramas de causa e efeito ou através de diagramas de repositórios e fluxos. Os diagramas de causa e efeito são o mecanismo mais simples para representação de modelos de Dinâmica de Sistemas. Posteriormente, esses diagramas podem ser refinados para diagramas de repositórios e fluxos, que são mais precisos na representação do modelo e apresentam um nível de detalhe maior que os diagramas de causa e efeito, forçando o responsável pela modelagem a refinar sua definição da estrutura do sistema (ALBIN et al., 2001).

Para apoiar a execução de modelos baseados em Dinâmicas de Sistemas, pode-se utilizar uma ferramenta específica para esse fim. Ferramentas como *InsightMaker*[5], *Vensim*[6] e *Simantics System Dynamics*[7] permitem a modelagem dos diagramas de causa

---

[5] https://insightmaker.com

[6] http://vensim.com

[7] http://sysdyn.simantics.org

e efeito bem como a modelagem em diagramas de fluxo e estoque, e a execução de tal modelo.

Uma das motivações de se produzir modelos através das técnicas da Dinâmica de Sistemas é a capacidade de descrever características que não são facilmente expressas em outros modelos. Barros (2001) descreve tais características conforme mostrado a seguir.

Comportamento Endógeno: a Dinâmica de Sistemas assume que o comportamento de um sistema é provocado pela estrutura formada pela conexão entre seus componentes (FORRESTER, 1993). Assim, os modelos são construídos para representar a estrutura do sistema, enquanto a simulação demonstra seu comportamento. A Dinâmica de Sistemas assume que o comportamento de fatores internos do modelo determina seu comportamento visível. No desenvolvimento de software, por exemplo, o aumento da taxa diária de trabalho dos desenvolvedores aumenta sua produtividade, mas pode reduzir a qualidade do trabalho. Essa redução de qualidade provoca um maior número de erros que, por sua vez, aumenta o esforço total para conclusão do projeto, em função da correção dos erros. Dessa forma, se o projeto se encontra atrasado, aumentar a taxa de trabalho pode aumentar ainda mais o atraso do projeto.

Integração: pela filosofia de procurar as causas de um desvio de comportamento na estrutura do sistema, os modelos da Dinâmica de Sistemas integram seus diversos "microcomponentes". A integração desses elementos permite a inferência de seu papel no comportamento do sistema como um todo. Exemplos de "microcomponentes" do processo de manutenção de software são as atividades do processo, os desenvolvedores que realizarão as tarefas, a data esperada para conclusão da requisição de mudança, a prioridade e severidade das tarefas, entre outros.

Sistemas Fechados: a Dinâmica de Sistemas modela com clareza sistemas fechados, ou seja, sistemas caracterizados por ciclos de realimentação. Sistemas fechados assumem que uma decisão provoca uma cadeia de reações, que pode vir a influenciar as condições que exigiram a decisão original.

Causas e Consequências Distantes no Tempo: em diversos sistemas, particularmente em sistemas complexos, as consequências de uma decisão podem transparecer apenas muito tempo depois da tomada da decisão. A simulação de modelos da Dinâmica de Sistemas, por considerar fortemente os ciclos de realimentação e permitir o isolamento de variáveis do modelo, tem a capacidade de acompanhar as

consequências de cada decisão no sistema.

Mapeamento de Modelos Mentais: os modelos da Dinâmica de Sistemas não se baseiam apenas em informações numéricas. Embora a principal ferramenta de diagramação e as técnicas de simulação exijam que sejam descritos através de equações, essas equações podem ser geradas a partir do conhecimento contido nos modelos mentais dos indivíduos. Tal propriedade permite, por exemplo, que um modelo de Dinâmica de Sistemas represente comportamentos genéricos, visto que pode gerar modos de referência que ocorrem em condições para as quais não existem dados para a construção de modelos estatísticos. Por esse motivo, o processo de simulação não pode ser considerado único. As particularidades de cada sistema representado são específicas e alteram seu comportamento futuro.

Os temas apresentados até aqui são a base deste trabalho e são úteis na formulação da solução apresentada no capítulo 4. A próxima seção abordará os trabalhos relacionados que foram utilizados como embasamento teórico para o presente trabalho.

2.7 TRABALHOS RELACIONADOS

Os artigos seminais de relevância na área de reputação (MUI et al., 2002), (JØSANG et al., 2007), (JØSANG, 2007), (LIU; MUNRO, 2012) e (HENDRIKX et al., 2015) apontam que há uma falta de coerência na área de pesquisa em reputação e seus diferentes sistemas, indicado pelo fato de que muitas vezes, autores propõem novos sistemas a partir do zero, sem tentar estender ou melhorar propostas anteriores. Diante disso, e dos conceitos discutidos nas seções anteriores, uma pesquisa foi conduzida nas bases de publicação IEEE Xplore, ScienceDirect e SCOPUS e são mostrados nesta seção trabalhos encontrados na literatura técnica, relacionados à abordagem. Uma tabela é apresentada, ao final, através da qual é estabelecido um comparativo entre as tecnologias encontradas nos trabalhos relacionados.

Gallardo-Valencia et al. (2010) apresentam um protótipo para buscar código fonte confiável na web, coletam informações de reputação de uma rede social para desenvolvedores e os resultados de um motor de busca de códigos. Com isso, associam a reputação do desenvolvedor ao código fonte desenvolvido e incluem um mecanismo de ranqueamento a partir do código de maior pontuação. A abordagem apresentada

neste trabalho também oferece um mecanismo para ranquear os desenvolvedores, mas diferentemente do protótipo mencionado que utiliza o código fonte, a abordagem utiliza repositórios de solicitação de mudança. Além disso, o uso de dados históricos também é uma característica importante que a diferencia, já que a reputação pode mudar com o tempo.

Ramarao et al. (2016) fazem uma análise de como a reputação de um relator de *bugs* pode impactar no tempo de correção dos bugs reportados. A partir de dados dos *bugs* extraídos da ferramenta Jira, avaliam a similaridades dos bugs reportados para construírem seu próprio modelo. Para atribuir a reputação dos relatores, consideram que esse valor é definido pelo número de *bugs* que foram relatados por um determinado relator e que eventualmente foram corrigidos. Uma melhoria de acurácia foi observada na estimativa de tempo para a correção utilizando o modelo proposto levando em consideração a reputação do relator em detrimento ao modelo que considera apenas a similaridade dos *bugs* relatados.

Existe uma diferença quanto à aplicação da reputação. Enquanto Ramarao et al. (2016) relacionam a reputação ao tempo de correção de *bugs* reportados, a AD-Reputation (LÉLIS et al., 2018), parte integrante da solução proposta nessa pesquisa, relaciona reputação à estimativa de tempo nas atividades de manutenção. O uso da similaridade entre *bugs* reportados na construção do modelo pode ser apontado como uma semelhança, no entanto não utilizaram mecanismos de visualização para apoiar a análise das informações de reputação.

Bosu e Carver (2014) realizaram uma análise sobre o impacto da reputação dos desenvolvedores no resultado da revisão de código em projetos *Open Source*. Separados em dois grupos, os desenvolvedores do núcleo do projeto são aqueles que possuem uma relação mais longa com o projeto, já os desenvolvedores periféricos contribuem eventualmente para o projeto, interagem com os desenvolvedores do núcleo e raramente com outro periférico. Assume-se que desenvolvedores do núcleo possuem melhor reputação em relação aos periféricos. Os dados coletados na análise foram o Tempo para *feedback*, Intervalo de revisão, Aceitação do código e Número de *patches*. Utilizaram para tal, dados de diversos projetos *Open Source* dentre eles: *Chromium*

OS[8], *LibreOffice*[9] e *OpenStack*[10]. Os resultados mostraram que desenvolvedores de núcleo tiveram melhor desempenho em cada ponto analisado.

Embora projetos *Open Source* sejam desenvolvidos de maneira distribuída e colaborativa, Bosu e Carver (2014) recomendam a alocação de recursos ou criação de ferramentas de apoio à decisão na triagem dos pedidos de revisão, motivando desenvolvedores em potencial.

No contexto de desenvolvimento distribuído de software, Nascimento e Santoro (2009) realizam um estudo com objetivo de analisar um conjunto de comunidades e a forma como os participantes interagem e como é realizada a gestão de seu conhecimento. A partir dessa análise, foi proposto um modelo para gestão de conhecimento em comunidades de software livre. Nessas comunidades, desenvolvedores se relacionam e produzem código fonte. O modelo proposto é centrado nos conceitos de interação e nos artefatos produzidos. Um participante é identificado pelo seu perfil e sua reputação. Essa reputação é apresentada explicitamente para os participantes que decidem com quem interagir e contam também com o apoio do meio de comunicação que armazena as mensagens trocadas entre os desenvolvedores. Assim, o modelo relaciona um artefato a um participante pela sua avaliação e identifica a interação ocorrida para gerar o artefato.

Embora Nascimento e Santoro (2009) tratem da reputação dos desenvolvedores com o foco na colaboração, não avançam na utilização dos elementos de visualização como forma de apoio à tomada de decisões.

O uso de representações visuais para as informações de reputação vem crescendo nos últimos anos, mas é aplicado com ênfase apenas em outros contextos como comércio eletrônico. Algumas abordagens criaram apenas visualizações estáticas de gráficos de relacionamento ou pontuação de reputação (O'DONOVAN et al., 2007), (HANSSON et al., 2011). Outras abordagens passaram a oferecer aspectos de interação em suas visualizações como o trabalho que aborda a detecção de avaliações injustas (SÄNGER et al., 2015) e o estudo de Sanger e Pernul (2014) sobre detecção visual interativa de ataques em sistemas de reputação baseados em contexto.

---

[8] https://www.chromium.org/chromium-os

[9] https://pt-br.libreoffice.org

[10] https://www.openstack.org

Sanger e Pernul (2014) desenvolveram uma proposta de uso de visualização em sistemas de reputação. Através dessa proposta os autores visam permitir uma exploração visual da reputação por meio de uma apresentação gráfica de referenciais e seus atributos de contexto. Os referenciais seriam as fontes de informação e os atributos seriam os critérios e os valores atribuídos a cada um desses critérios. Dessa forma, espera-se alcançar a transparência no processo de atribuição e cálculo de reputação já que o usuário pode verificar os dados de cada fonte.

Sanger e Pernul (2014) avançaram a pesquisa e combinam técnicas de análise automatizada com visualizações interativas, propondo TRIVIA (SÄNGER; PERNUL, 2016), uma ferramenta de análise visual para avaliar a reputação do vendedor em um contexto de comércio eletrônico. Apesar de inserido em um contexto diferente da solução proposta nesta pesquisa, TRIVIA oferece algumas das principais características de um ambiente interativo de múltiplas visões relativas à seleção navegação, coordenação e organização das informações (HEER; SHNEIDERMAN, 2012).

No entanto, como exposto pelos próprios autores, sistemas de análise visual de última geração não se limitam apenas a visualizar e manipular. Tais sistemas suportam um processo iterativo de descoberta e interpretação de conhecimento. O usuário deve ser guiado através da aplicação e as descobertas podem ser registradas, anotadas e compartilhadas. Como essas técnicas se relacionam principalmente com os processos de análise geral, não foram considerados para a TRIVIA. Desta forma, TRIVIA não oferece aspectos da visualização colaborativa. Além disso, apesar de terem utilizado simulações na avaliação da proposta e disponibilizar visualizações que permitem a análise histórica das informações de reputação, no entanto, não aplicam modelo dinâmico possibilitando projetar a tendência do comportamento futuro das informações, bem como, tomar decisões baseadas em tal projeção.

Kim et al. (2015) aplicaram sua abordagem no contexto de transações *online*. Os autores defendem que uma terceira força, a recomendação de terceiros, por exemplo, pode guiar os candidatos a efetivar uma transação comercial provocando discussões e provendo opiniões sobre cada um. Nesse estudo foram utilizadas visualizações gráficas 3D para apresentar a reputação através de métricas WOM (*Word of Mouth*) geradas a partir do *feedback* dos usuários. Isso é utilizado para reorganizar o grupo e aprimorar o nível de conhecimento fornecido pelos especialistas desse grupo.

Volk et al. (2014) apresentam uma visualização chamada T-Viz capaz de exibir valores de confiança para cada critério de um modelo baseado em multicritérios juntamente com seus valores de incerteza associados. T-Viz combina gráficos de radar, que são visualizações bidimensionais para dados multivariados, com gráficos circulares. Cada fatia representa a qualidade esperada de um critério. A altura de uma fatia denota a pontuação de confiança e sua largura é o nível da segurança. Mapeia então o valor de reputação para um gradiente de cor RGB de vermelho para verde. A pontuação global é a média ponderada de todos os critérios. Permite interação com o usuário de forma a exibir os dados de maneira mais clara.

Apesar de explorar mecanismos de visualização para informações de reputação, os trabalhos de Snager e Pernul (2014) e Volk et al. (2014) não avançam permitindo que aspectos de colaboração, por exemplo, comunicação e percepção, sejam oferecidos aos usuários, o que enriqueceria o processo de tomada de decisão.

As soluções existentes apresentadas nos trabalhos relacionados são comparadas a um conjunto de características de análise (Tabela 2.1). A escolha dessas características se baseia nas condições necessárias para a estruturação da solução apresentada na seção 4, são elas:

- **Repositório de dados**, cujo objetivo é descrever a fonte de informação conforme definido por Liu e Munro (2012), para relacionar às informações de reputação. Nesse caso, podem ser classificadas em código fonte, solicitações de mudança, mensagens, ou não informado.
- **Visualização de dados**, cujo objetivo é descrever qual tipo de visualização a abordagem oferece. Podem ser classificados em única visão, múltiplas visões e não suportado.
- **Suporte à decisão**, que objetiva descrever se a abordagem adota a reputação para apoiar tomada de decisão.
- **Atividade Colaborativa**, objetiva descrever se a abordagem permite a colaboração entre membros de uma equipe geograficamente distribuída.
- **Ranqueamento,** objetiva descrever se a abordagem oferece um mecanismo para relacionar e indicar as melhores pontuações.
- **Dinâmica de Sistemas**, cujo objetivo é descrever se a abordagem aplica modelos dinâmicos na análise de comportamento da reputação ao longo do tempo.

Tabela 2.1- Comparativo entre as tecnologias dos trabalhos relacionados com as características definidas

Abordagem apresentada em	Características					
	Repositório de dados	**Visualização de dados**	**Suporte à decisão**	**Atividade Colaborativa**	**Ranqueamento**	**Dinâmica de Sistemas**
(GALLARDO-VALENCIA et al., 2010)	Código fonte	Não suportado	Sim	Não	Sim	Não
(RAMARAO et al., 2016)	Solicitação de mudança	Não suportado	Sim	Não	Não	Não
(BOSU; CARVER, 2014)	Código fonte	Não suportado	Não	Sim	Não	Não
(NASCIMENTO; SANTORO, 2009)	Código fonte e mensagens	Não suportado	Sim	Sim	Não	Não
TRIVIA (SÄNGER; PERNUL, 2016)	Mensagens	Múltiplas visões	Sim	Não	Não	Não
(KIM et al., 2015)	Mensagens	Única visão	Sim	Sim	Não	Não
(VOLK et al., 2014)	Não informado	Única visão	Não	Não	Não	Não

De modo geral, aplicar a reputação para suporte à decisão é uma característica que está presente nos trabalhos apresentados, independente do tipo de decisão como, em quem confiar (KIM et al., 2015) (SÄNGER; PERNUL, 2016), com quem interagir (NASCIMENTO; SANTORO, 2009) (KIM et al., 2015) e compartilhar conhecimento (NASCIMENTO; SANTORO, 2009), qual artefato de código utilizar (GALLARDO-VALENCIA et al., 2010). A solução proposta apresentada a partir do próximo capítulo também oferece suporte à decisão ao alocar desenvolvedores em atividades de manutenção de software em equipes distribuídas.

A atividade colaborativa gera informação e mensagens trocadas, por exemplo. Foi possível perceber a aplicação de tais informações como repositório e fonte de

informação à reputação. A solução proposta também aplica essa lógica de realimentação com os próprios dados de interação.

Outra semelhança entre os estudos relaciona-se às tecnologias aplicadas como MySql, além de bibliotecas web acessíveis, como D3js que facilitam a portabilidade. Além disso, permitem a integração e o desenvolvimento a partir delas. Essas características auxiliaram a definição da proposta.

Entretanto, pode-se notar que, de uma forma geral, todos as características em conjunto não são abordadas pelos trabalhos apresentados. Esse fato reforça a importância do objetivo deste trabalho relativo a uma abordagem que, através de informações de reputação, trate a alocação de desenvolvedores no planejamento de atividades de manutenção de software. Essa abordagem está aliada a um AIMV e a um mecanismo para relacionar e indicar as melhores pontuações. Além disso, apoia a colaboração e a tomada de decisão.

A utilização da Dinâmica de Sistemas através de um modelo dinâmico constitui uma contribuição para a área de sistemas de reputação diante do fato de não ter sido observada sua aplicação em nenhuma das abordagens estudadas e relacionadas a este trabalho. Uma das vantagens alcançadas com a aplicação do modelo dinâmico está associada à possibilidade de representar a reputação por meio de multicritérios e estudar a influência entre eles. Outro aspecto positivo é a capacidade de simular e projetar a tendência futura da reputação, além de permitir o acompanhamento da evolução da reputação ao longo do tempo.

2.8 CONSIDERAÇÕES FINAIS SOBRE O CAPÍTULO

O objetivo deste capítulo foi apresentar os conceitos base deste trabalho. Foram abordados os principais aspectos relativos à reputação, sistemas de reputação, Visualização, Colaboração, Manutenção de Software e a Dinâmica de Sistemas, seguido de uma visão geral das relações existentes entre os respectivos conceitos. Foram apresentados também estudos relacionados encontrados na literatura técnica especializada, bem como um comparativo entre características de análise. Pôde-se observar que apesar de relevantes, não avançam ao reunir as características propostas pelo presente trabalho. No próximo capítulo será mostrado o processo que resultou no Modelo de Cálculo de Reputação para desenvolvedores de software.

3. MODELO DE CÁLCULO DE REPUTAÇÃO NO DESENVOLVIMENTO DISTRIBUÍDO DE SOFTWARE

O cálculo da reputação é um passo importante na execução de sistemas de reputação. Portanto, é necessário analisar e estabelecer os parâmetros que serão considerados na agregação de informações, pois influenciam o cálculo do valor da reputação.

Estudos anteriores realizaram pesquisas para analisar os sistemas de reputação existentes e propuseram taxonomias para classificar esses sistemas em diferentes dimensões (JØSANG et al., 2007) (LIU; MUNRO, 2012) (HENDRIKX et al., 2015). Como resultado, observou-se que embora exista uma lista de sistemas de reputação que suportam sistemas *on-line*, incluindo redes *peer-to-peer*, a reputação pode variar conforme o contexto ao qual é aplicada. Além disso, nenhum desses sistemas se concentrou em atividades de manutenção colaborativa de software.

Como foi mencionado, a reputação é vista como uma "ponte" para estabelecer a confiança. Assim, a influência que a reputação tem na percepção de confiança entre os membros das equipes distribuídas é conhecida. No entanto, os fatores que influenciam a reputação nessas equipes permaneciam desconhecidos.

Esse cenário revelou a oportunidade de realizar um mapeamento sistemático para coletar evidências da literatura técnica, especialmente, relacionadas à identificação e relação entre métricas, medidas, critérios e fatores para determinar o valor da reputação.

3.1 MAPEAMENTO SISTEMÁTICO

Estudos baseados em mapeamento sistemático são projetados para fornecer uma visão geral de um campo de pesquisa. O mapeamento sistemático conduzido por Lélis et al. (2016a) teve como questão de pesquisa: ''Quais são as métricas, medidas, critérios e fatores que podem ser usados na agregação e no cálculo da reputação dos envolvidos na manutenção e evolução distribuídas do software?''. O protocolo e processo geral adotado seguiram as orientações de Kitcheham e Charters (2007).

3.1.1 PLANEJAMENTO E CONDUÇÃO

A atividade de planejamento do mapeamento sistemático inclui a identificação dos objetivos e a definição de um protocolo de pesquisa. Devido à escassez de propostas no contexto da manutenção de software, a análise foi realizada assumindo o uso de dados de reputação em vários contextos. Além disso, estudos que consideraram métricas, medidas, fatores e critérios (aqui chamados de parâmetros) utilizados para determinar o grau de reputação podem fornecer indicações de uso no contexto de manutenção e evolução.

O objetivo para este mapeamento foi definido de acordo com a abordagem Goal/Question/Metric (GQM) (BASILI; WEISS, 1984). O objetivo, segundo essa abordagem, foi formulado como: "***Analisar*** as métricas, medidas, critérios e fatores utilizados na agregação e cálculo do grau de reputação ***com a finalidade de*** caracterizá-los ***com relação*** à efetividade ***do ponto de vista*** dos gerentes ou líderes de equipe em relação aos seus colaboradores ***no contexto de*** manutenção e evolução de software.

Uma questão de pesquisa conduz à definição de uma consulta a ser executada em bases de publicações. Para compor esta consulta, palavras-chave são definidas. Os termos de pesquisa são construídos em três etapas: a estruturação de questões de pesquisa, a fim de identificar palavras-chave, a identificação de sinônimos para cada uma das palavras-chave e a construção da sequência de pesquisa com base na combinação dos termos e seus sinônimos, usando os operadores OR e AND. A definição das palavras-chave de busca, utilizadas na pesquisa, foram feitas com base na estratégia PICOC (PETTICREW; ROBERTS, 2008). Neste trabalho, a **Comparação** não é relevante, uma vez que este mapeamento visa conceber uma visão geral do assunto através de um estudo exploratório. O resultado do processo é apresentado a seguir:

- **População:** entidade, usuário, desenvolvedor, líder, gerente, *stakeholder*
- **Intervenção:** reputação, confiança, cálculo, agregação
- **Comparação:** não definido
- **Resultado:** critério, métrica, medida, fator
- **Contexto:** manutenção de software, evolução de software

A seguinte *string* de busca foi definida a partir das palavras chave e aplicada nas bases Scopus, ScienceDirect, IEEExplore, EI Compendex e Web of Science:

“("entity" OR "user" OR "developer" OR "leader" OR "manager" OR "stakeholder") AND ("criteria" OR "criterion" OR "metric" OR "measure" OR "factor") AND ("reputation" OR "trust") AND ("calculation" OR "aggregation")”

Após a conclusão do protocolo, seguiu-se à condução do mapeamento. Inicialmente foram retornados 110 artigos e, após o processo de seleção, permaneceram 27 artigos que foram aceitos. Os parâmetros identificados foram catalogados e resumidos. Foi possível identificar parâmetros que não permitiriam uma correlação com o contexto de manutenção colaborativa de software, uma vez que são específicos em seus domínios.

Os parâmetros mais comumente mencionados nos artigos aceitos foram Opinião, Tempo, Relacionamento, Comportamento, Subjetividade e Confiabilidade, conforme apresentado por Lélis et al. (2016a). Com a evolução do trabalho e para melhor se adequar ao conceito utilizado na prática, a ideia de parâmetro foi substituída pelo termo “critério”. Outra mudança foi em relação ao parâmetro Subjetividade. Essa decisão foi tomada pois, conceitualmente, o parâmetro subjetividade era definido como o inverso da Similaridade e embora as abordagens analisadas chamassem de Subjetividade, na prática apresentavam métricas associadas ao cálculo da Similaridade. Assim, o parâmetro Subjetividade foi estabelecido como critério Similaridade. Outra mudança foi em relação ao parâmetro Tempo. O tempo já foi tratado como Expertise. Dessa forma, a expertise considera o tempo que um desenvolvedor está na equipe, sendo capaz de representar o conhecimento que este desenvolvedor possui das regras de negócio. Assim, o parâmetro Tempo foi definido como critério Expertise. Desta forma, a reputação seria estabelecida por meio de multicritérios.

3.1.2 RESULTADOS - CRITÉRIOS PARA CÁLCULO DE REPUTAÇÃO

Como resultado do mapeamento sistemático, foi possível perceber que o cálculo da reputação é complexo podendo envolver aspectos distintos e evidenciam o caráter multicritério do cálculo da reputação. Tais aspectos serão chamados daqui em diante de critérios e, brevemente, discutidos.

A maioria dos modelos atuais de evolução da confiança adota a expertise ou a média aritmética para calcular os valores de confiança das entidades (YIN et al., 2013).

Quanto maior é a expertise, mais importante é a contribuição da confiabilidade em relação à reputação. Para altos valores de expertise, uma fonte de informação “A” pode estar muito confiante em sua capacidade de julgar a capacidade do alvo “B”, e consequentemente calcular sua confiabilidade (ROSACI et al., 2012).

Os usuários interagem uns com os outros, compartilhando seus conhecimentos e são classificados de acordo com suas pontuações de reputação. Dependendo do valor de Expertise, há grande chance de alcançarem altas posições e ganhar muitos pontos de reputação (JAYASHREE; CHRISTY, 2015). A ideia de consumir e compartilhar conhecimento corresponderia à aquisição de expertise em um determinado domínio. No contexto desta pesquisa, foi definido que a aquisição de conhecimento se dá quando desenvolvedores tratam a mesma requisição de mudança e as áreas de conhecimento, como os projetos e módulos que passam por manutenção.

De modo geral o critério Opinião é considerado base para calcular a reputação, sendo tratado como a opinião das demais entidades a respeito de uma específica, conforme apresentado em (CAVERLEE et al., 2010) e (ROSACI et al., 2012). Quando isso ocorre, ou seja, entidades trocam avaliações sobre a Opinião uma da outra, tal fato constitui um dos escopos ao qual o Critério Opinião está associado, as entidades. No entanto, essas entidades formam grupos com o passar do tempo, mesmo que involuntariamente por possuírem características (ZUPANCIC; JURIC, 2015), ou opiniões (XU et al., 2015) semelhantes. Assim, a Opinião passa a representar uma visão da equipe, ou grupo, através da qual uma equipe oferece sua opinião a respeito de outra, ou os próprios membros do grupo avaliam o conjunto (JAYASHREE; CHRISTY, 2015). Essas equipes podem possuir origem semelhante ou estarem sob domínio distintos. Deste modo, o domínio interfere na reputação da equipe (ALNEMR; MEINEL, 2008) (KHIABANI et al., 2010) , o que caracteriza mais um escopo ao qual o critério Opinião pode estar associado. A reputação em relação a outro domínio é utilizada como parâmetro caso entidades de domínios diferentes queiram se relacionar.

O critério Relacionamento mede a qualidade do relacionamento entre os membros da equipe e a força dessa relação. Por exemplo, Caverlee et al. (2010) distinguem a confiança do usuário, da confiança sobre o relacionamento do usuário como sendo dois parâmetros distintos e que ambos influenciam na reputação. O objetivo é determinar a qualidade desse relacionamento sob o ponto de vista das entidades envolvidas. A relação social entre grupos é a medida considerada por Memarmoshrefi et

al. (2012) para determinar a confiabilidade de um grupo e define a noção de confiança entre grupos com base em relacionamentos anteriores.

Um relacionamento é importante por gerar uma opinião posterior. Isso pode ocorrer entre um grupo e uma entidade, como em (JAYASHREE; CHRISTY, 2015) que usa como exemplo a relação de confiança entre faculdades e estudantes no ambiente *e-learning*. O compartilhamento de informação sigilosa pode também ser visto como uma medida para a força da relação (WROBEL et al., 2013). O compartilhamento de informação privada para uma pessoa é uma expressão de confiança, assim quanto mais esse tipo de informação duas pessoas trocam, maior é a confiança entre elas.

O critério Comportamento é abordado por Caverlee et al. (2010) e constitui um dos fatores que influenciam a reputação na medida em que afeta a opinião dos demais usuários. Está centrado em uma entidade alvo e considera seus valores de reputação anteriores, assim o comportamento passa uma noção de evolução, baseado em um histórico, tal como considerado por Khiabani et al. (2010) e Yin et al. (2013). Embora ambos critérios Opinião e Comportamento estejam relacionados e tenham sentido semelhante, Opinião considera o ponto de vista das outras entidades sobre a entidade alvo e o Comportamento visa observar a evolução histórica do alvo baseada em seus valores anteriores de reputação.

O critério Confiabilidade representa o grau que a informação passada por um desenvolvedor é confiável. Em (KHIABANI et al., 2010), o histórico formado pela evolução dos valores de reputação é utilizado como subcritério para calcular confiabilidade da informação. Seguindo lógica semelhante, em (JIANG et al., 2012), o critério Confiabilidade é aplicado quando uma entidade "A" nunca teve relação direta com outra entidade "B", servindo assim às recomendações de terceiros sobre entidades conhecidas.

O critério Similaridade oferece prioridade às avaliações similares. Casos que possuam características similares recebem maior peso. Quando os membros oferecem sua opinião, o fato de cada um possuir uma personalidade distinta faz surgir um grau de subjetividade, apontado em (ZUPANCIC; JURIC, 2015), que interfere negativamente no valor da reputação. Essa subjetividade gera distorção nas avaliações e deve ser minimizado. Uma forma de minimizar seu efeito é considerar somente aquelas classificações compartilhadas por usuários com características semelhantes. Essas características são chamadas por Zupancic e Juric (2015) de atitudes de confiança, e são

implicitamente derivadas de suas classificações passadas. Algo semelhante é feito por Qu et al. (2006), ao considerar uma medida de similaridade com certas classificações de avaliadores, sendo descrita como "grau de coerência de critérios" e usada como peso para a agregação da reputação. Em (XU et al., 2015), também é chamado de fator de similaridade.

Para ilustrar esse mecanismo, suponha que se deseja analisar a reputação de um usuário (U1) usando 5 classificações anteriores, cada qual de um usuário diferente (U2, U3, U4, U5 e U6). Em uma escala Ruim, Médio e Bom para a classificação, suponha ainda, que as classificações foram respectivamente: Bom, Médio, Bom, Ruim e Ruim. À primeira vista, sem analisar as características de cada usuário, é distribuído um peso igual às classificações e chega-se a pensar que a reputação de U1 é “Médio”, no entanto por U2 e U4 terem características semelhantes ao U1 (conhecimento de linguagem de programação, tempo de prática ou opinião, por exemplo), suas classificações têm peso maior. Do mesmo modo, U5 e U6 possuem características que distanciam de U1 e assim suas classificações terão peso menor. Como resultado, a reputação de U1 tende a ser diferente, podendo chegar a Bom.

3.2 RELACIONANDO OS CRITÉRIOS AOS ALGORITMOS, FÓRMULAS E MÉTRICAS ASSOCIADAS

Uma vez apresentados e definidos os critérios que afetam a reputação de desenvolvedores e, por isso, utilizados no modelo em construção, é preciso determinar as formas disponíveis de calcular cada um dos critérios. Nesse sentido, foi definido um protocolo seguindo a abordagem Goal/Question/Metric (BASILI; WEISS, 1984) que define um objetivo a ser alcançado, estabelece questões de pesquisa e métricas que auxiliam a responder as questões definidas. O protocolo seguiu a seguinte estrutura:

- Objetivo (*Goal*): **Analisar** os artigos selecionados no mapeamento sistemático conduzido, **com o propósito de** caracterizar **com respeito ao** cálculo de reputação, **sob o ponto de vista** do critério *<critério_específico>*, **no contexto de** manutenção e evolução de software.
- Questões de Pesquisa (*Question*):
    - **Q1:** Quais modelos, algoritmos, fórmulas, mecanismos foram apresentados para o cálculo do critério *<critério_específico>*, e como foram avaliados?

- **Q2:** Como calcular o critério <*critério_específico*>?

A primeira questão visa descrever o processo de cálculo do critério <*critério_específico*>, enquanto a segunda questão objetiva coletar as variáveis e valores envolvidos no processo.

- Métricas (*Metrics*):
  - **M1:** Quantidade de modelos, algoritmos, fórmulas e mecanismos apresentados.
  - **M2:** Valor de pontuação da avaliação, segundo a escala:
    - 0: sem avaliação
    - 1: exemplos ou cenários
    - 2: estudo de caso ou prova de conceito
  - **M3:** Quantidade de variáveis e métricas utilizadas para o cálculo.

Foram definidas com base nas questões de pesquisa Q1 e Q2, sendo as métricas M1 e M2 referentes à questão Q1, e a métrica M3 referente à questão Q2. O resultado de cada métrica definida no protocolo foi computado, associado à referência que serviu de base para responder às questões Q1 e Q2. Foram omitidos os resultados que obtiveram pontuação inferior a 2, na métrica M2 relativa à forma de avaliação do processo de cálculo apresentado. Essa medida foi tomada pois, a forma de avaliar destaca quais abordagens foram implementadas e avaliadas, permitindo a um gerente compreender, na prática, a forma de coletar e medir o critério.

A expressão <*critério_específico*> foi substituída considerando cada um dos critérios definidos na subseção anterior. Diante do objetivo de identificar os algoritmos ou fórmulas que pudessem auxiliar no cálculo dos critérios e tais algoritmos e fórmulas agregarem variáveis e métricas, foi possível identificar e associar métricas a cada um desses algoritmos e fórmulas. A Figura 3.1 ilustra um exemplo desse processo para o critério Relacionamento.

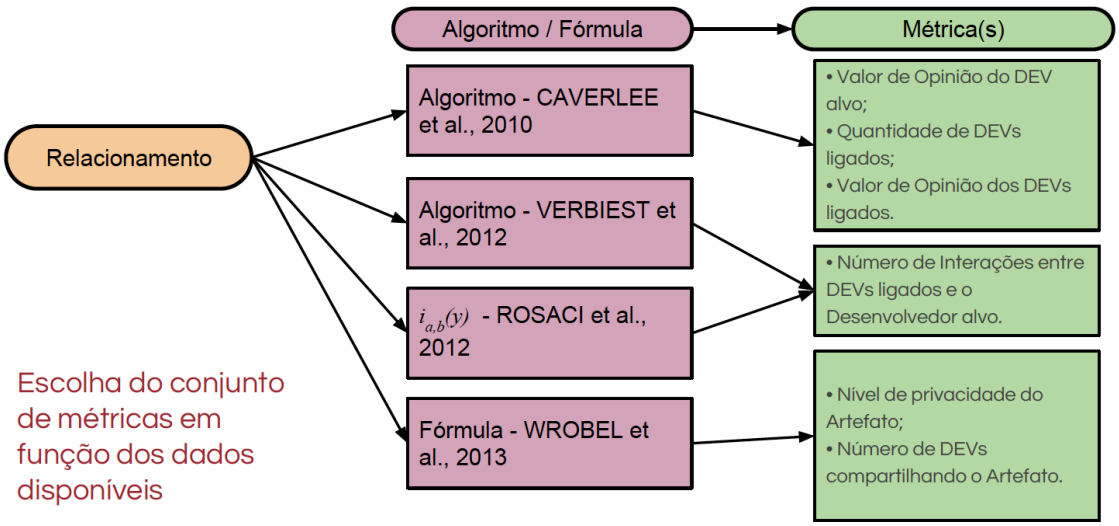

Figura 3.1: Fórmulas e algoritmos associados às métricas - exemplo critério Relacionamento

A partir do protocolo QGM foram identificados os algoritmos e fórmulas que possibilitaram o cálculo desse critério. Nesse ponto, a escolha sobre qual algoritmo ou fórmula utilizar é feita através da natureza dos dados disponibilizados para análise e o cálculo. Nesse sentido cada fórmula e cada algoritmo estão associados a um conjunto de métricas. Por outro lado, essas métricas são agregadas através de uma das fórmulas ou algoritmo identificado e, como consequência, obtêm-se o valor do critério, no caso Relacionamento.

Embora a Figura 3.1 apresente um exemplo para o critério Relacionamento esse processo é aplicado para cada critério, de forma que cada critério está associado a possibilidades variadas de fórmulas e algoritmos, ligados a diferentes métricas para seu cálculo.

Os resultados obtidos pela aplicação do protocolo, a descrição das fórmulas e algoritmos encontrados, bem como as métricas associadas, são apresentados nas subseções seguintes.

3.2.1 RESULTADO CRITÉRIO OPINIÃO

Com relação ao critério Opinião, o protocolo foi conduzido e a Tabela 3.1 mostra o resultado obtido para cada métrica definida no protocolo.

Tabela 3.1 - Resultado GQM Critério Opinião

REFERÊNCIA	M1	M2	M3
(CAVERLEE et al., 2010)	3	2	2

(ZUPANCIC; JURIC, 2015)	1	2	1
(ROSACI et al., 2012)	2	2	2

Para responder à questão Q1, o estudo apresentado em por Caverlee et al. (2010) mostrou 2 formas diferentes de calcular o critério Opinião mas seguindo a mesma fórmula como base. A Equação (3.1 apresenta essa fórmula na qual a Opinião é representada como a fração das avaliações positivas.

$$F(j) = \frac{\sum_i \mathcal{I}(v_i(j)^+)}{\sum_i \mathcal{I}(v_i(j)^+) + \mathcal{I}(v_i(j)^-)} \tag{3.1}$$

Assim, a Opinião é calculada com base em duas variáveis: (i) avaliações positivas onde cada avaliação é indicada com valor 1 e (ii) avaliações negativas, caso em que cada avaliação é indicada com valor 0.

O mecanismo de cálculo mostrado em (ZUPANCIC; JURIC, 2015) segue uma representação de matriz através da qual as linhas e colunas representam as entidades que trocam avaliações umas das outras, e a posição *Pi,j* representa a avaliação de *i* sobre *j*. A Equação (3.2 mostra a definição para a base de cálculo.

$$avg\left(\underline{\mathcal{M}_{nk,k}}\right) \in [-2, 2] \tag{3.2}$$

Considera-se a média das avaliações das outras entidades a respeito de k. Tais avaliações podem receber valores inteiros no intervalo de -2 a 2.

Outra abordagem mostrada em (ROSACI et al., 2012), também calcula o critério Opinião com base na taxa média de avaliações, tal como mostrado em (ZUPANCIC; JURIC, 2015). No entanto, trata-se da média ponderada das opiniões, e a ponderação é feita pelo ponto de vista da entidade que busca informações sobre a entidade alvo. O método de cálculo é mostrado na Equação (3.3.

$$\pi_{ab}^{\gamma} = \frac{\sum_{c \in C - \{a,b\}} \tau_{cb}^{\gamma} \cdot \tau_{ac}^{\gamma}}{\sum_{c \in C - \{a,b\}} \tau_{ac}^{\gamma}} \tag{3.3}$$

No mecanismo, assume-se que a Opinião que uma entidade A atribui a outra entidade alvo B, está relacionada a um domínio γ. Pode ser obtida por uma média ponderada de todas as medidas de confiança de cada entidade C associada a B, diferentes de A e B. Em outras palavras, a sugestão que cada entidade C oferece à entidade A, relativa à B, é representada pela confiança que C deposita em B. Essas

sugestões são ponderadas pela medida de confiança τ_{ab}^{γ}, para levar em consideração a confiança que A tem em C. A Figura 3.2 ilustra esse cenário.

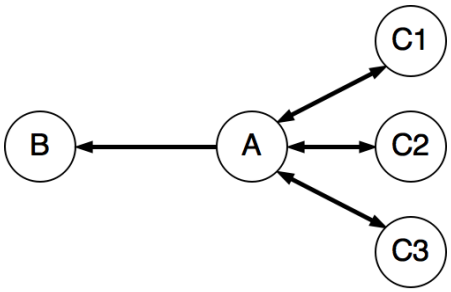

Figura 3.2 - Mecanismo de Cálculo do Critério Opinião (ROSACI et al., 2012)

Assim, foram apresentadas três abordagens possíveis para o cálculo do critério Opinião. No entanto, a abordagem a ser aplicada no modelo em construção depende dos dados disponibilizados para o processamento no modelo. O processamento dos dados é melhor explicado no próximo capítulo.

3.2.2 RESULTADO CRITÉRIO SIMILARIDADE

O protocolo definido foi aplicado para o critério Similaridade e a Tabela 3.2 mostra os valores obtidos para cada métrica.

Tabela 3.2 - Resultado GQM Critério Similaridade

REFERÊNCIA	M1	M2	M3
(ZUPANCIC; JURIC, 2015)	1	2	1
(XU et al., 2015)	1	2	5
(QU et al., 2006)	3	2	2

O modelo apresentado em (ZUPANCIC; JURIC, 2015) é baseado na eliminação da subjetividade pela aproximação e comparação do que os autores chamaram de atitudes de confiança (avaliações representada por um número inteiro na escala de -2 a 2). Por exemplo, se uma entidade *i* atribui a outras entidades alvo, frequentemente, valores 1 ou 2, considerados altos nesse contexto, então essa entidade *i* é similar a outra entidade *j*, que também atribui com frequência valores altos para entidades alvo. O mesmo também ocorre para entidades que atribuem a outros valores nos extremos (-2 e 2, por exemplo). O algoritmo apresentado possui uma fórmula, apresentado na Equação (3.4, para o cálculo da Similaridade.

$$sim(a_i, a_j) = 1 - \sup_{\omega} |F_{i,ni}(\omega) - F_{j,nj}(\omega)| \qquad (3.4)$$

Na Equação (3.4, $F_{i,ni}(w)$ e $F_{j,nj}(w)$ são as chamadas atitudes de confiança das entidades *i* e *j* e a similaridade entre as entidades dado por $sim(a_i,a_j)$ é definido em um intervalo entre 0 e 1. Basicamente, considera-se a similaridade entre duas entidades *i* e *j*. A comparação é feita com base na distribuição dos valores atribuídos a essas duas entidades. Duas entidades são similares se possuem quantidades semelhantes de cada uma das classificações de -2, 2, 0, 1 e 2. O teste Kolmogorov-Smirnov (MASSEY; FRANK, 1951) de comparação de duas amostras calcula a maior diferença entre as funções de distribuição cumulativa empírica como a estatística D. Se os valores provêm das mesmas distribuições, o resultado do teste KS de duas amostras é igual a 0. É utilizada a estatística de Kolmogorov-Smirnov D para medir a semelhança entre as atitudes de confiança das entidades.

No mecanismo apresentado em (XU et al., 2015), cada entidade é descrita por 5 características: localização, grau de conhecimento, trabalho, idade e gênero. Inicialmente, o algoritmo cria uma árvore de decisão com os valores possíveis de cada característica. Em seguida, calcula para cada caminho possível na árvore o valor de similaridade correspondente. A cada nó da árvore que ocorre uma correspondência é acrescido de 1 ponto ao valor do caminho. Ao final, têm-se um valor de similaridade associado a cada caminho.

O mecanismo que os autores seguem em (QU et al., 2006), se baseia na aproximação de *Hamming*, na aproximação Euclidiana, e no coeficiente de correlação de Pearson. As aproximações seguem a teoria de medir distância entre dois pontos, no entanto mais detalhes podem ser encontrados em (WOHLIN et al., 2012). A Equação (3.5 mostra a fórmula usada pelos autores na avaliação para calcular a similaridade entre duas entidades.

$$CCD_{<j,i>} = 1 - \frac{1}{n}\sum_{t=1}^{n}\left|PR_{<j,t>} - PR_{<i,t>}\right| \qquad (3.5)$$

Nesse caso, a Similaridade é calculada de forma semelhante ao mostrado na Equação 3.4. A distância entre as entidades *i* e *j* é calculada pela diferença entre os valores atribuídos a eles pelas demais entidades. O resultado é um valor entre 0 e 1. Assim, quanto mais próximo de 1 for o valor final, maior é a similaridade entre as duas entidades.

Nesta subseção, foram apresentadas três abordagens possíveis para o cálculo do critério Similaridade. No entanto, a abordagem a ser aplicada no modelo em construção

depende dos dados disponibilizados para o processamento no modelo, ficando a escolha associada aos dados. O processamento dos dados é melhor explicado no próximo capítulo.

3.2.3 RESULTADO CRITÉRIO RELACIONAMENTO

O critério Relacionamento foi submetido ao protocolo e o resultado sumarizado é apresentado na Tabela 3.3.

Tabela 3.3 - Resultado GQM Critério Relacionamento

REFERÊNCIA	M1	M2	M3
(CAVERLEE et al., 2010)	1	2	2
(WROBEL et al., 2013)	1	2	2
(ROSACI et al., 2012)	1	2	1
(VERBIEST et al., 2012)	3	2	3

O mecanismo considerado em (VERBIEST et al., 2012) considera as entidades tal qual dispostas em um grafo e obtêm o critério Relacionamento através da distância entre duas entidades *a* e *b*, ou seja, a quantidade de passos que devem ser dados para *a* chegar em *b*. Outra particularidade da abordagem é que quanto menor a quantidade de passos, mais intenso é o relacionamento.

O mecanismo considerado em (ROSACI et al., 2012) obtêm o critério Relacionamento através do número de interações entre duas entidades *a* e *b*, e é representado matematicamente por $i_{a,b}(y)$. Este método especifica em qual domínio, *y*, ocorreu a interação.

De forma semelhante, o compartilhamento de informação é considerado como meio de interação entre entidades, e foi aplicado em (WROBEL et al., 2013) para medir o Relacionamento. A Equação (3.6 apresenta a fórmula aplicada como base do cálculo.

$$DT_i = \frac{p_i}{1 + (n_i - 1)(1 - n_i))} \tag{3.6}$$

Nesse caso, as informações estão associadas a um nível de sigilo *p*, e o número de pessoas que compartilham determinada informação é dado por *n*. O resultado da função oferece uma noção da força do relacionamento entre entidades. Assim,

compartilhar informações sensíveis com um número reduzido de pessoas pode indicar alto grau de confiança entre essas pessoas.

O mecanismo mostrado em (CAVERLEE et al., 2010) segue a fórmula da Equação (3.7. Considerando que se deseja saber o valor do critério Relacionamento da entidade *i*, então o valor do critério Opinião da própria entidade *i* é agregado à média dos valores do critério Opinião de todas as entidades *j* que se relacionaram com a entidade *i*.

$$R_{[2]}(i) = F(i) \sum_{j \in rel(i)} F(j)/|rel(i)| \left[\sum_{l \in rel(j)} F(l)/|rel(j)| \right] \qquad (3.7)$$

O relacionamento do usuário *i* foi construído em termos de um modelo de caminhada aleatória, em que um caminhante aleatório origina sua caminhada a partir da entidade *i* e segue aleatoriamente as bordas dos relacionamentos da entidade *i* e as entidades subsequentes que são alcançadas em um pequeno número de passos. O caminhante aleatório escolherá finalizar sua caminhada depois de caminhar um número k de etapas de profundidade, longe da entidade de origem. Esse mecanismo considera o valor do critério Opinião de cada entidade para ajudar a guiar sua caminhada. Nos extremos, quando todas as entidades dentro do círculo de relacionamentos com profundidade k da entidade de origem tem um valor de Opinião perfeito (ou seja, 1 para todas as entidades dentro do limite k de profundidade), a entidade *i* tem relacionamento igual a 1. Por outro lado, se a entidade *i* possui valor do critério Opinião ruim (ou seja, 0), ou todas as entidades dentro de k passos da origem tiverem avaliações de Opinião pobres, então a qualidade do relacionamento da entidade *i* é 0. Para resumir, a qualidade do relacionamento da entidade *i* pode ser interpretada como a probabilidade que um caminhante aleatório possui de iniciar sua caminhada na entidade *i*, e terminar em uma entidade de alta qualidade, depois de caminhar até k passos longe de *i*.

3.2.4 RESULTADO CRITÉRIO COMPORTAMENTO

Com relação ao critério Comportamento, o protocolo foi aplicado e a Tabela 3.4 mostra o resultado sumarizado de cada métrica.

Tabela 3.4 - Resultado GQM Critério Comportamento

REFERÊNCIA	M1	M2	M3
(CAVERLEE et al., 2010)	1	2	2

(ZUPANCIC; JURIC, 2015)	1	2	2

Uma semelhança entre as propostas resultantes é a estreita relação com o critério Opinião. O critério Comportamento pode ser calculado seguindo o mecanismo apresentado em (ZUPANCIC; JURIC, 2015), considerado um conjunto de valores possíveis que podem ser atribuídos ao critério Opinião, no caso a escala de -2 a 2. Assim, o cálculo do critério comportamento é realizado através da quantidade de vezes que Opinião recebeu valor igual a uma variável *w* que indica um dos valores possíveis. A Equação (3.8 apresenta a função que representa esse caso.

$$F_{i,ni}(\omega) = \frac{1}{n}\sum_{j=1}^{ni} I(\omega_{i,j} \leq \omega) \qquad (3.8)$$

Dessa forma, é registrado um valor do critério Comportamento para cada valor possível do critério Opinião e assim, cada valor do critério Comportamento funciona como probabilidade da entidade se comportar tal qual estabelecido pela escala.

Outra maneira de calcular o critério Comportamento é apresentado por Caverlee et al. (2010) e considera a média entre valores anteriores do critério Opinião. A fórmula que representa esse caso é mostrada na Equação (3.9

$$Tr_h(i) = \sum_{m=1}^{M} ST(i,m) \cdot \frac{I_m}{\sum_{l=1}^{M} I_l} \qquad (3.9)$$

O coeficiente multiplicativo, I_m , da fórmula representa um grau de importância do valor ao qual está associado. Trata-se de uma ponderação feita no valor de Opinião que reduz o peso de acordo com a idade da avaliação. Dessa forma, notas atribuídas há mais tempo refletem um comportamento que já pode ter mudado e da mesma forma eleva o peso de notas atribuídas recentemente por refletirem um comportamento atual.

3.2.5 RESULTADO CRITÉRIO CONFIABILIDADE

O protocolo definido foi aplicado ao critério Confiabilidade e a Tabela 3.5 sumariza os valores obtidos nas métricas, por cada trabalho analisado.

Tabela 3.5 - Resultado GQM Critério Confiabilidade

REFERÊNCIA	M1	M2	M3
(CAVERLEE et al., 2010)	1	2	4
(ROSACI et al., 2012)	1	2	3

O cálculo do critério Confiabilidade também demonstrou um grau de dependência com valores de outros critérios. O mecanismo apresentado em (ROSACI et al., 2012) leva em consideração o valor de um critério que será abordado na subseção seguinte, o Expertise. Além disso, a Confiabilidade ainda depende do número de interações feitas no passado entre as entidades *a* e *b* ocorridas em um domínio específico *y*. Depende ainda, de um valor definido no intervalo [0,1] que *a* oferece para *b* após a interação no domínio *y* onde 0 indica desconfiança e 1, confiança total.

Outra forma de cálculo do critério Confiabilidade é mostrado por Caverlee et al. (2010) e depende dos valores dos critérios Relacionamento e Opinião. A fórmula para o cálculo é apresentada pela Equação (3.10.

$$Tr_q(i) = \lambda \sum_{j \in rel(i)} R(j) \cdot Tr_q(j)/|rel(j)| + (1 - \lambda)F(i) \qquad (3.10)$$

Considerando que se deseja calcular a Confiabilidade da entidade *i*, então é preciso saber o valor do critério Relacionamento, dado por *R(j)*, e também do próprio critério Confiabilidade, dado por $Tr_q(j)$ de cada entidade *j* ligada à entidade *i*. Esses valores são ponderados pela quantidade de entidades, |*rel(j)*|, ligadas a *i*. Também é dependente do valor do critério Opinião da própria entidade *i*. Pela dependência com o valor da própria Confiabilidade, o resultado pode ser obtido, segundo os autores, através de um algoritmo recursivo.

Assim conforme comentado anteriormente, embora tenham sido apresentadas duas abordagens possíveis para o cálculo do critério Confiabilidade, a escolha da abordagem a ser aplicada no modelo em construção, depende dos dados disponibilizados para o processamento no modelo.

3.2.6 RESULTADO CRITÉRIO EXPERTISE

A apresentação dos resultados do critério Expertise seguiu o formato utilizado para os demais critérios. A Tabela 3.6 sumariza os valores obtidos nas métricas, por cada trabalho analisado.

Tabela 3.6 - Resultado GQM Critério Expertise

REFERÊNCIA	M1	M2	M3
(ROSACI et al., 2012)	1	2	1
(ZHANG et al., 2007)	1	2	3

O critério Expertise se tornou, com o passar do tempo, fundamental para as empresas quando se trata de compartilhar conhecimento e os ganhos agregados ao conhecimento adquirido com os demais membros de uma equipe. O domínio ao qual o conhecimento adquirido está associado é outro ponto que merece atenção. O mecanismo apresentado em (ROSACI et al., 2012) leva em consideração isso, e define a expertise como um valor no intervalo [0,1] que representa o nível de conhecimento pessoal que uma entidade *a* assume ter em um domínio *y*. Essa Expertise depende do conhecimento adquirido por *a* no domínio *y*.

Outra maneira de calcular o critério Expertise leva em consideração as demais entidades que compartilham conhecimento quando trabalham em conjunto. O mecanismo mostrado em (ZHANG et al., 2007) se chama Expertise Rank, e utiliza um algoritmo iterativo para calcular a expertise baseado na expertise dos demais. Assume-se que a entidade *a* já se relacionou com as entidades U_1 até U_n, seja compartilhando conhecimento, trabalhando junto até respondendo perguntas em uma área de domínio específica. Então a Expertise da entidade *a* é dada conforme a fórmula base do cálculo mostrada na Equação (3.11.

$$ER(A) = (1-d) + d\,(ER(U_1)/C(U_1) + \ldots + ER(U_n)/C(U_n)) \qquad (3.11)$$

Os valores de $C(U_i)$ são definidos como o número total de entidades que se relacionaram com U_i conforme explicado anteriormente, e o parâmetro *D* é um fator que pode ser atribuído entre 0 e 1. Segundo os autores, *D* foi utilizado com valor de 0.85. No entanto, foram testados outros valores para este fator como 0.95 até 0.7 não obtiveram diferença significativa.

Vale destacar que a variável *D* se comporta como solução à partida fria de Expertise. Uma entidade que não possui relações, ou ainda não teve oportunidades de colaboração com nenhum outro, possui expertise *(1-D)*. Nesse sentido, as duas abordagens mostradas podem atuar em conjunto. Inicialmente, um valor é atribuído com base no depoimento do própria entidade. A partir disso, o valor é acompanhado e atualizado utilizando o Expertise Rank.

3.2.7 MÉTRICAS ASSOCIADAS

A partir do resultado obtido através dos algoritmos e fórmulas, com possibilidade de serem utilizados no cálculo de cada um dos critérios, definiu-se um conjunto de métricas que seriam aplicáveis ao cálculo do valor de cada critério.

A partir desse conjunto de métricas associadas a cada critério, representado na Tabela 3.7, pôde-se fazer um estudo no sentido de verificar as influências existentes no comportamento entre os critérios. Vale destacar que Desenvolvedor Alvo referencia um dos conceitos de Sistemas de Reputação já apresentado e se relaciona à entidade que se deseja analisar as informações de reputação. Os chamados Desenvolvedores (DEVs) ligados compreendem o conjunto de desenvolvedores que interagiram e colaboraram com o desenvolvedor alvo. Essas explicações são importantes para distinguir as diferentes fontes de informação e deixar clara a origem da qual o referido dado deve ser coletado.

Tabela 3.7 - Métricas associadas em conjunto por cada critério

Opinião	• Número de requisições tratadas pelo DEV alvo; • Número de requisições não tratadas pelo DEV alvo.
Comportamento	• Média aritmética de valores históricos de Opinião do DEV alvo.
	• Média ponderada de valores históricos de Opinião do DEV alvo.
Similaridade[11]	• Valor de localização; • Valor de grau educacional; • Valor de profissão; • Valor de idade; • Valor de gênero.
	• Quantidade de valores históricos de Comportamento do DEV alvo; • Somatório dos valores históricos de Comportamento do DEV alvo.
Relacionamento	• Valor de Opinião do DEV alvo; • Quantidade de DEVs ligados; • Valor de Opinião dos DEVs ligados.
	• Número de Interações entre DEVs ligados e o Desenvolvedor alvo.
	• Nível de privacidade do Artefato; • Número de DEVs compartilhando o Artefato.
Expertise	• Nível de conhecimento pessoal que o DEV alvo assume ter.
	• Medida histórica de Expertise dos DEVs ligados; • Qtde de ligações dos DEVs ligados.
Confiabilidade	• Somatório dos Valores de Confiabilidade dos DEVs ligados; • Somatório dos Valores de Relacionamento dos DEVs ligados; • Qtde de ligações dos DEVs ligados; • Valor de Opinião do DEV alvo.
	• Coeficiente arbitrário de confiabilidade; • Número de interações entre DEVs ligados; • Valor de Expertise do DEV alvo.

[11] Medidas devem ser coletadas de todas as duplas de desenvolvedores.

As métricas representam medidas opcionais a serem coletadas, onde um subconjunto destacado dentro de um mesmo critério é suficiente à análise daquele critério. O aspecto determinante para a escolha do conjunto de métricas a serem aplicadas no modelo é a natureza dos dados disponíveis para análise. Isso torna o modelo mais flexível, deixando explícito que nem todas as medidas precisam ser coletadas, tornando-o adaptável à equipe, à realidade da empresa e aos dados disponibilizados pela empresa para análise da reputação.

As métricas apresentadas devem ser coletadas de forma homogênea, mantendo a mesma unidade em todo o processo para todos desenvolvedores, a cada nova requisição de mudança, compondo uma base histórica de dados de forma a possibilitar o estudo de seus efeitos na evolução da reputação do desenvolvedor.

A operacionalização do processo de identificação da natureza dos dados de entrada, da escolha das métricas e, como consequência, do algoritmo, ou fórmula, a ser utilizado é feito em um módulo de Processamento de Dados para Reputação construído com esse objetivo que será apresentado no próximo capítulo.

3.3 INFLUÊNCIAS ENTRE OS CRITÉRIOS

No intuito de observar a evolução da reputação dos desenvolvedores ao longo do tempo como um todo, torna-se necessário avaliar o impacto entre os critérios de reputação considerados. O processo adotado no estudo das influências entre os critérios iniciou assim que foram definidos os critérios apresentados na Seção 3.1 e, para facilitar a compreensão de todo processo, dividiu-se em duas etapas. Através da primeira etapa foi possível identificar, o que se deu o nome de Influências Teóricas e ao término da segunda etapa, foi possível identificar as chamadas Influências Práticas.

A premissa para a observação de sistemas dinâmicos parte do princípio de que os critérios influenciam uns nos outros. Por isso, na primeira etapa, inicialmente foi realizado um levantamento entre os artigos aceitos no mapeamento sistemático. Esse levantamento objetivou identificar frases, teorias e definições que oferecessem indícios da existência de relações entre os critérios apresentados por cada trabalho. Um exemplo disso, pode ser notado em relação ao critério Comportamento ao ser abordado por Caverlee et al. (2010). Segundo os autores, o comportamento constitui um dos fatores que influenciam a reputação na medida em que afeta a opinião dos demais usuários, de

forma que um comportamento positivo leva a uma opinião positiva e vice-versa. Assim, tal afirmação foi registrada como um indício de que existe uma influência diretamente proporcional entre os critérios Comportamento e Opinião. A essas relações observadas nas teorias, e definições aplicadas aos critérios, foi dado o nome de Influências Teóricas.

Ao final da primeira etapa foi proposto um modelo inicial, ilustrado na Figura 3.3, utilizando-se um Diagrama de Causas e Efeitos (FORRESTER, 1961). Segundo a notação do Diagrama de Causas e Efeitos, uma ligação entre dois critérios de reputação indica uma relação de influência entre eles. As setas existentes nessas ligações vêm identificar a direção da influência entre os critérios, e por fim, os sinais associados a cada relacionamento indicam a tendência de aumento ou diminuição de um critério. Esse modelo inicial para a observação de evolução da reputação dos desenvolvedores foi construído com base nas Influências Teóricas identificadas.

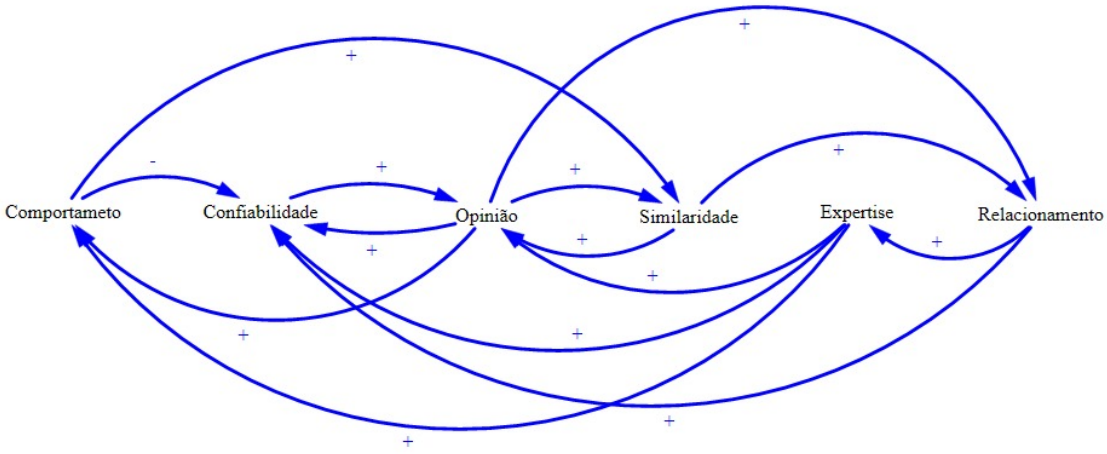

Figura 3.3- Diagrama de Causas e Efeitos Inicial

O inicio da segunda etapa do estudo das influências entre os critérios seguiu a partir do GQM conduzido e apresentado na Seção 3.2. As Influências Teóricas implementadas através das fórmulas e algoritmos analisados na seção anterior e que demonstraram indícios de que foram avaliados e verificados, conforme a pontuação de avaliação, foram chamadas de Influências Práticas. A distinção entre influências Teóricas e Práticas é importante já que estas últimas, por terem sido implementadas e avaliadas, permitem a um gerente coletar e medir tal influência.

Baseado neste cenário, as seguintes questões representam a base para a identificação das Influências Práticas entre os critérios de reputação.

- Questão 01: Avaliação da influência entre Opinião e Comportamento

- Questão 02: Avaliação da influência entre Opinião e Confiabilidade
- Questão 03: Avaliação da influência entre Comportamento e Expertise
- Questão 04: Avaliação da influência entre Expertise e Confiabilidade
- Questão 05: Avaliação da influência entre Comportamento e Similaridade
- Questão 06: Avaliação da influência entre Expertise e Similaridade
- Questão 07: Avaliação da influência entre Expertise e Relacionamento
- Questão 08: Avaliação da influência entre Relacionamento e Similaridade
- Questão 09: Avaliação da influência entre Opinião e Expertise
- Questão 10: Avaliação da influência entre Opinião e Similaridade
- Questão 11: Avaliação da influência entre Relacionamento e Opinião
- Questão 12: Avaliação da influência entre Comportamento e Confiabilidade
- Questão 13: Avaliação da influência entre Relacionamento e Comportamento
- Questão 14: Avaliação da influência entre Similaridade e Confiabilidade
- Questão 15: Avaliação da influência entre Relacionamento e Confiabilidade

Cada uma das 15 questões busca responder se existe influência entre o par de critérios, identificando o sentido da influência e as referências que oferecem suporte à existência da influência. A Tabela 3.8 apresenta os relacionamentos das Influências Práticas entre os critérios de reputação que foram considerados na atualização do modelo, baseados no refinamento a partir do GQM.

Tabela 3.8 – Influências entre Critérios de Reputação

Questão	Direção	Intensidade	Referências
Q01	Opinião -> Comportamento	Sim	(CAVERLEE et al., 2010); (ZUPANCIC; JURIC, 2015)
Q02	Opinião -> Confiabilidade	Sim	(CAVERLEE et al., 2010); (ROSACI et al., 2012)
Q03	Sem indícios	Não	
Q04	Sem indícios	Não	
Q05	Comportamento -> Similaridade	Sim	(QU et al., 2006); (ZUPANCIC; JURIC, 2015)
Q06	Sem indícios	Não	
Q07	Relacionamento -> Expertise	Sim	(ZHANG et al., 2007)
Q08	Similaridade -> Relacionamento	Sim	(CAVERLEE et al., 2010); (VERBIEST et al., 2012)
Q09	Sem indícios	Não	
Q10	Sem indícios	Não	

Q11	Opinião -> Relacionamento	Sim	(CAVERLEE et al., 2010); (VERBIEST et al., 2012)
Q12	Sem indícios	Não	
Q13	Sem indícios	Não	
Q14	Sem indícios	Não	
Q15	Relacionamento -> Confiabilidade	Sim	(CAVERLEE et al., 2010)

Durante a execução do processo e identificação das influências, um aspecto foi observado e deve ser destacado. O conjunto dos 6 critérios e suas influências descrevem a reputação de apenas um desenvolvedor, como se estivesse isolado. No entanto, o trabalho colaborativo leva os envolvidos a se relacionarem, trocarem experiências e conhecimentos, de modo que a colaboração entre os desenvolvedores afeta a reputação desses envolvidos. Assim, o critério de um desenvolvedor influencia o critério de outro desenvolvedor. Para definir o escopo de atuação da influência entre critérios foram criadas duas classificações:

- *Influências intradesenvolvedor*: relativo às influências internas entre critérios do próprio desenvolvedor, ou seja, associadas apenas ao desenvolvedor em análise.
- *Influências interdesenvolvedor*: relativo às influências dos critérios entre desenvolvedores, ou seja, esses colaboram e afetam critérios de outros desenvolvedores.

A Tabela 3.9 apresenta as influências interdesenvolvedor, destacando a direção da influência, baseado nos artigos analisados. Foram considerados ainda, dois desenvolvedores genéricos, Dev_i e Dev_j de modo que se Dev_i colabora com Dev_j então Dev_j influencia a reputação de Dev_i.

Tabela 3.9 - Influências interdesenvolvedor

Direção	Referências
Expertise Dev_j -> Expertise Dev_i	(ZHANG et al., 2007)
Confiabilidade Dev_j -> Confiabilidade Dev_i	(CAVERLEE et al., 2010); (ROSACI et al., 2012)
Opinião Dev_j -> Relacionamento Dev_i	(CAVERLEE et al., 2010)
Relacionamento Dev_j -> Confiabilidade Dev_i	(CAVERLEE et al., 2010); (ROSACI et al., 2012)

As influências intradesenvolvedor e interdesenvolvedor constituem a base para a construção do modelo de reputação para desenvolvedores, processo mostrado na próxima seção. A colaboração entre os membros das equipes é tratada no modelo proposto através das influências interdesenvolvedores.

3.4 CONSTRUÇÃO DE UM MODELO DE REPUTAÇÃO PARA DESENVOLVEDORES DE SOFTWARE

A construção do modelo é a consequência direta dos passos apresentados nas seções anteriores. Após a identificação dos critérios, o levantamento das métricas associadas a cada um e das fórmulas e algoritmos de cálculo, seguido pelo apontamento das influências entre os critérios e o diagrama de causa e efeito, segue-se a referida construção do modelo.

Assim, a Figura 3.4 apresenta um novo Diagrama de Causas e Efeitos resultado de um refinamento no modelo inicial da Figura 3.3, que descreve o modelo produzido a partir da identificação das influências intradesenvolvedor, baseado nas evidências encontradas na literatura técnica.

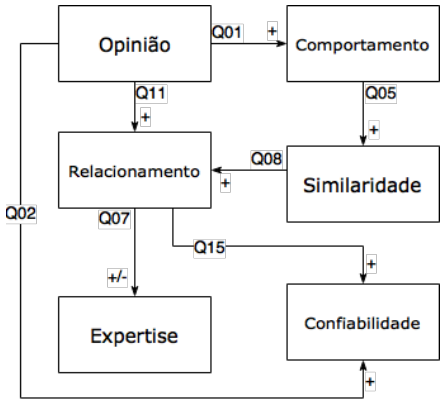

Figura 3.4 - Diagrama Causas e Efeitos com Influências intradesenvolvedor

Seguindo a notação para diagramas de causas e efeitos, as setas existentes nas ligações identificam a direção da influência entre os critérios de reputação e a qual questão (Tabela 3.8) a seta está associada, justificadas pelos indícios encontrados na literatura técnica. O modelo, ilustrado na Figura 3.4, representa a base para calcular a reputação de um desenvolvedor na infraestrutura, objeto deste trabalho. No entanto, para representar também as influências interdesenvolvedor o modelo foi acrescido com

as informações dos demais desenvolvedores envolvidos. A Figura 3.5 mostra essa representação, na qual cada camada indica um desenvolvedor e as setas pontilhadas que atravessam as camadas, são as influências interdesenvolvedores.

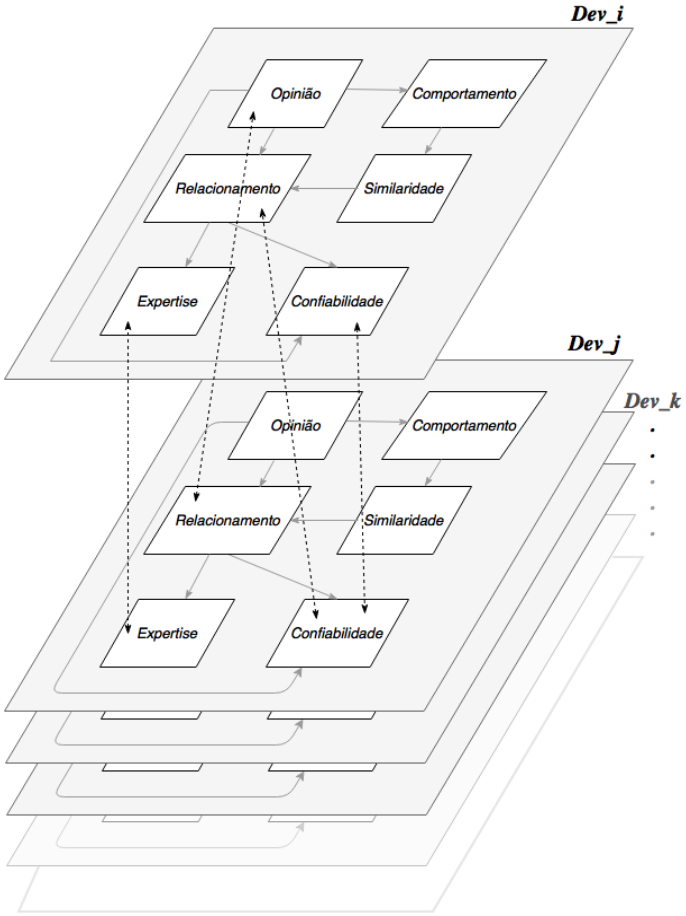

Figura 3.5 - Influências interdesenvolvedor

Os modelos apresentados demonstram a forma pela qual a reputação do desenvolvedor será calculada. Desenvolvedores que colaboram têm sua reputação afetada pelos demais desenvolvedores com os quais colaborou e, assim, será aplicado o modelo com as influências interdesenvolvedor. Tomando como exemplo o diagrama da Figura 3.5, a reputação do Dev_i será calculada através dos critérios do próprio Dev_i, além dos critérios associados às influências interdesenvolvedor dos demais desenvolvedores (Dev_j, Dev_k e assim por diante) que colaboraram com Dev_i. Do mesmo modo, aos desenvolvedores que não colaboram, ficando assim isolados, será aplicado o modelo com as influências intradesenvolvedores. No final, não se tratam de modelos distintos, são um só modelo porém, com complexidade e representação de acordo com o grau de colaboração (existente ou inexistente) dos desenvolvedores envolvidos.

3.5 CONCLUSÃO

Este capítulo apresentou um modelo para o cálculo da reputação de desenvolvedores de software. Nesse sentido, foram apresentados os passos do processo de construção. Iniciou-se por um mapeamento sistemático resultando os critérios de reputação e, na sequência, as possíveis métricas a serem coletadas para a efetiva obtenção dos valores de cada critério foram apresentadas. Em seguida, os estudos que nortearam a análise das influências entre critérios, que subsidiaram a construção desse modelo, foram discutidos.

No próximo capítulo será apresentada a infraestrutura para informações de reputação dinâmica, tomando como base o modelo apresentado neste capítulo.

4. INFRAESTRUTURA PARA INFORMAÇÕES DE REPUTAÇÃO DINÂMICA DE DESENVOLVEDORES DE SOFTWARE

Este capítulo tem como objetivo apresentar a solução para o problema destacado. Relaciona-se ao desenvolvimento de uma abordagem que viabilize o gerenciamento e acompanhamento de informações de reputação de desenvolvedores em equipes de desenvolvimento de software. Além disso, essa abordagem deve prover elementos de visualização e colaboração em um ambiente integrado às atividades de manutenção de software.

4.1 INTRODUÇÃO

A motivação desta pesquisa surgiu no contexto de um dos trabalhos desenvolvidos pelo grupo de pesquisa de pós-graduação da UFJF, o desenvolvimento da *GiveMe Infra* (TAVARES et al., 2015), uma infraestrutura para apoio à realização de atividades de manutenção e evolução colaborativa de software, realizadas por equipes co-localizadas ou geograficamente dispersas. Essa infraestrutura é parte de um Ambiente Interativo baseado em Múltiplas Visões (AIMV) (SILVA et al., 2012).

Uma entrada na infraestrutura da *GiveMe Infra* é a indicação de um módulo, ou componente, que se deseja dar manutenção. Em seguida, os dados históricos referentes ao projeto são importados e filtrados para uma análise estatística, bem como para o cálculo das métricas disponíveis, as quais são descritas na sequência:

- Tamanho: número de classes, número de métodos por classe, número de subsistemas, número de linhas de código fonte;
- Periodicidade: intervalo de tempo entre versões de um projeto;
- Complexidade: profundidade de herança por classe, número de filhos por classe, acoplamento entre objetos, falta de coesão entre métodos e complexidade ciclomática.

A informação gerada é então persistida e disponibilizada para exportação. Dessa forma, é oferecido um provedor de dados, ou seja, um único ponto através do qual as visualizações disponibilizadas pela *GiveMe Infra* e ferramentas externas à infraestrutura podem obter informações sobre os dados históricos processados.

Entretanto, a *GiveMe Infra* não apoia a estimativa de esforço nas atividades de manutenção de software. A atividade de estimativa de esforço visa calcular o tempo e planejamento das manutenções corretivas, adaptativas ou evolutivas. Diante disso, outro trabalho foi desenvolvido no contexto da *GiveMe Infra*, o *framework GiveMe Effort* (MIGUEL et al., 2016) baseado em múltiplas visões interativas para apoiar a manutenção e compreensão de software e está integrado a ambientes de compreensão de software interativos baseados em múltiplas visões para a estimativa de esforço nas atividades de manutenção. Integra-se à *GiveMe Infra* e, através dos recursos oferecidos pela infraestrutura, possibilita a utilização de dados históricos de software para auxiliar no planejamento de manutenções corretivas e adaptativas.

O *framework GiveMe Effort* foi desenvolvido tendo como requisito funcional, prover múltiplas técnicas para o cálculo da estimativa de esforço. Embora utilize dados históricos para apoiar a estimativa de tempo no planejamento de atividades de manutenção de software, esses dados são referentes apenas à perspectiva do sistema em manutenção e das solicitações de mudança por meio das marcações. Não são consideradas, por exemplo, a perspectiva do desenvolvedor responsável por casos anteriores de mudanças pelos quais o software passou. Além disso, o gerente deve associar os desenvolvedores às tarefas, utilizando apenas seu conhecimento prévio sobre cada um deles. Embora o *GiveMe Effort* ofereça suporte ao planejamento das atividades de manutenção, a tarefa de alocação dos desenvolvedores às atividades de manutenção não é abordada.

Apesar da relevância reconhecida, há uma escassez de ferramentas que fornecem suporte à reputação dos desenvolvedores no contexto da manutenção colaborativa de software. Nesse contexto, informações de reputação poderiam ser utilizadas no mecanismo de cálculo da estimativa de tempo de atividades de manutenção de software, levando em consideração a reputação do desenvolvedor, e seus dados históricos, na manutenção e evolução de software, em ambientes distribuídos.

A solução desta pesquisa foi proposta como uma infraestrutura para informação dinâmica de reputação, denominada *IRID (acrônimo para Infrastructure for Reputation Information Dynamics)*. A Figura 4.1, baseada na visão geral da arquitetura do *GiveMe Effort*, mostra a arquitetura da *GiveMe Infra* e a integração do *GiveMe Effort* com suas visualizações e as técnicas utilizadas para a construção da estimativa de esforço. As, setas em vermelho destacam os pontos em que *IRID* atua no *AIMV* para apoiar a

manutenção colaborativa, em projetos de equipes distribuídas. *IRID* faz a integração via dados com o repositório de solicitações de mudança e a ferramenta de gerência Mantis[12], usufruindo da integração existente entre essa ferramenta com a *GiveMe Infra*. Além disso, fornece as informações de reputação ao planejamento das atividades e estimativa do esforço envolvido. A Figura 4.1 destaca ainda, associado à infraestrutura IRID, os componentes ArchiRI, ArchiRIXCom e AD-Reputation, gerados durante o desenvolvimento desta pesquisa.

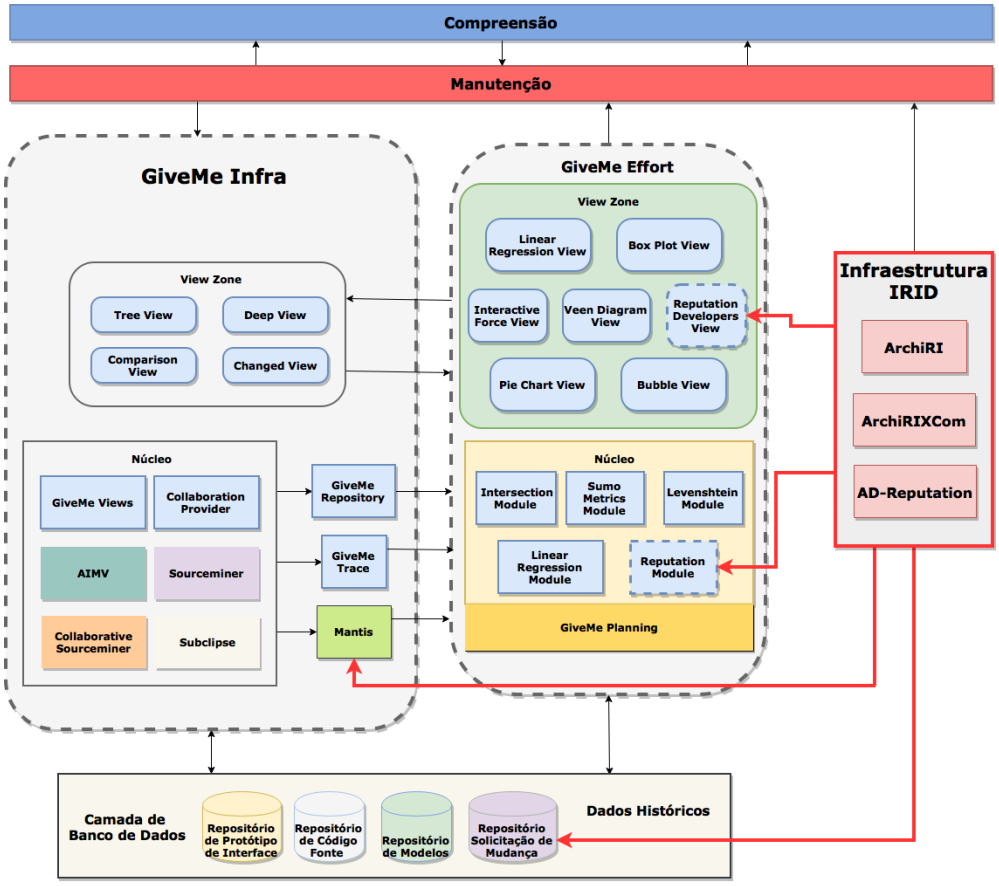

Figura 4.1 - Relação dos componentes integrados da GiveMe Infra *e a solução proposta*

4.2 HISTÓRICO E DEFINIÇÕES

Em uma proposta inicial da infraestrutura IRID, a possibilidade de integração com sistemas e redes sociais de desenvolvedores como o *Stack Overflow*[13] foi analisada.

[12] http://www.mantisbt.org

[13] http://pt.stackoverflow.com/

Stack Overflow é um site de perguntas e respostas para programadores profissionais e entusiastas. É construído e mantido pela comunidade como parte da rede de sites Q&A da *Stack Exchange*[14].

Conforme descrito no Capítulo 2, um dos problemas associados aos sistemas de reputação é a partida fria, ou seja, como tratar a reputação de uma entidade que não possui valores anteriores para o cálculo de sua reputação. Diante disso, o objetivo da integração com *Stack Overflow* foi apoiar o tratamento à partida fria ao estipular uma reputação inicial a uma entidade através da análise de dados provenientes da rede *Stack Overflow*.

A *Stack Exchange* oferece uma API[15] (*Application Programming Interface*) que objetiva oferecer um meio padronizado de busca, acesso e recuperação das informações referentes à participação dos usuários nas comunidades da rede. Embora as métricas envolvidas no cálculo da reputação mostrada no sistema *Stack Overflow* sejam conhecidas, a fórmula ou algoritmo utilizado para agregar essas métricas não é apresentado. Além disso, a API oferece apenas acesso à variação histórica do valor de reputação indicando se a variação é de acréscimo ou redução, sem a possibilidade de acesso ao valor específico de reputação atual ou em datas anteriores.

As restrições descritas foram constatadas no início do desenvolvimento da Infraestrutura IRID. No entanto, ainda que a integração com o sistema *Stack Overflow* fosse concretizada, alguns problemas ainda ficariam em aberto. São eles:

- Sendo a partida fria um problema a ser tratado por todos sistemas de reputação, então o problema se repete quando as entidades não possuem valores na rede *Stack Overflow*.
- Considerando que a reputação calculada está associada a um modelo dinâmico baseado em multicritério, não há uma forma de adaptar o valor inicial extraído pelo *Stack Overflow* aos valores subsequentes registrados pelo uso da solução desse trabalho.

Por se tratarem de questões que abrangem os sistemas de reputação de modo geral, seriam encontradas mesmo que fosse estabelecida integração com outras redes sociais e sistemas diferentes ao *Stack Overflow*. Diante desses desafios optou-se por

---

[14] https://stackexchange.com

[15] https://api.stackexchange.com/docs

outra solução à partida fria que é explicada durante o experimento mostrado no próximo Capítulo.

Considerando este histórico, a constatação dos multi-critérios que influenciam a reputação, bem como seu caráter dinâmico e, em seguida, a definição do modelo de reputação, processo mostrado no capítulo anterior, permitiram que fossem definidos os requisitos não funcionais e funcionais para a solução proposta.

Assim a *IRID* busca atender aos seguintes requisitos não funcionais: (i) a portabilidade, de forma a não estar vinculada a nenhum tipo de sistema operacional específico, além de utilizar padrões abertos que facilitem a sua portabilidade para outros ambientes; (ii) o reuso, possibilitará a utilização dos módulos da ferramenta no futuro; (iii) a interoperabilidade, deve possibilitar a troca de informações de reputação e disponibilização à *GiveMe Infra* e outras ferramentas externas; (iv) a segurança, ao disponibilizar os dados do sistema apenas para seus usuários e identificá-los sempre que esses acessarem o mesmo. Esse requisito tem com objetivo garantir que somente as pessoas autorizadas visualizem e executem ações para as quais possuem permissão.

Além desses aspectos, para que a infraestrutura IRID mantivesse a conformidade com os requisitos não funcionais definidos e os objetivos estabelecidos, o sistema deve atender aos seguintes requisitos funcionais:

- O sistema deve permitir calcular o grau de reputação de um desenvolvedor e da equipe.
- O sistema deve prover múltiplas visualizações para auxiliar na compreensão das informações de reputação de um desenvolvedor, sua relação com a equipe e a evolução da reputação com o passar do tempo.
- O sistema deve permitir que os membros da equipe colaborem durante o processo de análise das visualizações.
- O sistema deve permitir o apoio à tomada de decisão pela equipe.
- O sistema deve oferecer um mecanismo para relacionar e indicar as melhores pontuações de reputação.
- O sistema deve permitir o acompanhamento das pontuações de reputação.
- O sistema deve permitir a projeção futura das pontuações de reputação.
- O sistema deve permitir configurar parâmetros e graus de influência dos critérios no cálculo da reputação.
- O sistema deve permitir importar e exportar informações de reputação.

- O sistema deve permitir armazenar as informações de reputação calculadas e valores dos critérios que as originaram.

Na subseção seguinte, é apresentada a arquitetura da *IRID* considerando os requisitos funcionais e não funcionais apresentados.

4.3 ARQUITETURA

O modelo estrutural de 5 componentes (entrada, processamento, saída, ciclo de *feedback* e armazenamento) apresentado por Liu e Munro (2012), referenciado no capítulo 2, serviu de guia para a arquitetura da IRID. Dessa forma, a infraestrutura encapsula uma solução de sistema de reputação (Figura 4.2) além de agregar outros módulos e componentes referentes aos demais requisitos, como um módulo de inferência (composto por um Modelo Conceitual de representação das Informações de Reputação, denominado CoMoRI, uma ontologia, denominada ArchiRIOnt, e um módulo de análise de troca) que faz parte da ArchiRI. Além deste, um módulo de ranqueamento e filtragem de informações, um módulo de previsão e simulação, e ainda a integração realizada entre a infraestrutura IRID e uma ferramenta, chamada InsightMaker[16], para modelagem e execução de modelos dinâmicos. A Figura 4.2 destaca apenas os módulos pertencentes ao sistema de reputação para facilitar a associação com os componentes comentados por Liu e Munro (2012).

[16] https://insightmaker.com

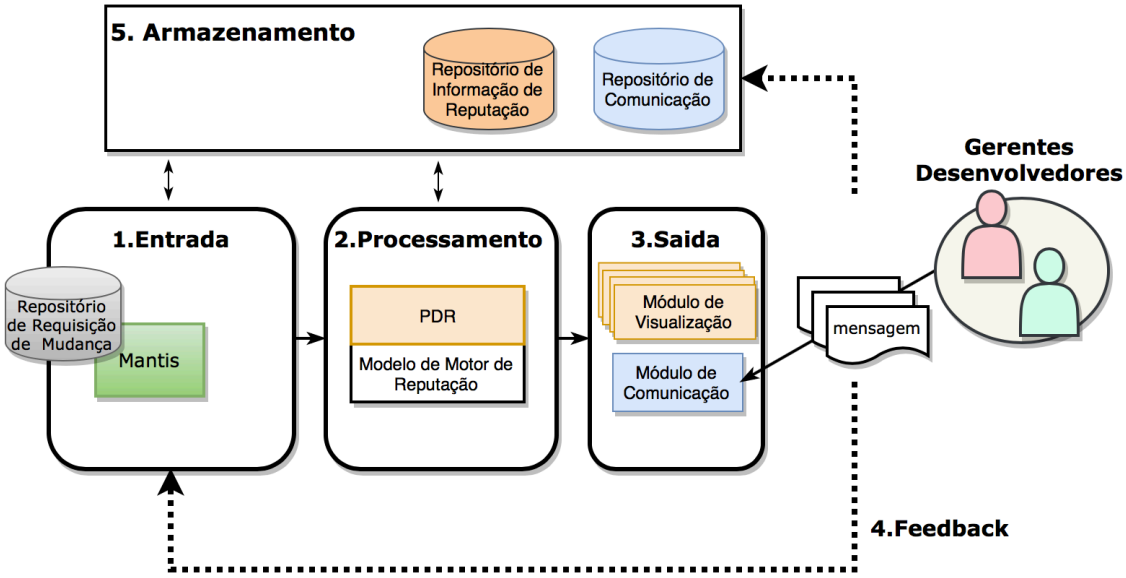

Figura 4.2 – Sistema de Reputação da Infraestrutura IRID

Para cada um desses componentes foram relacionados os módulos da infraestrutura que desempenham a função relativa ao componente. Além disso os usuários finais, gerentes e desenvolvedores, também foram destacados. O objetivo é diferenciar o que compõe o sistema de reputação dentro da infraestrutura e, principalmente, destacar que a infraestrutura oferece recursos que vão além de um sistema de reputação.

No primeiro componente, a entrada é considerada como o repositório de solicitação de mudanças através da integração com a ferramenta Mantis. O segundo componente, se relaciona com o Módulo de Processamento de Dados para Reputação, denominado PDR. Foi criado para transformar os dados brutos nas métricas associadas a cada critério do modelo de reputação. As métricas e valores dos critérios calculados compõem o repositório de informações de reputação. No terceiro componente, associado à saída, as visualizações são disponibilizadas para disseminar as informações de reputação. Além disso, o módulo de comunicação também é acionado disponibilizando o histórico de mensagens como parte importante no reconhecimento de informação relevante e descoberta de conhecimento.

Embora o ciclo de *Feedback* seja opcional, conforme comentado por Liu e Munro, no processo decisório desenvolvedores e gerentes utilizam o módulo de comunicação como ferramenta para a tomada de decisão. Essa decisão gera informações que são registradas no repositório de comunicação.

Durante todo o processo, as informações produzidas são arquivadas no componente de armazenamento composto pelo repositório de comunicação e o repositório de informações de reputação.

Antes de operacionalizar o módulo PDR, é necessário inicialmente atender duas funcionalidades, tais como (i) os dados de solicitação de mudança devem estar disponibilizados e prontos ao acesso, (ii) o conjunto de métricas que servem para medir cada critério deve estar selecionado. Nos casos de critérios com dois ou mais conjuntos de métricas a serem coletadas, deve ser escolhido apenas um conjunto. Por padrão, a escolha dos conjuntos é feita com base no conjunto de dados disponíveis de solicitação de mudança. O acesso aos dados de solicitação de mudança e a definição do conjunto de métricas é feita no momento da implantação da infraestrutura IRID, na empresa que decide utilizar seus recursos. Dessa forma, as medidas coletadas permanecem homogêneas para uma determinada empresa, ou seja, são mantidas sempre na mesma unidade para um determinado critério de reputação. O processo de coleta dos dados é abordado no segundo experimento mostrado no próximo capítulo.

O processo adotado pelo módulo PDR segue o fluxo representado pela Figura 4.3. Inicialmente, os dados brutos são coletados do banco e transformados de forma a representar o conjunto de métricas. Os algoritmos e fórmulas apresentados na Seção 3.2 que serviram de base para a análise das influências entre os critérios, foram implementados no módulo PDR. Assim, cada conjunto de métricas está associado a um algoritmo de cálculo que transforma os dados. Por exemplo, com relação ao critério Opinião, o conjunto de métricas associadas para o cálculo compreende o número de requisições tratadas pelo desenvolvedor alvo. Assim, o módulo PDR busca no banco apenas os dados das solicitações de mudança tratados pelo desenvolvedor alvo. Em seguida, o conjunto de métricas é aplicado no algoritmo, ou fórmula, ao qual está associado. No caso do critério Opinião, a cada requisição tratada é atribuído o valor de uma avaliação positiva (igual a 1), e o número de solicitações atendidas é ponderado pelo total de solicitações. Essa abordagem foi apresentada em (LÉLIS et al., 2018). O resultado da execução do algoritmo indica o Critério calculado e que gera a saída para o repositório de informações de reputação e para as visualizações associadas.

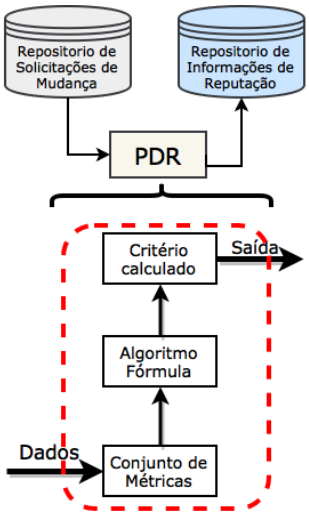

Figura 4.3 – Fluxo da informação na execução do Módulo PDR

A Figura 4.4 mostra a arquitetura da infraestrutura IRID. Segurança e a configuração são duas camadas associadas aos requisitos não funcionais e permeiam a utilização da infraestrutura.

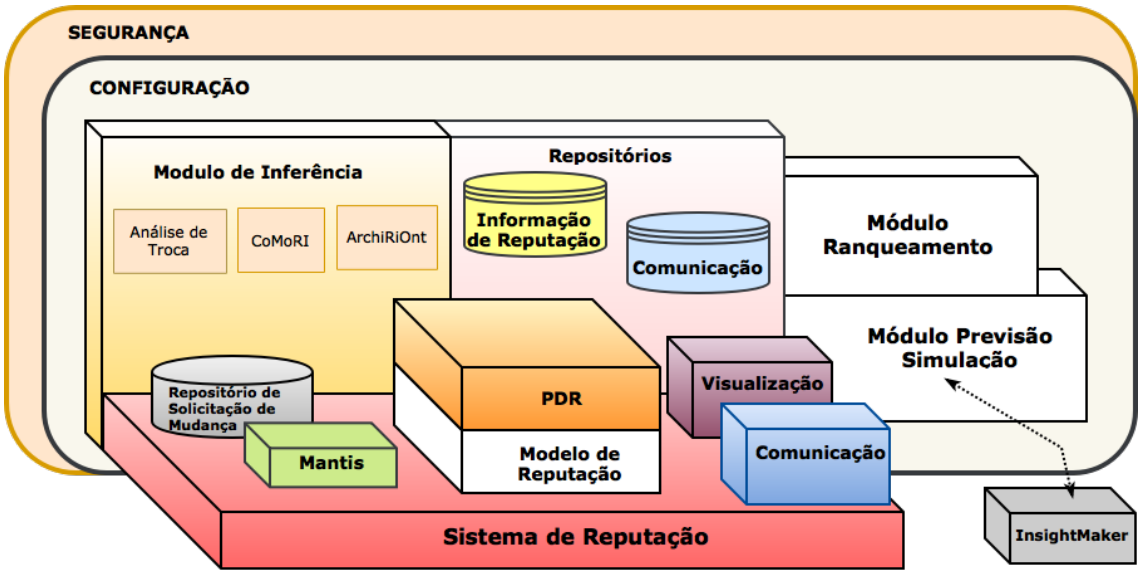

Figura 4.4 – Infraestrutura IRID

A configuração permite a personalização em alguns módulos da infraestrutura sendo possível, por exemplo, desabilitar a apresentação de alguma visualização, além de permitir alterar os temas das telas como tamanho de fonte e cores. A segurança controla o acesso ao sistema, limita a utilização de funcionalidades gerenciais, e a exibição das visualizações adequadas aos usuários. Controla ainda a troca de mensagens, síncronas e assíncronas.

Os demais módulos e recursos integrados que compõem a infraestrutura, bem como o fluxo das informações e a comunicação entre módulos, são mostrados nas próximas subseções.

4.2.1. MÓDULO DE INFERÊNCIA

A interoperabilidade e a possibilidade de troca de informações de reputação são características importantes a serem alcançadas (CASARE; SICHMAN, 2005) (HENDRIKX et al., 2015). No entanto, para que as informações sejam armazenadas e a coerência na informação trocada seja assegurada, é importante saber o nível em que as informações de reputação estão localizadas, bem como o que representam.

Os conceitos de reputação individual e reputação de grupo foram estudados por Mui et al. (2002), sendo representados pela camada de entidade e de grupo. Hendrikx et al. (2015) analisaram sistemas de reputação acadêmicos e comerciais e constataram que, na maioria deles, o foco foi estabelecer a reputação do indivíduo. Segundo (MUI et al., 2002), a reputação individual é determinada sob o ponto de vista de outra entidade após uma interação. A reputação de grupo pode ser definida como a média das reputações individuais de todos os seus membros. Por depender da experiência e características individuais, seu valor pode variar entre grupos.

Assim, houve a necessidade de considerar também o domínio ao qual a reputação está associada, ampliando a noção de reputação individual e de grupo. O conceito de reputação do domínio é definido como uma característica, uma propriedade, ou a categoria geral na qual a reputação foi criada (ALNEMR; MEINEL, 2012). Por exemplo, o cargo ou papel que um desenvolvedor desempenha em uma equipe pode ser classificado como reputação de domínio.

Quando os conceitos de entidade, grupo e domínio são utilizados separadamente, a capacidade de melhorar a tomada de decisões, combinando os dados, é perdida. CoMoRI (acrônimo para Conceptual Model for Reputation Information) é um modelo conceitual proposto nesta pesquisa para a representação das informações de reputação, sendo composto de três camadas que representam os pontos de vista que a reputação é percebida e pode ser analisada pelos envolvidos. A Figura 4.5 apresenta o modelo CoMoRI com suas camadas e o que representa cada uma.

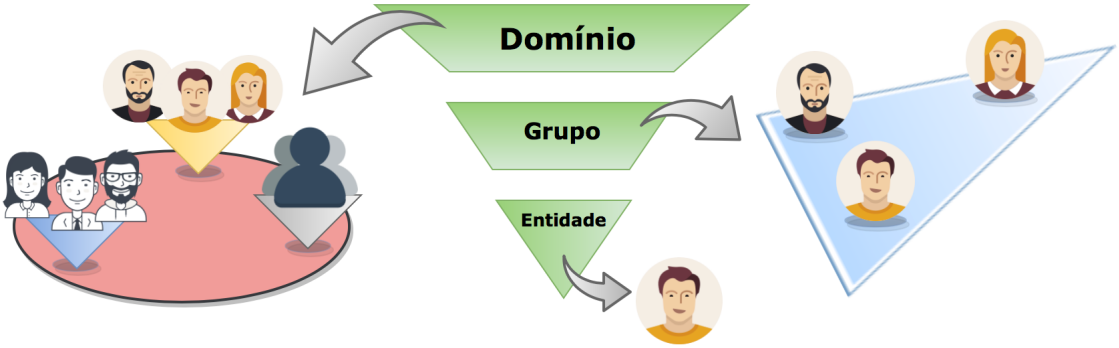

Figura 4.5 – As três camadas do modelo conceitual CoMoRI

A primeira camada é a de entidade, ou nível de percepção individual. Nesse estágio, o foco é o indivíduo que tem uma reputação, características próprias, oferece sua opinião sobre outros indivíduos e também recebe opiniões dos outros a seu respeito. Em um ambiente distribuído de manutenção e evolução de software, a entidade seria um desenvolvedor. Possui características próprias e competências que podem ser estratégicas e as distinguem dos demais, como a capacidade de comunicação.

No entanto, essas entidades podem se relacionar, trocar informações e ter opiniões semelhantes. Assim, podem ser agrupadas, tal qual uma sociedade ou uma equipe. Quando isso ocorre, essas entidades agrupadas ascendem à camada de grupo. Nesse nível, diferentes grupos podem manter as características específicas que as distinguem entre si. Um grupo também pode opinar sobre outro grupo e receber opiniões, de forma semelhante ao que ocorre na camada de entidade. A camada de grupo também tem características semelhantes e os grupos podem trocar informações e se relacionar, alcançando o terceiro nível. Como exemplo, pode-se citar as diferentes equipes formadas em uma organização para os ciclos da manutenção de um projeto de software. Trabalham em conjunto, sob as mesmas regras e trocam informações sobre os artefatos desenvolvidos.

A terceira camada do modelo representa o domínio ao qual indivíduos ou grupos pertencem, criando a noção de reputação de domínio. De forma análoga às camadas apresentadas, diferentes domínios têm características específicas. Essas características podem ser entendidas como as habilidades que se espera encontrar nos indivíduos pertencentes a um dado domínio. Em uma organização virtual, entidades individuais e em grupo podem unir-se temporariamente para resolver um problema comum ou trabalhar em uma tarefa (HENDRIKX et al., 2015). Um exemplo de domínio ao qual cada equipe e seus membros pertencem, pode estar associado à natureza da tarefa de

manutenção a ser realizada. Essa tarefa pode ser uma mudança em um trecho de código, ou o teste unitário do trecho alterado. Cada domínio possui competências específicas, por exemplo, sobre o desenvolvedor que realizará as alterações no código, espera-se que tenha conhecimento das boas práticas de programação e padrões de projeto. Já, ao realizar o teste no trecho alterado, espera-se que seja capaz de projetar casos de teste eficientes que consigam capturar o defeito, caso exista.

O módulo chamado de Análise de Troca foi desenvolvido para analisar duas maneiras possíveis para troca de informações de reputação. A primeira tem o foco no domínio do elemento analisado, e a segunda diz respeito ao grupo ao qual pertence. Na análise de troca de domínio, o objetivo é identificar os domínios aos quais os membros poderiam pertencer, além dos critérios que explicariam essa mudança de domínio. Na análise de troca de grupos, o objetivo é identificar grupos com domínios semelhantes, mostrando ao usuário cada grupo que poderia receber a entidade alvo e os critérios que seriam usados no alinhamento entre os grupos.

Para a operacionalização desse módulo era preciso que os dados de reputação estivessem com uma representação coerente que permitisse tais atividades como identificar grupos e domínios e realizar o alinhamento através de suas características. Uma ontologia, denominada ArchiRIOnt, foi criada. Ontologias permitem a reutilização do conhecimento ao compartilhar uma interpretação comum do domínio, facilitam a interoperabilidade e o processamento inteligente (VIPUL et al., 2008).

A literatura técnica mostra que o uso de ontologias vem sendo aplicado para representar as informações de reputação. A ontologia do objeto de reputação (ALNEMR; MEINEL, 2012), por exemplo, foi definida como uma ontologia genérica para representar a reputação de entidades, chamada *Reputation Object* (RO). Nessa ontologia, a reputação é representada com a forma de um valor de reputação, o objeto de reputação (RO). Esse objeto contém informações sobre a reputação de uma entidade em múltiplos domínios. A RO é construída *offline* ou *online* durante a interação dos indivíduos. É um objeto genérico que muda de acordo com o domínio e a preferência do usuário. Em geral, guarda um perfil (qualificações, por exemplo) sobre uma entidade que é coletada a partir de fontes de informação heterogêneas.

Outras ontologias também influenciaram esta pesquisa. A agregação de modelos e conceitos feita em (CASARE; SICHMAN, 2005) resultou em uma ontologia Funcional de Reputação. Essa ontologia tem seu foco em sistemas multiagentes, e é

utilizada como um conhecimento de reputação comum e compartilhado entre os agentes para ativar o nível de integração semântica envolvida na interação de agentes de software. Apesar disso, foi usada nesta pesquisa para expandir a noção de entidade considerando os diferentes tipos de entidades suportados pelos modelos apresentados em (CASARE; SICHMAN, 2005). Além da entidade "Pessoa", também é apresentado um modelo para a entidade "Produto", "Agente" e "Serviço". Essa expansão possibilita representar as interações que ocorrem na prática, em um ambiente real.

Considerando a necessidade de tratar requisitos tais como facilidade de integração e capacidade de estenderem-se a vários contextos, essas ontologias foram escolhidas como base. No entanto, não havia a representação explícita dos conceitos de entidade e seus tipos, do conceito de grupo e nem tampouco de domínio, entre outros, que permitiriam a correta correlação com o modelo de reputação proposto. Esse fato demandou a composição das ontologias para potencializar a representação das informações de reputação e torná-la mais abrangente.

Como resultado da fusão das ontologias apresentadas surgiu a ArchiRIOnt. Em seguida, foi realizado um mapeamento com o objetivo de integrar o modelo CoMoRI à ontologia. Buscou-se ainda, tornar possível estabelecer a reputação relacionada a cada nível: a reputação da entidade ou indivíduo, a reputação do grupo e a reputação do domínio.

Uma classe "Domain" foi criada na ontologia ArchiRIOnt cujo objetivo é explicitar a reputação que se altera a cada domínio considerado. Por exemplo, um analista de sistema pode ter alta reputação ao analisar o impacto e planejar mudanças no código fonte um uma equipe de manutenção. Porém, ao mesmo tempo, sua reputação como programador pode ser baixa. Nesse caso, o domínio estaria associado ao papel desempenhado pelo analista.

Com o nível de grupos, pode-se responder à seguinte pergunta: *Com quem a Entidade "A" estabeleceu um grupo?* Dessa forma, foram criadas duas propriedades de objeto "*isSourceOf*" associada ao conceito de fonte de informação e "*isTargetOf*" associada ao conceito de alvo, para indicar que duas entidades estão relacionadas formando um grupo. As relações entre grupos e entre grupos e entidades, caracterizam o domínio em que ambos estão inseridos. Assim, uma restrição foi adicionada na qual "*Source*" e "*Target*" estão no mesmo domínio para caracterizar o grupo formado.

Outras questões que merecem destaque são (1) *como definir a reputação de "D" conhecendo a reputação do grupo a que pertence?; e* (2) *como saber a intensidade da relação entre entidades ou entre grupos, ou entre os domínios?* Para responder a essas perguntas, foi criada uma propriedade de objeto, chamada "Coeficiente". Com isso, é possível determinar a escala de intensidade a adotar e o valor absoluto dessa intensidade. Por outro lado, esse coeficiente pode ser usado para representar o grau de influência que a reputação do grupo tem sobre a reputação individual, ou a ponderação entre os critérios de reputação.

Um critério no modelo dinâmico de reputação, apresentado no capítulo anterior, é calculado por fórmulas e algoritmos associados à métricas e variáveis. No entanto, essas métricas podem ser outros critérios anteriormente calculados. Um exemplo disso é o Critério Relacionamento, se for calculado segundo a fórmula de Caverlee et al. (2010) utiliza o critério Opinião como métrica. Dessa forma, métricas e variáveis foram chamados na ontologia de subcritérios, com a possibilidade de um critério ser também subcritério de outro. Esse recurso foi implementado na ontologia adicionando uma propriedade de objeto chamada "*hasCriterion*" na classe "Criteria", já presente na ontologia. A influência de cada critério no valor calculado de reputação também foi indicada pelo uso da propriedade de objeto "Coeficiente".

A reputação pode mudar ao longo do tempo evidenciando a necessidade de um ponto de vista evolutivo do seu valor. Esse recurso foi implementado na ontologia através de uma nova classe chamada "Evolution" composta por um histórico de valores, o valor atual da reputação e os momentos ao longo do histórico em que houve uma alteração de contexto.

A ontologia ArchiRIOnt, bem como as decisões de implementação, e exemplos de aplicação, são apresentados e discutidos em maiores detalhes em (LÉLIS et al., 2016b). A Figura 4.6 apresenta um modelo simplificado da ontologia ArchiRIOnt com o relacionamento entre as classes. As contribuições alcançadas pelo *merge* das ontologias e o mapeamento realizado são destacadas na Figura 4.6, tais como: (i) a inserção do conceito de Entidade, (ii) a especificação das entidades envolvidas, (iii) a ampliação do conceito de Critério, com a inserção do conceito de subcritério e a possibilidade de identificar critérios indiretos para a reputação, (iv) a definição do conceito de evolução, e (v) a separação e definição do conceito de Domínio.

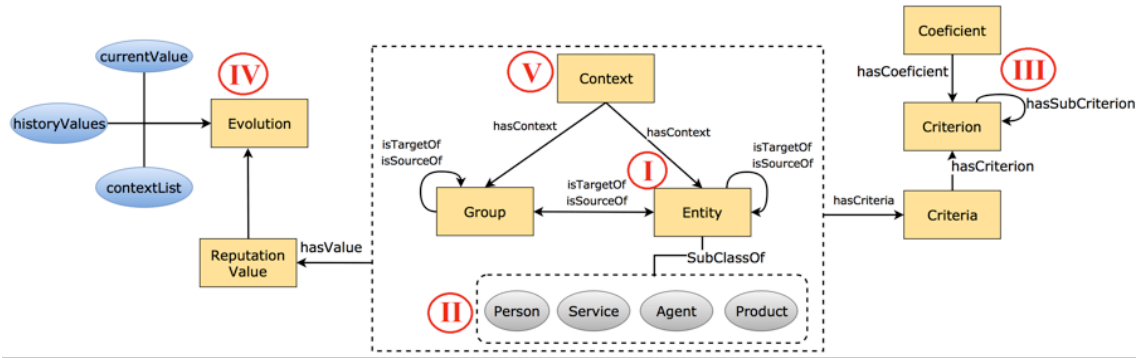

Figura 4.6 - Modelo simplificado da ontologia ArchiRIOnt

As informações manipuladas pelo módulo de análise de troca são dependentes das inferências feitas através da ontologia. Os grupos são formados através da ontologia e associados aos respectivos domínios. Através dos critérios usados pode-se indicar o grupo e domínio aos quais a entidade alvo pertence. Além disso, também é capaz de indicar os grupos e domínios que possuem interseção com o grupo e domínio da entidade alvo. Dessa forma, esse recurso é importante para apoiar os gestores na formação de grupos e intercâmbio de membros entre grupos e domínios.

4.2.2. MÓDULO DE RANQUEAMENTO

Trata-se de um módulo importante no processo de alocação dos desenvolvedores às solicitações de mudança. O ranqueamento é exibido automaticamente, ou quando solicitado via menu, antes que o gerente indique o desenvolvedor que deve tratar determinado caso. O processo de funcionamento inicia pela coleta de dados do último cálculo de reputação realizado, ou a partir de uma data escolhida pelo gerente. Em seguida, esses são comparados com as informações atuais de reputação. A apresentação do resultado é feita através de uma tabela que mostra um ranqueamento dos desenvolvedores pelo valor de reputação. A primeira coluna representa a posição, na sequência o nome, o valor de reputação e os valores de cada critério. É exibida ainda uma indicação relativa à posição do desenvolvedor na lista, representando se subiu, desceu ou continua na mesma posição da última leitura de dados.

A Figura 4.7 destaca um trecho do ranqueamento disponibilizado através da comparação de informações de reputação.

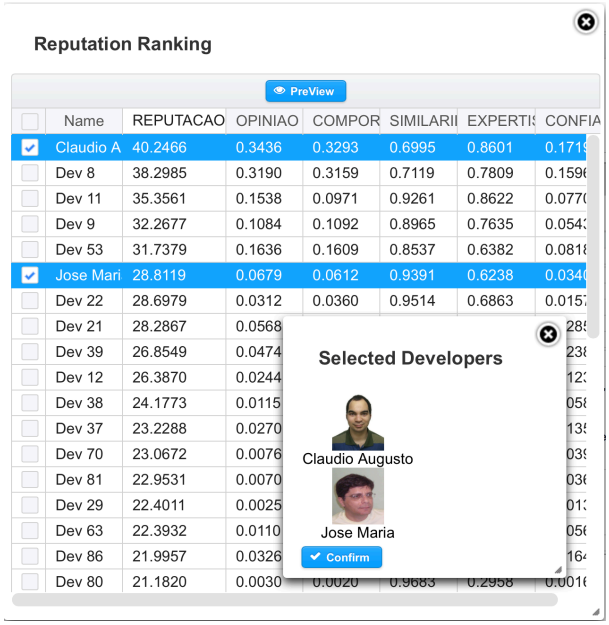

Figura 4.7 – Trecho do resultado de ranqueamento

Quando o módulo de ranqueamento é chamado via menu, simplesmente é mostrada a listagem como forma de consulta às informações. No entanto, quando o módulo é acionado através da tela responsável pela alocação dos desenvolvedores à solicitação de mudança, tem-se a possibilidade de selecionar uma ou mais linhas da listagem e confirmar a escolha. Como resultado, os desenvolvedores selecionados, como no exemplo da Figura 4.7, são associados à solicitação de mudança e deverão tratar em equipe a tarefa. Esse mecanismo de seleção é mostrado no experimento do próximo capítulo.

Neste momento não é o objetivo do módulo de ranqueamento recomendar desenvolvedores para as solicitações de mudança. No entanto, as informações processadas e inferidas pelo módulo de inferência, mostrado na subseção anterior, enriquecem a análise das informações de reputação e a descoberta de conhecimento, possibilitando a definição de perfis dos desenvolvedores que tratam os casos através do grupo e do domínio aos quais estão associados e as características de cada caso tratado. Dessa forma, o módulo de inferência, juntamente ao módulo de ranqueamento, pode ser considerado um primeiro passo para a criação de um módulo de recomendação para a infraestrutura IRID.

4.2.3. MÓDULO DE PREVISÃO E SIMULAÇÃO

Este módulo foi construído com o objetivo de ser uma ponte entre as informações de reputação armazenadas e a ferramenta Insight Maker[17], cuja integração à IRID é mostrada na próxima subseção. O modelo de reputação, ao ser representado por um diagrama de estoque e fluxo, deve seguir regras explícitas para sua construção. Esse tipo de diagrama se destina a servir de base para uma simulação de computador. Associada a cada ligação entre um estoque e seu fluxo existe uma fórmula que define explicitamente o relacionamento e é utilizada para calcular os valores durante a simulação.

No processo de simulação são utilizadas, como entrada, equações construídas através de regressão linear, aplicando o método de mínimos quadrados, para identificar a tendência de comportamento daquele critério de reputação, podendo apenas assumir valores reais não negativos. Regressão linear, apesar da possibilidade de aumento do erro, é condizente com uma análise semiquantitativa de dados, em que a tendência do comportamento de uma variável é o que deve ser considerado, mais do que seus valores individuais. O método de mínimos quadrados é uma técnica de otimização matemática que procura encontrar o melhor ajustamento para um conjunto de dados, tentando minimizar a soma dos quadrados das diferenças entre a curva ajustada e os dados, onde essas diferenças são chamadas de resíduos (MONTGOMERY, 2008).

Para a construção das equações utilizadas no processo de simulação, é preciso definir o período dos dados históricos a ser considerado, na IRID foi definido como padrão um ano, mas pode ser configurado. Diante disso, este módulo é responsável pela geração dessas equações. Uma possível abordagem seria a utilização de planilhas eletrônicas, no entanto, optou-se por automatizar o processo através deste módulo. Como o modelo proposto apresenta influências entre os critérios de reputação, ou seja, um determinado critério pode ser influenciado por outros, a construção das equações leva isso em consideração, sendo importante, nesse contexto, a normalização dos dados, quando necessário.

Dessa forma, as equações são construídas a partir das influências identificadas entre os critérios de reputação. O critério Opinião é uma variável considerada em função do tempo decorrido. Os demais critérios são calculados a partir das funções dos outros critérios que nele influenciam. Por sua vez, tais critérios são calculados a partir da

[17] https://insightmaker.com

regressão linear das informações de reputação, dos desenvolvedores em observação. O experimento, mostrado no próximo capítulo, deixa mais claro o processo de obtenção dessas equações.

O módulo de Previsão e Simulação gera um arquivo em formato xml, com utilização de *tags* para representar cada elemento do modelo, um exemplo é mostrado no APÊNDICE A. A ferramenta Insight Maker permite a manipulação gráfica do modelo. Embora a integração e o processo de simulação seja exemplificado na próxima subseção, a Figura 4.8 mostra a tela do sistema, na IRID, através da qual o gerente pode operacionalizar a ferramenta integrada. Inicialmente, o gerente escolhe o intervalo de tempo (destacado em "A"), em meses, a serem considerados na coleta dos dados históricos, necessários para a geração das equações. Por padrão esse valor é de 12 meses contados da data atual (destacado em "B"). Por ser um valor arbitrário, apenas para facilitar o processo, tanto a data de início, quanto o período podem ser alterados se o gerente desejar. Em seguida, é possível exportar o arquivo *.InsightMaker* (destacado em "C") com todas as informações para a simulação.

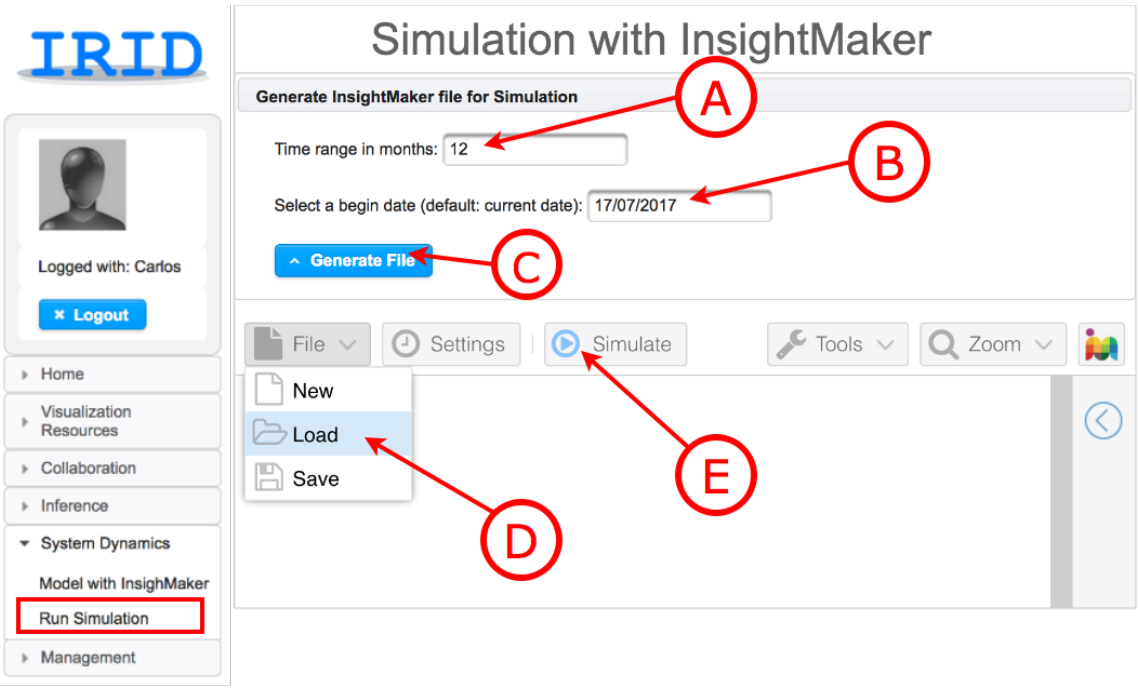

Figura 4.8 - Geração de Arquivo e Simulação

Os parâmetros de configuração necessários para rodar a simulação são padronizados durante a geração do arquivo, simplificando o processo. Dessa forma, o gerente pode focar na análise dos resultados. A Figura 4.8 mostra ainda, o menu da

ferramenta onde é feita a importação do arquivo gerado (destacado em “D”) e o botão de acionamento da simulação (destacado em “E”).

A partir das informações dos valores iniciais dos critérios de cada desenvolvedor em análise, e as respectivas equações do modelo, é gerado o arquivo com extensão *.InsightMaker*. Assim, pode-se executar o processo de simulação propriamente dito, realizado através da ferramenta Insight Maker, integrada à infraestrutura IRID, mostrada na próxima subseção.

O módulo de Previsão e Simulação também é responsável por receber os dados resultantes da simulação e armazená-los no repositório de informações de reputação. O objetivo é apoiar o acompanhamento da evolução da reputação. Através da comparação entre a tendência futura, simulada, dos valores de reputação e os valores reais calculados, a cada nova ocorrência de solicitação de mudança. Dessa forma, o gerente pode acompanhar se o comportamento da reputação que foi simulado, está de fato ocorrendo.

4.2.4. INTEGRAÇÃO COM A FERRAMENTA INSIGHTMAKER

O Insight Maker é um ambiente de modelagem e simulação. Tem suporte a multiusuário, é baseado em web, gratuito utilizando padrões livres e seu código é aberto. Oferece um ambiente flexível e fácil de usar para se desenvolver e compartilhar informações sistêmicas.

Embora o Insight Maker tenha sido originalmente destinado a ser um ambiente de simulação, percebeu-se que havia uma variedade de diferentes tipos de modelos que poderiam ser, prontamente, suportados no ambiente. Exemplos desses tipos são os mapas mentais, geralmente usados para descobrir níveis de detalhes em conceitos continuamente refinados, os mapas de diálogo que ligam um conceito e um questionamento a características positivas e negativas, os baseados em agente e os já comentados, diagrama de causas e efeitos e o diagrama de fluxo e estoque.

A escolha por integrar e utilizar a ferramenta InsightMaker se baseou no fato da ferramenta atender requisitos não funcionais como, a facilidade de uso e portabilidade, utilizando padrões e tecnologias livres. Além disso, permite a exportação e importação das informações para executar as simulações.

A integração foi realizada a partir do módulo de Previsão e Simulação que gera o arquivo a ser importado pela ferramenta. Essa é uma funcionalidade gerencial e, para acionar esse recurso da infraestrutura IRID, o gerente não necessita ter conhecimento

em modelagem de Diagramas de Fluxo e Estoque, bastando importar o arquivo de extensão *.InsightMaker* (um exemplo pode ser visto em APÊNDICE A) e acionar o botão de simulação, conforme destacado em "D" na Figura 4.8.

O resultado da simulação é mostrado, automaticamente, através de uma tela (Figura 4.9) que apresenta uma aba para cada desenvolvedor analisado (destacado em I). Ao clicar em alguma dessas abas, é exibido ao gerente a tendência de crescimento, ou decrescimento dos critérios do desenvolvedor escolhido. A última aba (destacada em II) reúne as tendências de crescimento, ou decrescimento, para a reputação de cada desenvolvedor, conforme ilustrado na Figura 4.9. O gráfico apresenta o comportamento de simulação da reputação de cada um dos desenvolvedores de software, em função do modelo considerado e das equações para elas construídas.

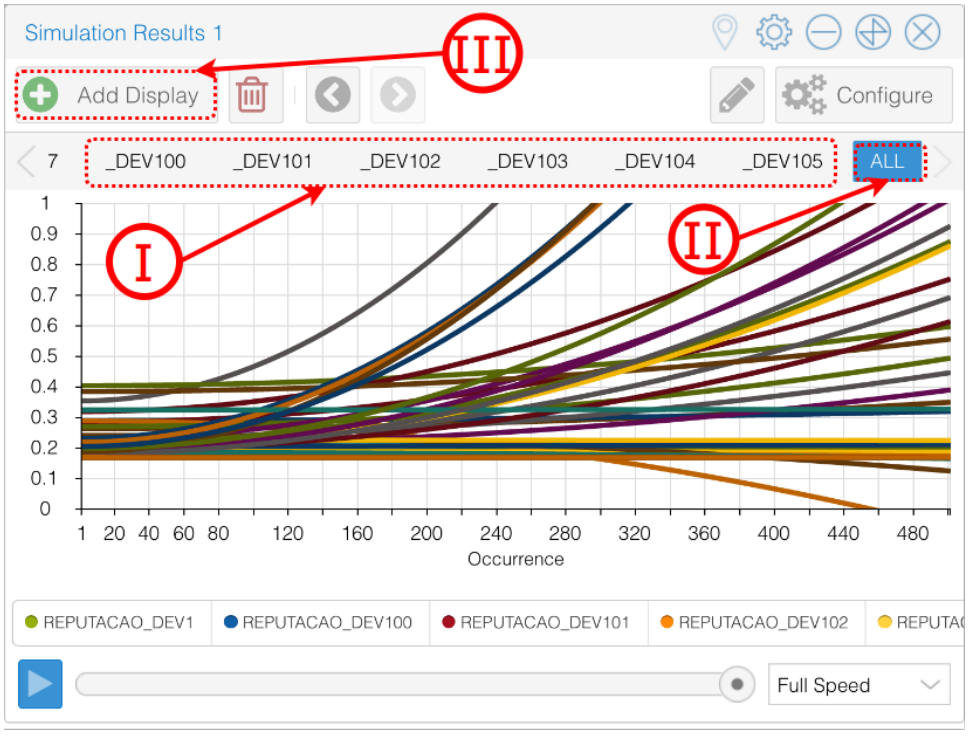

Figura 4.9 – Simulação com comportamento da reputação de desenvolvedores

É importante ressaltar que o eixo X apresenta o número de ocorrências futuras sendo simuladas, em função do tempo. Cada ocorrência representa uma solicitação de mudança que foi tratada, quando considerados os dados históricos e uma que será tratada quando considerada a simulação. No exemplo, um total de 500 ocorrências são simuladas. Esse número de ocorrências foi utilizado apenas com o objetivo de apresentar o comportamento das equações de forma visualmente mais clara.

O eixo Y representa o comportamento da reputação de cada desenvolvedor. Para o eixo Y, a escala foi limitada entre 0 e 1. Dependendo das escalas com que cada critério foi medido, é possível que tenha sido necessário a normalização do valor. No processo de simulação, as equações geradas podem levar algumas linhas a valores superiores ao limite de 1 e, por isso, pode ocorrer de nem todas as curvas serem visualizadas simultaneamente no gráfico. O objetivo aqui é mostrar o comportamento de cada uma delas, relacionado à tendência de crescimento focando em uma análise semiquantitativa.

As abas mostradas na Figura 4.9 são definidas automaticamente pelo processo de geração do arquivo de simulação. Novas abas podem ser criadas pelo botão "Add Display" (destacado em III). Uma nova aba pode ser utilizada para mostrar as informações em formatos diferentes e assim permite exportar o resultado da simulação.

São oferecidas 3 formas de apresentar os resultados da simulação, tais como: a Linha do Tempo, mostra como os valores de uma ou várias variáveis mudam através do tempo, o *Scatter Plot* permite analisar como duas variáveis mudam em conjunto. Outra forma de apresentar é através de Tabelas que oferecem os valores precisos das variáveis através do processo de simulação, e permitem ainda que os dados sejam exportados para arquivo nos formatos csv, xls e pdf. É possível escolher um nome para a aba e quais variáveis serão mostradas no formato de apresentação escolhido.

O gerente pode exportar e armazenar os dados da simulação para acompanhar e se antecipar caso os dados reais sigam a tendência simulada. Ao perceber uma tendência de decrescimento de um desenvolvedor, uma decisão pode ser tomada antes que se agrave. Por exemplo, oferecer um treinamento específico, ou formar grupos com desenvolvedores com mais conhecimento ou experiência. Esses dados ao serem armazenados são utilizados pelo módulo de Ranqueamento, já apresentado, e mostrados nas visualizações gerenciadas pelo módulo de Visualização mostrado mais a frente.

4.2.5. MÓDULO DE COMUNICAÇÃO

A primeira atividade para a concepção do módulo de comunicação foi o desenvolvimento da ArchiRIXCom (SILVA, 2017). Foi planejada para atuar em uma limitação que a ferramenta ArchiRI possuía. Com o desenvolvimento do ArchiRIXCom, ArchiRI passou a permitir a tomada de decisão conjunta através de aspectos de comunicação. A abertura de fóruns e pelas discussões registradas e

gerenciadas, as decisões entre desenvolvedores e o gerente responsável eram tomadas em relação ao esforço realizado para tratar as solicitações de mudança.

A Figura 4.10 apresenta um fórum de discussões iniciado através da ArchiRIXCom, destacando uma decisão já finalizada pelo gerente realçada em verde. No painel mostrado ficam disponíveis todas as discussões iniciadas por qualquer usuário do sistema juntamente com os comentários realizados nas mesmas. Através deste fórum, um usuário pode iniciar uma nova discussão, ou comentar uma discussão iniciada previamente. Para iniciar uma discussão, o desenvolvedor ou gerente deve se identificar através da caixa de seleção, adicionar a mensagem na caixa de texto e enviar. Para inserir um comentário é disponibilizado um botão "Comentar" referente a discussão escolhida e um campo de texto é exibido.

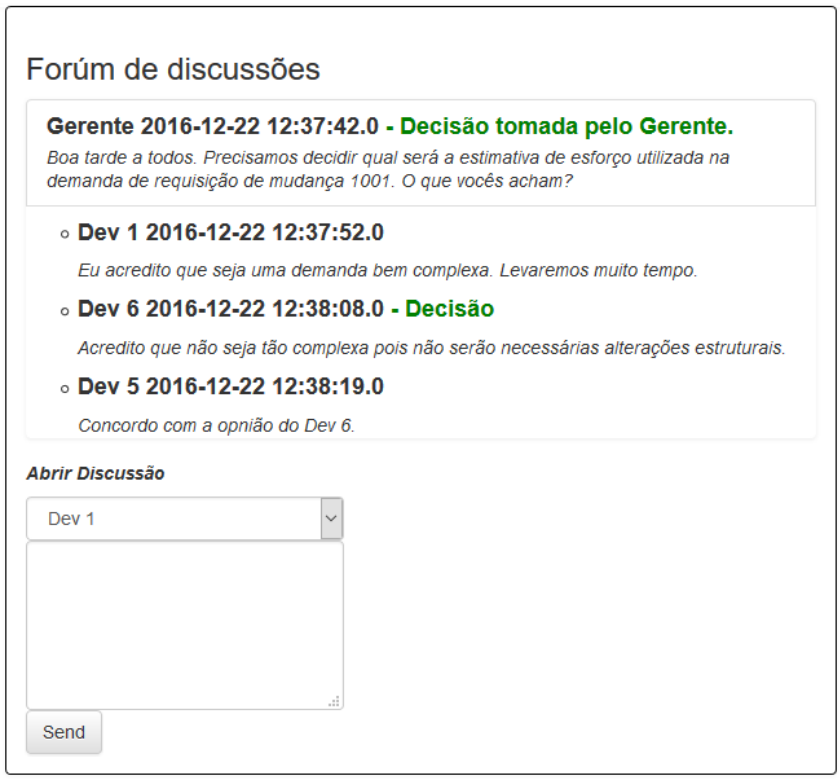

Figura 4.10 - Fórum de discussões da ArchiRIXCom (SILVA, 2017)

Com o desenvolvimento do ArchiRIXCom tornou-se viável o desenvolvimento de novas visualizações, tais como: *Forum Activity View*, *Last Mouth Forum Activity View* e *Historic View*, que posteriormente foram integradas aos *Dashboards* a serem mostrados na próxima subseção.

Atualmente, o módulo de comunicação é responsável também pela gerência do envio e recebimento de mensagens síncronas, e o envio de mensagens em massa para todos os membros ao mesmo tempo. O envio de mensagens em massa é disponibilizado através da tela de comunicação, conforme ilustrado pela Figura 4.11, onde desenvolvedor e gerente podem deixar sua contribuição, ou comentário.

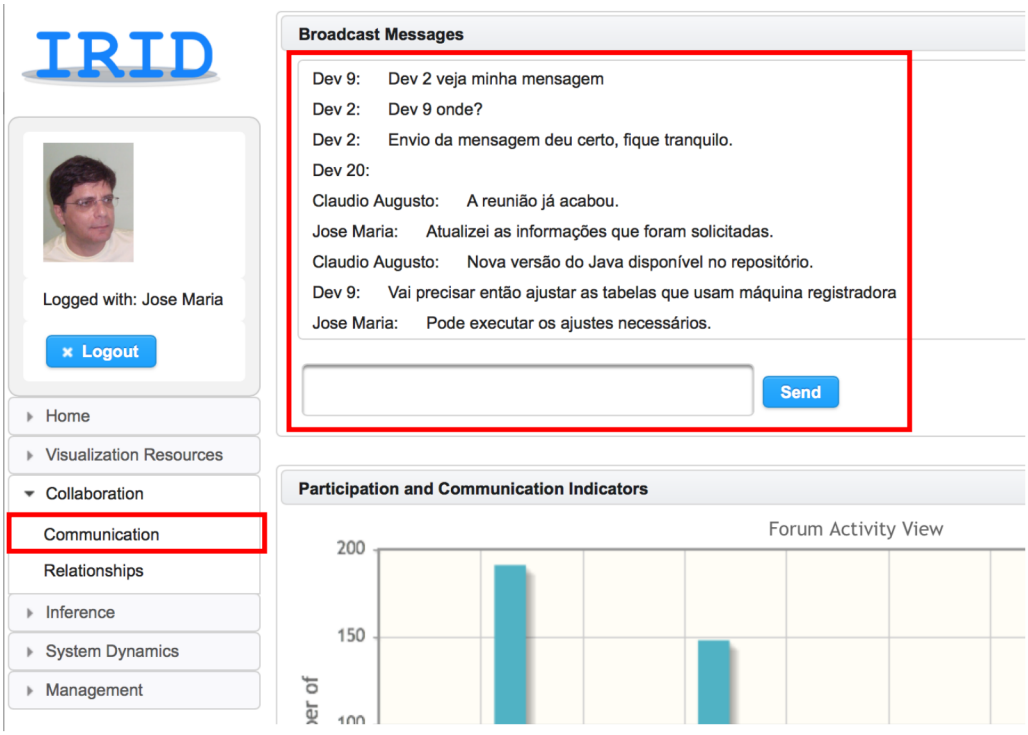

Figura 4.11 – Recurso de envio de mensagens em massa

Os recursos associados à gerência das mensagens privadas são ilustrados na Figura 4.12. Destaca os elementos de percepção que são aplicados para que um desenvolvedor saiba quais outros estão online e disponíveis para colaborar ao mesmo tempo. Ao perceber que determinado desenvolvedor está presente na lista Online seu nome pode ser clicado, abre-se uma tela e a partir disso a troca de mensagens privadas, e instantâneas, é iniciada. A lista Online é oferecida também durante toda utilização das visualizações através dos *Dashboards*, análise das informações apresentadas e tomada de decisão.

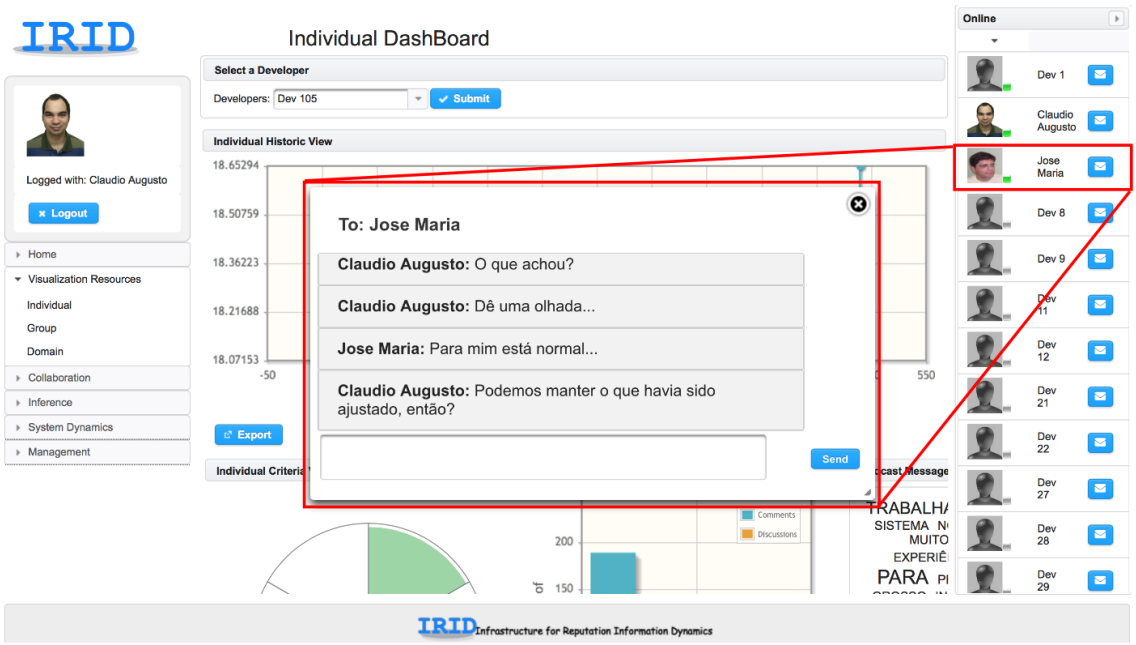

Figura 4.12 – Recurso de envio e recebimento de mensagens entre desenvolvedores

4.2.6. MÓDULO DE VISUALIZAÇÃO

Os recursos visuais oferecidos pelo módulo de Visualização, foram modificados e evoluíram a partir das versões iniciais da arquitetura ArchiRI (LÉLIS et al., 2016b), a ArchiRIXCom (SILVA, 2017) e a abordagem AD-Reputation (LÉLIS et al., 2018).

Vale destacar que o módulo de visualização é acessível por todos desenvolvedores e gerentes. Essa medida é coerente com o requisito para sistemas de reputação que devem auxiliar a disseminação da informação de reputação.

A maioria das visualizações que compõe o módulo de visualização oferecem a possibilidade de representar informações de reputação nos três níveis estabelecidos pelo modelo CoMoRI, apresentado anteriormente, ou seja no nível individual, no nível de Grupo bem como no nível de Domínio. Isso levou à composição das visualizações para formar painéis de controle, os *Dashboards*, associados a cada nível do modelo CoMoRI.

A implementação das visualizações foi realizada em JavaScript e seguiu as práticas de *frameworks* conhecidos, como D3js[18]. A seguir, são detalhadas cada uma das visualizações, bem como, suas utilidades e as informações adicionais incluídas em cada uma delas.

[18] https://d3js.org/

A visualização *Historic View* apresenta a evolução histórica do valor de reputação. Trata-se de um gráfico de linhas no qual é possível ainda indicar a linha média que representa a reputação no período histórico analisado. Por padrão, são analisadas as últimas 500 ocorrências que se tenha registros. Esse valor foi escolhido para que a visualização pudesse mostrar as informações de forma mais clara. A Figura 4.13 ilustra a visualização gerada para apenas um desenvolvedor, por essa razão recebe o prefixo "Individual".

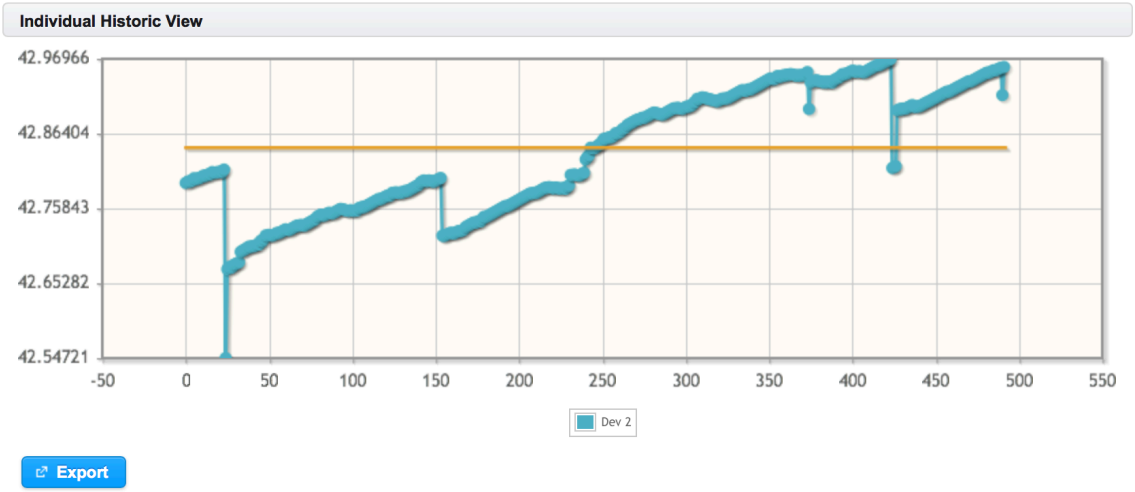

Figura 4.13 – Historic View

Caso seja do interesse do gerente, ou desenvolvedor, que analisa as informações, é possível exportar o gráfico como imagem PNG, GIF ou JPEG através do botão "Export". São oferecidos também, recursos de interação com o usuário por meio dos quais é possível aproximar em um efeito de "zoom" e destacar um período menor de ocorrências. Assim, tem-se a possibilidade de analisar a tendência de evolução da reputação, seus valores absolutos, no período histórico analisado bem como em subconjuntos selecionados diretamente na visualização.

Embora o exemplo da Figura 4.13 destaque a reputação de apenas um desenvolvedor, é possível também apresentar na mesma visualização, a evolução da reputação de um grupo de desenvolvedores. Tal fato é útil quando se deseja analisar as informações de cada desenvolvedor em conjunto, ou com o intuito de comparar seus respectivos históricos.

A Figura 4.14 mostra a visualização *Individual Criteria View* cujo objetivo é apresentar o valor específico de cada critério da reputação do desenvolvedor em análise. Para isso, é considerado o valor de reputação mais atual.

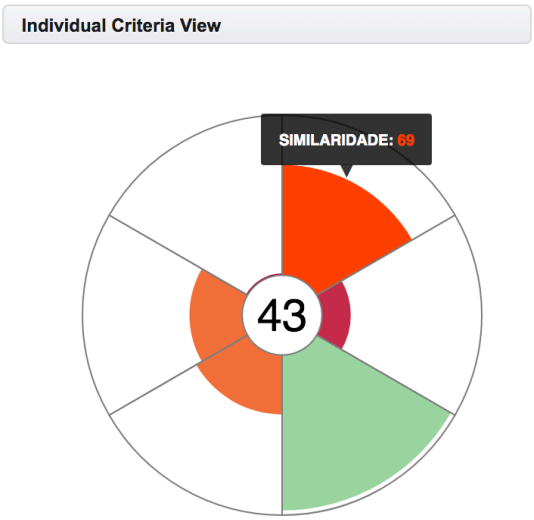

Figura 4.14 – Individual Criteria View

Na *Individual Criteria View* foi utilizada uma metáfora visual semelhante a um gráfico de setores onde o centro exibe o valor final de reputação, e em cada fração do gráfico é alocado um critério. Os efeitos de interação são aplicados ao passar o *mouse* sobre um dos critérios. A fração inteira é destacada em vermelho e o nome do critério e seu valor são exibidos em um *tooltip*. As cores de cada fração também possuem significado, e estão relacionadas aos valores de cada critério em uma escala de cores que inicia no vermelho-escuro, representando intervalo de 0 à 10, até o verde-escuro, indicando valores entre 90 e 100, respectivamente. Outra representação visual dos valores está relacionada ao raio da fração, indicado em um intervalo de 0 a 100. O raio da fração é maior ou menor, dependendo do valor de reputação. Quanto maior é o valor, maior é o raio, e vice-versa. Na Figura 4.14 é feito, como exemplo, um destaque ao critério "Similaridade", que tem um valor de "69".

Particularmente, essa visualização está associada apenas com o nível individual. Para o nível de grupo, a visualização *Compare View* atua de modo semelhante. Os valores de cada critério que compõe a reputação são dispostos em uma metáfora visual de radar circular, semelhante a uma teia de aranha, onde um círculo é dividido em eixos, de comprimento do raio a partir do centro. No entanto, o foco é comparar os valores dos critérios de 2 ou mais desenvolvedores. A Figura 4.15 ilustra a comparação entre 3 desenvolvedores.

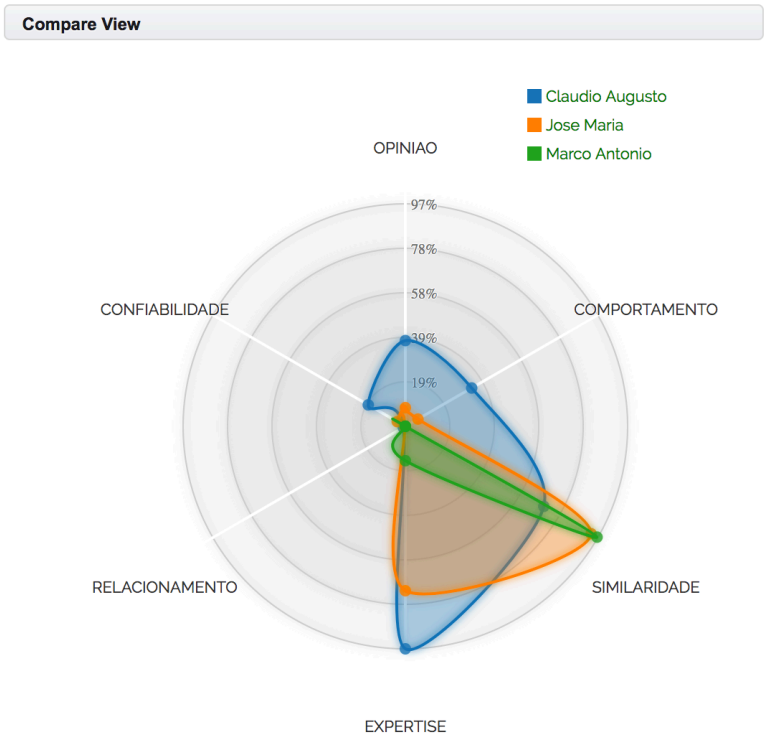

Figura 4.15 – Compare View

O nome de cada critério é mostrado em cada um dos eixos e os valores são pontos que interceptam os eixos. A ligação entre os pontos constitui uma área que indica a reputação de uma entidade a ser comparada. Utilizando os efeitos de interação, ao passar o *mouse* sobre uma das áreas, toda ela é destacada, o que facilita a análise dos valores. Outro efeito pode ser observado ao passar o *mouse* sobre um ponto que intercepta um dos eixos, o valor percentual é realçado.

A Figura 4.16 mostra a visualização *Individual Forum Activity View*, construída como uma união de duas outras visualizações, no entanto considerando as informações apenas de um desenvolvedor. A ideia é mostrar índices de participação do desenvolvedor nos fóruns de discussões, gerenciados pelo módulo de comunicação.

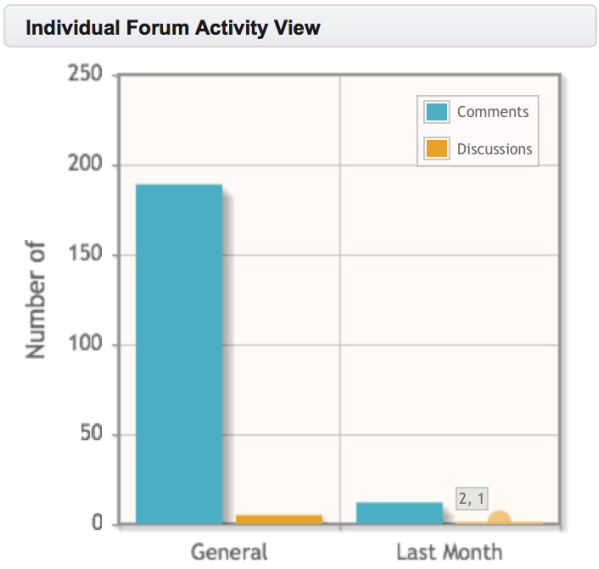

Figura 4.16 - Individual Forum Activity View

A metáfora visual utilizada é um gráfico de barras com a representação de duas séries: comentários e discussões. Assim, a quantidade de comentários feitos ao longo do tempo e a quantidade de discussões nas quais o desenvolvedor contribuiu são mostrados pela inscrição *General*. Do mesmo modo, a quantidade de comentários feitos no último mês e as discussões que receberam contribuição do desenvolvedor são mostrados na inscrição *Last Month*. Através de elementos de interação é possível observar o valor exato das quantidades.

As visualizações que serviram de base para essa simplificação são ilustradas na Figura 4.17 e Figura 4.18. Em ambos os casos a metáfora visual é a mesma, gráfico de barras. A *Forum Activity View* mostra os índices de participação de vários desenvolvedores nos fóruns de discussões. Foi limitada à apresentação de 10 desenvolvedores simultâneos. Para facilitar a visualização e torná-la mais clara é possível filtrar o tipo de informação a ser mostrada, somente a quantidade de comentários ou somente a quantidade de discussões.

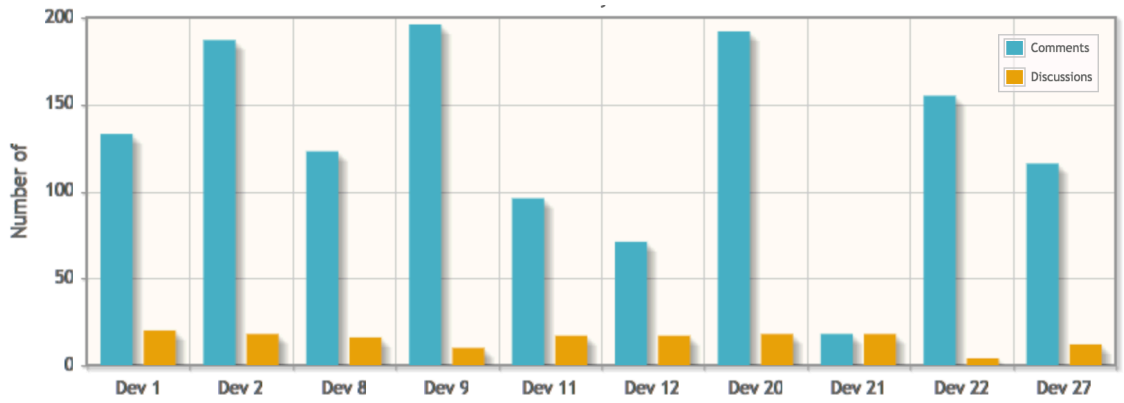

Figura 4.17 - Forum Activity View

De modo semelhante, a *Forum Last Month Activity View* (Figura 4.18) mostra as informações de vários desenvolvedores associadas à participação nos fóruns considerando apenas os últimos 30 dias.

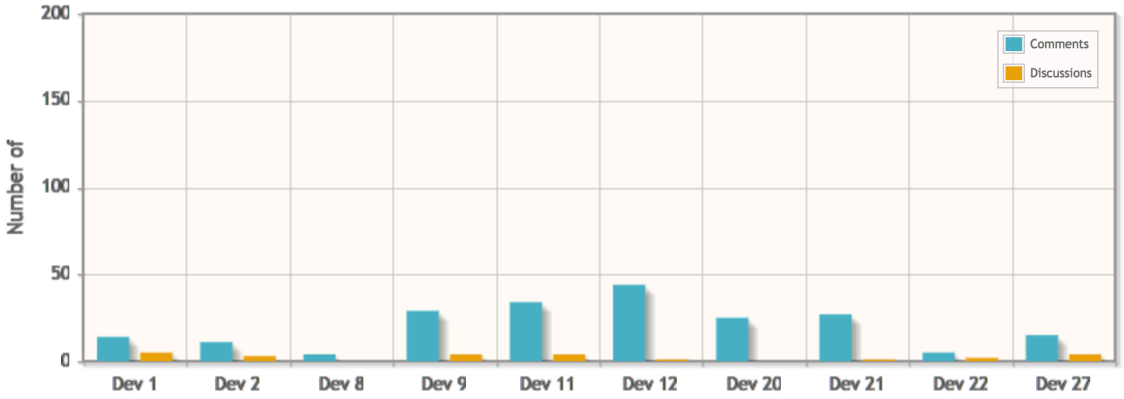

Figura 4.18 – Forum Last Month Activity View

A Figura 4.19 mostra a visualização *Broadcast Message Tag View*, que utiliza a representação de nuvem de *tags* para destacar os termos mais recorrentes nas mensagens. Tais mensagens podem ser os comentários nos fóruns de discussão, ou as mensagens enviadas, gerenciado através do módulo de comunicação.

Figura 4.19 – Broadcast Message Tag View

O processamento das mensagens é feito sob demanda e depende do nível de reputação e a *Dashboard* que está sendo acessada. Por exemplo, para o nível individual, são coletados os dados das mensagens e comentários feitos pelo desenvolvedor alvo. As ocorrências de cada palavra são sumarizadas e ranqueadas. A visualização filtra as 25 palavras que mais aparecem e exibe com tamanhos diferentes, dependendo da quantidade de ocorrências. Assim, o processo é o mesmo para os níveis de Grupo e Domínio: exibe-se as palavras mais recorrentes nas mensagens e comentários feitos pelo grupo em análise ou pelo domínio em análise, respectivamente. As informações mostradas pela *Broadcast Message Tag View* complementam as visualizações sobre a atividade nos fóruns. Enquanto *Forum Activity View* está associada à participação nos fóruns, a *Broadcast Message Tag View* oferece uma noção do conteúdo postado.

Enfim, a Figura 4.20 mostra a visualização *Individual Developer Related View*, baseada em uma representação visual de mapas mentais e grafos. Embora as informações sejam de um desenvolvedor, essa visualização pode representar também informações de um grupo ou de um domínio.

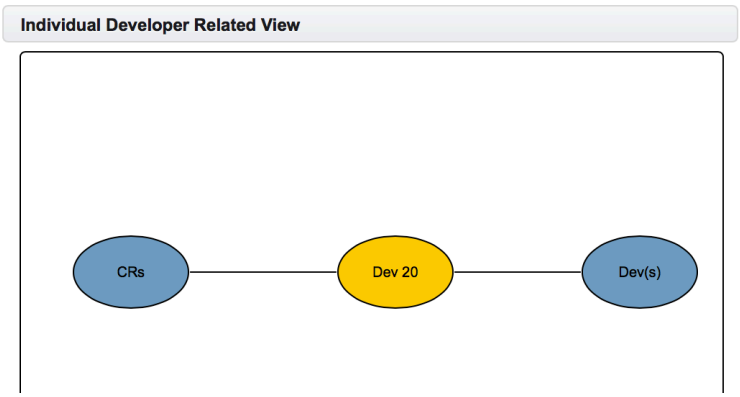

Figura 4.20 - Individual Developer Related View

Através dos elementos de interação, é possível selecionar o nó identificado por CRs (*Change Requests*). A Figura 4.21 ilustra essa situação. Exibindo as solicitações de mudança mais recentemente tratadas pelo desenvolvedor alvo. Vale destacar que o nó raiz é sempre o alvo, seja um desenvolvedor, um grupo, ou a representação de um domínio.

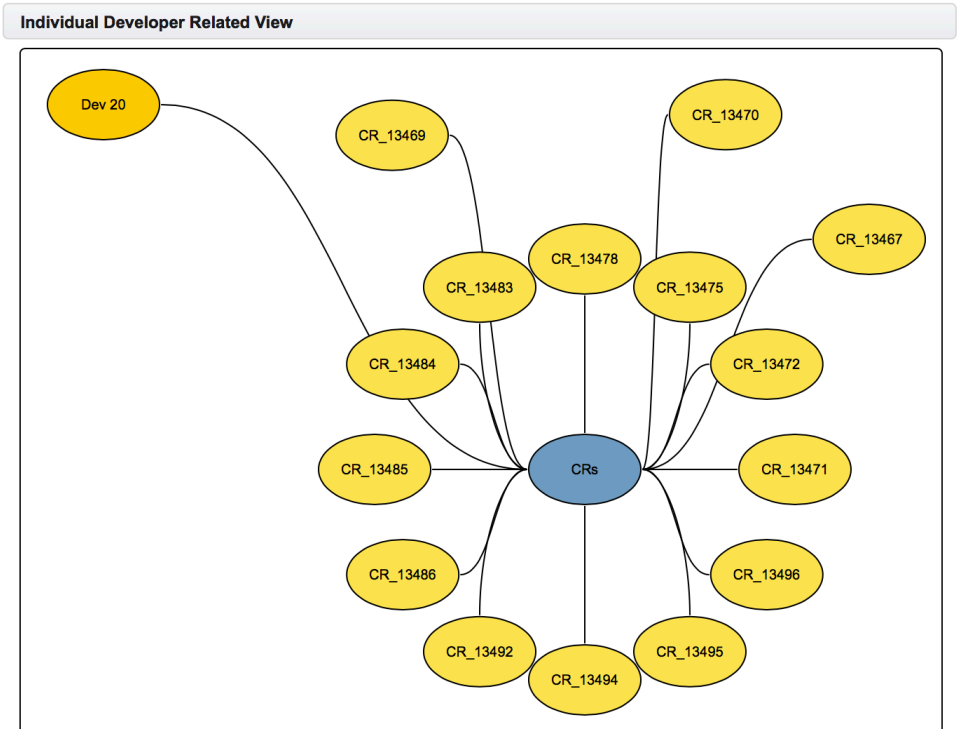

Figura 4.21 – Individual Developer Related View: destaque às solicitações de mudança

Ao retornar ao nó raiz é possível navegar pelo nó (Figura 4.22) com identificador Devs (Developers). Assim, são indicados por nome, os desenvolvedores que colaboraram com o desenvolvedor alvo, nos casos exibidos anteriormente.

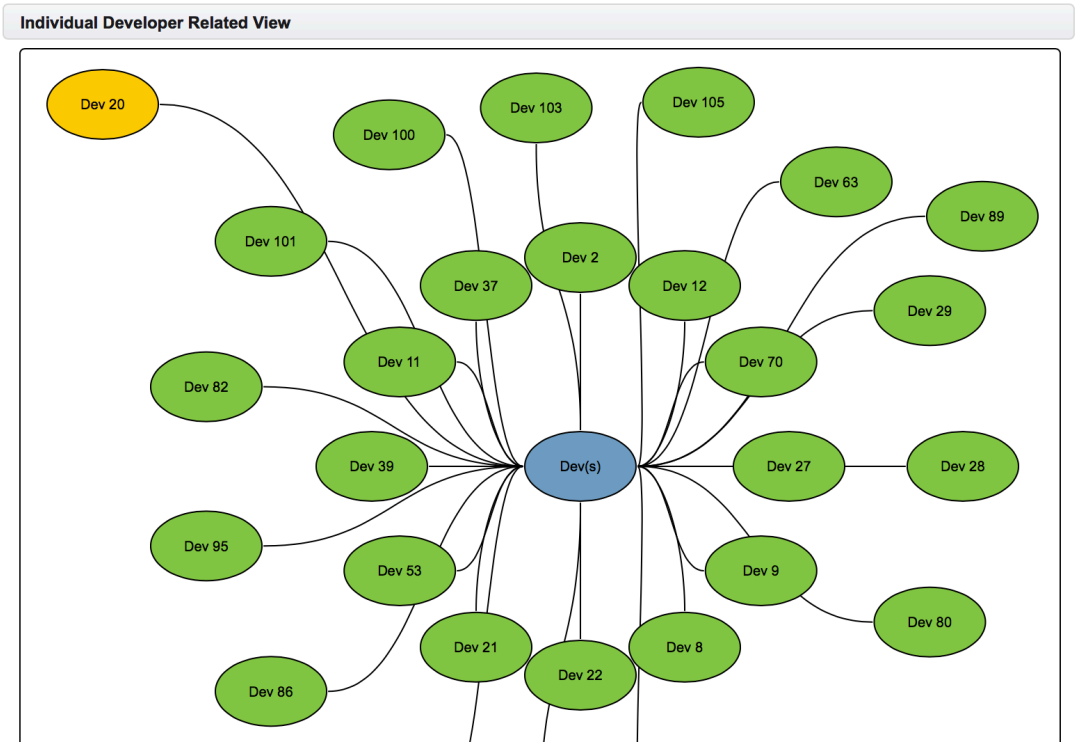

Figura 4.22 - Individual Developer Related View: destaque aos desenvolvedores

A análise conjunta das visualizações que compõem cada uma das *Dashboards*, se complementam e conferem sentido mais consistente às informações mostradas.

4.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Este capítulo apresentou uma infraestrutura para informações de reputação dinâmica, IRID, para o gerenciamento e acompanhamento de informações de reputação de Desenvolvedores em equipes de desenvolvimento, provendo elementos de visualização e colaboração, em um ambiente integrado às atividades de manutenção de software.

Para tanto, foi utilizado como base o modelo de reputação apresentado no capítulo anterior, associado a um módulo que processa dados transformando-os em informações. A ideia é que o modelo possa ser instanciado, de acordo com os dados disponibilizados como fonte e através do cálculo dos critérios de reputação, por métricas específicas a cada um deles. IRID é composta de um sistema de reputação, agrega elementos visuais e de integração, em um AIMV, facilitando a obtenção de conhecimento a partir das informações. Através de elementos de percepção, a colaboração é alcançada através dos diferentes recursos de comunicação oferecidos pela

infraestrutura (mensagens em massa, mensagens privadas e fórum de discussões). Dessa forma, auxilia na tomada de decisão conjunta em relação à alocação dos desenvolvedores nas tarefas de manutenção.

Conceitos de Dinâmica de Sistemas foram mostrados por serem específicos ao entendimento do processo de simulação e observação da reputação, em evolução. Foi apresentada a integração com a Ferramenta Insight Maker, através da qual é possível executar as técnicas de simulação provenientes da Dinâmica de Sistemas, sendo apresentado o comportamento dos critérios de forma a observar e predizer a tendência de crescimento ou decrescimento da reputação, considerando que os critérios de reputação não são independentes, mas influenciam uns nos outros.

O próximo capítulo mostra a avaliação conduzida da infraestrutura IRID, destacando os módulos acionados durante cada processo de avaliação.

5. AVALIAÇÃO DA INFRAESTRUTURA DE REPUTAÇÃO

Com o intuito de avaliar a infraestrutura IRID, foi conduzida inicialmente uma prova de conceito e, em seguida, um experimento. A prova de conceito teve como objetivo avaliar os recursos oferecidos pela infraestrutura, com relação às visualizações e utilização geral do sistema. O experimento contemplou avaliar a viabilidade do modelo de reputação e, através da coleta dos dados simulados, traçar um paralelo entre as escolhas feitas pelo gerente ao alocar os desenvolvedores, e aqueles com os maiores índices de reputação.

Os dados utilizados em todo processo de avaliação foram obtidos como resultado de uma parceria estabelecida com uma empresa de desenvolvimento de software para gerenciamento de negócios. Por razões de confidencialidade, a empresa será nomeada como empresa parceira. Utiliza atualmente a metodologia de desenvolvimento SCRUM (SMITH, 2016), que se trata de um *framework* de desenvolvimento iterativo e incremental, utilizado no gerenciamento de projetos e desenvolvimento de software ágil. A empresa possui um total de 40 desenvolvedores ativos, distribuídos geograficamente em regime de *home office* em localidades distintas à sede da empresa, inclusive fora do Brasil.

Nesse contexto real de utilização, a base de dados de solicitações de mudança registrados no sistema gerenciador MantisBT, disponibilizado pela empresa parceira, compreende informações dos últimos 5 anos. Os dados referentes às solicitações de mudança, o histórico de dados atualizados à medida que a requisição era tratada, bem como os dados registrados dos desenvolvedores foram considerados para a geração das informações de reputação.

Para a condução da prova de conceito e do experimento foi necessária a distribuição dos dados em conjuntos. A Figura 5.1 ilustra a caracterização dos dados utilizados.

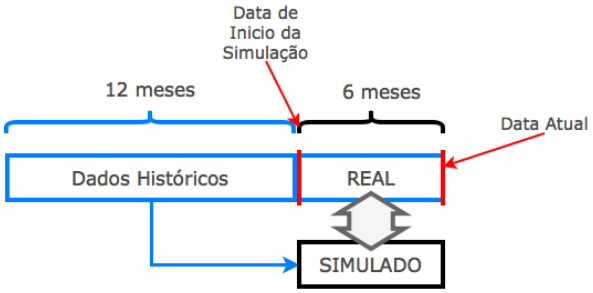

Figura 5.1 – Caracterização dos dados para as avaliações

Os seis meses anteriores a partir da data atual, caracterizam o conjunto de dados chamado Real, os doze meses anteriores à data de início do Real caracterizam os Dados Históricos e assim foram chamados. Os Dados Históricos foram utilizados para gerar as equações necessárias para instanciar o modelo dinâmico de reputação necessário para executar a simulação, com isso foi possível gerar seis meses de dados simulados os quais caracterizaram o conjunto chamado Simulado, para que fosse possível comparar o Real e o Simulado.

5.1 AVALIAÇÃO DOS RECURSOS DE VISUALIZAÇÃO E UTILIZAÇÃO DO SISTEMA

Esta avaliação foi realizada como uma prova de conceito com o objetivo de verificar a viabilidade de uso dos recursos de visualização e comunicação. Foi conduzida com o intuito, também, de verificar a aplicabilidade desses recursos, no processo de alocação de desenvolvedores às tarefas de manutenção.

A prova de conceito contou com a participação de um gerente de projetos, para reduzir o viés de observação e, em seguida, das análises realizadas. Trata-se de um gerente que não possui contato com o repositório de solicitações de mudança ou com qualquer conhecimento das técnicas internas utilizadas pela empresa parceira. Esse desconhecimento mostra a possibilidade de a abordagem auxiliar gerentes e equipes, que seguem processos de desenvolvimento e manutenção distintos da empresa parceira.

Inicialmente o gerente foi submetido a uma rápida apresentação, na qual foram explicados o objetivo da infraestrutura e seus requisitos funcionais e não funcionais. Além disso, foi estabelecido um cenário para guiar a condução da prova de conceito.

Utilizou-se como cenário uma ocorrência de manutenção em um módulo importante para um cliente chave. Assim, a equipe do gerente de projeto recebe uma demanda de mudança, em um produto de software que exige alta confiança dos

envolvidos e conhecimento das regras de negócio. O gerente decide analisar as possibilidades de alocação imediata, seja via equipe de desenvolvedores ou individualmente. No entanto, o gerente desconhece quais desenvolvedores estariam disponíveis e aptos a serem alocados.

O primeiro passo é a fase de Login, mostrada na Figura 5.2, na qual o gerente seleciona o ícone de usuário anônimo, destacado em "A", e preenche seus dados de usuário e senha e solicita acesso pelo botão destacado em "B".

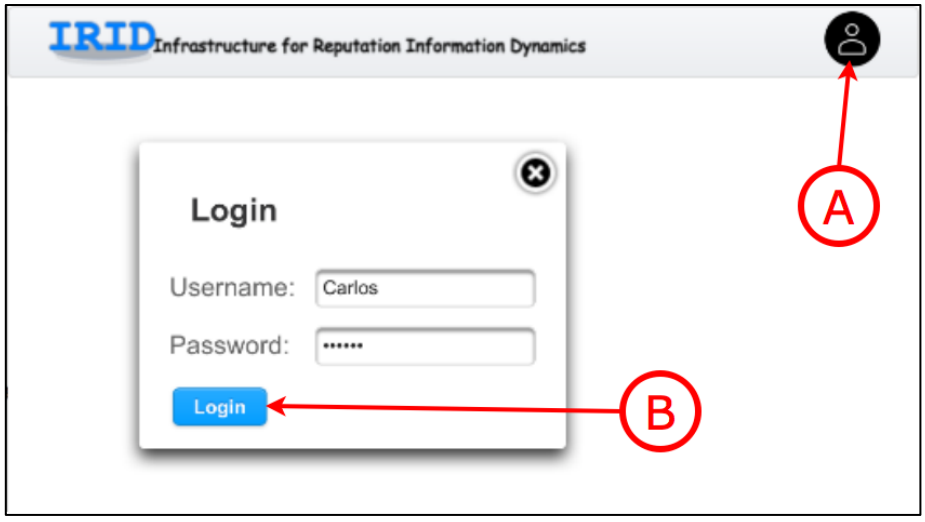

Figura 5.2 – Fase de Login no Sistema

Ao finalizar a fase de acesso, o gerente é direcionado para a tela de recepção do sistema, ilustrada na Figura 5.3, na qual pode verificar seu status de "logado" com sucesso e observar sua foto, cadastrada na página de configurações, o que permite facilitar a identificação do gerente pelos demais usuários do sistema, ambos destacados em "A". A Figura 5.3 mostra ainda, em "B", as notificações com as novidades ocorridas desde a última visita do gerente ao sistema. Em "C", são mostrados os casos reabertos no período e, em "B", os casos novos. Em ambas situações é possível obter mais detalhes através do botão. O gerente assim, deseja saber os detalhes do novo caso "12456".

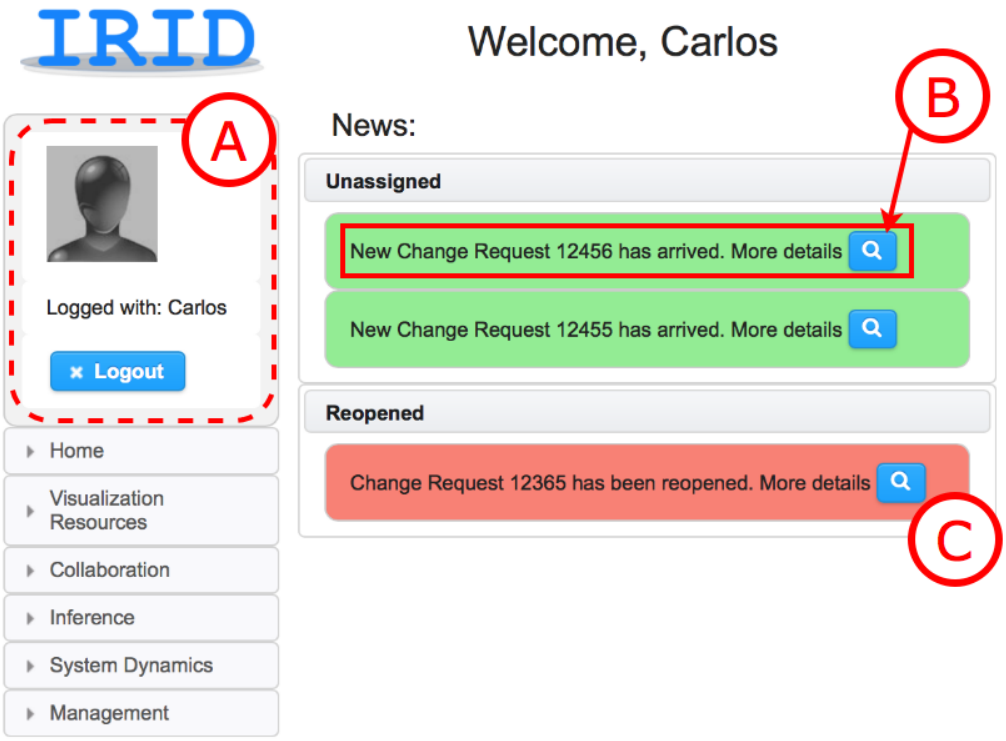

Figura 5.3 – Fase de Recepção e perceber as notificações de novidades

A Figura 5.4 mostra os detalhes do caso, conforme solicitado pelo gerente. Dessa forma, em "A", o gerente confere se os dados mostrados são realmente do caso solicitado. Em "B", o estado do caso, de fato recém-cadastrado, e ainda como "novo" para o sistema, bem como a data que foi inserido no sistema. Por essa razão, em "C", o gerente percebe que o caso permanece sem um responsável. Destaca-se também o botão através do qual o gerente seleciona o desenvolvedor ou os desenvolvedores, em equipe, para tratar o caso em análise. Em "D", observa-se através do símbolo em verde no *chat*, quais desenvolvedores estão *online*. Isso pode guiar a análise das visualizações oferecidas na *Dashboard* Individual, na medida em que o gerente escolhe quem está *online* e disponível para conversar sobre dúvidas, bem como ser alocado imediatamente.

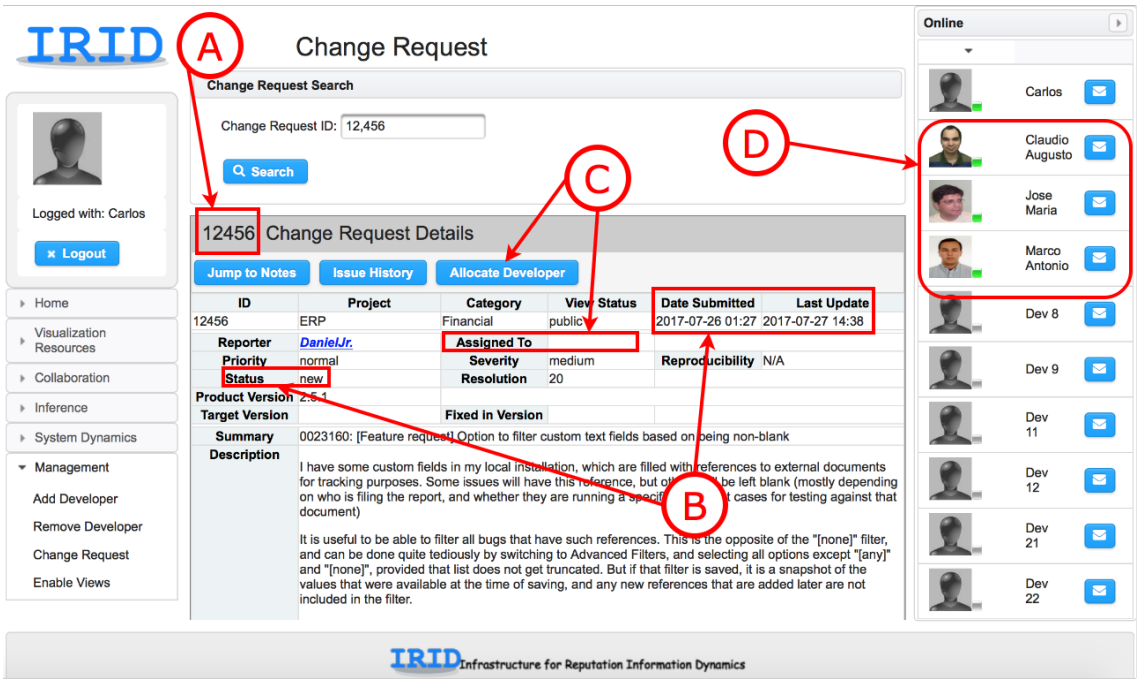

Figura 5.4 – Detalhes do caso de solicitação de mudança e percepção dos desenvolvedores online

Em seguida, o Gerente analisa as visualizações da *Dashboard* Individual para o desenvolvedor "Claudio", assim igualmente, analisa para o desenvolvedor "Jose" e assim por diante para os desenvolvedores *online*. Não há uma ordem de prioridade ou recomendação com base na qual o gerente deveria iniciar a análise. Na Figura 5.5, essa análise é ilustrada para o Desenvolvedor "José". Através das visualizações adquire-se uma visão histórica da evolução da reputação do desenvolvedor, destacado em "A", bem como a pontuação de cada critério que contribui para o valor atual de reputação, destacado em "B". Além disso, em "C", a visão geral da utilização dos aspectos de comunicação, bem como do conteúdo postado, destacado em "D". Enfim, em "E", o gerente pode, através dos elementos de interação, analisar quais casos anteriores o desenvolvedor já tratou e com quais desenvolvedores já colaborou.

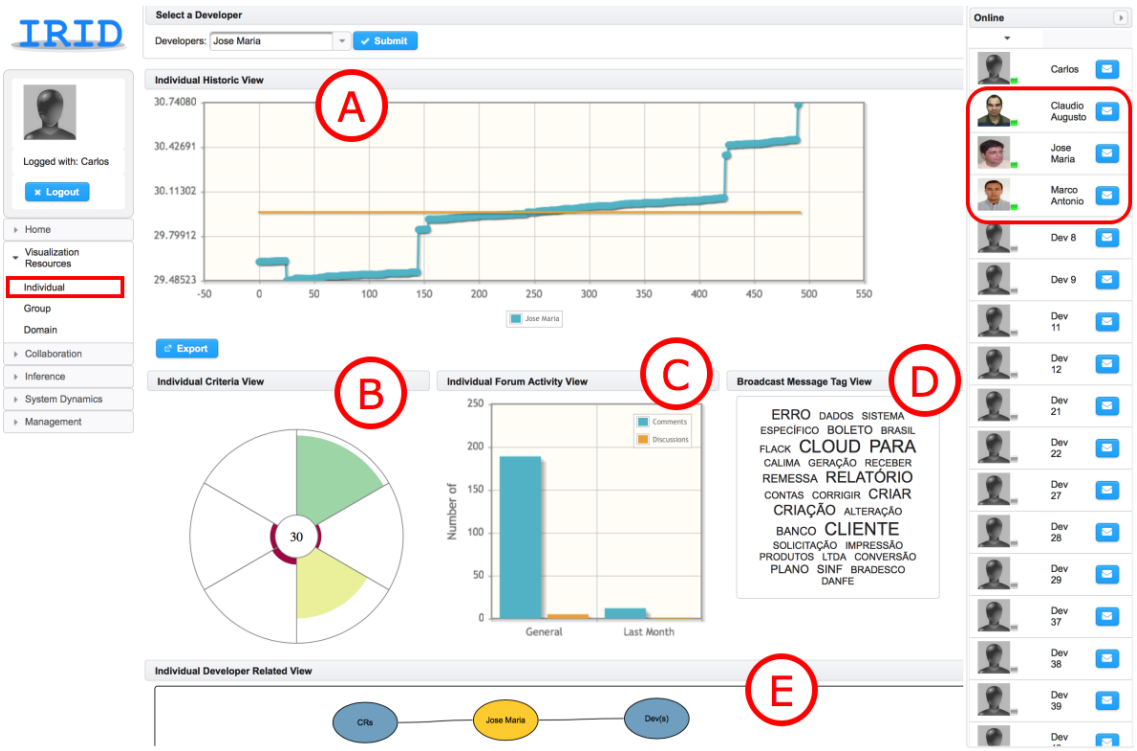

Figura 5.5 – Análise da reputação através da Dashboard Individual

De posse das informações individuais do desenvolvedor, o gerente pôde analisar o desenvolvedor sob o ponto de vista do último grupo ao qual fez parte. Neste ponto, para a descoberta de qual foi o ultimo grupo que o desenvolvedor participou são consideradas as inferências feitas pela ontologia. Assim, o gerente aciona as visualizações e inicia a análise através da *Dashboard* de Grupo. Embora as visualizações apresentem informações relativas ao grupo, a integração entre as *Dashboards* permite que se analise informações de um mesmo desenvolvedor nos 3 níveis estabelecidos pelo modelo CoMoRI, bastando navegar entre as páginas. Dessa forma para carregar as visualizações basta indicar o desenvolvedor.

Através da análise da *Dashboard* de Grupo, ilustrada na Figura 5.6, inicialmente o gerente obtêm a visão histórica da evolução da reputação do grupo, discriminado em "A" por cada desenvolvedor. Além disso, em "B", a comparação entre a pontuação dos critérios, de cada desenvolvedor pertencente ao grupo, que contribuem para o valor atual de reputação. Em "D", é exibida uma visão geral da participação nas trocas de mensagens, bem como, em "E", do conteúdo postado por cada membro do grupo. Do mesmo modo, em "F", é possível analisar quais casos anteriores o grupo tratou como equipe, e com quais desenvolvedores colaborou anteriormente. Através das legendas

destacadas em “C” e os recursos de interação das visualizações é possível, ao gerente, perceber qual o último grupo que o desenvolvedor escolhido fez parte.

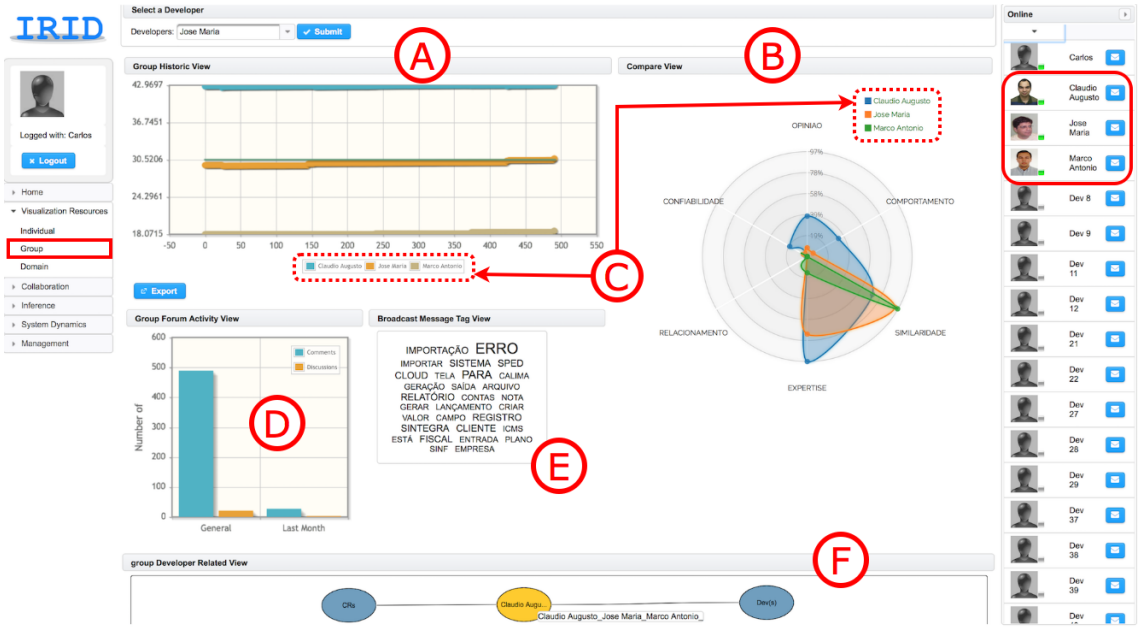

Figura 5.6 – Análise da Dashboard de Grupo

Neste ponto vale destacar que os passos anteriores, de análise das visualizações, podem ocorrer de forma iterativa e interativa. O gerente analisa as informações de um desenvolvedor, em seguida, descobre a que grupo pertence e as informações pertinentes a esse grupo, na sequência retorna à *Dashboard* Individual e pode analisar outro Desenvolvedor do mesmo grupo. Esse processo também é interativo pois, o gerente pode utilizar os recursos de colaboração para perceber outros gerentes e desenvolvedores que estejam *online* e capazes de tomar decisão conjunta.

A partir das informações individuais dos desenvolvedores online, durante as análises, e dos grupos em que fizeram parte, o gerente executou uma simulação. Objetivando descobrir a evolução dos comportamentos da reputação de cada desenvolvedor envolvido, se tratarem esse caso de solicitação como uma equipe. O módulo de Previsão e Simulação foi acionado e o arquivo gerado, foi importado para a ferramenta InsighMaker, conforme o fluxo apresentado no experimento na próxima seção.

Através da simulação, iniciada a partir do botão destacado em “A” na Figura 5.7, foi possível projetar o comportamento da reputação, em ocorrências futuras, e selecionar, em “B”, apenas os desenvolvedores de interesse do gerente, envolvendo os

Desenvolvedores “Claudio” e “Marco”, membros do último grupo formado pelo desenvolvedor “Jose”.

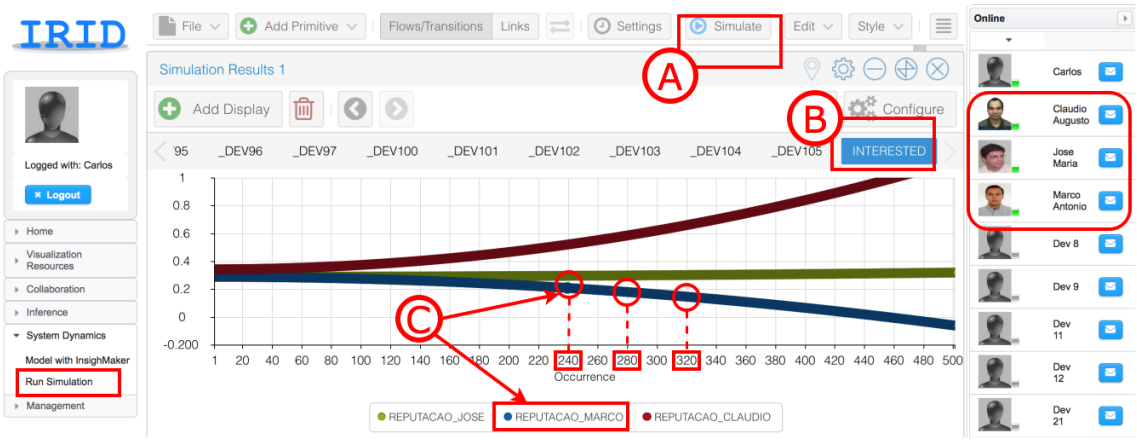

Figura 5.7 – Análise após rodar a simulação

Considerando que o caso fosse tratado pelos três desenvolvedores em equipe, as informações de reputação projetadas em ocorrências futuras, levou o gerente a observar a tendência de decrescimento, destacado pelas marcações em “C”, na reputação do desenvolvedor “Marco”. Considerando que a empresa receba 15 solicitações de mudança em média por dia, e a tendência de decrescimento se agrava a partir da ocorrência 240, isso significa que em 3 semanas esse agravamento será notado. Assim, o gerente interpretou que, para este caso específico, seja melhor não alocar o desenvolvedor “Marco”. Decide depois enviá-lo a um treinamento, e enquanto isso, associá-lo a outro grupo para um caso mais simples.

A Figura 5.8 ilustra a página de detalhes do caso de solicitação “12456”, acionada pelo gerente, para enfim fazer a alocação. Através do acionamento do botão “*Allocate Developer*”, destacado em “A”, uma janela foi aberta com o ranqueamento dos desenvolvedores e os valores de reputação. As linhas que indicavam um desenvolvedor escolhido, foram selecionadas, como é mostrado em “B”. A partir do botão “*PreView*”, em “C”, o gerente pôde observar os nomes, e fotos, de cada desenvolvedor selecionado e que formam a equipe a tratar o caso de solicitação. O gerente pôde confirmar sua escolha, mostrado em “D”, sendo registrada através do Mantis, no repositório de solicitações.

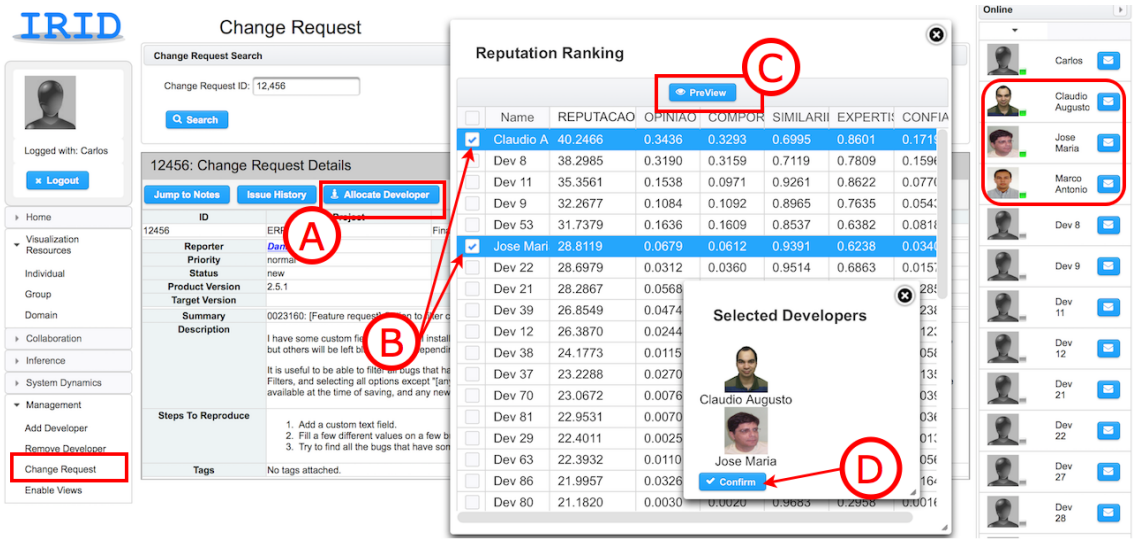

Figura 5.8 – Alocação dos desenvolvedores como equipe

Considerando que a empresa parceira trabalha com seus desenvolvedores distribuídos geograficamente, nesse sentido, os aspectos de colaboração estão presentes nos processos da empresa. Diante disso, esta prova de conceito mostrou como os aspectos de colaboração são abordados pela infraestrutura IRID. As informações do usuário que está acessando o sistema, bem como a indicação de desenvolvedores e gerentes *online* representam elementos de percepção. A possibilidade de trocar mensagens instantâneas ou enviar mensagens mesmo que o usuário não esteja online está associado ao elemento de comunicação. As *dashboards* com suas respectivas visualizações representam o ambiente de trabalho compartilhado associado ao elemento de cooperação. E por fim, a possibilidade de o gerente alocar desenvolvedores ou trocar os responsáveis por uma solicitação se refere à características associadas ao elemento de coordenação.

Além disso, esta prova de conceito mostrou como gestores, e desenvolvedores, podem analisar as informações de reputação, uns dos outros, independente do grupo ou domínio que façam parte. Por fim, como técnicas de visualização diferentes podem ser combinadas a elementos de colaboração para melhorar a análise e tomada de decisão.

5.2 AVALIAÇÃO DO MODELO DE REPUTAÇÃO

A fim de verificar a viabilidade de uso do modelo de reputação, bem como a sua aplicabilidade, foi realizada a simulação do modelo através da Ferramenta

InsightMaker, integrada à Infraestrutura IRID conforme apresentado no Capítulo anterior.

Foram considerados no processo de simulação 40 desenvolvedores da empresa parceira. Todos têm a possibilidade de serem escolhidos e alocados para tratar solicitações de mudança, sendo essa alocação realizada com base nas informações geradas pela aplicação do modelo.

O modelo proposto neste trabalho foi representado através de um Diagrama de Causas e Efeitos. Esse tipo de diagrama é considerado um precursor para a construção de um Diagrama de Repositórios e Fluxos, que exige o conhecimento e manipulação de valores absolutos das variáveis envolvidas. Essa argumentação leva à definição do tipo de análise estatística de dados que será utilizada. Segundo Camiletti e Ferracioli (2002), pode-se classificar os processos de análise de dados em três dimensões:

Análise Quantitativa: envolve uma variedade dos aspectos do reconhecimento de simples relacionamentos numéricos tais como comparar conjunto de números até a manipulação de relacionamentos algébricos. Essa dimensão envolve a compreensão de como uma mudança de uma variável afetará outra em um sistema específico.

Análise Qualitativa: envolve fazer distinções entre categorias e tomadas de decisões. Isso pode consistir em examinar um conjunto de escolhas e tomar uma decisão baseada na consideração de suas consequências, isto é, essa perspectiva exige a observação e a consideração de alternativas e da análise cuidadosa das evidências.

Análise Semiquantitativa: envolve a descrição de situações em que o sentido de uma mudança em uma parte de um sistema é conhecido, mas não o tamanho do efeito dessa mudança em outras partes. A análise desses efeitos pede a compreensão do sentido do relacionamento causal (aumento ou diminuição), mas não do conhecimento dos valores numéricos. A base para analisar essa dimensão reside no fato de que tanto a análise quantitativa quanto a qualitativa não capturam todos os aspectos importantes de um sistema. Consequentemente, a construção de modelos de uma maneira semiquantitativa pode ser baseada em uma visão de pensamento sistêmico, demandando que a compreensão do comportamento de um sistema é baseada nos relacionamentos causais entre as variáveis que o descrevem. Nesse sentido, a causalidade desempenha um papel fundamental na modelagem semiquantitativa, porque é importante no estabelecimento dos relacionamentos entre as variáveis de um modelo.

Dessa forma, os relacionamentos entre variáveis que descrevem um sistema podem ser entendidos e representados através de uma representação Causa-Efeito, quando considerando uma análise semiquantitativa, e simulados através de modelos de Dinâmica de Sistemas. Essa abordagem será mostrada, com a utilização do modelo proposto neste trabalho, a partir da infraestrutura que envolve o uso da Ferramenta InsightMaker.

O processo adotado na avaliação acompanha a sequência definida na Figura 5.9. Busca apresentar os recursos da infraestrutura envolvidos nesta avaliação, sendo eles: o módulo PDR, os recursos do módulo de Previsão e Simulação e a integração com a ferramenta Insight Maker.

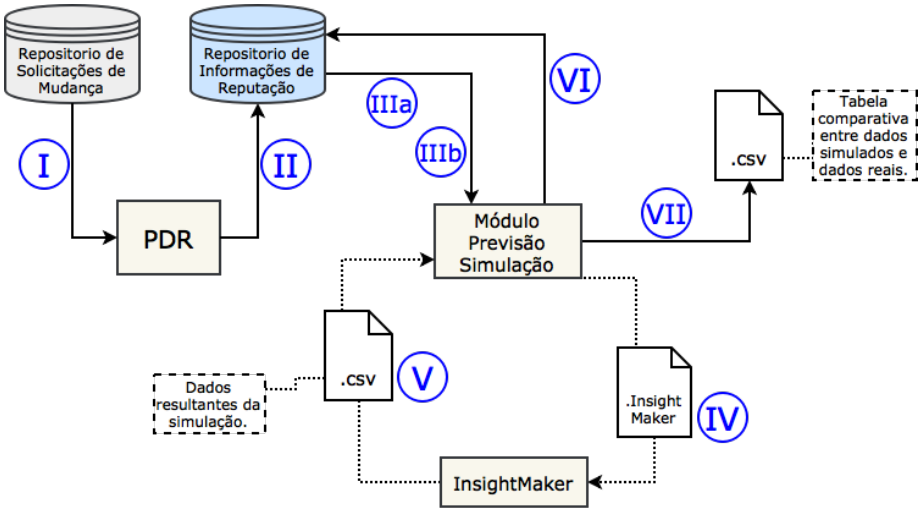

Figura 5.9 – Fluxo das informações no experimento

O processo inicia com o módulo de PDR (Processamento dos Dados para Reputação) que captura os dados do repositório de solicitação de mudança (I) e associa esses dados às métricas para o cálculo dos critérios. Os dados das requisições de mudança utilizados na geração das informações de reputação compreenderam um repositório de 5 anos. No entanto, para a avaliação, foram divididos em dois conjuntos de dados. O primeiro conjunto compreende os últimos 6 meses dos dados registrados, e o outro conjunto compreende o ano anterior a esses meses. Tal escolha será justificada mais a diante.

Para calcular cada critério foram coletadas medidas baseadas nas métricas definidas no Capítulo 3. Nos casos de critérios com mais de 1 conjunto de métricas a ser coletada, o conjunto foi definido com base nos dados disponibilizados pela empresa

parceira. Assim, o modo que cada critério foi calculado e as métricas coletadas é descrito a seguir.

O critério Opinião foi medido pelo número de requisições tratadas pelo desenvolvedor alvo, lembrando que esse é o desenvolvedor sobre o qual se deseja analisar a reputação. A cada requisição tratada é atribuído o valor de uma avaliação positiva (igual a 1). Essa abordagem foi apresentada em (LÉLIS et al., 2018).

O critério Comportamento foi medido através da média aritmética de valores históricos do critério Opinião do desenvolvedor alvo, baseado na abordagem apresentada em (CAVERLEE et al., 2010). Esta escolha objetiva registrar as mudanças bruscas de Comportamento. No entanto, apenas a partir da reincidência é que a mudança de comportamento passa a impactar efetivamente no cálculo.

O critério de Similaridade seguiu um algoritmo baseado em (QU et al., 2006) e foi medido inicialmente entre desenvolvedores, todas as combinações sem repetição tomados de 2 em 2, levando em consideração o somatório das medidas históricas de Comportamento de cada desenvolvedor e a quantidade dessas medidas. Como resultado, um desenvolvedor possui um valor de Similaridade diferente em relação a cada outro desenvolvedor da empresa parceira. Neste ponto do processo foi realizada a agregação dos valores de Similaridade e calculada a média. Esse procedimento foi necessário para que cada desenvolvedor permanecesse com apenas um valor de Similaridade entre 0 e 1 que representa o quanto o comportamento do desenvolvedor alvo se aproxima dos demais.

O critério Relacionamento foi medido pelo conjunto de métricas que abrange o valor do critério Opinião do desenvolvedor alvo, o somatório dos valores do critério Opinião dos desenvolvedores ligados, ponderado pela quantidade de desenvolvedores ligados ao alvo. Essa abordagem foi seguida por Caverlee et al. (2010).

O critério Expertise foi medido através de um algoritmo baseado no Expertise Rank (ZHANG et al., 2007), e utiliza o valor histórico do critério Expertise dos desenvolvedores ligados ao alvo e a quantidade de ligações que esses desenvolvedores possuem. A influencia do valor de Expertise dos desenvolvedores ligados é ponderada através do fator multiplicativo D, que recebe valor de 0.85, conforme apresentado em (ZHANG et al., 2007).

E, finalmente, o critério Confiabilidade foi medido através de um algoritmo (CAVERLEE et al., 2010) que utiliza os somatórios dos Valores do critério

Confiabilidade e do critério Relacionamento, dos desenvolvedores ligados. Esses somatórios são ponderados pela quantidade de ligações que os desenvolvedores ligados possuem. Além disso, é considerado também o valor do critério Opinião do desenvolvedor alvo. O fator multiplicativo lambda (λ), conforme apresentado por (CAVERLEE et al., 2010), recebe valor de 0.5.

Essa definição de cada critério é fundamental para que o módulo PDR possa transformar dados de solicitação de mudança em informações de reputação. Posteriormente, foram armazenadas no repositório de informações de reputação, compreendendo o passo II do processo.

No entanto, não se deve ficar restrito a uma análise pontual da reputação dos desenvolvedores, sendo necessário verificar o seu comportamento futuro, em função da sua própria evolução, através de um processo de simulação utilizando Dinâmica de Sistemas.

Essas informações podem ser utilizadas para gerar as equações utilizadas no processo de simulação através de regressão linear. Justifica-se o uso de regressão linear, apesar da possibilidade de uma maior margem de erro, em função de as observações dessa natureza, normalmente, serem realizadas através de análise semiquantitativa para a análise de dados, em que a tendência é o foco, mais do que a precisão dos valores reais.

Assim, a partir da definição da data de início e o intervalo em meses para coleta das informações históricas de reputação, o módulo de Previsão e Simulação coletou as informações dos 40 desenvolvedores em análise (etapa III do processo), e gerou o arquivo de extensão "*.InsightMaker*" com as equações e valores iniciais de cada estoque dos critérios (etapa IV do processo) dos respectivos desenvolvedores. Dessa forma, cada um dos 40 desenvolvedores estava associado a um estoque para cada critério e mais um estoque para a reputação, totalizando 240 equações e 280 estoques no modelo de reputação construído para essa avaliação, a partir dos dados da empresa parceira.

O conjunto de equações geradas pôde ser importado e computado através da ferramenta Insight Maker. No processo de simulação, foram considerados 1200 passos de simulação que significam as ocorrências de solicitação de mudanças futuras a serem simuladas. Esta quantidade de ocorrências corresponde aos últimos 6 meses de dados reais cadastrados, separados para a comparação com os dados simulados.

Após a simulação, os dados foram exportados em formato *.csv* (etapa V do processo). O módulo de Previsão e Simulação foi novamente acionado, dessa vez para armazenar os dados resultantes da simulação no repositório de informações de reputação (etapa VI do processo).

A etapa VII do processo representa o momento em que o módulo de Previsão e Simulação coleta as informações simuladas e as informações reais para o mesmo período da simulação. As tendências de crescimento e decrescimento foram calculadas através de regressão linear para ambos os tipos de informação, considerando os 40 desenvolvedores envolvidos. Como resultado, foi gerada e exportada a tabela comparativa, na etapa VIII do processo. Em seguida, o processo foi finalizado.

A Tabela 5.1 apresenta as tendências, real e simulada, para a reputação dos desenvolvedores envolvidos no processo de avaliação. Para tornar a análise das informações mais clara, foram destacados os casos em que a tendência simulada se comportou de modo diferente conforme a tendência real dos dados. Por questões de confidencialidade, objetivando preservar a identidade de cada um dos desenvolvedores, foram associados um número e uma identificação aleatória.

Tabela 5.1 - Tendências para a reputação dos desenvolvedores

Desenvolvedor	Real	Simulado
1	↓	↑
2	↑	↑
8	↓	↑
9	↑	↓
11	↑	↑
12	↑	↑
20	↑	↑
21	↑	↑
22	↓	↓
27	↓	↑
28	↑	↑
29	↑	↑

Desenvolvedor	Real	Simulado
71	↓	↓
75	↑	↑
80	↑	↑
81	↑	↑
82	↑	↑
86	↑	↑
88	↑	↑
89	↓	↑
92	↑	↑
93	↑	↑
94	↑	↑
95	↑	↑

37	↑	↑
38	↓	↓
39	↑	↑
49	↓	↑
53	↓	↑
57	↓	↓
63	↑	↑
70	↑	↑

96	↓	*
97	↑	*
100	↑	*
101	↑	*
102	↑	*
103	↑	*
104	↓	*
105	↑	*

Os casos dos desenvolvedores destacados na Tabela 5.1 foram analisados individualmente. Através da informação da data de criação do desenvolvedor no sistema Mantis, foi observado que os desenvolvedores 96, 97, 100, 101, 102, 103, 104 e 105 eram novatos na empresa. A inserção de uma nova entidade em um sistema de reputação caracteriza o problema da partida fria, comentado no Capítulo 2. Vale lembrar que existem formas de abordar o problema. Uma delas é atribuir valor zero aos recém-chegados, outra forma é aguardar um número padrão de ocorrências acontecerem e somente a partir disso iniciar o cálculo da reputação. Essas abordagens fazem o novato aguardar muito tempo até que se acumule dados históricos para que sua reputação comece a ser calculada. A outra abordagem é um valor médio entre as entidades ativas do sistema. Essa abordagem pode fazer com que o novato receba uma pontuação maior se comparado a outras que já estão ativas no sistema há mais tempo.

O modelo de reputação proposto oferece suporte à partida fria, conforme comentado no capítulo 3, sendo baseado no critério Expertise, através da variável D. Assim, é possível mostrar que mesmo que não hajam dados históricos a respeito de um desenvolvedor, sua partida fria na infraestrutura é um valor de reputação baixo, para não ser injusto com aqueles que possuem um histórico positivo, mas diferente de zero, de forma que o desenvolvedor novato não seja penalizado. Assim, é considerado pelo modelo como um desenvolvedor isolado, sua tendência de reputação permanece constante o que explica o sinal de asterisco (*) na Tabela 5.1, no entanto pode ser contabilizado, aparecer no módulo de ranqueamento e alocado a uma solicitação de mudança.

Dessa forma, podem ser consideradas válidas as análises de tendência simulada para a reputação dos Desenvolvedores 96, 97, 100, 101, 102, 103, 104 e 105 já que refletem o que se esperava da infraestrutura, nos casos de desenvolvedores recém-admitidos e, por isso, sem dados históricos.

Com os desenvolvedores restantes foi realizada uma caraterização a partir de dados que constavam no repositório. A Tabela 5.2 mostra a tendência de crescimento ou decrescimento para cada critério dos desenvolvedores que apresentaram divergência na tendência da reputação.

Tabela 5.2 – Tendência dos critérios de reputação dos casos divergentes

Desenvolvedor		Opinião	Comportamento	Similaridade	Relacionamento	Expertise	Confiabilidade
1	Real	↓	↑	↓	↓	↑	↓
	Simulado	↑	↑	↓	↓	↑	↑
8	Real	↓	↓	↓	↓	↑	↓
	Simulado	↓	↑	↓	↓	↑	↓
9	Real	↓	↓	↑	↓	↑	↓
	Simulado	↑	↑	↑	↑	↓	↑
27	Real	↓	↓	↓	↓	↑	↓
	Simulado	↓	↓	↑	↓	↑	↓
49	Real	↓	↑	↓	↓	↓	↓
	Simulado	↑	↑	↓	↑	↑	↑
53	Real	↓	↑	↓	↓	↑	↓
	Simulado	↑	↑	↓	↓	↑	↑
89	Real	↓	↑	↓	↓	↑	↓
	Simulado	↑	↑	↓	↑	↑	↑

Embora o modelo simbolize a dinâmica da reputação, e se espera que a simulação ofereça um comportamento mais próximo possível do real, trata-se de uma representação e entende-se que na realidade podem ocorrer situações que variam a tendência simulada. Por essa razão, buscou-se encontrar um padrão de comportamento, das tendências de cada critério, que pudesse resultar em tendências diferentes para a reputação real e simulada, dos desenvolvedores.

Como resultado da análise das informações presentes na Tabela 5.2, não foi possível identificar um critério que, sozinho, pudesse ter levado todos os casos analisados à divergência da tendência. Entretanto, foi possível perceber que dos sete desenvolvedores que apresentaram tendências divergentes, em apenas um caso o critério Similaridade apresentou divergência, seguido dos critérios Expertise e Comportamento com dois casos de divergência cada um. O critério Relacionamento segue a lista apresentando três casos de divergência na tendência. Os critérios Confiabilidade e Opinião possuem cinco casos de divergência cada um. Embora esses fatos tenham sido observados, necessitam ser mais explorados. Assim, dos 40 desenvolvedores analisados, em 82,5% deles, o modelo apresentou conformidade na indicação da tendência entre os dados reais e os simulados.

A partir do fluxo apresentado para utilização do módulo de previsão e simulação bem como a execução da simulação, através da ferramenta *Insight Maker*, e o armazenamento dos dados simulados, torna-se possível traçar um paralelo entre (i) os desenvolvedores que foram alocados historicamente, com seus valores de reputação, (ii) as informações obtidas a partir da simulação, e (iii) o ranqueamento dos desenvolvedores.

Com o intuito de reduzir o escopo de análise, foram considerados na avaliação os casos de solicitação de mudança que apresentaram desdobramentos. É considerado um desdobramento quando ocorre a reabertura da solicitação de mudança. Em função de, por exemplo, novas falhas identificadas ou o cliente não ter aprovado as alterações aplicadas.

Uma parte importante dos sistemas gerenciadores de solicitação de mudanças é classificar as solicitações de acordo com seu *status*. Cada equipe pode decidir ter um conjunto diferente de categorização para o *status* das solicitações. Portanto, MantisBT oferece a capacidade de personalizar a lista de status, assumindo que uma solicitação pode estar em um dos três estágios: aberto, resolvido e fechado. Assim, a lista de *status* personalizado é mapeada para essas três possibilidades. No caso, MantisBT oferece como padrão, os seguintes *status*: novo, *feedback*, reconhecido, confirmado, atribuído, resolvido e fechado. Por exemplo, a passagem do *status* de "novo" para "atribuído" é associado ao estágio aberto, "resolvido" significa resolvido e "fechado" indica o estágio fechado. A seguir são explicados o que significam cada um dos *status* padrão oferecidos pelo MantisBT:

- Novo: *status* inicial para novas solicitações, permanecendo nesse estado até serem atribuídas, reconhecidas, confirmadas ou resolvidas. O próximo status pode ser "reconhecido", "confirmado", "atribuído" ou "resolvido".
- Reconhecido: *status* é usado pela equipe de desenvolvimento para refletir o acordo deles com a solicitação para a mudança sugerida. Ou para concordar com o que é sugerido em um relatório de mudanças. O próximo *status* geralmente é "atribuído" ou "confirmado".
- Confirmado: normalmente é usado pela equipe de desenvolvimento para mencionar que eles concordam com quem criou a solicitação, e que está sugerido na mudança e que confirmaram e reproduziram o possível problema. O próximo *status* geralmente é "atribuído".
- Atribuído: usado para refletir que o problema foi atribuído a um dos membros da equipe e que esse membro da equipe está trabalhando ativamente na questão. O próximo *status* normalmente é "resolvido".
- Resolvido: usado para refletir que a solicitação foi resolvida. Um problema pode ser resolvido com uma das muitas resoluções (personalizável). Por exemplo, um problema pode ser resolvido como "fixo", "duplicado", "não consertará", "sem alteração necessária", entre outros. Os próximos estados geralmente são "fechado" ou, em caso de reabertura da solicitação, seria "feedback".
- Fechado: reflete que o problema está completamente fechado e não são necessárias mais ações nele. Algumas equipes usam "fechado" para refletir o consentimento de quem cadastrou a solicitação, e outros o usam para refletir o fato de que a solução foi liberada para os clientes.

Um fluxo entre estados deve ser definido para guiar o ciclo de vida da solicitação. Por padrão não é definido nenhum fluxo, isso significa que qualquer estado é acessível através de qualquer outro. No entanto, o fluxo de trabalho precisa ter um caminho dos estados maiores, ou iguais, ao estado "resolvido" de volta ao estado "*feedback*", caso contrário, a operação de reabertura não funcionará. Isso ocorre, pois, "*feedback*" é o estado a ser atribuído à solicitação quando esta é reaberta.

Essa explicação sobre os padrões de *status* e o que eles significam, feita até aqui, é necessária já que pode refletir uma particularidade do sistema MantisBT, ocorrendo de maneira diferente em outros gerenciadores do gênero.

Para classificar o estado de uma solicitação de mudança, foi utilizada a configuração padrão tal qual explicada na documentação do MantisBT, mesmo que os dados da empresa parceira utilizem estados intermediários, ou customizados, para seus casos. Essa escolha visa facilitar a replicação das análises para outras fontes de dados.

Inicialmente, na condução da análise era preciso descobrir a distribuição e a média de desdobramentos ao longo do tempo. Os dados de desdobramentos foram coletados via Mantis, do repositório de solicitação de mudança. A Tabela 5.3 mostra a média obtida em um período estabelecido de três meses e, em seguida, de seis meses. Também se destaca a distribuição das quantidades de solicitações que foram reabertos no período.

Tabela 5.3 – Dados de Desdobramentos

	MÉDIA	QUANTIDADE
ÚLTIMOS 3 MESES	1,088	3
ÚLTIMOS 6 MESES	1,288	20

A escolha da quantidade de meses, no intervalo de análise, se justifica por ter sido um período de troca de funcionários. A Tabela 5.4 apresenta características das 23 solicitações com desdobramentos acima da média, encontrados no período de 6 meses analisados.

Tabela 5.4 – Dados de desdobramentos acima da média

Solicitação de Mudança	Número de Desdobramentos	Número de Desenvolvedores	Prioridade	Severidade
SM_1	2	3	40	10
SM_2	5	2	40	60
SM_3	3	2	40	60
SM_4	2	7	40	50
SM_5	2	4	40	60
SM_6	2	1	30	50
SM_7	3	1	30	60
SM_8	2	3	30	60
SM_9	2	3	40	60

SM_10	3	3	40	60
SM_11	2	3	40	60
SM_12	2	2	30	50
SM_13	2	1	20	40
SM_14	2	3	40	60
SM_15	2	4	50	60
SM_16	3	3	40	60
SM_17	2	5	40	60
SM_18	2	4	30	50
SM_19	2	1	30	50
SM_20	4	1	30	50
SM_21	2	2	40	60
SM_22	2	4	40	50
SM_23	2	3	60	80

Com relação ao número de desenvolvedores não é possível, à princípio, observar um padrão de comportamento da variação, de acordo com o número de desdobramentos. O mesmo pode-se notar em relação ao grau de severidade atribuído ao caso e à prioridade do caso. Assim, com o objetivo de verificar o grau de correlação entre o número de desdobramentos e as características citadas de cada solicitação de mudança, foram realizados 3 testes de correlação rodados na ferramenta MiniTab[19], descritos a seguir.

O primeiro teste foi conduzido com o objetivo de investigar se o número de desdobramentos pode ser justificado pelo número de desenvolvedores que trataram as solicitações. A questão de pesquisa que guiou esse processo foi:

"Existe relação entre número de desdobramentos e o número de desenvolvedores? "

Assim, foi estabelecida a hipótese nula e sua alternativa, da seguinte forma:

*H_0: **Não** existe correlação entre as variáveis número de desdobramentos e número de desenvolvedores.*

[19] www.minitab.com/pt-BR/

*H_1: **Existe** correlação entre as variáveis número de desdobramentos e número de desenvolvedores.*

Para realizar esta análise de correlação, primeiro a normalidade foi verificada para cada variável. Como a quantidade de amostras não é superior a 30, o teste de Shapiro-Wilk foi utilizado. Os gráficos de dispersão para as variáveis número de desdobramentos e número de desenvolvedores podem ser observados na Figura 5.10 e os resultados para testes de normalidade são descritos na Tabela 5.5.

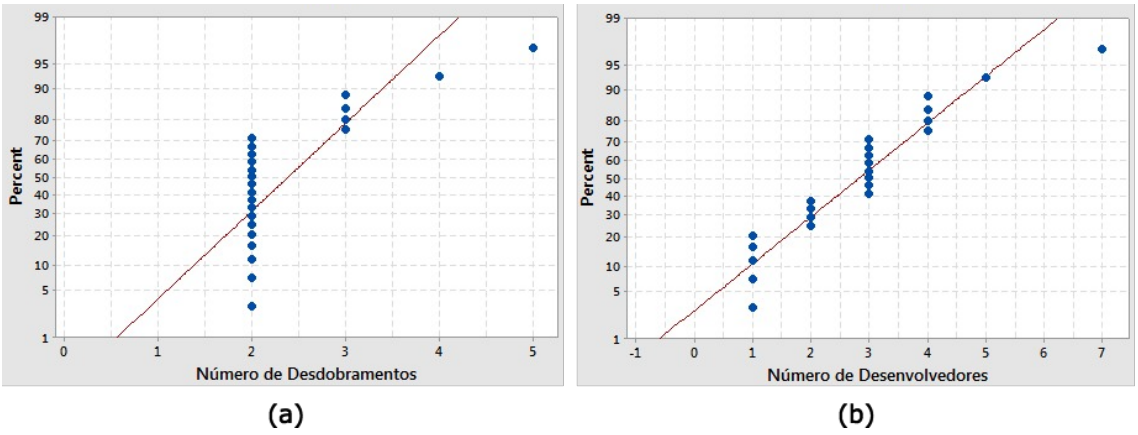

Figura 5.10 – Gráfico de dispersão (a) número de desdobramentos e (b) número de desenvolvedores

Tabela 5.5 – Resultados dos testes de normalidade para Número de Desdobramentos e Número de Desenvolvedores

Variável	Amostras	P-Value	Distribuição
Número de Desdobramentos	23	>0,100	Normal
Número de Desenvolvedores	23	>0,100	Normal

Após a verificação da normalidade, a análise de correlação foi realizada usando o método de Pearson, pois a distribuição de cada variável é normal. O resultado da análise apresentou valor do coeficiente de Correlação igual a -0,334, com p-value = 0,119. Como o *p-value* ficou acima do nível de significância estabelecido de 0,05, rejeita-se a hipótese alternativa, e é aceita a hipótese nula de que não existe correlação significativa entre as variáveis do ponto de vista estatístico. Dessa forma, é possível mostrar, nesse caso, que os desdobramentos não estão associados à quantidade de desenvolvedores alocados para tratar os casos.

O segundo teste foi conduzido com o objetivo de investigar se o número de desdobramentos pode ser justificado pela prioridade atribuída às solicitações. A questão de pesquisa que guiou este teste foi:

"Existe relação entre número de desdobramentos e a prioridade das solicitações de mudança? "

Assim, foi estabelecida a hipótese nula e sua alternativa, da seguinte forma:

H_0: ***Não*** *existe correlação entre as variáveis número de desdobramentos e prioridade.*

H_1: ***Existe*** *correlação entre as variáveis número de desdobramentos e prioridade.*

Para realizar esta análise de correlação, a normalidade das amostras da variável prioridade foi verificada, utilizando o teste de Shapiro-Wilk, uma vez que a quantidade de amostras é inferior a 30. Como as amostras para a variável número de desdobramentos são iguais ao primeiro teste, sabe-se que sua distribuição é normal. Assim, o gráfico de dispersão para a variável prioridade pode ser observado na Figura 5.11, destacando ainda a legenda com o resultado para o teste de normalidade.

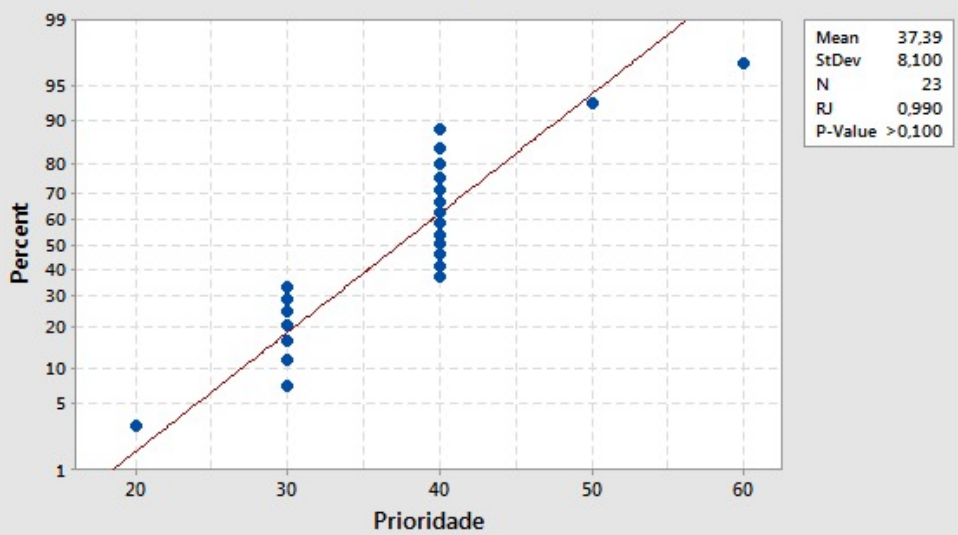

Figura 5.11 - Gráfico de dispersão para a variável Prioridade com resultado do teste de normalidade

Após a verificação da normalidade, a análise de correlação foi realizada usando o método de Pearson, pois a distribuição de cada variável é normal. O resultado da análise apresentou valor do coeficiente de Correlação igual a -0,047 e o *p-value* = 0,832. Como o *p-value* ficou acima do nível de significância estabelecido de 0,05, rejeita-se a hipótese alternativa e é aceita a hipótese nula de que não existe correlação entre as variáveis. Dessa forma, é possível mostrar, nesse caso, que os desdobramentos também não estão associados à prioridade atribuída às solicitações de mudança.

Finalmente, o terceiro teste foi conduzido com o objetivo de investigar se o número de desdobramentos pode ser justificado pela severidade associada às solicitações. A questão de pesquisa que guiou este teste foi:

"Existe relação entre número de desdobramentos e a severidade das solicitações de mudança? "

Assim, foi estabelecida a hipótese nula e sua alternativa, da seguinte forma:

H_0: ***Não*** *existe correlação entre as variáveis número de desdobramentos e severidade.*

H_1: ***Existe*** *correlação entre as variáveis número de desdobramentos e severidade.*

Para realizar esta análise de correlação, a normalidade das amostras da variável severidade foi verificada, utilizando o teste de Shapiro-Wilk, uma vez que a quantidade de amostras é inferior a 30. Mais uma vez, as amostras para a variável número de desdobramentos são iguais ao primeiro teste, portanto, sabe-se que sua distribuição é normal. Assim, o gráfico de dispersão para a variável severidade pode ser observado na Figura 5.12, destacando o resultado para o teste de normalidade.

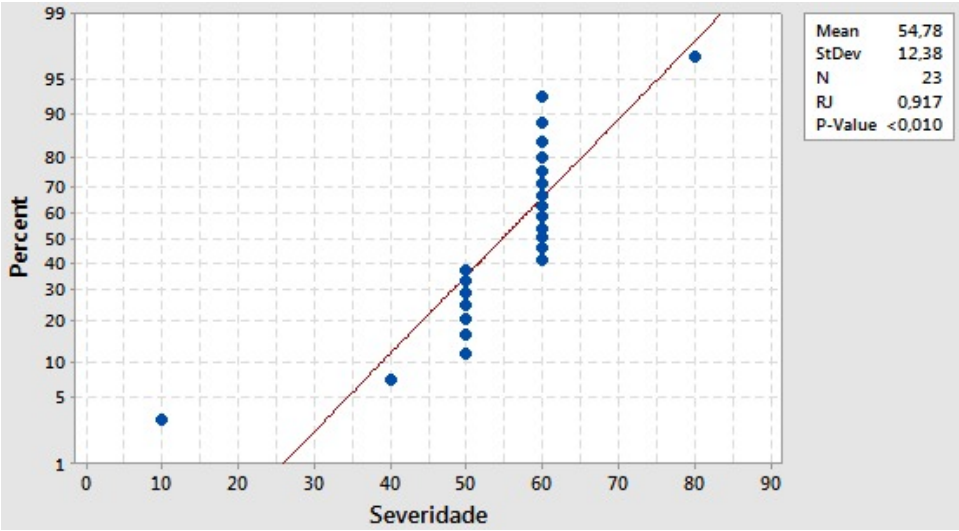

Figura 5.12 - Gráfico de dispersão para a variável Severidade

Com o *p-value* <0,01 que é menor que o nível de significância estabelecido de 0,05, observou-se que as amostras da variável Severidade não possuem uma distribuição normal. Assim, a análise de correlação foi realizada usando o método de Spearman. O resultado da análise apresentou valor do coeficiente de Correlação igual a 0,211 e o *p-value* = 0,335. Como o *p-value* ficou acima do nível de significância estabelecido de 0,05, rejeita-se a hipótese alternativa e é aceita a hipótese nula de que não existe correlação entre as variáveis. Dessa forma, é possível mostrar que os desdobramentos,

nesse caso, também não estão associados à severidade associada às solicitações de mudança.

Com os resultados obtidos nas análises de correlação, foi possível mostrar que, no caso dos dados analisados, a quantidade de desdobramentos não está associada a nenhuma das 3 características das solicitações de mudança. Caso alguma característica estivesse relacionada à quantidade de desdobramentos, a análise mostrada a seguir ficaria condicionada a esta característica, necessitando de maior investigação. Dessa forma o processo de avaliação prosseguiu e, na sequência, o Repositório de Informações de Reputação foi acessado via Módulo de Ranqueamento. A consulta retornou as informações indicando a posição de cada desenvolvedor no Ranqueamento. Vale destacar que as informações retornadas pelo módulo foram armazenadas durante o processo descrito no início deste experimento para avaliação da infraestrutura. Tal fato justifica a escolha do conjunto de informações de reputação, no período de 6 meses que foram simulados durante esse processo descrito para que pudessem ser comparados com as escolhas de alocação de desenvolvedores mostrada mais a frente.

Vale lembrar que o módulo de ranqueamento não constitui uma *feature* de recomendação. Assim, a intensão não é representar os desenvolvedores, com maior pontuação de reputação, como sendo os melhores recomendados para tratar um caso.

A Tabela 5.6 expõe os dados das solicitações de mudança analisados, com o número de desdobramentos, a relação dos desenvolvedores que trataram o referido caso de solicitação e os desenvolvedores mostrados através do módulo de Ranqueamento. Foram destacados os casos nos quais não houve correspondência entre desenvolvedores alocados e ranqueados. A análise foi realizada considerando os desenvolvedores que ocupavam as posições do ranqueamento, na mesma quantidade de desenvolvedores alocados. Por exemplo, para a solicitação SM_1 foram alocados 3 desenvolvedores, assim foram coletadas as informações das 3 primeiras posições do ranqueamento. Nesta etapa é válida a utilização dos valores absolutos já que o objetivo é comparar as informações entre desenvolvedores. A porcentagem disposta na Tabela 5.6, representa uma porcentagem de associação e mostra quantos desenvolvedores foram alocados entre os que estão ranqueados.

Tabela 5.6 – Relação Desenvolvedores Alocados e Ranqueados

Solicitação de Mudança	Desenvolvedores Alocados	Desenvolvedores Ranqueados	%
SM_1	DEV_11	DEV_11	66,67

	DEV_2	DEV_2	
	DEV_94	***DEV_8***	
SM_2	DEV_2	DEV_2	50,00
	DEV_95	***DEV_8***	
SM_3	DEV_2	DEV_2	50,00
	DEV_95	***DEV_8***	
SM_4	DEV_8	DEV_8	28,57
	DEV_20	DEV_20	
	DEV_28	***DEV_11***	
	DEV_81	***DEV_9***	
	DEV_39	***DEV_53***	
	DEV_70	***DEV_8***	
	DEV_12	***DEV_22***	
SM_5	DEV_2	DEV_2	25,00
	DEV_21	***DEV_8***	
	DEV_95	***DEV_11***	
	DEV_12	***DEV_9***	
SM_6	***DEV_94***	***DEV_2***	0,00
SM_7	***DEV_95***	***DEV_2***	0,00
SM_8	DEV_11	DEV_11	66,67
	DEV_8	DEV_8	
	DEV_39	***DEV_2***	
SM_9	DEV_11	DEV_11	66,67
	DEV_2	DEV_2	
	DEV_94	***DEV_8***	
SM_10	DEV_11	DEV_11	66,67
	DEV_2	DEV_2	
	DEV_94	***DEV_8***	
SM_11	DEV_11	DEV_11	66,67
	DEV_2	DEV_2	
	DEV_86	***DEV_8***	

SM_12	DEV_2	DEV_2	50,00
	DEV_94	***DEV_8***	
SM_13	DEV_2	DEV_2	100,00
SM_14	DEV_11	DEV_11	66,67
	DEV_2	DEV_2	
	DEV_94	***DEV_8***	
SM_15	DEV_11	DEV_11	50,00
	DEV_8	DEV_8	
	DEV_53	***DEV_2***	
	DEV_93	***DEV_9***	
SM_16	DEV_11	DEV_11	66,67
	DEV_2	DEV_2	
	DEV_94	***DEV_8***	
SM_17	DEV_11	DEV_11	40,00
	DEV_2	DEV_2	
	DEV_95	***DEV_8***	
	DEV_94	***DEV_9***	
	DEV_82	***DEV_53***	
SM_18	DEV_11	DEV_11	50,00
	DEV_8	DEV_8	
	DEV_39	***DEV_2***	
	DEV_93	***DEV_9***	
SM_19	***DEV_86***	***DEV_2***	0,00
SM_20	***DEV_21***	***DEV_2***	0,00
SM_21	DEV_2	DEV_2	50,00
	DEV_94	***DEV_8***	
SM_22	DEV_11	DEV_11	50,00
	DEV_8	DEV_8	
	DEV_63	***DEV_2***	
	DEV_93	***DEV_9***	
SM_23	DEV_11	DEV_11	66,67

	DEV_8	DEV_8	
	DEV_53	***DEV_2***	

Em um contexto onde a manutenção é feita de forma colaborativa, espera-se que os desenvolvedores alocados colaborem com o objetivo de tratar a tarefa de manutenção. Como mostrado anteriormente, as características (quantidade de desenvolvedores envolvidos, severidade e prioridade) das solicitações de mudança, não apresentaram relação com a quantidade de desdobramentos dos casos. Esse fato complementa o distanciamento observado na comparação, da Tabela 5.6, entre os desenvolvedores ranqueados pelos índices de reputação, e os desenvolvedores escolhidos pelo gerente. Essas duas situações podem oferecer indícios de que os desenvolvedores alocados, embora trabalhando juntos no mesmo caso, não tenham efetivamente colaborado como deveriam, tendo apenas cooperado, ou seja, cada desenvolvedor executa em sequência e, a seu tempo, uma parte da solução, o que pode ter levado à reabertura dos casos.

O distanciamento observado entre os desenvolvedores alocados pelo gerente e os desenvolvedores com os maiores índices de reputação é compreensível até pela ausência de informações de reputação e de um sistema com visualizações e elementos de colaboração que pudessem apoiar o gerente em sua escolha. Outra análise importante é reconhecer a possibilidade de que diferentes fatores tenham levado à diferença observada entre ranqueados e alocados, inclusive a ausência de informações de reputação, no momento do gerente fazer sua escolha. Além disso, o valor pode diferir porque a abordagem se baseia em um modelo que é uma representação da realidade, como comentado anteriormente, ou porque os diferentes fatores atuaram.

Diante disso, foi realizada uma análise com base nos dados oferecidos pela empresa parceira na tentativa de compreender esses fatores que influenciam na escolha dos gerentes. Buscou-se encontrar um padrão de comportamento a partir da caracterização dos desenvolvedores, das equipes formadas a partir das alocações feitas, do período de permanência do desenvolvedor na empresa, da disponibilidade dos desenvolvedores, da caracterização dos relatores das solicitações de mudança e dos papéis desempenhados por cada desenvolvedor. O objetivo foi investigar se existem perfis de alocação que os gestores seguiram nesses casos. Essa análise é pertinente pois, entende-se a reputação como um desses fatores e assim, um perfil de alocação.

Embora a disponibilidade do desenvolvedor tenha sido uma informação oferecida pela empresa, diferentes aspectos podem determinar sua indisponibilidade, tais como afastamento do trabalho por motivo de saúde, férias ou estava tratando outra solicitação. Essas possibilidades não foram verificadas pois não foi cedido esse nível de informação.

Inicialmente, foi conduzida uma caracterização dos grupos que trataram os casos, com a classificação dos desenvolvedores entre novatos e veteranos. Os desenvolvedores que entraram na empresa no período em que os dados históricos foram coletados, tiveram a classificação de novatos. Os anteriores a essa data dos dados históricos, foram considerados veteranos. Com isso, os novatos possuem tempo de permanência na empresa que varia até um ano, e os veteranos estão na empresa a mais de 1 ano. A Tabela 5.7 mostra a classificação entre veteranos e novatos, considerando a lista de desenvolvedores que aparecem nos casos analisados.

Tabela 5.7 – Caracterização dos desenvolvedores

Classificação dos desenvolvedores em relação ao tempo de empresa	Desenvolvedores
Veterano	DEV_2, DEV_8, DEV_11, DEV_12, DEV_20, DEV_21, DEV_22, DEV_28, DEV_39, DEV_53, DEV_63, DEV_70, DEV_82
Novato	DEV_86, DEV_88, DEV_89, DEV_92, DEV_93, DEV_94, DEV_95

Outra caracterização foi realizada, em relação aos casos, e se foram tratados por uma equipe ou individualmente. A Tabela 5.8 mostra a classificação geral com a quantidade de casos pertencente a cada classe.

Tabela 5.8 – Caracterização dos casos de solicitação

Classificação Geral	**Quantidade**
Casos tratados em equipe	18
Casos tratados individualmente	5

Relacionando a caracterização dos desenvolvedores e a caracterização dos casos de solicitação, foi possível dividir os casos de solicitação em 5 categorias, sendo 3 para os casos tratados em equipe, e 2 para os casos tratados individualmente. A Tabela 5.9 e

Tabela 5.10 mostram as categorias para casos tratados por equipes, e para casos tratados individualmente. É mostrada também, a quantidade de casos analisados aderente a cada categoria.

Tabela 5.9 – Categorias para os casos tratados em equipe

Classificação dos casos tratados em equipe quanto a sua formação	**Quantidade**
Novato + Veterano	15
Somente Novato	0
Somente Veterano	3

Tabela 5.10 – Categorias para os casos tratados individualmente

Classificação dos casos tratados individualmente quanto a sua formação	**Quantidade**
Novato	3
Veterano	2

A partir do resultado das classificações, observou-se que 18 dos 23 casos (78,26%), foram tratados por equipes. Dessas equipes, 15 eram formadas por veteranos e por novatos, trabalhando junto. Isso representa 65,22% dos casos analisados. O número de equipes formadas apenas por veteranos foi de 3, representando 13,04% dos casos analisados, e não foram encontradas equipes formadas apenas por novatos. Com relação aos casos tratados individualmente, 3 dos 5 casos (60%) foram tratados por novatos, representando 13,04% dos casos analisados. Isso também pode ser interpretado como oportunidades oferecidas pelo gerente, para o aprendizado e interação do desenvolvedor novato.

Diante disso, em 78,26% dos casos analisados foram encontrados indícios que permitem classificar esses casos como aderentes ao perfil de alocar desenvolvedores novatos e veteranos oferecendo a oportunidade para colaboração e interação. Essa porcentagem representa os casos tratados individualmente, por novatos, somados aos casos tratados em equipe, por veteranos e novatos.

Através da caracterização dos relatores das solicitações de mudança e dos papéis desempenhados por cada desenvolvedor, foi realizada a comparação entre os

desenvolvedores que trataram os casos e os que relataram e registraram os casos. Foi possível observar que em 7 dos 23 casos analisados, representando 30,43%, outro perfil de alocação foi encontrado, no qual o desenvolvedor que relatou e registrou, o caso de solicitação, também participou da equipe ou tratou individualmente o referido caso. O aspecto de ser o relator pode indicar sua familiaridade inicial com a solicitação. Diante disso, foram encontrados indícios que permitem classificar os 7 casos como aderentes à esse outro perfil de alocação encontrado.

Este experimento mostrou a viabilidade do modelo de reputação, pela simulação conduzida. Diferenças entre desenvolvedores alocados e ranqueados são possíveis e esperadas. No entanto, análises adicionais devem ser feitas com dados de outras empresas para a comparação com este estudo, além de traçar outros perfis de alocação. A partir da comparação dos desenvolvedores e seus valores de reputação, mostrou-se como esses dados simulados, podem ajudar o gerente ao alocar os desenvolvedores.

5.3 ANÁLISE GERAL

As análises gerais seguem perspectivas distintas, em função da realização da prova de conceito e do experimento. No tocante à base de dados utilizadas e à metodologia de desenvolvimento usada pela equipe, cabe ressaltar que se utilizou uma base de dados, durante toda a avaliação, que possuía os dados de requisições de mudanças bem documentados, sem dados faltantes ou espaços em branco. A aplicação desse estudo em bases mal documentadas, com informações inconsistentes ou dúbias, pode dificultar a aplicação do modelo apresentado. Por outro lado, isso serve de indicador e sugestão para o preenchimento e manutenção da corretude e completude dos dados registrados nas solicitações. Do mesmo modo, não houve nenhum tipo de teste em relação à utilização da infraestrutura a uma empresa que utiliza outra metodologia de desenvolvimento que não seja ágil, como por exemplo o SCRUM (SMITH, 2016), adotado pela empresa parceira que forneceu os dados para a avaliação.

Embora as informações de grupo e domínio sejam enriquecidas pela ontologia ArchiRIOnt, ou seja, sejam exibidas considerando as inferências feitas através da ontologia e, assim, tenham sido acessadas durante a avaliação, não foi objetivo do experimento ou da prova de conceito, mostrar o processo de inferência e aplicabilidade da ontologia, descritos em (LÉLIS et al., 2016b).

O processo de alocação representado na prova de conceito foi o resultado da composição de atividades que podem ser realizadas em paralelo, ou em situações diversas, não necessariamente no momento de alocar um ou mais desenvolvedores. Por exemplo, o procedimento usado para executar as simulações e armazenar no repositório de informações de reputação, pode ser conduzido a qualquer momento que o gerente julgar necessário ou para embasar uma intervenção junto a um desenvolvedor. Essa intervenção pode ser uma conversa, a oferta de um treinamento ou mesmo uma promoção. Por essa razão, não se julgou necessário medir o tempo despendido pelo gerente durante a condução da avaliação.

Com relação à utilização da infraestrutura no processo de alocação, a análise feita pelo gerente evidenciou a possibilidade de ser evoluída por meio de novas funcionalidades, como:

- **Identificação mais clara dos integrantes dos grupos:** embora as legendas das visualizações indiquem os integrantes, o gerente sugeriu a disponibilização das fotos abaixo da seleção de desenvolvedores. **Prioridade**: Baixa.
- **Identificação das mensagens enviadas e das referências feitas ao usuário "logado" nas mensagens em massa:** o gerente relatou dificuldade de diferenciar mensagens enviadas por ele, das demais mensagens. Além disso, reconhecer as mensagens que deveria responder para todos os usuários ou membros da equipe. **Prioridade:** Média.
- **Importação do arquivo na ferramenta de simulação:** embora o procedimento tenha sido considerado simples, a automatização da importação deixaria a simulação mais direta. **Prioridade:** Baixa.
- **Ordenação dos desenvolvedores:** o gerente relatou dificuldade ao escolher com qual desenvolvedor iria iniciar a análise. Uma sugestão seria apresentar o ranqueamento com a ordem dos valores de reputação. Além disso, um filtro baseado na prioridade e severidade dos casos anteriormente tratados. **Prioridade:** Alta.

5.4 CONSIDERAÇÕES FINAIS SOBRE O CAPÍTULO

Este capítulo apresentou uma avaliação da infraestrutura proposta neste trabalho. Foi apresentada uma prova de conceito e um experimento ambos baseados em dados

reais, fornecidos por uma empresa parceira. A prova de conceito mostrou a viabilidade de uso e aderência das visualizações, bem como, dos aspectos de colaboração, oferecidas pela infraestrutura. No experimento, os dados foram utilizados para a construção das equações necessárias para o processo de simulação e observação do comportamento da reputação dos desenvolvedores. Além disso, foi possível relacionar as informações simuladas com a escolha do gerente ao designar desenvolvedores às solicitações.

Por fim, foram apresentados e interpretados os resultados obtidos, bem como algumas limitações encontradas.

6. CONSIDERAÇÕES FINAIS E PERSPECTIVAS FUTURAS

Designar desenvolvedores para atividades de manutenção se torna uma tarefa não trivial, em um cenário distribuído, no qual a colaboração é fundamental para o bom funcionamento das atividades. Por outro lado, é importante garantir a "sobrevivência" de sistemas de software, devido à sua importância para as organizações, mantendo sua qualidade de forma a possibilitar a acomodação de novas mudanças mais facilmente.

A reputação é um elemento chave e influencia a colaboração entre desenvolvedores distribuídos. Acompanhar a evolução da reputação é importante para promover a colaboração nas atividades de manutenção. Existem poucos estudos que oferecem recursos de reputação no contexto da manutenção colaborativa e distribuída de software. Dentre as publicações encontradas na literatura técnica, observou-se uma escassez de propostas que permitam gerenciar e acompanhar as informações de reputação, de forma a apoiar a tomada de decisões de curto a longo prazo, como o exemplo da intervenção, mostrada no capítulo anterior.

Este trabalho apresentou um modelo, suportado pelas técnicas da Dinâmica de Sistemas, para calcular a reputação dos desenvolvedores de software. A teoria da Dinâmica de Sistemas pode ser aplicada e, através dos dados obtidos, é possível entender o passado, estabelecer o que acontece no presente e projetar o comportamento futuro do objeto em análise. Modelos desse tipo, podem ser considerados como modelos de predição, desde que exista estabilidade nas informações geradas e utilizem bases de dados confiáveis.

Existem duas maneiras possíveis de trabalhar com a construção de modelos baseados em Dinâmica de Sistemas. A primeira se dá através de atividades exploratórias, em que o usuário busca um modelo ou uma representação previamente modelada. Nesse caso, o usuário interage com as simulações e está limitado à manipulação dos parâmetros, o que promove habilidades básicas de pesquisa, como o entendimento e variações de causalidade e pode influenciar diretamente a aquisição de conhecimento, aceitando um nível mais baixo de conhecimento inicial por parte do usuário. A outra maneira se dá através de atividades expressivas, nas quais o usuário apresenta ou expõe sua visão ou o modelo mental da situação que está sendo modelada. A construção de modelos promove habilidades de resolução de problemas mais gerais e a transferência dessas habilidades para outras áreas de pesquisa.

Assim, o modelo proposto utiliza-se das influências identificadas entre os critérios de reputação, permitindo observar e simular como a reputação se comporta ao longo do tempo. Este modelo serviu de base para a construção de uma infraestrutura para informações de reputação dinâmica (IRID), que visa permitir a gestão e acompanhamento de informações de reputação, dos desenvolvedores distribuídos geograficamente, para apoiar a alocação desses desenvolvedores nas tarefas de manutenção. Além disso, fornece elementos de visualização e colaboração, em um ambiente integrado para atividades de manutenção de software.

Essa infraestrutura pode ser uma ferramenta importante no processo de tomada de decisão por parte dos gerentes de projeto. Auxiliando no planejamento de atividades de manutenção, nas estimativas do esforço a ser despendido em tais atividades, bem como, no suporte aos gerentes na formação de equipes.

6.1 CONTRIBUIÇÕES

As principais contribuições desta pesquisa são:

- Refinamento de um mapeamento sistemático, no intuito de buscar evidências na literatura técnica sobre como os critérios de reputação influenciam uns nos outros.
- Construção de um modelo dinâmico para o cálculo da reputação de desenvolvedores de software:

  - Definição de um conjunto de métricas para cada critério capazes de representar o critério, sendo utilizadas para a instanciação do modelo.
  - Indicações sobre as influências entre os critérios de reputação.
- Elaboração de uma infraestrutura para gerenciamento e acompanhamento da reputação de desenvolvedores de software:
  - Definição de um modelo conceitual, chamado CoMoRI, para a representação de informações de reputação.
  - Definição de uma ontologia, chamada ArchiRIOnt, apoiada pelo modelo CoMoRI, para enriquecer as informações de reputação obtidas.
  - Elaboração de visualizações disponibilizadas através de *Dashboards* compondo as informações de reputação nos níveis estabelecidos no modelo CoMoRI.
  - Construção de componentes de apoio computacional (módulo PDR e módulo de Previsão e Simulação), para automatizar a geração das equações a serem computadas através de modelos dinâmicos, no sentido de simular o comportamento futuro da reputação dos desenvolvedores.
  - Integração da ferramenta InsightMaker para a execução das simulações do modelo dinâmico e a análise dos resultados.
  - Formação de um repositório de informações de reputação, possibilitando a consulta aos dados por aplicações externas.
- Apresentação de uma prova de conceito e um experimento utilizando a infraestrutura proposta, avaliando o modelo e os demais recursos de visualização e colaboração, apresentados.

6.2 LIMITAÇÕES E AMEAÇAS À VALIDADE

Algumas limitações e ameaças à validade foram identificadas. Uma ameaça à validade advém do conjunto de premissas de observação apresentado. Essas premissas foram construídas a partir de indícios encontrados na literatura técnica e são trabalhadas em conjunto, em diferentes níveis de abstração (individual e entre desenvolvedores). Por isso, poderiam ser objeto de estudos experimentais para verificar se alcançaram seus objetivos de descrever o comportamento da reputação de cada desenvolvedor de software, sendo necessário medir e avaliar a distância conceitual entre os resultados

obtidos na simulação e a realidade apresentada das informações de reputação ao longo de sucessivas ocorrências de solicitações de mudança.

Uma dificuldade encontrada para este estudo, que implica em outra ameaça, foi a pouca disponibilidade de empresas que pudessem oferecer dados reais. Além disso, a integração da infraestrutura apenas com repositórios de solicitação de mudanças do sistema Mantis constitui outra limitação.

Embora as avaliações apresentadas tenham utilizado dados reais, as métricas não foram previamente coletadas com o objetivo de serem utilizadas para a geração das informações de reputação, ficando a abordagem aqui apresentada condicionada à qualidade dessas medidas, tanto em relação a seus registros históricos, quanto da coleta e interpretação a partir dos repositórios disponíveis pela empresa.

6.3 TRABALHOS FUTUROS

A realização deste trabalho de pesquisa levou ao desenvolvimento de uma infraestrutura que gerencia informações de reputação para apoiar a alocação de desenvolvedores a tarefas de manutenção. Embora a prova de conceito tenha sido conduzida por um gerente externo, novos estudos experimentais poderiam ser conduzidos. Como exemplo, estudos de caso com análise de usabilidade para avaliar com mais gestores e desenvolvedores se as visualizações, os recursos de interação e elementos de colaboração são aderentes e favorecem as atividades desempenhadas pela infraestrutura.

Com relação à utilização do sistema, os resultados obtidos abrem novas perspectivas para a evolução da infraestrutura proposta. Além das observações feitas pelo gerente de projeto durante a prova de conceito, outra melhoria seria oferecer a possibilidade de análise comparativa, em uma *dashboard* específica, a partir da *Dashboard* Individual. Dessa forma os desenvolvedores seriam comparados entre suas informações de reputação e não considerados uma equipe, como ocorre na *Dashboard* de Grupo. Além disso, enriquecer os elementos de percepção da lista de desenvolvedores online, oferecendo um resumo dos dados dos desenvolvedores para facilitar a escolha do mais adequado à alocação.

Para que a infraestrutura oferecesse indícios iniciais eram necessárias bases de dados históricos. Dessa forma, os processos utilizados pela empresa parceira e da natureza dos dados que foram disponibilizados associados à manutenção restringiram o

escopo de aplicação da IRID. Assim, a infraestrutura pode ser evoluída para dar suporte à análise de bases de dados diferentes daquela que foi disponibilizada pela empresa parceira, bem como, a adequação das métricas para incluir outros artefatos para as análises de forma histórica, tais como: protótipos de interface e artefatos gerados na engenharia de requisitos, que podem ser processados e gerar novos parâmetros para cálculo da reputação. Diante disso, conduzir novos experimentos envolvendo tarefas específicas da fase de desenvolvimento do software e verificar se os critérios de reputação na manutenção e evolução se aplicam também a tarefas da fase de desenvolvimento do software, ampliando o escopo de aplicação e domínio da infraestrutura IRID.

Outro ponto a ser destacado é a aplicação da infraestrutura a um número maior de empresas, com diferentes tamanhos e com repositórios para sistemas de diferentes domínios de aplicação. Como resultado, a utilização da solução poderia proporcionar um maior entendimento dos processos de acompanhamento e predição da reputação, visando uma melhor calibração do modelo e, consequentemente, da infraestrutura e seus recursos.

As fórmulas e algoritmos encontrados na literatura técnica ofereceram embasamento teórico para o cálculo dos critérios de reputação identificados, foi possível também identificar as influências entre esses critérios. No entanto, a agregação para o valor final de reputação foi realizada através da média aritmética desses critérios. Uma perspectiva futura interessante seria a possibilidade de aplicar uma ponderação sobre os valores de cada critério. Dessa forma, a ponderação seria relativa à perspectiva do gerente associado ao projeto específico em análise. Nesse sentido, outra possibilidade é a elaboração de um protocolo de medição que auxilie a implantação e utilização da infraestrutura em diferentes empresas, por exemplo, um framework para a implantação da IRID com recomendações sobre a aplicação do processo de coleta de métricas para ser usado pela indústria.

E por fim, evoluir os módulos de inferência e ranqueamento de forma a constituírem um Sistema de Recomendação integrado à infraestrutura IRID, com a possibilidade de recomendar: (i) um desenvolvedor para casos de manutenção, (ii) ordem de prioridade para a análise individual dos desenvolvedores, e (iii) formação de grupos.

REFERÊNCIAS

AINSWORTH, S. The functions of multiple representations. **Computers & Education**, v. 33, n. 2, p. 131–152, 1999.

AL-ANI, B.; REDMILES, D. In Strangers We Trust? Findings of an Empirical Study of Distributed Teams. **2009 Fourth IEEE International Conference on Global Software Engineering**, p. 121–130, 2009.

ALBIN, S.; FORRESTER, J. W.; BREIEROVA, L. **Building a System Dynamics Model: Part 1: Conceptualization**. MIT, 2001.

ALNEMR, R.; MEINEL, C. Getting More from Reputation Systems: A Context Aware Reputation Framework Based on Trust Centers and Agent Lists. **2008 The Third International Multi-Conference on Computing in the Global Information Technology (iccgi 2008)**, p. 137–142, 2008.

ALNEMR, R.; MEINEL, C. Reputation objects for interoperable reputation exchange: Implementation and design decisions. **Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2012 8th International Conference on**, p.672–680, 2012.

ARAÚJO, M. A. P. **Um Modelo para Observação de Evolução de Software**. 2009. Tese de Doutorado, PESC/COPPE, Rio de Janeiro, UFRJ. 2009.

AUDY, J. L. N.; PRIKLADNICKI, R. **Desenvolvimento Distribuído de Software**. Elsevier Editora, 2007.

BASILI, V. R.; WEISS, D. M. A Methodology for Collecting Valid Software Engineering Data. **IEEE Transactions on Software Engineering**, v. SE-10, n. 6, p. 728–738, 1984.

BOSU, A.; CARVER, J. C. Impact of developer reputation on code review outcomes in OSS projects: an empirical investigation. **Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement**, p.33, 2014.

CAMILETTI, G. G.; FERRACIOLI, L. The Use of Semiquantitative Computational Modelling in the Study of Predator-Pray System. **X International Organization for Science and Technology Education, 2002, Foz do Iguaçu**, 2002.

CARNEIRO, G. DE F.; CONCEIÇÃO, C. F. R.; DAVID, J. M. N. A multiple view environment for collaborative software comprehension. **ICSEA: The Seventh International Conf. on Software Engineering Advances, Portugal**, p.15–21, 2012.

CASARE, S.; SICHMAN, J. S. Using a functional ontology of reputation to interoperate different agent reputation models. **Journal of the Brazilian Computer Society**, v. 11, n. 2, p. 79–94, 2005.

CAVERLEE, J.; LIU, L.; WEBB, S. The SocialTrust framework for trusted social information management: Architecture and algorithms. **Information Sciences**, v. 180, n. 1, p. 95–112, 2010.

FORRESTER, J. W. **Industrial Dynamics**. 1961.

FORRESTER, J. W. Urban dynamics. **IMR; Industrial Management Review (pre-1986)**, v. 11, n. 3, p. 67, 1970.

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

FORRESTER, J. W. System dynamics and the lessons of 35 years. **A systems-based approach to policymaking**. p.199–240, 1993.

FUKS, H.; RAPOSO, A. B.; GEROSA, M. A.; LUCENA, C. J. P. Do modelo de colaboração 3c à engenharia de groupware. **Simpósio Brasileiro de Sistemas Multimídia e Web-Webmidia**, p. 0–8, 2003.

GALLARDO-VALENCIA, R. E.; TANTIKUL, P.; SIM, S. E. Searching for reputable source code on the web. **Proceedings of the 16th ACM international conference on Supporting group work**, p.183–186, 2010.

HANSSON, K.; KARLSTRÖM, P.; LARSSON, A.; VERHAGEN, H. Actory: A tool for visualizing reputation as a means to formalize informal social behavior. **2nd International Conference on Reputation: "Society, Economy, Trust" ICORE**, 2011.

HEER, J.; SHNEIDERMAN, B. Interactive Dynamics for Visual Analysis. **Communications of the ACM**, v. 55, n. 4, p. 45–54, 2012.

HENDRIKX, F.; BUBENDORFER, K.; CHARD, R. Reputation systems: A survey and taxonomy. **Journal of Parallel and Distributed Computing**, v. 75, p. 184–197, 2015.

ISENBERG, P.; ELMQVIST, N.; SCHOLTZ, J.; et al. Collaborative visualization: definition, challenges, and research agenda. **Information Visualization**, v. 10, n. 4, p. 310–326, 2011.

ISO/IEC 14764. International Standard - ISO/IEC 14764 IEEE Std 14764:2006 **(Revision of IEEE Std 1219-1998) Software Engineering & Software Life Cycle Processes & Maintenance**, 2006.

JARVENPAA, S. L.; SHAW, T. R.; STAPLES, D. S. Toward contextualized theories of trust: The role of trust in global virtual teams. **Information systems research**, v. 15, n. 3, p. 250–267, 2004.

JAYASHREE, R.; CHRISTY, A. Improving the Enhanced Recommended System Using Bayesian Approximation Method and Normalized Discounted Cumulative Gain. **Procedia Computer Science**, v. 50, p. 216–222, 2015.

JIANG, L.; XU, J.; ZHANG, K.; ZHANG, H. A new evidential trust model for open distributed systems. **Expert Systems with Applications**, v. 39, n. 3, p. 3772–3782, 2012.

JØSANG, A. Trust and reputation systems. **Foundations of security analysis and**

design IV. p.209–245, 2007.

JØSANG, A.; ISMAIL, R.; BOYD, C. A survey of trust and reputation systems for online service provision. **Decision support systems**, v. 43, n. 2, p. 618–644, 2007.

JØSANG, A.; QUATTROCIOCCHI, W. Advanced features in Bayesian reputation systems. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, v. 5695 LNCS, p. 105–114, 2009.

KHIABANI, H.; SIDEK, Z. M.; MANAN, J.-L. A. Towards a Unified Trust Model in Pervasive Systems. **2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops**, p.831–835, 2010.

KIM, S.; LEE, S.; KANG, H.; CHO, J. Hybrid WOM Collection and Visualization Method for Reputation Rating in Online Community. **Indian Journal of Science and Technology**, v. 8, n. 18, 2015.

KITCHEHAM, B.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering. **Technical report, Ver. 2.3 EBSE Technical Report. EBSE**, 2007.

LEHMAN, M. M. Laws of software evolution revisited. **European Workshop on Software Process Technology**, p.108–124, 1996.

LÉLIS, C. A. S.; ARAÚJO, M. A. P.; DAVID, J. M. N.; CARNEIRO, G. DE F. Investigating Reputation in Collaborative Software Maintenance: A Study Based on Systematic Mapping. In: S. Latifi (Ed.); **Information Technology: New Generations, 13th International Conference on Information Technology**. p.615–627, 2016.

LÉLIS, C. A. S.; BRAGA, R.; ARAÚJO, M. A. P.; DAVID, J. M. N. ArchiRI - An Ontology-based Architecture for the Exchange of Reputation Information. **Proceedings of the XII Brazilian Symposium on Information Systems on Brazilian Symposium on Information Systems: Information Systems in the Cloud Computing Era - Volume 1, SBSI 2016**, p.9:60--9:67, 2016.

LÉLIS, C. A. S.; MIGUEL, M. A.; ARAÚJO, M. A. P.; DAVID, J. M. N.; BRAGA, R. AD-Reputation: A Reputation-based Approach to Support Effort Estimation. In: S. Latifi (Ed.); **Information Technology - New Generations: 14th International Conference on Information Technology**. p.621--626, 2018.

LÉLIS, C. A. S.; TAVARES, J. F.; ARAÚJO, M. A. P.; DAVID, J. M. N. GiveMe Trace: A Software Evolution Traceability Support Tool. **IEEE Latin America Transactions**, v. 14, n. 7, p. 3444–3454, 2016.

LIU, L.; MUNRO, M. Systematic analysis of centralized online reputation systems. **Decision support systems**, v. 52, n. 2, p. 438–449, 2012.

MEMARMOSHREFI, P.; ALFANDI, O.; KELLNER, A.; HOGREFE, D. Autonomous Group-Based Authentication Mechanism in Mobile Ad Hoc Networks. **2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications**, p.1097–1102, 2012.

MIGUEL, M. A.; ARAUJO, M. A. P.; DAVID, J. M. N.; BRAGA, R. A Framework to Support Effort Estimation on Software Maintenance and Evolution Activities. **Proceedings of the XII Brazilian Symposium on Information Systems on Brazilian Symposium on Information Systems: Information Systems in the Cloud Computing Era - Volume 1, SBSI 2016,** p.232--239, 2016.

MONTGOMERY, D. C. **Introduction to Statistical Quality Control**. 6th ed. Wiley, 2008.

MUI, L.; MOHTASHEMI, M.; HALBERSTADT, A. Notions of reputation in multi-agents systems: a review. **Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1**, p.280–287, 2002.

NASCIMENTO, L. D. C.; SANTORO, F. Análise de interações nas Comunidades Virtuais de Software Livre. **V Simpósio Brasileiro de Sistemas de Informação**, v. 2, n. 2003, p. 12–23, 2009.

O'DONOVAN, J.; SMYTH, B.; EVRIM, V.; MCLEOD, D. Extracting and Visualizing Trust Relationships from Online Auction Feedback Comments. **IJCAI**, p.2826–2831, 2007.

OLIVEIRA BARROS, M. DE. **Gerenciamento de Projetos Baseado em Cenários: uma Abordagem de Modelagem Dinâmica e Simulação**. 2001. UNIVERSIDADE FEDERAL DO RIO DE JANEIRO. 2001.

PAREDES, J.; ANSLOW, C.; MAURER, F. Information visualization for agile software development. **Software Visualization (VISSOFT), 2014 Second IEEE Working Conference on**, p.157–166, 2014.

PETTICREW, M.; ROBERTS, H. **Systematic reviews in the social sciences: A practical guide**. John Wiley & Sons, 2008.

PRESSMAN, R. S.; MAXIM, B. R. **Software Engineering: A Practitioner's Approach (Irwin Computer Science)**. 8th ed. McGraw-Hill Education, 2014.

PRIKLADNICKI, R.; LOPES, L.; AUDY, J. L. N.; EVARISTO, R. Desenvolvimento Distribuído de Software: um Modelo de Classificação dos Níveis de Dispersão dos Stakeholders. **I Brazilian Symposium on Information Systems (SBSI 04)**, 2004.

PRIKLADNICKI, R.; NICOLAS AUDY, J. L.; EVARISTO, R. Global software development in practice lessons learned. **Software Process: Improvement and Practice**, v. 8, n. 4, p. 267–281, 2003.

QU, X.; ZHONG, J.; YANG, X. Towards reliable and trustworthy cooperation in grid: A pre-evaluating set based trust model. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, v. 3984 LNCS, p. 224–235, 2006.

RAMARAO, P.; MUTHUKUMARAN, K.; DASH, S.; MURTHY, N. L. B. Impact of Bug Reporter's Reputation on Bug-Fix Times. **Information Systems Engineering (ICISE), 2016 International Conference on**, p.57–61, 2016.

ROSACI, D.; SARNÉ, G. M. L.; GARRUZZO, S. Integrating trust measures in

multiagent systems. **International Journal of Intelligent Systems**, v. 27, n. 1, p. 1–15, 2012.

SANGER, J.; PERNUL, G. Visualizing Transaction Context in Trust and Reputation Systems. **Availability, Reliability and Security (ARES), 2014 Ninth International Conference on**, p.94–103, 2014.

SÄNGER, J.; PERNUL, G. TRIVIA: visualizing reputation profiles to detect malicious sellers in electronic marketplaces. **Journal of Trust Management**, v. 3, n. 1, p. 5, 2016.

SÄNGER, J.; RICHTHAMMER, C.; KUNZ, M.; MEIER, S.; PERNUL, G. Visualizing Unfair Ratings in Online. **Twenty-Third European Conference on Information Systems, Münster, Germany**, p. 1–15, 2015.

SILVA, A. N.; CARNEIRO, G. F.; ZANIN, R. B.; DAL POZ, A. P.; MARTINS, E. F. O. Propondo uma Arquitetura para Ambientes Interativos baseados em Multiplas Visões. **II Workshop Brasileiro de Visualização de Software**, p.1–8, 2012.

SILVA, L. R. M. **Apoiando Atividades de Estimativa de Esforço a partir de dados de Reputação de Desenvolvedores**. 2017. Universidade Federal de Juiz de Fora. 2017.

SMITH, R. **Scrum Guide: Agile Project Management Guide for Scrum Master and Software Development Team**. CreateSpace Independent Publishing Platform, 2016.

SOMMERVILLE, I. **Software Engineering**. 10th ed. Pearson, 2015.

SPENCE, R. **Information Visualization: Design for Interaction**. Pearson/Prentice Hall, 2007.

STOREY, M.-A.; BENNETT, C.; BULL, R. I.; GERMAN, D. M. Remixing visualization to support collaboration in software maintenance. **Frontiers of Software Maintenance, 2008 FoSM**, p.139–148, 2008.

TAVARES, J. F.; DAVID, J. M. N.; ARAÚJO, M. A. P.; et al. Uma Infraestrutura baseada em Múltiplas Visões Interativas para Apoiar Evolução de Software. **iSys-Revista Brasileira de Sistemas de Informação**, v. 8, n. 1, p. 65–101, 2015.

TRAINER, E. H.; REDMILES, D. F. Foundations for the design of visualizations that support trust in distributed teams. **Proceedings of the International Working Conference on Advanced Visual Interfaces - AVI '12**, p. 34–41, 2012.

VERBIEST, N.; CORNELIS, C.; VICTOR, P.; HERRERA-VIEDMA, E. Trust and distrust aggregation enhanced with path length incorporation. **Fuzzy Sets and Systems**, v. 202, p. 61–74, 2012.

VIPUL, K.; BUSSLER, C.; MORAN, M. **The Semantic Web**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

VOLK, F.; HAUKE, S.; DIETH, D.; MUHLHAUSER, M. Communicating and visualising multicriterial trustworthiness under uncertainty. **2014 Twelfth Annual International Conference on Privacy, Security and Trust**, p.391–397, 2014.

WANG, Y.; REDMILES, D. Understanding Cheap Talk and the Emergence of Trust in

Global Software Engineering : An Evolutionary Game Theory Perspective. **CHASE**, p. 149–152, 2013.

WANG BALDONADO, M. Q.; WOODRUFF, A.; KUCHINSKY, A. Guidelines for using multiple views in information visualization. **Proceedings of the working conference on Advanced visual interfaces**, p.110–119, 2000.

WROBEL, S.; HEUPEL, M.; THIEL, S. Evaluation of the di.me trust metric in CRM settings. **Second International Conference on Future Generation Communication Technologies (FGCT 2013)**, p.132–136, 2013.

XU, J.; JIANG, D.; WANG, B.; YANG, D.; REIFF-MARGANIEC, S. Local Reputation Management in Cloud Computing. **2015 IEEE World Congress on Services**, p.261–267, 2015.

YIN, G.; WANG, Y.; DONG, Y.; DONG, H. Wright–Fisher multi-strategy trust evolution model with white noise for Internetware. **Expert Systems with Applications**, v. 40, n. 18, p. 7367–7380, 2013.

ZHANG, J.; ACKERMAN, M. S.; ADAMIC, L. Expertise networks in online communities: structure and algorithms. **Proceedings of the 16th international conference on World Wide Web**, p.221–230, 2007.

ZUPANCIC, E.; JURIC, M. B. TACO: a novel method for trust rating subjectivity elimination based on Trust Attitudes COmparison. **Electronic Commerce Research**, v. 15, n. 2, p. 207–241, 2015.

APÊNDICE A – Estrutura do arquivo gerado para simulação

O código mostrado a seguir representa o arquivo gerado, destacando as informações para apenas um desenvolvedor. Para tornar mais clara a apresentação das informações foram omitidos os dados de configuração que controlam a apresentação gráfica do modelo, tais como: posição na tela, *layout* de fonte e cores. Essas definições de valores iniciais e as equações são geradas para cada desenvolvedor em análise. Cada trecho do código será explicado separadamente.

```
  ##################################
  SETTINGS
  ##################################
        <Setting TimeStart="1" TimeLength="10" TimeUnits="Days" TimeStep="1" ></Setting>
  ##################################
  STOCKs
  ##################################
  <Stock InitialValue="1" id="352" name="TEMPO_DEV12"/></Stock>
  <Stock InitialValue="(0.0238842982798815) " id="353" name="OPINIAO_DEV12"></Stock>
  <Stock InitialValue="(0.0225901489883711) " id="354" name="COMPORTAMENTO_DEV12"></Stock>
  <Stock InitialValue="(0.0120020788707945) " id="355" name="CONFIABILIDADE_DEV12"></Stock>
  <Stock InitialValue="(0.9604322141509829) " id="356" name="SIMILARIDADE_DEV12"></Stock>
  <Stock InitialValue="(0.0011976551404938) " id="357" name="RELACIONAMENTO_DEV12"></Stock>
  <Stock InitialValue="(0.5485234887196436) " id="358" name="EXPERTISE_DEV12"></Stock>
  ##################################
  VARIABLEs
  ##################################
<Variable Equation = "0 + (0.0000003401453019) * [TEMPO_DEV12]" id="346" name =
"DELTA_OPINIAO_DEV12" > </Variable>
<Variable Equation="0 + (0.3894541161352521) * [DELTA_OPINIAO_DEV12] " id="347" name =
"DELTA_COMPORTAMENTO_DEV12" > </Variable>
<Variable Equation= "0 + (0.4951647199220285) * [DELTA_RELACIONAMENTO_DEV12]  +
(0.4987738285919289) * [DELTA_OPINIAO_DEV12]  + (0.0460571864265914) *
Delay([DELTA_CONFIABILIDADE_DEV2],1,0)  + (0.0598896438387743) *
Delay([DELTA_CONFIABILIDADE_DEV8],1,0)  + (0.0128379682373108) *
Delay([DELTA_CONFIABILIDADE_DEV9],1,0)  + (0.0110694221972206) *
Delay([DELTA_CONFIABILIDADE_DEV11],1,0)  + (0.2080806539173141) *
Delay([DELTA_CONFIABILIDADE_DEV20],1,0)  + (0.2523796149703191) *
Delay([DELTA_CONFIABILIDADE_DEV21],1,0)  + (-0.1105664688823281) *
Delay([DELTA_CONFIABILIDADE_DEV22],1,0)  + (1.1057245136429883) *
Delay([DELTA_CONFIABILIDADE_DEV28],1,0)  + (0.0884210446614236) *
Delay([DELTA_CONFIABILIDADE_DEV37],1,0)  + (-0.3797504809414012) *
Delay([DELTA_CONFIABILIDADE_DEV38],1,0)  + (0.0686531914935148) *
Delay([DELTA_CONFIABILIDADE_DEV39],1,0)  + (0.2173420849580683) *
Delay([DELTA_CONFIABILIDADE_DEV53],1,0)  + (-20.2994748678014150) *
Delay([DELTA_CONFIABILIDADE_DEV57],1,0)  + (-0.0811264554266765) *
Delay([DELTA_CONFIABILIDADE_DEV63],1,0)  + (0.6188707574115284) *
Delay([DELTA_CONFIABILIDADE_DEV70],1,0)  + (0.7081109523918379) *
Delay([DELTA_CONFIABILIDADE_DEV80],1,0)  + (0.1231176176114304) *
Delay([DELTA_CONFIABILIDADE_DEV81],1,0)  + (1.0915542840819630) *
Delay([DELTA_CONFIABILIDADE_DEV82],1,0)  + (2.4112548232459665) *
Delay([DELTA_CONFIABILIDADE_DEV89],1,0)  + (0.0707397043775328) *
Delay([DELTA_CONFIABILIDADE_DEV94],1,0)  + (-0.3400587722691051) *
Delay([DELTA_RELACIONAMENTO_DEV2],1,0)  + (-0.0963545144560207) *
Delay([DELTA_RELACIONAMENTO_DEV8],1,0)  + (0.6257331338019672) *
Delay([DELTA_RELACIONAMENTO_DEV9],1,0)  + (0.1970201055920795) *
Delay([DELTA_RELACIONAMENTO_DEV11],1,0)  + (0.7725472141480799) *
Delay([DELTA_RELACIONAMENTO_DEV20],1,0)  + (-1.0497333429131486) *
Delay([DELTA_RELACIONAMENTO_DEV21],1,0)  + (-1.4439905865494307) *
Delay([DELTA_RELACIONAMENTO_DEV22],1,0)  + (5.5539968642268205) *
Delay([DELTA_RELACIONAMENTO_DEV28],1,0)  + (0.6326464426062013) *
Delay([DELTA_RELACIONAMENTO_DEV37],1,0)  + (-3.6779291737848770) *
```

Delay([DELTA_RELACIONAMENTO_DEV38],1,0) + (0.7863928085628252) *
Delay([DELTA_RELACIONAMENTO_DEV39],1,0) + (-0.7471152883988252) *
Delay([DELTA_RELACIONAMENTO_DEV53],1,0) + (-99.2835899086230100) *
Delay([DELTA_RELACIONAMENTO_DEV57],1,0) + (-2.8441517026940035) *
Delay([DELTA_RELACIONAMENTO_DEV63],1,0) + (2.4323137712981270) *
Delay([DELTA_RELACIONAMENTO_DEV70],1,0) + (9.0884569784870980) *
Delay([DELTA_RELACIONAMENTO_DEV80],1,0) + (1.3362659384628930) *
Delay([DELTA_RELACIONAMENTO_DEV81],1,0) + (11.2385804026301330) *
Delay([DELTA_RELACIONAMENTO_DEV82],1,0) + (22.8907310292576300) *
Delay([DELTA_RELACIONAMENTO_DEV89],1,0) + (0.4821019035379895) *
Delay([DELTA_RELACIONAMENTO_DEV94],1,0)" id="348" name = "DELTA_CONFIABILIDADE_DEV12">
</Variable>

<Variable Equation="0 + (-0.6929288134670968) * [DELTA_COMPORTAMENTO_DEV12] " id="349" name =
"DELTA_SIMILARIDADE_DEV12" > </Variable>

<Variable Equation="0 + (0.1040300713546633) * [DELTA_SIMILARIDADE_DEV12] + (0.0060976737176697)
* [DELTA_OPINIAO_DEV12] + (-0.0019055114555995) * Delay([DELTA_OPINIAO_DEV2],1,0) +
(0.0199906773684511) * Delay([DELTA_OPINIAO_DEV8],1,0) + (-0.0107809141898423) *
Delay([DELTA_OPINIAO_DEV9],1,0) + (-0.0011445626429062) * Delay([DELTA_OPINIAO_DEV11],1,0) + (-
0.0177726074322293) * Delay([DELTA_OPINIAO_DEV20],1,0) + (-0.0138059437844120) *
Delay([DELTA_OPINIAO_DEV21],1,0) + (0.0116717226781451) * Delay([DELTA_OPINIAO_DEV22],1,0) + (-
0.0940239694807855) * Delay([DELTA_OPINIAO_DEV28],1,0) + (-0.0071411080888717) *
Delay([DELTA_OPINIAO_DEV37],1,0) + (0.0543473657104923) * Delay([DELTA_OPINIAO_DEV38],1,0) + (-
0.0044552756603707) * Delay([DELTA_OPINIAO_DEV39],1,0) + (-0.0203178692356871) *
Delay([DELTA_OPINIAO_DEV53],1,0) + (2.0981477142280704) * Delay([DELTA_OPINIAO_DEV57],1,0) + (-
0.0884077849165304) * Delay([DELTA_OPINIAO_DEV63],1,0) + (-0.0306248105320729) *
Delay([DELTA_OPINIAO_DEV70],1,0) + (-0.1029065843667328) * Delay([DELTA_OPINIAO_DEV80],1,0) +
(-0.0121115575044090) * Delay([DELTA_OPINIAO_DEV81],1,0) + (-0.1560660939006026) *
Delay([DELTA_OPINIAO_DEV82],1,0) + (-0.3189752104058661) * Delay([DELTA_OPINIAO_DEV89],1,0) +
(-0.0045476341347808) * Delay([DELTA_OPINIAO_DEV94],1,0) " id="350" name =
"DELTA_RELACIONAMENTO_DEV12" > </Variable>

<Variable Equation="0 + (-483.5679341485652600) * [DELTA_RELACIONAMENTO_DEV12] +
(0.7246550972621539) * Delay([DELTA_EXPERTISE_DEV2],1,0) + (0.6324739520508249) *
Delay([DELTA_EXPERTISE_DEV8],1,0) + (-1.5560294844728932) * Delay([DELTA_EXPERTISE_DEV9],1,0)
+ (0.4702371565097980) * Delay([DELTA_EXPERTISE_DEV11],1,0) + (1.1798110911458926) *
Delay([DELTA_EXPERTISE_DEV20],1,0) + (0.4997524406331185) *
Delay([DELTA_EXPERTISE_DEV21],1,0) + (-3.9578518040271966) *
Delay([DELTA_EXPERTISE_DEV22],1,0) + (0.7004560880956803) *
Delay([DELTA_EXPERTISE_DEV28],1,0) + (0.2788801679885485) *
Delay([DELTA_EXPERTISE_DEV37],1,0) + (-10.2380159654153590) *
Delay([DELTA_EXPERTISE_DEV38],1,0) + (0.5937961062972358) *
Delay([DELTA_EXPERTISE_DEV39],1,0) + (0.9803530411403096) *
Delay([DELTA_EXPERTISE_DEV53],1,0) + (0.0000000000000000) *
Delay([DELTA_EXPERTISE_DEV57],1,0) + (0.9391843809596656) *
Delay([DELTA_EXPERTISE_DEV63],1,0) + (0.9305647495073754) *
Delay([DELTA_EXPERTISE_DEV70],1,0) + (1.1581155072572051) *
Delay([DELTA_EXPERTISE_DEV80],1,0) + (0.3990137164588886) *
Delay([DELTA_EXPERTISE_DEV81],1,0) + (1.2242065496615235) *
Delay([DELTA_EXPERTISE_DEV82],1,0) + (0.8442486775755575) *
Delay([DELTA_EXPERTISE_DEV89],1,0) + (1.0468159029618938) *
Delay([DELTA_EXPERTISE_DEV94],1,0) " id="351" name = "DELTA_EXPERTISE_DEV12"> </Variable>

<Variable Equation="([COEF_OPINIAO] * [OPINIAO_DEV12] + [COEF_COMPORTAMENTO] *
[COMPORTAMENTO_DEV12] + [COEF_SIMILARIDADE] * [SIMILARIDADE_DEV12] +
[COEF_RELACIONAMENTO] * [RELACIONAMENTO_DEV12] + [COEF_EXPERTISE] *
[EXPERTISE_DEV12] + [COEF_CONFIABILIDADE] * [CONFIABILIDADE_DEV12]) / ([COEF_OPINIAO] +
[COEF_COMPORTAMENTO] + [COEF_SIMILARIDADE] + [COEF_RELACIONAMENTO] +
[COEF_EXPERTISE] + [COEF_CONFIABILIDADE])" id="380" name = "REPUTACAO_DEV12"> </Variable>
  ##################################
  RATEs
  ##################################
<Flow id="359" name="RATE_TEMPO_DEV12" FlowRate="1"> </Flow>

<Flow id="360" name="RATE_OPINIAO_DEV12" FlowRate="[DELTA_OPINIAO_DEV12]"> </Flow>
<Flow id="361" name="RATE_COMPORTAMENTO_DEV12" FlowRate =

```
"[DELTA_COMPORTAMENTO_DEV12]" > </Flow>
<Flow id="362" name="RATE_CONFIABILIDADE_DEV12" FlowRate =
"[DELTA_CONFIABILIDADE_DEV12]" > </Flow>
<Flow id="363" name="RATE_SIMILARIDADE_DEV12"  FlowRate = "[DELTA_SIMILARIDADE_DEV12]">
</Flow>

<Flow id="364" name="RATE_RELACIONAMENTO_DEV12" FlowRate =
"[DELTA_RELACIONAMENTO_DEV12]" > </Flow>

<Flow id="365" name="RATE_EXPERTISE_DEV12" FlowRate="[DELTA_EXPERTISE_DEV12]"> </Flow>
```

A primeira parte representa o elemento de configuração da simulação e define uma constante *TimeStep* como 1, que estabelece o incremento de tempo utilizado no processo de simulação. Além disso, também define uma constante *TimeLength* que estabelece o número de iterações, ou no caso, as ocorrências a serem simuladas.

Em seguida, é definido um STOCK (estoque) para cada critério de reputação, com os valores iniciais de cada um dos estoques utilizados. Cada estoque é definido com um dos critérios de reputação do modelo, considerando o valor inicial como os valores de cada critério, coletados na data da última ocorrência de solicitação de mudança, do período em observação. O STOCK *Tempo* armazena o acúmulo de iterações desde o início do processo de simulação.

Na sequência, são apresentadas as equações construídas através de regressão linear, definidas como uma VARIABLE (variável). Observa-se a utilização de uma variável para cada critério prefixada por DELTA, evidenciando a variação ocorrida para aquele critério em uma determinada iteração no processo de simulação. É possível notar nas equações que critérios de outros desenvolvedores podem influenciar o valor de critérios do desenvolvedor em análise (influências interdesenvolvedor). Nesse caso, também é utilizada a regressão linear entre os critérios que se relacionam. A função *Delay* é utilizada para estabelecer um atraso na coleta do valor, dos critérios de outros desenvolvedores, apenas para garantir que o valor considerado seja um valor histórico, já calculado.

Por fim, é definida uma RATE (taxa) para cada equação, que efetivamente faz a transferência dos valores para os estoques, em função da taxa (*FlowRate*) definida em cada uma das equações (DELTA). Cada RATE foi construída de forma a não permitir que os critérios de reputação atinjam valores negativos, os quais não fariam sentido para os critérios considerados.