

LEONARDO ALEJANDRO GUERRA OTERO

**Análise da transformação de histórias de usuário em modelos i* em
um sistema real**

Monografia apresentada ao PECE –
Programa de Educação Continuada em
Engenharia da Escola Politécnica da
Universidade de São Paulo como parte
dos requisitos para conclusão do curso de
MBA em Tecnologia de Software.

São Paulo
2016

LEONARDO ALEJANDRO GUERRA OTERO

**Análise da transformação de histórias de usuário em modelos i* em
um sistema real**

Monografia apresentada ao PECE –
Programa de Educação Continuada em
Engenharia da Escola Politécnica da
Universidade de São Paulo como parte
dos requisitos para a conclusão do curso
de MBA em Tecnologia de Software.

Área de Concentração: Tecnologia de
Software

Orientador: Prof. Dr. Fábio Levy Siqueira

São Paulo
2016

FICHA CATALOGRÁFICA

DEDICATÓRIA

*Dedico este trabalho a todos aqueles
que me auxiliaram na sua
concretização em sua plenitude.*

AGRADECIMENTOS

À Universidade de São Paulo – USP que possibilitou a realização deste trabalho e a ampliação de meus horizontes.

À Escola Politécnica da Universidade de São Paulo – EPUSP que proporcionou todo o conhecimento necessário para a idealização e realização deste trabalho.

Ao PECE – Programa de Educação Continuada em Engenharia que permitiu que eu pudesse ter a oportunidade de melhorar meu conhecimento a respeito da minha área de atuação.

Aos meus pais pelo apoio à busca eterna pelo conhecimento e aprimoramento, pessoal e intelectual.

RESUMO

No atual cenário do mundo corporativo, os sistemas de informação desempenham um papel essencial nos processos de uma empresa. Eles permeiam praticamente todos os setores de uma empresa, automatizando processos manuais e tornando-os mais eficientes.

As constantes mudanças nas necessidades das empresas tornou a construção de software mais custosa uma vez que no processo de desenvolvimento clássico, todos os requisitos são definidos no começo do projeto e, portanto, para cada alteração de uma necessidade dentro do projeto há um custo alto.

Para minimizar os custos de uma alteração de uma necessidade de um projeto, foi feito o Manifesto Ágil, que propõe uma nova abordagem de desenvolvimento de software, que tem como pontos fortes a flexibilidade e o *feedback* instantâneo, o que o torna mais adequado às necessidades do atual cenário corporativo.

Entretanto, um dos desafios da utilização dessas abordagens é a documentação mínima, o que torna a análise do projeto como um todo mais difícil. Com base nesses fatos, Jaqueira (2013) definiu um método de transformação de histórias de usuário em modelos i^* , com o intuito de enriquecer os requisitos em um processo ágil. O presente trabalho apresenta uma análise a respeito desse método utilizando um cenário real para verificar a eficiência do mesmo em projetos maiores e mais complexos.

ABSTRACT

In the current business scenario, information systems play an important role in the processes of a company. They practically permeate throughout all the parts of the company, automating manual processes and making them more efficient.

The ever-changing needs of companies have made the software construction more expensive, because in the traditional development process all requirements are set at the beginning of the project, and therefore for each change of a need within the project there is a high cost.

To minimize the costs of changing a need for a project, the Agile Manifest was created. It proposes a new approach to software development, which has the advantage of flexibility and instant feedback, making it more suitable to the needs of today's business scenario.

However, one of the challenges of using this method is the minimum documentation, what makes the project analysis more difficult. Based on these facts, Jaqueira (2013) defines a method to transform user stories into i * models, in order to enrich the requirements in an agile process. This paper presents an analysis of this method using a real scenario to check its efficiency in a larger and more complex projects.

LISTA DE ILUSTRAÇÕES

Pág.

Figura 1 – Dois atores, A1 e A2, com uma dependência entre eles, O1.....	14
Figura 2 – Exemplo de modelo com grau de importância de objetivos.....	16
Figura 3 – Tarefa T1 atinge o objetivo O1 e, conseqüentemente, a meta- <i>soft</i> é influenciada positivamente.....	18
Figura 4 – Conjunto de heurísticas que definem o método de transformação para o modelo SD.....	27
Figura 5 – Conjunto de heurísticas que definem o método de transformação para o modelo SR.....	28
Figura 6 – Diagrama resultante da aplicação do método na história 09, da tabela 1.....	29
Figura 7 - Diagrama do modelo SD resultante das histórias	30
Figura 8 - Diagrama do modelo SR resultante das histórias.....	31
Figura 9 - Diagrama BPMN descrevendo o sistema Certificados.....	35
Figura 10 - Exemplo de sub modelo do sistema Certificados.....	40

LISTA DE TABELAS

Pág.

Tabela 1 – Histórias de usuário do sistema de login.....	30
Tabela 2 – Comparação entre o estudo de caso e o sistema Certificados.....	34
Tabela 3 – Grupo de histórias de usuário com dificuldade por causa da granularidade.....	37

LISTA DE ABREVIATURAS E SIGLAS

XP	Extreme Programming
BPMN	Business Process Model and Notation
SD	<i>Strategic Dependency Model</i>
SR	<i>Strategic Rationale Model</i>

SUMÁRIO

	Pág.
1. INTRODUÇÃO	10
1.1. Objetivo	11
1.2. Justificativas	11
1.3. Estrutura do Trabalho	12
2. REVISÃO BIBLIOGRÁFICA	13
2.1.1. Modelo de Dependência Estratégica	14
2.1.2. Modelo de Racional Estratégico	17
2.1.2.1. Arestas de decomposição de tarefas	17
2.1.2.2. Arestas do tipo <i>Means-Ends</i>	18
2.2. Histórias de Usuário	19
2.2.1. Modelo de história de usuário	20
2.2.2. Nível de Detalhe	21
2.2.3. INVEST	22
2.2.4. Requisitos não-funcionais	23
2.2.5. Criando histórias de usuário.....	24
2.3. Considerações do Capítulo	24
3. TRANSFORMAÇÃO DE HISTÓRIAS DO USUÁRIO EM UM MODELO DE METAS.....	26
3.1. Método de transformação	27
3.3. Discussão	31
4. APLICAÇÃO DA TRANSFORMAÇÃO EM UM SISTEMA REAL	33
4.1. Descrição do Sistema Certificados	35
4.2. Aplicação do método	36
4.3. Discussão	39
5. CONSIDERAÇÕES FINAIS	42
5.1. Contribuições do Trabalho.....	42
5.2. Trabalhos Futuros.....	44

REFERÊNCIAS.....	45
APÊNDICE A – Histórias de usuário do Sistema Certificados	47
APÊNDICE B – Modelo SD obtido para o Sistema Certificados.....	50
APÊNDICE C – Modelo SR obtido para o Sistema Certificados.....	51
APÊNDICE D – Exemplo de sub modelo SR para o Sistema Certificados	52

1. INTRODUÇÃO

No *Extreme Programming* (XP), um dos principais processos de desenvolvimento ágil, não se sugere tanta documentação como as existentes em processos dirigidos a planos, como o Processo Unificado. Esse processo indica que os únicos artefatos necessários para a construção de um software são o código e os testes, além das histórias de usuário (BECK, 2000).

As histórias de usuário são artefatos simples e sucintos em sua concepção, mas eles não contêm informações a respeito das relações com outras histórias, o que prejudica a visualização dessas interdependências.

Observando a necessidade de uma documentação mais precisa e objetiva, que dê uma visão mais ampla das necessidades de negócio e das dependências entre histórias, em Jaqueira (2013) foi proposto um método para transformar histórias de usuário em modelos do *framework* i*. Esse *framework* foi proposto por Yu (1995) e é um modelo orientado à metas, centrado nos *stakeholders* e seus relacionamentos, expressando quais são as dependências entre eles para que os objetivos sejam atingidos.

O método proposto define uma série de heurísticas cujo objetivo é transformar os elementos de uma história de usuário em elementos do *framework* i*. Com isso, atinge-se uma compreensão de como as necessidades do usuário, representadas por objetivos no modelo i*, estão sendo construídas e, principalmente, quais suas dependências.

Com o objetivo de analisar se o processo proposto por Jaqueira (2013) se adéqua a cenários mais próximos da realidade, uma vez que o estudo de caso conduzido por ela é simples se comparado com sistemas reais (tanto em tamanho

como em complexidade), o presente trabalho irá analisar o método utilizando um conjunto de histórias de um sistema real em produção.

1.1. Objetivo

O objetivo deste trabalho é analisar o método proposto por Jaqueira (2013) usando com um conjunto de histórias de usuário mais numeroso e complexo, referentes a um sistema de emissão de certificados de uma seguradora.

Serão avaliados os benefícios e os problemas do método proposto considerando esse cenário e se as heurísticas criadas contemplam o escopo do projeto supracitado.

1.2. Justificativas

O estudo de caso realizado em Jaqueira (2013) utiliza um conjunto de histórias de usuário para um software de pequeno porte e que, portanto, pode não se adequar às necessidades de um projeto maior e mais abrangente.

Dentre as conclusões do trabalho de Jaqueira (2013), existe a preocupação com relação à aplicabilidade do método em sistemas maiores e mais complexos, pois poderia gerar uma dificuldade ou uma confusão na leitura e compreensão dos modelos gerados pela aplicação do método.

Dessa forma, faz-se necessário uma verificação mais detalhada do método proposto para sistemas com essas características, o que pode ser considerado mais próximo de uma situação real.

1.3. Estrutura do Trabalho

O Capítulo 1 INTRODUÇÃO apresenta as motivações, o objetivo, as justificativas e a estrutura do trabalho.

O Capítulo 2 REVISÃO BIBLIOGRÁFICA apresentará a fundamentação teórica dos assuntos relevantes ao trabalho, apresentando a técnica de histórias de usuário e o framework i*.

O Capítulo 3 TRANSFORMAÇÃO DE HISTÓRIAS DO USUÁRIO EM UM MODELO DE METAS apresenta o processo proposto por Jaqueira (2012), explicando seu funcionamento e um exemplo de aplicação.

O Capítulo 4 APLICAÇÃO DA TRANSFORMAÇÃO EM UM SISTEMA REAL descreve a aplicação do método no conjunto de histórias de usuário do sistema Certificados e a discussão dos resultados obtidos.

O Capítulo 5 CONSIDERAÇÕES FINAIS descreve a conclusão do trabalho e apresentará os trabalhos futuros.

REFERÊNCIAS relaciona todas as referências utilizadas por este trabalho.

APÊNDICE A apresenta as histórias de usuário utilizadas no capítulo 4 para fazer a aplicação do método de transformação.

APÊNDICE B apresenta o modelo SD gerado a partir da execução do método de transformação.

APÊNDICE C apresenta o modelo SR gerado a partir da execução do método de transformação.

APÊNDICE D apresenta exemplos de submodelos SR a partir da discussão do capítulo 4.

2. REVISÃO BIBLIOGRÁFICA

Nas próximas subseções serão apresentados os conceitos necessários para o entendimento do método proposto por Jaqueira (2013). Primeiramente será apresentado o *framework* i*, descrevendo os seus dois modelos, Modelo de Dependência Estratégica (SD) e Modelo de Racional Estratégico (SR), e em seguida a definição de histórias de usuário.

2.1. Framework i*

Segundo Yu (1995), o *framework* i* tem como objetivo prover uma visão mais profunda do domínio de um determinado problema, com ênfase nas motivações, nas intenções e na análise racional do processo dos envolvidos.

Em modelos como por exemplo de processo de negócio, um processo é descrito através de um fluxo de atividades intercaladas com entradas e saídas. A visão proposta por Yu (1995) permite criar modelos mais ricos, mostrando não somente quais as atividades, mas também o porquê dessas atividades serem executadas.

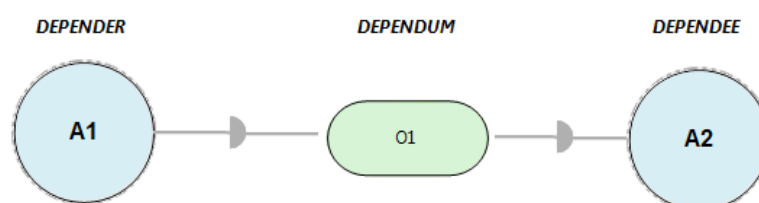
Para isso o *framework* primeiramente modela as dependências entre os atores, que nesse caso são todos os envolvidos no processo, tanto pessoas como sistemas e depois modela as alternativas dentro de um processo, usando o Modelo de Dependência Estratégica (SD) e o Modelo de Racional Estratégico (SR), respectivamente.

2.1.1. Modelo de Dependência Estratégica

O modelo de dependência estratégica (SD - *Strategic Dependency Model*) é um modelo que visa representar as intenções de todos os envolvidos em um processo, a fim de explicitar as dependências entre eles. As intenções e as motivações de cada envolvido, chamados de ator, são representadas nesse modelo, o que permite analisar o impacto de mudanças nas dependências de forma mais ampla, analisar oportunidades e vulnerabilidades e reconhecer padrões de relacionamento (YU, 1995).

O modelo se caracteriza como um grafo, em que os nós são atores e as arestas indicam as dependências para alcançar algum objetivo. Por exemplo, na figura 1 o ator A1 necessita, ou depende do ator A2 para alcançar um objetivo O1. Dessa forma, utilizando a nomenclatura do modelo, A1 é chamado de *dependor*, A2 é chamado de *dependee*, e O1 é chamado de *dependum*. Essa dependência explicita uma vulnerabilidade do processo, pois caso o ator A2 não consiga entregar, ou entregar com um grau de aceitação menor que o necessário, o ator A1 não será capaz de cumprir sua tarefa a contento.

Figura 1: Dois atores, A1 e A2, com uma dependência entre eles, O1. Fonte: o autor



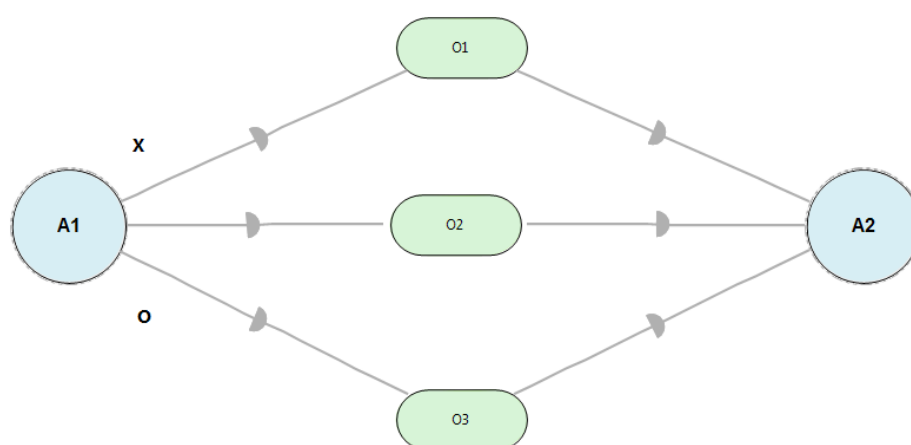
Existem vários tipos de *dependum* no modelo SD, que caracterizam a natureza da dependência entre dois atores: objetivo (*hardgoal*), tarefa, meta-soft (*softgoal*) e recurso. A seguir são apresentados cada um deles, seguindo Yu (1995):

- **Objetivo:** Indica uma dependência na qual um ator (*dependor*) tem a necessidade que outro ator (*dependee*) faça com que o sistema alcance um determinado estado, exigindo alguma ação do *dependee* de forma a alcançar esse estado. Portanto, o *dependor* pode assumir que aquele determinado estado se manterá dessa forma. É importante frisar que o objetivo deve estar especificado, mas a forma a alcançar esse objetivo não. A meta cria uma vulnerabilidade para o *dependor*, uma vez que a responsabilidade de manter o estado desejado é do *dependee*. Dessa forma, ao criar o modelo e perceber que um determinado ator tem dependências demais, pode-se modificar a forma como esse ator obtém suas dependências. Isso também se aplica aos demais tipos de dependências.
- **Meta-soft:** Da mesma forma que o objetivo, consiste em um estado ou condição que o negócio deve estar para que a necessidade ou vontade do ator seja satisfeita, mas difere no que diz respeito aos critérios para a satisfação que não são claros ou objetivos.
- **Tarefa:** Indica uma dependência na qual o ator espera que outro ator cumpra uma determinada atividade, que ao contrário da meta-soft e do objetivo, especifica a forma como fazer e não o motivo de fazer. Também é possível perceber que se criou uma vulnerabilidade para o *dependor*, uma vez que a responsabilidade de execução da tarefa é do *dependee*.
- **Recurso:** Indica uma dependência em que o ator espera que um recurso, tanto físico como uma informação, esteja disponível. Da mesma forma que as outras, também cria-se uma vulnerabilidade, já que o recurso em questão pode ficar indisponível, se esgotando, por exemplo, caso seja um recurso físico.

O modelo também define uma forma de explicitar o grau de importância de uma dependência a partir da importância que aquela tarefa, recurso ou meta-*soft* tenha no processo da atividade:

- **Aberta:** em caso de falha ao obter o *dependum*, a atividade será afetada, mas não drasticamente. É representada como um círculo na dependência do lado do *dependum*. Na figura 2, a relação A1-O3-A2 é do tipo aberta.
- **Compromissada:** em caso de falha ao obter o *dependum*, a atividade será afetada de forma significativa, com grandes chances de falha. Não tem representação específica no modelo e, portanto, a não ser que seja aberta ou crítica, a dependência é comprometida. Na figura 2, a relação A1-O2-A2 é do tipo aberta.
- **Crítica:** em caso de falha ao obter o *dependum*, a atividade vai falhar. É representada como um "X" do lado do *dependum*. Na figura 2, a relação A1-O1-A2 é do tipo crítica.

Figura 2: Exemplo de modelo com grau de importância de objetivos. Fonte: o autor



2.1.2. Modelo de Racional Estratégico

O modelo de racional estratégico (SR - *Strategic Rationale Model*) é um modelo que permite representar os componentes internos de um processo, descrevendo relações que o compõem.

A diferença principal entre o modelo SR e o SD consiste na diferença no nível de abstração dos componentes: SD visa detalhar os componentes de forma a exibir as dependências externas do ator, enquanto o SR foca nesse ator e exibe os componentes internos, usados para concluir aquela atividade. Tendo o conhecimento do negócio e do processo, este modelo permite propor novos processos e avaliar sua viabilidade.

O modelo, da mesma forma que o modelo SD, é um grafo em que os nós podem ser objetivos, tarefas, recursos e metas-*soft*, os mesmos usados no modelo SD, e as arestas podem representar meios para alcançar um determinado resultado (*means-end*) ou que uma decomposição de tarefas (*task- decomposition*).

2.1.2.1. Arestas de decomposição de tarefas

Uma tarefa é definida como o conjunto de seus componentes. Esses componentes podem ser um objetivo, uma tarefa, um recurso ou uma meta-*soft*.

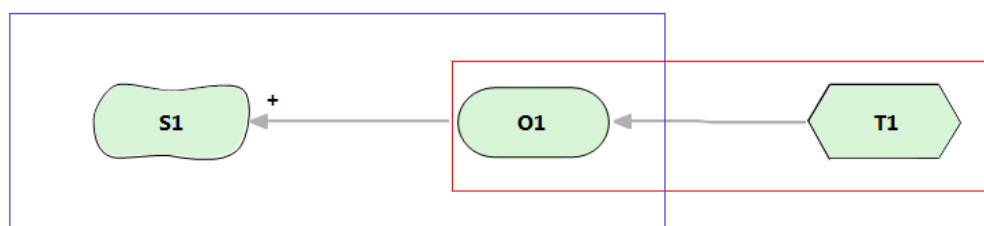
Com relação ao grau de importância de uma dependência, todos os nós podem ser *compromissados* ou *abertos*, da mesma forma que no modelo SD, e apenas as dependências externas podem ser *críticas*.

2.1.2.2. Arestas do tipo *Means-Ends*

As arestas que representam um meio para um resultado são uma ligação entre um determinado resultado, como um objetivo a ser alcançado, um recurso a ser produzido, uma tarefa a ser cumprida ou uma meta-*soft* a ser satisfeita, e os meios para alcançá-lo, normalmente sendo representados por tarefas, já que elas explicitam o como fazer e não o porquê. No modelo gráfico a aresta é direcionada, indo dos meios para o resultado.

Na figura 3, há dois exemplos de arestas *means-ends*: entre a tarefa T1 e o objetivo O1 circulos de vermelho, onde a tarefa T1 é o meio para alcançar o objetivo O1, e uma relação entre o objetivo O1 e a meta-*soft* S1 circulos de azul, onde o objetivo O1 é o meio para alcançar a meta-*soft* S1. Um detalhe importante é que o objetivo O1 é tanto um meio de alcançar um resultado como também um resultado.

Figura 3 - Tarefa T1 atinge o objetivo O1 e, consequentemente, a meta-*soft* é influenciada positivamente. Fonte: o autor.



Tanto arestas que resultam em uma meta ou recurso, se originam de uma tarefa e finalizam em uma meta e recurso respectivamente. No caso de arestas onde uma tarefa resulta em uma meta-*soft*, deve-se adicionar um marcador, que indica se aquela tarefa contribui positivamente ou negativamente para a meta-*soft*. Também é

possível ligar uma meta-*soft* em outra através de uma aresta *means-ends*, indicando que uma meta-*soft* contribui positivamente ou negativamente para outra meta-*soft*.

O modelo também permite uma ligação onde ambas as partes, meios e resultado, são tarefas. Esses casos são comuns em casos de especialização ou como uma alternativa entre duas tarefas.

Finalmente existem arestas onde ambas as partes, meios e resultado, são objetivos, como, por exemplo, em que um conjunto de objetivos menores resulta em um objetivo mais amplo, sem necessariamente ter componentes entre eles.

2.2. Histórias de Usuário

As metodologias ágeis buscam simplificar o processo de desenvolvimento, tornando-o mais objetivo e mais flexível às mudanças. Para isso torna o processo menos burocrático e a resposta à mudança mais rápida (MANIFESTO, 2001).

No *Extreme Programming* (XP) e em muitos projetos aplicando o *Scrum* utiliza-se de um artefato para explicitar quais são as *features* a serem desenvolvidas, chamadas de histórias de usuário.

Uma das grandes diferenças entre as metodologias ágeis, como XP e *Scrum*, e as metodologias orientadas a planos, como o Processo Unificado, é com relação à elicitação dos requisitos do software. Enquanto as metodologias orientadas a planos elicitem o requisito por completo no começo da iteração (ou mesmo em iterações anteriores), com toda a informação necessária para o desenvolvimento e testes de uma determinada *feature*, algumas metodologias ágeis ditam que seja criado um artefato chamado de *product backlog*, que nada mais é que um conjunto de itens que representam as necessidades que irão agregar efetivamente um valor para o negócio.

Inicialmente, esses itens são bastante amplos e pouco detalhados. Conforme cada item seja necessário para o uso do time de desenvolvimento, ele será refinado e novos itens serão criados, os quais são mais bem definidos e com escopo menor, em oposição aos itens iniciais. Uma das formas de se representar esses itens é através de histórias de usuário, ou *user stories*, que representam uma necessidade de negócio, principalmente funcionalidades, que serão atendidas por um sistema a ser construído (RUBIN, 2012).

As histórias devem ser construídas a fim de serem simples e concisas, e em que ambas as partes, tanto de negócio como técnica, tenham o mesmo entendimento. Elas podem ter vários níveis de granularidade, sendo facilmente refinada para níveis de menor granularidade.

2.2.1. Modelo de história de usuário

Um dos modelos que vem sendo amplamente utilizado é o formato descrito por Cohn (2006). Ele descreve a história no seguinte formato: "eu como <PAPEL>, quero <AÇÃO> para <META>", escrita em linguagem natural pelo próprio usuário. O elemento <PAPEL> refere-se ao tipo de usuário ou de função do usuário dentro do sistema. O elemento <AÇÃO> representa qual a funcionalidade que aquele tipo de usuário deseja efetuar dentro do sistema. Finalmente, o elemento <META> descreve qual o objetivo que aquele tipo de usuário deseja atingir ao efetuar a <AÇÃO>.

Segundo Jeffries (2001), é possível descrever uma história de usuário como 3 C's: cartão, conversa e confirmação.

- **Cartão:** diz respeito ao artefato físico em si, onde a equipe escreve a história de usuário a ser executada. É importante esclarecer que não

existe espaço físico para grandes explicações no cartão e essa limitação é intencional, para promover a discussão daquela história. Segundo Rubin (2012), os cartões contêm o título da história, o que deve ser feito, para qual finalidade e para qual *stakeholder* ela tem valor de negócio.

- **Conversação:** diz respeito à discussão em si que o cartão promove. Essa discussão é feita pela equipe de desenvolvimento, o *product owner* (que, grosso modo, representa o cliente) e o especialista do negócio com o objetivo de alinhar o conhecimento de todos os envolvidos. Nessa discussão as histórias são detalhadas e muitas vezes divididas em histórias menores.
- **Confirmação:** diz respeito aos critérios de aceitação daquela história. O time de desenvolvimento a utiliza para saber qual o comportamento desejado e o *product owner* utiliza para confirmar se a história foi implementada corretamente.

2.2.2. Nível de Detalhe

Conforme RUBIN (2012), as histórias de usuário podem ter vários níveis de detalhe (ou granularidade). Esses níveis têm como objetivo prover um ajuste melhor das histórias de usuário dentro do processo de desenvolvimento.

Histórias mais amplas, conhecidas como histórias épicas, são boas para o planejamento de várias iterações de desenvolvimento pois mostram em alto nível quais são as próximas *features* a serem implementadas, evidenciando qual o "caminho" que o produto em questão seguirá. Em termos de tempo, normalmente contempla meses de desenvolvimento e são maiores que um *release* (RUBIN, 2012).

Por sua vez, histórias épicas são compostas por um conjunto de histórias menores e mais detalhadas, que normalmente contemplam uma feature inteira, levando várias iterações para ser concluída. Em termos de tempo, normalmente levam semanas e são maiores que uma iteração.

No nível de iteração, tem-se as histórias que já tem um detalhamento maior e normalmente levam dias para serem finalizadas.

O mais importante no que diz respeito ao nível de detalhe, é justamente o processo de refinamento que constrói histórias menores a partir de uma história maior e torna as histórias cada vez mais detalhadas.

Dessa forma, é possível perceber que a granularidade das histórias deve ser feita de acordo com o processo de desenvolvimento, ou seja, respeitando cada uma das fases para criar e refinar histórias que atendam as necessidades da fase vigente.

2.2.3. INVEST

Conforme supracitado, uma história de usuário deve possuir certas características para que ela possa ser utilizada dentro do processo de desenvolvimento. Para avaliar se uma história possui essas características, Wake (2003) definiu um conjunto de 6 critérios, representados pelo acrônimo INVEST (RUBIN, 2012):

- **Independente (*Independent*)**: a história deve ser independente, ou no máximo, ser pouco acoplada a outras histórias, pois a dependência torna o planejamento, estimativa e priorização mais difíceis;

- **Negociável (*Negotiable*):** a história deve ter origem em comum acordo entre o *product owner* e a equipe de desenvolvimento, pois isso torna mais flexível a escolha das histórias para a iteração;
- **Valiosa (*Valuable*):** a história deve agregar valor para o *product owner* ou para o usuário final do sistema, senão não existe motivo para implementá-la;
- **Estimável (*Estimatable*):** a história deve ser passível de ser estimada em termos de esforço ou tempo, pois dessa forma o *product owner* saberá qual o custo de implementá-la, e assim priorizá-la de acordo com o esforço necessário para completá-la;
- **Pequena (*Small*):** a história deve ser adequada à fase do processo de desenvolvimento, ou seja, para cada fase do desenvolvimento a história deve ser mais ou menos detalhada para fazer-se uso dela. Além disso, com histórias pequenas, é possível tomar ações corretivas em tempo suficiente para minimizar o impacto de atrasos dentro de uma iteração; e
- **Testável (*Testable*):** a história deve ser testável, cumprindo os critérios de aceitação discutidos anteriormente, para que seja possível determinar se foi finalizada ou não.

2.2.4. Requisitos não funcionais

Requisitos não funcionais são restrições em nível de sistema e podem ser representados por histórias de usuário da mesma maneira que requisitos funcionais. Entretanto, podem ter um formato mais adequado às suas características, podendo não conter todos os elementos de uma história de usuário (BECK, 2001).

Requisitos não funcionais influenciam outras histórias de usuário. Isso altera a forma como determinar se uma história está finalizada, ou seja, para ser dada como finalizada uma história deve ser testada tanto em seus requisitos funcionais como também os não funcionais.

2.2.5. Criando histórias de usuário

A criação de histórias de usuário pode ser feita de várias maneiras. Entre elas duas se destacam: *workshop* de histórias de usuário e o mapeamento de histórias (RUBIN 2012).

Workshop de histórias de usuário é feito de forma a criar uma discussão em torno do que o software ou serviço deveria fazer, com o intuito de gerar um conjunto inicial de histórias para que o processo possa começar. Deve-se evitar criar todas as histórias nesse primeiro momento, pois ainda não se tem todo o contexto necessário do domínio de negócio.

O processo de mapeamento de histórias consiste em decompor histórias de alto nível em histórias menores de forma a destacar qual o fluxo do processo que o sistema ou serviço deve efetuar.

2.3. Considerações do Capítulo

As histórias de usuário são artefatos simples e que contém pouca informação a respeito da *feature* em si, pois elas devem ser sucintas e fazer com que um canal de comunicação entre os desenvolvedores e os usuários de negócio seja criado, para assim elicitar os requisitos daquela *feature*. Além disso, segundo BECK (2000), os únicos artefatos a serem produzidos pelo XP, que é uma das principais

metodologias ágeis, são o código e os testes, ou seja, não existe um documento formalmente construído para descrever as *features* do sistema.

Outro ponto a ser levado em consideração é a dificuldade de visualizar as dependências entre histórias de usuários, pois elas não contêm informação suficiente para fazer essa análise. Quando o sistema é pequeno essas dependências são perceptíveis, porém quando o sistema aumenta de tamanho e escopo, as dependências se tornam mais difíceis de serem observadas.

Em contrapartida, modelos orientados a metas, como o i*, tem como característica prover modelos que explicitam as dependências entre os objetivos dos diversos *stakeholders* (VAN LAMSWEERDE, 2001), o que torna a rastreabilidade mais simples.

Dessa forma uma transformação de histórias de usuário em modelos i* tornaria possível explicitar as dependências entre elas e assim melhorar o planejamento da execução das mesmas.

3. TRANSFORMAÇÃO DE HISTÓRIAS DO USUÁRIO EM UM MODELO DE METAS

A proposta feita por Jaqueira (2013) tem como objetivo transformar histórias de usuário em modelos *i** para assim prover uma visão gráfica das relações entre histórias de usuário. As histórias de usuários são artefatos simples e sucintos, que não expressam suas relações com outras histórias de usuário. Já os modelos do *i** permitem visualizar as dependências entre requisitos.

Essa visualização provida pelos modelos *i** permite visualizar as dependências entre as histórias de usuário e assim determinar com mais facilidade, já que se trata de um modelo visual, o contexto em que essas histórias estão envolvidas, permitindo um entendimento mais profundo do domínio do problema. Esses modelos ainda explicitam quais são os *stakeholders* que estão de alguma forma ligados às histórias de usuário e assim entender quais os objetivos e metas de cada um deles.

Assim o uso de modelos *i** dentro de um projeto que utiliza histórias de usuário, pode melhorar o entendimento do sistema a ser desenvolvido, pois considera os objetivos dos *stakeholders* e suas inter-relações, pois os modelos gerados explicitam as dependências entre requisitos expressos em histórias, além de permitir o acesso mais fácil e rápido à informação dos requisitos.

Além disso, os modelos gerados podem ser utilizados como forma de documentação para análise e discussão a respeito dos requisitos, em adição às histórias de usuário.

Por outro lado, a criação de um novo modelo torna o processo mais burocrático e demorado, uma vez que existe a necessidade de criação e manutenção desses modelos. Se levarmos em consideração sistemas maiores e

mais complexos, essa dificuldade se torna ainda maior, pois o modelo gerado pelo processo terá um grande número de informações, o que pode dificultar seu entendimento.

3.1. Método de transformação

O processo de transformação proposto por Jaqueira (2013) consiste em mapear cada elemento de uma história de usuário em um elemento i^* . É considerado para o processo de transformação o formato de história de usuário proposto por Cohn (2006): "eu como <PAPEL> quero <AÇÃO> para <META>", discutido na seção 2.2.1

A transformação é baseada na similaridade entre os elementos de uma história de usuário e os elementos do modelo i^* , criando uma correspondência entre eles: o elemento "papel" da história de usuário tem características muito similares ao "ator" dentro do modelo i^* , da mesma forma que os pares "ação" e "tarefa" e "meta" e "objetivo" (Jaqueira, 2013).

Ainda segundo Jaqueira (2013), com objetivo de simplificar o processo de transformação, foram criadas heurísticas para efetuar o processo para o modelo SD, descritas na Figura 4.

Figura 4 - Conjunto de heurísticas que definem o método de transformação para o modelo SD. Fonte: (JAQUEIRA, 2013)

- SD-H1: Criar o ator Sistema;
- SD-H2: Criar um ator no modelo i^* para cada papel diferente nas histórias de usuário;
- SD-H3: Criar uma meta no modelo i^* para cada meta nas histórias de

usuário;

- SD-H4: Se houver metas repetidas para atores diferentes, criar um ator genérico;
- SD-H4.1: Criar um relacionamento IS_A do ator genérico para os demais atores específicos que compartilham a mesma meta e
- SD-H5: Relacionar as dependências de cada ator com suas metas.

Analisando as heurísticas supracitadas, é possível perceber que o modelo SD gerado explicita as ligações entre o sistema a ser desenvolvido e os atores/papéis envolvidos. Nesse primeiro momento não são colocadas as tarefas referentes às histórias, apenas as metas e os atores ligados ao ator Sistema.

No passo seguinte, o modelo SR é criado a partir de heurísticas similares, adicionando as tarefas relacionadas nas histórias de usuário, uma vez que elas serão operacionalizadas pelo sistema. Essas heurísticas são apresentadas na Figura 5.

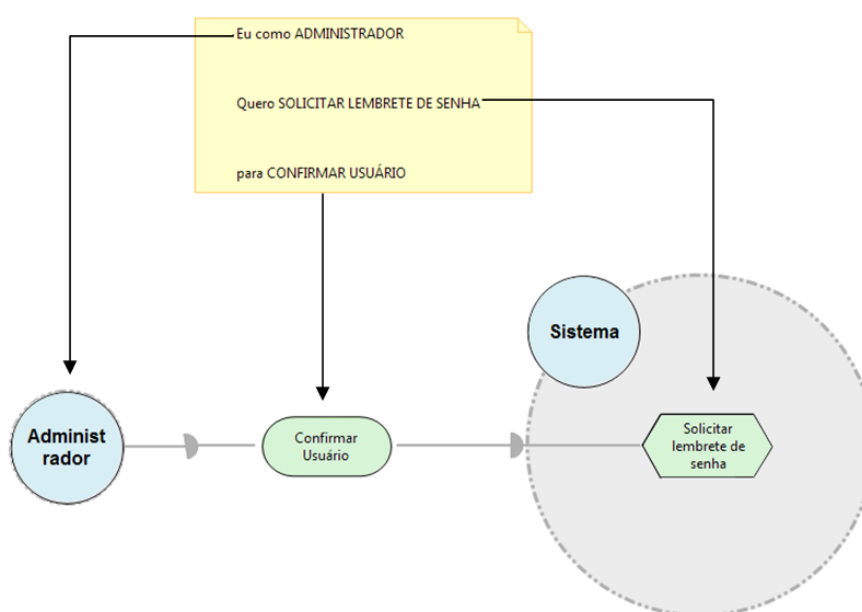
Figura 5 - Conjunto de heurísticas que definem o método de transformação para o modelo SR. Fonte: Jaqueira 2013

- SR-H1: Criar uma tarefa dentro do ator Sistema para cada ação das histórias de usuário;
- SR-H2: Se houver ações diferentes para a mesma meta, criar uma tarefa genérica;
- SR-H2.1: Decompor a tarefa genérica em sub-tarefas que representem as ações associadas à mesma meta;

- SR-H3: Relacionar as dependências de cada meta com as tarefas correspondentes de acordo com as histórias de usuário;
- SR-H4: Se houver tarefas que dependem do próprio ator a que estão relacionadas, gerar um recurso com o nome da tarefa e
- SR-H5: Relacionar o recurso criado dependendo do ator.

Para facilitar o entendimento do processo será apresentada a seguir a transformação de uma história de usuário. Na figura 6, há as correlações entre os elementos de uma história de usuário e os elementos do modelo i*.

Figura 6 - Diagrama resultante da aplicação do método na história 09, da tabela 1.
Fonte: o autor.



3.2. Exemplo de uso

A fim de exemplificar o uso do método de transformação, será apresentado um dos exemplos de uso, descrito por Jaqueira (2013). Ele contempla histórias de usuário de um sistema de login. Na Tabela 1 é apresentado um conjunto de 09 histórias de usuário desse sistema.

Tabela 1 - Histórias de usuário do sistema de login. Fonte: (JAQUEIRA, 2013).

	Papel	Ação	Meta
01	Usuário	Ter nome de usuário	Acessar conteúdo seguro
02	Usuário	Ter senha	Acessar conteúdo seguro
03	Usuário	Escolher nome de usuário	Personalizar a conta
04	Usuário	Alterar senha padrão	Personalizar senha
05	Administrador	Atribuir senha ao usuário	Registro ser automatizado
06	Administrador	Enviar e-mail de registro	Confirmar ativação da conta no e-mail
07	Administrador	Solicitar login ao usuário	Garantir segurança do conteúdo
08	Usuário	Cadastrar lembrete de senha	Lembrar a senha
09	Administrador	Solicitar a informação do lembrete de senha	Confirmar usuário

Através dessas histórias de usuário e aplicando o método proposto por Jaqueira, obtêm-se os diagramas apresentados na Figura 7 e na Figura 8.

Figura 7 - Diagrama do modelo SD resultante das histórias Fonte: o autor, com base em Jaqueira 2013.

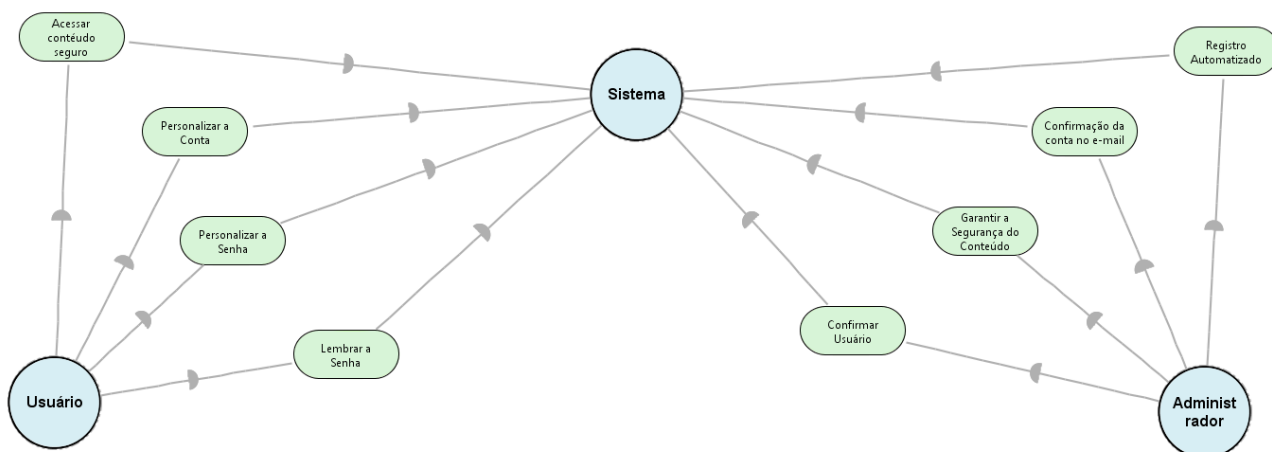
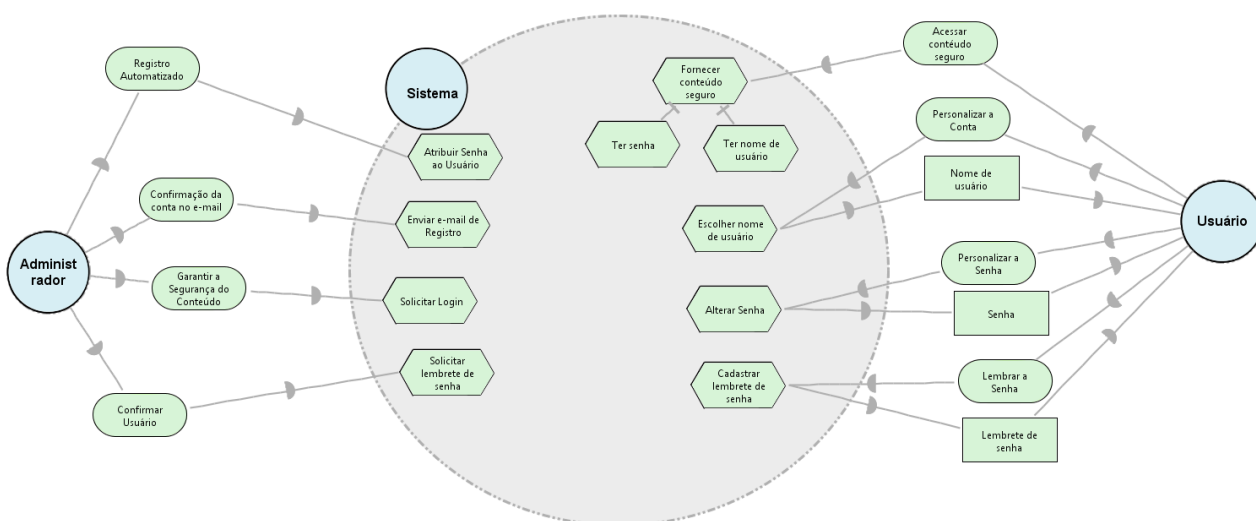


Figura 8 - Diagrama do modelo SR resultante das histórias. Fonte: o autor, com base em (JAQUEIRA, 2013).



3.3. Discussão

Conforme discutido no capítulo 2, uma dependência entre atores (um objetivo, recurso, tarefa ou meta-soft) contém uma informação sobre o seu grau de importância: crítica, compromissada ou aberta. Dessa forma, uma dependência que tenha um grau de importância crítica deve receber uma atenção maior que uma dependência de grau aberto, por exemplo.

Sendo a história de usuário um artefato sucinto, ele não contém informações sobre o valor daquela história quando comparado com outras histórias, dependendo somente do conhecimento do *stakeholder* de negócio para fazer a priorização dessas histórias. Isto não é levado em consideração pelo método proposto por Jaqueira (2013).

Outro ponto a ser levado em consideração é com relação à granularidade das histórias de usuário. Histórias utilizadas no nível de iteração se assemelham a atividades e são mais próximas à implementação e, portanto, existe um alto grau de acoplamento entre histórias. A aplicação do método nesse tipo de história cria um diagrama com muitos elementos, o que dificulta seu entendimento. Ao se utilizar de histórias, que normalmente precisam de várias iterações para serem desenvolvidas, torna o modelo gerado mais simples e objetivo, mantendo as informações sobre os objetivos de cada ator.

Além disso, nem toda história é contemplada pelo método, uma vez que é esperada que a história tenha um determinado formato. Histórias de usuário que refletem requisitos não funcionais podem não se encaixar no formato proposto por Cohn (2006). Dentro do modelo *i**, requisitos não funcionais são representados por elementos *meta-soft* e portanto, poderiam ser incluídas dentro do método com a finalidade de representar esses requisitos não funcionais no mesmo modelo que as histórias de requisitos funcionais.

4. APLICAÇÃO DA TRANSFORMAÇÃO EM UM SISTEMA REAL

Partindo do pressuposto de que em um ambiente corporativo os sistemas de informação estão cada vez maiores e mais complexos, o estudo de caso realizado por Jaqueira (2013) consiste em um conjunto de histórias de um sistema gerenciador de projetos, que totaliza 17 histórias de usuário. Através de um questionário foram analisadas questões qualitativas do método de transformação, sendo que os participantes foram divididos em dois grupos: um grupo que atuaria como engenheiros de requisitos e outro que atuaria como desenvolvedores.

Esse cenário difere de um ambiente real por ter uma quantidade de histórias pequena (apenas 17 histórias), por possuir apenas histórias simples, uma vez que o domínio do problema é de fácil entendimento e por ser bastante conhecido, por conter histórias de uma mesma granularidade e por não conter requisitos não funcionais. Dessa forma, o presente trabalho analisará o método de Jaqueira (2013) em um sistema real.

O sistema escolhido para essa análise é um sistema de geração e controle de documentos dos clientes de uma companhia de seguros, e será doravante chamado de Sistema Certificados. Esse sistema tem por objetivo centralizar a geração dos diversos documentos que o cliente, chamado de segurado, deve ter acesso ao contratar um serviço de seguro. Entretanto, para cada contrato estabelecido entre a companhia e um contratante, existe uma especificidade diferente, pois o contrato será determinado pela negociação entre a companhia e o contratante. Assim, o sistema deve gerenciar não somente os documentos e suas características, mas também a parametrização individual de cada contrato estabelecido.

O sistema teve sua construção finalizada em 6 meses, utilizando aproximadamente 1.500 horas de trabalho e processa por dia cerca de 60 mil documentos. Em termos de número de histórias de usuário, o sistema possui 72 histórias, apresentadas no Apêndice A. Essas histórias possuem várias granularidades e são complexas, sendo de um produto extremamente específico da companhia em questão e, portanto, é necessário que o *stakeholder* tenha parte desse conhecimento para entender a história. Por fim, o sistema possui 2 histórias tratando de requisitos não funcionais. Dessa forma, esse sistema permite uma melhor análise do método proposto por Jaqueira (2013) uma vez que é um ambiente real e portanto possui características mais complexas do que o ambiente utilizado por Jaqueira (2013) no seu estudo de caso.

Um ponto interessante de ser analisado é que nem todas as histórias de usuário utilizadas no processo de análise tem todas as características descritas no item 2.2.3, descritas no acrônimo INVEST. Com o intuito de utilizar as histórias de usuário reais, elas não sofreram nenhuma transformação para se adequar a essas características, justamente para ter um ambiente mais próxima da realidade possível.

Um resumo das diferenças entre o sistema de estudo de caso de Jaqueira (2013) e do sistema de Certificados é apresentado na Tabela 3.

Tabela 2 - Comparação entre o estudo de caso e o sistema Certificados.

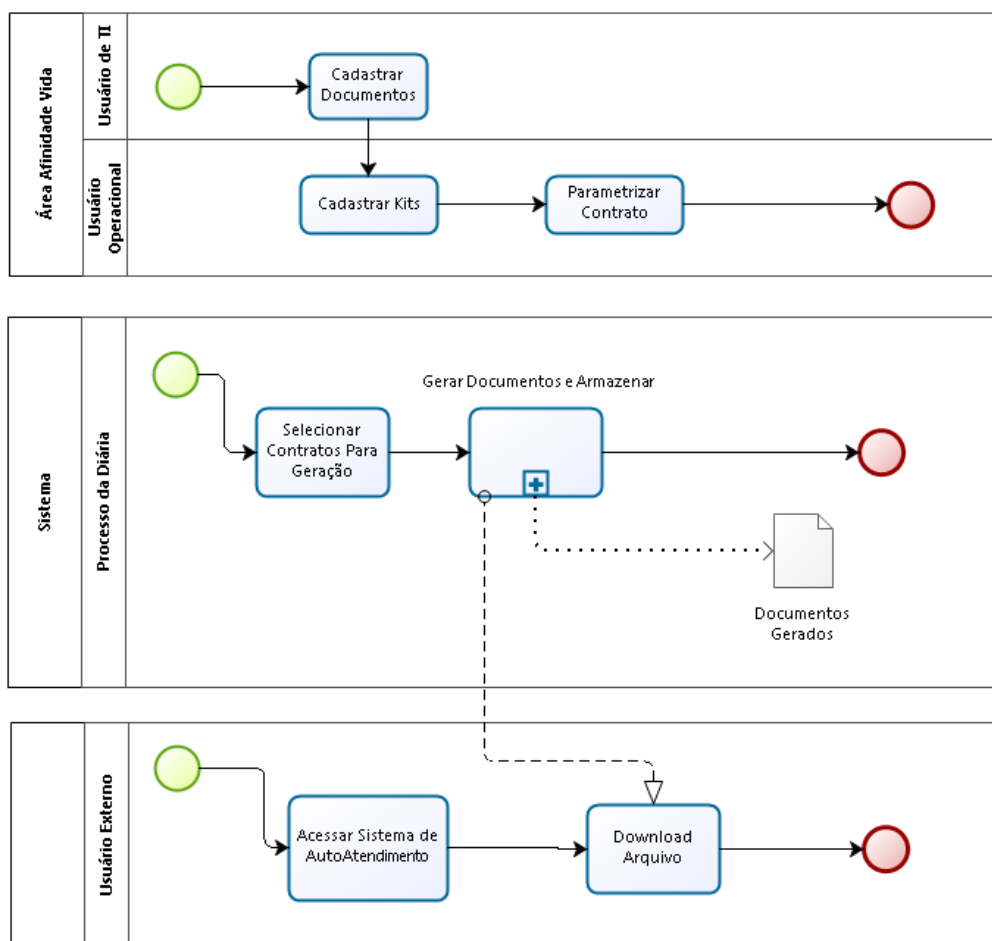
Tópico	Estudo de Caso	Sistema Certificados
Granularidade	Homogênea	Heterogênea
Complexidade	Baixa	Alta
Quantidade de histórias	17	72

História Requisitos Não funcionais	Não há	2
------------------------------------	--------	---

4.1. Descrição do Sistema Certificados

O funcionamento do sistema é apresentado na forma de um diagrama BPMN na figura 7.

Figura 9 - Diagrama BPMN descrevendo o Sistema Certificados



O usuário de TI é o responsável por fazer o cadastro dos documentos que poderão ser associados a um determinado kit. O cadastro de documentos necessita de algumas informações que serão utilizadas no armazenamento e na geração desses documentos como, por exemplo, o tempo de armazenamento que aquele

documento deve ter por uma necessidade legal. Essa informação é utilizada durante o processo da diária, que contempla o processo de geração propriamente dito.

Após o cadastro de documentos, o usuário operacional deve fazer o cadastro de kits. Um kit é um conjunto de documentos que serve a uma determinada situação dentro ciclo de vida do contrato, como adesão, movimentação ou cancelamento.

Com os kits cadastrados, o usuário operacional deve parametrizar o contrato utilizando um ou mais kits cadastrados anteriormente. É nesse processo em que as especificidades do contrato entre a companhia e o estipulante, serão definidos como, por exemplo, as coberturas e assistências contempladas pelo contrato.

Com a parametrização finalizada, o sistema Certificados, através de uma rotina *batch*, seleciona os contratos que sofreram algum dos eventos possíveis - adesão, movimentação ou cancelamento - e inicia o processo de geração e armazenamento, utilizando as características definidas na parametrização do contrato e do documento.

Após a geração dos documentos, os usuários externos - estipulante, segurado e corretor - podem acessar, através de seus respectivos portais os documentos associados a eles, cada um com uma visão específica de acordo com seu papel dentro do sistema.

4.2. Aplicação do método

A seguir será descrito a aplicação do método e a análise dos resultados com base no Sistema Certificados. Os modelos gerados pelo método estão nos Apêndices B e C.

Com relação à granularidade, o conjunto de histórias do Sistema Certificados continha histórias com diversos níveis de detalhamento (granularidade).

Isso tornou o processo de mapeamento mais complexo, pois foi necessário juntar várias histórias para extrair o objetivo daquele conjunto, agrupando-as de forma a transformar histórias com alto detalhamento em histórias de baixo detalhamento.

Como exemplo dessa dificuldade, foi identificado um conjunto de histórias listadas na Tabela 3. Em nenhuma delas existe o elemento “meta”, conforme o modelo de história apresentado na seção 2.2.1, o que as tornam ideal para uma história que deve ser implementada em uma iteração. Portanto, foi necessário agrupá-las e extrair qual a meta associada a elas, tendo como resultado uma história no formato correto (ressaltando as mudanças em negrito): "Eu como **usuário interno**, quero **configurar documentos fixos e não fixos** para que eu possa **enviar para o cliente documentos fixos, como folders e carteirinhas, e documentos variáveis, como certificado e características gerais, em um mesmo kit.**"

Tabela 3 - Grupo de histórias de usuário com dificuldade por causa da granularidade

Eu como usuário de TI quero criar um documento do tipo NÃO FIXO
Eu como usuário de TI quero criar um documento do tipo FIXO
Eu como usuário de TI quero ativar um documento do tipo FIXO
Eu como usuário operacional quero ativar um documento do tipo NÃO FIXO
Eu como usuário operacional quero mudar a VERSÃO CSF de um documento NÃO FIXO

Quanto à complexidade das histórias, por se tratar de um domínio extremamente específico, o modelo gerado requer um conhecimento prévio a respeito do sistema, ou seja, o modelo em si não contém todas as informações

necessárias para a compreensão do domínio. Por exemplo, ao analisar os objetivos "Atender uma negociação específica entre estipulante e seguradora" e "CSF fazer o download no processo da diária", parte do conhecimento não está explícito no que diz respeito aos termos *estipulante* e *CSF*. Portanto, a eficácia da transmissão de conhecimento do modelo gerado é prejudicada quando o domínio do problema é de maior complexidade e especificidade.

Dentro do método de transformação não existe uma heurística para transformar uma história de requisito não funcional para um elemento i^* . O conjunto de histórias utilizada no estudo de caso de Jaqueira (2013) também não contém nenhuma história desse tipo. Em sistemas reais, requisitos não funcionais são comuns uma vez que descrevem necessidades de qualidade no uso do software. No sistema Certificados existem histórias de requisitos não funcionais, e por não serem abordadas pelo método de transformação, não foi possível inseri-las no modelo.

Por exemplo, a história "O tempo de resposta do serviço do portal 360, deve ser de no máximo 0,5s." deveria ser representada como um elemento *meta-soft*. Porém, ao usar o método de transformação, a ação dessa história se tornaria uma tarefa ao invés de uma *meta-soft*, pois o método de transformação apenas tem heurística para a transformação de ações nas histórias de usuário em elementos do tipo tarefa no i^* .

Devido à diferença entre a quantidade de histórias no estudo de caso de Jaqueira (2013) e a quantidade de histórias no sistema Certificados, o modelo resultante do método de transformação tornou-se mais complexo para seu entendimento e para extrair informações relevantes dentro do contexto do domínio do problema. Assim, foi necessário, em vários casos, particionar o modelo para conseguir o entendimento do mesmo, ou seja, foi necessário criar submodelos com

apenas parte dos elementos para que o entendimento fosse alcançado, como mostra o Apêndice D.

Os modelos gerados pelo método podem ser vistos nos apêndices B e C, e os submodelos feitos para melhorar o entendimento no apêndice D.

Finalmente, dentro do método proposto não existe uma heurística para a definição do grau de importância de um elemento, conforme apresentado na seção 2.1.1. Essa informação é extremamente importante para identificar pontos críticos dentro do processo de negócio. Ao se definir o grau de importância de um determinado elemento, como um recurso ou tarefa, tem-se um conjunto de informações mais completo a respeito daquele elemento, o que auxilia no processo de priorização das histórias.

4.3. Discussão

Ao aplicar-se o método de transformação de Jaqueira (2013) em um conjunto de histórias de usuário com características diferentes daquelas do estudo de caso realizado no mesmo trabalho, foram identificados alguns pontos que divergem dos resultados citados. O processo de transformação teve que ser adequado em alguns pontos, por conta dessa diferença.

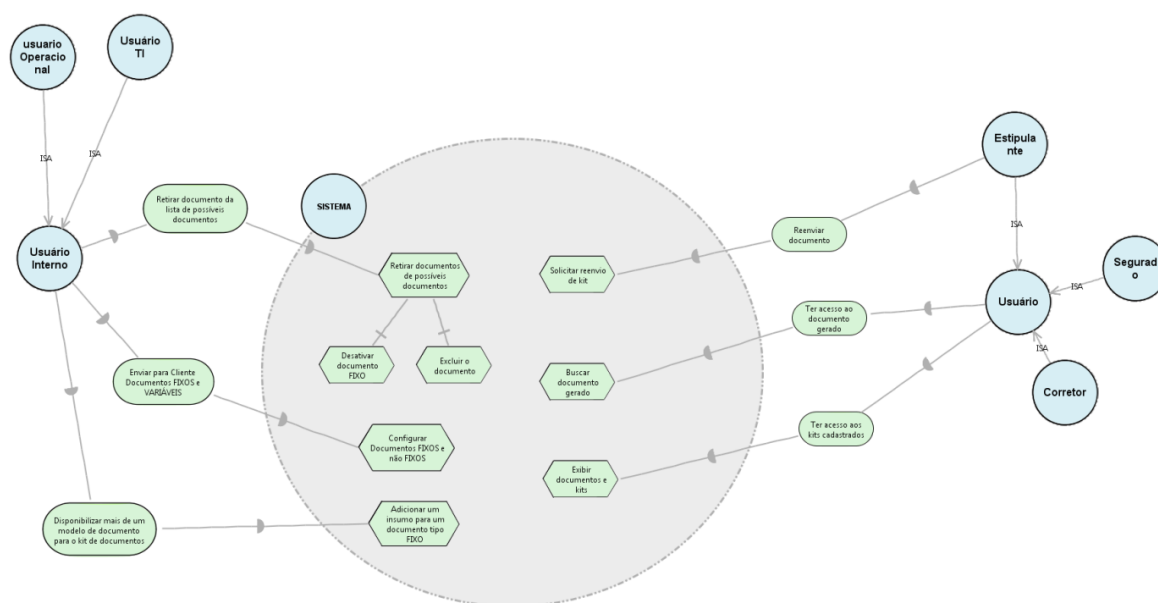
Com relação à granularidade, a diferença de homogeneidade das histórias fez com que um processo de homogeneização fosse necessário antes de aplicar o método propriamente dito, isso por que o conjunto de histórias do Sistema Certificados continha histórias que foram implementadas tanto em uma iteração como em várias iterações, justificando assim as diferentes granularidades.

O processo de homogeneização das histórias simplificou a execução do método, mas tornou sua aplicação mais demorada e trabalhosa uma vez que foi

necessário analisar todo o conjunto de histórias para só depois aplicar o método corretamente.

Segundo Jaqueira (2013), o método de transformação torna mais fácil o acesso à informação dos requisitos. Entretanto, devido à quantidade de histórias envolvidas, o acesso, e o posterior entendimento foi dificultado, pois havia muitos elementos no modelo. Assim, foi necessário criar submodelos que dividissem os elementos por algum critério, como por exemplo, por atores ou por partes do processo.

Figura 10 - Exemplo de submodelo do sistema Certificados. Fonte: o autor



A falta de heurísticas para mapear histórias de usuário de requisitos não funcionais fez com que algumas histórias não fossem mapeadas. Seria necessária uma heurística que mapeasse essas histórias em *meta-soft*.

Outro ponto a ser considerado é que, o conceito de criticidade de um elemento dentro do modelo i* poderia ser utilizado como parâmetro para priorização

de histórias e assim fazer com que os processos críticos fossem evidenciados e assim minimizar os riscos do processo como um todo.

Por último, no estudo de caso feito por Jaqueira (2013), os participantes relatam a "preocupação com a quantidade de ligações nos modelos e compreensão dos mesmos para sistemas maiores" e a "preocupação com a visualização para muitos relacionamentos". Essas duas situações se referem ao grau de complexidade e a quantidade de histórias utilizadas para a aplicação do método. Assim os modelos gerados, por sua complexidade e grande número de elementos prejudicaram a transmissão de informações referentes ao domínio do problema.

5. CONSIDERAÇÕES FINAIS

Neste capítulo será feita a conclusão do trabalho e serão propostos trabalhos futuros.

5.1. Contribuições do Trabalho

Segundo Jaqueira (2013), após a finalização do estudo de caso os participantes expressaram uma preocupação com relação à confusão na visualização em sistemas maiores e mais complexos após a aplicação do método. Dessa forma, o presente trabalho analisou a aplicação do método em um cenário com um conjunto de histórias maior e mais complexo, um sistema de geração e controle de documentos dos clientes de uma companhia de seguros, que tem como objetivo centralizar e padronizar a geração e a configuração de todos os documentos gerados pela companhia.

Concluiu-se que o método proposto realmente apresenta alguns problemas quando o cenário é diferente daquele do estudo de caso realizado em Jaqueira (2013). Os modelos gerados a partir do método se tornaram difíceis de serem entendidos e analisados, pois havia um grande número de elementos nos diagramas. Além disso, verificou-se que algumas características de um cenário mais complexo não foram abordadas pelo método, como, por exemplo, histórias de requisitos não funcionais.

Para minimizar os efeitos dessa dificuldade de leitura e análise dos modelos, a seguir foram feitas propostas de melhoria do método:

- **Processo de homogeneização das histórias para um mesmo nível de granularidade.** Durante a aplicação do método foi necessário agrupar conjuntos de histórias com o intuito de definir qual era a meta implícita naquele conjunto de histórias. Portanto, como uma fase anterior ao início da aplicação do método, seria necessário um processo para homogeneizar as histórias em um mesmo nível de granularidade.
- **Definição de uma heurística de transformação de uma história de requisito não funcional para uma meta-soft.** Algumas histórias no sistema Certificados têm como característica ser uma história de requisito não-funcional, o que não foi contemplado pelo método e, portanto, não foi mapeada em nenhum dos modelos do i^* . Esse tipo de história poderia ser mapeada como uma meta-soft no *framework* i^* , uma vez que este elemento pode representar uma característica de qualidade do software. Para isso, acrescentar-se-ia uma nova heurística no final do processo de geração do modelo SR para mapear todas as histórias de requisitos não funcionais para elementos meta-soft.
- **Definição de uma heurística para estabelecer o nível de criticidade de um elemento.** O *framework* i^* define uma forma de representar nos seus modelos a criticidade de um determinado elemento. Essa característica não é utilizada para a construção dos modelos durante a aplicação do método. Porém, ao se determinar a criticidade de um elemento, seria possível utilizar essa informação para o processo de priorização de histórias. Ou seja, ao se identificar elementos com maior criticidade é possível tornar as histórias que deram origem a eles mais prioritárias que outras. Esse processo deve ser

feito após a heurística SR-H3, que relaciona as dependências de cada meta com as tarefas correspondentes.

5.2. Trabalhos Futuros

As propostas de melhoria foram aplicadas somente em um cenário que é o objeto de estudo do presente trabalho. Para trabalhos futuros, destacam-se os seguintes tópicos:

- Aplicação das novas heurísticas propostas em cenários equivalentes ao do presente trabalho, a fim de determinar se existe melhora na aplicação do método.
- Definição e formalização de um método de homogeneização do nível de granularidade das histórias de usuário.
- Aplicação do método contrário, de modelos i^* para histórias de usuário, utilizando as informações de criticidade dos elementos para auxiliar o processo de priorização de histórias, utilizando o processo descrito em Agra (2015).

REFERÊNCIAS

Agra, C. et al. Specifying guidelines to transform i* Model into User Stories: an overview. Proceedings of the Eighth International i* Workshop. In: INTERNATIONAL I* WORKSHOP. Canada: CEUR, 2015

Beck, Kent. *Extreme programming explained: embrace change*. Addison-Wesley Professional, 2000.

Cohn, Mike, e Robert C. Martin. *Agile estimating and planning*. Robert C. Martin series. Upper Saddle River, NJ: Prentice Hall Professional Technical Reference, 2006.

Cohn, Mike. "Non-functional Requirements as User Stories". *Mountain Goat Software*. 2008. Disponível em: <<https://www.mountaingoatsoftware.com/blog/non-functional-requirements-as-user-stories>>. Acessado em 21 de julho de 2016.

Jaqueira, Aline de Oliveira Prata. Uso de modelos i* para enriquecer requisitos em métodos ágeis. 2013. Dissertação (Mestrado em Ciência da Computação) - Universidade Federal do Rio Grande do Norte, Natal, 2013.

Jaqueira, A., et al. Desafios de Requisitos em Métodos Ágeis: uma revisão sistemática. 3rd Brazilian Workshop on Agile Methods, São Paulo. 2012.

BECK K., et al; Manifest, 2001, "Manifesto para o desenvolvimento ágil de software". Disponível em: <<http://www.manifestoagil.com.br>>. Acessado 25 de junho de 2016.

Rubin, Kenneth S. *Essential Scrum: a practical guide to the most popular agile process*. Upper Saddle River, NJ: Addison-Wesley, 2012.

Wake, William C. "INVEST in Good Stories, and SMART Tasks – XP123". 2003. Disponível em: <<http://xp123.com/articles/invest-in-good-stories-and-smart-tasks/>>. Acessado 17 de maio de 2016.

Van Lamsweerde, Axel. "Goal-oriented requirements engineering: A guided tour". In *Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on*, 249–262. IEEE, 2001.

YU, E. S. Modeling Strategic Relationships for Process Reengineering. Thesis—Toronto, Canada: University of Toronto, 1995.

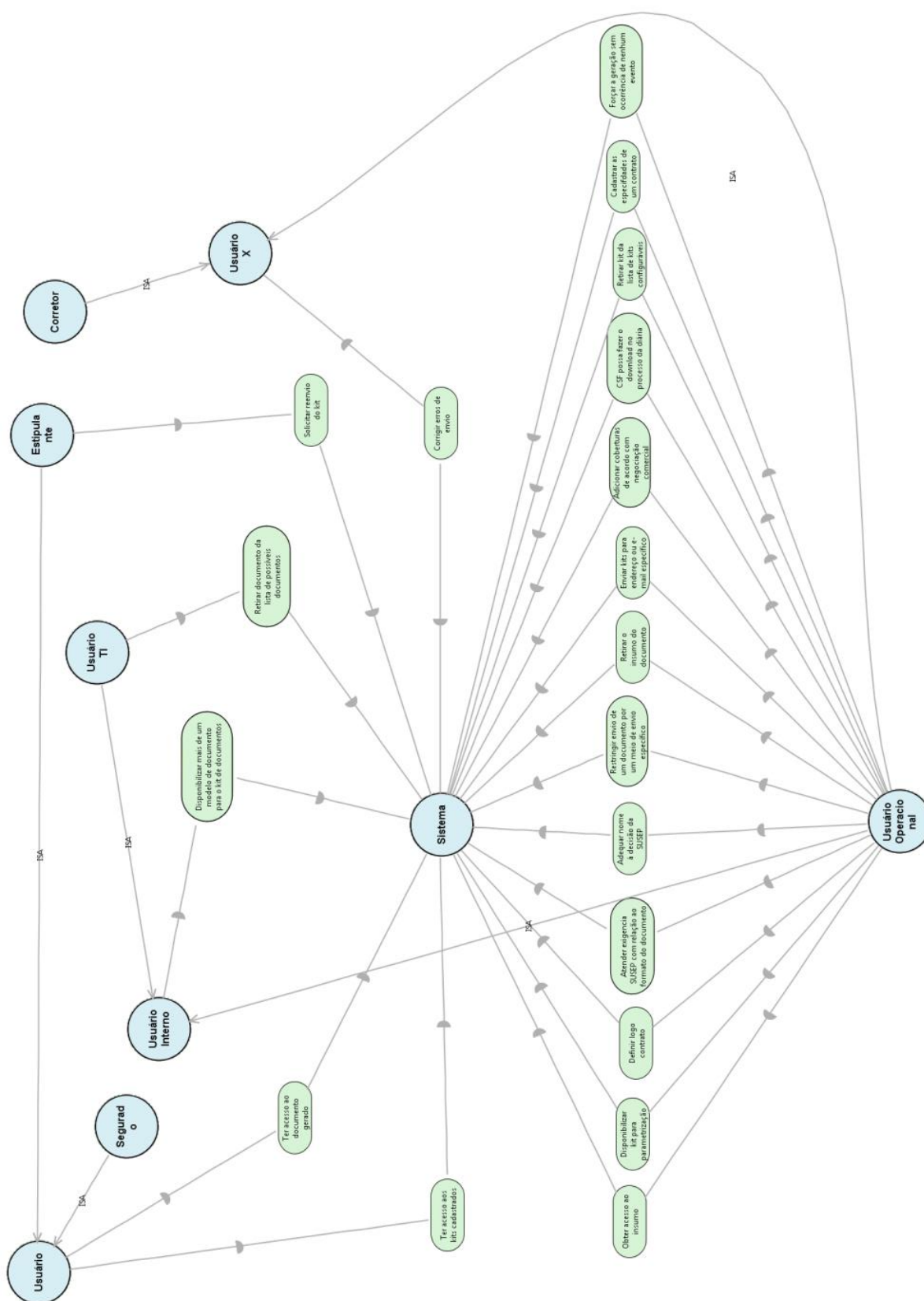
APÊNDICE A – Histórias de usuário do Sistema Certificados

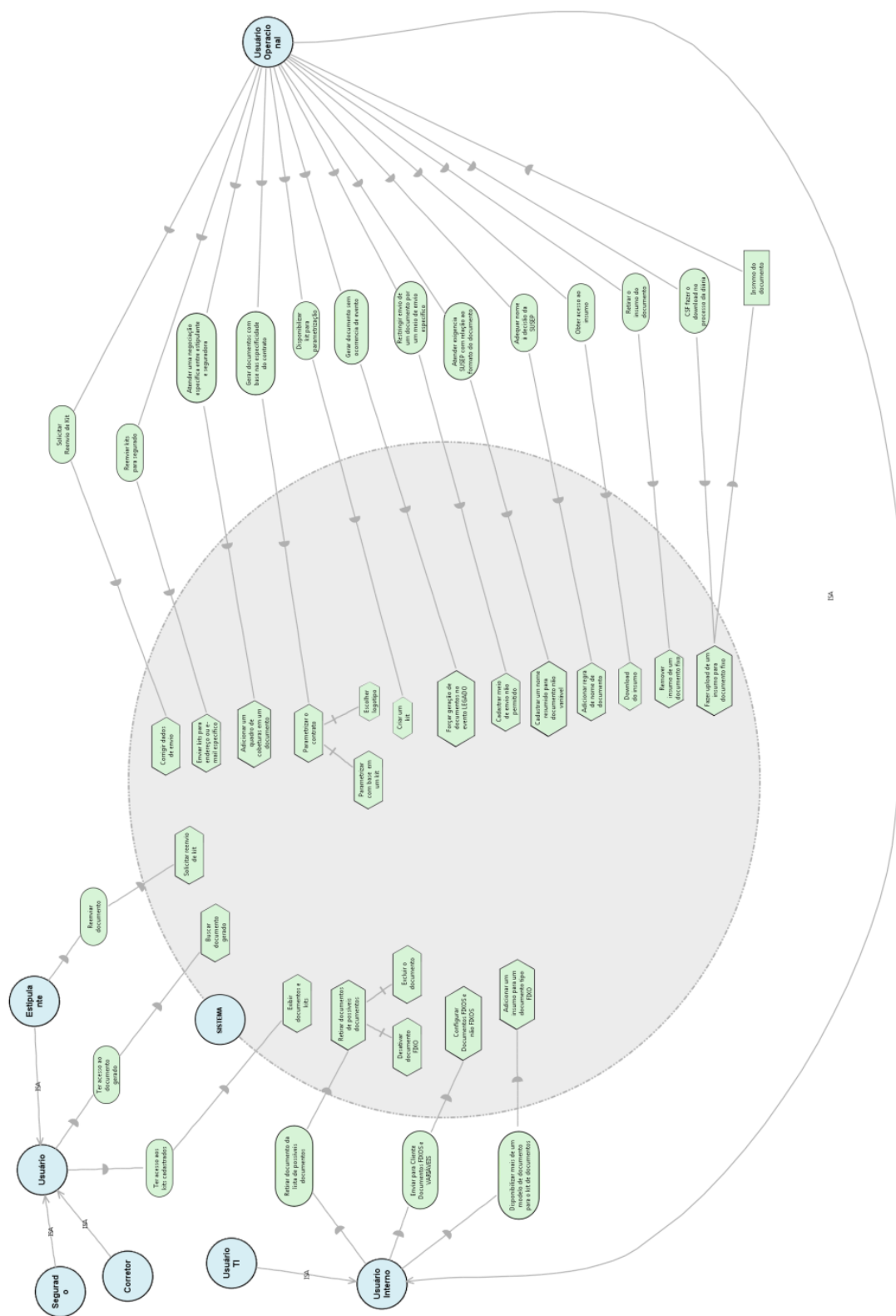
Este anexo representa as histórias de usuário do sistema de Certificados.

Eu como usuário de TI quero criar um documento do tipo NÃO FIXO
Eu como usuário de TI quero criar um documento do tipo FIXO
Eu como usuário de TI quero ativar um documento do tipo FIXO
Eu como usuário operacional quero ativar um documento do tipo NÃO FIXO
Eu como usuário operacional quero mudar a VERSÃO CSF de um documento NÃO FIXO
Eu como usuário operacional quero adicionar um insumo em um documento do tipo FIXO, para disponibilizar mais de um modelo para o kit de documentos
Eu como usuário operacional quero fazer o <i>upload</i> de um insumo para um documento FIXO, para que o CSF possa fazer o download no processo da diária
Eu como usuário operacional quero remover um insumo de um documento FIXO, para que o usuário não possa mais selecioná-lo na tela de parametrização de contrato
Eu como usuário operacional quero fazer o download do insumo previamente cadastrado, para que o usuário possa ter acesso ao insumo
Eu como usuário de TI quero desativar um documento do tipo FIXO, para que ele não possa mais ser incluído em nenhum KIT
Eu como usuário operacional quero incluir uma regra de nome em um documento, apenas para o canal AFINIDADES, para que o nome esteja de acordo com decisão SUSEP
Eu como usuário operacional quero cadastrar um nome resumido para um documento NÃO "nome variável", para que a apresentação do documento esteja de acordo com a SUSEP
Eu como usuário operacional quero excluir um documento, desde que não haja nenhum KIT com aquele documento
Eu como qualquer usuário com acesso ao sistema, quero pesquisar documentos com critérios: nome, canal, versão de referência e status.
Eu como usuário operacional quero criar um kit de documentos, para que esteja disponível para fazer a parametrização no contrato
Eu como usuário operacional quero adicionar documentos num kit
Eu como usuário operacional quero adicionar eventos ao kit
Eu como usuário operacional quero adicionar portais ao kit
Eu como usuário operacional quero configurar um insumo de um documento ao kit
Eu como usuário operacional quero configurar um meio de envio para o documento, para que o contrato não possa ser configurado com aquele meio de envio na parametrização do contrato
Eu como usuário operacional quero excluir um documento do kit, desde que não haja kits já gerados pelo processo da diária
Eu como usuário operacional quero ativar ou desativar um documento, evento ou portal de um kit, para que todos kits gerados depois NÃO usem esses itens
Eu como usuário operacional quero forçar a geração do kit no evento LEGADO, para que mesmo sem que o evento cadastrado no kit aconteça o kit seja gerado

(caso do legado nos APP ANUAL)
Eu como usuário de TI quero desativar um kit, desde que ele não tenha gerado nenhum kit ainda
Eu como qualquer usuário com acesso ao sistema, quero pesquisar kits com critérios: nome, canal, versão de referencia, status
Eu como usuário operacional quero parametrizar um contrato, com base em um kit previamente cadastrado, podendo configurar as especificidades do contrato, para que o documento reflita as configurações do contrato
Eu como usuário operacional quero selecionar um kit dentre os possíveis para aquele contrato, levando em consideração o canal, versão de referência e existência de parametrização, para que o usuário parametrize apenas o que é possível
Eu como usuário operacional quero selecionar o insumo de um documento do tipo FIXO diferente do que foi configurado no kit, para que o contrato tenha flexibilidade na parametrização
Eu como usuário operacional quero configurar os campos de texto genérico com marcações de título e parágrafo, para que o CSF possa gerar o arquivo com as devidas marcações
Eu como usuário operacional quero adicionar um quadro de coberturas em um documento, podendo fazer a configuração específica pelo contrato, para que atenda uma negociação específica entre estipulante e seguradora
Eu como usuário operacional quero ordenar as coberturas no quadro de coberturas num documento, para suprir necessidade legal pela SUSEP
Eu como usuário operacional quero configurar os campos do textos da coberturas para facilitar o entendimento pelo segurado
Eu como usuário operacional quero cadastrar o endereço do estipulante na parametrização do contrato, para que o endereço seja configurável por contrato e não por estipulante
Eu como usuário operacional quero cadastrar o email do estipulante na parametrização do contrato, para que o email seja configurável por contrato e não por estipulante
Eu como usuário operacional quero configurar o evento LEGADO de um contrato como incondicional para que o mesmo seja gerado na primeira diária depois da configuração (caso do APP ANUAL)
Eu como usuário operacional quero ativar ou desativar, marcar como enviar e marcar como obrigatório um documento, para flexibilizar a configuração do contrato
Eu como usuário operacional quero escolher um logo específico para um contrato, para que mesmo debaixo de um contrato, os sub tenham logotipos específicos
Eu como usuário operacional quero remover um logo do contrato, para que o documento seja gerado apenas com o logotipo da Seguradora
Eu como usuário operacional quero buscar no Docstore todos os logotipos para aquele estipulante ou corretor do contrato
Eu como usuário operacional quero configurar o texto do capital do segurado
Eu como usuário operacional quero configurar o texto referente à carência do contrato
Eu como usuário operacional quero configurar o texto referente à franquia do contrato
Eu como usuário operacional quero configurar o título e o rodapé da cobertura, para

incluir notas referentes às coberturas
Eu como usuário operacional quero cadastrar informações de local de risco, para os adicionar o local do bem no caso de contrato de AFINIDADES
Eu como usuário operacional quero cadastrar informações do bem segurado, para os contratos de AFINIDADES - GARANTIA ESTENDIDA
Eu como usuário operacional quero configurar o texto do beneficiários do contrato, para que o documento saia com informações específicas do contrato
Eu como usuário operacional quero configurar o texto do premio de capitalização do contrato, para os contratos que se utilizam de capitalização
Eu como usuário operacional quero configurar o texto do sorteio de capitalização do contrato, para os contratos que se utilizam de capitalização
Eu como usuário operacional quero configurar o texto das centrais de atendimento, para que os contratos com centrais diferentes, caso AFINIDADES, tenham uma configuração por contrato
Eu como estipulante quero consultar os documentos de um contrato para enviar para o segurado
Eu como estipulante quero fazer o download dos documentos de um determinado segurado para validação de condições gerais do seguro
Eu como estipulante quero solicitar o reenvio de um kit de documentos para um segurado em caso de correio, para casos de extravio ou documentação incorreta
Eu como segurado quero consultar os documentos de um seguro que eu contratei para poder consultar quais as condições do seguro
Eu como segurado quero fazer o download dos documentos para guardar ou imprimir as condições do seguro
Eu como corretor quero consultar os documentos de um contrato para confirmar informações da proposta de um seguro
Eu como corretor quero visualizar quais erros um determinado contrato apresentou para ajustar e solicitar o reenvio dos documentos
Eu como usuário operacional quero consultar um segurado para ter acesso aos documentos gerados pelo sistema
Eu como usuário operacional quero regerar um kit de um segurado para corrigir erros na configuração do contrato
Eu como usuário operacional quero antecipar a geração dos documentos de um segurado para incluir os documentos num determinado movimento
Eu como usuário operacional quero gerar a prévia de uma configuração de um contrato para validar a configuração de um contrato
Eu como usuário operacional quero visualizar os envios de uma determinada geração para confirmar informações de envio
Eu como usuário operacional quero corrigir os dados de correio, email ou telefone para que o kit seja enviado corretamente ao segurado
Eu como usuário operacional quero consultar o endereço de um corretor para enviar o kit para o corretor ao invés do segurado
O tempo de resposta do serviço do portal 360, deve ser de no máximo 0,5s.
O sistema deve ser responsivo.





APÊNDICE D – Exemplo de sub modelo SR para o Sistema Certificados

