UNIVERSIDADE ESTADUAL DO NORTE DO PARANÁ

***CAMPUS* LUIZ MENEGHEL - CENTRO DE CIÊNCIAS TECNOLÓGICAS**

SISTEMAS DE INFORMAÇÃO

MAYCONN WILLIAN GOMES DE OLIVEIRA

FERRAMENTA DE AUXÍLIO AO MONITORAMENTO REATIVO DE CONTRATOS MULTILATERAIS

Bandeirantes

2015

MAYCONN WILLIAN GOMES DE OLIVEIRA

FERRAMENTA DE AUXÍLIO AO MONITORAMENTO REATIVO DE CONTRATOS MULTILATERAIS

Trabalho de Conclusão de Curso submetido à Universidade Estadual do Norte do Paraná, como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Orientador: Prof. Wellington Della Mura

Bandeirantes

2015

MAYCONN WILLIAN GOMES DE OLIVEIRA

FERRAMENTA DE AUXÍLIO AO MONITORAMENTO REATIVO DE CONTRATOS MULTILATERAIS

Trabalho de Conclusão de Curso submetido à Universidade Estadual do Norte do Paraná, como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação.

COMISSÃO EXAMINADORA

_____
Prof. Esp. Wellington Della Mura
UENP –*Campus* Luiz Meneghel

_____
Prof. Me. Glauco Carlos Silva
UENP – *Campus* Luiz Meneghel

_____
Prof. Me. Rafaella Aline Lopes da Silva
UENP – *Campus* Luiz Meneghel

Bandeirantes, __ de_____ de2015

RESUMO

A utilização de contratos é uma necessidade do Homem, desde os primórdios da civilização. Atualmente com o surgimento dos contratos eletrônicos e da diversificação de tipos e tamanhos de contratos surge a necessidade de ter um controle maior sobre estes contratos, busca-se uma maneira de se obter informações em tempo real sobre o estado que o contrato se encontra, se houve alguma falha até o momento, ou se há possibilidades de falhas ao meio do caminho e também encontrar o verdadeiro culpado pela falha do contrato. Este monitoramento precisa ser de uma forma genérica e eficiente. Para a realização destas tarefas estes contratos precisam estar em uma representação formal e dentro de um padrão determinado, para que possam ser interpretados por um algorítimo de computador. Implementamos aqui a parte da formalização do contrato e um início a ferramenta de monitoramento destes contratos.

Palavras-chave: Contrato Eletrônico, Monitoramento, Formalização de Contratos.

ABSTRACT

The use of contracts is a necessity of man, and they exist since life in society was acquired. Currently the emergence of electronic contracts and diversification of types and sizes contracts arises the need to have more control over these contracts, search is a way to get real-time information on the state that the contract is if there was a failure to date, or if there is fault chances in half the way and also find the real culprit for the failure of the contract. This monitoring needs to be of a generic and efficiently. To carry out these tasks these contracts need to be in a formal representation and within a certain standard so that they can be interpreted by a computer algorithm. Implemented here the part of the formalization of the contract and start monitoring tool such contracts.

Key-words: Electronic contract, Monitoring, formal Agreement.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Atualmente, com a enorme gama de tecnologia encontrada em todos os lugares, tudo caminha cada vez mais para a conectividade, compartilhando informações de um jeito ou de outro, e quanto mais esse caminho é percorrido, maior a necessidade de acordos entre as partes envolvidas (COLOURIS; DOLLIMORE; KINDBERG, 2007). Um contrato é necessário para que haja formas legais de responsabilizar as partes. Um contrato eletrônico pode assumir duas formas, ele pode ser entre pessoas, por exemplo uma compra na Internet onde um comprador tem compromissos e deveres junto com um vendedor, ou ele pode ser entre dispositivos, por exemplo uma conexão de rede sem fio onde os dispositivos possuem regras que devem seguir Xu(2003).

Os contratos são importantes nesses meios, pois são eles que dizem as responsabilidades, os papéis de cada parte dentro de um acordo, tanto no caso jurídico ou entre dispositivos, é nele que estarão as regras a serem seguidas, o que cada parte pode fazer, até onde cabe a ele tomar decisões Xu (2003).

Conforme Xu (2003) as partes envolvidas em um contrato podem ser mais de duas, o que os definem como contratos multilaterais, caso sejam apenas duas partes são contratos bilaterais. Multilaterais são de difícil monitoramento, pois no geral, quanto mais partes envolvidas, mais complexo se torna o contrato.

A quebra do contrato acontece quando uma parte deixa de fazer algo que foi previamente tratado, apesar de poder parecer um pouco simples inicialmente, quando se trata de contratos multilaterais, essa tarefa de encontrar o culpado por uma quebra de contrato pode ser um pouco árdua e difícil, e pior do que isso, pode acontecer desses erros serem descobertos tardiamente, podendo levar a sérias conseqüências Xu (2003).

O primeiro passo para monitorar um contrato é formalizá-lo seguindo um padrão definido. A formalidade de um contrato se dá por conta da necessidade de ser entendível por um algoritmo computacional. Formalizar um contrato é pegar algo que é entendível pelo Homem e transformar em algo que um computador possa entender Xu (2003, 2009).

Um contrato formal passa maior confiabilidade do que está acontecendo, podem existir erros no contrato, como também podem haver erros em contratos com linguagem natural, porém, no caso da máquina, é mais fácil enxergar isso em um

contrato formal, e também a velocidade de se processar isso é muito superior para um computador do que para o Homem.

Com um contrato eletrônico formal em mãos, existem uma infinidade de processos que podem ser implementados, por exemplo: a verificação de conflitos, recuperação rápida de informações, o monitoramento, entre outros. Xu (2003, 2009) demonstra como detectar conflitos em contratos eletrônicos, passando uma confiança muito maior as partes envolvidas.

Os contratos podem possuir diversas cláusulas, e estas podem se conflitarem ao longo do contrato. Um exemplo de conflito é quando em um determinado momento um contrato autoriza uma ação que mais tarde está sendo negada, e isso irá inviabilizar o contato, pois se em um determinado momento X deve ser verdadeiro e de repente seu valor precisa ser falso, X não pode assumir dois valores simultaneamente, o trabalho de (KYAS; PRISACARIU; SCHNEIDER, 2008) demonstra um processo de buscar esses conflitos entre cláusulas de um contrato.

Para monitorar um contrato, o algoritmo que será responsável pela tarefa, precisa conhecer os processos e as partes envolvidas dentro do contrato. Para tal, será preciso enviar ao servidor que esteja rodando o algoritmo um modelo do contrato, formal, onde o servidor irá dividir as etapas do contrato, sua ordem de execução e as responsabilidades, podendo assim prever falhas. Exemplificando, uma "Empresa X precisa entregar uma mercadoria no dia seguinte, porém seu fornecedor só irá entregar o produto em dois dias úteis", logo o servidor poderá acusar o risco da falha acontecer, ela ainda não aconteceu porém, se nada for feito ela irá acontecer, nesse caso o servidor precisa alertar não somente a "Empresa X", mas também as outras partes envolvidas.

1.1 Contexto e Delimitação do Trabalho

O uso de contratos é uma tarefa essencial no processo de negócios. A contribuição para tal, veio com a interação entre empresas e pessoas e também com o avanço da tecnologia da informação, isso criou uma necessidade maior dessas partes em poder expressar seus papéis dentro do processo. As garantias de direitos ou deveres se dão a partir das regras de negócio de um contrato. Apesar dos

contratos trazerem essas garantias, essas regras podem ser quebradas por alguma das partes envolvidas.

Com o foco em contratos multilaterais, uma vez que uma ferramenta capaz de monitorar um contrato multilateral poderá de maneira semelhante monitorar contratos bilaterais, esta pesquisa contribui para a área dos contratos eletrônicos. Conforme o estudo de vários autores é nítido enxergar a maior dificuldade de um contrato multilateral se comparado ao contrato bilateral, esta dificuldade mostra se desde a busca por um responsável por uma violação até a maior complexidade de representar um contrato formalmente.

Com isso implementamos aqui a formalização dos contratos e damos início a uma ferramenta que busca monitorar contratos multilaterais, seguindo um algoritmo proposto por Xu(2003).

1.2 Formulação do Problema

A dificuldade de controle que os contratos de uma forma geral oferece, cria a necessidade de um monitoramento eficaz por parte das organizações, que precisam ter mais segurança em seus contratos eletrônicos, neste caso, segurança que o contrato está sendo seguido conforme tratado. A dificuldade do monitoramento se torna ainda maior quando tratamos de contratos multilaterais.

De forma eminente, contratos que envolvam muitas partes, e muitos processos, são muito complexos para uma organização, que podem ter o receio se o contrato está sendo seguido ou não, essa complexidade pode trazer atrasos para as partes caso elas precisem ficar revisando cláusula por cláusula dentro de um contrato cada vez que um processo ocorrer. Além de que uma parte pode querer acompanhar em tempo real o andamento de um contrato.

1.3 Objetivos

Nesta seção são apresentados o objetivo geral e os objetivos específicos deste trabalho.

1.3.1 Objetivo Geral

Expor a formalização de contratos e propor um sistema computacional que monitore contratos eletrônicos, utilizando como base uma das linguagens formais estudadas.

1.3.2 Objetivos Específicos

Formalizar um contrato, mostrando detalhadamente as peças que o compõe.

Demonstrar um serviço que possa receber um contrato formal, armazená-lo e mostrá-lo de maneira que possa ser entendível por uma pessoa.

Transformar a técnica matemática proposta por Xu para formalização de contratos e monitoramento dos mesmos em uma ferramenta computacional.

Demonstrar um estudo de caso de um contrato multilateral e seu monitoramento.

1.4 JUSTIFICATIVA

As organizações, pequenas ou grandes, necessitam acompanhar o andamento de seus processos. Os contratos multilaterais, em especial pela sua maior complexidade em relação aos bilaterais, para serem monitorados manualmente exige um esforço considerável da parte em questão, além da confiança que uma cláusula está sendo obedecida ou não.

A necessidade da formalização de um contrato e de um monitoramento automático de contratos eletrônicos multilaterais é explorada neste trabalho. Este monitoramento traz a segurança para as partes envolvidas que seu contrato está sendo seguido, apresentando em tempo real o estado que o contrato se encontra, apontando as ações tomadas pelas partes, e em caso de quebra de contrato, mostra o real culpado pela quebra.

A formalização dos contratos e o algoritmo de monitoramento teve como base o trabalho de Xu (2003), pois apesar de outros autores apresentarem outros métodos para estas tarefas, a maioria deles defendem teses voltadas para contratos bilaterais. Em seus vários trabalhos, Xu conta com um framework completo a respeito de contratos eletrônicos, dando os detalhes para trabalho com contratos multilaterais.

1.5 METODOLOGIA

O projeto possui uma abordagem exploratória afim de implementar uma técnica matemática proposta por Xu (2003). Podendo assim julgar a efetividade deste algoritmo e a confiança que pode trazer seu uso.

A pesquisa exploratória acontece quando um trabalho é de natureza exploratória quando envolver levantamento bibliográfico, entrevistas com pessoas que tiveram (ou tem) experiências práticas com o problema pesquisado e análise de exemplos que estimulem a compreensão. Possui ainda a finalidade básica de desenvolver, esclarecer e modificar conceitos e idéias para a formulação de abordagens posteriores. Dessa forma, este tipo de estudo visa proporcionar um maior conhecimento para o pesquisador acerca do assunto, a fim de que esse possa formular problemas mais precisos ou criar hipóteses que possam ser pesquisadas por estudos posteriores (GIL, 1999, p. 43). As pesquisas exploratórias, segundo Gil (1999, 0. 43) visam proporcionar uma visão geral de um determinado fato, do tipo aproximativo.

A pesquisa exploratória é realizada para um problema que não foi claramente definido. Muitas vezes ocorre antes de sabermos o suficiente para fazer distinções conceituais ou postular uma relação explicativa. A pesquisa exploratória ajuda a determinar o melhor projeto de pesquisa, o método de coleta de dados e seleção de temas. Conclusões definitivas devem ser tiradas com extrema cautela. Dada a sua natureza fundamental, investigação exploratória muitas vezes conclui que um problema percebido na verdade não existe.(GIL, 1999)

1.6 ORGANIZAÇÃO DO TRABALHO

O trabalho está dividido em 4 capítulos, são eles:

Capítulo 1: apresentado a introdução ao trabalho, bem como a contextualização, o escopo do problema, as justificativas, os objetivos e a metodologia;

Capítulo 2: apresentará a fundamentação teórica do trabalho, contendo dentre as várias sub-seções, uma que diz respeito ao monitoramento que traz junto consigo os trabalhos relacionados a questão;

Capítulo 3: apresentará o desenvolvimento, a proposta do trabalho, os detalhes de um caso de contrato multilateral e o monitoramento do mesmo, como será implementado tal ferramenta, quais tecnologias serão utilizadas, o modelo a ser seguido;

Capítulo 4: apresentará a conclusão do trabalho e possíveis trabalhos futuros.

2 FUNDAMENTAÇÃOTEÓRICA

Neste capítulo são apresentados os conceitos que embasam o desenvolvimento deste trabalho. A seção mostra referências de autores com relevante conhecimento relacionado a área de contratos e monitoramento do mesmo.

2.1 CONTRATOS

Um contrato é o resultado de um acordo entre partes, ele é quem garante de maneira legal os direitos e deveres firmados entre si. O contrato pode ser entre pessoas ou entre dispositivos e pode ser visto sob duas perspectivas, jurídica e formal. A visão jurídica é o meio legal do contrato, é a parte estruturada e que permite a interpretação humana, a visão formal é o contrato jurídico transformado em um modelo que possa ser interpretado pela máquina.(XU, 2003, 2009)

No contrato, as partes declaram suas vontades, que posteriormente se integram e possibilitam aos contratantes a aquisição, a conservação, a transferência, a modificação ou a extinção de direitos e obrigações.(SEIXAS, 1997)

Quando falamos em “parte” de um contrato entende-se por cada lado representante do contrato.

2.1.1 Contrato Bilateral

Os contratos bilaterais ocorre quando há apenas duas partes envolvidas, segundo Xu (2003) trata-se de um contrato teoricamente simples e fácil de se monitorar, levando em conta que, caso aconteça alguma divergência ou atraso, não será difícil achar o motivo que levou a falha, e como as duas partes se envolvem diretamente entre si o contrato como um todo fica transparente para ambos.

2.1.2 Contrato Multilateral

Também denominados como “Multi-party”, estes são o grande foco deste trabalho, trata-se de acordos firmados por mais de duas partes envolvidas. Em seu

trabalho, XU (2003) demonstra um caso grande de contrato que envolve uma companhia de apólices de seguros. Por conter várias partes e muitas vezes uma parte nem sequer se envolve diretamente com uma ou várias outras, esse contrato é de difícil controle, e descobrir ou prever falhas se torna uma tarefa complicada, é onde a autora propõe a formalização dos contratos.

Na figura 1 é possível ver de forma simples a interação entre as partes, pode-se observar que a parte “A” se envolve diretamente com todas as partes do contrato, enquanto que “B” e “D” se relacionam de forma indireta. Isso demonstra um pouco a complexidade encontrada dentro de um contrato multilateral.

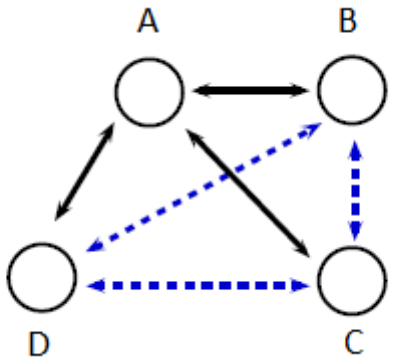

Figura 1 - Exemplo de interação em um contrato multilateral

Fonte: Autoria própria

2.1.3 Contratos – Lado Jurídico

Segundo o código civil, objetiva-se de um contrato produções de efeito jurídico, várias partes expressam seus interesses dentro da formalidade de um documento.

O contrato é o acordo de dois ou mais agentes, na conformidade da ordem jurídica, destinada a estabelecer uma regulamentação de interesses entre as partes, com o escopo de adquirir, modificar ou extinguir relações jurídicas de natureza patrimonial (BRASIL, 1987).

A partir de uma perspectiva genérica, pode-se conceituar o contrato como sendo qualquer ato jurídico em sentido amplo em que a coordenação de vontades dos contraentes é apta a produzir efeitos jurídicos (SEIXAS, 1997, p.15).

2.1.4 Contratos Meios Formais

A formalização de contratos é quando contratos convencionais são convertidos para uma linguagem padrão, com o objetivo de torná-lo entendível não somente pelo homem, mas com o princípio de ser interpretado por um algoritmo. Quando o assunto é "*e-contract*", defende-se muito essa ideia de contratos formais.

Um contrato formal é um contrato eletrônico, um "*e-contract*" que segundo Xu (2003), são contratos convencionais convertidos em modelos formais que possibilitam uma leitura por algum algoritmo de computador, sendo assim é possível seu monitoramento, correção, verificação de conflitos.

2.2 CONTRATOS ELETRÔNICOS

Um contrato eletrônico tem todas as caracterísicas de um contrato convencional, objetivando construir, modificar, conservar ou extinguir direitos entre as partes envolvidas, porém celebrado ou executado por via eletrônica (ALMEIDA, 2004).

Essa modalidade de contrato tem duas classificações, “strictu sensu“ que conforme Almeida(2004) acontece quando um contrato é construído e celebrado eletronicamente, e “latu senso” que é formado por modo tradicional e apenas executado eletronicamente.

Em se tratando de contratos eletrônicos, é possível encontrar diversos desafios em seu uso, entre eles observa-se: os conflitos em contratos eletrônicos, a negociação entre as diversas partes de um contrato, a assinatura de um contrato, os tipos de “*e-contract*”, e, o foco desse trabalho em particular, o monitoramento dos mesmos. Quanto a assinatura, no código civil até temos algo para referenciar, porém diversas vezes isso vai da interpretação de quem julga.

2.2.1 Representação dos Contratos

Em Xu (2009) são apresentados métodos de transformar contratos, em

contratos eletrônicos, de uma maneira genérica, permitindo dessa forma que um contrato qualquer possa se encaixar em seu trabalho, apresentando o formalismo necessário para que o contrato fique entendível por uma máquina. Esta possibilidade vem através de um algoritmo expressado por Xu(2003).

Na seção “Formalização de Contratos” será demonstrado cada peça que compõe o contrato, as classes e variáveis que Xu determina dentro do contrato.

2.2.2 Conflito de Interesses

Em contratos multilaterais, não exclusivamente, pode haver divergências entre as várias cláusulas do contrato, como citado por Xu (2003) vários acordos podem ser escritos por suas várias partes e posteriormente reunidos em um único contrato, sendo assim, uma cláusula pode estar permitindo algo que em uma posterior esteja negando, essa é a ideia de verificação de um contrato. A verificação de conflitos deve acontecer antes de executar o contrato.

Em seu trabalho Xu (2003) conta com um algoritmo capaz de detectar conflitos entre cláusulas dentro de uma linguagem formal própria.

2.2.3 Negociação entre Partes

As negociações ocorrem se baseando nos conflitos encontrados, aqui se propõe as saídas possíveis para que o contrato seja plausível a todos. Essa fase ocorre após a verificação de conflitos, ao encontrar um conflito eminente, este precisa ser resolvido para que o contrato possa ser validado, cabe a negociação entre partes indicar soluções para os conflitos encontrados.

2.2.4 Assinatura

Quando o assunto são contratos eletrônicos, muitos não são assinados da maneira tradicional, um exemplo disso é quando se compra algo na Internet. Entre o

cliente e o vendedor há um contrato de compra e venda, que apesar de "não assinado" tem o poder de direitos e deveres para ambas as partes.

No Código Civil (BRASIL, 1987) são vários artigos referentes a contratos, e um deles chama a atenção, o Artigo 428 inciso I "...Considera-se também presente a pessoa que contrata por telefone ou por meio de comunicação semelhante", seguindo este texto de lei pode-se notar a legalidade de um contrato feito eletronicamente. Logo abaixo o artigo 434 complementa essa forma legal de contrato, quando diz-se assim: "Os contratos entre ausentes tornam-se perfeitos desde que aceitação é expedida...".

Apesar de no Brasil não termos um código exclusivo para a área digital, no artigo 332 do Código de Processo Civil (RT, 2008), temos: "Todos os meios legais, bem como os moralmente legítimos, ainda que não especificados neste Código, são hábeis para provar a verdade dos fatos, em que se funda a ação ou a defesa."

2.3 MONITORAMENTO

O monitoramento consiste em detectar e prever falhas ao longo do caminho de um contrato. Quando uma parte ignora uma cláusula, estará violando um contrato. O monitoramento torna a execução confiável e eficiente, dando as partes a confiança que o contrato está sendo seguido a risca. O monitoramento ainda precisa saber quando que uma violação tem um nível aceitável (Xu 2003, 2009).

Em contratos multilaterais a tarefa de monitorar um contrato é um tanto quanto complicada, segundo Xu (2003) e (2009), através de exemplos práticos, a autora demonstra a dificuldade de se monitorar um contrato com várias partes envolvidas.

Posteriormente no tópico "Monitorando Contratos Eletrônicos Multilaterais Xu(2003, 2009)" será demonstrado os passos para monitoramento dos contratos.

2.3.1 Monitorando Contratos Eletrônicos em Tempo de Execução (KYAS, PRISACARIU, SCHNEIDER, 2008)

Nesse trabalho os autores discutem a importância e a necessidade de se monitorar um contrato em tempo de execução, objetivando o cumprimento de cada processo dentro do contrato e avaliando o comportamento esperado por cada parte envolvida.

O trabalho mostra como obter um monitoramento em tempo de execução de contratos escritos em CL, entende-se por CL uma linguagem de especificação formal que permite escrever obrigações condicionais, permissões e proibições sobre as ações que devem ser efetuadas dentro de um contrato. Através da linguagem apresentada é exposto como obter, para um determinado contrato, um autômato que aceita exatamente o caminho que deve ser percorrido pelo contrato.

O autômato extraído do contrato é a base para obtenção de uma máquina de estados finitos, que age como um monitor em tempo de execução do contrato.

2.3.2 Monitoramento de Contratos Eletrônicos Baseados em Características (SANTOS, 2011)

Neste trabalho o autor foca bastante no monitoramento de processos de negócio dentro das empresas e desenvolve uma arquitetura para execução e monitoramento do mesmo. A abordagem proposta aqui visa garantir meios para que os contratos eletrônicos sejam de fato monitorados, e que de maneira automática, atitudes sejam tomadas a fim de prevenir a quebra de contrato, fornecendo um conjunto de ações a serem executadas quando a quebra de contrato é eminente.

O autor não se preocupa muito em como transformar um contrato comum em contrato eletrônico, porém fornece uma ideia de como fazê-lo, deixando isso como um trabalho futuro a sua tese.

Com a criação de um serviço web responsável por fazer esse monitoramento, poucas alterações são feitas no processo que a empresa já segue, alterando apenas o endereço onde uma requisição dentro do contrato será feita, em vez de enviar a requisição diretamente ao servidor da parte encarregada, envia ao

servidor de monitoramento, que por sua vez irá armazenar esse pedido, e enviar ao servidor responsável, esperando pela resposta, podendo monitorar seu processo.

Para que o processo de negócio possa ser monitorado, é necessário que o administrador do processo de negócio envie o processo de negócio para a ferramenta SMPN. Essa ferramenta se responsabiliza por criar um serviço Web monitor, chamado WS-Monitor, para cada serviço Web presente no WS-Contract.

Os serviços Web monitores são responsáveis por monitorarem a comunicação entre o processo de negócio e os serviços Web. Dessa forma, os WS-Monitor's trabalharão como intermediários entre consumidores e fornecedores de serviços (SANTOS, 2011, p.45).

2.4 MONITORANDO CONTRATOS ELETRÔNICOS MULTILATERAIS (XU, 2003, 2009)

Este trabalho propõe um framework repleto de funções para contratos eletrônicos, partindo desde o princípio quando o trabalho apresenta um algoritmo para transformar um contrato comum em um contrato eletrônico, passando por detecção de conflitos, negociação entre partes, assinatura e chegando ao monitoramento, que ainda ganha dois níveis, o momento em que a ação acontece ou deixa de acontecer e o posterior que são as atitudes a serem tomadas em cima desses acontecimentos.

Em seu trabalho Xu expressa a dificuldade em monitorar contratos que envolvam mais de duas partes, Xu explica isso com seu caso de teste, seu exemplo é um contrato contendo três partes "A", "B" e "C" onde "A" tem contado direto com "B", "B" tem contato direto com "C" e "A" contato indireto com "C". Nesta perspectiva Xu aponta a dificuldade em se encontrar possíveis falhas, tendo em vista que "A" depende de "C", mas não mantém contato diretamente com o mesmo. Em uma possível falha de "C", "A" pode culpar "B" por desconhecer as funções de "C" dentro do escopo. Xu ainda aponta que uma descoberta de falha tardiamente pode parar todo um processo que poderia ter dado certo caso a falha fosse apontada antes de ocorrer ou no mínimo no ato em que ocorreu.

Em seu trabalho não é defendido apenas a causa da falha mas também

quando ocorreu e quem cometeu, e além disso, a possibilidade de se prever eventuais falhas, alertando as partes envolvidas diretas ou indiretamente. Quanto a busca pelo responsável real pela falha, usando o exemplo acima, a eventualidade pode ter ocorrido em "C" porém pode ser que apenas o tenha ocorrido por uma falha em "B". Em um monitoramento capaz de prever falhas, o sistema poderia alertar as partes, para que as mesmas possam se corrigir a tempo evitando que o erro aconteça.

O monitoramento de violação de contratos defendidos pela autora são apresentados em duas abordagens, inicialmente o monitoramento reativo e posteriormente o pró-ativo.

2.4.1 Monitoramento reativo e pró-ativo

Xu(2003, 2009) propõe dois algoritmos matemáticos para monitoramento reativo e pró-ativo.

O algoritmo reativo consiste em acusar uma violação no ato em que a mesma ocorre, parando o contrato e avisando as partes envolvidas, para isso este monitoramento precisa saber quais são as partes do contrato e suas responsabilidades.

O algoritmo pró-ativo é uma melhoria sobre o reativo e depende diretamente deste primeiro algoritmo, este módulo consiste em prever falhas, bem como dar alternativas para que não ocorra a quebra do contrato.

A figura 2 mostra o algoritmo para o monitoramento reativo e a figura 3 mostra o algoritmo para o monitoramento pró-ativo.

Algorithm 1 Maintaining Guards

$U = \emptyset$;
while $\Gamma - U \neq \emptyset$ **do**
  take next u out of $\Gamma - U$;
  $guard[u] = \mathsf{G}(constraint, u)$
  if $guard[u] \equiv \top$ **then**
    $do(u)$;
    $U = U \cup \{u, \overline{u}\}$;
    $V = \Gamma - U$;
    while $V \neq \emptyset$ **do**
      take any v out of V;
      $guard[v] = \mathsf{G}(guard[u], v)$;
      $guard[\overline{v}] = \mathsf{G}(guard[u], \overline{v})$;
      $V = V - \{v, \overline{v}\}$;
    end while
  else if $guard[u] \equiv 0$ **then**
    reject occurrence of u;
  else
    send reminding or warning messages;
  end if
end while

Figura 2 - Algoritmo reativo

Fonte: Xu(2003)

Algorithm 2 Pro-active Detection

if $\mathsf{G}(w, a) \equiv \top$ **then**
  while $\neg done(a) \wedge f_w(a) < t \leq a.t$ **do**
    $warn(a)$
    $\forall b \in \{w - \{a\}\}$ $remind(b)$
  end while
  if $done(a) \equiv TRUE$ **then**
    $update(M_0)$
    Call Scheduting Guards Algorithm
    $remind(b)$
  end if
end if

Figura 3 - Algoritmo pró-ativo

Fonte: Xu(2003)

2.4.2 Formalização do Contrato

Em (XU, 2003) um contrato é o acordo entre duas ou mais partes e é baseado em compromissos mútuos. Conforme a "Figura 4" nosso modelo de contrato pode ser dividido em dois módulos principais: o Elemento Monitorável (Monitorable Element "ME") e o Mecanismo do Monitoramento (Monitoring Mechanism "MM").

O Elemento Monitorável inclui dois sub módulos o Processo de Negócios e a Lógica de Relacionamento (Trading Process e Logic Relationship). O Processo de Negócios é quem contém o contrato em si, nele estará as partes envolvidas e suas responsabilidades. A Lógica de Relacionamento se obtém através deste módulo anterior e serve como base para funcionamento do algoritmo pró-ativo, é nele que estará as restrições contratuais.

O módulo reativo e pró-ativo encontram-se dentro do Mecanismo de Monitoramento. A figura 4 demonstra como Xu desenvolve seu algoritmo matemático. O módulo reativo possui quatro submódulos: lembrar, avisar, rastrear e módulo de compensação.

Xu começa a formalização do contrato dividindo seu modelo proposto em três partes: Ação, Compromisso e Grafo de Compromissos.

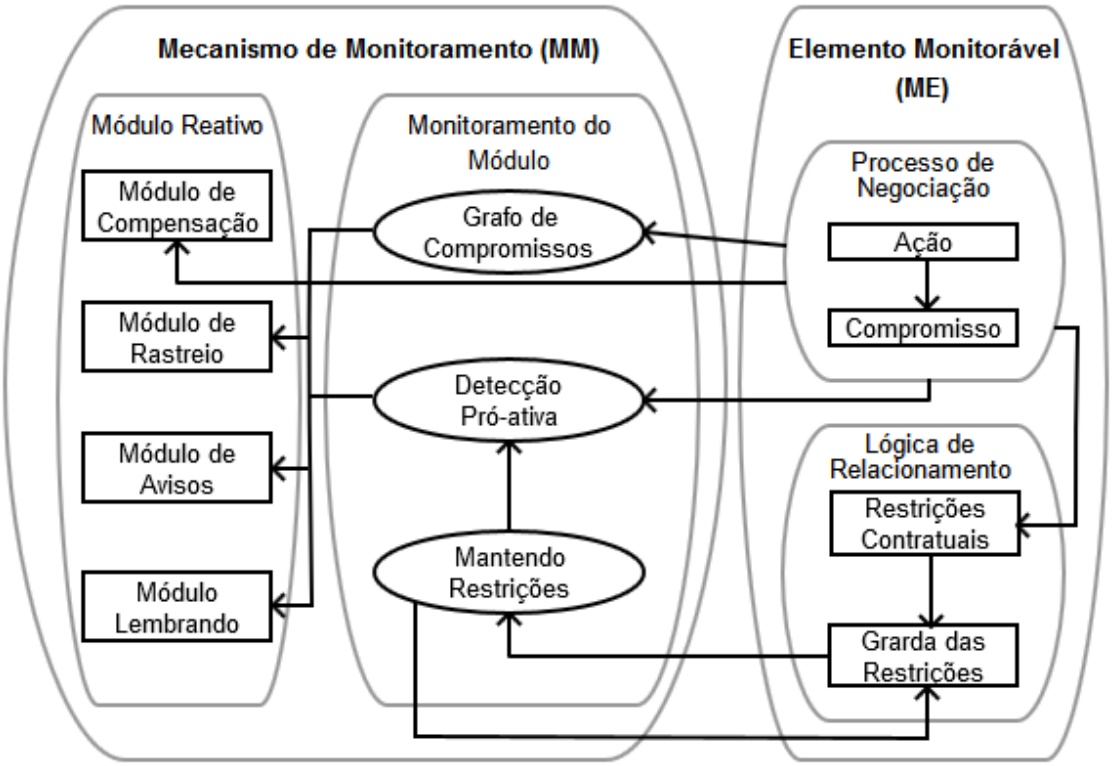

Figura 4 – Modelo traduzido e adaptado de Xu

2.4.3 Contrato Multilateral proposto por Xu

Para explicar cada componente que forma um contrato formalizado Xu prepara um contrato multilateral que possui vários agentes e vários compromissos entre eles. Seu caso de uso trata-se de uma apólice de seguros onde um usuário logado solicita reparos em seu automóvel à seguradora.

Processo de reparo ao automóvel:

1. O Segurado faz uma ligação gratuita para o Atendimento 24h para solicitar ao seguro o reparo de seu automóvel.

2. O Atendimento 24h vai registrar as informações, e sugerir ao cliente a oficina mais adequada para seu caso e notificará a seguradora o pedido de reparo.

3. A Seguradora irá verificar se a apólice deste Segurado é válida e se a solicitação é coberta pelo plano do segurado.

4. Após receber esta solicitação, a Seguradora vai enviar os detalhes para a

Equipe de Consultoria.

5. A Seguradora irá enviar uma carta para o Segurado com um formulário de solicitação.

6. A Equipe de Consultoria irá autorizar o reparo caso o dano do carro seja pequeno, caso seja verificado um estrago maior, um Avaliador será solicitado para o caso.

7. O Avaliador irá verificar o carro danificado e irá impor um acordo de valores com a oficina responsável pelo reparo.

8. Após receber o acordo de reparação do automóvel, a Oficina inicia os reparos no veículo.

9. Realizado os reparos, a Oficina irá emitir um relatório técnico e a fatura para a Equipe de Consultoria que irá verificar o valor com o orçamento pré estabelecido.

10. A Equipe de Consultoria envia toda a documentação emitida pela Oficina para a Seguradora que armazena os relatórios junto a documentação de seu cliente e por fim efetua o pagamento das faturas recebidas.

11. A qualquer momento do processo, caso a solicitação seja considerada inválida, todos os participantes do processo (indivíduos) serão notificados e o processo será interrompido.

Com isso podemos agora analisar e formalizar este contrato, explicando cada peça que será necessária.

2.4.4 Partes

Xu expressa em seu trabalho que cada agente participante de um contrato é chamado de parte, e para formalizar o contrato é necessário identificá-las. Um conjunto de partes é denominado conjunto $\mathcal{P}$. Para seu caso de uso Xu identifica e faz uma breve descrição das partes, esta descrição serve apenas para fins didáticos, pois na modelagem mesmo das partes ela não será necessária.

Tabela 1 - Partes envolvidas em um contrato

Parte	Função
Segurado (Policyholder)	Solicita o reparo de seu automóvel
Seguradora (AGFil)	Empresa responsável do seguro do automóvel
Atendimento (Europ Assist)	Empresa responsável por atendimento telefônico 24 horas
Consultoria (Lee Consulting Services)	Acompanha todos os processos em nome da seguradora
Oficina (Garage)	Empresa que efetua os reparos no automóvel
Avaliador (Assessor)	Inspeciona o automóvel e o serviço prestado

Para cada umas das partes é especificado um ID, que deverá ser único e servirá para identificar a parte em qualquer lugar do contrato, são eles : “P” para Policyholder (Segurado), “AG” para AGFil (Seguradora), “E” para Europ Assist (Serviço telefônico 24 horas), “L” para Lee Consulting Services (Consultoria), “G” para Garage (Oficina) e “A” para Assessor (Avaliadores). Um conjunto $\mathcal{P}$ é especificado contendo os IDs de cada parte:

$$\mathcal{P} = \{P, AG, E, L, G, A\}$$

Uma parte ainda pode desenvolver vários papéis dentro do contrato, como é o caso da oficina, ela interage em diferentes compromissos. Sabendo que a oficina assumi três papéis diferentes, seus papéis são representados como G’, G’’ e G’’’, assim temos um conjunto de papéis para a oficina denominado $\mathcal{R}_{garage}$:

$$\mathcal{R}_{garage} = \{G', G'', G''' \}$$

O conjunto com todas os papéis pode ser especificado como:

$$\mathbb{R} = \{P', P'', E, AG, L, A, G', G'', G'''\}$$

2.4.5 Ação

Uma ação indica o papel de cada parte dentro do contrato, trazendo suas tarefas, o que a mesma deve realizar.

Segundo (XU, 2003) uma ação representa um átomo dentro do modelo de contrato, cada ação é representado por seu nome de ação, geralmente um verbo

que indica o que deve ser feito, a ação contém um transmissor que deverá praticar a ação, e um receptor, quem recebe a ação e o prazo da ação. Um agente pode estar envolvido em diversos compromissos e desempenhar diferentes papéis dentro do contrato.

A representação de uma ação:

action = (name, sender, receiver, t)

Onde "name" ∈ ID e é o identificador da ação, esse identificador deve ser único para permitir seu rastreamento, "sender e receiver" ∈ R ou seja, representam seus papéis dentro da ação referida, indicam o trasmissor e o receptor respectivamente, "t" ∈ T, representa o tempo de duração da ação, este tempo da ação pertence ao conjunto tempo, este conjunto é definido por quem implementa a formalização, ele pode ser segundos, minutos, horas, dias, etc. Um conjunto de ações é representado pelo conjunto $\mathbb{A}$.

$$\mathbb{A} = \bigcup_{\forall x \in \mathcal{P}} \{action\}$$

A tabela 2 mostra algumas ações possíveis para seu caso de uso proposto:

Tabela 2 - Ações envolvidas em um contrato

Ações para Segurado	
(A_AberturaDeChamado,P,E,0)	O segurado entra em contato com o atendimento (Europ Assist) informando que seu carro teve algum dano. Seu retorno deve ser imediato, por isso o tempo 0.
(A_enviarCarro, P, G, 1)	O segurado envia o carro para a oficina indicada pelo atendimento, esta ação deve ser efetuada em no máximo um dia.
Ações para atendimento (Europ Assist)	
(A_atribuirOficina,E,P,0)	O atendimento informa imediatamente a oficina em que o segurado deve levar seu carro.

Seu caso de uso possui várias outras ações, mas não cabe apresentar todas aqui neste momento.

2.4.6 Compromisso

Em (XU, 2003) um compromisso é a garantia de uma parte sobre outras, que uma sequência de ações especificadas serão executadas completamente. Um compromisso é basicamente um conjunto de ações.

As ações possuem atributos para indicar seu papel dentro do compromisso, logo elas podem iniciar, participar ou finalizar um compromisso. Uma ação pode assumir mais de um atributo, mas nunca no mesmo compromisso, sendo assim uma ação pode estar presente em diferentes compromissos. A representação de um conjunto de atributos de ações:

$$\mathcal{U} = \{tr, in, fi\}$$

Dentro deste conjunto $\mathcal{U}$ temos três variações que a ação pode assumir, “tr” significa trigger, quando a ação obtiver este atributo dentro do compromisso ela será a responsável por inicia-lo, “fi” significa finish, quando a ação for responsável por fechar um compromisso, garantindo que tudo anteriormente já foi executado, “in” significa involve, por fim todas as ações que precisam do compromisso aberto para rodar e não possuem autonomia para finalizar o compromisso estarão envolvidas e receberão este atributo.

Um compromisso é descrito por um nome de compromisso, um transmissor do compromisso, um receptor do compromisso e uma serie de ações e seus atributos.

$$commitment = (name, sender, receiver, \{(a1,u1),...,(an,un):a1 \in \mathbb{A}, u1 \in \mathcal{U}\})$$

Onde name é o id do compromisso e deve ser único, sender e receiver são as partes responsáveis por enviar e receber o compromisso respectivamente, a1 .. an são as ações dentro do compromisso seguindo por u1 .. un que representam os atributos destas ações.

O conjunto de todos os compromissos é denotado por $\mathbb{M}$ e representado:

$$\mathbb{M} = U \{compromissos\}$$

$$\forall \boldsymbol{x} \epsilon \mathcal{P}$$

Um exemplo de compromisso para seu caso de uso:

(C_ServiçoDeReparo, G, P, {(A_EnviarCarro,tr), (A_estimativaDeCustos,fi),
(A_AceitarReparo,tr), (A_RepararCarro,fi) })

O nome do compromisso será C_ServiçoDeReparo, o transmissor do compromisso será a oficina e o receptor será o segurado, este compromisso possui quatro ações, sendo duas de gatilho, ou seja, duas responsáveis por iniciar o compromisso e possui duas ações de finalizar, responsáveis por fechar o compromisso.

2.4.7 Grafo de Compromissos

A principal parte que compõe os contratos são os compromissos, que se torna um conceito ainda mais importante quando se trata de especificar contratos multilaterais. No Grafo de Compromissos são apresentados os relacionamentos complexos entre os vários compromissos de um contrato, nele será possível visualizar o processo de cada ação dentro de cada compromisso, bem como seu papel dentro do escopo, por exemplo, uma ação que inicia um compromisso ou uma ação que finaliza um compromisso. Os compromissos tem características dinâmicas.

O modelo apresentado por (XU, 2009) mostra vários compromissos e várias ações, ações que iniciam os compromissos, que envolvem o compromisso e que os finalizam, a tabela 3 apresenta seu grafo de compromissos com alguns compromissos e ações de seu caso de uso.

Tabela 3 - Classificação das ações e seus compromissos

Compromisso	Classificação das ações e compromissos			Abrev.
	Gatilho(trigger)	Envolver(involve)	Finalizar(finish)	
C_ServiçoFone	A_abrirChamado			PS.1

(PS)		A_receberInformações		PS.2
			A_atribuirOficina	PS.3
			A_notificarChamado	PS.4
C_ServiçoReparo (RS)	A_enviarCarro			RS.1
			A_estimativaCusto	RS.2
	A_aceitarReparo			RS.3
			A_repararCarro	RS.4
C_Formulario (CF)	A_notificarChamado			CF.1
		A_enviarFormulario		CF.2
			A_retornarFormulario	CF.3

A complexidade de um grafo de compromisso pode ser expresso no compromisso C_serviçoReparo:

(C_serviçoReparo, G, P, {(A_enviarCarro,tr), (A_estimativaCusto,fi),
(A_aceitarReparo,tr), (A_repararCarro,fi)}

Onde o compromisso possui duas ações que iniciam e duas que finalizam, lembrando que as ações precisam seguir uma ordem, então a primeira ação de gatilho precisa sempre ser a primeira a executar dentro do compromisso, depois a segunda e assim sucessivamente.

Os compromissos devem seguir uma ordem, para isso cada relaciomento de um compromisso para outro deverá ser expresso da seguinte maneira:

order_commitment = {(m1.m2) : m1, m2 ϵ $\mathbb{M}$}

Onde m1 é o compromisso que deve ocorrer primeiro e só então o

compromisso m2 poderá ocorrer, e ambos devem estar contidos no conjunto dos compromissos.

O conjunto da ordem em que os compromissos devem seguir é representado pelo conjunto $\mathbb{O}$:

$$\mathbb{O} = U \{(m1 . m2)\}$$

$$\forall \boldsymbol{x} \epsilon \mathcal{P}$$

Para seu exemplo o conjunto com a ordem dos compromissos será:

$\mathbb{O}$ = {C_ServiçoFone.C_ServiçoReparo, C_ServiçoFone.C_Formulario,

C_ServiçoReparo.C_ServiçoCustos}

2.5 TECNOLOGIA

Nesta seção são apresentado as tecnologias utilizadas no trabalho.

2.5.1 Ruby on Rails

Rails foi originado a partir de uma aplicação chamada Basecamp que foi criada por David Heinemeier Hansson, que é era um gerenciador de projetos, desenvolvido para a 37signals. Depois do grande sucesso do Basecamp, a 37signals moveu para área de desenvolvimento e produção (LENZ, 2007). Ruby on Rails inicialmente era um *framework stand alone*, mas perceberam que a real aplicação do *framework* não seria essa. O *framework* tinha funcionalidades como abstração da base de dados e dos *templates*. Em julho de 2004 foi lançado a primeira versão beta e a primeira release em dezembro de 2005 (LENZ, 2007).

Rails é um *framework* de desenvolvimento web, feito em Ruby e tem a finalidade de diminuir o tempo do processo de desenvolvimento de sistemas web. Como Ruby é uma linguagem interpretada, o *framework* é capaz de acrescentar e alterar funcionalidades a linguagem. Também é chamado de um *meta-framework*, por ser composto pela junção de vários componentes. Os seus componentes

constituintes são o *ActionPack*, *ActiveRecord*, *Active Support*, *ActionMailer*, e *ActiveResource*, este último que surgiu substituindo a *ActiveWebService* (FERNANDEZ, 2007).

2.5.2 Web Service

Um serviço web (Web Service) fornece uma interface de serviço, que permite aos clientes deste serviço comunicarem com este servidor, por meio de uma padronização de dados de entradas e saídas, o usuário informa ao serviço web o dados necessários e exigidos pelo serviço que irá processar esses dados e devolver uma resposta (COLOURIS; DOLLIMORE; KINDBERG, 2007).

As requisições e as respostas aceitas e geradas pelo servidor utilizam um padrão formatado em XML (COLOURIS; DOLLIMORE; KINDBERG, 2007).

A infra-estrutura fornecida por um serviço web torna mais rica e estruturada a interoperabilidade entre clientes e servidores, e é através dele que é possível a comunicação entre um determinado cliente em uma organização com um servidor de uma outra organização (COLOURIS; DOLLIMORE; KINDBERG, 2007).

A utilização de um serviço web se da por meio de uma requisição HTTP, onde se inicia a troca de mensagens entre cliente e servidor utilizando o XML (COLOURIS; DOLLIMORE; KINDBERG, 2007).

2.5.3 HTTP

HTTP significa HyperText Transfer Protocol, ou seja, Protocolo de Transferência de Hipertexto. É um protocolo para comunicação entre Sistemas que permite transferência de dados entre redes de computadores, principalmente na internet (www.w3schools.com).

É o protocolo utilizado para transferência de páginas HTML do computador para a Internet. Por isso, os endereços dos websites (URL) utilizam no início a expressão "http://", definindo o protocolo usado. Esta informação é necessária para estabelecer a comunicação entre a URL e o servidor Web que armazena os dados, enviando então a página HTML solicitada pelo usuário.

O HTTP se comunica através de TCP/IP. Um cliente http conecta a um

servidor usando TCP. Depois que a conexão foi estabelecida o cliente pode enviar uma requisição http para o servidor.

Com isso o servidor irá processar a requisição e devolverá uma resposta para o cliente, essa resposta contém um código de status indicando um erro ou o sucesso da solicitação, e o conteúdo da resposta.

2.5.4 XML

O XML é uma linguagem de marcação, ou seja, um conjunto de convenções utilizadas para a codificação de textos (ALMEIDA, 2002), permite ao autor especificar a forma dos dados no documento.

O XML possui uma estrutura de identificação muito simples, para evitar uma confusão de conceitos é utilizada uma estrutura de "namespaces", utilizados para diferenciar uma tag de outra.

2.5.5 GITHUB

GitHub é um Serviço de Web Hosting Compartilhado para projetos que usam o controle de versionamento Git. É escrito em Ruby on Rails pelos desenvolvedores da Logical Awesome (Chris Wanstrath, PJ Hyett e Tom Preston - Wernder). O GitHub possui planos comerciais e gratuitos para projetos de código aberto.

Este site possui funcionalidades de uma rede social como *feeds, followers, wiki* e um gráfico que mostra como os desenvolvedores trabalham as versões de seus repositórios.

3 DESENVOLVIMENTO

A primeira parte do desenvolvimento é a formalização e disponibilização do contrato, demonstrar as classes matemáticas utilizadas por Xu e transformar em xml. Para guardar e apresentar o contrato será utilizado para desenvolvimento o framework Ruby on Rails. Em seguida será demonstrado um pseudocódigo com a implementação do algoritmo matemático de Xu responsável por buscar o culpado pela falha do contrato.

3.1 ESTUDO DE CASO PROPOSTO

Para provar o conceito da formalização e monitoramento de um contrato multilateral, será abordado o caso de uso da realização de uma compra no Mercado Livre.

Processos para compra de um produto:

1 - O cliente acha um produto de seu interesse e inicia a compra;

2 - O cliente passa por uma tela de pagamento onde é apresentado as formas de pagamentos possíveis, o cliente então escolhe sua forma e preenche os dados solicitados;

3 - O Mercado Pago responsável por receber o pagamento e intermediar a compra irá validar o pagamento do cliente;

4 - O Mercado Pago irá avisar o vendedor sobre o pagamento;

5 - O vendedor irá solicitar a uma transportadora que faça a entrega da mercadoria;

6 - Recebendo o produto o cliente avisa o Mercado Pago que está tudo certo;

7 - O Mercado Pago então libera o pagamento para o vendedor e assim o contrato finaliza.

Apresentando isso em um grafo:

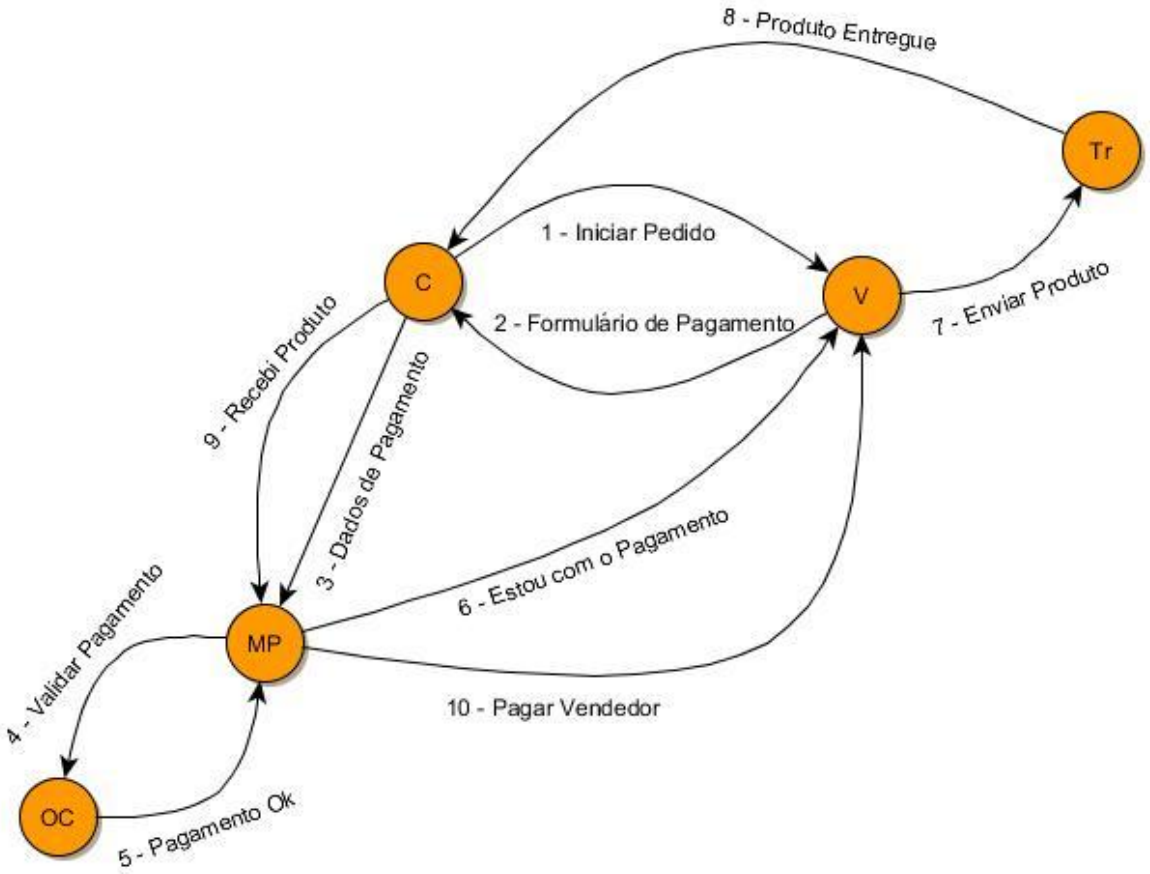

Figura 5 - Grafo demonstrando o fluxo de uma compra no Mercado Livre

O próprio grafo já permite que vejamos as cinco partes envolvidas no contrato, porém podemos analisar isto mais detalhadamente na tabela 4:

Tabela 4 - Partes do caso de uso Mercado Livre

Parte	Função
Cliente	Efetua a compra de um produto
Mercado Pago	Empresa responsável por intermediar a compra entre o vendedor e o comprador dando a segurança que os dois lados irão receber suas partes.
Operado de Cartão	Empresa responsável pelas transações do cartão
Transportadora	Empresa responsável pela entrega de mercadorias
Vendedor	Efetua a venda de produtos no Mercado Livre

A tabela 5 demonstra as ações de cada parte, detalhando além de seu nome de ação, transmissor e receptor, uma breve descrição do que se espera que a ação deva realizar:

Tabela 5 - Ações do caso de uso Mercado Livre

Ações para Cliente	
(A_IniciarPedido,Cliente, Vendedor)	O cliente acha um produto de seu interesse e inicia a compra com o vendedor, esta ação deve ser imediata.
(A_DadosPagamento, Cliente, MercadoPago)	O cliente recebe um formulário com instruções para o pagamento e as formas de pagamento. Envia seus dados de pagamento.
(A_RecebiProduto, Cliente, MercadoPago)	O cliente avisa que recebeu o produto e que está tudo certo.
Ações para Mercado Pago	
(A_ValidarPagamento, MercadoPago, OperadoraCartao)	O intermediador irá repassar os dados do cartão a Operadora de Cartão para que o pagamento seja validado.
(A_PagamentoRecebido, MercadoPago, Vendedor)	O intermediador irá avisar ao Vendedor que já está com o pagamento do Cliente.
(A_PagarVendedor, MercadoPago, Vendedor)	O intermediador irá repassar o pagamento ao Vendedor.
Ações para Operadora de Cartão	
(A_TransaçãoEfetuada, OperadoraCartao, MercadoPago)	A empresa irá avisar ao Mercado Pago que deu tudo certo com a transação do cartão
Ações para Transportadora	
(A_EntregarProduto, Transportadora, Cliente)	A transportadora terá 8 dias para efetuar a entrega da mercadoria ao cliente.
Ações para Vendedor	
(A_FormularioPagamento, Vendedor, Cliente)	O vendedor envia ao cliente as opções de pagamento.
(A_EnviarProduto, Vendedor, Transportadora)	O vendedor envia o produto ao cliente.

Por fim, podemos escrever os compromissos traçados dentro do contrato, sendo assim, a tabela 6 traz o nome de cada compromisso e uma breve descrição do que se espera do mesmo:

Tabela 6 - Compromissos do caso de uso Mercado Livre

Compromisso	Descrição
C_Compra	Compromisso entre o comprador e o vendedor, garantia de que cliente irá comprar algo e receber por aquilo.
C_Pagamento	Compromisso entre o comprador e o Mercado Pago, garantia de que o pagamento irá ocorrer.
C_Validacao	Compromisso entre o Mercado Pago e a Operadora de cartão, garantia de que os dados passados pelo cliente serão válidos.
C_Intermediador	Compromisso entre o Mercado Pago e o cliente e vendedor, garantia de que ambas as partes receberá o que foi combinado ou tudo será devolvido.
C_Entrega	Compromisso entre o vendedor e a transportadora, garantia de que um produto será entregue ao seu destino final.

3.2 FORMALIZAR

Para formalizar as partes são necessários uma descrição e um ID para cada parte envolvida, como pode-se observar na tabela 7:

Tabela 7 - Formalização das partes do caso de uso Mercado Livre

Partes	
Nome	**ID**
Cliente	Cliente
Mercado Pago	MercadoPago
Operadora de Cartão	OperadoraCartao
Transportadora	Transportadora
Vendedor	Vendedor

Com as ações descritas é possível formalizar as ações:

Tabela 8 - Formalização das ações do caso de uso Mercado Livre

Ações			
Nome	**Transmissor**	**Receptor**	**Tempo em minutos**
A_IniciarPedido	Cliente	Vendedor	0
A_DadosPagamento	Cliente	MercadoPago	10
A_RecebiProduto	Cliente	MercadoPago	1440
A_ValidarPagamento	MercadoPago	OperadoraCartao	60
A_PagamentoRecebido	MercadoPago	Vendedor	30
A_PagarVendedor	MercadoPago	Vendedor	180
A_TransaçãoEfetuada	OperadoraCartao	MercadoPago	30
A_EntregarProduto	Transportadora	Cliente	11520
A_FormularioPagamento	Vendedor	Cliente	1
A_EnviarProduto	Vendedor	Transportadora	1440

Por fim é possível classificar as ações e montar o Grafo de Compromissos como é possível verificar na tabela 9:

Tabela 9 - Formalização dos compromissos e classificação das ações do caso de uso Mercado Livre

Compromissos	Classificação das Ações nos Compromissos		
	Gatilho(Trigger)	Implicar(Involve)	Finalizar(Finish)
C_Compra	A_IniciarPedido		
		A_FormularioPagamento	
		A_DadosPagamento	
			A_RecebiProduto

C_Pagamento	A_FormularioPagamento		
		A_DadosPagamento	
			A_TransaçãoEfetuada
C_Validacao	A_ValidarPagamento		
			A_TransaçãoEfetuada
C_Intermediador	A_DadosPagamento		
		A_ValidarPagamento	
		A_TransaçãoEfetuada	
			A_PagamentoRecebido
	A_RecebiProduto		
			A_PagarVendedor
C_Intermediador	A_PagamentoRecebido		
		A_EnviarProduto	
		A_EntregarProduto	
			A_RecebiProduto

Tendo toda essa separação em mãos o contrato já pode ser passado para o modelo computacional formal proposto, para isso apresenta-se um diagrama de classes baseado na linguagem de Xu(2003, 2009). Neste diagrama é possível visualizar todo o conceito utilizado por Xu para formalização do contrato.

Esta representação é uma leitura do trabalho de Xu, apesar de ser sua representação formal ainda é uma representação matemática e genérica, para que seja possível a leitura computacional, a mesma será transformada em linguagem XML, assim será possível armazenar seu conteúdo, ler, e ter as mesmas propriedades matemáticas.

O resultado expresso em XML foi dividido em 3 seções:

Na primeira parte, figura 6, é relacionado o conjunto das partes (*parts*), proposto por Xu(2003), com seu resultado em xml, cada parte participante do contrato estará presente no XML dentro de uma tag denominada "*part*", que possuirá dois atributos o nome (*name*) e o ID, todas as partes serão filhas da tag "*parts"*.

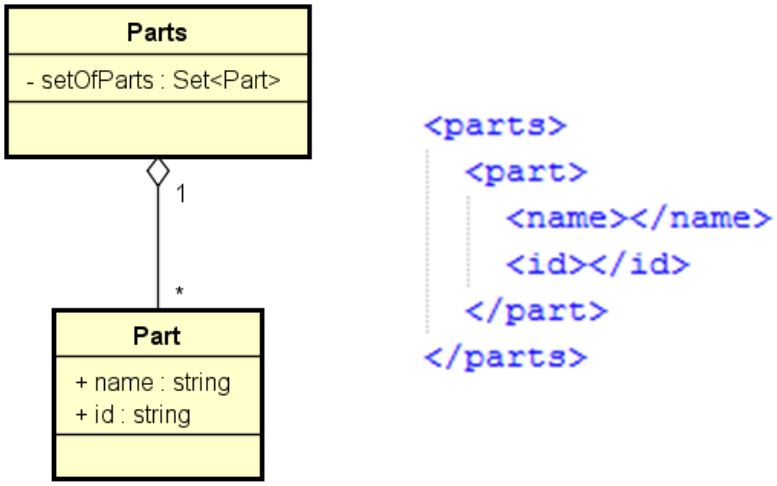

Figura 6 - XML parte 1 - Parts

Na segunda parte do XML, figura 7**,** é relacionado o conjunto das ações (*actions*) ao XML correspondente, cada ação disponíveis no contrato será representada por uma tag "*action*" que possuirá quatro atributos, o nome (*name*), o transmissor (*sender*), o receptor (*receiver*) e o tempo da ação (*time*). Todas as tags que representam cada ação serão filhas de uma tag "*actions*".

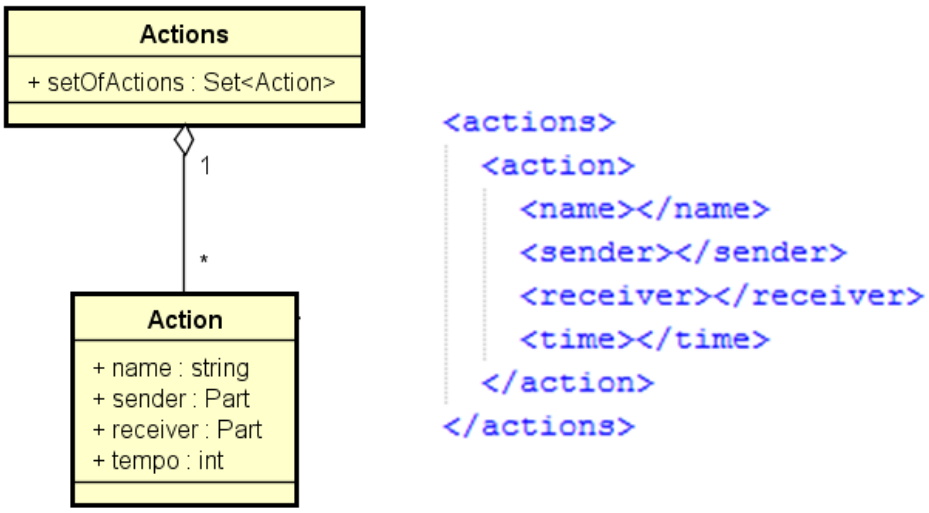

Figura 7 - XML parte 2 - actions

Na terceira parte do XML, figura 8, é relacionado os compromissos com o XML correspondente, este diagrama é o que apresenta maior complexidade devido ao maior número de relacionamentos. Um conjunto de compromissos é representado por uma tag “*commitments*”, e cada compromisso aí presente por uma tag “*commitment*” que possui quatro subtags, o nome do compromisso “*name*”, o transmissor e o receptor “*sender* e *receiver*”, e a das ações “*actions*” esta tag se diferencia por ser um novo conjunto que possui cada ação “*action*” com seus respectivos nomes “*name*” e atributos “*attribute*”.

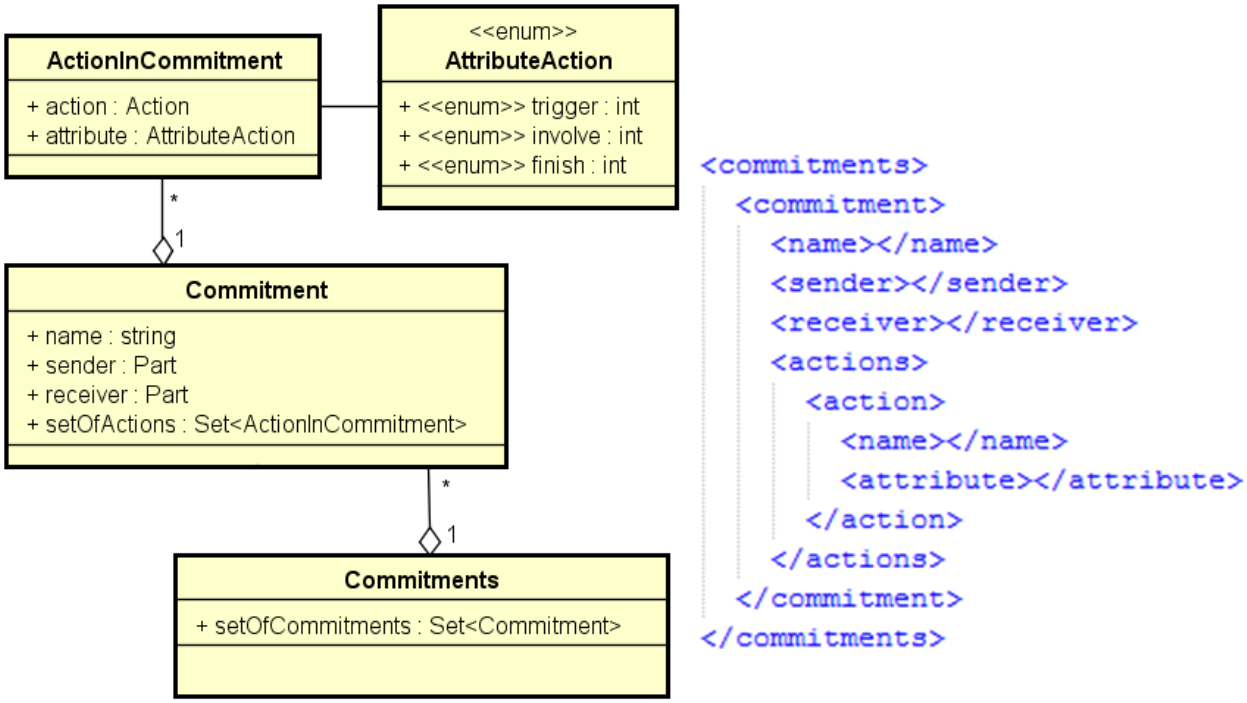

Figura 8 - XML parte 3 - commitments

Na quarta e última parte do XML, figura 11, é relacionado a ordem em que os compromissos devem ser chamados com o XML correspondente, é apresentado então uma tag “*order_commitment*” que agrupa todos os relacionamentos, cada relacionamento é representado pela tag “*relationship*” que possui apenas duas tag filhas, o compromisso que está sendo executado e o próximo que irá executar, “*name_commitment_master*” e “*name_commitment_slave*” respectivamente.

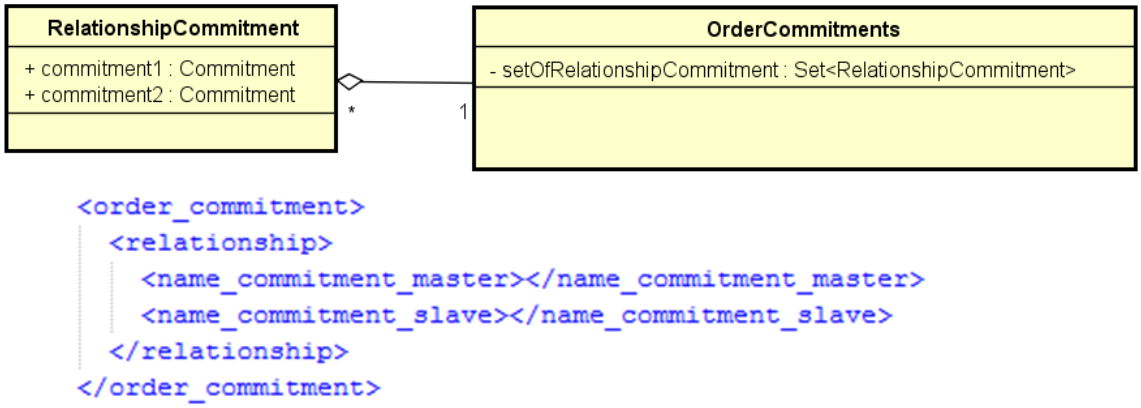

Figura 9 - XML parte 4 - order commitment

Nos seguintes apêndices podem ser encontrados:

- Apêndice A: diagrama de classes completo baseado na linguagem matemática de XU (2003);

- Apêndice B: modelo completo do XML proposto com base no diagrama de classes;

- Apêndice C: XML para o estudo de caso do Mercado Livre proposto.

Esta parte da formalização dos contratos representa uma pequena contribuição no desenvolvimento do trabalho de Xu (2003, 2009), e isto é expressado na figura 10.

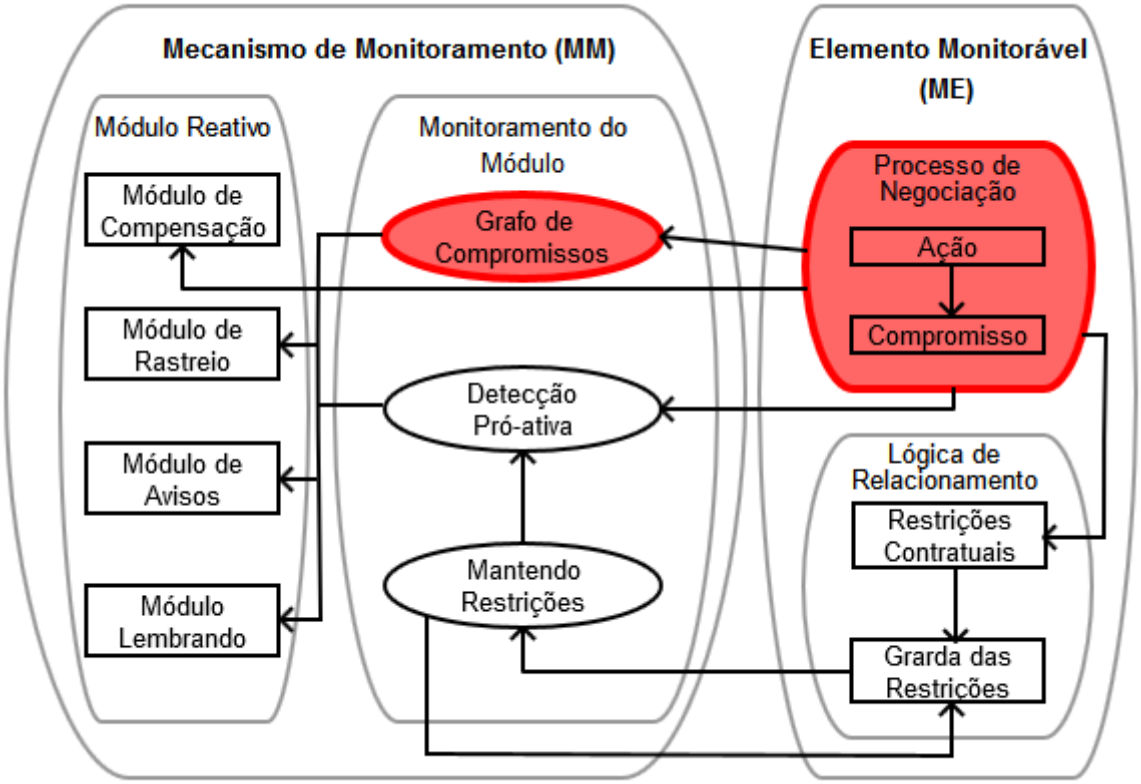

Figura 10 - Modelo traduzido e adaptado de Xu

3.3 FERRAMENTA PARA GUARDAR O CONTRATO FORMAL E DISPONIBILIZÁ-LO

Foi desenvolvido uma ferramenta capaz de guardar este XML, e a partir da inserção do XML a ferramenta devolve um link de acesso para o contrato, mostrando sua formalização. Este aplicativo foi feito utilizando o framework Ruby on Rails e está hospedada no link https://ide.c9.io/mayconn/trampo-facu2, seu código fonte foi postado no Github https://github.com/MayconnW/trabalho-faculdade.git.

A figura 11 representa a página administrativa inicial do sistema, onde é possível ver os contratos armazenados que encontram-se com o status ativo, também é possível ver a chave de acesso de cada contrato, cada linha possui três links, uma para demonstrar o contrato, um para edição do mesmo e um para excluí-lo, este último na verdade coloca o status do contrato como inativo, para preservação dos dados.

Esta chave deve ser usada para que as partes possam acessar seus contratos e ver como que está seu andamento, para isso deverá ser acessado o link

http://trampo-facu2-mayconn.c9.io/xml/key/CHAVEDEACESSO, substituindo o “CHAVEDEACESSO” por sua chave.

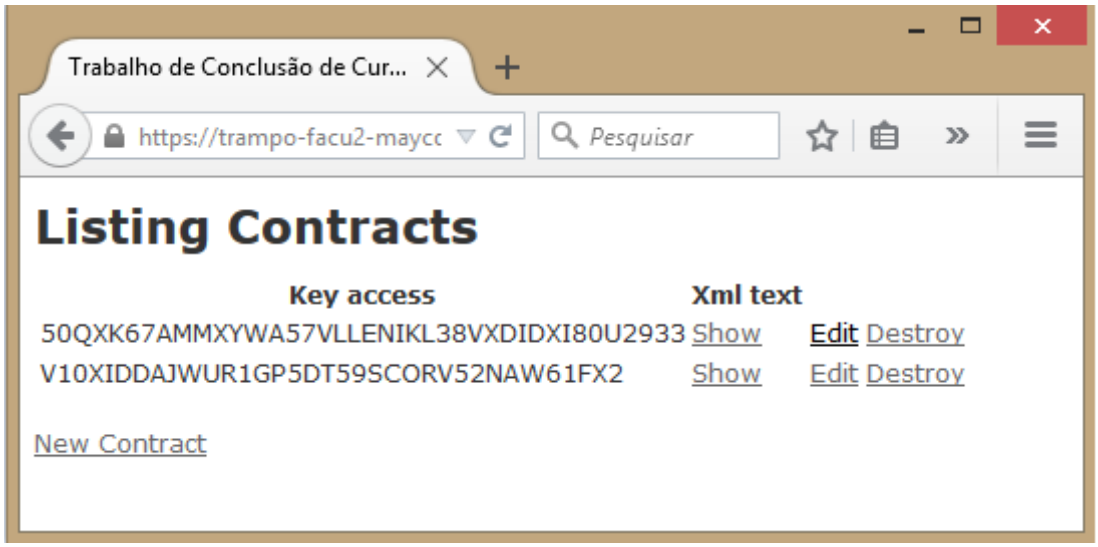

Figura 11 - Aplicativo rodando - Página inicial

Na figura 12 é apresentado a inserção de um novo contrato, o campo “key access" é gerado automaticamente pelo sistema e deve ser guardado pelas partes pois sem ele não será possível acessar o contrato, no próximo campo deverá ser colocado o XML do contrato formalizado. Ao confirmar a inserção do contrato o mesmo já estará pronto para rodar.

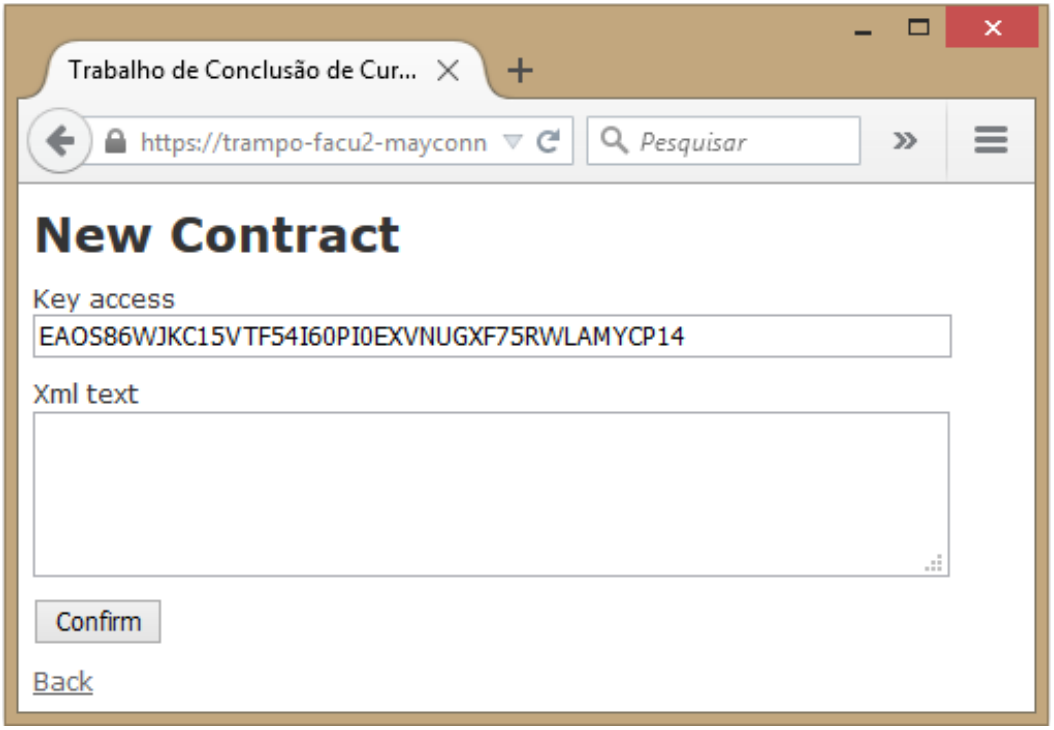

Figura 12 - Aplicativo rodando - Inserindo um contrato

Finalmente o próximo passo é a visualização do todo do contrato, figura 13, onde é possível ver todas as partes, ações e compromissos. Vale destacar aqui que a página possui a chave de acesso, o próprio link de acesso, e a tabela mais importante que é a classificação das ações nos compromissos.

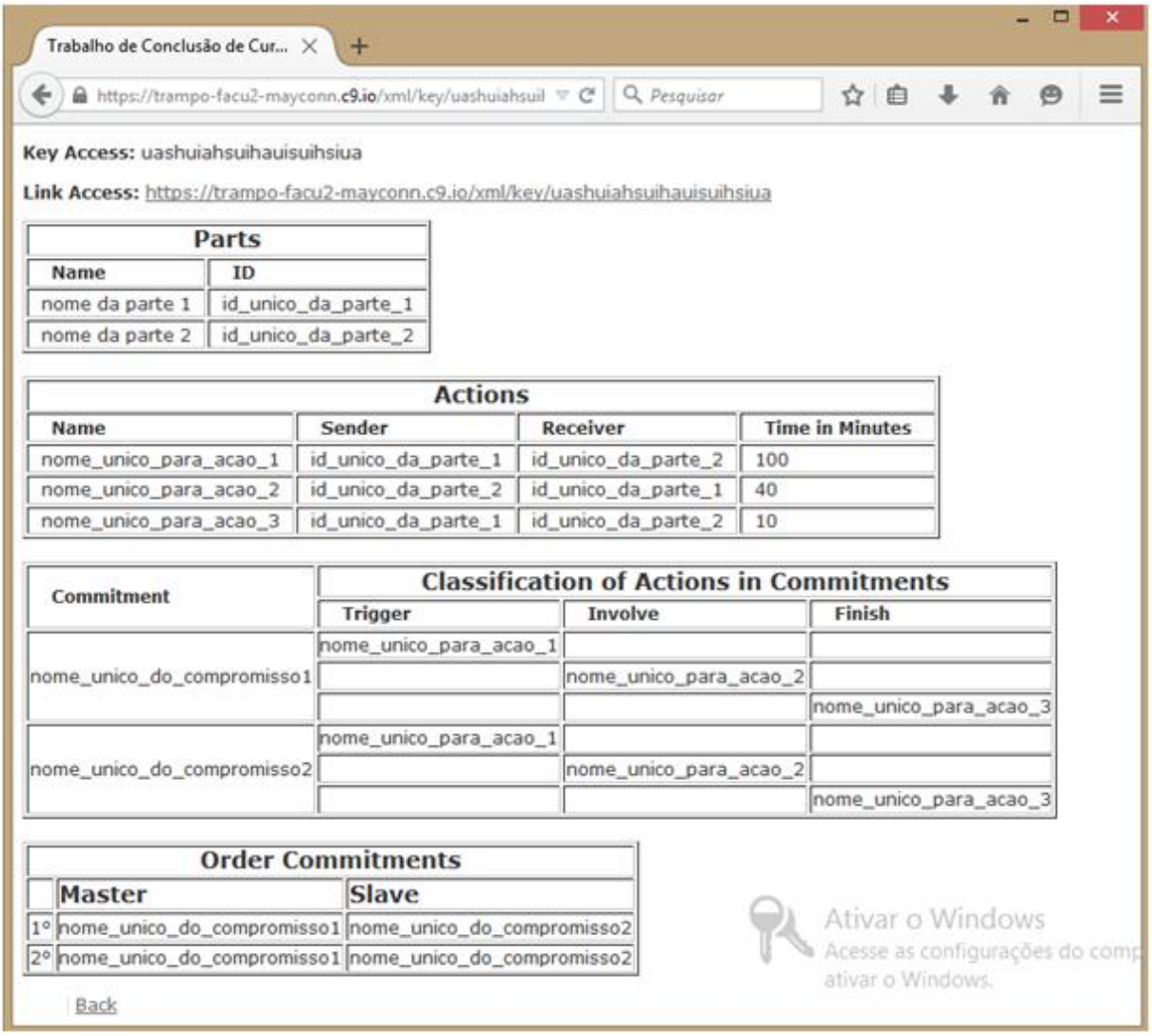

Figura 13 - Aplicativo rodando - Exibindo os dados do contrato

A partir deste momento, onde o contrato já está armazenado e rodando as partes podem executar ações, para isso, aproveitando a tecnologia Rest que foi aplicada, basta ao fim do link de acesso adicionar "/IdDaParte/IdDaAção". Por exemplo para a "Parte 1" executar a "Ação 1" o seguinte link deverá ser acessado http://~./xml/key/CHAVEDEACESSO/id_unico_da_parte_1/nome_unico_para_acao_

1, com isso o sistema irá verificar se a ação pode ser executada e se esta ação pertence a parte que tentou executar a ação. A seção 3.4 irá demonstrar como é feito este processo de monitoramento.

Uma pequena ilustração do Web Service com os contratos rodando e conectado aos sistemas que representam as partes pode ser visto na figura 14.

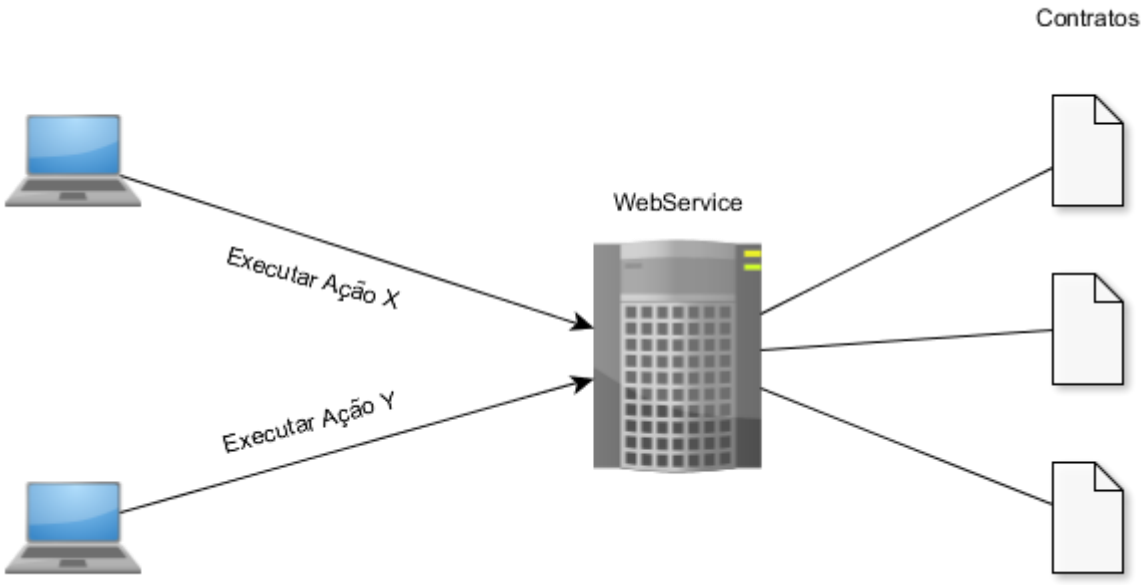

Figura 14 - WebService com contratos rodando

3.4 PSEUDOCÓDIGO DO MONITORAMENTO REATIVO

O algoritmo “*trace route*” responsável por apontar o culpado da quebra de um contrato, figura 15, trabalha da seguinte forma:

O Algoritmo ficará rodando enquanto o conjunto das guardas, que é onde fica armazenado os compromissos, possuir algum compromisso dentro dele. Na segunda linha uma variável “compromisso_da_vez” irá armazenar o compromisso da posição 0, ou seja o compromisso que precisa rodar neste momento, feito isso o “conjunto_das_guardas” já descarta este compromisso de sua fila.

Em seguida será percorrido todas as ações que estão dentro do “compromisso_da_vez” para que possa ser pego o tempo máximo que este compromisso deve finalizar, a variável que será responsável por guardar este valor será a “tempo_maximo_do_compromisso”.

Ao receber uma ação o algoritmo irá verificar se a “ação_executada” pertence ao conjunto de ações do “compromisso_da_vez”, se a ação não for deste compromisso a ação será descartada, caso contrário novas verificações serão feitas. O algoritmo irá verificar se o compromisso está iniciado, caso não esteja o algoritmo só irá permitir que ações com o atributo “trigger” sejam executados, qualquer outra ação será descartada. Se o compromisso já está iniciado o algoritmo irá verificar se o tempo máximo de execução do mesmo foi estourado ou não, e caso positivo ele irá percorrer as ações deste compromisso, comparando o tempo de execução da ação com o tempo limite dela, para que possa achar o culpado pela quebra, e caso o tempo máximo do compromisso ainda for maior que o tempo de execução do mesmo, ele fará uma última verificação se a “ação executada” é do tipo “finish” para que assim possa fechar o compromisso.

Vale ressaltar que o tempo máximo do compromisso sempre é verificado antes que o tempo máximo da ação, pois uma ação pode demorar mais que o esperado para ela e mesmo assim não haver quebra no contrato, desde que as outras ações consigam trabalhar no tempo que lhes restam.

```
while (conjunto_das_guardas > 0){
    compromisso_da_vez = conjunto_das_guardas[0];
    conjunto_das_guardas.dropItem(0);

    foreach(acao in compromisso_da_vez.acoes){
        tempo_maximo_do_compromisso += acao.tempo;
    }

    if (acao_executada in compromisso_da_vez.acoes){
        if(!compromisso_da_vez.iniciado){
            if (acao_executada.attributo == trigger){
                iniciaCompromisso(compromisso_da_vez.iniciado);
            }
            else{
                rejeitaAcao(acao_executada);
            }
        }else{
            if(tempo_decorrido > tempo_maximo_do_compromisso){
                foreach(acao in compromisso_da_vez.acoes){
                    if (acao.tempoDecorrido > acao.tempoMaximo > ){
                        culpado = acao;
                        break;
                    }
                }
                quebraDeContrato(acao);
            }else if(acao_executada.atributo == finish){
                finalizaCompromisso(compromisso_da_vez);
            }
        }
    }else{
        rejeitaAcao(acao_executada);
    }

}
```

Figura 15 - Pseudocódigo do trace route

A figura 16 representa a diagramação deste algoritmo reativo do *trace route*, deixando mais claro o entendimento do algoritmo e a explicação do mesmo.

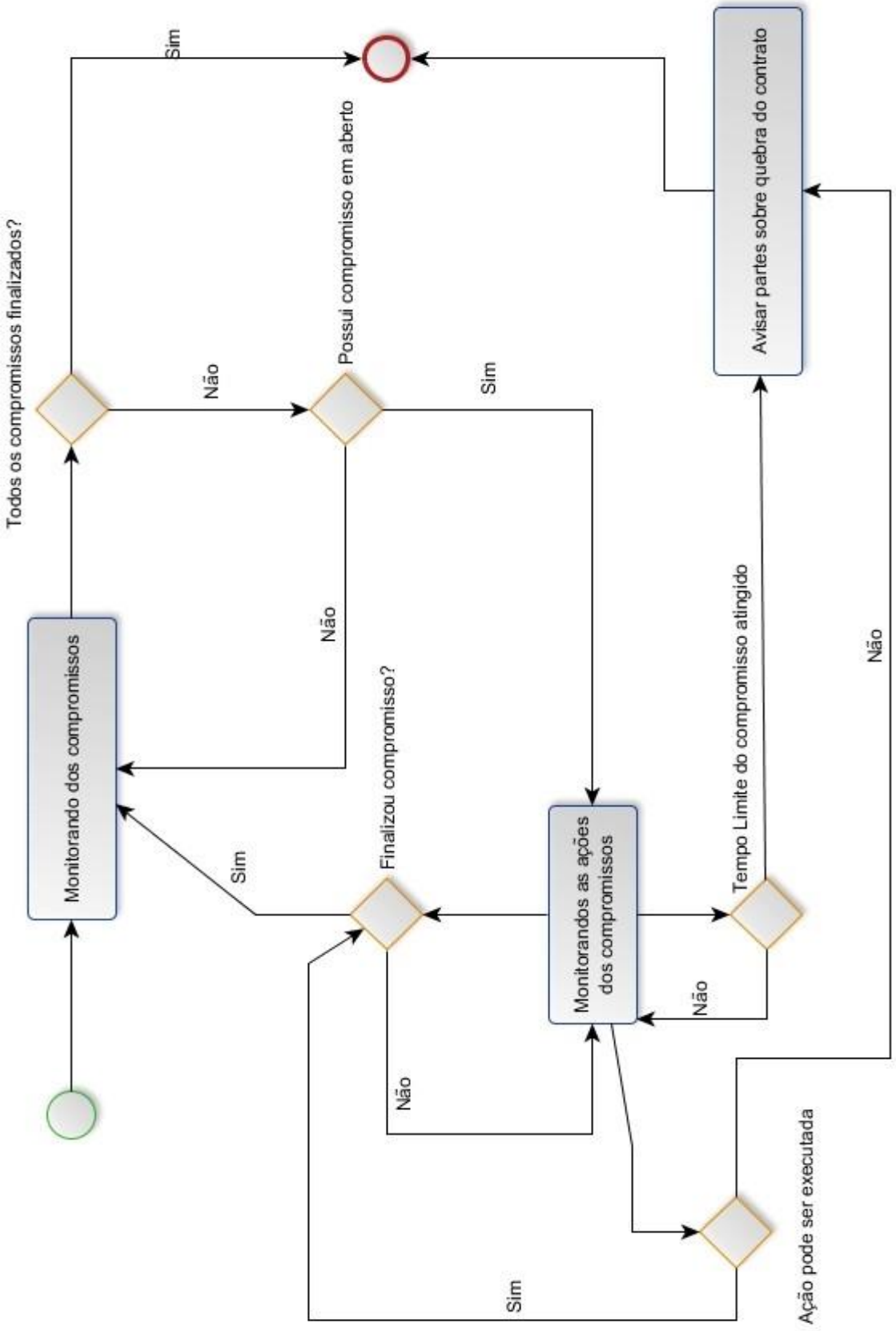

Figura 16 - Fluxo de trabalho do algoritmo reativo baseado na linguagem de Xu (2003)

3.5 FLUXO DE TRABALHO DE UM CONTRATO

O fluxo de trabalho indica como é o comportamento do sistema de monitoramento, desde a chegada do contrato ao servidor, até seu término. O sistema irá pegar o contrato formalizado e no formato XML e distribuirá as ações de cada parte, irá separar os compromissos e iniciar o monitoramento.

Na figura 17 é possível observar a comunicação entre os sistemas e o WebService, neste exemplo existem três partes envolvidas, o Sistema 1 ficou responsável por enviar o contrato ao *WebService.* Na sequência o Sistema 2 tenta executar uma ação que não é aceita no momento, então sua ação é rejeitada. O Sistema 1 executa uma ação esperada, a de trigger, e inicia o compromisso. Em seguida o tempo máximo do compromisso é atingido, o *WebService* então envia uma mensagem para as partes envolvidas sobre a quebra do contrato e o responsável pela mesma, para encontrar o responsável faz se uso do algoritmo apresentado na seção anterior.

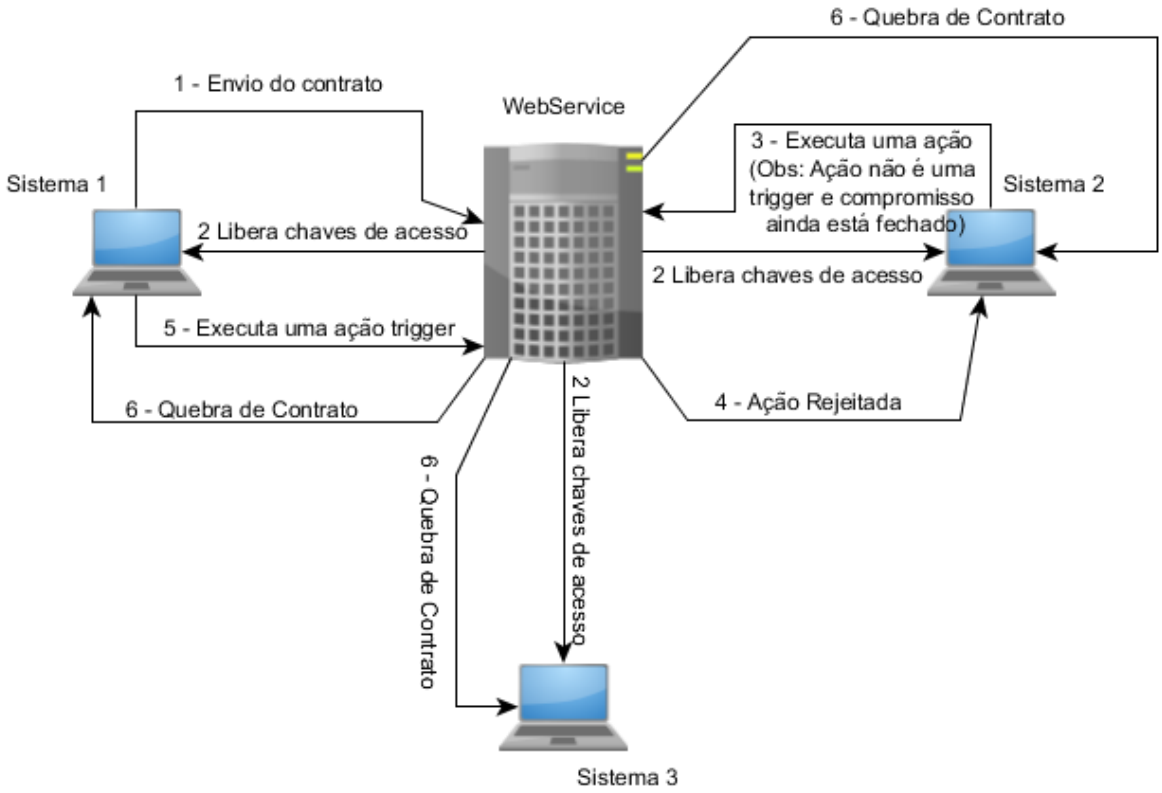

Figura 17 - Comunicação entre os sistemas e quebra de contrato

Por fim na figura 18 é possível ver o fluxo de trabalho deste monitoramento.

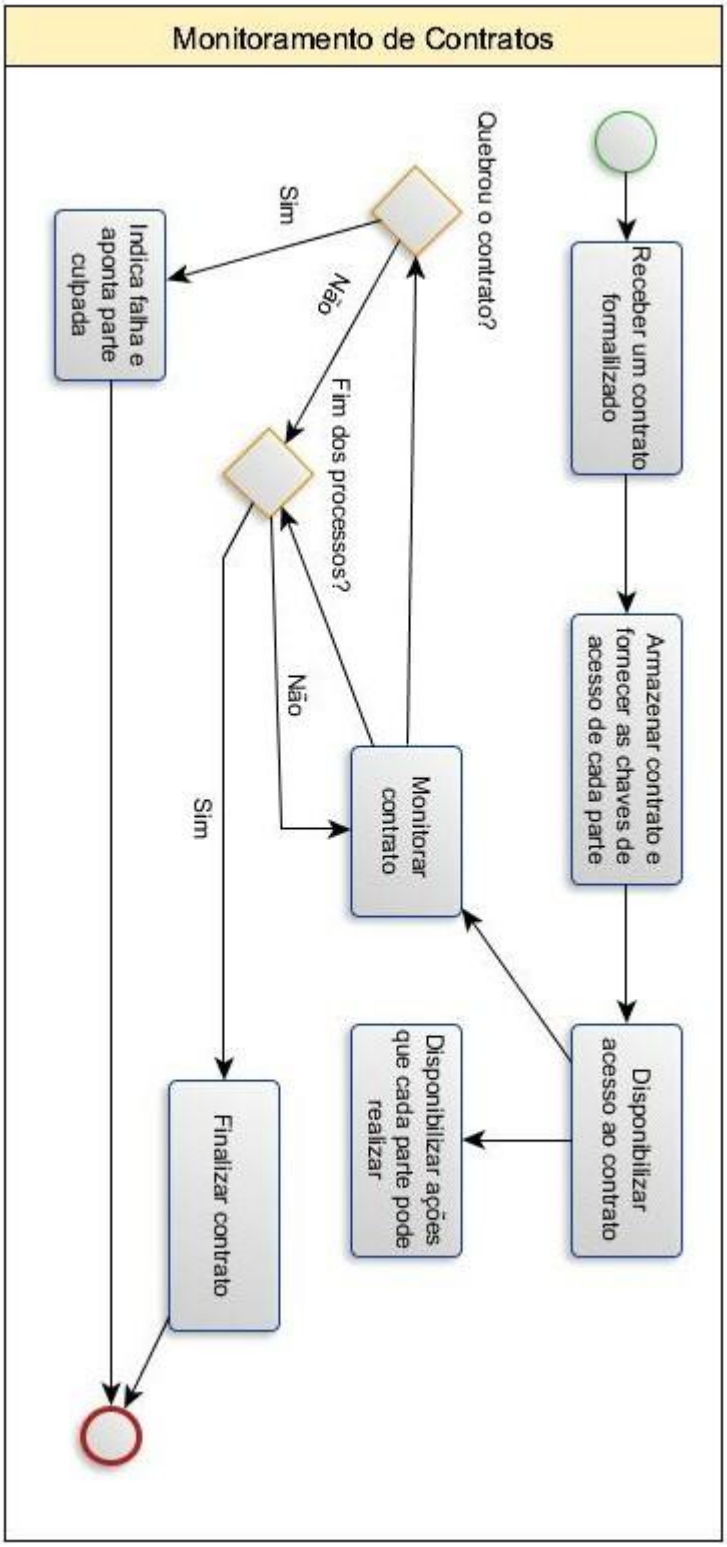

Figura 18 - Fluxo de trabalho do monitoramento de um contrato

4 RESULTADOS OBTIDOS

Este capítulo explora os testes realizados e os resultados obtidos.

O primeiro resultado obtido foi a formalização de um contrato, em seguida expressá-lo em XML. Esta contribuição serve como base para um monitoramento de contratos. Este XML é o resultado da formalização dos contratos proposto por Xu(2003).

O algoritmo de trace route foi demonstrado, e explicado linha a linha, esta base serve para que possa ser desenvolvido uma aplicação real.

5 CONCLUSÕES

Este trabalho propôs a formalização de contratos e parte de um monitoramento baseado em algoritmos matemáticos do trabalho de Xu(2003, 2009). Como resultado obtivemos um contrato formalizado seguindo como base o modelo proposto por Xu, e em uma linguagem aceita computacionalmente. E a tradução em diagrama e pseudocódigo de sua linguagem matemática para rastreio de culpados em uma quebra de contrato.

Uma dificuldade encontrada foi que em seu trabalho Xu, utiliza apenas conceitos matemáticos e nem todos seus conceitos funcionam da mesma forma quando passado para um algoritmo de computador.

Um problema encontrado com essa implementação foi que em seu algoritmo Xu prevê que dois compromissos podem estar funcionando em um mesmo momento, porém não resolve este problema, assim a implementação, apesar de trabalhar com contratos multilaterais, trata um compromisso por vez, o algoritmo até abre dois compromissos, porém ele só volta ao compromisso anterior quando termina o compromisso em questão.

Para trabalhos futuros podemos ter o desenvolvimento do módulo pró-ativo, e o aperfeiçoamento do módulo reativo para que trabalhe com vários compromissos simultaneamente.

Para facilitar na figura 19 pode-se observar nas cores vermelhas o que foi desenvolvido, e nas cores azuis o que pode vir a ser desenvolvido em trabalhos futuros.

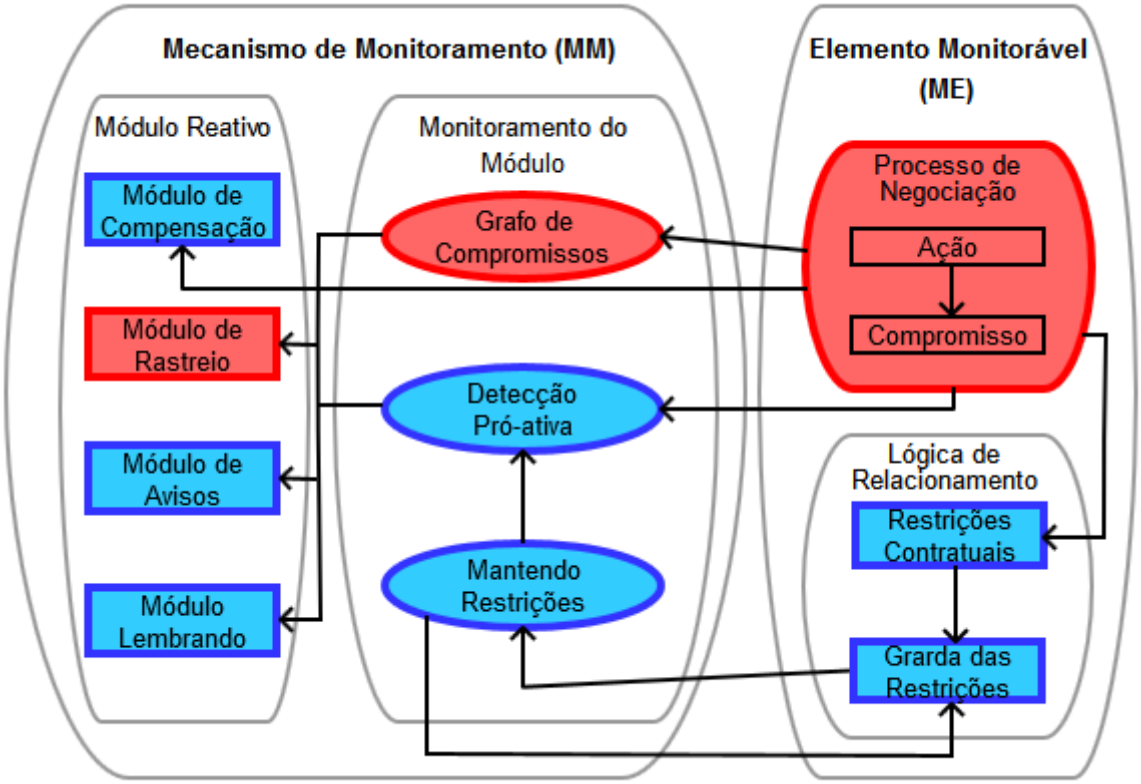

Figura 19 - Modelo traduzido e adaptado de Xu

REFERÊNCIAS

,; ALMEIDA, M. B. Uma introdução ao XML, sua utilização na Internet e alguns conceitos complementares. 2002. 13f. Dissertação (Mestrado em Ciência da Informação) - Universidade Federal de Minas Gerais, Brasil, 2002.

ALMEIDA, V. I. Contratos Eletrônicos. 2004. 124f. Monografia (Bacharelado em Direito) - Centro Universitário das Faculdades Metropolitanas Unidas - UniFMU, São Paulo, 2004. Disponível em: <http://arquivo.fmu.br/prodisc/direito/vida.pdf> Acesso em: 22 out. 2014

BRASIL. Código Civil, de 12 de setembro de 1988. Artigo X. CONTRATOS, Código Civil, n. 31, p. 31, Dezembro de 1987.

COLOURIS, G.; DOLLIMORE, J.; KINDBERG, T. Sistemas Distribuídos Conceitos e Projeto. 4. ed. Brasil: ARTMED Editora S.A., 2007. 790p.

DALL'OGLIO, P. PHP Programando com Orientação a Objetos. 2. ed. Brasil: Novatec, 2009. 574p.

FERNANDEZ, Obie. The Rails Way. Editora Addison Wesley. Dallas,2008.

GIL, A.C. (2002) Como elaborar projetos de pesquisa. 4ª. ed. São Paulo: Atlas S/A.

GitHub, Ferramenta. Disponível em: < https://github.com/about/> Acesso em: 15 jul. 2015

KYAS, M.; PRISACARIU, C.; SCHNEIDER, G. Runtime monitoring of electronic contracts. In 6th International Symposium on Automated Technology for Verification and Analysis (ATVA'08), Springer-Verlag, n.5311, p. 397-407, 2008.

LENZ, Patrick. Build owner Ruby on Rails web application.Editora Sitepoint. 2007.

Manual do php. Disponível em: <http://php.net/manual/pt_BR/> Acesso em: 19 nov. 2014

RT, E. Código de Processo Civil. 13. ed. Brasil: RT, 2008. 832p.

SANTOS, L. L. Monitoramento de Contratos Eletrônicos Baseados em Características. 2011. 113f. Dissertação (Mestrado em Sistemas de Informação) - Universidade Estadual de Campinas, Campinas, 2011. Disponível em: <http://www.bibliotecadigital.unicamp.br/document/?down=000798017> Acesso em: 16 março. 2015

SAUDETE, A. SOA aplicado - Integrando com web services e além. 1. ed. Brasil: Casa do Código, 2012. 273p.

SEIXAS, R. TEORIA GERAL DOS CONTRATOS. Volume 1. ed. Desconhecido: Desconhecido, 1997. 141p. Disponível em: <http://renatoseixas.files.wordpress.com/2009/06/teoria-geral-dos-contratos-e28093-v-11.pdf> Acesso em: 1 nov. 2014

XU, L. MONITORING MULTI-PARTY CONTRACTS FOR E-BUSINESS. Alemanha: VDM Verlag Dr. Müller, 2009. 168p.

XU, L. A concept for monitoring of electronic contracts. 2003. 19f. Monografia (Bacharelado em Informática) - Tilburg University - CRISM/Infolab, Holanda, 2003.

w3schools. Disponível em: <www.w3schools.com> Acesso em: 20 nov. 2014

APÊNDICE A - DIAGRAMA DE CLASSES COMPLETO BASEADO NA LINGUAGEM MATEMÁTICA DE XU (2003);

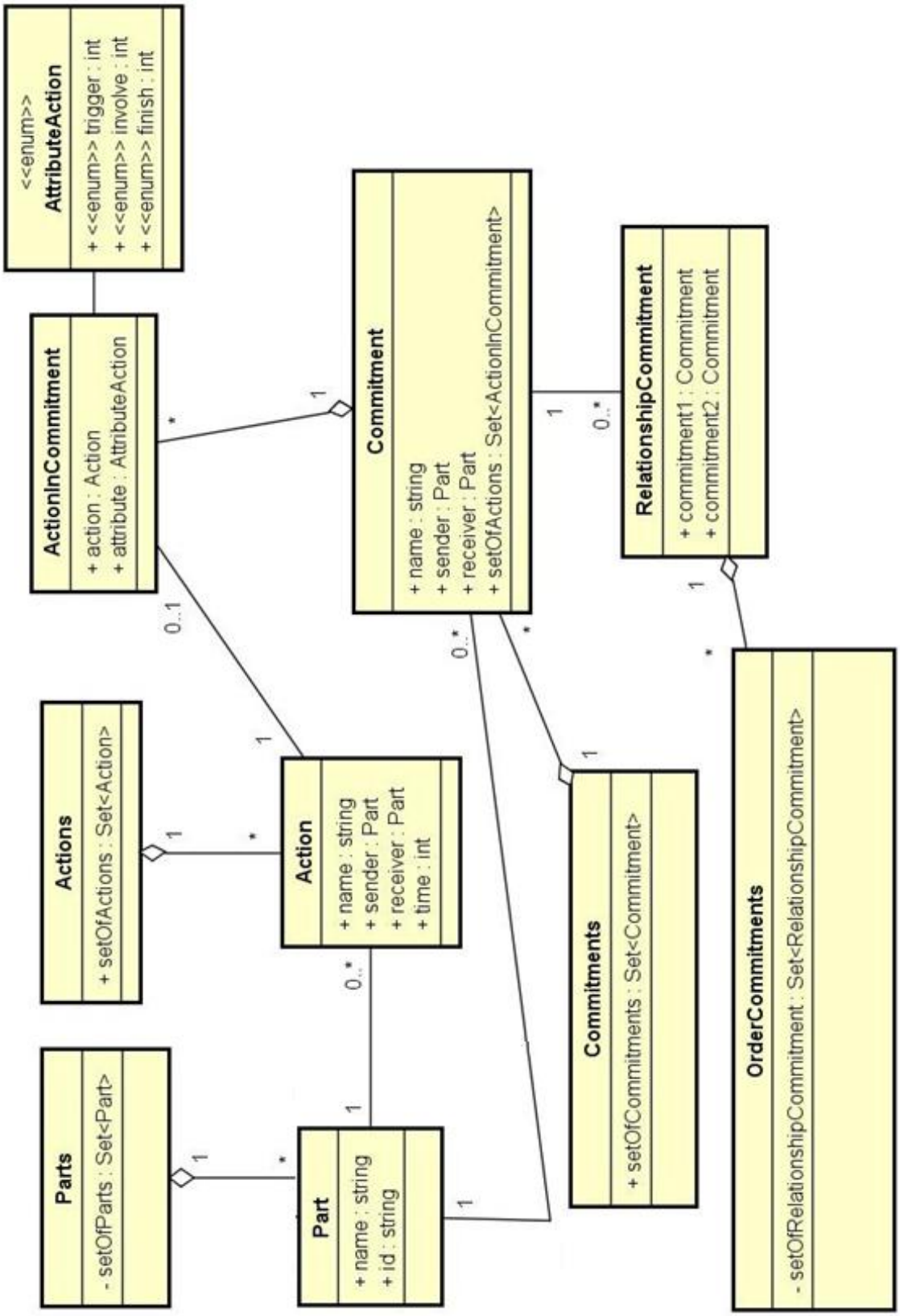

Figura 20 - Diagrama de classes baseado na linguagem matemática de Xu(2003)

APÊNDICE B - MODELO COMPLETO DO XML PROPOSTO COM BASE NO DIAGRAMA DE CLASSES

```
<xml>
    <parts>
        <part>
            <name>   </name>
            <id>     </id>
        </part>
    </parts>
    <actions>
        <action>
            <name>        </name>
            <sender>      </sender>
            <receiver>    </receiver>
            <time>        </time>
        </action>
    </actions>
    <commitments>
        <commitment>
            <name>        </name>
            <sender>      </sender>
            <receiver>    </receiver>
            <actions>
                <action>
                    <name>         </name>
                    <attribute>    </attribute>
                </action>
            </actions>
        </commitment>
    </commitments>
    <order_commitment>
        <relationship>
            <name_commitment_master>   </name_commitment_master>
            <name_commitment_slave>    </name_commitment_slave>
        </relationship>
    </order_commitment>
</xml>
```

Figura 21 - Modelo XML proposto sem preenchimento

APÊNDICE C – XML PARA O ESTUDO DE CASO DO MERCADO LIVRE

```
<xml>
 <parts>
  <part>
   <name>Cliente</name>
   <id>Cliente</id>
  </part>
  <part>
   <name>Mercado Pago</name>
   <id>MercadoPago</id>
  </part>
  <part>
   <name>Operadora de Cartão</name>
   <id>OperadoraCartao</id>
  </part>
  <part>
   <name>Transportadora</name>
   <id>Transportadora</id>
  </part>
  <part>
   <name>Vendedor</name>
   <id>Vendedor</id>
  </part>
 </parts>

 <actions>
  <action>
   <name>A_IniciarPedido</name>
   <sender>Cliente</sender>
   <receiver>Vendedor</receiver>
   <time>0</time>
  </action>
  <action>
   <name>A_DadosPagamento</name>
```

```
  <sender>Cliente</sender>
  <receiver>MercadoPago</receiver>
  <time>10</time>
</action>
<action>
  <name>A_RecebiProduto</name>
  <sender>Cliente</sender>
  <receiver>MercadoPago</receiver>
  <time>1440</time>
</action>
<action>
  <name>A_ValidarPagamento</name>
  <sender>MercadoPago</sender>
  <receiver>OperadoraCartao</receiver>
  <time>60</time>
</action>
<action>
  <name>A_PagamentoRecebido</name>
  <sender>MercadoPago</sender>
  <receiver>Vendedor</receiver>
  <time>30</time>
</action>
<action>
  <name>A_PagarVendedor</name>
  <sender>MercadoPago</sender>
  <receiver>Vendedor</receiver>
  <time>180</time>
</action>
<action>
  <name>A_TransaçãoEfetuada</name>
  <sender>OperadoraCartao</sender>
  <receiver>MercadoPago</receiver>
  <time>30</time>
</action>
```

```
  <action>
    <name>A_EntregarProduto</name>
    <sender>Transportadora</sender>
    <receiver>Cliente</receiver>
    <time>11520</time>
  </action>
  <action>
    <name>A_FormularioPagamento</name>
    <sender>Vendedor</sender>
    <receiver>Cliente</receiver>
    <time>1</time>
  </action>
  <action>
    <name>A_EnviarProduto</name>
    <sender>Vendedor</sender>
    <receiver>Transportadora</receiver>
    <time>1440</time>
  </action>
</actions>

<commitments>
  <commitment>
    <name>C_Compra</name>
    <sender>Cliente</sender>
    <receiver>Vendedor</receiver>
    <actions>
      <action>
        <name>A_IniciarPedido</name>
        <attribute>trigger</attribute>
      </action>
      <action>
        <name>A_FormularioPagamento</name>
        <attribute>involve</attribute>
      </action>
```

```
    <action>
      <name>A_DadosPagamento</name>
      <attribute>involve</attribute>
    </action>
    <action>
      <name>A_RecebiProduto</name>
      <attribute>finish</attribute>
    </action>
  </actions>
</commitment>
<commitment>
  <name>C_Pagamento</name>
  <sender>Vendedor</sender>
  <receiver>Cliente</receiver>
  <actions>
    <action>
      <name>A_FormularioPagamento</name>
      <attribute>trigger</attribute>
    </action>
    <action>
      <name>A_DadosPagamento</name>
      <attribute>involve</attribute>
    </action>
    <action>
      <name>A_TransaçãoEfetuada</name>
      <attribute>finish</attribute>
    </action>
  </actions>
</commitment>
<commitment>
  <name>C_Validacao</name>
  <sender>MercadoPago</sender>
  <receiver>OperadoraCartao</receiver>
  <actions>
```

```
    <action>
      <name>A_ValidarPagamento</name>
      <attribute>trigger</attribute>
    </action>
    <action>
      <name>A_TransaçãoEfetuada</name>
      <attribute>finish</attribute>
    </action>
  </actions>
</commitment>
<commitment>
  <name>C_Intermediador</name>
  <sender>Cliente</sender>
  <receiver>MercadoPago</receiver>
  <actions>
    <action>
      <name>A_DadosPagamento</name>
      <attribute>trigger</attribute>
    </action>
    <action>
      <name>A_ValidarPagamento</name>
      <attribute>involve</attribute>
    </action>
    <action>
      <name>A_TransaçãoEfetuada</name>
      <attribute>involve</attribute>
    </action>
    <action>
      <name>A_PagamentoRecebido</name>
      <attribute>finish</attribute>
    </action>
    <action>
      <name>A_RecebiProduto</name>
      <attribute>trigger</attribute>
```

```
      </action>
      <action>
        <name>A_PagarVendedor</name>
        <attribute>finish</attribute>
      </action>
    </actions>
  </commitment>
  <commitment>
    <name>C_Entrega</name>
    <sender>MercadoPago</sender>
    <receiver>Vendedor</receiver>
    <actions>
      <action>
        <name>A_PagamentoRecebido</name>
        <attribute>trigger</attribute>
      </action>
      <action>
        <name>A_EnviarProduto</name>
        <attribute>involve</attribute>
      </action>
      <action>
        <name>A_EntregarProduto</name>
        <attribute>involve</attribute>
      </action>
      <action>
        <name>A_RecebiProduto</name>
        <attribute>finish</attribute>
      </action>
    </actions>
  </commitment>
</commitments>

<order_commitment>
  <relationship>
```

```
    <name_commitment_master>C_Compra</name_commitment_master>
    <name_commitment_slave>C_Pagamento</name_commitment_slave>
   </relationship>
   <relationship>
    <name_commitment_master>C_Compra</name_commitment_master>
    <name_commitment_slave>C_Intermediador</name_commitment_slave>
   </relationship>
   <relationship>
    <name_commitment_master>C_Pagamento</name_commitment_master>
    <name_commitment_slave>C_Intermediador</name_commitment_slave>
   </relationship>
   <relationship>
    <name_commitment_master>C_Intermediador</name_commitment_master>
    <name_commitment_slave>C_Validacao</name_commitment_slave>
   </relationship>
   <relationship>
    <name_commitment_master>C_Validacao</name_commitment_master>
    <name_commitment_slave>C_Intermediador</name_commitment_slave>
   </relationship>
   <relationship>
    <name_commitment_master>C_Intermediador</name_commitment_master>
    <name_commitment_slave>C_Entrega</name_commitment_slave>
   </relationship>
  </order_commitment>

</xml>
```